MATHÉMATIQUES
à l'Université

Cours et exercices corrigés

Collection dirigée par
Charles-Michel Marle
Philippe Pilibossian

L'algèbre discrète de la transformée de Fourier

niveau M1

Gabriel Peyré

ellipses

MATHÉMATIQUES À L'UNIVERSITÉ

Collection dirigée par Charles-Michel MARLE et Philoppe PILIBOSSIAN

niveau M1

L'ALGÈBRE DISCRÈTE DE LA TRANSFORMÉE DE FOURIER

Gabrile PEYRÉ
Professeur agrégé

ellipses

Dans la même collection *Mathématiques à l'Université*

- *L'algèbre discrète de la transformée de Fourier*, G. Peyré, 336 pages, 2004.
- *Algèbre et théorie des nombres – cryptographie, primalité, vol. 1*, S. Al Fakir, 288 pages, 2003.
- *Algèbre linéaire*, F. Bories-Longuet, 160 pages, 2000.
- *Algèbre linéaire numérique – cours et exercices*, G. Allaire et S. M. Kaber, 240 pages, 2002.
- *Analyse complexe et distributions*, A. Yger, 400 pages, 2001.
- *Calcul différentiel*, G. Christol, A. Cot, Ch.-M. Marle, 224 pages, 1997.
- *Cours d'algèbre*, R. Elkik, 192 pages, 2002.
- *Cours de calcul formel – algorithmes fondamentaux*, Ph. Saux Picart, 192 pages, 1999.
- *Cours de calcul formel – corps finis, systèmes polynomiaux, applications*, Ph. Saux Picart et E. Rannou, 224 pages, 2002.
- *Distributions – espaces de Sobolev, applications*, M.-Th. Lacroix-Sonrier, 160 pages, 1999.
- *Éléments d'analyse convexe et variationnelle*, D. Azé, 240 pages, 1997.
- *Éléments d'intégration et d'analyse fonctionnelle*, A. El Kacimi Alaoui, 256 pages, 1999.
- *Équations aux dérivées partielles et leurs approximations*, B. Lucquin, 240 pages, 2004.
- *Géométrie différentielle avec 80 figures*, C. Doss-Bachelet, J.-P. Françoise et Cl. Piquet, 208 pages, 2000.
- *Les Groupes finis et leurs représentations*, G. Rauch, 192 pages, 2000.
- *Intégration et théorie de la mesure – une approche géométrique*, P. Krée, 240 pages, 1997.
- *Une introduction à la géométrie projective*, D. Lehmann, 128 pages, 2003.
- *Introduction à Scilab – exercices pratiques corrigés d'algèbre linéaire*, G. Allaire et S. M. Kaber, 240 pages, 2002.
- *Logique, ensemble, catégories – le point de vue constructif*, P. Ageron, 128 pages, 2000.
- *Méthodes d'approximation, équations différentielles, applications Scilab*, S. Guerre-Delabrière et M. Postel, 224 pages, 2004.
- *Précis d'analyse réelle – topologie, calcul différentiel, méthodes d'approximation, vol. 1*, V. Komornik, 208 pages, 2001.
- *Précis d'analyse réelle – analyse fonctionnelle, intégrale de Lebesgue, espaces fonctionnels, vol. 2*, V. Komornik, 256 pages, 2002.
- *Quelques aspects des mathématiques actuelles*, ouvrage collectif, 256 pages, 1999.
- *Systèmes dynamiques – une introduction*, Ch.-M. Marle, 256 pages, 2003.
- *Théorie de Galois*, I. Gozard, 224 pages, 1997.
- *Topologie*, G. Christol, A. Cot et Ch.-M. Marle, 192 pages, 1997.

ISBN 2-7298-1867-7

Présentation de la Collection

Mathématiques à l'Université

Depuis 1997, cette collection (alors appelée "Mathématiques pour le deuxième cycle") se propose de mettre à la disposition des étudiants de troisième, quatrième et cinquième années d'études supérieures en mathématiques des ouvrages couvrant l'essentiel des programmes actuels des universités françaises. Certains de ces ouvrages pourront être utiles aussi aux étudiants qui préparent le CAPES ou l'Agrégation, ainsi qu'aux élèves des Grandes Écoles et aux ingénieurs désirant actualiser leurs connaissances.

Nous avons voulu rendre ces livres accessibles à tous : les sujets traités sont présentés de manière simple et progressive, tout en respectant scrupuleusement la rigueur mathématique. Chaque volume comporte un exposé du cours avec des démonstrations détaillées de tous les résultats essentiels, des énoncés d'exercices et leurs solutions.

L'ouvrage de Monsieur Gabriel Peyré, que nous sommes heureux d'accueillir dans cette collection, est tout à fait novateur et d'un grand intérêt. Alors que la transformation de Fourier est traditionnellement enseignée d'abord en Analyse, pour les fonctions d'une variable réelle, l'auteur a choisi de présenter cette théorie pour les fonctions définies sur un groupe fini. Cette présentation lui permet, d'une part d'éviter les difficultés liées à la convergence des intégrales (remplacées par des sommes finies), d'autre part et surtout de mettre en lumière le rôle des symétries provenant de la structure de groupe, à l'origine des remarquables propriétés de cette transformation.

L'auteur a su présenter de manière naturelle et facile à assimiler des notions riches et profondes. Dans son ouvrage, les chapitres présentant des aspects théoriques alternent de manière très heureuse avec ceux traitant d'applications. Les débutants ne seront pas déroutés, et les lecteurs plus avancés trouveront dans cet ouvrage des points de vue nouveaux qui enrichiront leurs connaissances et approfondiront leur compréhension du sujet.

Charles–Michel Marle

Philippe Pilibossian

Avant-propos

Loin du temps, de l'espace, un homme est égaré,
Mince comme un cheveu, ample comme l'aurore,
Les naseaux écumants, les deux yeux révulsés,
Et les mains en avant pour tâter le décor.

RAYMOND QUENEAU, *L'explication des métaphores, Les Ziaux* (1943).

Il existe de très nombreux livres sur la transformée de Fourier ; cependant, rares sont ceux s'adressant à un public pluridisciplinaire. Ecrire un livre pour des ingénieurs avec des concepts algébriques est un vrai défi, autant, si ce n'est plus, qu'écrire un livre d'algèbre qui fasse toucher du doigt les applications des théories rencontrées. C'est ce défi que ce livre a tenté de relever. Ainsi, chaque lecteur pourra se faire un programme « à la carte » et puiser dans des énoncés ou des programmes informatiques des éléments précis pour asseoir ses connaissances dans le domaine, ou les appliquer à des problèmes plus concrets.

L'exposé est volontairement très détaillé et ne nécessite que peu de connaissances préalables, mentionnées au début des chapitres concernés. Nul doute qu'un bon élève de licence devrait pouvoir aborder cet exposé sans grande difficulté. Le lecteur pourra avoir besoin de façon ponctuelle de quelques notions avancées sur les groupes finis ainsi que d'une certaine familiarité avec les actions de groupes. Un élève agrégatif devrait quant à lui pouvoir trouver de nombreuses applications et développements autour du programme officiel.

Je n'ai pas hésité à répéter les définitions et notations importantes. Par exemple, la notion de convolution, abordée sous de nombreux angles (groupe abélien, traitement du signal, groupe non commutatif), est à chaque fois replacée dans son contexte. Ainsi, les différents paragraphes, bien que suivant une progression logique, ont une vraie unité et peuvent être lus de façon non linéaire.

Le premier chapitre utilise le langage de la théorie des groupes pour expliquer les notions principales et démontrer les énoncés dont il sera fait usage par la suite. Le deuxième chapitre applique les résultats obtenus à des problèmes divers, et constitue un premier contact avec les algorithmes rapides (*transformée de Walsh* par exemple). Le troisième chapitre est un exposé sur la transformée de Fourier discrète. Même s'il réinvestit les résultats du premier chapitre, il peut être lu par exemple par un informaticien souhaitant comprendre les mécanismes des algorithmes de transformées discrètes. Le quatrième chapitre présente des applications diverses de la transformée de Fourier discrète, et constitue un complément indispensable du chapitre précédent, pour bien comprendre les mécanismes mis en jeu ainsi que leur utilisation dans des situations pratiques. Le cinquième chapitre décline des idées et des algorithmes plus originaux autour de la transformée de Fourier, donnant

lieu à de nombreuses applications. Le sixième chapitre nécessite quelques connaissances un peu plus poussées, notamment un peu de familiarité avec la théorie des corps finis. Il étudie les transformées à valeurs dans un corps fini, et présente des applications aux codes correcteurs. Les deux derniers chapitres (les plus difficiles), sont de nature plus algébrique, et se proposent de généraliser les constructions déjà effectuées au cas des groupes finis non commutatifs. Le septième chapitre expose la théorie des représentations linéaires. Le huitième et dernier chapitre applique cette théorie dans des champs à la fois théoriques (étude de la simplicité des groupes) et pratiques (analyse spectrale).

Une bonne connaissance des propriétés algébriques de la transformée de Fourier est, à mon sens, très utile pour construire des leçons d'agrégation à la fois tournées vers les applications et avec des bases théoriques solides. De nombreuses notions au programme de l'agrégation seront passées en revue dans ce livre. Tout d'abord la notion de groupes finis est au cœur du problème abordé dans ce livre. Les groupes cycliques tels $\mathbb{Z}/n\mathbb{Z}$ sont plus particulièrement mis en avant : ce sont les groupes les plus simples, mais aussi les plus utilisés dans la pratique. Les nombres complexes de module 1 sont présents tout au long de l'exposé. L'utilisation d'espaces hermitiens et de transformations unitaires est une constante dans la théorie de Fourier. La transformée de Fourier continue et les séries de Fourier ne seront abordées que dans les exercices, toutefois, la transformée de Fourier discrète permet d'enrichir leur analyse. La résolution d'équations aux dérivées partielles utilise pleinement les propriétés algébriques des transformées de Fourier. De plus, le calcul des coefficients de Fourier par l'algorithme de transformée de Fourier rapide, ainsi que les considérations sur la convolution lors de la résolution par différences finies, font de la transformée de Fourier discrète un outil incontournable. Enfin, la théorie des corps finis peut, elle aussi, être abordée à travers le monde de Fourier.

Un certain nombre de programmes informatiques sont présentés ; ils sont rédigés en MATLAB pour la plupart, et en MAPLE pour ceux qui nécessitent des manipulations algébriques (calculs dans les corps finis, etc.). Bien qu'il s'agisse de logiciels payants, on peut en trouver des versions pour les étudiants à un prix raisonnable, et de nombreuses facultés et écoles d'ingénieurs en sont équipées. De plus, des logiciels gratuits à la syntaxe très proche existent, principalement SCILAB et MUPAD. Le choix d'un langage particulier pour implémenter les algorithmes est bien évidemment discutable, mais l'utilisation de MATLAB et MAPLE semble assez naturelle, ces logiciels permettant de tester rapidement les programmes écrits. On pourra par la suite les traduire dans un langage compilé et plus rapide, tel que le C ou le C++. De plus, ces langages sont utilisés pour l'épreuve de modélisation à l'oral de l'agrégation de mathématiques. Les agrégés ou futurs agrégés ne seront donc pas dépaysés. Il est à noter que tous les programmes présents dans cet ouvrage sont disponibles au téléchargement, ainsi que de nombreux autres, sur le site `http://www.cmap.polytechnique.fr/~peyre/adtf/`.

Je tiens à remercier mes parents, Lucien et Marie-Noëlle, qui m'ont soutenu pendant toute l'écriture de ce livre. Je dédie ce livre à Elisa Maugein. Enfin, j'adresse ma plus profonde gratitude à mes relecteurs, qui ont apporté leur expérience pour m'aiguiller dans la bonne direction : Abdellah Bechata, Vincent Beck, Christophe Bertault, Charles-Michel Marle, Jérôme Malick et Jean Starynkévitch.

Gabriel Peyré.

Table des matières

Table des notations

Notation	Signification	Page
$\widehat{G}$	Dual d'un groupe.	2
χ	Caractère d'un groupe.	2
$\mathbb{U}_n$	Ensemble des racines $n^{\text{ièmes}}$ de l'unité.	2
i	Nombre complexe tel que $\mathrm{i}^2 = -1$.	2
$\lvert z\rvert, \bar{z}$	Module et conjugué d'un nombre complexe.	2
$\mathfrak{Re}(z), \mathfrak{Im}(z)$	Parties réelle et imaginaire.	2
$\lvert G\rvert$	Ordre d'un groupe G.	3
$\mathbb{C}[G]$	Espaces des fonctions de G dans $\mathbb{C}$.	3
$a \overset{\text{déf.}}{=} b$	Egal par définition.	3
$\langle f, g\rangle$	Produit hermitien de deux fonctions.	3
$\lVert f\rVert_2$	Norme d'une fonction.	3
$L^2(\mathbb{R})$	Fonctions de carré intégrable.	3
ω	Racine primitive de l'unité.	4
$G \simeq G'$	Isomorphisme (de groupe, etc).	4
δ_p^q	Symbole de Kroneker.	6
$[G : H]$	Indice de H dans G.	6
$\widehat{\widehat{G}}$	Bidual du groupe G.	8
$\mathfrak{S}_n$	Groupe symétrique.	11
$D(G)$	Groupe dérivé de G.	12
$[x, y]$	Commutateur de x et y.	12
$\mathfrak{A}_n$	Groupe des permutations paires.	13
$c_f(\chi)$	Coefficient de Fourier.	14
$\mathscr{F}(f)$	Transformée de Fourier d'une fonction f.	14
$f * g$	Produit de convolution de f et g.	16
p.p.t.	Pour presque tout.	22
$\left(\frac{n}{p}\right)$	Symbole de Legendre.	28
$\mathrm{Tr}_{K/k}$	Trace d'un corps fini K sur k.	30
ψ_1	Caractère canonique.	31
η	Caractère quadratique.	32
$G(\chi, \psi)$	Somme de Gauss.	32
$\mathscr{F}_{\text{mul}}(f)$	Transformée de Fourier multiplicative.	33
$\mathscr{F}_{\text{add}}(f)$	Transformée de Fourier additive.	33

Notation	Signification	Page
W_{2^k}	Matrices de Walsh.	40
$\mathscr{W}_k(f)$	Transformée de Walsh de f.	41
E^*	Dual d'un espace vectoriel E.	43
$A^\perp$	Sous-espace orthogonal d'une partie A de E.	43
F^0	Sous-espace orthogonal d'une partie F de E^*.	43
$H^\sharp$	Orthogonal d'un sous-groupe.	44
$d(x,y)$	Distance de Hamming entre deux vecteurs de $\mathbb{F}_q^k$.	48
$w(z)$	Poids d'un vecteur de $\mathbb{F}_q^k$.	48
$A_H(X,Y)$	Polynôme énumérateur d'un sous-espace H de $(\mathbb{F}_2)^k$.	48
Π_s	Peigne de Dirac.	62
$\{f[k]\}_{k=0}^n$	Vecteur fini de taille N.	64
$\widehat{f}[k]$	Coefficient de Fourier discret.	64
f^0, f^1	Parties paire et impaire d'un échantillon f.	66
$\mathscr{S}_N^x$	Opérateur $\mathscr{S}$.	67
f_g, f_d	Parties gauche et droite d'un échantillon f.	67
M^*, M^{T}	Matrices adjointe et transposée.	74
$f^\sharp$	Fonction symétrisée.	82
f_s, f_a	Parties symétrique et anti-symétrique	82
$M_n(k)$	Matrices de taille n.	90
$GL_n(k)$	Matrices inversibles de taille n.	90
$\widehat{\mathbb{R}}$	Dual de $\mathbb{R}$.	96
Φ^g	Filtre linéaire.	100
$\ell^1(\mathbb{Z})$	Suites absolument sommables.	104
$\mathrm{Corr}(f,g)$	Corrélation de f et g.	125
$\overline{\mathrm{Corr}}(f,g)$	Corrélation normalisée de f et g.	125
β^n	Fonction B-spline d'ordre n.	129
$\mathrm{cas}(x)$	Noyau de Hartley.	132
$\mathscr{H}(f)$	Transformée de Hartley.	132
$\mathscr{Z}(f)$	Transformée en Z.	136
Φ_a^b	Filtre récursif.	138
$\mathscr{L}(f)$	Transformée de Laplace.	143
$\mathscr{G}_z(f)$	Transformée en Z vectorielle.	145
Φ_n	Polynôme cyclotomique.	159
Fer_n	$n^{\text{ième}}$ nombre de Fermat.	166
$w(x)$	Poids d'un mot.	169
$d(x,y)$	Distance de Hamming.	169
$\#E$	Cardinal d'un ensemble.	170

Notation	Signification	Page
$\mathscr{C}^{\perp}$	Dual d'un code.	180
$GL(V)$	Groupe linéaire d'un espace V.	194
δ_g	Fonction de base de $\mathbb{C}[G]$.	194
$K[G]$	Algèbre d'un groupe.	194
$\rho_{V\oplus W}$	Représentation somme.	196
$V \oplus W$	Somme directe de sous-espaces.	196
$\mathscr{L}(V,W)$	Espace des morphismes de V dans W.	196
$\rho_{\mathscr{L}(V,W)}$	Représentation des morphismes.	196
V^*	Dual d'un espace vectoriel V.	197
ρ_{V^*}	Représentation duale.	197
φ^{T}	Application transposée.	197
V^G	Sous-représentation invariante.	203
$\mathrm{Hom}_G(V,W)$	Opérateurs d'entrelacement.	204
$\ker(f)$	Noyau de f.	204
$\mathrm{Im}(f)$	Image de f.	204
$\dim_K(V)$	Dimension d'un espace vectoriel.	205
$Z(G)$	Centre de G.	205
R_G	Opérateur de Reynolds.	205
$\mathrm{tr}(f)$	Trace de l'endomorphisme f.	205
$\widetilde{f}$	Application moyennée.	206
χ_ρ	Caractère d'une représentation ρ.	207
$V_i^{\oplus a_i}$	Somme directe multiple.	210
$\mathrm{End}(V)$	Endomorphismes de V.	211
$\pi f(\rho)$	Transformée de Fourier de f en ρ.	211
$\mathbb{C}[G]^G$	Espace des fonctions centrales.	212
$c_f(\rho_i)$	Coefficient de Fourier.	215
$\Theta(G)$	Déterminant d'un groupe.	221
$\Theta_\rho(G)$	Déterminant d'un groupe en une représentation.	221
$c_f(k,i,j)$	Coefficients de Fourier généralisés.	234

Chapitre premier
Transformée de Fourier sur un groupe fini

> En fait, pour construire les ondelettes de base, on utilise ce qui « fonctionne bien » dans l'analyse de Fourier, c'est-à-dire le formalisme algébrique.
>
> YVES MEYER [54] (1990)

Dans ce premier chapitre, nous allons aborder l'étude de la transformée de Fourier sous un angle original, celui de la théorie des groupes. La théorie de Fourier, qu'elle soit envisagée d'un point de vue algébrique ou non, consiste avant tout en l'analyse de fonctions. Il faut ici prendre le mot analyse au pied de la lettre, dans son sens étymologique. Il s'agit de décomposer des données complexes sous une forme plus simple. Il va donc être question de construire un moyen systématique pour obtenir cette décomposition, et c'est précisément là où les outils de Fourier interviennent. Pour parvenir à réaliser de façon efficace cette décomposition, on doit tout de même utiliser un certain nombre d'informations a priori sur les fonctions que l'on étudie. Ce premier chapitre portera sur l'étude des fonctions sur un groupe fini ; la décomposition en « briques élémentaires » que réalise l'analyse de Fourier résultera des symétries inhérentes à la structure de groupe.

Le cadre le plus élémentaire pour mener à bien ce projet est celui des groupes finis commutatifs, puisque l'on n'a à se soucier ni de la régularité des fonctions rencontrées, ni de la convergence des séries manipulées (puisqu'elles sont finies !). Bien sûr, on sera tenté de crier au scandale tant le travail alors accompli semble simpliste par rapport à la théorie « analytique » des séries de Fourier. Cependant, cette étude permet d'amener de nouveaux points de vue et de poser de nouvelles questions qui seront abordées dans les prochains chapitres.

- En quoi l'étude de la transformée de Fourier sur un groupe fini peut-elle nous aider à comprendre la construction de la transformée de Fourier continue ?
- En quoi la transformée de Fourier sur les groupes finis rejoint-elle la transformée de Fourier discrète ?
- Quelles utilisations peut-on faire de la transformée de Fourier sur un groupe fini ? Comment construire des algorithmes efficaces, et comment les implémenter ?
- Enfin, que devient cette théorie quand on essaie de l'appliquer aux groupes non commutatifs ? Cette question motivera l'introduction de nouveaux outils, décrits en détail dans les deux derniers chapitres.

C'est à cet ensemble de questions que nous allons tenter de répondre. Les méthodes mises en œuvre sont multiples, elles empruntent souvent à plusieurs disciplines mathématiques.

Les références au sujet de la dualité sur les groupes finis sont nombreuses. Il y a bien sûr le livre de J.P.SERRE [65], mais aussi par exemple celui de WARUSFEL [76], pour une présentation plus détaillée. Pour une introduction à la transformée de Fourier sur un groupe commutatif, on pourra regarder l'ouvrage de DYM et MAC KEAN [29].

1 Dual d'un groupe fini

Le but de ce livre est d'étudier, d'un point de vue algébrique, les fonctions à valeurs complexes dont l'espace de départ est un groupe fini noté G. Il s'agit d'utiliser au maximum les propriétés du groupe pour obtenir des décompositions fonctionnelles intéressantes. L'idée de base de ce chapitre, celle qui guidera nos réflexions jusqu'à la fin de ce livre, consiste à étudier la façon dont on peut représenter une fonction sur un groupe G. La façon généralement la plus commune d'envisager une fonction $f : G \to \mathbb{C}$ est de considérer l'ensemble de ses valeurs $f(g)$ pour $g \in G$. L'inconvénient majeur de cette représentation est qu'elle n'exploite pas du tout la structure de notre groupe G. En quelque sorte, c'est une représentation universelle, qui ne dépend pas du tout du groupe que l'on a choisi. Pour étudier une fonction de manière efficace, il semble logique de construire une nouvelle représentation qui exploite les symétries que l'on peut trouver dans un groupe G. L'exemple le plus simple de ces symétries est le caractère cyclique du groupe $\mathbb{Z}/n\mathbb{Z}$, mais on peut bien sûr envisager des constructions plus complexes.

1.1 Définitions

Pour comprendre comment une fonction peut être plus ou moins simple à représenter, on abordera en premier lieu les fonctions les plus simples, celles qui n'opposent aucune résistance à la structure du groupe de départ G. Nous allons donc nous intéresser aux fonctions qui transportent la structure du groupe. Ces fonctions sont les morphismes du groupe G dans un groupe de l'ensemble d'arrivée, c'est-à-dire un sous-groupe de $\mathbb{C}^*$. Nous allons donc introduire les définitions adéquates.

Définition 1.1 (Caractères et dual d'un groupe). Soit G un groupe fini. Par définition, un *caractère* χ est un morphisme du groupe G dans le groupe multiplicatif $\mathbb{C}^*$. On note $\widehat{G}$ l'ensemble des caractères, qu'on appelle le *dual* de G.

$\widehat{G}$ est un groupe pour la multiplication des applications. On rappelle que cette multiplication est définie de la manière suivante.

$$\forall(\chi_1, \chi_2) \in \widehat{G}^2, \quad \chi_1\chi_2 : x \mapsto \chi_1(x)\chi_2(x).$$

Nous verrons, notamment au paragraphe 2.3, que la dualité sur un groupe fini possède de nombreux points communs avec la dualité entre les espaces vectoriels. Au prochain chapitre, plus précisément au paragraphe 3.1, chap. II, nous verrons même que dans certains cas, ce rapprochement peut devenir une identité entre les deux structures (de groupe et d'espace vectoriel). La dualité sur un groupe fini permet alors de démontrer des propriétés linéaires intéressantes. En attendant, commençons par étudier l'image d'un caractère $\chi \in \widehat{G}$.

Proposition 1.2. *Soit G un groupe fini de cardinal $|G| = n$. Les éléments de $\widehat{G}$ sont en fait les morphismes de G dans le groupe des racines $n^{ièmes}$ de l'unité,*

$$\mathbb{U}_n = \left\{ \exp\left(\frac{2ik\pi}{n}\right) \;\middle\backslash\; 0 \leqslant k < n \right\}.$$

En particulier,

$$\forall g \in G, \quad |\chi(g)| = 1, \quad \chi(g^{-1}) = \chi(g)^{-1} = \overline{\chi(g)},$$

où l'on a noté $|z|$ le module d'un nombre complexe z, et $\bar{z}$ son conjugué.

Démonstration. Notons 1 l'élément neutre de G. Il faut remarquer que, pour tout élément $g \in G$, on a $g^n = 1$. Ceci entraîne donc, pour tout $\chi \in \widehat{G}$, que $\chi(g)^n = \chi(g^n) = 1$, ce qui signifie bien que χ est à valeurs dans l'ensemble des racines $n^{\text{ième}}$ de l'unité. □

***Remarque* 1.3.** Il en découle qu'en particulier, $\widehat{G}$ est un groupe fini (car il n'y a qu'un nombre fini d'applications de G dans $\mathbb{U}_n$, qui sont des ensembles finis), commutatif. De plus, tout élément $\chi \in \widehat{G}$ est constant sur les classes de conjugaison de G, puisque

$$\forall (g,h) \in G^2, \quad \chi(h^{-1}gh) = \chi(h)^{-1}\chi(g)\chi(h) = \chi(e)\chi(g) = \chi(g). \tag{1.1}$$

Définition 1.4 (Espace des fonctions sur G). On note $\mathbb{C}[G]$ l'ensemble des fonctions de G dans $\mathbb{C}$. La notation $\mathbb{C}[G]$ sera expliquée au paragraphe 4.2. C'est un espace vectoriel sur $\mathbb{C}$. On y définit un produit scalaire hermitien, par

$$\forall (f,g) \in \mathbb{C}[G]^2, \quad \langle f,g\rangle \overset{\text{déf.}}{=} \frac{1}{|G|}\sum_{x\in G} f(x)\overline{g(x)}. \tag{1.2}$$

On définit aussi une norme $\|\cdot\|_2$ sur $\mathbb{C}[G]$ par $\|f\|_2^2 \overset{\text{déf.}}{=} \langle f,f\rangle$.

***Remarque* 1.5.** Le produit hermitien que nous venons de définir sur $\mathbb{C}[G]$ présente de fortes similitudes avec celui que l'on peut définir entre deux fonctions de $L^2(\mathbb{R})$ de la façon suivante :

$$\forall (f,g) \in L^2(\mathbb{R})^2, \quad \langle f,g\rangle = \int_{\mathbb{R}} f(x)\overline{g(x)}\mathrm{d}x.$$

La principale différence est le changement de la somme en intégrale. Une des propriétés communes de ces deux produits scalaires est l'invariance par translation. En effet, si on note $T_y(f)$ la fonction $x \in G \mapsto f(xy) \in G$ (ou son analogue continu $T_y(f) = f(\cdot + y)$), on a

$$\langle T_h(f), T_h(g)\rangle = \langle f,g\rangle .$$

Cette propriété sera constamment utilisée par la suite, entre autres pour démontrer les relations d'orthogonalité entre les caractères.

Dans le but d'étudier les fonctions de $\mathbb{C}[G]$, nous allons introduire une base canonique. La décomposition dans cette base correspond à la façon standard de représenter une fonction d'un ensemble dans $\mathscr{C}$.

Proposition 1.6. *Une base de $\mathbb{C}[G]$ est donnée par les fonctions $(\delta_g)_{g\in G}$ suivantes :*

$$\delta_g(h) \overset{\text{déf.}}{=} \begin{cases} 1 & si \quad h = g \\ 0 & si \quad h \neq g \end{cases}. \tag{1.3}$$

En particulier, $\mathbb{C}[G]$ est un espace vectoriel de dimension $n = |G|$ sur $\mathbb{C}$.

Démonstration. On vérifie de façon immédiate que la famille est orthonormée pour le produit hermitien (1.2). Comme ces fonctions ne sont pas nulles, ceci implique qu'elles forment une famille libre de $\mathbb{C}[G]$. Le fait que cette famille soit aussi génératrice provient de la décomposition canonique

$$\forall f \in \mathbb{C}[G], \quad f = \sum_{g \in G} f(g)\delta_g, \tag{1.4}$$

ce qui termine la démonstration. □

Cette proposition permet de plonger G dans $\mathbb{C}[G]$ de façon canonique par $g \mapsto \delta_g$. De plus, nous avons vu que toute fonction $f \in \mathbb{C}[G]$ se décompose dans la base $\{\delta_g\}_{g \in G}$, c'est ce qu'exprime l'équation (1.4). Cette décomposition est en apparence très simple. Nous verrons cependant au paragraphe 4.3, avec la notion de *convolution*, qu'elle ne facilite nullement les calculs. C'est pourquoi nous allons chercher une base qui présente les deux propriétés suivantes.

- Elle doit être simple à utiliser (la décomposition dans cette base doit être simple à calculer).
- Elle doit avoir des propriétés intéressantes pour les opérations algébriques que l'on souhaite utiliser (combinaison linéaire, produit, et produit de convolution de fonctions).

La famille des caractères, formée des éléments de $\widehat{G}$, pourrait être une bonne candidate ; il reste à démontrer qu'elle possède les qualités requises.

1.2 Dual d'un groupe cyclique

Avant de nous lancer dans l'étude générale de la dualité sur un groupe quelconque, prenons le temps de voir comment tout ceci se comporte dans le cas le plus simple, celui d'un groupe cyclique (dont l'archétype est $\mathbb{Z}/n\mathbb{Z}$, pour un entier n donné). En fait, cet exemple est de première importance, d'une part parce que dans la pratique, c'est la structure que l'on rencontre le plus souvent (nous verrons au chapitre III que les calculs unidimensionnels en traitement du signal utilisent la structure de $\mathbb{Z}/n\mathbb{Z}$), et d'autre part parce que nous allons utiliser ces résultats pour démontrer le cas général.

Proposition 1.7 (Le cas cyclique). *Soit $G = \{1, g_0, g_0^2, \ldots, g_0^{n-1}\}$ un groupe cyclique de cardinal n et de générateur g_0.Soit ω une racine primitive $n^{ième}$ de l'unité, par exemple $\omega = e^{\frac{2i\pi}{n}}$. Les éléments de $\widehat{G}$ sont de la forme, pour $j \in \{0, \ldots, n-1\}$,*

$$\chi_j : \begin{cases} G & \longrightarrow & \mathbb{C}^* \\ g = g_0^k & \longmapsto & (\omega^j)^k = e^{\frac{2i\pi jk}{n}} \end{cases}.$$

En particulier, on a $G \simeq \widehat{G}$.

Démonstration. Pour déterminer un caractère $\chi \in \widehat{G}$, il nous faut calculer la valeur de $\chi(g_0^k)$, pour $k \in \{0, \ldots, n-1\}$, ce qui donne

$$\chi(g_0^k) = (\omega^j)^k = \omega^{jk}.$$

Dans cette égalité, on a noté $\omega^j \overset{\text{déf.}}{=} \chi(g_0)$, avec $0 \leqslant j \leqslant n-1$, puisque, comme nous l'avons vu dans la proposition 1.2, cette quantité est une racine $n^{\text{ième}}$ de l'unité. Donc

notre caractère $\chi \in \widehat{G}$ est bien un des $\{\chi_0, \ldots, \chi_{n-1}\}$. Réciproquement, on constate que, pour $j \in \{0, \ldots, n-1\}$, les applications χ_j sont bien des morphismes de G dans $\mathbb{C}^*$, donc sont bien des éléments de $\widehat{G}$.
Enfin, si l'on identifie les éléments de $\mathbb{Z}/n\mathbb{Z}$ et leurs représentants dans $\{0, \ldots, n-1\}$, on définit une application $\psi : j \mapsto \chi_j$ de $\mathbb{Z}/n\mathbb{Z}$ dans $\widehat{G}$. Nous avons vu que cette application était surjective. D'autre part, cette application est injective (il suffit d'évaluer $\chi_j = \psi(j)$ en g_0) et c'est un morphisme (vérification élémentaire). C'est donc un isomorphisme et donc $\widehat{G}$ est isomorphe à $\mathbb{Z}/n\mathbb{Z}$, lui-même isomorphe à G. □

La figure 1.1 montre les quatre premiers caractères du groupe $\mathbb{Z}/12\mathbb{Z}$. En abscisse, on a noté $\{0, \ldots, 11\}$ les représentants du groupe $\mathbb{Z}/12\mathbb{Z}$, et les valeurs des caractères sont notées $*$. La ligne du haut montre les parties réelles des caractères, et la ligne du bas les parties imaginaires. On voit bien que les points sont régulièrement espacés le long des courbes d'équations $y = \cos\left(\frac{2\pi}{N}x\right)$ et $y = \sin\left(\frac{2\pi}{N}x\right)$.

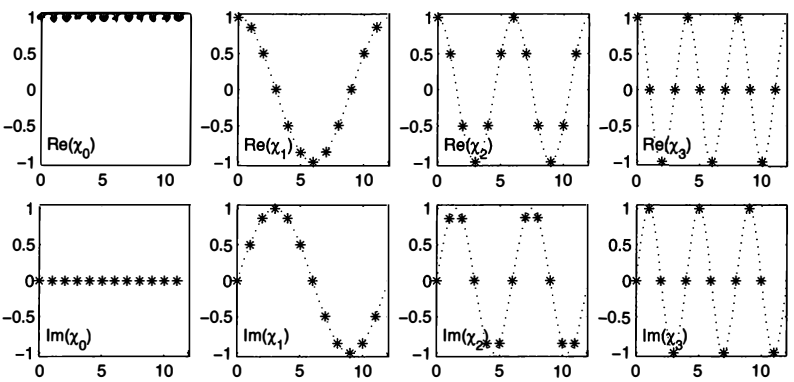

FIG. 1.1 – *Les quatres premiers caractères du groupe* $\mathbb{Z}/12\mathbb{Z}$

***Remarque* 1.8.** On peut déjà remarquer que cet isomorphisme n'est pas canonique, puisqu'il dépend du choix de la racine primitive de l'unité ω choisie. Ce phénomène est récurrent dans l'étude de la dualité (on le retrouve dans la dualité linéaire entre espaces vectoriels), le dual n'étant pas canoniquement isomorphe au groupe de départ. Tout ceci sera précisé par la suite, notamment au paragraphe 2.3.

***Remarque* 1.9.** Tout groupe cyclique est isomorphe à $\mathbb{Z}/n\mathbb{Z}$ pour $n = |G|$. En quelque sorte, l'étude de la dualité sur le groupe $\mathbb{Z}/n\mathbb{Z}$ résume celle que l'on peut faire sur n'importe quel autre groupe cyclique. Dans la suite de l'exposé, on considèrera des groupes construits à partir des briques élémentaires que sont les groupes du type $\mathbb{Z}/n\mathbb{Z}$. C'est pour cela qu'il faut garder à l'esprit la structure de la dualité sur ces groupes particulièrement simples. En appliquant la proposition 1.7, on obtient l'isomorphisme qu'il faut retenir :

$$\forall n \in \mathbb{N}^*, \quad \widehat{\mathbb{Z}/n\mathbb{Z}} \simeq \mathbb{Z}/n\mathbb{Z}.$$

Nous allons voir qu'en fait cette propriété s'étend aux groupes finis commutatifs quelconques.

Avant de clore ce paragraphe, remarquons que l'on a $|\widehat{G}| = |G| = \dim_{\mathbb{C}}(\mathbb{C}[G])$. On a même une propriété plus forte.

Proposition 1.10. *Soit G un groupe cyclique. $\widehat{G}$ forme une base orthonormale de $\mathbb{C}[G]$, ce qui signifie que*

$$\forall (p,q) \in \{0,\dots,n-1\}^2, \quad \langle \chi_p, \chi_q \rangle = \delta_p^q,$$

où on définit le symbole de Kroneker δ_p^q de la manière suivante :

$$\delta_p^q = \begin{cases} 0 & si \quad p \neq q \\ 1 & si \quad p = q \end{cases}.$$

Démonstration. On peut supposer que $G = \mathbb{Z}/n\mathbb{Z}$.
On note $\widehat{G} = \{\chi_i\}_{i=0}^{n-1}$, avec $\chi_i(k) = \omega^{ik}$, où ω est une racine $n^{\text{ième}}$ primitive (conformément à la proposition 1.7). On a alors

$$\forall (p,q) \in \{0,\dots,n-1\}^2, \quad \langle \chi_p, \chi_q \rangle = \frac{1}{n} \sum_{i=0}^{n-1} (\omega^{p-q})^i = \delta_p^q \tag{1.5}$$

(on obtient la dernière égalité en sommant la série géométrique de raison ω^{p-q}).
La famille $\widehat{G} = \{\chi_i\}_{i=0}^{n-1}$ est donc orthonormale, donc en particulier libre. Pour conclure qu'elle forme bien une base, il suffit de remarquer que son cardinal est égal à la dimension de $\mathbb{C}[G]$, puisque nous avons vu que $|G| = |\widehat{G}|$. □

***Remarque* 1.11.** La démonstration de l'orthogonalité des caractères dans le cas général d'un groupe abélien est à peine plus compliquée, et sera exposée à la proposition 2.11. Cependant, la démonstration que nous venons de faire est essentielle puisqu'elle est à la base des résultats de la transformée de Fourier unidimensionnelle, qui sera présentée à la section 1, chap. III.

2 Dual d'un groupe abélien

Notre but est d'étendre le résultat que nous venons de démontrer ($\widehat{G}$ est une base orthonormale de $\mathbb{C}[G]$) à un groupe abélien fini quelconque. Pour y parvenir, nous allons utiliser une démarche purement algébrique, qui utilise un théorème d'extension des caractères. Ensuite, nous établirons un résultat plus fort, à savoir que l'on a en fait un isomorphisme entre $\widehat{G}$ et G, propriété qu'une fois de plus, nous avons démontrée dans le cas des groupes cycliques.

2.1 Approche algébrique

Le lemme suivant est le résultat principal dans l'étude de la structure de $\widehat{G}$.

Lemme 2.1 (Prolongement de caractères). *Soit G un groupe fini commutatif et $H \subset G$ un sous-groupe. Tout caractère χ de H peut être prolongé en un caractère de G.*

Démonstration. On effectue une récurrence sur $[G:H] = |G/H|$ l'indice de H dans G. La propriété étant triviale pour $[G:H] = 1$, puisque $G = H$; on suppose donc $[G:H] > 1$, ce qui nous autorise à prendre $x \in G$ tel que $x \notin H$. Soit $K = \langle H, x \rangle$ le groupe engendré par x et H. Soit n le plus petit entier tel que $x^n \in H$. Tout élément $z \in K$ s'écrit de façon

unique sous la forme $z = yx^k$ avec $y \in H$ et $k \in \{0, \dots, n-1\}$. En effet, si $yx^k = y'x^{k'}$, avec $0 \leqslant k \leqslant k' \leqslant n-1$, alors on a $x^{k-k'} \in H$ et $k - k' < n$, donc nécessairement $k - k' = 0$ par définition de n.

Analyse : Supposons que l'on dispose d'un prolongement $\widetilde{\chi}$ de χ.
Posons $\zeta = \chi(x)$. Il nous faut $\zeta^n = \chi(x^n) = \chi(1) = 1$. Donc ζ doit être une racine $n^{\text{ième}}$ de l'unité. On a alors, nécessairement, si $z \in K$ s'écrit $z = yx^k$ avec $y \in H$ et $0 \leq k \leq n-1$,

$$\widetilde{\chi}(z) = \widetilde{\chi}(yx^k) = \chi(y)\zeta^k. \tag{2.1}$$

Synthèse : soit ζ une racine $n^{\text{ième}}$ de l'unité. Définissons, pour $z \in K$ décomposé comme précédemment sous la forme $z = yx^k$, le prolongement $\widetilde{\chi}$ par l'équation (2.1). Il s'agit de montrer que (2.1) définit bien un élément de $\widehat{K}$. L'unicité de la décomposition montre que la définition n'est pas ambiguë. Pour montrer que c'est bien un morphisme, il suffit de prendre $h = yx^k$ et $h' = y'x^{k'}$ deux éléments de K, et de distinguer deux cas.

– Si $0 \leqslant k + k' \leqslant n-1$, on a alors
$\widetilde{\chi}(hh') = \widetilde{\chi}(yy'x^{k+k'}) = \chi(yy')\zeta^{k+k'} = \chi(y)x^k\chi(y')x^{k'} = \widetilde{\chi}(h)\widetilde{\chi}(h')$.
– Si $n \leqslant k + k' \leqslant 2n-1$, on peut se ramener au cas précédent,
$\widetilde{\chi}(hh') = \widetilde{\chi}(yy'x^nx^{k+k'-n}) = \chi(y)\chi(y')\chi(x^n)\zeta^{k+k'-n} = \widetilde{\chi}(h)\widetilde{\chi}(h')$.

La propriété de multiplicativité des degrés nous dit que

$$[G:H] = [G:K][K:H], \quad \text{avec} \quad [K:H] > 1.$$

On a donc $[G:K] < [G:H]$. On peut avec l'hypothèse de récurrence prolonger $\widetilde{\chi}$ à G. □

Comme le montre le choix (arbitraire) de la racine $n^{\text{ième}}$ de l'unité ζ, le prolongement du caractère n'est bien sûr pas unique. Cependant, c'est ce résultat qui va permettre de démontrer que G et $\widehat{G}$ ont même cardinal. Pour ce faire, commençons par traduire le résultat de prolongement des caractères en termes de groupe quotient.

Lemme 2.2. *On note $\rho : \widehat{G} \twoheadrightarrow \widehat{H}$ le morphisme de restriction et $j : \widehat{G/H} \hookrightarrow \widehat{G}$ le morphisme d'extension, défini par*

$$j : \begin{cases} \widehat{G/H} & \longrightarrow & \widehat{G} \\ \chi & \longmapsto & \widetilde{\chi} \end{cases} \qquad \text{avec} \quad \widetilde{\chi}(x) \overset{\text{déf.}}{=} \chi(xH).$$

On a la suite exacte :

$$\{1\} \to \widehat{G/H} \overset{j}{\hookrightarrow} \widehat{G} \overset{\rho}{\twoheadrightarrow} \widehat{H} \to \{1\}.$$

Démonstration. D'après le lemme 2.1, ρ est surjectif.
De plus, si on considère $\chi \in \ker(\rho)$, alors $H \subset \ker(\chi)$, et donc par la propriété universelle du quotient, il existe un unique $\widetilde{\chi} \in \widehat{G/H}$ tel que $\chi(x) = \widetilde{\chi}(xH)$, c'est-à-dire $j(\widetilde{\chi}) = \chi$.
Réciproquement, un élément de $\mathrm{Im}(j)$ est trivial sur H, ce qui montre que l'on a bien $\ker(\rho) = \mathrm{Im}(j) = \widehat{G/H}$. □

Corollaire 2.3. *Soit G un groupe fini commutatif. Alors $\widehat{G}$ est de même ordre que G.*

Démonstration. On raisonne par récurrence sur $n = |G|$. Pour $n = 1$, le résultat est trivial car $\widehat{G} = \{1\}$, où l'on a noté 1 le caractère trivial sur G (c'est-à-dire la fonction qui a tout élément associe 1). Soit donc $n \geqslant 2$, ce qui permet de considérer un groupe cyclique non trivial $H \subset G$. Si $H = G$, on peut utiliser l'étude menée à la section 1.2 sur les groupes cycliques pour conclure. Dans le cas contraire, on voit par l'hypothèse de récurrence que $|\widehat{H}| = |H|$ et $|\widehat{G/H}| = |G/H|$, et le lemme 2.2 montre que $|\widehat{G}| = |\widehat{H}||\widehat{G/H}|$.
On déduit donc que $|\widehat{G}| = |H||G/H| = |G|$. □

2.2 Théorème d'isomorphisme

Les groupes G et $\widehat{G}$ ont donc le même cardinal. Bien que ce résultat soit suffisant pour la suite de l'exposé, on peut néanmoins donner un résultat plus précis, en l'occurrence expliciter un isomorphisme entre G et $\widehat{G}$. Pour ce faire, nous allons utiliser le résultat obtenu pour les groupes cycliques à la section 1.2, et nous allons nous y ramener en utilisant le théorème de structure des groupes abéliens. On peut trouver une démonstration de ce résultat important dans le livre d'ARTIN [3]. On rappelle l'énoncé du théorème, sans donner de démonstration.

Théorème 2.4 (Théorème de structure des groupes abéliens). *Soit G un groupe abélien fini. Il existe des entiers strictement positifs $n_1,\dots,n_r$ uniquement déterminés tels que n_k divise n_{k+1}, et tels que l'on ait l'isomorphisme*

$$G \simeq \mathbb{Z}/n_1\mathbb{Z} \times \mathbb{Z}/n_2\mathbb{Z} \times \cdots \times \mathbb{Z}/n_r\mathbb{Z}.$$

Corollaire 2.5 (Théorème d'isomorphisme). *Soit G un groupe fini commutatif. Alors $\widehat{G}$ est isomorphe à G. En particulier, G et $\widehat{G}$ ont même ordre.*

Démonstration. Il suffit de remarquer que si G et H sont deux groupes finis commutatifs, on a $\widehat{G\times H} \simeq \widehat{G}\times\widehat{H}$. En effet, si on note $i_G : G \to G\times H$ et $i_H : H \to G\times H$ les injections canoniques, alors l'application

$$\Phi : \left\{ \begin{array}{ccc} \widehat{G\times H} & \longrightarrow & \widehat{G}\times\widehat{H} \\ \chi & \longmapsto & (\chi\circ i_G, \chi\circ i_H) \end{array} \right.$$

est un isomorphisme. Elle est trivialement injective et, pour $(\chi_1,\chi_2)\in\widehat{G}\times\widehat{H}$, l'application $\chi : (g,h)\mapsto\chi_1(g)\chi_2(h)$ vérifie bien $\chi\in\widehat{G\times H}$ et $\Phi(\chi)=(\chi_1,\chi_2)$.
On conclut ensuite en utilisant le théorème de structure 2.4 ainsi que la remarque 1.9. □

***Remarque* 2.6.** L'isomorphisme $G\simeq\widehat{G}$ que nous venons de mettre à jour n'a absolument rien de canonique. En effet, ce dernier dépend totalement de choix arbitraires pour décrire la structure du groupe, telle qu'elle est donnée par le théorème 2.4. En effet, si on conserve les notations de ce théorème, chaque choix d'un élément d'ordre n_1 envoyé sur $(1,0,\dots,0)\in\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_r\mathbb{Z}$ permet de construire un nouvel isomorphisme. Il faut rapprocher ce phénomène de l'isomorphisme d'espaces vectoriels $E\simeq E^*$ qui est réalisé via le choix (arbitraire) d'une base. Enfin, on peut ajouter que même l'isomorphisme $\widehat{\mathbb{Z}/n\mathbb{Z}}\simeq\mathbb{Z}/n\mathbb{Z}$ n'est pas canonique, puisqu'il dépend du choix d'une racine primitive de l'unité, comme expliqué à la proposition 1.7.

2.3 Le bidual

Nous avons vu au paragraphe 2.2 que l'isomorphisme $G\simeq\widehat{G}$ n'était pas canonique. Cependant, toujours par analogie avec la dualité en algèbre linéaire, on peut s'intéresser à l'étude du *bidual*. Nous allons voir que, dans ce cas, on a bien un isomorphisme canonique avec le groupe de départ.

Définition 2.7 (Bidual). Nous avons construit le dual $\widehat{G}$ d'un groupe fini commutatif G, qui à son tour est un groupe fini commutatif. On peut lui associer son dual que l'on notera $\widehat{\widehat{G}}$, le *bidual* de G.

Proposition 2.8 (Isomorphisme canonique). *On a un isomorphisme canonique $G \simeq \widehat{\widehat{G}}$, qui est donné par l'application*

$$\Phi : \left\{ \begin{array}{ccc} G & \longrightarrow & \widehat{\widehat{G}} \\ g & \longmapsto & \big(\Phi(g) : \chi \mapsto \chi(g)\big) \end{array} \right. . \tag{2.2}$$

Démonstration. Tout d'abord, on constate que Φ est bien un morphisme de groupes. Comme G et $\widehat{\widehat{G}}$ ont même cardinal (en effet, un groupe et son dual ont même cardinal, et on applique ce résultat d'une part au groupe G, d'autre part au groupe $\widehat{G}$), il suffit de montrer que Φ est injective. Dire $g \in \ker(\Phi)$ signifie que $\forall \chi \in \widehat{G}$, $\chi(g) = 1$. Pour en conclure que $g = 1$ il suffit d'exhiber, pour $h \in G$ différent de 1, un caractère $\chi \in \widehat{G}$ tel que $\chi(h) \neq 1$. Pour construire ce caractère, on peut considérer le groupe $H \subset G$ engendré par $h \neq 1$. Comme il est cyclique, de cardinal plus grand que 1, on sait construire un caractère χ_0 tel que $\chi_0(h) \neq 1$ (à la section 1.2 on a énuméré tous les caractères d'un groupe cyclique). Le lemme 2.1 montre qu'on peut prolonger χ_0 en un caractère $\chi \in \widehat{G}$ qui vérifie encore $\chi(h) \neq 1$ puisque $\chi(h) = \chi_0(h) \neq 1$. □

***Remarque* 2.9.** On retrouve un phénomène semblable à celui que l'on rencontre sur les espaces vectoriels de dimension *finie* avec l'isomorphisme canonique $E \simeq E^{**}$ qui est défini de la même manière qu'en (2.2). Bien sûr, cette remarque ne tient plus si l'espace vectoriel est de dimension *infinie*, ou si le groupe est *infini*. On est tout d'abord obligé d'introduire des contraintes de continuité sur les applications que l'on envisage, et même sous ces conditions, il arrive rarement que le dual soit isomorphe à la structure de départ. Un bon exemple est donné au paragraphe 1.1, chap. IV. Nous verrons en effet que le dual du *tore* $\mathbb{R}/2\pi\mathbb{Z}$ est isomorphe à $\mathbb{Z}$. L'exercice I.2 propose de traiter le cas d'un groupe infini non commutatif, $SO(3)$.

2.4 Relations d'orthogonalité

On peut étendre sans grande difficulté l'orthogonalité des caractères obtenue dans le cas cyclique (proposition 1.10) au cas d'un groupe abélien quelconque. Commençons par démontrer un lemme qui sera très utile pour la suite.

Lemme 2.10. *Soit G un groupe abélien fini. Pour $\chi \in \widehat{G}$, on a*

$$\sum_{g \in G} \chi(g) = \left\{ \begin{array}{lll} 0 & si & \chi \neq 1 \\ |G| & si & \chi = 1 \end{array} \right. . \tag{2.3}$$

Démonstration. Si $\chi = 1$, alors la propriété à démontrer est bien sûr vérifiée. Supposons donc que $\chi \neq 1$. Soit $t \in G$ tel que $\chi(t) \neq 1$. On a alors

$$\chi(t) \sum_{g \in G} \chi(g) = \sum_{g \in G} \chi(tg) = \sum_{h \in G} \chi(h),$$

où l'on a fait le changement de variable $h = tg$ dans la dernière somme (qui est valide car $g \mapsto tg$ est une bijection de G). On en déduit donc que

$$(\chi(t) - 1) \sum_{g \in G} \chi(g) = 0 \quad \Longrightarrow \quad \sum_{g \in G} \chi(g) = 0.$$

Ce qui termine la démonstration. □

Proposition 2.11 (Orthogonalité des caractères). *Soit G un groupe fini commutatif. Alors $\widehat{G}$ est une famille orthonormale d'éléments, c'est-à-dire :*

$$\forall(\chi_1,\chi_2)\in\widehat{G}^2,\quad \langle\chi_1,\chi_2\rangle=\begin{cases}0 & si\quad \chi_1\neq\chi_2\\ 1 & si\quad \chi_1=\chi_2\end{cases}.$$

Démonstration. On note $\chi \overset{\text{déf.}}{=} \chi_1\overline{\chi_2}=\chi_1\chi_2^{-1}$ (les $\chi_2(s)$ sont de module 1, donc il vient $\overline{\chi_2(s)}=\chi_2(s)^{-1}$). On a

$$\langle\chi_1,\chi_2\rangle=\frac{1}{|G|}\sum_{g\in G}\chi(g).$$

Il ne reste plus qu'à remarquer que si $\chi_1=\chi_2$, alors $\chi=1$, et que sinon, $\chi\neq 1$. On termine en appliquant le lemme 2.10. □

Corollaire 2.12. *Soit G un groupe fini commutatif. Alors $\widehat{G}$ est une base orthonormale de $\mathbb{C}[G]$.*

Démonstration. Le fait que $\widehat{G}$ soit une famille orthogonale implique en particulier que c'est une famille libre de $\mathbb{C}[G]$. Comme G et $\widehat{G}$ ont même cardinal, qui est aussi la dimension de $\mathbb{C}[G]$ en tant que $\mathbb{C}$-espace vectoriel, c'est une base. □

Nous avons donc mené à bien le programme que nous nous étions fixé, en explicitant une base de l'espace des fonctions $\mathbb{C}[G]$ à la fois simple (comme on le verra à la section 2, chap. III, les propriétés des racines de l'unité vont permettre des calculs rapides des projections sur notre base), et avec des propriétés algébriques intéressantes (qui seront exploitées entre autres au paragraphe 4.3).

Une fois démontrées ces relations d'orthogonalité entre les caractères, on peut démontrer d'autres relations, qui sont en quelque sorte « duales ».

Proposition 2.13. *Soit g et h deux éléments de G. On a alors*

$$\sum_{\chi\in\widehat{G}}\chi(g)\overline{\chi(h)}=\begin{cases}0 & si\quad g\neq h\\ |G| & si\quad g=h\end{cases}.$$

Démonstration. Il s'agit juste d'appliquer la proposition 2.11 au groupe abélien $\widehat{G}$. Pour g et $h\in\widehat{\widehat{G}}$, on obtient alors

$$\sum_{\chi\in\widehat{G}}g(\chi)\overline{h(\chi)}=\begin{cases}0 & \text{si}\quad g\neq h\\ |\widehat{G}|=|G| & \text{si}\quad g=h\end{cases}. \tag{2.4}$$

Nous avons vu au paragraphe précédent que l'on peut en fait identifier canoniquement un élément $g\in\widehat{\widehat{G}}$ à un élément $\widetilde{g}\in G$ en posant $g(\chi)=\chi(\widetilde{g})$. Si on réécrit l'équation (2.4) en utilisant ces nouvelles notations, on obtient exactement la formule voulue. □

***Remarque* 2.14.** On peut représenter les caractères d'un groupe G sous la forme d'une matrice carrée $M=\{m_{ij}\}_{1\leqslant i,j\leqslant n}$ de taille $n\overset{\text{déf.}}{=}|G|$. Chaque ligne représente les valeurs d'un caractère. Plus précisément, si on note $G=\{g_1,\ldots,g_n\}$ et $\widehat{G}=\{\chi_1,\ldots,\chi_n\}$, alors on pose $m_{ij}=\chi_i(g_j)$. Dans ce cadre, la proposition 2.11 énonce des relations d'orthogonalité entre les lignes de la matrice, tandis que la proposition 2.13 énonce des relations d'orthogonalité entre les colonnes de la matrice.

3 Dual d'un groupe non commutatif

Après avoir mené à bien l'étude de la dualité sur un groupe abélien fini, on peut vouloir étendre ces résultats au cas des groupes finis non commutatifs. Cependant, nous allons voir que la belle mécanique que nous venons de développer tombe très vite en défaut, même sur des groupes extrêmement courants comme les groupes symétriques. Nous allons ensuite voir que cette situation est générale, puisque nous allons démontrer que pour tout groupe non commutatif, on est systématiquement confronté à un manque de caractères.

3.1 Exemple du groupe symétrique

Le fait que $\widehat{G}$ soit isomorphe à G, et même que $|\widehat{G}| = |G|$ tombe en défaut lorsque G n'est plus commutatif. Nous allons le voir sur un exemple concret, le groupe symétrique $\mathfrak{S}_n$. Rappelons tout d'abord la définition de la *signature* ainsi que quelques propriétés fondamentales.

Définition 3.1 (Signature). On considère la décomposition d'une permutation $\sigma \in \mathfrak{S}_n$ en produit de cycles disjoints. On rappelle en effet qu'une telle décomposition existe et est unique à l'ordre près des facteurs. Pour démontrer ceci, on pourra regarder le livre de LANG [43]. Si $\sigma \in \mathfrak{S}_n$ se décompose en produit de k cycles disjoints, on pose

$$\varepsilon(\sigma) \overset{\text{déf.}}{=} (-1)^{n-k}.$$

Cette définition est non ambiguë, et pour vérifier que c'est bien un morphisme, on revient à la définition en termes de transpositions en utilisant le lemme suivant.

Lemme 3.2. *Soit $\sigma \in \mathfrak{S}_n$ et τ une transposition. Alors on a $\varepsilon(\sigma\tau) = -\varepsilon(\sigma)$.*

Démonstration. On note τ la transposition (a, b). Pour démontrer le lemme, il faut compter le nombre de cycles dans chaque décomposition et considérer deux cas. Tout d'abord, si a et b interviennent dans un même cycle c de la décomposition de σ. Alors, $\sigma\tau$ va avoir la même décomposition, à l'exception du cycle c qui va être scindé en deux. Dans le deuxième cas, on suppose que a et b interviennent dans deux cycles disjoints c_1 et c_2 dans l'écriture de σ. Dans ce cas, l'écriture de $\sigma\tau$ va présenter un cycle de moins, puisque les cycles c_1 et c_2 seront réunis. Dans les deux cas, les nombres de cycles intervenant dans les écritures de σ et de $\sigma\tau$ diffèrent d'une unité, ce qui prouve le lemme. □

On peut alors démontrer la propriété fondamentale suivante.

Proposition 3.3. *On suppose que $\sigma \in \mathfrak{S}_n$ s'écrit comme le produit de p transpositions. On a alors $\varepsilon(\sigma) = (-1)^p$.*

Démonstration. On démontre cette propriété par une récurrence sur la longueur de la décomposition en transposition, et en utilisant le lemme précédent. □

Il faut insister sur le fait que cette propriété ne permet pas de définir directement la signature ε, car la décomposition en transposition n'est pas unique. On est obligé d'utiliser la décomposition en cycles disjoints. Une fois ce travail de construction effectué, on est en mesure de déterminer le dual de $\mathfrak{S}_n$.

Proposition 3.4. *Le seul caractère non trivial de $\mathfrak{S}_n$ est la signature ε.*

Démonstration. Soit χ un caractère de $\mathfrak{S}_n$. Comme les transpositions engendrent $\mathfrak{S}_n$, il suffit de déterminer les valeurs que peut prendre χ sur les transpositions. Or on constate que deux transpositions $\tau_1 = (a,b)$ et $\tau_2 = (c,d)$ de $\mathfrak{S}_n$ sont toujours conjuguées. En effet, construisons une permutation g dans $\mathfrak{S}_n$ telle que $g(a) = c$, $g(b) = d$.
On a $\tau_2 = g\tau_1 g^{-1}$. Ceci implique que χ, qui est constant sur les classes de conjugaison (comme nous l'avons vu à l'équation (1.1)), prend une seule et même valeur sur toutes les transpositions. Comme $\chi(\tau_1^2) = \chi(\tau_1)^2 = 1$, on a $\chi(\tau_1) = +1$ ou $\chi(\tau_1) = -1$. Donc nécessairement, un caractère non trivial χ doit vérifier $\chi(\tau_1) = -1$. De plus, cette condition suffit, sous réserve d'existence, à déterminer χ.
Réciproquement, on a établi l'existence d'un caractère non trivial : la signature. C'est donc le seul. □

On voit donc que l'on a $\widehat{\mathfrak{S}_n} \simeq \mathbb{Z}/2\mathbb{Z}$. Cette étude faite dans le cas du groupe symétrique peut être généralisée ; c'est ce que nous allons voir dans le prochain paragraphe.

3.2 Utilisation du groupe dérivé

On peut en fait décrire de façon précise le dual d'un groupe en termes de groupe dérivé, puis appliquer cette description pour retrouver le dual du groupe symétrique $\mathfrak{S}_n$. Commençons par rappeler la définition ainsi que les principales propriétés du groupe dérivé.

Définition 3.5 (Groupe dérivé). Soit G un groupe, on note $[x,y] \overset{\text{déf.}}{=} xyx^{-1}y^{-1}$ le *commutateur* associé au couple $(x,y) \in G^2$. On note $D(G)$ le groupe engendré par les commutateurs de G, que l'on nomme *groupe dérivé* de G. c'est-à-dire $D(G) \overset{\text{déf.}}{=} \langle [x,y]\ ;\ (x,y) \in G^2 \rangle$.

Proposition 3.6 (Propriétés du groupe dérivé). *On a $D(G) \lhd G$ (c'est-à-dire $D(G)$ est distingué dans G), et $G/D(G)$ est un groupe commutatif. De plus, $D(G) = \{1\}$ si et seulement si G est commutatif.*

Démonstration. Si $\varphi \in Aut(G)$ est un automorphisme de G, on a

$$\forall (x,y) \in G^2, \quad \varphi([x,y]) = [\varphi(x), \varphi(y)],$$

de sorte que les commutateurs sont conservés par les automorphismes. Il en est donc de même du groupe dérivé qui est engendré par ces commutateurs. En particulier, $D(G)$ est conservé par les automorphismes intérieurs, ce qui est la définition d'un sous-groupe distingué.
Si on note $\bar{x}$ et $\bar{y}$ les classes de x et y éléments de $G/D(G)$, alors $[x,y] \overset{\text{déf.}}{=} xyx^{-1}y^{-1}$ est un élément de $D(G)$, donc $\bar{x}\bar{y}\bar{x}^{-1}\bar{y}^{-1} = 1$ dans $G/D(G)$, ce qui veut dire que $\bar{x}$ et $\bar{y}$ commutent.
La dernière propriété est claire avec la définition du groupe dérivé. □

Proposition 3.7. *Soit G un groupe fini. On a $\widehat{G} \simeq G/D(G)$.*

Démonstration. On peut introduire le morphisme suivant :

$$\Phi : \begin{cases} \widehat{G} & \longrightarrow & \widehat{G/D(G)} \\ \chi & \longmapsto & \overline{\chi} \end{cases},$$

où $\overline{\chi}$ est défini par $\overline{\chi}(\bar{x}) \overset{\text{déf.}}{=} \chi(x)$, où l'on a noté $\bar{x}$ la classe de x dans $G/D(G)$. Cet élément $\overline{\chi} \in \widehat{G/D(G)}$ est bien défini. En effet, comme $\mathbb{C}$ est commutatif, pour tout commutateur

$[x,y]$ on a $\chi([x,y]) = [\chi(x),\chi(y)] = 1$. Ainsi, la définition de $\overline{\chi}(\bar{x})$ ne dépend pas du représentant choisi.
Ce morphisme Φ est trivialement injectif, puisque $\forall x \in G$, $\chi(x) = \overline{\chi}(\bar{x})$. De plus, on peut construire explicitement un antécédent pour un élément $\chi_1 \in \widehat{G/D(G)}$, il suffit de construire le caractère χ défini par l'égalité $\chi(x) = \chi_1(\bar{x})$.
Nous avons donc montré que $\widehat{G} \simeq \widehat{G/D(G)}$. Mais comme $G/D(G)$ est commutatif (proposition 3.6), on peut utiliser le théorème d'isomorphisme 2.5 et conclure à l'isomorphisme $\widehat{G/D(G)} \simeq G/D(G)$, ce qui termine la preuve de cette proposition. □

***Remarque* 3.8.** En fait, la propriété que l'on a utilisée dans la démonstration est qu'un morphisme qui est trivial sur les commutateurs passe au quotient par $D(G)$, ce qui conduit au diagramme commutatif suivant :

$$\begin{array}{ccc} G & \xrightarrow{\chi} & \mathbb{C}^* \\ \downarrow{\scriptstyle \pi} & & \| \\ D(G) & \xrightarrow{\overline{\chi}} & \mathbb{C}^* \end{array}.$$

Montrons maintenant que l'on retrouve bien la description du dual du groupe $\mathfrak{S}_n$ obtenue à la section précédente 3.1. Rappelons tout d'abord que l'on note $\mathfrak{A}_n$ le sous-groupe des permutations paires, c'est-à-dire

$$\mathfrak{A}_n \overset{\text{déf.}}{=} \{\sigma \in \mathfrak{S}_n \setminus \varepsilon(\sigma) = 1\},$$

où ε désigne la signature. $\mathfrak{A}_n$ est un sous-groupe distingué de $\mathfrak{S}_n$, puisque $\mathfrak{A}_n = \ker(\varepsilon)$, et que ε est un morphisme (à valeurs dans $\{-1,1\}$). Mais avant toute chose, voici un lemme qui précise la structure du groupe $\mathfrak{A}_n$.

Lemme 3.9. *Pour $n \geqslant 3$, $\mathfrak{A}_n$ est engendré par les cycles de longueur* 3.

Démonstration. La première chose à remarquer est que $\mathfrak{S}_n$ est engendré par les transpositions $(1,i)$ pour $i = 2 \ldots n$. Ceci est évident en remarquant que pour $i \neq j$, on a $(i,j) = (1,i)(1,j)(1,i)$. Maintenant, il suffit de réaliser qu'un élément de $\mathfrak{A}_n$ ne peut être engendré que par un nombre pair de transpositions. On voit donc que $\mathfrak{A}_n$ est engendré par les éléments de la forme $(1,i)(1,j) = (1,i,j)$ qui sont des 3-cycles. □

Proposition 3.10 (Cas du groupe symétrique). *Pour $n \geqslant 3$, on a $D(\mathfrak{S}_n) = \mathfrak{A}_n$. On a donc $\widehat{\mathfrak{S}_n} \simeq \mathfrak{S}_n/\mathfrak{A}_n \simeq \mathbb{Z}/2\mathbb{Z}$.*

Démonstration. Comme ε est un caractère, on a $D(\mathfrak{S}_n) \subset \mathfrak{A}_n$. Comme pour $n \geqslant 3$, $\mathfrak{A}_n$ est engendré par les 3-cycles, il suffit de montrer que tout 3-cycle est un commutateur pour montrer l'inclusion inverse. Pour tout 3-cycle $\sigma = (a,b,c)$ on a $\sigma^2 = (a,c,b)$ qui est encore un trois cycle. Comme deux cycles de même longueur sont conjugués dans $\mathfrak{S}_n$ (résultat classique qui fait l'objet du lemme 1.36, chap. VII), on peut trouver un élément $\tau \in \mathfrak{S}_n$ tel que $\sigma^2 = \tau\sigma\tau^{-1}$. On a donc $\sigma = [\tau,\sigma]$ et on a terminé.
Le fait que $\mathfrak{S}_n/\mathfrak{A}_n \simeq \mathbb{Z}/2\mathbb{Z}$ résulte du passage au quotient de $\varepsilon : \mathfrak{S}_n \to \{-1,1\}$ par $\mathfrak{A}_n$ qui est le noyau de ce morphisme. □

***Remarque* 3.11.** La solution pour contourner ce problème de « manque » de caractères est d'introduire la notion de représentation linéaire, qui généralise la notion de caractère ($\widehat{G}$ est constitué des caractères des représentations de dimension 1). En quelque sorte, un groupe non commutatif n'a pas assez de représentations en dimension 1, et il faut passer aux dimensions supérieures. Tout ceci sera l'objet du chapitre VII.

4 Transformée de Fourier

L'idée directrice des paragraphes suivants est de formaliser de manière algébrique la transformée de Fourier en utilisant la structure de groupe de l'ensemble de départ. On retrouvera de nombreuses similitudes avec la transformée de Fourier sur $\mathbb{R}$ (les intégrales étant remplacées par des sommes finies), et les propriétés utiles de la transformée de Fourier (par exemple celles qui sont liées au produit de convolution) seront expliquées en termes de morphismes de groupes (en l'occurrence finis).

4.1 Coefficients de Fourier et transformée de Fourier

Ce paragraphe présente la construction des coefficients de Fourier puis de la transformée de Fourier, dans le cadre d'un groupe abélien fini. Il s'agit simplement d'exploiter la propriété d'orthogonalité des caractères que nous venons de démontrer.

Définition 4.1 (Coefficients de Fourier). Pour $f \in \mathbb{C}[G]$ on définit, pour $\chi \in \widehat{G}$, le *coefficient de Fourier* $c_f(\chi)$ par

$$\forall \chi \in \widehat{G}, \quad c_f(\chi) \overset{\text{déf.}}{=} \langle f, \chi \rangle .$$

Ceci permet donc de définir l'application c :

$$c : \left\{ \begin{array}{ccc} \mathbb{C}[G] & \longrightarrow & \mathbb{C}[\widehat{G}] \\ f & \longmapsto & c_f \end{array} \right. .$$

Dans la pratique, on utilise souvent une autre notation que celle des coefficients de Fourier, en introduisant la *transformée de Fourier*.

Définition 4.2 (Transformée de Fourier). L'application *transformée de Fourier*, notée $\mathscr{F}$, est définie par

$$\mathscr{F} : \left\{ \begin{array}{ccc} \mathbb{C}[G] & \longrightarrow & \mathbb{C}[\widehat{G}] \\ f & \longmapsto & \widehat{f} \end{array} \right. , \tag{4.1}$$

où $\widehat{f}$ est définie par

$$\forall \chi \in \widehat{G}, \quad \widehat{f}(\chi) \overset{\text{déf.}}{=} |G| c_f(\overline{\chi}) = \sum_{x \in G} f(x)\chi(x).$$

Cette définition est en fait très naturelle, comme le montrera la proposition 4.15. La figure 1.2 montre les valeurs de la transformée de Fourier d'une fonction « en cloche » f définie sur $\mathbb{Z}/17\mathbb{Z}$. En abscisse, on a noté les indices $i \in \{-8, \ldots, 0, \ldots, 8\}$ des caractères χ_i (les indices sont pris dans $[-8,8]$ plutôt que $[0,16]$ pour que les dessins soient plus jolis). En ordonnée, on trouve les valeurs de la transformée de Fourier $\widehat{f}(\chi_i)$. On pourra vérifier que la valeur centrale (pour $i = 0$) est bien la somme des valeurs de la fonction f.

***Remarque* 4.3.** Les morphismes c et $\mathscr{F}$ sont bien sûr linéaires, ce sont donc des morphismes d'espaces vectoriels de $\mathbb{C}[G]$ dans $\mathbb{C}[\widehat{G}]$. Ce sont en fait des *isomorphismes* d'espaces vectoriels, et pour le démontrer, nous allons utiliser la formule d'inversion de Fourier suivante.

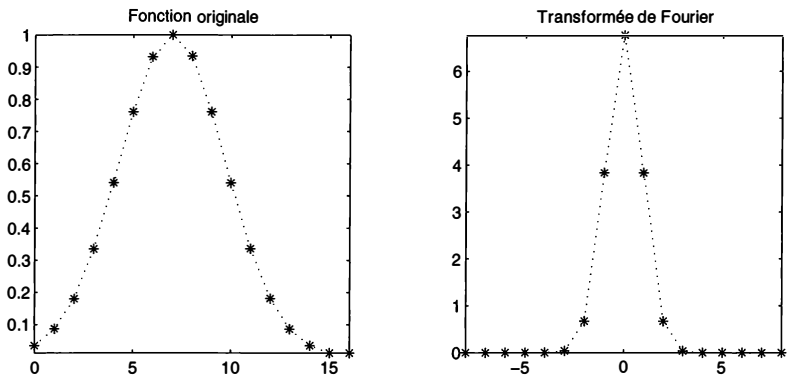

FIG. 1.2 – *Exemple de transformée de Fourier*

Proposition 4.4 (Formule d'inversion). *Pour $f \in \mathbb{C}[G]$, on a la formule d'inversion*

$$f = \sum_{\chi \in \widehat{G}} c_f(\chi)\chi = \frac{1}{|G|} \sum_{\chi \in \widehat{G}} \widehat{f}(\chi)\chi^{-1}. \tag{4.2}$$

Démonstration. L'équation (4.2) résulte immédiatement du fait que $\widehat{G}$ est une base orthonormale de $\mathbb{C}[G]$, en décomposant f dans cette base. □

Proposition 4.5 (Isomorphisme de Fourier). *c et $\mathscr{F}$ sont des isomorphismes d'espaces vectoriels de $\mathbb{C}[G]$ dans $\mathbb{C}[\widehat{G}]$.*

Démonstration. Montrons que c est injectif. Si $c_f = 0$, alors la formule d'inversion (4.2) montre que $f = 0$. Comme G et $\widehat{G}$ ont même cardinal (proposition 2.3) les espaces $\mathbb{C}[G]$ et $\mathbb{C}[\widehat{G}]$ ont même dimension $|G| = |\widehat{G}|$. On en conclut donc que c est bien un isomorphisme. Le raisonnement est identique pour $\mathscr{F}$. □

***Remarque* 4.6.** En réalité, $\mathscr{F}$ est plus qu'un isomorphisme d'espaces vectoriels, puisqu'il conserve aussi une structure d'algèbre bien particulière, celle définie par le produit de convolution. Tout ceci est l'objet du paragraphe 4.3.

Pour l'instant, continuons à énoncer les formules que l'on obtient en utilisant la décomposition dans la base des caractères.

Proposition 4.7 (Formule de Plancherel). *Pour $(f,g) \in \mathbb{C}[G]^2$ on a les formules suivantes :*

$$\sum_{s \in G} f(s)\overline{g(s)} = |G| \sum_{\chi \in \widehat{G}} c_f(\chi)\overline{c_g(\chi)} \tag{4.3}$$

$$= \frac{1}{|G|} \sum_{\chi \in \widehat{G}} \widehat{f}(\chi)\overline{\widehat{g}(\chi)}. \tag{4.4}$$

Démonstration. En décomposant f et g sous la forme $f(s) = \sum_{\chi \in \widehat{G}} c_f(\chi)\chi(s)$ ainsi que $g(s) = \sum_{\chi \in \widehat{G}} c_g(\chi)\chi(s)$, il vient

$$\sum_{s \in G} f(s)\overline{g(s)} = |G| \langle f, g \rangle = |G| \sum_{(\chi_1,\chi_2) \in \widehat{G}^2} c_f(\chi_1)\overline{c_g(\chi_2)} \langle \chi_1, \chi_2 \rangle.$$

D'où l'équation (4.3) en utilisant les relations d'orthogonalité entre les caractères. On démontre de même l'équation (4.4). □

***Remarque* 4.8. (Lien avec la théorie L^2).** Cette formule est en tout point semblable à la formule que l'on obtient pour la transformée de Fourier de deux fonctions de $L^2(\mathbb{R})$. Elle traduit la conservation du produit scalaire (à une constante près) par la transformée de Fourier, puisqu'on peut la réécrire sous la forme :

$$\forall (f,g) \in \mathbb{C}[G]^2, \quad \langle f,g\rangle = \frac{1}{|G|}\left\langle \widehat{f},\widehat{g}\right\rangle,$$

le deuxième produit scalaire étant bien sûr celui de $\mathbb{C}[\widehat{G}]$. Ceci sera expliqué en détail à la section 1, chap. IV, où l'on aborde le lien entre la transformée de Fourier sur les vecteurs de $\mathbb{C}^N$ (appelée *transformée discrète*) et la transformée de Fourier *continue*.

4.2 Algèbre d'un groupe abélien

G désigne toujours un groupe abélien fini. Depuis le début de cet exposé, on a noté $\mathbb{C}[G]$ l'espace (vectoriel) des fonctions de G dans $\mathbb{C}$. On peut lui conférer une structure d'algèbre grâce au produit de fonctions défini de la façon suivante :

$$\forall (f_1,f_2) \in \mathbb{C}[G]^2, \forall g \in G, \quad (f_1 \cdot f_2)(g) \overset{\text{déf.}}{=} f_1(g)f_2(g). \tag{4.5}$$

Cependant, ce n'est pas cette structure qui va nous être utile pour la suite, mais plutôt celle définie par le produit de convolution. En effet, comme nous allons le voir, le produit de convolution dépend intimement de la structure du groupe G considéré, contrairement au produit terme à terme défini par l'équation (4.3). Nous verrons ainsi au paragraphe 4.3 que la transformée de Fourier se comporte de façon très agréable pour le produit de convolution.

Nous avons déjà vu (à la proposition 1.6) qu'une base de l'espace vectoriel $\mathbb{C}[G]$ est donnée par les fonctions $\{\delta_g\}_{g\in G}$, où la fonction δ_g, pour $g \in G$, vérifie $\delta_g(g) = 1$ et $\delta_g(h) = 0$ pour $h \in G$ tel que $g \neq h$. De même, nous avons vu qu'une fonction $f \in \mathbb{C}[G]$ se décomposait dans la base $\{\delta_g\}_{g\in G}$ en

$$f = \sum_{g\in G} f(g)\delta_g.$$

On injecte alors le groupe G dans l'espace vectoriel $\mathbb{C}[G]$ via l'application

$$j : g \in G \mapsto \delta_g \in \mathbb{C}[G].$$

Ceci permet de définir (par transport de structure) une multiplication notée $*$ entre les éléments $\{\delta_g\}_{g\in G}$:

$$\forall (g,h) \in G^2, \quad \delta_g * \delta_h \overset{\text{déf.}}{=} \delta_{gh}.$$

Il ne reste alors plus qu'à étendre par bilinéarité cette multiplication à $\mathbb{C}[G]$ tout entier pour munir $\mathbb{C}[G]$ d'un structure d'algèbre. Ce produit est nommé produit de convolution, et on calcule facilement la formule donnant l'expression d'un produit de deux fonctions.

Définition 4.9 (Produit de convolution). Pour f_1 et f_2 deux fonctions de $\mathbb{C}[G]$, le produit de convolution $f_1 * f_2$ est donné par

$$\forall g \in G, \quad (f_1 * f_2)(g) \overset{\text{déf.}}{=} \sum_{\substack{(h,k)\in G^2 \\ hk = g}} f_1(h)f_2(k) = \sum_{h\in G} f_1(h)f_2(h^{-1}g). \tag{4.6}$$

***Remarque* 4.10.** Pour $f \in \mathbb{C}[G]$ on a

$$\forall (g,h) \in G^2, \quad (f * \delta_g)(h) = f(hg^{-1}).$$

Ainsi la convolution par un élément de G (c'est-à-dire la convolution par une fonction δ_g, en utilisant l'identification) correspond à une translation de la fonction. Ces propriétés seront expliquées à nouveau dans le cadre simple de $G = \mathbb{Z}/n\mathbb{Z}$ à la section 3, chap. III. Contentons-nous d'énoncer les premières propriétés du produit de convolution.

Proposition 4.11. *Le produit de convolution est commutatif, associatif, et l'application* $(f_1, f_2) \mapsto f_1 * f_2$ *est bilinéaire. On munit ainsi l'espace vectoriel* $\mathbb{C}[G]$ *d'une structure d'algèbre.*

Démonstration. La commutativité se vérifie aisément en faisant le changement de variable $h' = h^{-1}g$ dans la somme de l'équation (4.6). L'associativité peut être démontrée à la main, ou en utilisant le théorème 4.15. Le reste est sans difficulté. □

Avant de continuer, voyons « graphiquement » ce que donne un produit de convolution. La figure 1.3 montre le produit de convolution avec elle même d'une fonction « porte », sur $\mathbb{Z}/16\mathbb{Z}$. En abscisse, on a noté $\{0,\ldots,15\}$ des représentants de $\mathbb{Z}/16\mathbb{Z}$. C'est la pre-

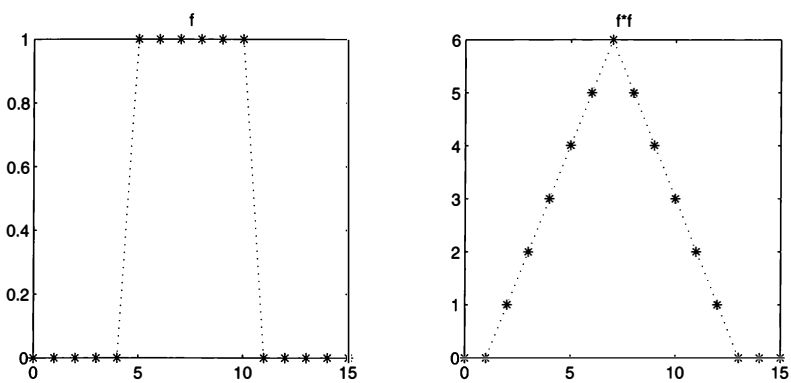

FIG. 1.3 – *Exemple de calcul de convolution*

mière fois que l'on aborde ce genre de figures. Celles-ci peuvent être un peu déroutantes et les résultats ne sont pas nécessairement évidents. Il y a plusieurs possibilités.

- On peut faire le calcul à la main, et vérifier que l'on obtient bien une fonction « triangle ».
- On peut attendre la section 3, chap. III, qui étudie en détail la convolution cyclique discrète. Nous serons alors en mesure de faire les calculs avec MATLAB, en utilisant l'algorithme FFT.
- On peut lire la section 5, chap. IV, qui explique le lien entre le calcul de convolution et la multiplication de polynômes modulo $X^n - 1$.

Le fait que le produit de convolution soit défini par extension du produit des éléments de G nous permet d'énoncer la proposition suivante.

Proposition 4.12 (Morphisme d'algèbre). *Soit* $\rho : G \to \mathbb{C}^*$ *un morphisme de groupe. Il existe une unique façon de l'étendre en un morphisme d'algèbre* $\widetilde{\rho} : \mathbb{C}[G] \to \mathbb{C}$.

Démonstration. En effet, la construction de $\mathbb{C}[G]$ nous dit que $\widetilde{\rho}$ est uniquement déterminé par la donnée des valeurs de $\widetilde{\rho}(\delta_g)$, pour $g \in G$. Or dans l'identification de G comme base canonique de $\mathbb{C}[G]$, on a $\widetilde{\rho}(\delta_g) = \rho(g)$, ce qui montre l'unicité de la construction. Il suffit alors de montrer que le morphisme construit est bien un morphisme d'algèbre. Par définition du produit de convolution, on peut se contenter de montrer la conservation du produit sur les éléments $\{\delta_g\}_{g\in G}$, ce qui est équivalent au fait que ρ soit un morphisme de groupe. □

Cette proposition nous dit qu'il y a une correspondance parfaite entre les morphismes de groupes de G dans $\mathbb{C}^*$ et les morphismes d'algèbres de $\mathbb{C}[G]$ dans $\mathbb{C}$.

***Remarque* 4.13. (Interprétation probabiliste).** Le produit de convolution, qui a été introduit comme l'extension par linéarité d'une opération de groupe, possède une interprétation probabiliste très importante. Soient X et Y deux variables aléatoires *indépendantes* à valeurs dans un groupe fini commutatif G. On note P_X et P_Y les distributions de probabilité correspondantes, c'est-à-dire $\forall g \in G, P_X(g) = \mathbb{P}(\{X = g\})$. Le résultat fondamental est que la distribution de probabilité de la variable aléatoire $X + Y$ est le produit de convolution des distributions de X et Y. Ceci s'écrit donc $P_{X+Y} = P_X * P_Y$. Ce théorème s'étend aux variables continues (à valeurs dans $\mathbb{R}$) et discrètes (à valeurs dans $\mathbb{Z}$), à condition d'utiliser le produit de convolution adéquat. Ce résultat est très simple à montrer (le lecteur peut en faire la vérification immédiate), et on pourra consulter [55] sur les applications du produit de convolution en probabilité (dans le cadre continu et discret). Les exercices I.9 et I.10 étudient l'utilisation de la transformée de Fourier sur un groupe fini pour résoudre des problèmes de probabilité.

4.3 Convolution et transformée de Fourier

Soit G un groupe fini commutatif d'ordre n. La proposition suivante montre que la définition de la transformée de Fourier est en fait très naturelle.

Proposition 4.14 (Morphisme d'algèbre). *Soit $\chi \in \widehat{G}$. L'application*

$$\mathscr{F}_\chi : \begin{cases} \mathbb{C}[G] & \longrightarrow & \mathbb{C} \\ f & \longmapsto & \widehat{f}(\chi) \end{cases}$$

correspond à l'unique façon d'étendre le morphisme de groupe χ en un morphisme d'algèbre.

Démonstration. L'unicité résulte directement de la proposition 4.12. Il ne reste plus qu'à montrer sur les éléments δ_g que $\mathscr{F}_\chi$ correspond à χ, ce qui est trivial :

$$\forall g \in G, \quad \mathscr{F}_\chi(\delta_g) = \sum_{x\in G} \delta_g(x)\chi(x) = \chi(g). \qquad \square$$

Cette propriété, qui justifie à posteriori l'introduction de la transformée de Fourier, est d'une importance capitale, et l'on peut la résumer sous la forme du théorème de convolution suivant.

Théorème 4.15 (Convolution et transformée de Fourier). *Pour f et g deux fonctions de $\mathbb{C}[G]$ on a*

$$\widehat{f * g} = \widehat{f} \cdot \widehat{g} \quad \text{et} \quad c_{f*g} = |G| c_f \cdot c_g, \tag{4.7}$$

où l'on a noté $\cdot$ le produit terme à terme de deux fonctions. La transformée de Fourier $\mathscr{F}$ est donc un isomorphisme d'algèbre de $(\mathbb{C}[G], *)$ dans $(\mathbb{C}[\widehat{G}], \cdot)$.

Cette propriété de convolution est sans doute la propriété de la transformée de Fourier la plus utilisée, puisqu'elle permet de changer un problème assez complexe (le calcul d'une convolution de deux fonctions) en un problème plus simple (le calcul du produit terme à terme). Les occurrences de ce principe de simplification seront nombreuses à travers le livre, qu'il s'agisse d'études théoriques (calcul de déterminant circulant, formule de Poisson, etc.) où bien plus appliquées (filtrage, produit de grands entiers, décodage de codes correcteurs, etc.).

5 Exercices

Exercice I.1 (Déterminant circulant). Soit G un groupe cyclique. On fixe $f \in \mathbb{C}[G]$. On souhaite calculer le déterminant de l'endomorphisme

$$\Phi^f \left\{ \begin{array}{ccc} \mathbb{C}[G] & \longrightarrow & \mathbb{C}[G] \\ u & \longmapsto & f * u \end{array} \right. .$$

On rencontrera souvent ce type d'applications, notamment au paragraphe 2.1, chap. IV, où il sera question de filtrage.

1. Expliquer pourquoi les éléments $\chi \in \widehat{G}$ sont les vecteurs propres de Φ^f. Quelles sont les valeurs propres associées ?
2. Quelle est la matrice A de l'endomorphisme Φ^f dans la base $\{\delta_g\}_{g\in G}$ de $\mathbb{C}[G]$? Déduire de la question précédente une expression de $\det(A)$.
3. En choisissant judicieusement le groupe G et l'application f, montrer que l'on a

$$\det \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_{n-1} \\ a_{n-1} & a_0 & a_1 & \dots & a_{n-2} \\ \vdots & \vdots & \vdots & & \vdots \\ a_1 & a_2 & a_3 & \dots & a_0 \end{pmatrix} = \prod_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} a_j \omega^{ij} \right),$$

   où $(a_0, \dots, a_{n-1}) \in \mathbb{C}^n$, et $\omega \overset{\text{déf.}}{=} e^{\frac{2i\pi}{n}}$ (Un tel déterminant est appelé *déterminant circulant*).
4. Après avoir lu le chapitre III consacré à la transformée de Fourier discrète et à l'algorithme FFT, proposer une implémentation rapide du calcul de déterminant circulant.

Exercice I.2 (Dual de SO(3)). On note $SO(3)$ le groupe des matrices 3×3 réelles, orthogonales, et de déterminant 1. Il correspond aux rotations de $\mathbb{R}^3$. On souhaite montrer que $SO(3)$ n'a pas de caractère non trivial.

1. Montrer que deux rotations de même angle sont conjuguées.
2. Soit χ un élément du dual de $SO(3)$. Pour $g \in SO(3)$, montrer que $\chi(g)$ ne dépend que de l'angle de g.
3. On note r_α la rotation d'angle α autour de $(1,0,0)$, et s_α la rotation d'angle α autour de $(0,1,0)$. On considère $t_\beta \overset{\text{déf.}}{=} r_\alpha s_\alpha^{-1}$. Montrer que t_β est une rotation d'un certain angle β, et que lorsque α parcours $[0,\pi]$, alors β fait de même.
4. En déduire que $\chi = 1$.

Exercice I.3 (Dénombrement de solutions). Soit G un groupe abélien fini, et une fonction $\varphi : G^n \to G$. Pour $h \in G$, on note $N(h)$ le nombre de n-uplets $(g_1, \ldots, g_n)$ tels que $\varphi(g_1, \ldots, g_n) = h$. Montrer que l'on a

$$N(h) = \frac{1}{|G|} \sum_{g_1 \in G} \cdots \sum_{g_n \in G} \sum_{\chi \in \widehat{G}} \chi\left(\varphi(g_1, \ldots, g_n)\right) \overline{\chi}(h).$$

Exercice I.4 (Fonctions indicatrices). Soit G un groupe abélien fini et $A \subset G$. On note f_A la fonction indicatrice de A.

1. Montrer que
$$\|f_A\|_2 = \sqrt{\frac{|A|}{|G|}} \qquad \text{et} \qquad \widehat{f_A}(\chi_0) = |A|,$$
où l'on a noté χ_0 le caractère trivial.
2. On suppose que $|A| \leqslant \frac{1}{2}|G|$. On définit
$$\Phi(A) \stackrel{\text{déf.}}{=} \max\left\{ |\widehat{f_A}(\chi)| \setminus \chi \in \widehat{G}, \chi \neq \chi_0 \right\}. \tag{5.1}$$
Montrer que l'on a
$$\sqrt{\frac{|A|}{2}} \leqslant \Phi(A) \leqslant |A|. \tag{5.2}$$
3. On se place dans le cas où $|A| > \frac{1}{2}|G|$. Montrer que l'on a $\Phi(A) = \Phi(G \backslash A)$, où l'on a noté $G \backslash A$ le complémentaire de A dans G. En déduire une minoration de $\Phi(A)$ similaire à (5.2).
4. Montrer que si α est un automorphisme de G, alors $\Phi(\alpha(A)) = \Phi(A)$.

Intuitivement, plus $\Phi(A)$ est proche de la borne inférieure, plus les éléments de A sont distribués uniformément dans G. L'exercice I.9 étudie et quantifie ce phénomène. On peut voir une analogie avec l'étude de la transformée de Fourier d'une fonction continue : plus les coefficients de Fourier hautes fréquences sont faibles, plus la fonction est « lisse ». Ces fonctions indicatrices seront utilisées au paragraphe 4.2, chap. VI dans le cadre où A est employé comme code correcteur. Une fois encore, ce sont les propriétés spectrales de f_A qui seront utilisées pour étudier la « géométrie » de l'ensemble A.

Exercice I.5 (Equations sur un groupe abélien fini). Cet exercice utilise les notations et résultats de l'exercice I.4. Il est tiré de l'article de synthèse de Babai [4].

1. On considère $A_1, \ldots, A_k \subset G$, et on étudie l'équation
$$x_1 + \cdots + x_k = a \qquad \text{avec} \quad x_i \in A_i, \quad i = 1, \ldots, k. \tag{5.3}$$
Expliquer comment on peut se ramener au cas $a = 0$. On note N le nombre de solutions de (5.3), dans le cas $a = 0$. En utilisant le résultat de l'exercice I.3, montrer que
$$N = \frac{1}{|G|} \sum_{\chi \in \widehat{G}} \sum_{x_i \in A_i} \chi(x_1 + \cdots + x_k) = \frac{|A_1| \cdots |A_k|}{|G|} + R,$$
avec
$$R \stackrel{\text{déf.}}{=} \frac{1}{|G|} \sum_{\chi \neq \chi_0} \prod_{i=1}^{k} \widehat{f_{A_i}}(\chi).$$

2. On suppose que $k = 3$. Montrer que

$$|R| \leqslant \frac{\Phi(A_3)}{|G|} \sum_{\chi \in \widehat{G}} |\widehat{f_{A_1}}(\chi)||\widehat{f_{A_2}}(\chi)| \leqslant \Phi(A_3)\sqrt{|A_1||A_2|},$$

où Φ est défini à l'équation (5.1) (on pourra utiliser l'inégalité de Cauchy-Schwartz). Montrer, en utilisant le résultat de l'exercice I.4, question 4., que ceci est encore valable quand $a \neq 0$.

3. En déduire que si

$$\frac{\Phi(A_3)}{|A_3|} < \frac{\sqrt{|A_1||A_2|}}{|G|},$$

alors l'équation $x_1 + x_2 + x_3 = a$, avec $x_i \in A_i$, $i = 1, 2, 3$, a au moins une solution.

Ce résultat surprenant nous dit donc que si au moins l'un des trois ensembles A_i est bien réparti, et que si les trois ensembles sont suffisamment grands, alors l'équation considérée a au moins une solution. L'exercice II.2 applique ce résultat sur $G = \mathbb{F}_q$ pour étudier le théorème de Fermat sur les corps finis.

Exercice I.6 (Groupe de Heisenberg). Soit G un groupe abélien fini. On note $\mathscr{U}$ le groupe des nombres complexes de module 1. On note $\mathscr{H}(G) \overset{\text{déf.}}{=} \mathscr{U} \times G \times \widehat{G}$ muni de l'opération

$$(\lambda, x, \chi) \cdot (\mu, y, \tau) = (\lambda\mu\tau(x), xy, \chi\tau)$$

le groupe de Heisenberg associé à G.

1. Montrer que l'on définit bien ainsi une structure de groupe. En particulier, quel est l'élément neutre et quel est l'inverse d'un élément générique $(\lambda, x, \chi) \in \mathscr{H}(G)$?
2. Montrer que l'on peut définir une action de $\mathscr{H}(G)$ sur $\mathbb{C}[G]$ en posant

$$\forall f \in \mathbb{C}[G], \forall (\lambda, x, \chi) \in \mathscr{H}(G), \quad (\lambda, x, \chi) \cdot f : z \mapsto \lambda\chi(z) f(xz).$$

3. Pour $(\lambda, x, \chi) \in \mathscr{H}(G)$ et $f \in \mathbb{C}[G]$, on définit respectivement les opérateurs de dilatation, translation, et modulation par

$$D_\lambda(f)(z) = \lambda f(z), \qquad T_x(f)(z) = f(xz), \qquad M_\chi(f)(z) = \chi(z) f(z).$$

Exprimer l'action de $\mathscr{H}(G)$ sur $\mathbb{C}[G]$ en fonction de ces trois opérateurs. Comment se comportent ces trois opérateurs vis-a-vis de la transformée de Fourier définie à l'équation (4.1)? Quel liens y a-t-il avec la transformée de Fourier continue sur $\mathbb{R}$?

4. En identifiant canoniquement G à $\widehat{\widehat{G}}$ comme décrit au paragraphe 2.3, comment le produit sur $\mathscr{H}(\widehat{G}) = \mathscr{U} \times \widehat{G} \times G$ est-il défini ? Comment définir une action de $\mathscr{H}(\widehat{G})$ sur $\mathbb{C}[\widehat{G}]$?
5. On définit la fonction

$$\alpha : \begin{cases} \mathscr{H}(G) & \longrightarrow & \mathscr{H}(\widehat{G}) \\ (\lambda, x, \chi) & \longmapsto & (\lambda\chi^{-1}(x), \chi, x^{-1}) \end{cases}.$$

Montrer que α est un isomorphisme de groupes, et que l'on a

$$\forall f \in \mathbb{C}[G], \forall (\lambda, x, \chi) \in \mathscr{H}(G), \quad \mathscr{F}((\lambda, x, \chi) \cdot f) = \alpha(\lambda, x, \chi) \cdot \mathscr{F}(f),$$

où $\mathscr{F}$ désigne la transformée de Fourier définie à l'équation (4.1).

6. On suppose que $\Phi : \mathbb{C}[G] \to \mathbb{C}[G]$ commute avec l'action de $\mathscr{H}(G)$, c'est-à-dire que

$$\forall f \in \mathbb{C}[G], \forall (\lambda,x,\chi) \in \mathscr{H}(G), \quad \Phi((\lambda,x,\chi)\cdot f) = (\lambda,x,\chi)\cdot\Phi(f).$$

Montrer qu'il existe $r \in \mathbb{C}^*$ tel que $\forall f \in \mathbb{C}[G], \Phi(f) = rf$. On pourra raisonner sur la matrice de Φ exprimée dans la base $\{\delta_g\}_{g\in G}$ de $\mathbb{C}[G]$, ou alors utiliser le lemme de Schur 2.5, chap. VII. Que se passe-t-il si $\Phi : \mathbb{C}[G] \to \mathbb{C}[\widehat{G}]$ fait commuter les actions de $\mathscr{H}(G)$ et $\mathscr{H}(\widehat{G})$?

Exercice I.7 (Transformée de Fourier et orthogonalisation). On considère une fonction $f \in L^2(\mathbb{R})$. On note τ_r la translation de r sur $L^2(\mathbb{R})$, c'est-à-dire $\tau_r(f) = f(\cdot - r)$.

1. Montrer que la famille $\{\tau_n(f)\}_{n\in\mathbb{Z}}$ est orthormée si et seulement si

$$\text{p.p.t. } \omega \in \mathbb{R}, \quad \sum_{k\in\mathbb{Z}} |\widehat{f}(\omega+2k\pi)|^2 = 1,$$

où l'on a noté $\widehat{f} \in L^2(\mathbb{R})$ la transformée de Fourier de f, définie, pour les fonctions $f \in L^1(\mathbb{R})$ par

$$\text{p.p.t. } \omega \in \mathbb{R}, \quad \widehat{f}(\omega) \overset{\text{déf.}}{=} \int_{\mathbb{R}} f(x)e^{-i\omega x}dx,$$

et étendue par densité à $L^2(\mathbb{R})$ tout entier.

2. On suppose qu'il existe $A > 0$ tel que

$$\text{p.p.t. } \omega \in \mathbb{R}, \quad A \leqslant \sum_{k\in\mathbb{Z}} |\widehat{f}(\omega+2k\pi)|^2.$$

Montrer que si on note φ la fonction de $L^2(\mathbb{R})$ telle que

$$\text{p.p.t. } \omega \in \mathbb{R}, \quad \widehat{\varphi}(\omega) \overset{\text{déf.}}{=} \frac{\widehat{f}(\omega)}{\left(\sum_{k\in\mathbb{Z}} |\widehat{f}(\omega+2k\pi)|^2\right)^{1/2}},$$

alors la famille $\{\tau_n(\varphi)\}_{n\in\mathbb{Z}}$ est orthonormée (les égalités sont à considérer pour presque tout ω).

L'exercice suivant I.8 propose d'étudier le même problème d'orthogonalisation, mais dans le cadre d'un groupe abélien fini.

Exercice I.8 (Orthogonalisation sur un groupe abélien). Cet exercice est inspiré de l'article de BERNARDINI et KOVACEVIC [7]. Soit V un $\mathbb{C}$-espace vectoriel de dimension n, muni d'un produit hermitien $\langle\cdot,\cdot\rangle$. Soit G un groupe abélien fini de transformations unitaires de V, et soit $b \in V$. On dit que b est orthonormé pour l'action de G sur V si l'ensemble $G_b \overset{\text{déf.}}{=} \{Ab \setminus A \in G\}$ est orthonormé. Ceci signifie que

$$\forall (x,y) \in G_b^2, \quad \langle x,y\rangle = \delta_x^y.$$

1. On note

$$\psi_b : \left\{ \begin{array}{ccc} G & \longrightarrow & \mathbb{C} \\ A & \longmapsto & \langle Ab,b\rangle \end{array} \right. .$$

Montrer que b est orthonormé pour l'action de G si et seulement si $\widehat{\psi_b} \equiv 1$, c'est-à-dire si et seulement si $\forall \chi \in \widehat{G}$, $|G|\left\langle \mathscr{U}_\chi b,b\right\rangle = 1$, où l'on a noté

$$\forall \chi \in \widehat{G}, \quad \mathscr{U}_\chi \overset{\text{déf.}}{=} \frac{1}{|G|}\sum_{A\in G} \chi(A)A \quad \in \mathscr{L}(V,V).$$

2. Montrer que les opérateurs $\mathscr{U}_\chi$ sont des projecteurs orthogonaux, et qu'ils sont deux à deux orthogonaux, c'est-à-dire

$$\forall(\chi_1,\chi_2)\in\widehat{G}^2,\quad \mathscr{U}_{\chi_1}\mathscr{U}_{\chi_2}=\begin{cases}0 & \text{si } \chi_1\neq\chi_2\\ \mathscr{U}_{\chi_1} & \text{si } \chi_1=\chi_2\end{cases},$$

et $\mathscr{U}_\chi^*=\mathscr{U}_\chi$ (A^* désigne l'adjoint de A).

3. On suppose que $\widehat{\psi_b}$ ne s'annule pas. On note

$$\widetilde{b}\overset{\text{déf.}}{=}\sum_{\chi\in\widehat{G}}\frac{1}{\sqrt{\widehat{\psi_b}(\chi)}}\mathscr{U}_\chi b,$$

où $\sqrt{\widehat{\psi_b}(\chi)}$ désigne l'une des deux racines possibles. Montrer que $\widetilde{b}$ est orthonormé pour l'action de G.

4. Quel rapprochement peut-on faire avec l'exercice I.7 ?

Pour une étude plus poussée de cette méthode dans le cas des groupes cycliques $\mathbb{Z}/n\mathbb{Z}$, on pourra regarder l'exercice III.11.

Exercice I.9 (Répartition de probabilité). Soit G un groupe abélien fini, et $P:G\to\mathbb{R}^+$ la fonction de répartition d'une loi de probabilité sur G, ce qui signifie que $\sum_{g\in G}P(g)=1$. On note U la répartition uniforme, c'est-à-dire $U(g)=\frac{1}{|G|}$ pour tout $g\in G$. On note χ_0 le caractère trivial de G.

1. Calculer $\widehat{P}(\chi_0)$ ainsi que $\widehat{U}(\chi)$, pour $\chi\in\widehat{G}$. En déduire une expression de $\|P-U\|_2^2$.
2. Montrer que l'on a

$$\forall g\in G,\quad \left|P(g)-\frac{1}{|G|}\right|^2\leqslant\frac{1}{|G|}\sum_{\chi\neq\chi_0}|\widehat{P}(\chi)|^2.$$

En quelque sorte, la quantité $\|P-U\|_2^2$ mesure l'uniformité de la distribution P, et comme nous l'avons déjà vu pour les fonctions caractéristiques (exercice I.4), ceci est caractérisé par les coefficients de Fourier $\widehat{P}(\chi)$, pour $\chi\neq\chi_0$.

Exercice I.10 (Marche aléatoire). On considère une marche aléatoire sur $\mathbb{Z}/n\mathbb{Z}$ construite comme suit. La variable aléatoire $X_k\in\mathbb{Z}/n\mathbb{Z}$ désigne une position sur le cercle $\mathbb{Z}/n\mathbb{Z}$ à l'instant $k\in\mathbb{N}$. Le déplacement entre l'instant k et $k+1$ est donné par une probabilité de transition $p_{i,j}\overset{\text{déf.}}{=}\mathbb{P}(X_{k+1}=j\,|\,X_k=i)$.

1. On note $p^{(k)}:G\to[0,1]$ la répartition de probabilité de X_k, c'est-à-dire que pour $0\leqslant i<n$, $p^{(k)}(i)=\mathbb{P}(X_k=i)$. On peut aussi noter $p^{(k)}$ sous la forme d'un vecteur de taille n. Montrer que $p^{(k+1)}=Pp^{(k)}$, où on a noté P la matrice de transition $\{p_{i,j}\}_{0\leqslant i,j\leqslant n-1}$. En déduire que $p^{(k)}=P^kp^{(0)}$, où $p^{(0)}=\{1,0,\dots,0\}$ est la distribution initiale.
2. Soit $0<p<1$. On considère la marche aléatoire la plus simple, donnée par

$$\begin{cases}p_{i,i-1}=p, & p_{i,i+1}=1-p,\\ p_{i,j}=0 \quad \text{si} & i\notin\{i-1,i+1\}\end{cases}.$$

Montrer que l'on a alors $Px=\nu*x$, où $*$ désigne le produit de convolution et $\nu=\{0,p,0,\dots,0,1-p\}$. Comment peut on retrouver ce résultat en considérant des sommes de variables aléatoires, et en utilisant la remarque 4.13 ?

3. Exprimer la transformée de Fourier $\widehat{p^{(k)}}$ à partir de $\widehat{p^{(0)}}$. En déduire que si n est impair, alors
$$p^{(k)} \xrightarrow[k\to+\infty]{} u \qquad \text{où} \qquad u \overset{\text{déf.}}{=} \{1/n,\dots,1/n\}.$$
Que se passe-t-il si n est pair ?
4. Généraliser ce résultat à une marche aléatoire invariante par translation, c'est-à-dire telle que
$$p_{i,j} = v_{j-i} \qquad \text{avec} \qquad v \in ([0,1])^n.$$

La figure 1.4 montre la progression de la répartition de probabilité $p^{(k)}$ dans le cas de deux marches aléatoires. On voit que la deuxième marche converge plus rapidement vers la répartition uniforme (on pourra faire le lien avec la taille des coefficients de Fourier).

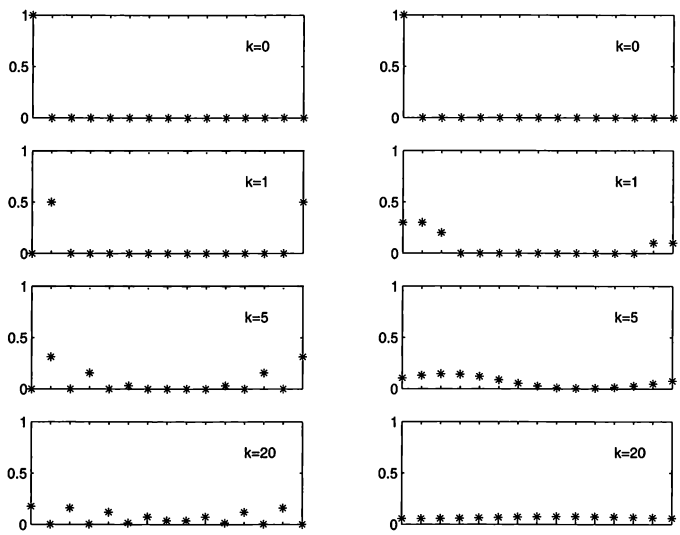

FIG. 1.4 – *Marches aléatoires pour les probabilités de transition* $\{0,1/2,0,\dots,0,1/2\}$ *(gauche) et* $\{0.3,0.3,0.2,0,\dots,0,0.1,0.1\}$ *(droite)*

Exercice I.11 (Principe d'incertitude discret). Soit G un groupe fini, et $f \in \mathbb{C}[G]$ une fonction non nulle. On souhaite montrer que l'on a
$$|\operatorname{Supp}(f)| \times |\operatorname{Supp}(\widehat{f})| \geqslant |G|, \tag{5.4}$$
où $|\operatorname{Supp}(f)|$ désigne la taille du support de f.

1. On considère, tout d'abord, le groupe $G = \mathbb{Z}/n\mathbb{Z}$. Montrer que si f a p éléments non nuls, alors, $\widehat{f}$ ne peut pas avoir p zéros consécutifs. En déduire l'équation (5.4). Ce résultat a été démontré en premier par DONOHO et STARK [28].
2. On revient au cas général d'un groupe abélien fini G.
On note $M \overset{\text{déf.}}{=} \sup\{f(x) \setminus x \in G\}$ et $\|f\|_2^2 \overset{\text{déf.}}{=} \langle f, f\rangle$. Montrer que
$$\|f\|_2^2 \leqslant \frac{M^2}{|G|}|\operatorname{Supp}(f)| \qquad \text{et} \qquad M \leqslant \frac{1}{|G|}\sum_{\chi\in\widehat{G}} |\widehat{f}(\chi)|.$$

En utilisant l'inégalité de Cauchy-Schwartz, en déduire que

$$M^2 \leqslant \frac{|\mathrm{Supp}(\widehat{f})|}{|G|}\|\widehat{f}\|_2^2, \qquad \text{puis,} \qquad M^2 \leqslant \|f\|_2^2 |\mathrm{Supp}(\widehat{f})|.$$

En conclure que

$$\|f\|_2^2 \leqslant \frac{\|f\|_2^2}{|G|}|\mathrm{Supp}(f)| \times |\mathrm{Supp}(\widehat{f})|.$$

3. Soit $H \subset G$, un sous-groupe et $f_H \in \mathbb{C}[G]$, sa fonction caractéristique. Montrer que l'on a

$$\widehat{f_H} = |H| f_{H^\sharp},$$

où l'on a noté $H^\sharp \subset \widehat{G}$ l'orthogonal de H, comme défini en 3.1, chap. II. En déduire que la fonction f atteint la borne de l'équation (5.4). En fait, on peut montrer que toute fonction qui atteint cette borne est reliée à une telle fonction f_H par une translation et une dilatation. Ceci est démontré dans l'article de MATUSIAK [53].

Cette conclusion sur la localisation des supports temporels et fréquentiels peut paraître négative au premier abord: elle nous interdit de construire des signaux bien localisés à la fois en temps et en fréquence. Cependant, on peut l'utiliser à profit, par exemple pour construire des codes correcteurs efficaces, comme le montrera la proposition 3.22, chap. VI.

Chapitre II
Applications de la dualité sur un groupe fini

> Actually Gauss is often called the greatest mathematician of all time. So it's nice to be able to understand at least one of his discoveries.
>
> R. GRAHAM, O. PATASHNIK, D.E. KNUTH [37] (1994)

Pour mieux comprendre la théorie des caractères sur un groupe commutatif, il faut la mettre en application dans des situations où elle est vraiment utile. Le but de ce chapitre est donc de comprendre, grâce à des exemples, pourquoi cette théorie est si puissante. Nous allons ainsi démontrer sans trop d'efforts des formules qui peuvent paraître complexes, du moins pour quelqu'un qui les aborde pour la première fois. Le meilleur exemple est la formule de réciprocité quadratique, dont la démonstration repose sur l'utilisation de deux types de caractères.

1 Sommes de Gauss

> [Parlant du théorème 1.20]
> Théorèmes remarquables par leur élégance. [...] Ces théorèmes conservent toute leur élégance, ou plutôt en acquièrent encore davantage, lorsque n est un nombre composé quelconque; mais nous sommes forcés de supprimer ces recherches qui demanderaient trop de développements, et de les réserver pour une autre occasion.
>
> C.F. GAUSS *Disquisitiones Arithmeticae* (1807)

L'idée qui sous-tend l'exposé de ce paragraphe est très simple. Il s'agit de mieux comprendre les *corps finis*, et plus précisément, de percevoir de façon plus claire comment les deux structures qui composent un tel corps (groupe multiplicatif et groupe additif) peuvent co-exister, et s'influencer mutuellement. L'outil principal sera bien sûr la *dualité* sur un groupe abélien, et l'idée à développer sera la combinaison de deux types de caractères. C'est justement pour combiner ces caractères que l'on va introduire la notion de *somme de Gauss*, qui est présentée au paragraphe 1.3.

La principale référence pour cet exposé est le livre de LIDL et NIEDERREITER [48], qui constitue une véritable encyclopédie des corps finis. On pourra aussi lire avec beaucoup

d'intérêt le très bel exposé de LANGEVIN [44], qui relie de nombreux sujets connexes, tels les sommes de Gauss, les codes correcteurs, et la théorie des nombres.

1.1 Résidus quadratiques

Avant de se lancer dans l'étude de la dualité sur un corps fini, donnons un exemple simple qui va justifier l'introduction de la théorie des caractères. Considérons l'équation suivante, qui est à inconnues entières :

$$x^2 = 2y^2 + 7k \qquad \text{avec} \quad (x,y,k) \in \mathbb{Z}^3. \tag{1.1}$$

Pour la résoudre de façon quasi triviale, il suffit de la remplacer par son homologue modulo le nombre premier 7, et d'utiliser la structure de corps de $\mathbb{Z}/7\mathbb{Z}$ qui nous autorise à effectuer des divisions :

$$(x/y)^2 = 2 \mod 7,$$

pour $x \neq 0$. Il ne reste plus qu'à dresser la liste des carrés de $\{0,1,\ldots,6\}$, pris modulo 7. On obtient facilement $\{0,1,4,2,2,4,1\}$. On conclut donc que les solutions non nulles de l'équation sont données par

$$\frac{x}{y} = 3 + 7k' \quad \text{et} \quad \frac{x}{y} = 4 + 7k', \quad \text{pour } k' \in \mathbb{Z}.$$

De façon évidente, maintenant que l'on connaît les racines carrées de 2 dans $\mathbb{Z}/7\mathbb{Z}$, on peut considérer la factorisation suivante de l'équation (1.1) :

$$(x-3y)(x-4y) = 0 \mod 7,$$

ce qui conduit bien sûr au même résultat.

Cette démarche naïve est évidemment à proscrire dans le cas d'étude de grands nombres. On est amené à considérer, pour un nombre premier p le *symbole de Legendre*, défini de la manière suivante :

$$\forall n \in \mathbb{Z}, \quad \left(\frac{n}{p}\right) = \begin{cases} 0 & \text{si } n \text{ est divisible par } p \\ 1 & \text{si } n \text{ est un carré modulo } p \\ -1 & \text{si } n \text{ n'est pas un carré modulo } p \end{cases}$$

Il est évident que l'on peut restreindre l'étude de ce symbole aux seuls éléments de $\mathbb{Z}/p\mathbb{Z}$, que l'on note usuellement $\mathbb{F}_p$. La remarque capitale pour la suite est que l'application

$$\eta : \begin{cases} \mathbb{F}_p^* & \longrightarrow & \{-1,1\} \\ n & \longmapsto & \left(\frac{n}{p}\right) \end{cases} \tag{1.2}$$

est un caractère du groupe multiplicatif $\mathbb{F}_p^*$, puisque l'on a

$$\left(\frac{n_1 n_2}{p}\right) = \left(\frac{n_1}{p}\right)\left(\frac{n_2}{p}\right).$$

Cette propriété résulte directement du lemme suivant.

Lemme 1.1 (Formule d'Euler). *Soit p un nombre premier impair. Un élément $x \in \mathbb{F}_p^*$ est un carré si et seulement si $x^{\frac{p-1}{2}} = 1$. Un élément $x \in \mathbb{F}_p^*$ n'est pas un carré si et seulement si $x^{\frac{p-1}{2}} = -1$. En conséquence, on a la formule d'Euler*

$$\forall x \in \mathbb{F}_p^*, \quad \left(\frac{x}{p}\right) = x^{\frac{p-1}{2}}.$$

Démonstration. On considère le groupe multiplicatif $\mathbb{F}_p^*$.Comme p est impair, le morphisme $x \mapsto x^2$ a pour noyau le sous-groupe $\{1, -1\}$, de sorte que les éléments qui sont des carrés modulo p forment un sous-groupe de $\mathbb{F}_p^*$ de cardinal $\frac{p-1}{2}$.
Si $x = y^2$, alors $x^{\frac{p-1}{2}} = y^{p-1} = 1$. Donc les $\frac{p-1}{2}$ résidus quadratiques modulo p sont tous racines du polynôme $X^{\frac{p-1}{2}} - 1$, qui ne saurait avoir plus de $\frac{p-1}{2}$ racines. Les résidus sont donc exactement ces racines, c'est-à-dire les éléments x tels que $x^{\frac{p-1}{2}} = 1$.
Pour conclure la démonstration, il suffit de constater que $\left(x^{\frac{p-1}{2}}\right)^2 = 1$ implique que $x^{\frac{p-1}{2}}$ est un élément de $\{-1, 1\}$. Les non résidus quadratiques sont caractérisés par $x^{\frac{p-1}{2}} = -1$, ce qui achève de démontrer la formule d'Euler. □

Le but de ce chapitre est de démontrer la propriété importante de ces caractères de Legendre, que l'on appelle formule de réciprocité quadratique. Elle relie le fait d'être un carré modulo p à celui d'être un carré modulo q. En son temps, *Euler* avait déjà remarqué, en calculant à la main de nombreux cas, que pour deux premiers impairs distincts p et q, on a $\left(\frac{p}{q}\right) = \left(\frac{q}{p}\right)$, sauf dans le cas où p et q sont tous deux de la forme $4k-1$. Ce résultat est cependant loin d'être évident, et il fallut attendre *Gauss* pour obtenir une preuve complète de ce résultat.

Grâce à cette formule, nous allons pouvoir calculer facilement le caractère de Legendre, en appliquant successivement des inversions du symbole $\left(\frac{q}{p}\right)$ (avec la formule de réciprocité) et des réductions de q modulo p (car le symbole ne dépend que de la classe de n modulo p).

1.2 Caractères additifs et multiplicatifs

Dans le chapitre précédent, nous nous sommes intéressés aux groupes finis commutatifs. Nous allons maintenant imposer une structure plus rigide à notre groupe, puisque nous allons nous intéresser à un corps fini $\mathbb{F}_q$, avec $q = p^r$ où p est un nombre premier. C'est un corps de caractéristique p, et il peut être vu comme un espace vectoriel de dimension finie r sur son corps premier $\mathbb{F}_p$. Une description plus détaillée des corps finis sera faite à la section 1, chap. VI, lors de la construction d'une transformée de Fourier à valeurs dans un corps fini.

Sur notre corps, on peut dégager deux structures de groupe. Tout d'abord on peut considérer $\mathbb{F}_q$ comme un groupe additif (en fait un espace vectoriel sur $\mathbb{F}_p$). Ensuite, on peut aussi considérer le groupe multiplicatif $\mathbb{F}_q^* \overset{\text{déf.}}{=} \mathbb{F}_q - \{0\}$, qui est un groupe cyclique d'ordre $q-1$. Ceci conduit à considérer deux types de caractères.

Définition 1.2 (Caractères additifs et multiplicatifs). Les éléments de $\widehat{\mathbb{F}_q}$ sont appelés *caractères additifs*. Ce sont donc les morphismes

$$\psi : (\mathbb{F}_q, +) \longrightarrow (\mathbb{C}^*, *).$$

Les éléments de $\widehat{\mathbb{F}_q^*}$ sont appelés *caractères multiplicatifs*. Ce sont donc les morphismes

$$\chi : (\mathbb{F}_q^*, *) \longrightarrow (\mathbb{C}^*, *).$$

Les caractères les plus simples à déterminer sont les caractères multiplicatifs c'est-à-dire les éléments de $\widehat{\mathbb{F}_q^*}$. En effet, le groupe $\mathbb{F}_q^*$ est cyclique. Soit donc ζ un générateur de ce

groupe, de sorte que l'on ait $\mathbb{F}_q^* = \{1, \zeta, \zeta^2, \ldots, \zeta^{q-2}\}$. On peut alors énumérer les $q-1$ caractères multiplicatifs

$$\forall j = 0, \ldots, q-1, \quad \chi_j : \begin{cases} \mathbb{F}_q^* & \longrightarrow & \mathbb{F}_q^* \\ \zeta^k & \longmapsto & e^{\frac{2i\pi}{q-1}jk} \end{cases}.$$

On obtient ainsi une description complète du groupe dual $\widehat{\mathbb{F}_q^*}$, et on constate bien sûr que l'on a $\widehat{\mathbb{F}_q^*} \simeq \mathbb{F}_q^*$. Cette description n'est pas canonique, dans le sens où elle nécessite le choix (arbitraire) d'une racine primitive ζ.

En ce qui concerne le groupe additif $\mathbb{F}_q$, la situation est un peu plus complexe, puisque ce groupe n'est pas cyclique. Cependant, en tant que groupe additif, $\mathbb{F}_q$ est en fait isomorphe au groupe produit $(\mathbb{Z}/p\mathbb{Z})^r$. Comme $\mathbb{Z}/p\mathbb{Z}$ est un groupe (additif) cyclique, il va être relativement aisé de dresser la liste des caractères additifs de $\mathbb{F}_q$. Cependant, dans le but de produire le moins d'efforts possible, et de simplifier la description du dual, nous allons introduire la notion suivante.

Définition 1.3 (Application trace). Soit K un corps fini, contenant un sous-corps k de cardinal s. On note $t \overset{\text{déf.}}{=} [K : k]$ la dimension de K en tant que k-espace vectoriel, de sorte que $|K| = s^t$. Soit $\alpha \in K$. On définit l'application *trace* de K sur k de la façon suivante :

$$\mathrm{Tr}_{K/k}(\alpha) \overset{\text{déf.}}{=} \alpha + \alpha^s + \cdots + \alpha^{s^{t-1}}.$$

Dans la suite, nous allons nous intéresser aux corps $k = \mathbb{F}_p$ et $K = \mathbb{F}_q$ (on fixe donc $s = p$ et $t = r$). Lorsqu'il n'y aura pas de risque de confusion, on notera simplement Tr à la place de $\mathrm{Tr}_{\mathbb{F}_q/\mathbb{F}_p}$.

On rappelle quelques propriétés importantes des corps finis, que l'on trouvera démontrées dans le livre de PERRIN [58].

Proposition 1.4 (Propriétés des corps finis). *Soit K un corps fini de caractéristique p.*

(i) *Soit k un sous-corps de K de cardinal s Un élément $x \in K$ appartient à k si et seulement si $x^s = x$.*

(ii) *L'application $\Phi : x \mapsto x^p$ est un morphisme, appelé morphisme de Frobenius. Les itérés $\Phi^k : x \mapsto x^{p^k}$ sont aussi des morphismes.*

Voyons les principales propriétés de cette application.

Proposition 1.5 (Propriétés de la trace). *La trace de K sur k est une forme k-linéaire non nulle à valeurs dans k.*

Démonstration. La première chose à montrer est que pour $\alpha \in K$, on a $\mathrm{Tr}_{K/k}(\alpha) \in k$. Il suffit de montrer que l'on a $x^s = x$. En utilisant la linéarité du morphisme $x \mapsto x^s$ (qui est un itéré du Frobenius), il vient

$$\mathrm{Tr}_{K/k}(\alpha)^s = \alpha^s + \alpha^{s^2} + \cdots + \alpha^{s^t}.$$

Comme K^* est un groupe de cardinal $s^t - 1$, on a $\forall \alpha \in K^*$, $\alpha^{s^t-1} = 1$, d'où le résultat souhaité, puisque $\forall \alpha \in K$, $\alpha^{s^t} = \alpha$.

En utilisant le morphisme de Frobenius et le fait que $\lambda^s = \lambda$ pour un scalaire $\lambda \in k$, il est clair que l'application trace est bien k-linéaire. Il reste à montrer qu'elle n'est pas triviale, c'est-à-dire qu'il existe un élément $\alpha \in K$ tel que $\mathrm{Tr}_{K/k}(\alpha) \neq 0$. Or, si $\mathrm{Tr}_{K/k}(\alpha) = 0$, ceci signifie que α est racine du polynôme

$$P(X) \overset{\text{déf.}}{=} X + X^s + \cdots + X^{s^{t-1}},$$

qui est de degré s^{t-1}. Ce polynôme a donc au plus s^{t-1} racines, et comme K a s^t éléments, il existe bien un certain $\alpha \in K$ tel que $P(\alpha) \neq 0$. □

On peut maintenant fournir une description complète des caractères additifs de $\mathbb{F}_q$. Pour ce faire, on introduit un caractère dit *canonique*, dans le sens où il est indépendant de la façon dont on construit le corps $\mathbb{F}_q$.

Définition 1.6 (Caractère canonique). On définit le caractère additif *canonique* ψ_1, élément de $\widehat{\mathbb{F}_q}$ par

$$\psi_1 : \begin{cases} \mathbb{F}_q & \longrightarrow & \mathbb{C}^* \\ x & \longmapsto & e^{\frac{2i\pi}{p}\mathrm{Tr}(x)} \end{cases}.$$

Le théorème suivant nous explicite la construction des autres caractères à partir de ce caractère canonique.

Proposition 1.7. *Soit, pour $a \in \mathbb{F}_q$, l'application*

$$\psi_a : \begin{cases} \mathbb{F}_q & \longrightarrow & \mathbb{C}^* \\ x & \longmapsto & \psi_1(ax) \end{cases}.$$

C'est un caractère additif, $\psi_a \in \widehat{\mathbb{F}_q}$, et réciproquement, tout caractère additif s'écrit de cette façon.

Démonstration. Il est évident que l'on a bien construit ainsi des caractères. Montrons qu'ils sont tous différents.
Comme la trace est non identiquement nulle, le caractère canonique est non trivial. Si on considère deux éléments $a \neq b$ de $\mathbb{F}_q$, alors on peut trouver un autre élément $c \in \mathbb{F}_q$ tel que

$$\frac{\psi_a(c)}{\psi_b(c)} = \psi_1\left((a-b)c\right) \neq 1.$$

On a donc $\psi_a \neq \psi_b$. Ceci signifie que le nombre de caractères du type ψ_a est égal à q. Or nous savons, avec le corollaire 2.3, chap. I, que $|\widehat{\mathbb{F}_q}| = |\mathbb{F}_q| = q$. Nous avons donc bien construit ainsi tous les caractères. □

***Remarque* 1.8. (Caractère trivial).** Nous avons ainsi construit un isomorphisme entre $\mathbb{F}_q$ et son dual $\widehat{\mathbb{F}_q}$ par l'application $a \mapsto \psi_a$. On note $\psi_0 = 1$ le caractère additif trivial. Il ne faut pas le confondre avec le caractère multiplicatif trivial χ_0, puisque ce dernier n'est pas défini en 0. Nous verrons plus tard que l'on prolonge souvent les caractères multiplicatifs $\chi \in \widehat{\mathbb{F}_q^*}$ en posant $\chi(0) = 0$, ce qui lève toute ambiguïté entre les deux caractères triviaux.

***Remarque* 1.9. (Extension des caractères).** Soit K un sur-corps fini de $\mathbb{F}_q$, ce que l'on peut écrire sous la forme $K = \mathbb{F}_{q^t}$, où $t \overset{\text{déf.}}{=} [K : \mathbb{F}_q]$. On vérifie aisément que pour $\beta \in K$, on a

$$\mathrm{Tr}_{K/\mathbb{F}_p}(\beta) = \mathrm{Tr}_{\mathbb{F}_q/\mathbb{F}_p}\left(\mathrm{Tr}_{K/\mathbb{F}_q}(\beta)\right).$$

Si on note μ_1 le caractère canonique de K, ceci implique que

$$\mu_1(\beta) = \psi_1(\mathrm{Tr}_{K/\mathbb{F}_q}(\beta)). \tag{1.3}$$

Avant de détailler les propriétés des caractères additifs et multiplicatifs, donnons un exemple fondamental.

Exemple 1.10 (Caractère quadratique). Soit q un entier impair. On définit un caractère multiplicatif $\eta \in \widehat{\mathbb{F}_q^*}$ de la façon suivante :

$$\forall x \in \mathbb{F}_q^*, \quad \eta(x) \overset{\text{déf.}}{=} \begin{cases} 1 & \text{si } x \text{ est un carré dans } \mathbb{F}_q \\ -1 & \text{sinon} \end{cases}.$$

On voit facilement que l'on a $\eta = \chi_{\frac{q-1}{2}}$. De plus, dans le cas ou $q = p$ est un nombre premier, on a $\eta(x) = \left(\frac{x}{p}\right)$, ce qui signifie que η est le symbole de *Legendre*.

Nous allons maintenant rappeler les propriétés d'orthogonalité des caractères démontrées au paragraphe 2.4, chap. I, en les énonçant pour les caractères additifs et multiplicatifs.

Proposition 1.11 (Propriétés des caractères additifs). *Soient a et b des éléments de $\mathbb{F}_q$. On a alors*

$$\sum_{x \in \mathbb{F}_q} \psi_a(x)\overline{\psi_b(x)} = \begin{cases} 0 & si \quad a \neq b \\ q & si \quad a = b \end{cases} \tag{1.4}$$

$$\sum_{x \in \mathbb{F}_q} \psi_a(x) = 0 \quad si\, a \neq 0 \tag{1.5}$$

$$\sum_{x \in \mathbb{F}_q} \psi_x(a)\overline{\psi_x(b)} = \begin{cases} 0 & si \quad a \neq b \\ q & si \quad a = b \end{cases}. \tag{1.6}$$

Proposition 1.12 (Propriétés des caractères multiplicatifs). *Soient a et b des éléments de $\mathbb{F}_q^*$ et soient χ et τ deux éléments de $\widehat{\mathbb{F}_q^*}$. On a alors*

$$\sum_{x \in \mathbb{F}_q^*} \chi(x)\overline{\tau(x)} = \begin{cases} 0 & si \quad \chi \neq \tau \\ q-1 & si \quad \chi = \tau \end{cases} \tag{1.7}$$

$$\sum_{x \in \mathbb{F}_q^*} \chi(x) = 0 \quad si\, \chi \neq \chi_0 \tag{1.8}$$

$$\sum_{\chi \in \widehat{\mathbb{F}_q^*}} \chi(a)\overline{\chi(b)} = \begin{cases} 0 & si \quad a \neq b \\ q-1 & si \quad a = b \end{cases}. \tag{1.9}$$

***Remarque* 1.13.** Comme nous l'avons déjà expliqué au paragraphe 2.4, chap. I, on peut représenter les caractères d'un groupe fini abélien sous la forme d'une matrice (chaque ligne représente un caractère). Dans ce cadre, les équations (1.4) et (1.7) représentent des relations d'orthogonalité entre les lignes de la matrice, et les équations (1.6) et (1.9) représentent des relations d'orthogonalité entre les colonnes de la matrice.

1.3 Sommes de Gauss

On peut maintenant définir l'objet important de ce chapitre, qui fait la liaison entre les caractères additifs et multiplicatifs d'un corps fini.

Définition 1.14 (Sommes de Gauss). Soient $\chi \in \widehat{\mathbb{F}_q^*}$ et $\psi \in \widehat{\mathbb{F}_q}$ des caractères respectivement multiplicatif et additif. On définit la *somme de Gauss* $G(\chi, \psi)$ associée à ces deux caractères, par

$$G(\chi, \psi) \overset{\text{déf.}}{=} \sum_{x \in \mathbb{F}_q^*} \psi(x)\chi(x). \tag{1.10}$$

Cette somme de Gauss, qui met en jeu les deux structures du corps fini, est en fait très proche de la transformée de Fourier, comme on a pu la définir à l'équation (4.1), chap. I. En effet, rappelons la définition de la transformée de Fourier sur le groupe multiplicatif $\mathbb{F}_q^*$:

$$\forall f \in \mathbb{C}[\mathbb{F}_q^*], \quad \mathscr{F}_{\text{mul}}(f) : \begin{cases} \widehat{\mathbb{F}_q^*} & \longrightarrow & \mathbb{C} \\ \chi & \longmapsto & \sum\limits_{x \in \mathbb{F}_q^*} f(x)\chi(x) \end{cases} .$$

On peut donc écrire la somme de Gauss de l'équation (1.10) comme la transformée de Fourier multiplicative d'un caractère additif. Plus précisément, on a

$$\forall \psi \in \widehat{\mathbb{F}_q}, \forall \chi \in \widehat{\mathbb{F}_q^*}, \quad G(\chi, \psi) = \mathscr{F}_{\text{mul}}(\psi)(\chi).$$

Cependant, dans la suite de l'exposé, nous allons être amenés à considérer le point de vue inverse, c'est-à-dire que nous allons plutôt nous intéresser à la transformée de Fourier sur le groupe additif. C'est pour cela que l'on étend un caractère multiplicatif $\chi \in \widehat{\mathbb{F}_q^*}$ en une fonction $\widetilde{\chi} \in \mathbb{C}[\mathbb{F}_q]$ en posant $\widetilde{\chi}(0) = 0$. Dans ces conditions, on peut aussi voir une somme de Gauss comme la transformée de Fourier additive d'un caractère multiplicatif. Rappelons la définition de la transformée additive :

$$\forall f \in \mathbb{C}[\mathbb{F}_q], \quad \mathscr{F}_{\text{add}}(f) : \begin{cases} \widehat{\mathbb{F}_q} & \longrightarrow & \mathbb{C} \\ \psi & \longmapsto & \sum\limits_{x \in \mathbb{F}_q} f(x)\psi(x) \end{cases} . \tag{1.11}$$

On obtient alors la formule remarquable

$$\forall \psi \in \widehat{\mathbb{F}_q}, \forall \chi \in \widehat{\mathbb{F}_q^*}, \quad G(\chi, \psi) = \mathscr{F}_{\text{add}}(\widetilde{\chi})(\psi). \tag{1.12}$$

On prendra donc garde au fait que la fonction $\widetilde{\chi}$ correspond au caractère χ prolongé en 0.

Comme application de ces constatations, on peut décomposer un caractère multiplicatif en série de Fourier additive.

Proposition 1.15. *Soit $\chi \in \widehat{\mathbb{F}_q^*}$. On a*

$$\chi = \frac{1}{q} \sum_{\psi \in \widehat{\mathbb{F}_q}} G(\chi, \overline{\psi})\psi.$$

Démonstration. En appliquant à la fonction $\widetilde{\chi}$ la formule de décomposition en série de Fourier, proposition 4.4, chap. I, on obtient

$$\widetilde{\chi} = \sum_{\psi \in \widehat{\mathbb{F}_q}} \langle \widetilde{\chi}, \psi \rangle \, \psi.$$

Il ne reste plus qu'à remarquer que $\langle \widetilde{\chi}, \psi \rangle \overset{\text{déf.}}{=} \frac{1}{q}\mathscr{F}_{\text{add}}(\widetilde{\chi})(\overline{\psi}) = \frac{1}{q}G(\chi, \overline{\psi})$ pour conclure. □

Dans la pratique, on est bien souvent incapable de calculer simplement les valeurs de ces sommes de Gauss. La seule majoration (triviale) dont on dispose est $|G(\chi, \psi)| \leqslant q - 1$. Cependant, la proposition 1.17 va nous donner la valeur de son module. Commençons par énoncer une série de propriétés plus où moins évidentes des sommes de Gauss.

Proposition 1.16 (Propriétés des sommes de Gauss). *On rappelle que p est la caractéristique du corps $\mathbb{F}_q$, c'est-à-dire que $q = p^r$. Alors, si on note $\chi \in \widehat{\mathbb{F}_q^*}$ et $\psi \in \widehat{\mathbb{F}_q}$:*

(i) *pour a et $b \in \mathbb{F}_q$, on a $G(\chi, \psi_{ab}) = \overline{\chi(a)}G(\chi, \psi_b)$.*

(ii) $G(\chi,\overline{\psi}) = \chi(-1)G(\chi,\psi)$.
(iii) $G(\overline{\chi},\psi) = \chi(-1)\overline{G(\chi,\psi)}$.

Démonstration. Démontrons (i) :

$$G(\chi,\psi_{ab}) = \sum_{x\in\mathbb{F}_q^*} \chi(x)\psi_b(ax) = \chi(a^{-1}) \sum_{y\in\mathbb{F}_q^*} \chi(y)\psi_b(y),$$

en effectuant le changement de variable $ax = y$.
La propriété (ii) s'obtient à partir de (i) en prenant $b = -1$.
La propriété (iii) découle de (ii) en passant à la conjugaison et en utilisant le fait que $\chi(-1) \in \mathbb{R}$. □

Proposition 1.17 (Calcul des sommes de Gauss). *On conserve les notations de la définition 1.14. On a alors*

$$G(\chi,\psi) = \begin{cases} q-1 & si \quad \chi = \chi_0 \quad et \quad \psi = \psi_0 \quad (cas\ 1) \\ -1 & si \quad \chi = \chi_0 \quad et \quad \psi \neq \psi_0 \quad (cas\ 2) \\ 0 & si \quad \chi \neq \chi_0 \quad et \quad \psi = \psi_0 \quad (cas\ 3) \end{cases}.$$

Dans les autres cas, on a $|G(\chi,\psi)| = q^{1/2}$. *De plus, on a*

$$G(\chi,\psi)G(\overline{\chi},\psi) = q\chi(-1), \quad pour \quad \chi \neq \chi_0 \quad et \quad \psi \neq \psi_0 \tag{1.13}$$

Démonstration. Le cas 1 est trivial.
Le cas 2 résulte immédiatement de l'équation (1.5) (il manque le terme $\psi(0) = 1$ dans la somme).
Le cas 3 résulte, lui, de l'équation (1.8).
Enfin, pour le cas général, nous allons exploiter le fait que la fonction $\psi \mapsto G(\chi,\psi)$ est la transformée de Fourier (additive) de la fonction $\widetilde{\chi}$ prolongée en 0, comme nous l'avons déjà remarqué à l'équation (1.12). En utilisant la formule de Plancherel (4.3), chap. I, on obtient

$$\langle G(\chi,\cdot), G(\chi,\cdot)\rangle = q\,\langle\widetilde{\chi},\widetilde{\chi}\rangle = q. \tag{1.14}$$

Choisissons donc un caractère additif $\psi = \psi_a \in \widehat{\mathbb{F}_q}$. On peut réécrire l'équation (1.14) de la façon suivante :

$$\frac{1}{q}\sum_{b\in\mathbb{F}_q} G(\chi,\psi_{ab})\overline{G(\chi,\psi_{ab})} = q.$$

Il ne reste plus qu'à utiliser le résultat de la proposition 1.16, (i), pour conclure

$$\frac{1}{q}\sum_{b\in\mathbb{F}_q} |\chi(b)|^2|G(\chi,\psi_a)|^2 = |G(\chi,\psi_a)|^2\,\langle\chi,\chi\rangle = |G(\chi,\psi_a)|^2 = q.$$

On peut maintenant démontrer l'égalité (1.13). En utilisant la proposition 1.16, (iii), on obtient

$$G(\chi,\psi)G(\overline{\chi},\psi) = \chi(-1)|G(\chi,\psi)|^2.$$

On obtient le résultat souhaité en utilisant le fait que $|G(\chi,\psi)| = q^{1/2}$. □

Ces propriétés montrent bien l'importance de la connaissance de $\chi(-1)$. A priori, on sait seulement que $\chi(-1) \in \{-1,1\}$. La proposition suivante va nous en dire plus.

Proposition 1.18. *Soit χ un caractère multiplicatif, et soit m son ordre dans $\widehat{\mathbb{F}_q^*}$, c'est-à-dire le plus petit entier positif k tel que $\chi^k = \chi_0$. Alors $\chi(-1) = -1$ si et seulement si m est pair et $\frac{q-1}{m}$ est impair.*

Démonstration. La première chose à remarquer est que comme $\chi^{q-1} = \psi_0$, on a $m|q-1$. De plus, comme χ est à valeurs dans l'ensemble des racines $m^{\text{ièmes}}$ de l'unité, la valeur -1 ne peut apparaître que si m est pair. Donc l'énoncé de cette proposition a bien un sens, ce qui est rassurant.
On note g_0 un générateur de $\mathbb{F}_q^*$, ce qui implique que $\chi(g_0)$ est une racine $m^{\text{ième}}$ primitive de l'unité (car $\chi(g_0)$ est un élément d'ordre m dans le groupe des nombres complexes de module 1). Alors, si m est pair (donc q est nécessairement impair), on a

$$\chi(-1) = \chi\left(g_0^{(q-1)/2}\right) = \zeta^{(q-1)/2}.$$

Donc on a $\chi(-1) = -1$ si et seulement si $\zeta^{(q-1)/2} = \zeta^{m/2}$, c'est-à-dire $(q-1)/2 \equiv m/2$ modulo m. Ceci est équivalent à $(q-1)/m \equiv 1$ modulo 2, ce qui signifie que $(q-1)/m$ est impair. □

1.4 La réciprocité quadratique

Le but de ce paragraphe est d'étudier de plus près le caractère quadratique, pour au final démontrer la fameuse formule de réciprocité quadratique. Afin d'y parvenir, nous allons utiliser la transformée de Fourier additive $\mathscr{F}_{\text{add}}$, définie par l'équation (1.11).

A $\mathscr{F}_{\text{add}}$, nous préférerons utiliser un endomorphisme de $\mathbb{C}[\mathbb{F}_q^*]$, pour plus de commodité. Celui-ci sera noté $T : \mathbb{C}[\mathbb{F}_q^*] \to \mathbb{C}[\mathbb{F}_q^*]$. Il est défini de la façon suivante :

$$\forall f \in \mathbb{C}[\mathbb{F}_q^*], \quad Tf : \left\{ \begin{array}{ccc} \mathbb{F}_q^* & \longrightarrow & \mathbb{F}_q^* \\ a & \longmapsto & \displaystyle\sum_{x \in \mathbb{F}_q^*} f(x)\psi_a(x) \end{array} \right. .$$

La différence par rapport à la transformée de Fourier additive tient à peu de choses, puisque l'on a

$$\forall f \in \mathbb{C}[\mathbb{F}_q^*], \quad Tf(a) = \mathscr{F}_{\text{add}}\left(\widetilde{f}\right)(\psi_a),$$

où l'on a prolongé f en 0 par $\widetilde{f}(0) = 0$. L'utilité de cet opérateur par rapport à la transformée de Fourier additive est qu'il rend les formules dans lesquelles interviennent les sommes de Gauss plus simples. L'opérateur T n'est en fait rien d'autre que $R\mathscr{F}_{\text{add}}P$, où P est le plongement de $\mathbb{C}[\mathbb{F}_q^*]$ dans $\mathbb{C}[\mathbb{F}_q]$ déjà décrit et R la surjection de l'espace vectoriel $\mathbb{C}[\mathbb{F}_q]$ dans $\mathbb{C}[\mathbb{F}_q^*]$. Ainsi, en reprenant l'équation (1.12), on obtient

$$\forall \chi \in \widehat{\mathbb{F}_q^*}, \forall \psi \in \widehat{\mathbb{F}_q}, \quad T\chi(x) = G(\chi, \psi_x) = \overline{\chi}(x)G(\chi, \psi_1). \tag{1.15}$$

La dernière égalité a été obtenue grâce à la proposition 1.16, propriété (i).

Dans la suite de l'exposé, on se restreindra au cas où $q = p$, de sorte que l'on travaillera dans le corps $\mathbb{F}_p$. Dans ce cas, l'opérateur T s'exprime de la manière suivante :

$$\forall f \in \mathbb{C}[\mathbb{F}_p^*], \quad Tf : \left\{ \begin{array}{ccc} \mathbb{F}_p^* & \longrightarrow & \mathbb{F}_p^* \\ a & \longmapsto & \displaystyle\sum_{x \in \mathbb{F}_p^*} f(x)\zeta^{ax} \end{array} \right. ,$$

où l'on a noté $\zeta \overset{\text{déf.}}{=} e^{\frac{2i\pi}{p}}$. Nous allons maintenant démontrer un lemme qui fait le lien entre l'opérateur T et le caractère quadratique η.

Lemme 1.19. *Soit $\eta \in \widehat{\mathbb{F}_p^*}$ le caractère quadratique sur le corps $\mathbb{F}_p$. On a alors*

$$\det(T) = (-1)^{\frac{p-1}{2}} i^{\frac{(p-1)(p-3)}{4}} p^{\frac{p-3}{2}} G(\eta, \psi_1).$$

Démonstration. Il s'agit d'écrire la matrice de T dans la base des caractères multiplicatifs $\{\chi_0, \dots, \chi_{p-2}\}$. Les deux seuls caractères qui sont à valeurs réelles sont χ_0 et η. On peut regrouper les autres caractères par paires $(\chi, \overline{\chi})$, et en utilisant l'équation (1.15), on obtient une matrice du type

$$\begin{pmatrix} G(\chi_0,\psi_1) & & & & & & \\ & G(\eta,\psi_1) & & & & & \\ & & \begin{matrix} 0 & G(\chi_1,\psi_1) \\ G(\overline{\chi_1},\psi_1) & 0 \end{matrix} & & & & \\ & & & \ddots & & & \\ & & & & \begin{matrix} 0 & G\left(\chi_{\frac{p-3}{2}},\psi_1\right) \\ G\left(\overline{\chi_{\frac{p-3}{2}}},\psi_1\right) & 0 \end{matrix} & & \\ & & & & & G(\eta,\psi_1) & \\ & & & & & & G(\chi_0,\psi_1) \end{pmatrix}.$$

On sait, avec la proposition 1.17 (cas 2), que $G(\chi_0, \psi_1) = -1$. La valeur de $G(\eta, \psi_1)$ est pour l'instant inconnue. Il ne reste qu'à calculer les sous-déterminants de taille 2

$$\det \begin{pmatrix} 0 & G(\chi,\psi_1) \\ G(\overline{\chi},\psi_1) & 0 \end{pmatrix} = -G(\chi,\psi_1)G(\overline{\chi},\psi_1) = -\chi(-1)p.$$

Pour ce calcul, on a utilisé l'égalité (1.13). On obtient donc la valeur du déterminant

$$\begin{aligned} \det(T) &= -G(\eta,\psi_1)(-p)^{\frac{p-3}{2}} \prod_{j=1}^{\frac{p-3}{2}} \chi_j(-1) \\ &= (-1)^{\frac{p-1}{2}} p^{\frac{p-3}{2}} G(\eta,\psi_1) \prod_{j=1}^{\frac{p-3}{2}} \chi_j(-1). \end{aligned}$$

Or, $\chi_j(-1) = \chi_1(-1)^j = (-1)^j$, ce qui fournit une évaluation du produit de droite

$$\prod_{j=1}^{\frac{p-3}{2}} \chi_j(-1) = (-1)^{1+2+\cdots+\frac{p-1}{2}} = (-1)^{\frac{(p-1)(p-3)}{8}} = \mathrm{i}^{\frac{(p-1)(p-3)}{4}}.$$

Ceci correspond bien au résultat annoncé. □

On peut maintenant énoncer un résultat important. Il s'agit de calculer les sommes de Gauss mettant en jeu le caractère quadratique. Il a été démontré par *Gauss*, et lui a permis de fournir, en 1807, la $6^{\text{ème}}$ de ses 8 démonstrations de la formule de réciprocité quadratique.

Proposition 1.20 (Signes des sommes de Gauss). *Soit p un nombre premier impair. On note η le caractère quadratique de $\mathbb{F}_p$. Alors*

$$G(\eta,\psi_1) = \begin{cases} p^{1/2} & si\ p \equiv 1 \mod 4 \\ \mathrm{i}p^{1/2} & si\ p \equiv 3 \mod 4 \end{cases}.$$

Démonstration. Comme $\eta = \overline{\eta}$, en appliquant l'égalité (1.13), il vient

$$G(\eta,\psi_1)^2 = \eta(-1)p.$$

On peut alors utiliser la proposition 1.18, et voir que

$$\eta(-1) = \begin{cases} 1 & \text{si } p \equiv 1 \mod 4 \\ -1 & \text{si } p \equiv 3 \mod 4 \end{cases}.$$

On obtient donc presque le résultat voulu, c'est-à-dire

$$G(\eta,\psi_1)=\begin{cases}\varepsilon_p p^{1/2} & \text{si } p\equiv 1 \mod 4\\ \varepsilon_p \mathrm{i} p^{1/2} & \text{si } p\equiv 3 \mod 4\end{cases} \qquad \text{avec } \varepsilon_p\in\{+1,-1\}.$$

Toute la difficulté réside dans la détermination des signes, c'est-à-dire de ε_p (quantité qui dépend a priori de p).
Commençons par réécrire la dernière égalité de façon plus compacte :

$$G(\eta,\psi_1)=\varepsilon_p \mathrm{i}^{\frac{(p-1)^2}{4}} p^{1/2}.$$

Cette égalité vient simplement du fait que

$$\mathrm{i}^{\frac{(p-1)^2}{4}}=\begin{cases}1 & \text{si } p\equiv 1 \mod 4\\ \mathrm{i} & \text{si } p\equiv 3 \mod 4\end{cases}.$$

Nous allons pouvoir utiliser le calcul de $\det(T)$ que nous venons d'effectuer au lemme 1.19. En insérant la valeur de $G(\eta,\psi_1)$ dans ce déterminant, il vient

$$\det(T)=\varepsilon_p(-1)^{\frac{p-1}{2}}\mathrm{i}^{\frac{(p-1)(p-3)}{4}}\mathrm{i}^{\frac{(p-1)^2}{4}}p^{\frac{p-3}{2}}p^{\frac{1}{2}} \tag{1.16}$$

$$=\varepsilon_p(-1)^{\frac{p-1}{2}}\mathrm{i}^{\frac{(p-1)(p-2)}{2}}p^{\frac{p-2}{2}}. \tag{1.17}$$

Pour déterminer le signe qui apparaît dans cette expression, nous allons calculer le déterminant dans une autre base, celle des fonctions Dirac $\{\delta_1,\dots,\delta_{p-1}\}$. Comme on a $T\delta_k(x)=\zeta^{xk}$, on obtient

$$\det(T)=\det\left(\zeta^{jk}\right)_{1\leqslant j,k\leqslant p-1}=\prod_{1\leqslant m<n\leqslant p-1}(\zeta^n-\zeta^m).$$

La dernière égalité s'obtient en calculant un déterminant de *Vandermonde*. En passant à l'angle moitié, c'est-à-dire en posant $\mu\overset{\text{déf.}}{=}e^{\frac{\mathrm{i}\pi}{p}}$, il vient

$$\begin{aligned}\det(T)&=\prod(\mu^{2n}-\mu^{2m})=\prod_{m<n}\mu^{m+n}(\mu^{n-m}-\mu^{m-n})\\ &=\prod\mu^{n+m}\prod\mathrm{i}\prod 2\sin\left(\frac{\pi(n-m)}{p}\right),\end{aligned}$$

où le signe $\prod$ signifie $\prod_{m<n}$. Il faut évaluer les trois produits qui apparaissent dans cette expression. En ce qui concerne le produit de droite, il est positif, et ceci est suffisant pour ce que l'on veut en faire :

$$\prod 2\sin\left(\frac{\pi(n-m)}{p}\right)=A>0.$$

Pour le produit du milieu, il suffit de remarquer que

$$\sharp\{(m,n)\setminus 1\leqslant m<n\leqslant p-1\}=\frac{(p-1)(p-2)}{2}$$

(on peut faire un dessin et compter le nombre de points à coordonnées entières à l'intérieur d'un triangle). On obtient donc

$$\prod\mathrm{i}=\mathrm{i}^{\frac{(p-1)(p-2)}{2}}.$$

Quant au produit de gauche, on utilise le calcul suivant :

$$\begin{aligned}\sum_{1\leqslant m<n\leqslant p-1} n+m &= \sum_{n=2}^{p-1}\sum_{m=1}^{n-1} n+m = \frac{3}{2}\sum_{n=1}^{p-2} n(n-1)\\ &= \frac{3}{2}\left(\frac{(p-2)(p-3)(2p-3))}{6}+\frac{(p-1)(p-2)}{2}\right)\\ &= \frac{p(p-1)(p-2)}{2},\end{aligned}$$

d'où

$$\prod_{1\leqslant m<n\leqslant p-1}\mu^{n+m} = \mu^{\frac{p(p-1)(p-2)}{2}} = (-1)^{\frac{(p-1)(p-2)}{2}}.$$

On obtient enfin l'expression du déterminant de T :

$$\det(T) = (-1)^{\frac{p-1}{2}}\mathrm{i}^{\frac{(p-1)(p-2)}{2}}A \quad \text{avec } A>0.$$

En comparant cette expression à l'équation (1.17), on voit que $\varepsilon_p = +1$. □

***Remarque* 1.21.** Ce résultat se généralise au cas d'un corps $\mathbb{F}_q$ quelconque, c'est-à-dire pour $q = p^s$. On peut en effet énoncer :

$$G(\eta,\psi_1) = \begin{cases} (-1)^{s-1}q^{1/2} & \text{si } q\equiv 1 \mod 4\\ (-1)^{s-1}\mathrm{i}^s q^{1/2} & \text{si } q\equiv 3 \mod 4\end{cases}.$$

La démonstration de ce résultat passe par la démonstration d'un lemme que l'on trouvera dans le livre de LIDL et NIEDERREITER [48].

Après ces démonstrations quelque peu calculatoires, on est enfin en mesure de prouver la fameuse formule de réciprocité quadratique. Elle fut énoncée par *Legendre* en 1788, et démontrée pour la première fois par *Gauss* en 1801. A ce jour, on dénombre plusieurs centaines de démonstrations différentes. FRANZ LEMMERMEYER en a réuni une grande quantité, et présente la première partie de l'historique de cette formule dans [46].

Théorème 1.22 (Réciprocité quadratique). *Pour tous nombres premiers impairs distincts p et r, on a*

$$\left(\frac{p}{r}\right)\left(\frac{r}{p}\right) = (-1)^{\frac{(p-1)(r-1)}{4}}. \tag{1.18}$$

Démonstration. Soit η le caractère multiplicatif quadratique de $\mathbb{F}_p$, et ψ_1 le caractère additif canonique. Avec le théorème précédent, on sait que

$$G(\eta,\psi_1)^2 = (-1)^{\frac{p-1}{2}}p \overset{\text{déf.}}{=} \widetilde{p}.$$

On note maintenant $G \overset{\text{déf.}}{=} G(\eta,\chi_1)$. On a

$$G^r = \left(G^2\right)^{\frac{r-1}{2}}G = \widetilde{p}^{\frac{r-1}{2}}G.$$

Dans la suite, nous allons effectuer nos calculs dans l'anneau R des entiers algébriques, c'est-à-dire des nombres complexes qui sont racines de polynômes unitaires à coefficients entiers. Comme les caractères sont des sommes de racines de l'unité, les valeurs des

sommes de Gauss sont des entiers algébriques, donc $G \in R$. On note (r) l'idéal engendré par r dans R. Comme l'anneau quotient $R/(r)$ est de caractéristique r, on obtient

$$G^r = \left(\sum_{x \in \mathbb{F}_p^*} \eta(x)\psi_1(x) \right)^r = \sum_{x \in \mathbb{F}_p^*} (\eta(x)^r \psi_1(x)^r) \mod (r).$$

Comme $\eta(x)^r = \eta(x)$ (car $\eta(x) \in \{-1, 1\}$ et r est impair) et que $\psi_1(x)^r = \psi_r(x)$, on obtient

$$G^r = \sum_{x \in \mathbb{F}_p^*} \eta(x)\psi_r(x) = G(\eta, \psi_r) = \psi(r)G \mod (r).$$

Pour la dernière égalité, on a utilisé le résultat (i) de la proposition 1.16. On a donc

$$G^r = \eta(r)G = \widetilde{p}^{\frac{r-1}{2}} G \mod (r).$$

En multipliant cette égalité par G, et en utilisant le fait que $G^2 = \widetilde{p}$, on obtient l'égalité suivante :

$$\widetilde{p}^{\frac{r-1}{2}} \widetilde{p} = \eta(r)\widetilde{p} \mod (r).$$

Cette égalité est en fait une égalité sur $\mathbb{Z}/r\mathbb{Z}$. Comme p et r sont premiers entre eux, on peut simplifier par $\widetilde{p}$, pour obtenir

$$\widetilde{p}^{\frac{r-1}{2}} = (-1)^{\frac{(p-1)(r-1)}{4}} p^{\frac{r-1}{2}} = \eta(r) \mod r.$$

Comme nous l'avons déjà fait remarquer, on a $\eta(r) = \left(\frac{r}{p}\right)$. De plus, avec la formule d'Euler (lemme 1.1), on a $p^{\frac{r-1}{2}} = \left(\frac{p}{r}\right)$. On a donc en fait l'égalité suivante :

$$(-1)^{\frac{(p-1)(r-1)}{4}} \left(\frac{p}{r}\right) = \left(\frac{r}{p}\right) \mod r.$$

Comme les deux membres de cette égalité sont en fait à valeurs dans $\{-1, 1\}$, et que $r \geqslant 3$, cette égalité est en fait valide sur $\mathbb{Z}$, ce qui est la conclusion à laquelle on voulait arriver. □

2 Transformée de Walsh

Avant de présenter, au chapitre suivant, une série d'algorithmes rapides pour calculer des transformées de Fourier sur un groupe cyclique, nous allons décrire une transformée qui dispose elle aussi d'un algorithme rapide. Il s'agit de la transformée de *Walsh*. Derrière ce nom se cache en fait une réécriture de la transformée de Fourier sur un groupe abélien, dans un cas très particulier, celui du groupe $G = (\mathbb{Z}/2\mathbb{Z})^k$. Ce groupe est souvent appelé *cube booléen*, et on peut voir un dessin du cube de dimension 4 à la figure 2.1.

2.1 Présentation

En suivant de près la démonstration du corollaire 2.5, chap. I, on peut construire facilement le dual d'un groupe qui s'écrit comme un produit de groupes cycliques élémentaires. En effet, chaque caractère va s'exprimer comme un produit des différents caractères des

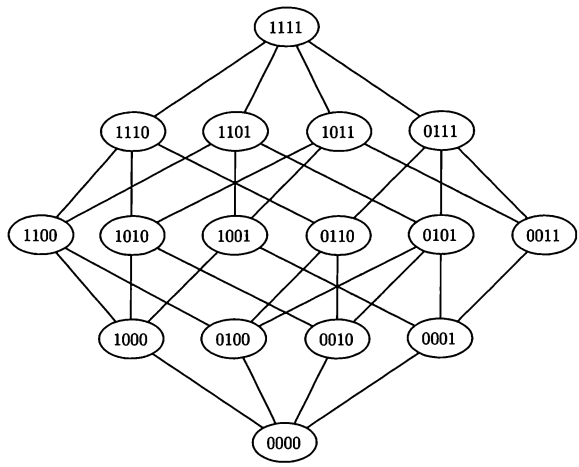

FIG. 2.1 – *Cube booléen* $(\mathbb{F}_2)^4$

groupes élémentaires. Dans le cas du groupe $(\mathbb{Z}/2\mathbb{Z})^k$, c'est extrêmement simple, puisque le seul caractère non trivial du groupe $\mathbb{Z}/2\mathbb{Z} = \{0,1\}$ est défini par

$$\chi_\varepsilon(0) = 1 \qquad \chi_\varepsilon(1) = -1.$$

A chaque élément $a = \{a_0, \ldots, a_{k-1}\} \in (\mathbb{Z}/2\mathbb{Z})^k$ on peut donc assigner un caractère

$$\chi_a : \begin{cases} (\mathbb{Z}/2\mathbb{Z})^k & \longrightarrow & \{-1,1\} \\ x = \{x_0, \ldots, x_{k-1}\} & \longmapsto & (-1)^{a_0 x_0}(-1)^{a_1 x_1} \cdots (-1)^{a_{k-1}x_{k-1}} = (-1)^{\langle a,x\rangle} \end{cases} . \quad (2.1)$$

où l'on a noté $\langle a,x\rangle$ la forme bilinéaire canonique sur $(\mathbb{Z}/2\mathbb{Z})^k$ définie par

$$\langle a,x\rangle \stackrel{\text{déf.}}{=} \sum_{i=0}^{k-1} a_i x_i.$$

Dans la pratique, on représente les éléments du groupe $(\mathbb{Z}/2\mathbb{Z})^k$ comme des entiers compris entre 0 et $2^k - 1$, en assimilant l'élément $x \in G = (\mathbb{Z}/2\mathbb{Z})^k$ avec l'entier $\sum_{i=0}^{k-1} x_i 2^i$. Ceci permet de voir les caractères χ_a (où a peut être vu comme un élément de G ou comme un entier de $\{0, \ldots, 2^k - 1\}$) comme des vecteurs de taille 2^k remplis de -1 et de 1.

Exemple 2.1. Considérons le groupe $(\mathbb{Z}/2\mathbb{Z})^3$, de cardinal 8. On peut représenter sa table des caractères comme une matrice carrée d'ordre 8, noté W_8, dont la ligne i représente les valeurs du caractère χ_i, c'est-à-dire que $(W_8)_{ij} = \chi_i(j)$. Voici la table :

$$W_8 \stackrel{\text{déf.}}{=} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} . \quad (2.2)$$

La figure 2.2 montre les matrices de Walsh W_{32} et W_{64}, où l'on a mis en blanc les entrées égales à 1, et en noir celles égales à -1.

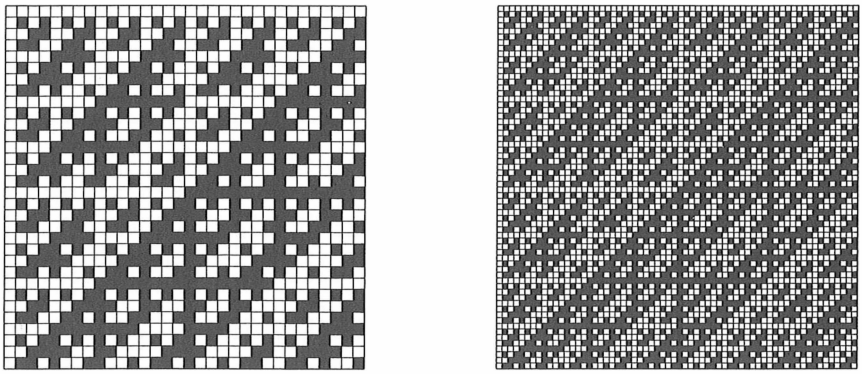

FIG. 2.2 – *Matrices de Walsh W_{32} et W_{64}*

On peut alors définir la *transformée de Walsh.*

Définition 2.2 (Transformée de Walsh). On définit la *transformée de Walsh* $\mathscr{W}_k(f)$ d'un vecteur complexe $f = \{f[0], \ldots, f[2^k - 1]\}$ de taille 2^k par

$$\forall i \in \{0, \ldots, 2^k - 1\}, \quad \mathscr{W}_k(f)[i] \stackrel{\text{déf.}}{=} \sum_{j=0}^{2^k-1} f[j]\chi_i(j) = 2^k \langle f, \chi_i \rangle . \tag{2.3}$$

***Remarque* 2.3. (Lien avec la transformée de Fourier).** On convient de noter encore $f : (\mathbb{Z}/2\mathbb{Z})^k \to \mathbb{C}$ la fonction correspondant au vecteur f, c'est-à-dire le vecteur correspondant à la décomposition de f dans la base des fonctions de Dirac $\{\delta_x\}_{x \in G}$. Alors le calcul de $\mathscr{W}(f)$ est celui d'une transformée de Fourier, puisque

$$\mathscr{W}(f)[i] = 2^k \langle f, \chi_i \rangle = \widehat{f}(\chi_i),$$

où l'on a noté $\widehat{f} : \widehat{G} \to \mathbb{C}$ la transformée de Fourier de f. En particulier, nous allons donc pouvoir énoncer sans effort la formule d'inversion pour la transformée de Walsh.

Si l'on représente les fréquences $i \in \{0, \ldots, 2^n - 1\}$ sur le cube booléen de la figure 2.1, alors les fréquences situées en bas du schéma sont souvent appelées *basses fréquences.* On retrouve de nombreuses analogies avec les spectres de Fourier déjà rencontrés (transformées de Fourier sur un groupe fini, transformée continue, etc.). Ainsi, l'exercice VI.5 propose d'utiliser les propriétés spectrales de la transformée de Walsh pour réaliser des prédictions booléennes (ceci est en rapport avec la *théorie de l'apprentissage*).

L'opérateur $\mathscr{W}_k : \mathbb{C}^{2^k} \to \mathbb{C}^{2^k}$ est une application linéaire dont la matrice dans la base canonique est W_{2^k}, la table des caractères du groupe $(\mathbb{Z}/2\mathbb{Z})^k$. On a ainsi $\mathscr{W}_k(f) = W_{2^k} f$. La formule d'inversion obtenue à la proposition 4.4, chap. I, nous apporte les informations suivantes.

Proposition 2.4. *La transformée de Walsh est inversible, et son inverse est* $\frac{1}{2^k}\mathscr{W}_k$. *D'un point de vue matriciel, ceci signifie que la matrice* $W_{2^k} = \{w_{ij}\}$, *définie par* $w_{ij} = (-1)^{\langle i,j \rangle}$, *vérifie* $W_{2^k} W_{2^k} = 2^k Id$.

La matrice de Walsh permet d'étudier des problèmes concrets, par exemple en statistiques, comme le montre l'exercice II.3.

2.2 Algorithme de calcul rapide

L'un des intérêts de la transformée de Walsh est que l'on dispose d'un algorithme rapide pour la calculer. En effet, bien que l'équation qui la définit puisse sembler un peu compliquée, elle peut se décomposer de façon simple. Ceci va permettre de mettre en œuvre un algorithme récursif de calcul, bien plus efficace que l'évaluation naïve de la somme qui définit la transformée. Nous étudierons au chapitre suivant la construction de *l'algorithme FFT*, et on retrouvera exactement les mêmes idées, qui sont à la base de la « philosophie » algorithmique appelée *diviser pour régner*.

Plutôt que de passer du temps sur l'analyse d'un tel algorithme (son coût, son implémentation, etc.), nous allons simplement décrire l'équation de récurrence que nous allons mettre en œuvre. Les discussions sur l'efficacité d'une telle implémentation sont repoussées au chapitre suivant, à propos de *l'algorithme FFT*. Voici donc la fameuse idée qui est à la base de notre algorithme. Il s'agit de réécrire l'équation (2.3), de la façon suivante :

$$\mathscr{W}_k(f)[i] = \sum_{j=0}^{2^{k-1}-1} f[j](-1)^{\sum_{p=0}^{k-2} j_p i_p} + (-1)^{i_{k-1}} \sum_{j=0}^{2^{k-1}-1} f[j+2^{k-1}](-1)^{\sum_{p=0}^{k-2} j_p i_p}.$$

Pour écrire cette expression d'une façon plus simple, introduisons les vecteurs f_1 et f_2 de longueur 2^{k-1} définis de la manière suivante :

$$\forall j \in \{0,\dots,2^{k-1}-1\}, \quad f_1[j] = f[j] \quad \text{et} \quad f_2[j] = f[j+2^{k-1}].$$

De même, on écrira $\mathscr{W}_k(f)_1$ (respectivement $\mathscr{W}_k(f)_2$) pour désigner les 2^{k-1} premiers (respectivement derniers) indices du vecteur transformé $\mathscr{W}_k(f)$. On a alors l'équation de récurrence

$$\begin{aligned} \mathscr{W}_k(f)_1 &= \mathscr{W}_{k-1}(f_1) + \mathscr{W}_{k-1}(f_2) \\ \mathscr{W}_k(f)_2 &= \mathscr{W}_{k-1}(f_1) - \mathscr{W}_{k-1}(f_2). \end{aligned}$$

D'un point de vue matriciel cette décomposition s'écrit sous la forme

$$W_{2^k} = \begin{pmatrix} W_{2^{k-1}} & W_{2^{k-1}} \\ W_{2^{k-1}} & -W_{2^{k-1}} \end{pmatrix}.$$

La décomposition de W_{2^k} trouvée correspond à une structure de produit tensoriel, comme le précise l'exercice II.7.Cette décomposition donne tout naturellement naissance à un algorithme de calcul très rapide, nommé *FWT* pour **F**ast **W**alsh **T**ransform. De façon plus précise, si l'on compte le nombre d'additions nécessaires pour calculer la transformée de Walsh d'un vecteur de taille $n = 2^k$, on voit que l'on obtient $k = \log_2(n)$ appels récursifs, avec à chaque fois n additions ou soustractions. D'où un coût de $n\log_2(n)$ opérations. C'est un gain substantiel par rapport à l'implémentation naïve de l'équation (2.3), qui nécessite n^2 opérations. Le programme MATLAB de cet algorithme est présenté au paragraphe 1, annexe A.

2.3 Utilisation de la transformée de Walsh

L'intérêt principal de la transformée de Walsh est qu'elle permet de décomposer n'importe quelle fonction de $\{0,\dots,2^k-1\}$ dans $\mathbb{C}$ sur la base orthogonale des caractères de $(\mathbb{F}_2)^k$.

De façon plus précise, on a

$$\forall i \in \{0,\dots,2^k-1\}, \quad f[i] = \frac{1}{2^k}\sum_{j=0}^{2^{k-1}} \mathscr{W}_k(f)[j]\chi_j(i).$$

Cette transformée est très rapide à utiliser, car elle n'emploie que des additions et des soustractions (pas de multiplication). De plus, nous avons vu que l'on dispose d'un algorithme redoutablement efficace pour calculer de façon récursive la transformée de Walsh. Cependant, la transformée de Walsh à un point faible de taille : elle n'a aucune propriété agréable vis-a-vis de la convolution des fonctions de $\{0,\dots,2^k-1\}$ dans $\mathbb{C}$, contrairement à la transformée de Fourier, comme nous l'avons vu au paragraphe 4.3, chap. I. En effet, la transformée de Walsh est une transformée de Fourier sur le cube binaire $(\mathbb{Z}/2\mathbb{Z})^k$, pas du tout sur le groupe cyclique $\mathbb{Z}/2^k\mathbb{Z}$. C'est essentiellement à cause de ceci que nous allons être amenés au chapitre suivant à étudier de plus près la transformée de Fourier sur un groupe cyclique. Ceci va nous permettre de construire un algorithme pour calculer des convolutions sur $\mathbb{Z}/2^k\mathbb{Z}$ de façon extrêmement rapide.

L'exercice II.5 montre comment on peut utiliser la transformée de Walsh pour réaliser de la *compression de signaux*. Il propose aussi d'étendre la transformée de Walsh au cadre bidimensionnel.

Enfin, la transformée de Walsh permet d'étudier les *fonctions booléennes*, et en particulier leur *non-linéarité*. Comme cette étude suppose de considérer des fonctions à valeurs dans le corps fini $\mathbb{F}_2$, elle n'est abordée qu'à la fin du chapitre VI, à l'exercice VI.4. Dans le même ordre d'idées, l'exercice VI.5 introduit des notions probabilistes pour étudier les fonctions booléennes et leur apprentissage.

3 Formule de Poisson

Nous avons vu, notamment au paragraphe 2.3, chap. I consacré au bidual, la grande similarité entre la dualité sur un groupe fini abélien, et la dualité sur un espace vectoriel. Dans ce paragraphe, nous allons donner une autre incarnation de ce fait, en l'occurrence en étudiant la notion d'orthogonalité entre un groupe et son dual. Le point central de cette approche est la *formule de Poisson*. Nous allons ainsi voir que si l'on applique cette formule dans le cas d'un groupe qui est aussi un espace vectoriel (sur un corps fini), on obtient des relations très puissantes, nommées identités de *MacWilliams*.

3.1 La formule sur un groupe fini abélien

Avant d'énoncer la formule de Poisson sur un groupe fini, il est nécessaire de clarifier la notion d'orthogonalité. Commençons par rappeler brièvement la théorie de l'orthogonalité sur un espace vectoriel. Pour une plus ample description, on pourra se référer à l'ouvrage de RAMIS, DECHAMPS et ODOUX [59].

Si E est un k-espace vectoriel de dimension finie, on note $E^* \overset{\text{déf.}}{=} \mathrm{Hom}(E,k)$ son dual, qui est constitué des formes linéaires. On définit classiquement une forme bilinéaire sur $E^* \times E$, que l'on nomme crochet de la dualité, de la façon suivante :

$$\forall (f,x) \in E^* \times E, \quad \langle x, f\rangle \overset{\text{déf.}}{=} f(x).$$

On définit alors, pour une partie $A \subset E$, son orthogonal

$$A^{\perp} \overset{\text{déf.}}{=} \{f \in E^* \setminus \forall x \in A,\ \langle x, f\rangle = 0\}. \tag{3.1}$$

On vérifie que c'est un sous-espace vectoriel de E, et l'on a $E^{\perp} = \{0\}$. De même, on définit l'orthogonal de $B \subset E^*$ par

$$B^0 = \{x \in E \setminus \forall f \in B,\ \langle x, f\rangle = 0\}.$$

C'est un sous-espace vectoriel de E^*, et on a $(E^*)^0 = \{0\}$. Notons que ces propriétés sont encore vraies en dimension infinie, mais on doit utiliser l'axiome du choix. En dimension *finie*, les applications $F \mapsto F^{\perp}$ et $G \mapsto G^0$ sont des bijections réciproques entre sous-espaces de E et E^*. Elles renversent l'inclusion.

Une fois en mémoire ces notions linéaires d'orthogonalité, il est naturel d'introduire la définition suivante :

Définition 3.1 (Orthogonal d'un sous-groupe). Soient G un groupe fini abélien, et $H \subset G$ un sous-groupe. On note $H^{\sharp}$ l'orthogonal de H qui est le sous-groupe de $\widehat{G}$ défini de la manière suivante :

$$H^{\sharp} \overset{\text{déf.}}{=} \left\{\chi \in \widehat{G} \setminus \forall h \in H,\ \chi(h) = 1\right\}.$$

Nous avons déjà vu lors de la démonstration du lemme 2.2, chap. I, que tout caractère de $\widehat{G}$ trivial sur H s'identifie de manière unique à un élément de $\widehat{G/H}$, et réciproquement. De façon plus précise, on a un isomorphisme $H^{\sharp} \simeq \widehat{G/H}$. On constate donc que $H^{\sharp}$ est un sous-groupe de $\widehat{G}$ de cardinal $|G|/|H|$. Par exemple, on a $G^{\sharp} = \{1\}$. De plus, l'application $H \mapsto H^{\sharp}$ renverse les inclusions. Ici encore, la ressemblance avec la dualité entre les espaces vectoriels est frappante.

On peut maintenant énoncer la *formule de Poisson* dans le cadre des groupes abéliens finis.

Théorème 3.2 (Formule de Poisson). *Soit G un groupe abélien fini, et $H \subset G$ un sous-groupe. Alors, pour $f : G \to \mathbb{C}$, c'est-à-dire $f \in \mathbb{C}[G]$, on a*

$$\forall g \in G, \quad \sum_{h \in H} f(gh) = \frac{|H|}{|G|} \sum_{\chi \in H^{\sharp}} \widehat{f}(\overline{\chi})\chi(g). \tag{3.2}$$

On a noté $\widehat{f} : \widehat{G} \to \mathbb{C}$ la transformée de Fourier de f.

Démonstration. Pour simplifier les notations, on considère S un système de représentants de G/H dans G. On note $\overline{g}$ l'image de $g \in S$ dans G/H (c'est-à-dire l'image par la projection canonique). Commençons par définir une fonction $\widetilde{f}$ sur G/H par

$$\forall g \in S, \quad \widetilde{f}(\overline{g}) \overset{\text{déf.}}{=} \sum_{h \in H} f(gh).$$

Ceci revient à remplacer f par une fonction invariante sous la translation par des éléments de H (on la « périodise »). On constate que cette fonction est définie sans ambiguïté, puisque, si l'on considère un autre représentant g' de la classe gH, on a $g' = gh'$ avec h' un élément de H, et en conséquence, $\widetilde{f}(\overline{g}) = \widetilde{f}(\overline{g'})$. On peut donc décomposer la fonction $\widetilde{f} \in \mathbb{C}[G/H]$ en série de Fourier :

$$\forall g \in S, \quad \widetilde{f}(\overline{g}) = \sum_{\chi \in \widehat{G/H}} \left\langle \widetilde{f}, \chi \right\rangle \chi(\overline{g}). \tag{3.3}$$

On peut alors expliciter la valeur des coefficients de Fourier :

$$\left\langle \widetilde{f}, \chi \right\rangle = \frac{1}{|G/H|} \sum_{g \in S} \widetilde{f}(\overline{g}) \overline{\chi(\overline{g})} = \frac{|H|}{|G|} \sum_{g \in S} \sum_{h \in H} f(gh) \overline{\chi(\overline{g})}. \tag{3.4}$$

En remarquant que l'application

$$\begin{cases} S \times H & \longrightarrow & G \\ (g,h) & \longmapsto & gh \end{cases}$$

est une bijection et en utilisant le fait que pour $\chi \in \widehat{G/H}$, $\chi(\overline{gh}) = \chi(\overline{g})$, on peut réécrire la somme (3.4) sous la forme

$$\left\langle \widetilde{f}, \chi \right\rangle = \frac{|H|}{|G|} \sum_{g \in G} f(g) \overline{\chi(\overline{g})} \overset{\text{déf.}}{=} \frac{|H|}{|G|} \widehat{f}(\overline{\chi}).$$

Il ne reste plus qu'à reporter la valeur de ces coefficients dans l'équation (3.3) et de remarquer que l'expression de $\widetilde{f}(\overline{g})$ nous donne le membre de gauche de la formule de Poisson. □

En appliquant l'équation (3.2) avec $g = 1$, on obtient la forme sous laquelle la formule est souvent écrite :

$$\sum_{h \in H} f(h) = \frac{|H|}{|G|} \sum_{\chi \in H^\perp} \widehat{f}(\chi). \tag{3.5}$$

Dans le but de mieux comprendre la formule de Poisson, nous allons donner une autre preuve, qui utilise uniquement des arguments d'algèbre linéaire sur l'espace vectoriel $\mathbb{C}[G]$. Avant de donner cette preuve, étudions plus précisément l'espace $\mathbb{C}[G/H]$.
Si on note $\pi : G \to G/H$ la projection canonique, on peut définir une application

$$\pi^* : \begin{cases} \mathbb{C}[G/H] & \longrightarrow & \mathbb{C}[G] \\ f & \longmapsto & f \circ \pi \end{cases}.$$

π^* est en fait une application linéaire injective (car π est surjective). L'espace $\mathbb{C}[G/H]$ s'identifie donc à un sous-espace vectoriel de $\mathbb{C}[G]$, qui est en fait formé des fonctions constantes sur chacune des classes à gauche modulo H. Pour démontrer ce fait, il suffit de constater que π^* peut s'inverser sur son image de la façon suivante :

$$(\pi^*)^{-1} \begin{cases} \mathrm{Im}(\pi^*) & \longrightarrow & \mathbb{C}[G/H] \\ f & \longmapsto & \widetilde{f} \end{cases},$$

où on note $\widetilde{f}(\overline{x})$ (pour $x \in G$) la valeur de f sur la classe à gauche de x modulo H. Cette identification étant faite, on peut donner une nouvelle démonstration de la formule de Poisson.

Démonstration. Comme précédemment, on se fixe une fonction $f \in \mathbb{C}[G]$. Rappelons que l'espace $\mathbb{C}[G]$ est muni d'une structure d'algèbre grâce au produit de convolution $*$, dont l'expression est donnée à l'équation (4.6), chap. I. On peut alors considérer un opérateur de filtrage

$$\Phi^f : \begin{cases} \mathbb{C}[G] & \longrightarrow & \mathbb{C}[G] \\ \varphi & \longmapsto & f * \varphi \end{cases}.$$

Nous verrons à la section 2, chap. IV, pourquoi on nomme de tels opérateurs des opérateurs de filtrage. En utilisant l'identification entre les fonctions de $\mathbb{C}[G/H]$ et les fonctions

de $\mathbb{C}[G]$ constantes sur les classes à gauche gH, on peut montrer que $\mathbb{C}[G/H]$ est un sous-espace de $\mathbb{C}[G]$ stable par Φ^f. En effet, si on se donne φ constante sur les classes à gauche, et $g' = gh'$, pour $g \in G$ et $h' \in G$, on obtient

$$\Phi^f(\varphi)(g') = \sum_{x\in G} f(gh'x^{-1})\varphi(x) = \sum_{x\in G} f(g(xh'^{-1})^{-1})\varphi(xh'^{-1}) = \Phi^f(\varphi)(g).$$

Ceci étant montré, on peut donc considérer $\widetilde{\Phi}^f$, la restriction de Φ^f à $\mathbb{C}[G/H]$. L'astuce, pour trouver la formule de Poisson, est de calculer la trace de $\widetilde{\Phi}^f$. On dispose d'une base évidente de $\mathbb{C}[G/H]$, à savoir $\mathscr{B} \overset{\text{déf.}}{=} \{\delta_g\}_{g\in S}$ (on note toujours S un système de représentant de G/H). On rappelle que δ_g est la fonction qui vaut 1 en g, et 0 partout ailleurs. Dans cette base, l'expression de la trace de l'opérateur est simple à calculer, puisque le calcul des images des vecteurs de base donne

$$\forall s \in S, \quad \widetilde{\Phi}^f(\delta_s) : g \mapsto \sum_{x\in S}\sum_{h\in H} f(gx^{-1}h^{-1})\delta_s(x),$$

car il faut se rappeler que δ_s est vu comme une fonction constante sur les classes à gauche. En conséquence, on obtient

$$\forall s \in S, \quad \widetilde{\Phi}^f(\delta_s) : g \mapsto \sum_{h\in H} f(gs^{-1}h^{-1}).$$

La trace se calcule donc sans effort :

$$\operatorname{tr}\left(\widetilde{\Phi}^f\right) = \sum_{s\in S}\sum_{h\in H} f(ss^{-1}h^{-1}) = \frac{|G|}{|H|}\sum_{h\in H} f(h).$$

A une constante près, on obtient le membre de gauche de la formule de Poisson. Pour obtenir le membre de droite, il va suffire de calculer la trace de $\widetilde{\Phi}^f$ dans une autre base. Et bien sûr, nous allons réinvestir le travail effectué au chapitre précédent en choisissant la base orthonormale des caractères de G/H, c'est-à-dire les éléments de $H^\natural$. L'intérêt est que les caractères se comportent de façon particulièrement agréable vis-a-vis de la convolution. En effet, le théorème de convolution 4.15, chap. I, permet de montrer que les éléments de $H^\natural$ sont les vecteurs propres de l'opérateur $\widetilde{\Phi}^f$, puisque

$$\forall \chi \in \widehat{G/H} = H^\natural, \quad \widetilde{\Phi}^f(\chi) = \widehat{f}(\chi)\chi.$$

L'exercice I.1, question 1, détaille la démontration de ceci. Il ne reste plus qu'à exploiter le fait que la trace de l'opérateur $\widetilde{\Phi}^f$ est égale à la somme de ses valeurs propres, pour conclure que

$$\operatorname{tr}\left(\widetilde{\Phi}^f\right) = \sum_{\chi\in H^\natural} \widehat{f}(\chi).$$

On retrouve donc bien la formule de Poisson simplifiée (3.5). Pour obtenir l'équation complète (3.2), il suffit d'appliquer l'équation obtenue à la fonction $f_g : x \to f(gx)$, pour $g \in G$, puisque l'on a

$$\forall \chi \in \widehat{G}, \quad \widehat{f_g}(\chi) = \chi(g^{-1})\widehat{f}(\chi).$$

Ce raisonnement montre d'ailleurs que les équations (3.5) et (3.2) sont complètement équivalentes. □

3.2 Application aux identités de MacWilliams

Dans ce paragraphe, nous allons utiliser la formule de Poisson dans le cadre du groupe G égal à $(\mathbb{Z}/2\mathbb{Z})^k = (\mathbb{F}_2)^k$. Nous allons donc nous placer dans le cadre où nous avons mené l'étude de la transformée de Walsh (section 2), et nous allons employer les mêmes notations. L'intérêt d'un tel groupe est qu'il est en plus un espace vectoriel, en l'occurrence sur le corps fini $\mathbb{F}_2$. En réalité, la situation est très simple, puisque la notion de sous-groupe coïncide avec celle de sous-espace vectoriel (l'opération de groupe correspond à l'addition des vecteurs, et comme le corps n'a que deux éléments, la multiplication d'un vecteur par un scalaire est une opération qui respecte trivialement la structure de sous-groupe). On ne perd donc pas d'information en traduisant les énoncés issus de la dualité sur un groupe (la formule de Poisson) dans le langage de l'algèbre linéaire. Nous allons voir que dans ce cadre, les notions de dualité et d'orthogonalité sont les mêmes pour les deux structures.

Toute l'étude faite dans ce paragraphe se généralise sans modification au cas de l'espace vectoriel K^k, où K est un corps fini quelconque. L'exercice II.8 reprend étape par étape cette construction. Cependant, pour rester dans le cadre développé pour la transformée de Walsh, nous allons nous restreindre au cas de l'espace $(\mathbb{F}_2)^k$.

On rappelle que l'on a une description complète du dual $\widehat{F_2^k}$, puisqu'à chaque élément $a = \{a_0, \ldots, a_{k-1}\} \in (\mathbb{Z}/2\mathbb{Z})^k$ on fait correspondre un caractère

$$\chi_a : \begin{cases} (\mathbb{Z}/2\mathbb{Z})^k & \longrightarrow & \{-1,1\} \\ x = \{x_0, \ldots, x_{k-1}\} & \longmapsto & (-1)^{\langle a,x\rangle} \end{cases}.$$

Ceci nous permet de calculer, pour un groupe $H \subset G$, l'orthogonal $H^\sharp$. En effet, dire que $\chi_a \in H^\sharp$ est équivalent à

$$\forall h \in H, \quad \chi_a(h) = (-1)^{\langle a,h\rangle} = 1.$$

Ceci signifie donc que

$$\forall h \in H, \quad \langle a,h\rangle = 0 \Leftrightarrow a \in H^\perp, \tag{3.6}$$

où l'on a noté $H^\perp$ l'orthogonal de H lorsque l'on considère H comme un sous-espace vectoriel de $(\mathbb{F}_2)^k$. Cet orthogonal peut être vu bien sûr comme l'orthogonal pour la forme bilinéaire symétrique canonique sur $(\mathbb{F}_2)^k$. On peut aussi le voir comme l'orthogonal au sens de la dualité (définition (3.1)), si l'on a identifié l'espace vectoriel $(\mathbb{F}_2)^k$ et son dual en identifiant la base canonique à sa base duale. Il faut faire attention cependant au fait que l'espace orthogonal $H^\perp$ n'a aucune raison d'être un supplémentaire de H, par exemple, dans $(\mathbb{F}_2)^4$, le vecteur $(1,1,1,1)$ est orthogonal à lui même. L'exercice VIII.9 étudie justement les cas où l'espace H coïncide avec son dual (il utilise le langage des codes correcteurs d'erreurs et des actions de groupes).

Au final, si l'on identifie les éléments $a \in G$ et $\chi_a \in \widehat{G}$, on obtient la propriété remarquable que $H^\sharp = H^\perp$. On peut maintenant énoncer la formule de Poisson en terme d'espaces vectoriels.

Proposition 3.3 (Formule de Poisson vectorielle). *Soit H un sous-espace vectoriel de $(\mathbb{F}_2)^k$. Soit f une fonction $f : (\mathbb{F}_2)^k \to \mathbb{C}$. On a alors les deux relations*

$$\sum_{a\in H} f(a) = \frac{|H|}{2^k} \sum_{u\in H^\sharp} \widehat{f}(\chi_u). \tag{3.7}$$

$$\sum_{a\in H^\sharp} f(a) = \frac{1}{|H|} \sum_{u\in H} \widehat{f}(\chi_u). \tag{3.8}$$

On rappelle que

$$\widehat{f}(\chi_u) \overset{\text{déf.}}{=} \sum_{x\in(\mathbb{F}_2)^k} (-1)^{\langle u,x\rangle} f(x).$$

Démonstration. La première équation est l'exacte traduction de la formule de Poisson, avec l'identification $H^\perp = H^\sharp$ que l'on a mise à jour. Pour la deuxième équation, il suffit de remplacer dans la première H par son orthogonal $H^\perp$. Comme $|H^\perp| = \frac{|(\mathbb{F}_2)^k|}{|H|}$, on obtient bien le résultat voulu. □

Nous allons maintenant pouvoir appliquer la formule de Poisson à des fins combinatoires. Le but de cette étude est de mieux comprendre la structure de l'espace $(\mathbb{F}_2)^k$ lorsqu'on le munit d'une distance un peu particulière, que l'on nomme distance de *Hamming*. Pour l'instant, il s'agit surtout d'un exercice calculatoire, qui va permettre de révéler des relations assez spectaculaires. Cependant, nous verrons à la section 3, chap. VI, que tout ceci a des applications dans l'étude des codes correcteurs binaires. Mais contentons nous, dans un premier temps, de définir cette fameuse distance.

Définition 3.4 (Distance de Hamming). Soit x et $y \in (\mathbb{F}_2)^k$. On définit la *distance de Hamming* $d(x,y)$ entre ces deux vecteurs de la façon suivante :

$$d(x,y) \overset{\text{déf.}}{=} w(x-y) \quad \text{avec} \quad w(z) = \sharp\{i = 0,\dots,k-1 \setminus z_i \neq 0\}.$$

On appelle $w(z)$ le poids du vecteur z.

La figure 2.1 représente le groupe $(\mathbb{F}_2)^4$ où l'on a relié les éléments à une distance de 1. Dans le but d'étudier la structure d'un sous-espace H vis-a-vis de la distance d, on s'intéresse à la répartition des poids des mots qui forment H. On introduit alors les définitions suivantes.

Définition 3.5 (Polynôme énumérateur). Soit H un sous-espace vectoriel de $(\mathbb{F}_2)^k$. On note $A_H \in \mathbb{Z}[X,Y]$ le *polynôme énumérateur de poids* de H, qui est défini par

$$A_H(X,Y) \overset{\text{déf.}}{=} \sum_{c\in H} X^{k-w(c)}Y^{w(c)} = \sum_{i=0}^{k} A_i X^{k-i}Y^i,$$

où on a noté A_i le nombre de vecteurs de H de poids i.

La relation suivante, découverte par MACWILLIAMS, met en relation les poids des mots de l'espace H avec les poids des mots de son orthogonal.

Théorème 3.6 (Identité de MacWilliams). *Soit H un sous-espace vectoriel de $(\mathbb{F}_2)^k$. On a alors*

$$A_{H^\perp}(X,Y) = \frac{1}{|H|} A_H(X+Y, X-Y).$$

Démonstration. Soient x et y deux nombres complexes fixés. On définit alors la fonction $f \in \mathbb{C}[(\mathbb{F}_2)^k]$ par

$$\forall a \in (\mathbb{F}_2)^k, \quad f(a) \overset{\text{déf.}}{=} x^{k-w(a)}y^{w(a)}.$$

Dans le but d'appliquer la formule de Poisson, il nous faut calculer $\widehat{f}$:

$$\forall a \in (\mathbb{F}_2)^k, \quad \widehat{f}(\chi_a) \overset{\text{déf.}}{=} \sum_{t\in(\mathbb{F}_2)^k} x^{k-w(t)}y^{w(t)}(-1)^{\langle t,a\rangle}.$$

En utilisant le fait que $w(t) = \sum_{i=0}^{k-1} t_i$ dans $\mathbb{N}$, où $t_i \in \{0,1\}$, on obtient

$$\forall a \in (\mathbb{F}_2)^k, \quad \widehat{f}(\chi_a) = \sum_{t\in(\mathbb{F}_2)^k} \prod_{i=0}^{k-1} x^{1-t_i} y^{t_i} (-1)^{a_i t_i} = \prod_{i=0}^{k-1} \sum_{t_i=0}^{1} x^{k-t_i} y^{t_i} (-1)^{a_i t_i}.$$

Si $a_i = 0$, la somme intérieure vaut $x+y$, alors que si $a_i = 1$, on trouve $x-y$. On a donc

$$\forall a \in (\mathbb{F}_2)^k, \quad \widehat{f}(\chi_a) = (x+y)^{k-w(a)}(x-y)^{w(a)}.$$

On peut maintenant appliquer la formule de Poisson (3.8). L'égalité du théorème est ainsi vraie si on considère les valeurs des polynômes quel que soit le point $(x,y) \in \mathbb{C}^2$. Elle est donc aussi vraie en tant qu'égalité polynomiale sur $\mathbb{Z}[X,Y]$. □

Cette identité, outre son intérêt esthétique certain, constitue l'outil principal pour l'étude combinatoire des codes correcteurs. La section 4, chap. VI, reformule l'identité de Mac-Williams dans le cadre de la théorie des codes, et explique les multiples applications qui en découlent.

3.3 La formule de Poisson continue

La formule de Poisson que nous venons d'utiliser sur un groupe fini abélien a en fait un énoncé semblable dans le cadre des fonctions continues définies sur $\mathbb{R}$. Pour que cet énoncé soit agréable, nous allons définir la transformée de Fourier continue de la manière suivante :

$$\forall f \in L^1(\mathbb{R}), \forall x \in \mathbb{R}, \quad \widehat{f}(x) \overset{\text{déf.}}{=} \int_{\mathbb{R}} f(t) e^{-2i\pi t x} \mathrm{d}t. \tag{3.9}$$

On peut alors énoncer la formule de Poisson sur $\mathbb{R}$. On notera avec beaucoup d'intérêt que sa démonstration est en grande partie semblable à celle faite dans le cadre des groupes finis. La seule difficulté du cas continu réside dans les problèmes de convergence des séries manipulées, ce qui nous oblige à imposer des hypothèses plus contraignantes sur les fonctions que l'on analyse.

Théorème 3.7 (Formule de Poisson continue). *Soit $f \in L^1(\mathbb{R})$ une fonction continue telle que*

$$\exists M > 0, \exists \alpha > 1, \quad |f(x)| \leqslant M(1+|x|)^{-\alpha}, \tag{3.10}$$

$$\sum_{n=-\infty}^{+\infty} |\widehat{f}(n)| < +\infty. \tag{3.11}$$

Sous ces hypothèses, on a

$$\sum_{n=-\infty}^{+\infty} f(n) = \sum_{n=-\infty}^{+\infty} \widehat{f}(n). \tag{3.12}$$

Démonstration. On commence par périodiser la fonction f en introduisant la fonction

$$\forall x \in \mathbb{R}, \quad f_1(x) \overset{\text{déf.}}{=} \sum_{n=-\infty}^{+\infty} f(x+n)$$

Si on se restreint au compact $\{x \in \mathbb{R}, |x| \leqslant A\}$, pour $A > 0$, l'hypothèse (3.10) permet d'affirmer, pour $|n| \geqslant 2A$,

$$|f(x+n)| \leqslant M(1+|x+n|)^{-\alpha} \leqslant M(1+|n|-A)^{-\alpha} \leqslant M(1+|n|/2)^{-\alpha},$$

ce qui définit le terme général d'une série convergente. On en conclut donc que sur tout compact, la série qui définit f_1 est normalement convergente, donc que f_1 est continue. De plus, on voit facilement que f_1 est 1-périodique, puisque

$$f_1(x+1) = \sum_{n=-\infty}^{+\infty} f(x+n+1) = \sum_{n'=-\infty}^{+\infty} f(x+n') = f_1(x),$$

où l'on a fait le changement de variable $n' = n+1$ (autorisé par l'absolue convergence de la série). On peut donc calculer ses coefficients de Fourier :

$$\forall m \in \mathbb{Z}, \quad c_m(f_1) = \int_0^1 f_1(t) e^{-2\mathrm{i}m\pi t} \mathrm{d}t = \sum_{n\in\mathbb{Z}} \int_0^1 f(t+n) e^{-2\mathrm{i}m\pi t} \mathrm{d}t,$$

où l'interversion entre la somme et l'intégrale est justifiée par la convergence normale de la série de terme général $f(t+n)e^{-2\mathrm{i}m\pi t}$ pour $t \in [0,1]$. On peut ainsi poursuivre les calculs pour obtenir, par changement de variable $u = t+n$,

$$\forall m \in \mathbb{Z}, \quad c_m(f_1) = \sum_{n\in\mathbb{Z}} \int_n^{n+1} f(u) e^{-2\mathrm{i}m\pi u} \mathrm{d}u = \int_{-\infty}^{+\infty} f(u) e^{-2\mathrm{i}m\pi u} \mathrm{d}u = \widehat{f}(m)$$

(la dernière égalité est justifiée par le théorème de convergence dominé de Lebesgue, car $x \mapsto f(x)e^{-2\mathrm{i}m\pi x} \in L^1(\mathbb{R})$). On constate donc, avec l'hypothèse (3.11) que la série de Fourier associée à f_1 converge absolument. En utilisant en plus le fait que f_1 est une fonction continue, on conclut donc qu'elle est somme de sa série de Fourier. On peut donc écrire

$$\forall x \in \mathbb{R}, \quad f_1(x) = \sum_{m\in\mathbb{Z}} c_m(f_1) e^{2\mathrm{i}m\pi x} = \sum_{m\in\mathbb{Z}} \widehat{f}(m) e^{2\mathrm{i}m\pi x}.$$

Au final, on obtient donc l'égalité

$$\forall x \in \mathbb{R}, \quad \sum_{n\in\mathbb{Z}} f(x+n) = \sum_{m\in\mathbb{Z}} \widehat{f}(m) e^{2\mathrm{i}m\pi x},$$

ce qui donne bien la formule de Poisson voulue en faisant $x = 0$. □

***Remarque* 3.8. (Lien avec la formule de Poisson sur un groupe fini).** La formule de Poisson continue que nous venons de démontrer est en tout point semblable à la formule (3.2), valable sur un groupe fini abélien. En effet, dans le cas continu, il faut considérer le groupe $G = \mathbb{R}^1$, qui est la droite réelle munie de l'addition, ainsi que le sous-groupe discret $\mathbb{Z} \subset \mathbb{R}$. Le groupe quotient n'est rien d'autre que le cercle $\mathbb{R}/\mathbb{Z} \simeq S^1$. De plus, nous verrons au paragraphe 1.1, chap. IV, que l'on dispose d'une description complète des caractères du cercle, puisqu'ils correspondent aux exponentielles $e_n : t \mapsto e^{2\mathrm{i}n\pi t}$. On dispose donc d'un isomorphisme explicite $\widehat{S^1} \simeq \mathbb{Z}$. Dès lors, on peut écrire la formule de Poisson dans le cas continu sous la forme

$$\sum_{n\in\mathbb{Z}} f(n) = \sum_{e_n \in \widehat{\mathbb{R}/\mathbb{Z}}} \langle f, e_n \rangle.$$

Donc au facteur $\frac{|H|}{|G|}$ près, cette formule est en tout point semblable à (3.2).

Une des nombreuses applications de la formule de Poisson concerne la fonction *Thêta* de *Jacobi*, qui est définie de la manière suivante.

Définition 3.9 (Fonction Thêta de Jacobi). On définit la fonction *Thêta* par

$$\forall t > 0, \quad \theta(t) \overset{\text{déf.}}{=} \sum_{n=-\infty}^{+\infty} e^{-\pi n^2 t}.$$

Avant d'énoncer l'équation fonctionnelle que vérifie θ, il nous faut démontrer un lemme classique sur la transformée de Fourier d'une Gaussienne.

Lemme 3.10. *Soit g_t, pour $t > 0$, la Gaussienne définie par*

$$\forall x \in \mathbb{R}, \quad g_t(x) = e^{-\frac{x^2}{2t}}.$$

Alors on a

$$\forall x \in \mathbb{R}, \quad \widehat{g_t}(x) = \sqrt{2\pi t} e^{-2\pi^2 t x^2}$$

où l'on a conservé la définition de la transformée de Fourier (3.9).

Démonstration. La transformée de la fonction g_t s'écrit

$$\widehat{g_t}(x) = \int_{\mathbb{R}} e^{-\frac{u^2}{2t}} e^{-2i\pi u x} \mathrm{d}u.$$

En utilisant le théorème de dérivation sous le signe somme, on voit que la fonction obtenue est C^1, et que l'on a

$$\frac{\mathrm{d}\widehat{g_t}}{\mathrm{d}x}(x) = -2i\pi \int_{\mathbb{R}} u e^{-\frac{u^2}{2t}} e^{-2i\pi u x} \mathrm{d}u = -4\pi^2 x t \widehat{g_t}(x),$$

où la dernière égalité s'obtient par intégration par parties. En résolvant l'équation différentielle dont g_t est solution, on obtient

$$\widehat{g_t}(x) = \widehat{g_t}(0) e^{-2\pi^2 t x^2}.$$

Il ne reste plus qu'à calculer la valeur de $\widehat{g_t}(0) = \sqrt{t} I$, avec $I = \int_{\mathbb{R}} e^{-x^2/2} \mathrm{d}x$. Pour ce faire, il convient de passer en coordonnées polaires lors du calcul de I^2 :

$$I^2 = \int_{\mathbb{R}} \int_{\mathbb{R}} e^{-\frac{x^2+y^2}{2}} \mathrm{d}x\mathrm{d}y = 2\pi \int_0^{+\infty} r e^{-\frac{r^2}{2}} \mathrm{d}r = 2\pi.$$

En mettant bout à bout tous ces résultats, on obtient bien la transformée de Fourier annoncée. □

Voici enfin l'identité de Jacobi sur la fonction θ.

Théorème 3.11 (Identité de Jacobi).

$$\forall t > 0, \quad \theta(t) = \frac{1}{\sqrt{t}} \theta\left(\frac{1}{t}\right) \tag{3.13}$$

Démonstration. Il suffit d'appliquer la formule de Poisson à la fonction g_t. Il est évident que g_t vérifie bien les hypothèses du théorème 3.7. On obtient ainsi l'égalité suivante :

$$\sum_{n\in\mathbb{Z}} g_t(n) = \sum_{n\in\mathbb{Z}} e^{-\frac{n^2}{2t}} = \sum_{n\in\mathbb{Z}} \widehat{g_t}(n) = \sqrt{2\pi t} \sum_{n\in\mathbb{Z}} e^{-2\pi^2 t n^2}.$$

Ce n'est rien d'autre que l'identité que l'on cherchait à démontrer, évaluée en $2\pi t$. □

L'un des intérêts de cette identité est de permettre de calculer la fonction θ pour des petites valeurs du paramètre t, et ceci avec une précision très grande. Par exemple, pour $t = 0.001$, le membre de droite de (3.13) fournit instantanément le résultat avec une précision supérieure à celle de MATLAB en double précision (qui est donnée par le commande `eps=2.2204e-016`), et ceci avec simplement le terme d'indice $n = 0$ dans la somme. Si on utilise de façon naïve le membre de gauche de l'identité, on observe une diminution géométrique de l'erreur relativement lente.

4 Exercices

Exercice II.1 (Preuve géométrique de la réciprocité quadratique). Cet exercice est tiré d'un article de LAUBENBACHER [45]. Nous allons détailler pas à pas la preuve de la réciprocité quadratique qu'EISENSTEIN a donnée en 1844. Cette preuve est assez originale puisqu'elle utilise essentiellement des arguments de nature géométrique. Dans la suite de cet exercice, on note $[x]_p$ le reste de la division euclidienne de x par p (qui est toujours un élément de $\{0,\dots,p-1\}$, même si $x \leqslant 0$), et $\lfloor x \rfloor$ la partie entière de x. On considère deux nombres premiers impairs distincts p et r, et on souhaite démontrer la formule de réciprocité quadratique (1.18).

1. On note $A \overset{\text{déf.}}{=} \{2,4,6,\dots,p-1\}$ et $B \overset{\text{déf.}}{=} \{[ra]_p \setminus a \in A\}$. Montrer que l'on a
$$A = \left\{ \left[(-1)^b b\right]_p \setminus b \in B \right\}.$$
2. En déduire que modulo p, on a l'égalité suivante :
$$\prod_{b\in B} b = r^{\frac{p-1}{2}} \prod_{a\in A} a = r^{\frac{p-1}{2}} (-1)^{\sum_{b\in B} b} \prod_{b\in B} b \mod p,$$
puis que
$$\left(\frac{r}{p}\right) = (-1)^{\sum_{b\in B} b}.$$
3. Démontrer l'égalité suivante :
$$\sum_{a\in A} ra = p \sum_{a\in A} \left\lfloor \frac{ra}{p} \right\rfloor + \sum_{b\in B} b.$$
En déduire
$$\left(\frac{r}{p}\right) = (-1)^{\sum_{a\in A} \left\lfloor \frac{ra}{p} \right\rfloor}.$$
4. Dans le but de donner une signification géométrique à cette équation, on construit la figure 2.3. Montrer qu'aucun point à coordonnées entières ne se trouve sur $]AB[$. Montrer alors que le nombre de points d'abscisse paire dans le triangle ABD est égal à $\sum_{a\in A} \left\lfloor \frac{ra}{p} \right\rfloor$.
5. On considère une abscisse entière $a > \frac{p}{2}$. Montrer que, modulo 2, le nombre de points d'abscisse a situés en dessous de (AB) (marqués + sur la figure) est égal au nombre de points de même abscisse mais situés au-dessus de (AB) (marqués $\times$). Montrer que ce nombre est aussi égal au nombre de points d'abscisse $p-a$ situés au-dessous de (AB) (notés $\bullet$). En conclure que $\sum_{a\in A} \left\lfloor \frac{ra}{p} \right\rfloor$ est égal, modulo 2, au nombre de points à coordonnées entières dans l'intérieur du triangle AHK.
6. En échangeant les rôles de p et r, puis en comptant les points dans le rectangle $ALHK$, en déduire la loi de réciprocité quadratique.

Exercice II.2 (Théorème de Fermat sur un corps fini). Cet exercice utilise les notations et les résultats de l'exercice I.5. Soit $q = p^r$ où p est un nombre premier. On souhaite montrer que si k est un entier tel que $q \geqslant k^4 + 4$, alors l'équation de Fermat sur $\mathbb{F}_q$
$$x^k + y^k = z^k \qquad x,y,z \in \mathbb{F}_q^* \tag{4.1}$$

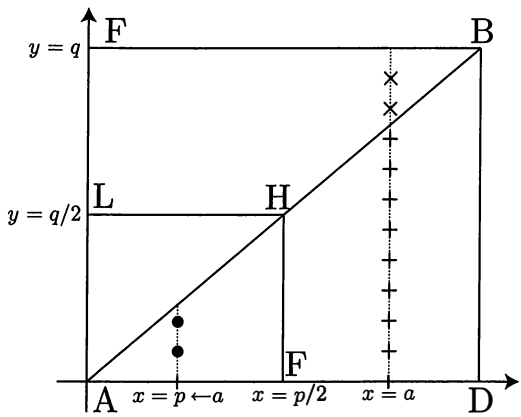

FIG. 2.3 – *Démonstration géométrique de la loi de réciprocité quadratique*

a une solution. Pour ce faire, nous allons utiliser la transformée de Fourier sur le groupe $(\mathbb{F}_q, +)$.

1. Soit $k|q-1$. Montrer qu'il existe un unique sous-groupe H_k d'indice k dans $\mathbb{F}_q^*$ et que
$$H_k = \left\{ x^k \setminus x \in \mathbb{F}_q^* \right\}.$$

2. On note $\chi_0, \ldots, \chi_{k-1}$ les caractères multiplicatifs du groupe quotient $\mathbb{F}_q^*/H_k$. On les étend de manière canonique en des caractères multiplicatifs de $\mathbb{F}_q^*$ en composant par la projection canonique. Montrer alors que pour tout caractère additif ψ on a
$$\widehat{f_{H_k}}(\psi) = \frac{1}{k} \sum_{i=0}^{k-1} G(\chi_i, \psi).$$
En utilisant la proposition 1.17, montrer alors que $\Phi(H_k) < \sqrt{q}$, où Φ est définie à l'équation (5.1), chap. I.

3. Soient $A_1, A_2 \subset G$. On note N et N' respectivement le nombre de solutions des équations
$$x + y = z^k, \qquad \text{avec} \quad x \in A_1, y \in A_2, z \in \mathbb{F}_q^*, \tag{4.2}$$
$$x + y = u, \qquad \text{avec} \quad x \in A_1, y \in A_2, u \in H_k. \tag{4.3}$$
Montrer que $N = kN'$, puis montrer que
$$\left| N - \frac{|A_1||A_2|(q-1)}{q} \right| < k\sqrt{|A_1||A_2|q}. \tag{4.4}$$
On pourra commencer par démontrer une inégalité similaire pour N' en utilisant le résultat de l'exercice I.5, question 2.

4. Si on note $l_i \overset{\text{déf.}}{=} \frac{q-1}{|A_i|}$, alors montrer que si $q \geqslant k^2 l_1 l_2 + 4$ l'équation (4.2) admet une solution. Dans le cas où k ne divise pas $q-1$, montrer que
$$\left\{ x^k \setminus x \in \mathbb{F}_q^* \right\} = \left\{ x^d \setminus x \in \mathbb{F}_q^* \right\},$$
où $d \overset{\text{déf.}}{=} \operatorname{pgcd}(q-1, k)$. En déduire que le résultat est valide pour tout k vérifiant $q \geqslant k^2 l_1 l_2 + 4$.

5. En utilisant les ensembles $A_1 = A_2 = H_k$, montrer que si $q \geqslant k^4 + 4$, alors l'équation (4.1) admet au moins une solution.

Exercice II.3 (Transformée de Walsh et étude statistique). Cet exercice est tiré d'un article de ROCKMORE et MALSEN [52], qui présente une technique connue sous le nom d'analyse de *Yates*, du nom du statisticien qui a inventé cette méthode. On considère la situation suivante : un fermier souhaite connaître l'influence de trois paramètres sur sa production de blé. Ces paramètres sont l'éclairage (représenté par la variable a), la quantité d'herbicide (variable b), et la quantité d'engrais (variable c). Chacune de ces variables peut prendre deux valeurs : forte quantité (notée $+$) et faible quantité (notée $-$). Un compte rendu d'expériences est donné sous la forme du tableau suivant regroupant les valeurs pour la taille moyenne du blé (en centimètres) sous les différentes conditions :

a	b	c	α_{abc}
$+$	$+$	$+$	69
$-$	$+$	$+$	81
$+$	$-$	$+$	63
$-$	$-$	$+$	77
$+$	$+$	$-$	61
$-$	$+$	$-$	92
$+$	$-$	$-$	54
$-$	$-$	$-$	89

On peut donc représenter ces résultats sous la forme d'une fonction

$$f : \begin{cases} (\mathbb{Z}/2\mathbb{Z})^3 \simeq \{-,+\}^3 & \longrightarrow & \mathbb{R} \\ (a,b,c) & \longmapsto & \alpha_{abc} \end{cases}.$$

Dans le but d'analyser ces résultats, on définit l'interaction d'ordre 0, notée μ, qui est simplement la moyenne :

$$\mu \overset{\text{déf.}}{=} \frac{1}{8} \sum_{(a,b,c)\in\{+,-\}^3} \alpha_{abc}.$$

On définit ensuite les interactions d'ordre 1, notées μ_a, μ_b et μ_c, comme correspondant à l'effet d'un seul paramètre, les deux autres étant supposés constants. Par exemple on a

$$\mu_a \overset{\text{déf.}}{=} \frac{1}{4} \sum_{(b,c)\in\{+,-\}^2} \alpha_{+bc} - \frac{1}{4} \sum_{(b,c)\in\{+,-\}^2} \alpha_{-bc}.$$

Dans le même ordre d'idée, définir les 3 interactions d'ordre 2, notées μ_{ab}, μ_{bc} et μ_{ac}, ainsi que l'interaction d'ordre 3, notée μ_{abc}. Comment peut-on calculer toutes ces interactions à l'aide de la transformée de Walsh ? En déduire un algorithme de calcul rapide. Faire le calcul dans le cas du fermier.

Exercice II.4 (Ondelette de Haar). On note ψ_0 la fonction indicatrice de $[0,1]$ et ψ la fonction qui vaut 1 sur $[0,\frac{1}{2}[$, -1 sur $[\frac{1}{2},1]$, et 0 partout ailleurs. On définit ensuite une suite de fonctions ψ_n par

$$\forall j \geqslant 1, \forall k \in \{0,\dots,2^{j-1}-1\}, \quad \psi_n(x) \overset{\text{déf.}}{=} \psi_{j,k}(x) \overset{\text{déf.}}{=} 2^{\frac{j}{2}} \psi\left(2^j x - k\right),$$

où $n = 2^{j-1} + k$. On note, pour $j \geqslant 0$, F_j l'espace des fonctions de $[0,1]$ dans $\mathbb{R}$ constantes sur chacun des intervalles $I_k \overset{\text{déf.}}{=} [k2^{-j}, (k+1)2^{-j}[$, pour $k \in \{0,\dots,2^j-1\}$ (on inclut le point 1 dans le dernier intervalle).

1. Montrer que $\{\psi_n\}_{n=0}^{2^j-1}$ forme une base orthonormée de F_j pour le produit scalaire usuel de $L^2([0,1])$.

2. Soit f une fonction continue de $[0,1]$ dans $\mathbb{R}$. Pour $J \geqslant 0$, on note f_J la projection de f sur F_J :

$$f_J \overset{\text{déf.}}{=} \sum_{j=0}^{J} \sum_{k=0}^{2^{j-1}-1} \langle f, \psi_{j,k} \rangle \psi_{j,k}.$$

Montrer que f_J converge uniformément sur $[0,1]$ vers f quand $J \to +\infty$. On définit ensuite, pour $n \geqslant 0$, la fonction

$$\widetilde{f}_n \overset{\text{déf.}}{=} \sum_{m=0}^{n} \langle f, \psi_m \rangle \psi_m.$$

Montrer que $\widetilde{f}_n$ converge uniformément vers f lorsque $n \to \infty$. Montrer ensuite que $\{\psi_n\}_{n\in\mathbb{N}}$ forme une base de Hilbert de $L^2([0,1])$. La figure 2.4 représente la décomposition d'une fonction f sur les premiers vecteurs de la base des ψ_n, que l'on nomme base de Haar.

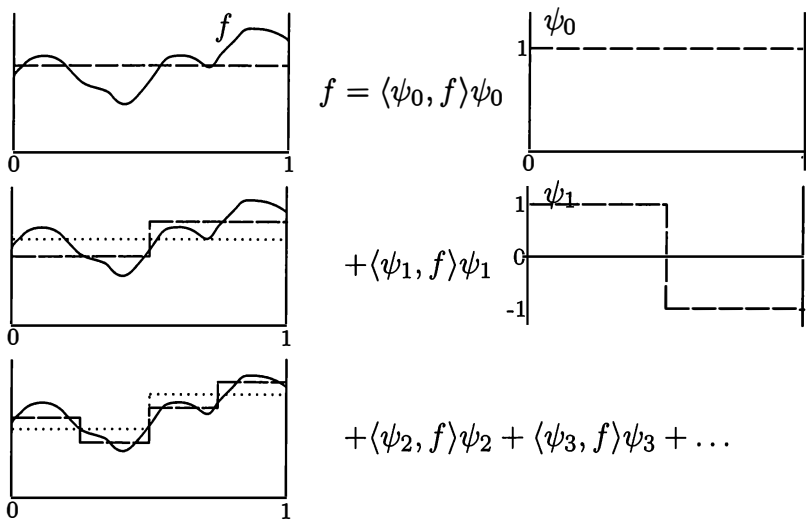

FIG. 2.4 – *Décomposition sur la base de Haar*

3. On introduit, pour $j \geqslant 0$, les fonctions « en escalier » $\varphi_{j,k}(x) \overset{\text{déf.}}{=} 2^{\frac{j}{2}} \psi_0\left(2^j x - k\right)$. Montrer que $\varphi_{j,k}$, pour $k \in \{0,\ldots,2^j-1\}$, forme une base orthonormée de F_j. On définit G_{j-1} l'espace vectoriel tel que $F_j = F_{j-1} \oplus G_{j-1}$ (espace des « détails »). Montrer que $\{\psi_{j,k}\}_{k=0}^{2^{j-1}-1}$ est une base orthonormée de G_{j-1}. Exprimer ensuite la fonction $\psi_{j,k}$ comme combinaison linéaire des $\varphi_{j+1,s}$, $s \in \{0,\ldots,2^j-1$.

4. Soit $f \in F_j$. On note $x^{(0)} \in \mathbb{R}^{2^j}$ le vecteur des produits scalaires $x^{(0)}[k] = \langle f, \varphi_{j,k} \rangle$. Comment les calcule-t-on à partir de f ? Pour $i \in \{1,\ldots,j\}$, on définit des vecteurs $x^{(i)}$ et $d^{(i)}$ de taille 2^{j-i}, par les relations, pour $k \in \{0,\ldots,2^{j-i}-1\}$,

$$x^{(i)}[k] \overset{\text{déf.}}{=} \frac{x^{(i-1)}[2k+1] + x^{(i-1)}[2k]}{\sqrt{2}}$$
$$d^{(i)}[k] \overset{\text{déf.}}{=} \frac{x^{(i-1)}[2k+1] - x^{(i-1)}[2k]}{\sqrt{2}}.$$

De façon intuitive, $x^{(i)}$ représente la tendance dans le signal $x^{(i-1)}$, et $d^{(i)}$ représente les détails. La figure 2.5 symbolise la succession des calculs à effectuer pour obtenir les vecteurs $d^{(i)}$ ainsi que le dernier coefficient $x^{(j)}[0]$. Montrer que l'opérateur $x^{(i)} \mapsto (x^{(i+1)}, d^{(i+1)})$ peut être vu comme une isométrie (plus précisément une rotation) de $\mathbb{R}^{j-i}$. Montrer que les $x^{(i)}$ ont tous la même moyenne, et donc que $x^{(j)}[0]$ représente la moyenne du signal $x^{(0)}$ d'origine.

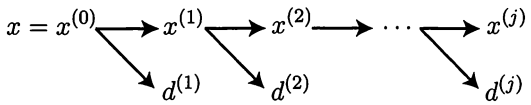

FIG. 2.5 – *Calcul en cascade des coefficients de décomposition*

5. Montrer que l'on a, pour $i \in \{1, \ldots, j\}$ et pour $k \in \{0, \ldots, 2^{j-i} - 1\}$,

$$d^{(i)}[k] = \langle f, \psi_{j-i+1,k} \rangle \qquad \text{et} \qquad x^{(j)}[0] = \langle f, \psi_0 \rangle .$$

   Grâce à cet algorithme, quel est le nombre d'opérations nécessaires pour décomposer une fonction de F_j sur la base des ψ_n ?

6. On suppose que $n = 2^j$. Montrer que l'opérateur Γ qui à $x \in \mathbb{R}^n$ associe le vecteur $(d^{(1)}, x^{(2)}, \ldots, d^{(j)}, x^{(j)})$ est une isométrie de $\mathbb{R}^n$ pour le produit scalaire canonique (on a mis bout à bout les vecteurs $d^{(i)}, x^{(i)}, \ldots$). En déduire que l'application de cet opérateur correspond à la décomposition de x dans une base orthonormée de $\mathbb{R}^n$ que l'on précisera. Comparer cette base à la base de Walsh décrite au paragraphe 2.1. Comparer en termes de complexité l'algorithme de décomposition dans la base de Haar et celui de décomposition dans la base de Walsh (algorithme FWT, paragraphe 2.2).

La figure 2.6 montre deux exemples de transformées $y = \Gamma x$. On peut voir que comme les signaux sont réguliers, seuls les coefficients correspondant aux échelles grossières (c'est-à-dire pour j petit, les indices de droite de la transformée) sont grands. La base de Haar est l'exemple le plus simple de base d'ondelettes. L'algorithme de transformation en ondelettes rapide a été introduit par MALLAT, [51]. Les différences entre les bases de Walsh et de Haar illustrent le passage de la transformée de Fourier (ici sur $(\mathbb{F}_2)^k$) à la transformée en ondelettes. L'exercice VII.11 présente la construction d'ondelettes sur les corps finis.

Exercice II.5 (Compression d'images). Le but de cet exercice est d'ordonner de façon appropriée les fonctions de Walsh pour réussir à compresser des signaux 1D et 2D.

1. Généraliser la transformée de Walsh discrète au cas bidimensionnel. On utilisera les fonctions

$$\chi_{i,j}(s,t) = \chi_i(s)\chi_j(t).$$

   Ecrire un algorithme de calcul rapide de la transformée de Walsh 2D.

2. Montrer que l'on peut classer les fonctions de Walsh discrètes (définies à l'équation (2.1)) par ordre croissant du nombre de changements de signe. La figure 2.7 montre les matrices de Walsh obtenues en classant les fonctions dans l'ordre usuel et dans l'ordre du nombre de changements de signe.

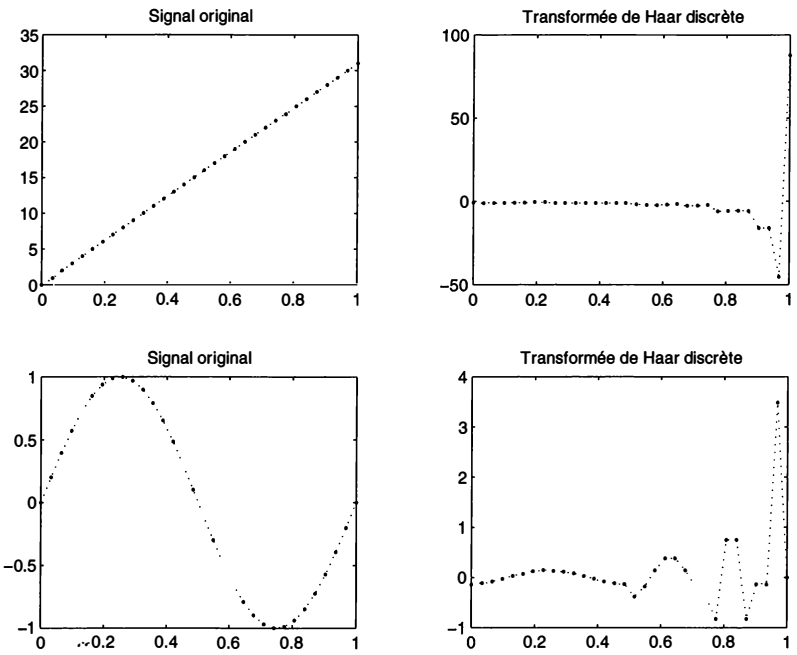

FIG. 2.6 – *Exemples de transformées de Haar discrètes*

3. Intuitivement, un tel classement permet d'ordonner le spectre de Walsh depuis les tendances (basses fréquences) jusqu'aux détails (hautes fréquences). Calculer pour quelques fonctions les spectres obtenus avec les deux classements, et vérifier cette interprétation. La figure 2.8 montre les spectres de la fonction représentée en haut à gauche de la figure 2.9.
4. Nous avons donc classé les fonctions de Walsh selon un ordre $\chi_{i_0}, \ldots, \chi_{i_N}$. On considère un signal $f \in \mathbb{C}^N$. Pour $0 \leqslant n < N$, on construit la fonction

$$f_n \stackrel{\text{déf.}}{=} \sum_{k=0}^{n} \langle f, \chi_{i_k} \rangle \chi_{i_k}.$$

Expliquer pourquoi ce procédé permet de compresser le signal f. La figure 2.9 montre la compression progressive d'un signal. On indique à chaque fois le pourcentage de coefficients de Walsh qui ont été conservés. Après avoir étudié la transformée de Fourier discrète au chapitre III, on pourra effectuer le même procédé, mais avec le spectre de Fourier. Quels sont les avantages et les désavantages de chaque méthode (temps de calcul, qualité de la reconstruction, etc.)?

5. Quel(s) classement(s) peut-on adopter pour les fonctions de Walsh 2D? La figure 2.10 propose un tel classement (de gauche à droite et de haut en bas). Appliquer ce classement pour compresser des images 2D. Ecrire un programme MATLAB permettant d'effectuer cette compression. La figure 2.11 montre la compression progressive d'une image représentant la lettre A.

Exercice II.6 (Matrices de Hadamard). Cet exercice fait la liaison entre les matrices de Walsh considérées au paragraphe 2.1 et les résidus quadratiques introduits au début de ce chapitre. Une matrice H_n, de taille $n \times n$, dont les entrées sont $+1$ ou -1, est dite matrice de Hadamard si elle vérifie

$$H_n H_n{}^{\mathrm{T}} = n\mathrm{Id}_n,$$

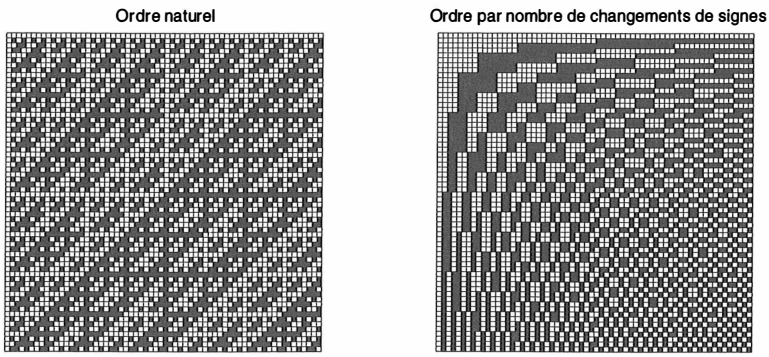

FIG. 2.7 – *Deux façons de classer les fonctions de Walsh*

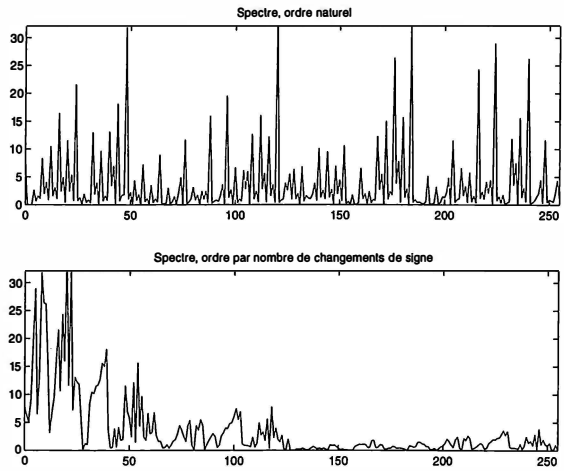

FIG. 2.8 – *Spectre de Walsh d'un signal 1D*

où on a noté ${H_n}^{\mathrm{T}}$ la matrice transposée de H_n.

1. Expliquer pourquoi la matrice W_n, définie à la proposition 2.4, pour $n = 2^k$, est une matrice de Hadamard.
2. Montrer que s'il existe une matrice de Hadamard H_n de taille $n \times n$, alors, n vaut 1, ou 2, ou est un multiple de 4. On pourra commencer par montrer que l'on peut supposer que H_n est normalisée, c'est-à-dire avec des 1 sur la première ligne et la première colonne. Ensuite on montrera que si $n \geqslant 3$, on peut supposer que les trois premières lignes de H_n s'écrivent sous la forme

$$\begin{array}{cccccccccccc}
1 & \dots & 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 & \dots & 1 \\
1 & \dots & 1 & 1 & \dots & 1 & -1 & \dots & -1 & -1 & \dots & -1 \\
1 & \dots & 1 & -1 & \dots & -1 & 1 & \dots & 1 & -1 & \dots & -1 \\
\underbrace{}_{i} & & & \underbrace{}_{j} & & & \underbrace{}_{k} & & & \underbrace{}_{l} & &
\end{array}$$

   où les entiers i, j, k, et l désignent les longueurs de chaque portion (ils peuvent éventuellement être nuls). Enfin, on montrera que l'on a en fait $i = j = k = l$.
3. Le problème inverse, à savoir la construction d'une matrice H_n pour un n multiple de 4 donné, est très complexe. En fait, on conjecture qu'il est toujours possible de le

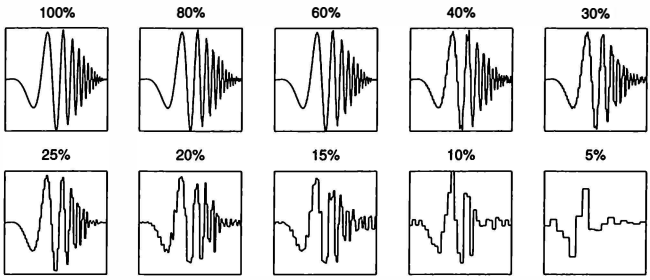

FIG. 2.9 – *Compression d'un signal 1D*

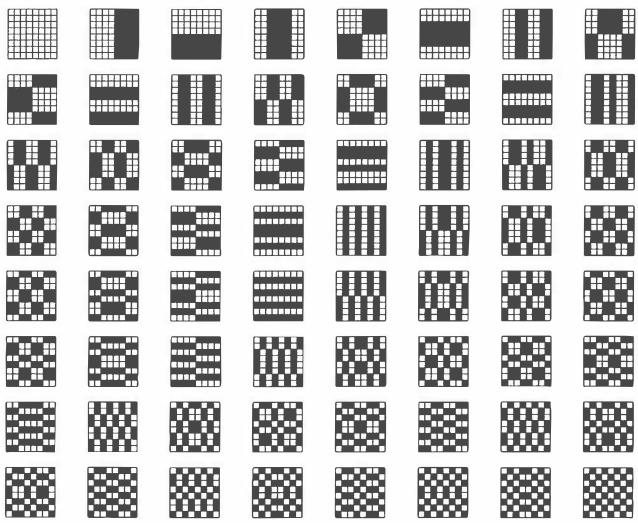

FIG. 2.10 – *Classement des fonctions de Walsh 2D*

faire, bien qu'on ne l'ait pas encore prouvé. On suppose que $n = p+1$, où p est un nombre premier impair. On suppose aussi que n est un multiple de 4, et nous allons montrer que l'on peut alors construire une matrice H_n. Nous utiliserons le caractère de résidu quadratique modulo p, noté η, qui est défini à l'équation (1.2). On définit une matrice Q de taille $p \times p$ par

$$Q \overset{\text{déf.}}{=} \{\eta(j-i)\}_{0 \leqslant i,j \leqslant p-1}.$$

Montrer que Q est anti-symétrique, et que l'on a $QQ^{\mathrm{T}} = p\mathrm{Id}_p - J$, où J est la matrice dont toutes les entrées valent 1. Montrer aussi que l'on a $QJ = JQ = 0$. Une telle matrice est appelée *matrice de Paley.*

4. On définit maintenant la matrice H_n de taille $n \times n$ par

$$H_n \overset{\text{déf.}}{=} \begin{pmatrix} 1 & v \\ v^{\mathrm{T}} & Q - \mathrm{Id}_p \end{pmatrix},$$

où on a noté $v = (1, \ldots, 1) \in \mathbb{R}^p$. Montrer que H_n est une matrice de Hadamard.

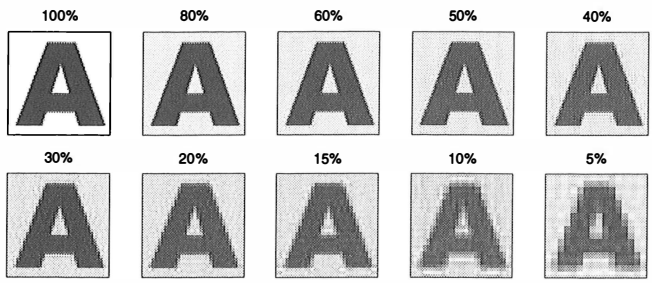

FIG. 2.11 – *Compression d'une image 2D*

Voici un exemple pour $p = 7$:

$$H_8 \overset{\text{déf.}}{=} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 \end{pmatrix}.$$

5. Soit A une matrice de taille $n \times n$ telle que ses entrées a_{ij} vérifient

$$\forall (i,j) \in \{1,\dots,n\}^2, \quad |a_{ij}| \leqslant 1.$$

Montrer l'inégalité de Hadamard :

$$|\det(A)| \leqslant n^{\frac{n}{2}}. \tag{4.5}$$

Montrer que s'il existe une matrice de Hadamard, alors cette dernière atteint cette borne.

L'interprétation géométrique de la borne (4.5) est très simple. Il s'agit de considérer un système de n vecteurs (les colonnes de la matrice) à l'intérieur du cube $|x_i| \leqslant 1$, (où on note $\{x_i\}_{i=1}^n$ un système de coordonnées), enfermant un parallélépipède rectangle de volume maximal. Dans le cas des matrices de Hadamard, ces vecteurs sont des grandes diagonales du cube, et sont donc de longueurs maximales. De plus, elles sont orthogonales, de façon à produire le volume maximal. Dans les dimensions où les matrices de Hadamard n'existent pas, il n'est pas possible de produire des diagonales orthogonales, même si l'on pense que les vecteurs qui minimisent (4.5) sont proches des grandes diagonales. Ceci reste un problème ouvert.
L'exercice VI.11 présente une application des matrices de Hadamard pour la construction de codes correcteurs bi-orthogonaux.

Exercice II.7 (Produit tensoriel matriciel). Soit A une matrice carrée de taille s et B une matrice carrée de taille t. On définit le produit tensoriel $A \otimes B$ comme la matrice de taille $s \times t$

$$A \otimes B \overset{\text{déf.}}{=} \begin{pmatrix} a_{11}B & \cdots & a_{1s}B \\ \vdots & & \vdots \\ a_{s1}B & \cdots & a_{ss}B \end{pmatrix}.$$

1. On suppose que A vérifie $AA^* = s\mathrm{Id}_s$. Montrer que $A^{\otimes n} = A \otimes \cdots \otimes A$ (n produits) vérifie $A^{\otimes n}(A^{\otimes n})^* = s^n \mathrm{Id}_{s^n}$.
2. Quel rapprochement faire avec la transformée de Walsh ?
3. En vous inspirant de l'algorithme rapide FWT, écrire un algorithme rapide qui calcule la transformée $y = A^{\otimes n}x$. Comment se calcule la transformée inverse ?
4. On prend comme matrice de base

$$A_\alpha \overset{\text{déf.}}{=} \sqrt{2}\begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ \sin(\alpha) & -\cos(\alpha) \end{pmatrix}.$$

En quoi la transformée $x \mapsto A_\alpha^{\otimes n} x$ peut être vue comme une transformée de Walsh intermédiaire ?

La figure 2.12 montre les transformées d'une fonction « triangle » pour des valeurs de α dans $[0,\pi/2]$. La transformée de Walsh ordinaire correspond à la $5^{\text{ième}}$ courbe. Pour $\alpha = \pi/2$, on trouve le signal d'origine symétrisé. On pourra regarder l'exercice VIII.7,

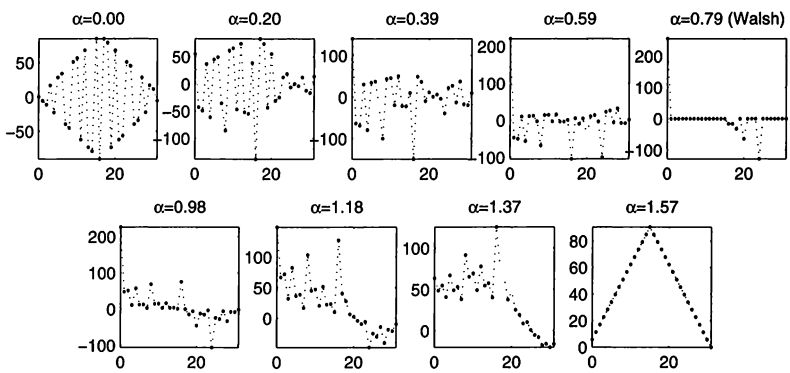

FIG. 2.12 – *Transformée de Walsh intermédiaire*

qui utilise la théorie des représentations linéaires pour construire une matrice A de taille 8.

Exercice II.8 (Généralisation de l'identité de MacWilliams). Dans cet exercice, on propose d'étendre l'identité de MacWilliams au cas de l'espace vectoriel $E = \mathbb{F}_q^k$.

1. On définit la forme bilinéaire suivante sur $E \times E$, à valeur dans $\mathbb{F}_q$:

$$\forall (a,x) \in E^2, \quad \langle a,x \rangle \overset{\text{déf.}}{=} \sum_{i=0}^{p-1} a_i x_i.$$

Expliquer en quoi elle représente la forme bilinéaire de la dualité (crochet de la dualité) entre l'espace E et son dual E^* (correctement identifié à E).

2. On note χ_1 le caractère additif canonique de $\mathbb{F}_q$, comme défini par l'équation (1.6). Soit $a = \{a_0, \ldots, a_{k-1}\} \in (\mathbb{Z}/q\mathbb{Z})^k$. On définit

$$\chi_a : \begin{cases} E & \longrightarrow & \mathbb{C}^* \\ x & \longmapsto & \chi_1(\langle a,x \rangle) \end{cases}.$$

Expliquer pourquoi les applications χ_a permettent de définir un isomorphisme entre E et son dual en tant que groupe additif, $\widehat{E}$.

3. Soit H un sous-groupe de E. Montrer que c'est aussi un sous-espace vectoriel. Déduire de la question précédente que l'on peut identifier l'orthogonal de H pour la structure de groupe, noté $H^{\sharp}$, et celui pour la structure d'espace vectoriel, noté $H^{\perp}$.
4. Démontrer l'identité de MacWilliams dans le cadre de l'espace E :

$$A_{H^\perp}(X,Y) = \frac{1}{|H|} A_H(X+(q-1)Y, X-Y).$$

Exercice II.9 (Formule de Poisson et distributions). Cet exercice demande certaines connaissances en théorie des distributions, notamment la définition de la transformée de Fourier d'une distribution. Ici on prend comme convention la transformée définie à l'équation (1.1), chap. IV, qui diffère d'un facteur 2π de celle employée en (3.9). On note Π_s le peigne de Dirac de pas s, c'est-à-dire

$$\Pi_s \overset{\text{déf.}}{=} \sum_{k\in\mathbb{Z}} \delta_{ks}, \tag{4.6}$$

où δ_t est la distribution définie par $\langle \delta_t, \varphi\rangle \overset{\text{déf.}}{=} \varphi(t)$ pour $\varphi \in \mathscr{C}_0^\infty(\mathbb{R})$ (fonctions de classe $\mathscr{C}^\infty$ à support compact). Montrer que la formule de Poisson (3.12) implique l'égalité suivante :

$$\widehat{\Pi_s} = \frac{2\pi}{s} \Pi_{\frac{2\pi}{s}}.$$

Exercice II.10 (Echantillonnage de Shannon). Soit $T > 0$. On note

$$I_T \overset{\text{déf.}}{=} \left[-\frac{\pi}{T}, \frac{\pi}{T}\right] \quad \text{et} \quad E_T \overset{\text{déf.}}{=} \left\{ f \in L^2(\mathbb{R}) \setminus \operatorname{Supp}(\widehat{f}) \subset I_T \right\}.$$

Soit $f \in E_T$. On veut démontrer le théorème d'échantillonnage de Shannon, qui dit que f peut être reconstruite (interpolée) à partir des échantillons $f(nT)$, pour $n \in \mathbb{Z}$. D'une façon plus précise, si on note

$$\operatorname{sinc}_T(t) \overset{\text{déf.}}{=} \frac{\sin(\pi t/T)}{\pi t/T}, \tag{4.7}$$

alors on veut montrer que

$$f(t) \overset{\text{déf.}}{=} \sum_{n\in\mathbb{Z}} f(nT) \operatorname{sinc}_T(t-nT).$$

1. Montrer que f est de classe $\mathscr{C}^\infty$.
2. On note f_d la distribution qui correspond à l'échantillonnage de f :

$$f_d \overset{\text{déf.}}{=} \sum_{n\in\mathbb{Z}} f(nT)\delta_{nT}.$$

   En utilisant l'égalité (4.6), montrer que l'on a

$$|\omega| \leqslant \frac{\pi}{T} \Longrightarrow \widehat{f_d}(\omega) = \frac{1}{T}\widehat{f}(\omega).$$

3. Calculer la transformée de Fourier inverse de la fonction indicatrice de l'intervalle I_T. En déduire le théorème d'échantillonnage.
4. Montrer que la famille $\{t \mapsto \operatorname{sinc}_T(t-nT)\}_{n\in\mathbb{Z}}$ forme une base orthogonale (base de Hilbert) de l'espace E_T. Comment se calcule la projection d'une fonction $f \in L^2(\mathbb{R})$ sur cet espace ?

Chapitre III
Transformée de Fourier discrète

> Le développement de l'informatique au cours des années 1960 a donné beaucoup d'importance aux programmes de calcul rapide. [...] Des programmes de calcul rapide de transformée de Fourier (FFT) furent créés à cette époque et leur usage se répandit immédiatement à une vaste échelle, en même temps que la réputation de leurs initiateurs, Cooley et Tukey.
>
> JEAN-PIERRE KAHANE [38] (1998)

L'utilisation de la transformée de Fourier discrète est à la base de la quasi-totalité des algorithmes numériques digitaux. La découverte de l'algorithme de transformation rapide *FFT* (pour ***F**ast **F**ourier **T**ransform*, en français *Transformée de Fourier Rapide*) a révolutionné l'univers du traitement du signal en permettant des calculs numériques en des temps raisonnables. C'est en grande partie cette découverte qui a fait comprendre que l'on pouvait travailler de façon aussi rapide dans le monde digital (constitué de signaux discrets) que dans le monde analogique (constitué de signaux continus). De plus amples détails sur l'histoire de cette découverte, et de ces conséquences, se trouvent dans l'article de ROCKMORE [60]. Plus qu'un simple cas particulier de la transformée de Fourier sur un groupe fini, la transformée de Fourier discrète possède son propre langage et surtout des algorithmes efficaces nettement moins évidents que les formules limpides du chapitre précédent. Ce chapitre vise en quelque sorte à faire un tour du propriétaire ; il montre en tout cas que la multitude d'algorithmes FFT existants est impressionnante. Mais le plus important, au-delà d'une compréhension totale des différentes déclinaisons de l'algorithme, est de percevoir la stratégie de l'algorithme, pour pouvoir décider, le cas échéant, quelle implémentation utiliser.

L'algorithme FFT en version décimation temporelle est relativement bien décrit (et surtout bien implémenté) dans les NUMERICAL RECIPES [31]. En ce qui concerne la version décimation fréquentielle ainsi que de nombreuses améliorations, on se réfèrera au livre de BRIGHAM [11]. Pour des détails d'implémentation en langage C, on pourra regarder le livre de ARNT [2].

1 Le langage du traitement du signal

Dans ce paragraphe, nous allons traduire les propriétés algébriques de la transformée de Fourier dans le langage de la théorie des signaux discrets. Dans un premier temps,

nous nous restreindrons à une étude unidimensionnelle (pour présenter les algorithmes et quelques applications), puis nous ferons le lien entre la transformée de Fourier sur un groupe abélien produit et la transformée de Fourier discrète en dimension deux et plus.

Pour fixer les idées, nous allons considérer des signaux temporels à valeurs complexes. Ces derniers correspondent à des fonctions $\widetilde{f} : t \in \mathbb{R} \to \widetilde{f}(t) \in \mathbb{C}$. Pour traiter de façon numérique ce signal, nous n'allons considérer qu'un nombre fini de valeurs du signal, et travailler sur ces valeurs. On nommera donc échantillon de taille N du signal original $\widetilde{f}$ le vecteur $f \overset{\text{déf.}}{=} \{f[n]\}_{n=0}^{N-1}$, où l'on a noté $f[n] \overset{\text{déf.}}{=} \widetilde{f}(t_n)$ la valeur du signal f à l'instant t_n. La notation entre accolades est censée rappeler que l'on considère nos vecteurs comme des échantillons d'un signal (continu), mais il arrivera que l'on considère ces éléments comme de simples vecteurs de $\mathbb{C}^N$. Pour que l'analyse qui suit ne soit pas biaisée (particulièrement lors du rapprochement avec la transformée continue à la section 1, chap. IV), les valeurs des $\{t_n\}_{n=0}^{N-1}$ sont supposées espacées régulièrement dans un intervalle $[a,b]$, c'est-à-dire $t_n = a + \frac{b-a}{N} n$.

Définition 1.1 (Transformée de Fourier discrète). On définit la *Transformée de Fourier discrète* (en abrégé *TFD*) de l'échantillon $f = \{f[n]\}_{n=0}^{N-1}$ comme étant le vecteur $\widehat{f} = \{\widehat{f}[k]\}_{k=0}^{N-1} \in \mathbb{C}^N$ avec

$$\widehat{f}[k] \overset{\text{déf.}}{=} \sum_{n=0}^{N-1} f[n] \omega_N^{-nk} \qquad \text{pour } k = 0, \ldots, N-1, \tag{1.1}$$

où l'on a noté $\omega_N = e^{\frac{2i\pi}{N}}$ une racine $N^{\text{ième}}$ primitive de l'unité.
On notera aussi $\mathscr{F}(f) \overset{\text{déf.}}{=} \widehat{f}$, ce qui permet de définir

$$\mathscr{F} : \left\{ \begin{array}{ccc} \mathbb{C}^N & \longrightarrow & \mathbb{C}^N \\ f & \longmapsto & \mathscr{F}(f) = \widehat{f} \end{array} \right. .$$

Cette notation peut prêter à confusion avec la transformée de Fourier sur un groupe fini définie par l'équation (4.1), chap. I, cependant, la grande similitude entre les deux applications (tout ceci est justifié un peu plus bas) fait qu'il est commode d'employer la même notation.

***Remarque* 1.2.** On aurait pu choisir une autre racine primitive de l'unité à la place de ω_N. Cela revient à choisir un autre générateur pour le groupe de départ $\mathbb{Z}/N\mathbb{Z}$, et donc à numéroter dans un ordre différent les éléments de f.

***Remarque* 1.3. (Lien avec la transformée de Fourier sur un groupe fini).** Nous avons déjà vu à la section 1.2, chap. I, que les caractères $(\chi_k)_{k=0}^{N-1}$ sur le groupe cyclique $\mathbb{Z}/N\mathbb{Z}$ peuvent être définis par

$$\forall s \in \mathbb{Z}/N\mathbb{Z}, \quad \chi_k(s) \overset{\text{déf.}}{=} \omega_N^{-ks}. \tag{1.2}$$

On remarque que notre échantillon $f \in \mathbb{C}^N$ permet de définir une fonction $f_1 : \mathbb{Z}/N\mathbb{Z} \to \mathbb{C}$, et réciproquement. On peut faire le lien entre transformée de Fourier discrète et caractères :

$$\widehat{f}[k] = \widehat{f_1}(\chi_k).$$

On peut donc réécrire la formule d'inversion de Fourier de la proposition 4.4, chap. I, en termes de transformée de Fourier discrète.

Proposition 1.4 (Transformée de Fourier inverse). *On a la formule d'inversion suivante :*

$$\forall n = 0, \ldots, N-1, \quad f[n] = \frac{1}{N} \sum_{k=0}^{N-1} \widehat{f}[k] \omega_N^{nk}. \tag{1.3}$$

Corollaire 1.5. $\mathscr{F} : \mathbb{C}^N \to \mathbb{C}^N$ *est un isomorphisme d'espaces vectoriels.*

Rappelons aussi la formule de Plancherel.

Proposition 1.6 (Formule de Plancherel). *Soient f et g deux échantillons de taille N. On a la formule suivante :*

$$\sum_{i=0}^{N} f[i]\overline{g[i]} = \frac{1}{N} \sum_{i=0}^{N} \widehat{f}[i]\overline{\widehat{g}[i]}.$$

2 Transformée de Fourier rapide

> The discrete Fourier transform can, in fact, be computed in $O(N \log_2 N)$ operations with an algorithm called the fast Fourier transform, or FFT. The difference between $N \log_2 N$ and N^2 is immense. With $N = 10^6$, for example, it is the difference between, roughly, 30 seconds of CPU time and 2 weeks of CPU time on a microsecond cycle time computer.
>
> W.H. PRESS et Al. [31] (1988)

Ce paragraphe se veut directement tourné vers les applications informatiques de la TFD. Il ne nécessite pas de connaissance en théorie des groupes. Les connexions entre l'algorithme FFT et l'algèbre sont discutées dans certains exercices, par exemple lors de l'étude de la méthode de *Good-Thomas* III.2. En parallèle à la lecture de ce chapitre, il faut bien sûr avoir un œil sur les algorithmes référencés au paragraphe 3, annexe A, pour faire le lien entre implémentation concrète et formules mathématiques.

2.1 Présentation de l'algorithme

Pour un signal f dont on connaît un échantillon $\{f[n]\}_{n=0}^{N-1}$, le calcul direct des N coefficients de la transformée de Fourier discrète

$$\widehat{f}[k] \overset{\text{déf.}}{=} \sum_{n=0}^{N-1} f[n] e^{-kn\frac{2i\pi}{N}} \qquad \text{pour } k = 0, \ldots, N-1 \tag{2.1}$$

nécessite $2N^2$ opérations (additions et multiplications complexes). L'algorithme *FFT* permet, en réordonnant les calculs de manière dichotomique, de réduire considérablement le temps de calcul en le ramenant à un ordre de $O(N \log(N))$. Dans tout ce chapitre, nous allons présenter différentes versions de l'algorithme FFT, en commençant par la version originale, et sans doute la plus simple, l'algorithme de COOLEY et TUKEY. Cependant, nous verrons que cet algorithme s'est décliné en un nombre quasi infini de versions plus savantes les unes que les autres, pour s'adapter à différentes conditions (longueur des

vecteurs principalement), et obtenir le résultat toujours plus rapidement. Derrière une transformation en apparence très simple, la TFD, se cache donc une multitude d'idées de nature combinatoire et algébrique.

Avant de nous lancer dans une description périlleuse de l'algorithme, notons un fait rassurant : nous allons pouvoir réinvestir facilement notre algorithme pour calculer la transformée inverse, comme le précise la remarque suivante.

***Remarque* 2.1. (Transformée inverse).** On remarque que la formule de transformée inverse (1.3) peut s'obtenir en remplaçant ω_N par ω_N^{-1} dans l'algorithme de calcul, puis en divisant le résultat par N. En conséquence, on peut aussi calculer la transformée de Fourier discrète inverse en temps $O(N\log(N))$, en modifiant de façon évidente l'algorithme utilisé. De façon plus synthétique, en considérant l'échantillon $\{f_1[n]\}_{n=0}^{N-1}$ défini par

$$\forall n \in \{1,\dots,N-1\}, \quad f_1[n] = \frac{1}{N} f[N-n] \qquad f_1[0] \overset{\text{déf.}}{=} \frac{1}{N} f[0],$$

on dispose d'une écriture de la transformée de Fourier inverse de f en terme d'une transformée de Fourier directe :

$$\mathscr{F}^{-1}(f) = \mathscr{F}(f_1).$$

L'algorithme que nous nous apprêtons à décrire a été découvert par COOLEY et TUKEY en 1965. Il permet, lorsque l'on dispose d'une « bonne » décomposition de l'entier N, de calculer la transformée de Fourier discrète de façon très rapide. Nous verrons dans la suite de l'exposé d'autres algorithmes qui permettent d'exploiter certaines décompositions moins optimales de N. Cependant, dans cette première approche de l'algorithme FFT, nous allons supposer que $N = 2^p$. Cette factorisation très simple de N va permettre d'employer la célèbre « philosophie » *diviser pour régner*, en effectuant une progression dichotomique dans le calcul de la TFD. Pour mettre en œuvre cette dichotomie, regroupons les termes de la somme d'une TFD suivant la parité des indices.
On obtient alors, pour $k \in \{0,\dots,N-1\}$,

$$\widehat{f}[k] = \sum_{n=0}^{N/2-1} f[2n] e^{-2i\pi k(2n)/N} + \sum_{n=0}^{N/2-1} f[2n+1] e^{-2i\pi k(2n+1)/N} \tag{2.2}$$

$$= \sum_{n=0}^{N/2-1} f[2n] e^{-2i\pi kn/(N/2)} + \omega_N^{-k} \sum_{n=0}^{N/2-1} f[2n+1] e^{-2i\pi kn/(N/2)}, \tag{2.3}$$

où l'on a noté $\omega_N = e^{2i\pi/N}$. Donc si on note

$$f^0 \overset{\text{déf.}}{=} \{f[0], f[2], \dots, f[N-2]\} \tag{2.4}$$

$$f^1 \overset{\text{déf.}}{=} \{f[1], f[3], \dots, f[N-1]\} \tag{2.5}$$

les vecteurs d'indices pairs (resp. impairs) formés à partir de f, on remarque que pour les $N/2$ premiers indices $k \in \{0,1,\dots,N/2-1\}$, l'équation (2.3) s'écrit comme la somme de deux transformées de Fourier discrètes :

$$\widehat{f}[k] = \widehat{f^0}[k] + \omega_N^{-k} \widehat{f^1}[k]. \tag{2.6}$$

Pour les indices $k \in \{N/2,\dots,N-1\}$, si on note $k' = k - N/2$, en utilisant le fait que les vecteurs $\widehat{f^0}$ et $\widehat{f^1}$ représentent des échantillons de période $N/2$, et que $\omega_N^k = -\omega_N^{k'}$ on obtient cette fois la différence de deux transformées de Fourier :

$$\widehat{f}[k] = \widehat{f^0}[k'] - \omega_N^{-k'} \widehat{f^1}[k']. \tag{2.7}$$

Définition 2.2 (Quelques notations). Pour résumer tout ceci sous une forme plus algorithmique, notons

$$\widehat{f}_g \overset{\text{déf.}}{=} \left\{\widehat{f}[0], \widehat{f}[1], \ldots, \widehat{f}[N/2-1]\right\} \tag{2.8}$$

$$\widehat{f}_d \overset{\text{déf.}}{=} \left\{\widehat{f}[N/2], \widehat{f}[N/2+1], \ldots, \widehat{f}[N-1]\right\}. \tag{2.9}$$

Ce sont les parties droite et gauche du vecteur transformé $\widehat{f} = \mathscr{F}(f)$. Nous allons aussi définir l'opérateur $\mathscr{S}_N^x$, pour $x \in \mathbb{R}$, qui prend un vecteur $a = \{a_0, \ldots, a_{N-1}\} \in \mathbb{C}^N$ de longueur N et renvoie

$$\mathscr{S}_N^x a \overset{\text{déf.}}{=} \left\{a_j e^{-xj\frac{2i\pi}{N}}\right\}_{j=0}^{N-1} = \left\{a_j \omega_N^{-xj}\right\}_{j=0}^{N-1} \in \mathbb{C}^N. \tag{2.10}$$

On a alors l'expression très simple de la récurrence que nous allons utiliser pour implémenter l'algorithme FFT :

$$\widehat{f}_g = \widehat{f^0} + \mathscr{S}_{N/2}^{1/2} \widehat{f^1} \tag{2.11}$$

$$\widehat{f}_d = \widehat{f^0} - \mathscr{S}_{N/2}^{1/2} \widehat{f^1}. \tag{2.12}$$

Les équations (2.11) et (2.12), aussi appelées équations de *Danielson-Lanczos*, expriment le fait que la transformée de Fourier discrète d'un signal de longueur N peut se calculer en fonction de deux signaux de longueur $N/2$, ici notés f^0 et f^1. On appelle cette approche *décimation temporelle* (en anglais *Decimation In Time*, ou *DIT*), par opposition à une autre approche, la *décimation fréquentielle*, qui sera décrite rapidement au paragraphe 2.6. C'est la décimation temporelle qui va être développée (et optimisée) au paragraphe suivant, mais avant toute chose, commençons par présenter une implémentation naïve.

***Remarque* 2.3. (L'effet papillon).** L'opération consistant à mélanger deux entrées des parties paire et impaire d'un vecteur en suivant les équations (2.11) et (2.12) est appelée *schéma papillon* (en anglais *butterfly scheme*). La figure 3.1 montre de façon schématique les opérations effectuées. Elle donne aussi une idée du câblage à réaliser pour effectuer une telle opération directement sur une carte dédiée au traitement de signaux. En effet, comme le montre la figure 3.2, une itération dans l'algorithme FFT (ici pour une entrée de taille 8) n'est qu'une succession de schémas papillons effectués en cascade.

La façon la plus simple de mettre en œuvre les équations (2.11) et (2.12) est d'utiliser une procédure récursive. C'est un fait connu qu'une procédure récursive puisse être écrite, au moyen de boucles, de façon non récursive (mais ce procédé peut être parfois périlleux). Nous verrons au paragraphe suivant 2.5 que l'algorithme FFT a beaucoup à gagner à être écrit de façon non récursive, et ce, pas seulement à cause d'un gain de temps. Mais dans un but pédagogique, et afin de présenter quelques optimisations qui peuvent être faites sur l'implémentation de la FFT, nous allons nous attarder sur l'implémentation récursive écrite à la section 3, annexe A.

La procédure `fft_rec` prend donc en entrée un entier `dir` qui vaut $+1$ ou -1 selon que la transformée de Fourier est directe ou inverse. Pour simplifier la compréhension du code, une procédure `operateur_s` a été écrite pour réaliser l'opérateur $\mathscr{S}_N^x$: elle prend en entrée un vecteur ainsi qu'un nombre réel x (qui dépend du signe de la transformée).

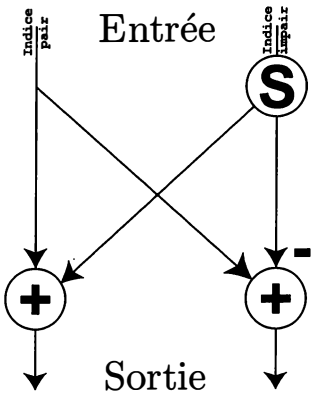

FIG. 3.1 – *Schéma papillon élémentaire*

2.2 Analyse du coût

En utilisant les équations de récurrence (2.11) et (2.12), on calcule facilement le coût de l'algorithme.

Proposition 2.4 (Complexité de la FFT). *Si on note $C(N)$ le coût de l'algorithme FFT pour une entrée de longueur N, alors $C(N)$ vérifie l'équation fonctionnelle*

$$C(N) = 2C(N/2) + KN,$$

où K est une certaine constante. Au final, on arrive à l'expression $C(N) = KN\log_2(N)$.

Démonstration. Le calcul de $\widehat{f}$ nécessite le calcul de $\widehat{f^1}$ et $\widehat{f^2}$ (soit $2C(N/2)$ opérations), puis le mélange des deux transformées par le schéma papillon (soit KN opérations). Pour se ramener à une récurrence linéaire, il suffit de poser $P = \log_2(N)$, et $C'(N) = \frac{C(N)}{N}$ vérifie l'équation fonctionnelle $C'(P) = C'(P-1) + K$. Comme $C'(0) = 0$, on en déduit $C'(P) = KP$, ce qui permet de conclure. □

L'algorithme FFT peut sembler un peu magique, toujours est-il que sa découverte a rendu possibles de nombreux calculs en réduisant le coût du calcul de N coefficients de Fourier de $O(N^2)$ pour une approche naïve à $O(N\log(N))$. Sa découverte assez récente (au milieu des années 1960) a été une mini révolution : un calcul qui nécessitait jusqu'alors deux semaines sur un ordinateur de l'époque était tout à coup réalisable en à peine trente secondes [1].

2.3 Variations autour de l'algorithme

Avant de décrire une implémentation plus efficace de l'algorithme FFT, faisons quelques remarques complémentaires, qui fournissent de nombreuses variations autour de l'implémentation récursive proposée.

***Remarque* 2.5. (Longueur des entrées).** Dans le cas où la longueur N des données d'un échantillon $\{f[n]\}_{n=0}^{N-1}$ ne serait pas une puissance de deux, on peut faire un calcul

1. Source : [31], pour N de l'ordre de 10^6

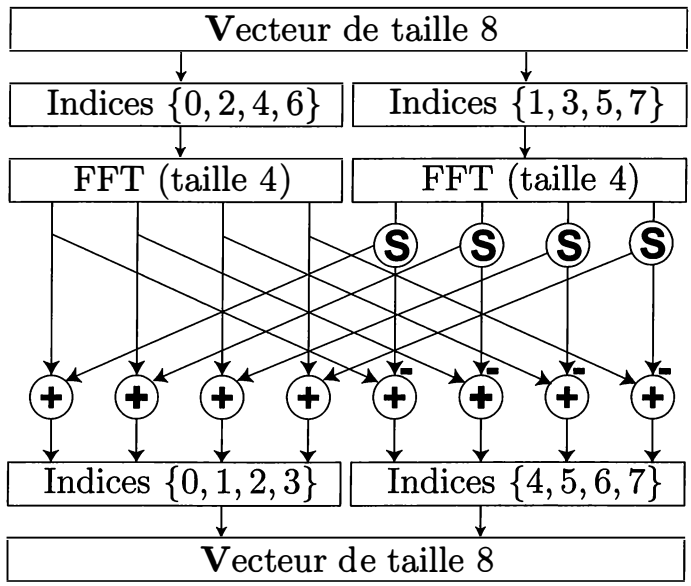

FIG. 3.2 – *Une itération de l'algorithme FFT*

approché en complétant l'échantillon par une suite de zéros, pour obtenir un échantillon $\{f_1[n]\}_{n=0}^{M-1}$, avec $M = 2^p$. Bien sûr, on ne calcule plus exactement la même transformée, mais dans le cas d'un calcul approché (calcul de transformées continues, comme c'est expliqué à la section 1, chap. IV), cela revient à calculer la transformée à des fréquences légèrement différentes, ce qui est souvent acceptable.

***Remarque* 2.6. (Base de calcul).** Les équations (2.11) et (2.12) qui nous ont servi pour implémenter l'algorithme sont la conséquence du partage des vecteurs en deux sous-vecteurs de taille $N/2$. C'est ce que l'on appelle une FFT en *base 2* (*radix-2* en anglais). On peut penser utiliser une autre base, par exemple 4, ce qui amène à considérer des sommes des quatre sous-FFT de longueur $N/4$. L'avantage d'un tel choix (par rapport à la base 2) est que l'on évite de faire les calculs évidents des racines quatrièmes de l'unité (qui sont codées simplement par des soustractions à la place d'additions dans les formules), ce qui diminue un peu le nombre d'opérations à effectuer. Par contre, il faut faire attention, car les signes ne sont pas les mêmes pour la transformée directe et pour la transformée inverse. Pour écrire les notations, introduisons les sous-vecteurs f^0, f^1, f^2 et f^3, de longueur $N/4$, qui sont construits à partir de f en ne considérant que les indices congrus respectivement à 0, 1, 2 et 3 modulo 4. On utilise aussi σ qui vaut $+1$ pour la transformée directe, et -1 pour la transformée inverse. Pour discerner les différentes portions de longueur $N/4$ du résultat, on écrira $\widehat{f}^{(0/4)}$ pour le premier quart, etc. Voici les équations :

$$\begin{aligned}
\widehat{f}^{(0/4)} &= \mathscr{S}_{N/4}^{0/4}\widehat{f}^0 + \mathscr{S}_{N/4}^{1/4}\widehat{f}^1 + \mathscr{S}_{N/4}^{2/4}\widehat{f}^2 + \mathscr{S}_{N/4}^{3/4}\widehat{f}^3 \\
\widehat{f}^{(1/4)} &= \mathscr{S}_{N/4}^{0/4}\widehat{f}^0 - \mathrm{i}\sigma\mathscr{S}_{N/4}^{1/4}\widehat{f}^1 - \mathscr{S}_{N/4}^{2/4}\widehat{f}^2 + \mathrm{i}\sigma\mathscr{S}_{N/4}^{3/4}\widehat{f}^3 \\
\widehat{f}^{(2/4)} &= \mathscr{S}_{N/4}^{0/4}\widehat{f}^0 - \mathscr{S}_{N/4}^{1/4}\widehat{f}^1 + \mathscr{S}_{N/4}^{2/4}\widehat{f}^2 - \mathscr{S}_{N/4}^{3/4}\widehat{f}^3 \\
\widehat{f}^{(3/4)} &= \mathscr{S}_{N/4}^{0/4}\widehat{f}^0 + \mathrm{i}\sigma\mathscr{S}_{N/4}^{1/4}\widehat{f}^1 - \mathscr{S}_{N/4}^{2/4}\widehat{f}^2 - \mathrm{i}\sigma\mathscr{S}_{N/4}^{3/4}\widehat{f}^3.
\end{aligned}$$

En choisissant une base p quelconque, et en réalisant des calculs analogues, on peut manipuler des vecteurs de taille p^s, ce qui peut être avantageux. Voici d'ailleurs la formule de récurrence en toute généralité :

Proposition 2.7. *On garde les notations définies précédemment, mais cette fois pour le calcul d'une TFD en utilisant une base $p \geqslant 2$. On a les équations*

$$\forall q = 0,\dots,p-1, \quad \widehat{f}^{(q/p)} = \sum_{k=0}^{p-1} e^{-\sigma \frac{2i\pi}{p} kq} \cdot \mathscr{D}_{N/p}^{k/p} \widehat{f}^k.$$

***Remarque* 2.8.** Bien sûr, cette formule n'est intéressante en pratique que quand on sait calculer explicitement et simplement les facteurs $e^{\frac{2i\pi}{p}kq}$, par exemple pour $p = 2,4,8$. L'exercice III.3 montre comment, en mélangeant à la fois des transformées en base 2 et en base 4, on peut optimiser encore un peu le nombre d'opérations.

2.4 La transformation de Cooley-Tukey

Nous venons donc de voir un algorithme FFT qui permet de calculer très rapidement la transformée de Fourier d'un vecteur dont la taille est 2^p. Mais que se passe-t-il si la taille N du signal ne s'écrit pas sous cette forme ? La solution de facilité, si on se contente de faire des calculs approchés, consiste à ajouter des zéros pour atteindre une taille raisonnable, qui sera bien entendu la puissance de 2 immédiatement après N. Mais souvent, on ne peut pas agir aussi directement, et il faut trouver un algorithme plus fin, pour tirer parti d'autres propriétés de l'entier N. C'est ainsi que de nombreuses autres versions de l'algorithme FFT ont vu le jour depuis l'article fondateur de Cooley-Tukey. Dans ce chapitre, différentes variantes de l'algorithme sont présentées, et certaines permettent réellement de se tirer de mauvaises passes (par exemple l'algorithme de *Good-Thomas* ou celui *split-radix*, présentés aux exercices III.2 et III.3).

Dans le cas où le nombre N est un entier que l'on sait factoriser, il y a cependant une méthode très simple, qui consiste à regarder de plus près le travail effectué par la méthode de Cooley-Tukey dans le cas où $N = 2^s = 2 \times 2^{s-1}$. Ainsi, sans que N soit nécessairement une puissance de 2, supposons que l'on dispose d'une factorisation $N = p \times q$. Dans le cas où les entiers p et q sont premiers entre eux, une remarquable propriété algébrique (le lemme chinois) permet d'optimiser les calculs, et donne naissance à l'algorithme de Good-Thomas déjà cité. Mais pour l'instant, ne nous préoccupons pas de tels raffinements, contentons-nous de suivre pas à pas les transformations déjà effectuées « à la main » au paragraphe 2.1. Rappelons la définition de la TFD d'un vecteur $f \in \mathbb{C}^N$:

$$\widehat{f}[k] \stackrel{\text{déf.}}{=} \sum_{n=0}^{N-1} f[n] \omega_N^{-kn} \qquad \text{pour} \quad k = 0,\dots,N-1. \tag{2.13}$$

L'idée clef pour obtenir une factorisation de cette expression est de réaliser un changement de variables en utilisant les deux bijections suivantes :

$$\varphi : \left\{ \begin{array}{ccc} \{0,\dots,q-1\} \times \{0,\dots,p-1\} & \longrightarrow & \{0,\dots,N-1\} \\ (a,b) & \longmapsto & ap+b \end{array} \right.$$

$$\psi : \left\{ \begin{array}{ccc} \{0,\dots,p-1\} \times \{0,\dots,q-1\} & \longrightarrow & \{0,\dots,N-1\} \\ (c,d) & \longmapsto & cq+d \end{array} \right. .$$

On peut en effet réécrire la somme (2.13) sous la forme

$$\begin{aligned}\widehat{f}[\psi(c,d)] &= \sum_{a=0}^{q-1}\sum_{b=0}^{p-1} \omega_N^{-(ap+b)(cq+d)} f[\varphi(a,b)] \\ &= \sum_{b=0}^{p-1} \omega_N^{-b(d+cq)} \sum_{a=0}^{q-1} \omega_q^{-ad} f[\varphi(a,b)].\end{aligned}$$

Si on note $f_b[a] \overset{\text{déf.}}{=} f[\varphi(a,b)]$ (ce qui correspond à ne prendre qu'une colonne de f, si on la représente sous la forme d'une matrice de taille $p \times q$), alors on obtient

$$\widehat{f}[\psi(c,d)] = \sum_{b=0}^{p-1} \omega_p^{-cb} \left(\omega_N^{-bd} \widehat{f}_b[d] \right). \tag{2.14}$$

Nous avons donc réussi à modifier l'algorithme de calcul pour obtenir un algorithme fonctionnant en 2D, sur la matrice de taille $p \times q$ que constitue $F \overset{\text{déf.}}{=} \{f[\varphi(a,b)]\}_{a,b}$. En fait, si nous n'avions pas les termes parasites ω_N^{-bd} (souvent appelés « twiddle factor » en anglais, voir l'exercice III.3), nous serions simplement en train de calculer la TFD bidimensionnelle de la fonction 2D F (que l'on peut aussi considérer comme une image).

Si l'on compte le nombre d'opérations nécessaires pour calculer la TFD de f par cette méthode, on obtient $Cpq(p+q)$, où C représente une constante prenant en compte le temps de calcul des additions et multiplications complexes. Mais l'intérêt de la méthode est que l'on peut l'appliquer récursivement sur chacune des sous-TFD à calculer. Ainsi, si N se factorise sous la forme $p_1 \times p_2 \times \cdots \times p_s$, on obtient un nombre d'opérations proportionnel à $N\sum p_i$. Bien sûr, dans le cas où $N = 2^s$, on retrouve l'algorithme FFT traditionnel déjà décrit au paragraphe 2.1. Cependant, on voit qu'avec un peu d'adaptation, on peut aisément prendre en compte des N admettant des décompositions plus complexes. Attention cependant à ne pas tomber dans un excès d'optimisme : cette méthode va être totalement inefficace lorsque N se factorise mal. Il faut dans ce cas opter pour d'autres approches, comme celle suggérée à l'exercice V.9. De plus, lorsque la factorisation $N = pq$ possède des particularités (typiquement si p et q sont premiers entre eux), il existe des algorithmes plus optimisés, comme celui de *Good-Thomas* présenté à l'exercice III.2.

2.5 Implémentation concrète

L'implémentation naïve présentée au paragraphe 2.1 (dans le cas $N = 2^p$) souffre de nombreux points faibles, parmi lesquels on peut relever :

- *une structure récursive* : les appels récursifs nécessitent des instructions systèmes supplémentaires, ce qui fait perdre beaucoup de temps.
- *une utilisation de mémoires temporaires* : le calcul explicite des deux sous-vecteurs f^0 et f^1 de taille $N/2$ est à l'évidence une perte de mémoire énorme (puisque l'on crée de l'information redondante).

Nous allons voir dans ce paragraphe comment implémenter une routine qui permet de résoudre ces deux problèmes d'un seul coup. L'idée principale est de réarranger le vecteur de départ. On veut que les éléments du vecteur soient rangés de façon à ce qu'à chaque subdivision (sous forme de deux vecteurs de taille moitié), le premier vecteur soit les $N/2$ premières entrées, et le deuxième vecteur soit les $N/2$ dernières (et non pas les indices

pairs et impairs). Pour une implémentation dans un langage classique (C ou C++ par exemple), le gain sera énorme : par l'utilisation de pointeurs (ou, pour les non-initiés, en déplaçant le début du tableau), la seule mémoire utilisée par le vecteur d'origine permet de loger les deux sous-tableaux.

Dans la suite, nous allons noter les indices sous forme binaire, c'est-à-dire

$$i = [i_{p-1} \dots i_0]_b = \sum_{t=0}^{p-1} i_t 2^t.$$

Notre but est de démarrer l'algorithme avec un vecteur $g \stackrel{\text{déf.}}{=} \{f[n_p(0)], \dots, f[n_p(N-1)]\}$, où $i \mapsto n_p(i)$ désigne une permutation des indices. On veut que lors de l'application de l'équation de *Danielson-Lanczos*

$$\widehat{g}[k] = \widehat{g^0}[k] + \omega_N^{-k} \widehat{g^1}[k], \tag{2.15}$$

le vecteur g^0 soit constitué des entrées de f d'indices $0, \dots, N/2-1$, et que le vecteur g^1 soit constitué des entrées de f d'indices $N/2, \dots, N-1$. Ainsi le partage de g en deux s'effectue sans avoir à déplacer de valeurs dans la mémoire de l'ordinateur. Pour que cette construction marche encore lors des appels récursifs sur g^0 et g^1, ces deux sous vecteurs sont, eux, permutés depuis f^0 et f^1 par n_{p-1}, qui répond aux mêmes exigences que n_p. Cette condition, traduite sur la permutation n_p s'exprime de la façon suivante :

$$n_p([i_{p-1} \dots i_0]_b) = i_0 2^{p-1} + n_{p-1}([i_{p-1} \dots i_1]_b).$$

En itérant cette équation p fois, on trouve l'expression de la permutation n_p :

$$n_p(i) = n_p([i_{p-1} \dots i_0]_b) = \sum_{t=0}^{p-1} i_t 2^{p-1-t}.$$

De façon plus concise, $n_p(i)$ est le transposé de i écrit en binaire. Par exemple, pour $N = 8$, si $i = 6$, qui s'écrit 110 en binaire, alors $n_p(i)$ va s'écrire 011, c'est-à-dire $n_p(6) = 3$.

Au final, on voit que l'on doit classer les éléments du vecteur selon l'écriture binaire renversée des indices. C'est ce que réalise la procédure `rev_bits`, décrite au programme 3.5, annexe A. Cette procédure nécessite $O(N)$ opérations. Pour une implémentation plus fine, on pourra regarder les NUMERICAL RECIPES [31]. La figure 3.3 montre la matrice de permutation correspondant à n_p, c'est-à-dire la matrice $M^{(p)}$ telle que $M_{ij}^{(p)} = \delta_i^{n_p(j)}$. Les points noirs représentent les entrées non nulles (égales à 1) dans la matrice $M^{(p)}$.

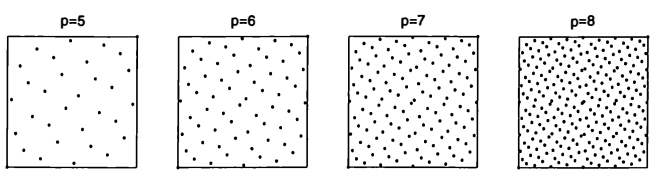

FIG. 3.3 – *Matrice d'inversion de bits*

L'exercice III.1 propose d'écrire une fonction récursive pour effectuer le renversement de bits. L'utilisation de la procédure `rev_bits` permet d'écrire une fonction `fft_dit` qui n'utilise pas de mémoire temporaire. La fin de cette procédure remplace les appels récursifs par des boucles **`for`** imbriquées. La figure 3.4 montre les opérations à effectuer pour inverser les entrées d'un vecteur, en mettant en évidence les permutations nécessaires.

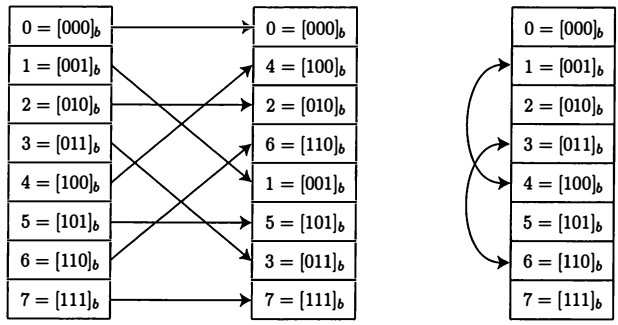

FIG. 3.4 – *Inversion de bits par permutation des entrées*

2.6 Décimation fréquentielle

Nous allons refaire les calculs qui ont mené aux équations (2.11) et (2.12), mais cette fois-ci en effectuant un regroupement selon les fréquences de la transformée. L'algorithme que nous obtiendrons sera en quelque sorte le symétrique de l'algorithme « classique » proposé par COOLEY et TUKEY. Même si cette nouvelle implémentation ne fera pas gagner en vitesse d'exécution, il est important d'avoir à l'esprit les deux versions duales de la FFT, au même titre qu'il est important de maîtriser les propriétés temporelles et fréquentielles de la transformée de Fourier.

Conformément aux notations (2.8), on note f_g (resp. f_d) les $N/2$ premières entrées (resp. $N/2$ dernières) du vecteur f. On a

$$\widehat{f}[k] = \sum_{n=0}^{N/2-1} \left(f_g[n] + e^{-kN/2\frac{2i\pi}{N}} f_d[n] \right) e^{-nk\frac{2i\pi}{N}}.$$

On est donc amené à faire une distinction selon la parité de k. En suivant les notations de l'équation (2.4), on considère $(\widehat{f})^0$ (resp. $(\widehat{f})^1$) la partie paire (resp. impaire) du vecteur transformé. Attention, il ne faut pas confondre ces vecteurs avec $\widehat{f^0}$ et $\widehat{f^1}$, qui sont les transformés des vecteurs f^0 et f^1. On écrit donc, pour $k \in \{0,\ldots,N/2-1\}$,

$$(\widehat{f})^0[k] = \widehat{f}[2k] = \sum_{n=0}^{N/2-1} (f_g[n] + f_d[n])\, e^{-nk\frac{2i\pi}{N/2}}$$
$$(\widehat{f})^1[k] = \widehat{f}[2k+1] = \sum_{n=0}^{N/2-1} e^{-k\frac{2i\pi}{N}} (f_g[n] - f_d[n])\, e^{-nk\frac{2i\pi}{N/2}}.$$

En utilisant l'opérateur $\mathscr{S}_N^x$ introduit en (2.10), on obtient les équations de récurrence suivantes :

$$(\widehat{f})^0 = \mathscr{F}(f_g + f_d)$$
$$(\widehat{f})^1 = \mathscr{F}\left(\mathscr{S}_{N/2}^{1/2}(f_g - f_d)\right).$$

Contrairement à la technique de la décimation temporelle, on voit que les sous-vecteurs dont on doit calculer la transformée de Fourier sont directement obtenus à partir du vecteur d'entrée (il suffit de prendre les parties gauche et droite). Par contre, le vecteur de

sortie doit être composé, selon la parité de l'indice, soit des valeurs d'une transformée soit de l'autre. Pour ne pas avoir à utiliser de mémoire temporaire, nous allons utiliser la même astuce que pour la décimation temporelle, mais dans l'autre sens. Nous allons nous contenter de juxtaposer les deux transformées, c'est-à-dire de mettre à la suite les vecteurs $(\widehat{f})^0$ puis $(\widehat{f})^1$. Pour obtenir le bon résultat, il suffira, à la fin de la procédure, de remettre les fréquences dans le bon ordre, en appelant la fonction `rev_bits`. On peut alors écrire une version non itérative de la FFT qui utilise le principe de décimation fréquentielle, c'est la procédure `fft_dif` qui est écrite au paragraphe 3.3, annexe A.

***Remarque* 2.9.** (**Temporel et fréquentiel**). On voit bien que la décimation fréquentielle est l'exact symétrique de la décimation temporelle. Le fait d'agir sur les indices du vecteur résultat (c'est-à-dire sur les fréquences) au lieu d'agir sur les indices du vecteur d'entrée se traduit par un renversement des bits en phase finale de l'algorithme.

Pour conclure, on peut d'ores et déjà remarquer la grande variété des déclinaisons de l'algorithme FFT à notre disposition. De nombreuses autres méthodes seront en outre décrites dans les chapitres et exercices qui suivent. La littérature tournant autour de la FFT est gigantesque, c'est sans doute l'un des domaines les plus fournis de l'analyse numérique. Des articles récapitulatifs ont été écrits, par exemple par BURRUS [12]. La question est donc de savoir quelle est la meilleure méthode. Bien sûr, il n'y a pas de réponse définitive, car de trop nombreux facteurs entrent en jeu, non seulement concernant la longueur de la transformée et le type de données (réelles, complexes, etc.), mais surtout le type d'architecture (machine, système d'exploitation, architecture parallèle, cache mémoire, etc.) et le type de précision voulue. Dans le doute, mieux vaut rester sur une implémentation simple, mais robuste, quitte à sacrifier un peu d'efficacité.

2.7 Ecriture matricielle

Si l'on écrit la matrice Ω_N de l'opérateur linéaire $\mathscr{F} : \mathbb{C}^N \to \mathbb{C}^N$ dans les bases canoniques, on obtient

$$\Omega_N \overset{\text{déf.}}{=} \begin{pmatrix} 1 & 1 & 1 & & 1 \\ 1 & \omega_N^{-1} & \omega_N^{-2} & & \omega_N^{-(N-1)} \\ 1 & \omega_N^{-2} & \omega_N^{-4} & & \omega_N^{-2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \dots & \omega_N^{-(N-1)(N-1)} \end{pmatrix}. \tag{2.16}$$

Cette matrice correspond à une matrice de *Vandermonde*. Ces matrices interviennent lorsque l'on écrit le système linéaire correspondant à la recherche de l'unique polynôme de degré N passant par N points distincts. Il n'y a rien d'étonnant à cela, puisque nous verrons à la section 5, chap. IV, que le calcul de TFD inverse correspond au calcul des coefficients du polynôme d'interpolation en des points bien particuliers, les racines $N^{\text{ièmes}}$ de l'unité.

La formule de la transformée de Fourier inverse (1.3) se traduit par le fait que l'inverse de la matrice Ω_N est la matrice $\frac{1}{N}\Omega_N^*$, où l'on note $M^* \overset{\text{déf.}}{=} \overline{M}^{\mathrm{T}}$ la matrice adjointe de M. Ceci signifie que la matrice $\frac{1}{\sqrt{N}}\Omega_N$ est unitaire, c'est-à-dire $\Omega_N\Omega_N^* = N\mathrm{Id}_N$. Les équations de *Danielson-Lanczos* (2.11) et (2.12) peuvent alors s'écrire sous la forme d'une factorisation

de la matrice Ω_N :

$$\Omega_N \begin{pmatrix} a_0 \\ \vdots \\ a_{N-1} \end{pmatrix} = \begin{pmatrix} \Omega_{N/2} & \Delta_{N/2}\Omega_{N/2} \\ \Omega_{N/2} & -\Delta_{N/2}\Omega_{N/2} \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_{N-2} \\ a_1 \\ \vdots \\ a_{N-1} \end{pmatrix},$$

où l'on a noté $\Delta_{N/2} = \mathrm{diag}(1, \omega_N^{-1}, \ldots, \omega_N^{-(N/2-1)})$.

3 Convolution circulaire

Nous avons défini au paragraphe 4.3, chap. I, le produit de convolution sur un groupe abélien quelconque, et nous allons maintenant appliquer cette définition ainsi que le théorème de convolution 4.15, chap. I, dans le cas simple d'un groupe cyclique, et plus précisément en employant le langage de la transformée de Fourier discrète qui a été définie au paragraphe 1.

3.1 Convolution circulaire

Commençons par rappeler la définition du produit de convolution ainsi que les principaux résultats déjà obtenus.

Définition 3.1 (Produit de convolution discret). Soient $\{f[n]\}_{n=0}^{N-1}$ et $\{g[n]\}_{n=0}^{N-1}$ deux échantillons discrets (supposés représenter des signaux échantillonnés à des mêmes instants, espacés de façon régulière). On définit le produit de convolution $f * g$ des deux signaux par l'équation

$$(f*g)[n] \overset{\text{déf.}}{=} \sum_{k=0}^{N-1} f[k]g[n-k], \qquad n = 0, \ldots, N-1. \tag{3.1}$$

***Remarque* 3.2.** Dans l'équation (3.1), la quantité $n-k$ est bien sûr calculée modulo N, ce qui revient à considérer les échantillons f et g comme des fonctions périodiques de période N. Cette formule est la traduction de l'équation (4.6), chap. I, dans le cas du groupe $G = \mathbb{Z}/N\mathbb{Z}$, en prenant soin d'utiliser une notation additive à la place de la notation multiplicative. Dans l'optique d'une implémentation informatique, on peut donner une formule plus explicite :

$$(f*g)[n] \overset{\text{déf.}}{=} \sum_{k=0}^{n} f[k]g[n-k] + \sum_{k=n+1}^{N-1} f[k]g[n-k+N], \qquad n = 0, \ldots, N-1.$$

Proposition 3.3. *Le produit de convolution circulaire est commutatif, et l'application* $(f,g) \mapsto f*g$ *munit* $\mathbb{C}^N$ *d'une structure d'algèbre.*

Démonstration. La seule chose non triviale à vérifier est la commutativité, que l'on obtient en faisant le changement de variable $k' = n-k$ dans l'équation (3.1). □

On peut maintenant énoncer le théorème de convolution 4.15, chap. I, en termes de transformée de Fourier discrète.

Proposition 3.4 (Convolution et TFD). *Soient* $\{f[n]\}_{n=0}^{N-1}$ *et* $\{g[n]\}_{n=0}^{N-1}$ *deux échantillons discrets. On a la formule de convolution*

$$\forall n \in \{0,\dots,N-1\}, \quad \widehat{f*g}[n] = \widehat{f}[n]\widehat{g}[n]. \tag{3.2}$$

Démonstration. Pour que les explications soient plus claires, nous allons noter f_1 et g_1 les fonctions de $\mathbb{Z}/N\mathbb{Z}$ dans $\mathbb{C}$ associées aux échantillons f et g (qui sont de taille N). On a alors, pour $n \in \{0,\dots,N-1\}$ (où, en termes de groupe abélien, $n \in \mathbb{Z}/N\mathbb{Z}$),

$$\widehat{f}[n] = \widehat{f_1}(\chi_n) \qquad \text{et} \qquad \widehat{g}[n] = \widehat{g_1}(\chi_n), \tag{3.3}$$

où l'on a noté $\{\chi_0,\dots,\chi_{N-1}\}$ les caractères, c'est-à-dire les éléments du dual $\widehat{\mathbb{Z}/N\mathbb{Z}}$ (voir l'équation (1.2)). En utilisant le théorème 4.15, chap. I, pour les fonctions f_1 et g_1 sur $G = \mathbb{Z}/N\mathbb{Z}$, on obtient

$$\widehat{f_1 * g_1}(\chi_n) = \widehat{f_1}(\chi_n)\widehat{g_1}(\chi_n).$$

Or, on a aussi

$$\widehat{f*g}[n] = \widehat{f_1 * g_1}(\chi_n).$$

Ceci qui permet donc d'écrire, en utilisant les équations (3.3),

$$\widehat{f*g}[n] = \widehat{f_1}(\chi_n)\widehat{g_1}(\chi_n) = \widehat{f}[n]\widehat{g}[n]. \qquad \square$$

***Remarque* 3.5. (Signaux finis et périodisation).** La principale difficulté théorique de la transformée de Fourier discrète est l'assimilation entre notre échantillon $\{f[n]\}_{n=0}^{N-1}$ et une fonction f définie sur $\mathbb{Z}/N\mathbb{Z}$. Cette assimilation a pour avantage d'obtenir à moindres frais des formules algébriques comme le résultat d'inversion 1.4 ainsi que celui de convolution 3.4. Cependant, cette démarche implique que notre fonction f, si on la regarde comme un signal dans le temps est en fait une fonction périodique, de période N. Ceci va à l'encontre de l'intuition naturelle qui veut que l'on considère notre signal (fini) f comme nul en dehors de l'intervalle où il est défini. C'est sur ce point qu'il va falloir faire attention lorsque nous allons vouloir calculer des produits de convolution entre deux signaux finis. C'est justement ce problème qui est soulevé au paragraphe 3.3 lors de l'étude de la convolution non circulaire.

3.2 Calcul avec la FFT

Une implémentation naïve de l'équation (3.1) mène à un nombre d'opérations (multiplications et additions complexes) de l'ordre de $O(n^2)$. En effet, il faut calculer les N valeurs de la convolée, et à chaque fois, une somme de N produits apparaît. Cependant, en utilisant la formule de convolution (3.2) et la formule d'inversion (1.3), on peut écrire une équation qui va s'avérer très utile :

$$f * g = \mathscr{F}^{-1}\left(\widehat{f}\cdot\widehat{g}\right),$$

où l'on a noté f et $g \in \mathbb{C}^N$ deux échantillons de taille N. Grâce à l'algorithme FFT, le calcul des transformées $\widehat{f}$ et $\widehat{g}$ peut se faire en un nombre d'opérations de l'ordre de $O(N\log(N))$, et le calcul du produit $\widehat{f}\cdot\widehat{g}$ nécessite bien sûr seulement N multiplications complexes. Au final, on parvient ainsi à calculer un produit de convolution avec un nombre d'opérations de l'ordre de $O(N\log(N))$.

3.3 Convolution acyclique

Nous allons quitter pour un court instant les transformations liées à la structure de groupe de $\mathbb{Z}/N\mathbb{Z}$ pour définir une opération qui ne respecte pas du tout cette structure cyclique, la convolution acyclique (aussi appelée convolution linéaire), notée $\star$ (à ne pas confondre avec le $*$ de la convolution cyclique). Le support d'un signal $f \in \mathbb{C}^{\mathbb{Z}}$ est défini par

$$\mathrm{Supp}(f) \stackrel{\text{déf.}}{=} \{n \in \mathbb{Z} \setminus f[n] \neq 0\}.$$

Commençons par définir la convolution acyclique pour deux signaux $\{f_1[n]\}_{n\in\mathbb{Z}}$ ainsi que $\{f_2[n]\}_{n\in\mathbb{Z}}$ dont le support est supposé fini, ce qui signifie que $\mathrm{Supp}(f_1)$ et $\mathrm{Supp}(f_2)$ sont des ensembles finis. On définit alors la suite $f_1 \star f_2$ par

$$\forall n \in \mathbb{Z}, \quad f_1 \star f_2[n] = \sum_{k\in\mathbb{Z}} f_1[k] f_2[n-k]. \tag{3.4}$$

Il est à noter que l'on a l'équation très utile :

$$\mathrm{Supp}(f_1 \star f_2) \subset \mathrm{Supp}(f_1) + \mathrm{Supp}(f_2) \stackrel{\text{déf.}}{=} \{n+p \setminus n \in \mathrm{Supp}(f_1), p \in \mathrm{Supp}(f_2)\}.$$

La convolution linéaire n'a donc rien à voir avec à la convolution cyclique, qui, elle, est une opération sur les vecteurs de $\mathbb{C}^N$ (et donne pour résultat un vecteur de $\mathbb{C}^N$). Cependant, en créant à partir de nos deux suites, deux vecteurs $\widetilde{f_1}$ et $\widetilde{f_2}$ de taille N suffisamment grands, nous allons voir que l'on peut calculer les valeurs non nulles de $f_1 \star f_2$ comme certaines entrées du vecteur $\widetilde{f_1} * \widetilde{f_2}$.

Commençons par remarquer que la taille nécessaire pour stocker les entrées de $f_1 \star f_2$ est $N \stackrel{\text{déf.}}{=} N_1 + N_2 - 1$, où l'on a noté N_1 et N_2 les tailles des supports de f_1 et f_2. On peut translater les indices de f_1, ce qui permet de supposer que ces derniers sont $\{0,\ldots,N_1-1\}$. Ceci implique qu'il faut effectuer la même translation sur le vecteur f. Commençons donc par créer un vecteur $\widetilde{f_1} \in \mathbb{C}^N$ en recopiant d'abord les N_1 entrées non nulles de f_1, puis en ajoutant des zéros. La construction du vecteur $\widetilde{f_2}$ est un peu plus difficile, puisqu'il faut tenir compte des indices négatifs. Recopions dans $\widetilde{f_2} \in \mathbb{C}^N$ les entrées d'indices positifs de f_1, puis mettons suffisamment de zéros, puis recopions les entrées d'indices négatifs. De façon plus précise, si on écrit $\mathrm{Supp}(f_2) = \{-P,\ldots,0,\ldots,Q\}$, avec $N_2 = Q+P+1$, alors on aura

$$\widetilde{f_2} \stackrel{\text{déf.}}{=} \{f_2[0], f_2[1], \ldots, f_2[Q], 0, \ldots, 0, f_2[-P], \ldots, f_2[-1]\} \in \mathbb{C}^N.$$

Une fois toutes ces transformations effectuées, on peut enfin écrire :

$$\forall n \in \{0,\ldots,N_1+Q-1\}, \quad f_1 \star f_2[n] = \widetilde{f_1} * \widetilde{f_2}[n].$$

Pour les indices situés dans l'intervalle $\{-P,\ldots,-1\}$, il faut faire attention car, à cause de la convolution circulaire, ils ont été déplacés dans l'intervalle $\{N-P,\ldots,N-1\}$. Cependant, dans la pratique (par exemple, pour le filtrage), on n'utilise que les indices $\{0,\ldots,N_1\}$.

Une fois cette transformation effectuée, on peut bien sûr utiliser l'algorithme présenté à la section 3.2 pour calculer rapidement la convolution. Cet algorithme, qui va de pair avec la technique d'ajout de zéros que nous venons d'expliquer va permettre de réaliser rapidement des filtrages. Tout ceci sera expliqué en détail au paragraphe 2, chap. IV. On pourra noter que lorsque la taille d'un des deux vecteurs est beaucoup plus petite que

celle de l'autre, il existe une stratégie qui permet d'éviter d'ajouter trop de zéros à la fin du vecteur le plus court. Cette méthode est exposée à l'exercice III.4.

Dans la suite, on considérera souvent directement la convolution linéaire de deux vecteurs de $\mathbb{C}^N$, et dans ce cas, les indices négatifs seront placés à la fin du vecteur, (il faudra donc ajouter des zéros entre les indices positifs et ces indices négatifs pour pouvoir utiliser l'algorithme FFT). Il faut cependant bien se rappeller que les convolutions cycliques et acycliques donnent des résultats bien différents. Par exemple, la figure 3.5 montre une comparaison des deux convolutions. Le filtrage par g réalise en quelque sorte une « moyenne locale ». Pour les valeurs centrales de k, plus précisément $2 \leqslant k \leqslant N-4$, on a $f * g[k] = f \star g[k]$. Cependant, pour les valeurs du bord, on trouve des résultats différents. Ainsi, dans la majeure partie des applications où le vecteur x représentera un signal

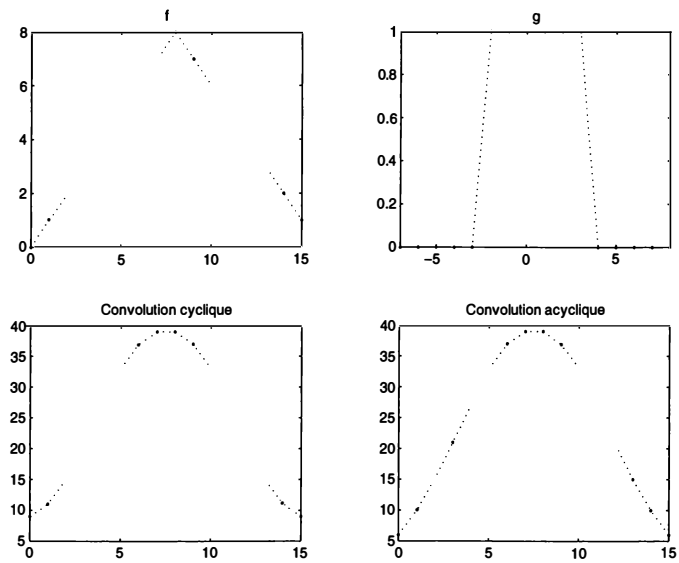

FIG. 3.5 – *Convolutions cyclique et acyclique*

temporel, la convolution acyclique sera préférée, pour ne pas altérer les valeurs sur les bords. Tout ceci sera repris en détail lors de l'explication des différents types de filtrages, à la section 2, chap. IV.

4 En dimension supérieure

> The FFT is part of the revolution in *digital image processing*. A typical image contains a million values; they are the responses to the original image. [...] It hardly needs saying that all deconvolutions are computed by the convolution rule–transform, multiply, and transform back. With the FFT what else would we do?
>
> G. STRANG [68] (1986)

Dans ce paragraphe, pour simplifier les explications, nous allons nous restreindre à des calculs de transformées en dimension 2. La généralisation aux dimensions supérieures,

même si elle peut être périlleuse du point de vue de la programmation, ne présente pas de difficultés théoriques.

4.1 Transformée de Fourier discrète en 2D

Définition 4.1 (TFD bidimensionnelle). Un échantillon bidimensionnel est représenté par une matrice $\{f[i,j]\} \in \mathbb{C}^{N\times P}$.
Les indices sont donc $i \in \{0,\dots,N-1\}$ et $j \in \{0,\dots,P-1\}$. Sa transformée de Fourier discrète est une matrice $N \times P$ définie par

$$\widehat{f}[k,l] \stackrel{\text{déf.}}{=} \sum_{i,j} f[i,j] e^{-\frac{2i\pi}{N}ik} e^{-\frac{2i\pi}{P}jl}, \tag{4.1}$$

où $k \in \{0,\dots,N-1\}$ et $l \in \{0,\dots,P-1\}$.

Comme pour le cas unidimensionnel, on peut encore faire le lien avec la transformée de Fourier sur un groupe abélien, en considérant le groupe $G \stackrel{\text{déf.}}{=} \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/P\mathbb{Z}$. Les caractères de ce groupe sont les χ_{ij}, pour $0 \leqslant i < N$ et $0 \leqslant j < P$, définis par

$$\forall (n,p) \in \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/P\mathbb{Z}, \quad \chi_{ij}(n,p) \stackrel{\text{déf.}}{=} (\omega_N)^{-in} (\omega_P)^{-jp}.$$

On peut donc traduire l'équation (4.1) par

$$\forall k \in \{0,\dots,N-1\}, \forall l \in \{0,\dots,P-1\}, \quad \widehat{f}[k,l] = \widehat{f}(\chi_{kl}),$$

où l'on a noté f à la fois l'échantillon et la fonction associée $f : G \to \mathbb{C}$.

Encore une fois, on constate que la fonction

$$\mathscr{F} : \left\{ \begin{array}{ccc} \mathbb{C}^{N\times P} & \longrightarrow & \mathbb{C}^{N\times P} \\ f & \longmapsto & \mathscr{F}(f) = \widehat{f} \end{array} \right.$$

est un isomorphisme d'algèbre dont on connaît explicitement l'inverse.

Proposition 4.2 (Formule d'inversion 2D). *Soit $f \in \mathbb{C}^{N\times P}$ un échantillon 2D de taille $N \times P$. On a la formule d'inversion*

$$f[i,j] = \frac{1}{NP} \sum_{k,l} \widehat{f}[k,l] e^{\frac{2i\pi}{N}ik} e^{\frac{2i\pi}{P}jl},$$

pour $i \in \{0,\dots,N-1\}$ et $j \in \{0,\dots,P-1\}$.

Le point important est bien sûr de savoir si l'on dispose encore d'un algorithme rapide pour calculer la TFD en dimension deux. La réponse est donnée par une simple réécriture de l'équation (4.1), pour $k \in \{0,\dots,N-1\}$ et $l \in \{0,\dots,P-1\}$:

$$\widehat{f}[k,l] = \sum_{i=0}^{N-1} \left(\sum_{j=0}^{P-1} f[i,j] e^{-\frac{2i\pi}{P}jl} \right) e^{-\frac{2i\pi}{N}ik} = \sum_{i=0}^{N-1} \widehat{F_i}[l] e^{-\frac{2i\pi}{N}ik},$$

où l'on a noté $F_i \in \mathbb{C}^P$ le vecteur formé par la $i^{\text{ième}}$ ligne de la matrice f, et $\widehat{F_i}$ sa TFD unidimensionnelle.

Pour calculer la TFD en 2D d'une matrice f, il suffit donc de calculer la TFD de chacune de ses lignes, puis de calculer la TFD des colonnes de la matrice obtenue. De façon plus synthétique, on peut écrire matriciellement :

$$\widehat{f} = \mathscr{F}_{1D}\left(\mathscr{F}_{1D}(f)^{T}\right)^{T},$$

où l'opérateur $\mathscr{F}_{1D}$ réalise la TFD unidimensionnelle sur les lignes d'une matrice. On peut également effectuer les calculs en sens inverse, c'est-à-dire calculer d'abord la transformée sur les colonnes, puis sur les lignes. Matriciellement, comme ${\Omega_N}^{T} = \Omega_N$, l'équation de transformation s'écrit $\widehat{f} = \Omega_N f \Omega_P$, où Ω_N est défini à l'équation (2.16).

La figure 3.6 montre la transformée de Fourier 2D d'une image, qui est une façon comme une autre de représenter un échantillon 2D (les valeurs de la fonction sont représentées par des niveaux de gris, variant du noir pour 0 au blanc pour 1). On peut interpréter intuitivement le spectre obtenu. La valeur de $\widehat{f}[i,j]$, que l'on peut « lire » directement sur l'image représentant le spectre, correspond à une certaine quantité d'oscillations (bidimensionnelles) présentes dans l'image. Attention, pour la transformée de Fourier (image de droite), les grands coefficients sont représentés en noir. Ces oscillations sont caractérisées par une fréquence, $\frac{1}{N}\sqrt{i^2+j^2}$, et une direction, celle du vecteur (i,j).

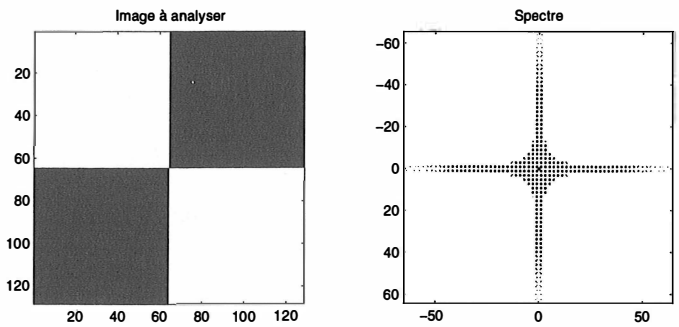

FIG. 3.6 – *Transformée de Fourier 2D*

4.2 Convolution 2D

La convolution entre deux signaux bidimensionnels est une généralisation directe de la convolution cyclique décrite à la section 3.1. Encore une fois, on peut garder en mémoire la définition de la convolution sur un groupe fini (définie au paragraphe 4.3, chap. I). Il s'agit bien sûr de considérer le groupe $G \overset{\text{déf.}}{=} \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/P\mathbb{Z}$. On peut alors interpréter les fonctions de $\mathbb{C}[G]$ comme des images de taille $N \times P$, que l'on aurait étendues par périodicité selon les deux axes. Voici la définition de la convolution entre deux signaux bidimensionnels. On vérifie qu'il s'agit d'une traduction immédiate de la définition donnée dans le cadre des groupes finis abéliens.

Définition 4.3 (Convolution bidimensionnelle). Soient f et g deux échantillons de taille $N \times P$. On définit leur produit de convolution cyclique $f * g$, qui est une matrice de taille $N \times P$, de la façon suivante :

pour $i \in \{0,\dots,N-1\}$ et $j \in \{0,\dots,P-1\}$. Bien sûr, toutes les opérations sur les indices doivent être effectuées modulo N (resp. P) pour les indices de gauche (resp. de droite).

Un calcul naïf du produit de convolution directement par la formule (4.2) nécessite $(NP)^2$ opérations. Dans le but de calculer de façon rapide un tel produit, il faut utiliser la propriété de morphisme d'algèbre de la transformée de Fourier sur un groupe fini, qui est ici rappelée dans le cadre de la transformée de Fourier 2D.

Proposition 4.4 (Convolution 2D et TFD). *Soient f et g deux échantillons de taille $N \times P$. On a la formule de convolution*

$$\forall i \in \{0,\dots,N-1\}, \forall j \in \{0,\dots,P-1\}, \quad \widehat{f * g}[i,j] = \widehat{f}[i,j]\widehat{g}[i,j]. \tag{4.3}$$

Démonstration. La démonstration est la copie conforme de celle de la proposition 3.4. Il convient simplement de changer le cardinal du groupe (qui vaut NP et non plus N), et d'utiliser un indexage adapté pour les caractères et les indices des échantillons, c'est-à-dire $i \in \{0,\dots,N-1\}$ et $j \in \{0,\dots,P-1\}$. □

Ce théorème suggère, pour calculer une convolution, d'utiliser la technique à laquelle nous commençons à être habitués. Il faut dans un premier temps calculer les TFD des deux signaux que l'on souhaite convoler. Ensuite, il faut les multiplier point à point, et enfin calculer la transformée inverse du signal obtenu. On prendra garde au fait que pour implémenter cet algorithme, il faut déplacer les entrées d'indices négatifs dans les deux signaux, de façon à avoir un signal N périodique sur les abscisses, P périodique sur les ordonnées, et avec des indices (i,j) tels que $i \in \{0,\dots,N-1\}$ et $j \in \{0,\dots,P-1\}$.

Nous verrons au paragraphe 2.3, chap. IV, où il sera question de filtrage 2D, quelles sont les propriétés « intuitives » de la convolution cyclique, ainsi que des applications immédiates à l'analyse d'image. On peut cependant donner un exemple de convolution sur des fonctions représentées par leur graphe en 3D. Ainsi la figure 3.7 représente une fonction f irrégulière que l'on a convolée avec une fonction g ayant la forme d'une bosse (et d'intégrale égale à 1). La convolution a un effet de régularisation puisqu'elle réalise une moyenne pondérée de la fonction d'origine au voisinage de chaque point.

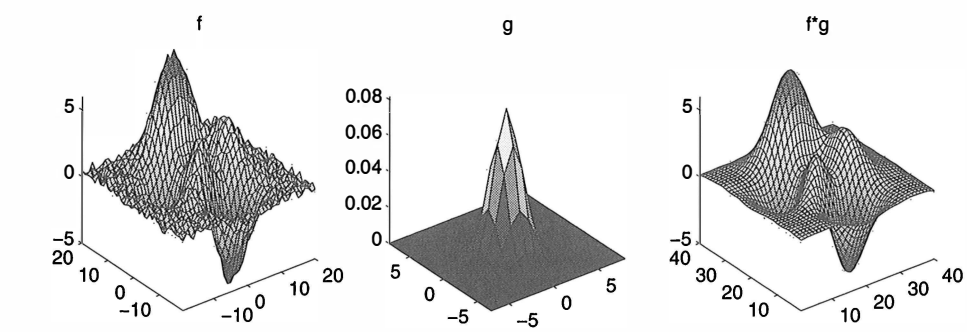

FIG. 3.7 – *Convolution 2D*

5 Symétrie et transformée discrète

Dans ce paragraphe, nous allons donner quelques propriétés annexes de la transformée de Fourier discrète, et les ainsi créer ses vecteurs propres à partir de vecteurs donnés.

5.1 Propriétés de symétrie

Commençons par définir différentes opérations sur les fonctions de $\mathbb{Z}/N\mathbb{Z}$ dans $\mathbb{C}$.

Définition 5.1 (Opérateur de symétrie). Soit $f = \{f[0], \dots, f[N-1]\}$, un vecteur de taille N auquel on associe une fonction périodique f_1, que l'on peut voir comme une fonction $f_1 : \mathbb{Z}/N\mathbb{Z} \to \mathbb{C}$. On définit *la fonction symétrisée* $f^\sharp$ par :

$$\forall n \in \{0, \dots, N-1\}, \quad f^\sharp[n] \overset{\text{déf.}}{=} f_1(-n). \tag{5.1}$$

Ainsi, on a $f^\sharp = \{f[0], f[N-1], f[N-2], \dots, f[1]\}$.
Un vecteur f est dit *symétrique* s'il vérifie $f^\sharp = f$. Il est dit *anti-symétrique* si $f^\sharp = -f$.

Définition 5.2 (Décomposition). Pour $f \in \mathbb{C}^N$, on note f_s et f_a les parties *symétrique* et *anti-symétrique* de f, définies par les équations

$$f_s \overset{\text{déf.}}{=} \frac{1}{2}\left(f + f^\sharp\right),$$
$$f_a \overset{\text{déf.}}{=} \frac{1}{2}\left(f - f^\sharp\right).$$

On a bien sûr la décomposition $f = f_s + f_a$.

Proposition 5.3 (Propriétés de symétrie). *Soit $f \in \mathbb{C}^N$ un échantillon. On a les propriétés suivantes.*

(i) *$\mathscr{F}(f^\sharp) = N\mathscr{F}^{-1}(f)$ ainsi que $\mathscr{F}^2(f^\sharp) = Nf$.*
(ii) *Si f est symétrique, alors $\mathscr{F}^2(f) = Nf$ et $\mathscr{F}(f)$ est symétrique.*
(iii) *Si f est anti-symétrique, alors $\mathscr{F}^2(f) = -Nf$ et $\mathscr{F}(f)$ est anti-symétrique.*
(iv) *Si $f \in \mathbb{R}^N$ est symétrique, alors $\mathscr{F}(f) \in \mathbb{R}^N$.*
(v) *Si $f \in \mathbb{R}^N$ est anti-symétrique, alors $\mathscr{F}(f) \in (\mathrm{i}\mathbb{R})^N$.*

Démonstration. Prouvons (i) et (iv) :
Pour (i), on a

$$\mathscr{F}(f^\sharp) = \sum_{k \neq 0} f[-k]\omega_N^{-kn} + f[0] = \sum_{k=0}^{N-1} f[k]\omega_N^{kn} = \mathscr{F}(f)[n].$$

Pour (iv), si on note $\overline{z}$ le conjugué de $z \in \mathbb{C}$, on a

$$\overline{\mathscr{F}(f)[n]} = \sum_k \overline{f[k]} e^{+\frac{2\mathrm{i}\pi}{N}kn} = \sum_k f^\sharp[k] e^{\frac{2\mathrm{i}\pi}{N}kn} = \mathscr{F}(f^\sharp)[n] = \mathscr{F}(f)[n]. \qquad \square$$

5.2 Valeurs propres de la TFD

L'étude d'un opérateur linéaire est grandement facilitée par la connaissance de ses valeurs propres et des vecteurs propres associés. Bien que la matrice $\frac{1}{\sqrt{N}}\Omega_N$ soit sans doute la matrice unitaire la plus importante, la recherche de ses vecteurs propres est un sujet difficile. Nous allons maintenant donner un moyen simple pour construire des vecteurs propres de la TFD.

Théorème et définition 5.4. Soit $f \in \mathbb{C}^N$ un échantillon. On définit

$$\mathscr{U}_+(f) \overset{\text{déf.}}{=} \sqrt{N} f_s + \mathscr{F}(f_s) \quad \text{et} \quad \mathscr{U}_-(f) \overset{\text{déf.}}{=} \sqrt{N} f_s - \mathscr{F}(f_s)$$
$$\mathscr{V}_+(f) \overset{\text{déf.}}{=} \sqrt{N} f_a + \mathrm{i}\mathscr{F}(f_a) \quad \text{et} \quad \mathscr{V}_-(f) \overset{\text{déf.}}{=} \sqrt{N} f_a - \mathrm{i}\mathscr{F}(f_a).$$

On a alors

$$\mathscr{F}(\mathscr{U}_+(f)) = \sqrt{N}\mathscr{U}_+(f) \quad \text{et} \quad \mathscr{F}(\mathscr{U}_-(f)) = -\sqrt{N}\mathscr{U}_-(f)$$
$$\mathscr{F}(\mathscr{V}_+(f)) = -\mathrm{i}\sqrt{N}\mathscr{V}_+(f) \quad \text{et} \quad \mathscr{F}(\mathscr{V}_-(f)) = \mathrm{i}\sqrt{N}\mathscr{V}_-(f).$$

Ceci signifie que les vecteurs $\mathscr{U}_+(f)$, $\mathscr{U}_-(f)$, $\mathscr{V}_+(f)$ et $\mathscr{V}_-(f)$ sont des *vecteurs propres* de la transformée de Fourier discrète.

Démonstration. Démontrons la première égalité : $\mathscr{F}(\mathscr{U}_+(f)) = \sqrt{N}\mathscr{F}(f_s) + \mathscr{F}^2(f_s)$. Et comme f_s est symétrique, on a $\mathscr{F}^2(f_s) = Nf_s$, d'où le résultat. □

***Remarque* 5.5.** On peut ajouter que les *valeurs propres* que nous venons de trouver sont les seules, puisque la transformée de Fourier vérifie $\mathscr{F}^4(f) = Nf$. Donc ses valeurs propres sont nécessairement des racines $4^{\text{èmes}}$ de N^2.

La figure 3.8 montre les différents vecteurs propres construits à partir de la fonction que l'on peut voir à gauche de la figure 3.9 (c'est-à-dire pour $\lambda = 0$). Cette proposition permet

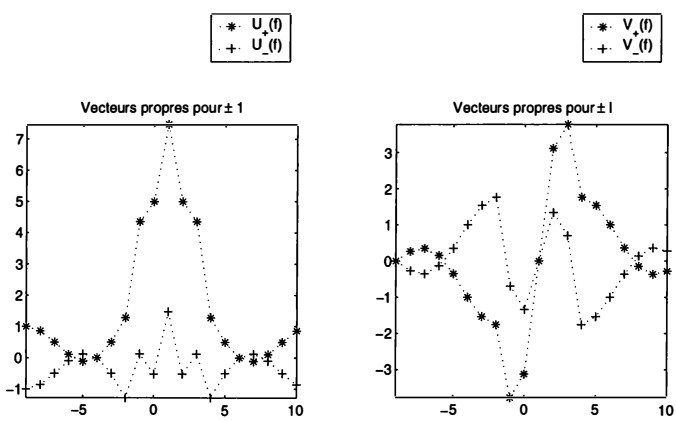

FIG. 3.8 – *Vecteurs propres* $\mathscr{U}_+(f), \mathscr{U}_-(f), \mathscr{V}_+(f)$ *et* $\mathscr{V}_-(f)$

une construction intéressante, simplement en écrivant la décomposition d'un vecteur $f \in \mathbb{C}^N$ en fonction des vecteurs propres de la transformée de Fourier :

$$f = \mathscr{U}_+(f) + \mathscr{U}_-(f) + \mathscr{V}_+(f) + \mathscr{V}_-(f).$$

Ceci permet de considérer l'opérateur $\sqrt{\mathscr{F}}$ défini de la manière suivante :

$$\sqrt{\mathscr{F}}(f) \overset{\text{déf.}}{=} N^{1/4}\mathscr{U}_+(f) + \mathrm{i}N^{1/4}\mathscr{U}_-(f) + (-\mathrm{i})^{1/2}N^{1/4}\mathscr{V}_+(f) + \mathrm{i}^{1/2}N^{1/4}\mathscr{V}_-(f),$$

où l'on aura choisi pour $\mathrm{i}^{1/2}$ une *racine carrée* de i (choix arbitraire).
On a alors $\sqrt{\mathscr{F}} \circ \sqrt{\mathscr{F}} = \mathscr{F}$: l'opérateur $\sqrt{\mathscr{F}}$ est une racine carrée de la transformée de Fourier discrète. De même, pour $\lambda \in \mathbb{R}$, on peut construire ainsi $\mathscr{F}^\lambda$, une transformée $\lambda^{\text{ième}}$ de $\mathscr{F}$ (encore une fois, la construction n'a absolument rien de canonique).

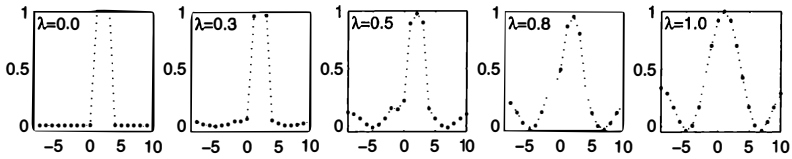

FIG. 3.9 – *Vecteurs transformés intermédiaires* $\mathscr{F}^{\lambda}(f)$ *pour* $\lambda \in [0,1]$

La figure 3.9 montre différentes transformées intermédiaires. Pour $\lambda = 0.5$ on obtient $\sqrt{\mathscr{F}}$. L'exercice III.9 permet de manipuler une *transformée de Fourier partielle*, qui généralise la construction que nous venons d'effectuer. On pourra voir, grâce à l'exemple d'une gaussienne, que ces manipulations correspondent à des notions très intuitives. Pour plus d'informations sur la transformée de Fourier partielle (continue comme discrète), on pourra consulter l'article de CARIOLARO [14]. Enfin, l'exercice III.10 propose une méthode pour diagonaliser de manière canonique la matrice de la TFD.

6 Exercices

Exercice III.1 (Inversion de bits). On définit, pour $n \geqslant 0$, des vecteurs $u^{(n)}$ de taille 2^n, par $u^{(0)} = \{0\}$ et

$$\forall n > 0, \forall k \in \{0,\dots,2^n-1\}, \quad u^{(n)}[k] \stackrel{\text{déf.}}{=} \begin{cases} 2u^{(n-1)}[k] & \text{si } k < 2^{n-1} \\ 2u^{(n-1)}[k-2^{n-1}]+1 & \text{si } k \geqslant 2^{n-1} \end{cases}.$$

1. Calculer la valeur de $u^{(n)}$ pour $n = 1,2,3$.
2. Montrer que $u^{(n)}$ est en fait la suite $0,\dots,2^n-1$, classée en considérant les écritures binaires inversées des entrées.
3. Soit f un vecteur de taille 2^n. On note $\widetilde{f}$ la suite déterminée par

$$\forall k \in \{0,\dots,2^n-1\}, \quad \widetilde{f}[k] \stackrel{\text{déf.}}{=} f[u^{(n)}[k]].$$

   Quelle est l'utilité de $\widetilde{f}$ lors du calcul de la transformée de Fourier discrète de f ?
4. On note f^0 et f^1 les parties paire et impaire de f. On note $\widetilde{f}_d$ et $\widetilde{f}_g$ les parties gauche et droite de $\widetilde{f}$. Quelle relation lie tous ces vecteurs ?
5. Implémenter en MATLAB un algorithme récursif pour calculer $\widetilde{f}$. Comparer sa complexité à celle de la procédure `rev_bits`.

Exercice III.2 (Algorithme de Good-Thomas). Nous avons vu au paragraphe 2.4 que l'algorithme FFT de *Cooley-Tukey* se généralisait sans problème au cas où l'on disposait d'une factorisation adéquate de N, la taille du vecteur transformé. Dans cet exercice, nous allons voir que si certains entiers de cette factorisation sont premiers entre eux, on peut concevoir un algorithme encore plus rapide.

1. On suppose que $N = pq$ où p et q sont deux nombres premiers entre eux. On rappelle le *lemme Chinois*, qui dit que l'application

$$\varphi \begin{cases} \mathbb{Z}/N\mathbb{Z} & \longrightarrow & \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z} \\ n & \longmapsto & (n \mod p, n \mod q) \end{cases}$$

   est un isomorphisme d'anneaux. Expliciter le morphisme inverse ψ.

2. Soit $f \in \mathbb{C}^N$. On définit alors un signal 2D :

$$\forall k_1 \in \{0,\dots,p-1\}, \forall k_2 \in \{0,\dots,q-1\}, \quad F[k_1,k_2] \overset{\text{déf.}}{=} f[\psi(k_1,k_2)].$$

Montrer que l'on a

$$\widehat{f}[(s_1 q + s_2 p) \mod N] = \widehat{F}[s_1,s_2].$$

3. Montrer que lorsque s_1 parcourt $\{0,\dots,p-1\}$ et s_2 parcourt $\{0,\dots,q-1\}$, alors $(s_1 q + s_2 p) \mod N$ parcourt $\{0,\dots,N-1\}$. En déduire comment on peut calculer la transformée de Fourier de f à partir de celle de F, en explicitant le changement d'indices.
4. Quel est le gain par rapport à une étape de l'algorithme FFT classique ? En particulier, que devient l'opérateur $\mathscr{S}_N^x$ introduit à l'équation (2.10) ? Proposer une procédure récursive qui, suivant la factorisation de N obtenue à chaque étape, appelle la procédure de calcul de TFD optimale. En plus des procédures FFT de Cooley-Tukey et Good-Thomas, on pourra inclure la procédure Chirp décrite à l'exercice V.9, qui est intéressante lorsque $N-1$ est un nombre premier.

Exercice III.3 (Algorithme Split-Radix). Nous avons vu au paragraphe 2.3 qu'il était possible d'étendre la méthode de dichotomie de Cooley-Tukey pour calculer des TFD de longueur p^r, en réalisant un regroupement des entrées par paquets de taille p. Dans cet exercice, nous allons montrer comment, en choisissant astucieusement des paquets de tailles variables, on peut réduire le nombre d'opérations. Ce choix part de la constatation que l'algorithme de Cooley-Tukey passe beaucoup de temps à calculer l'opérateur $\mathscr{S}_N^x$, alors que pour certaines valeurs de N (par exemple 2 ou 4), ce dernier est trivial. Dans la littérature anglo-saxonne, on nomme les racines de l'unité ajoutées par cet opérateur « twiddle factors » (littéralement, « qui se tournent les pouces »). On pourra comparer cette approche avec celle de l'algorithme de Good-Thomas, exercice III.2, qui dans le cadre d'une certaine factorisation de N, permet d'éliminer l'opérateur $\mathscr{S}_N^x$.

1. On considère un schéma de décimation fréquentielle. On suppose que N est une puissance de 2. On rappelle que l'algorithme classique DIF organise le regroupement suivant :

$$\left\{\widehat{f}[k] \setminus k = 0,\dots,N-1\right\} = \left\{\widehat{f}[2k] \setminus k = 0,\dots,N/2-1\right\} \bigcup \left\{\widehat{f}[2k+1] \setminus k = 0,\dots,N/2-1\right\}.$$

Expliquer pourquoi l'on n'a pas intérêt à toucher à la première partie de ce regroupement. En ce qui concerne la deuxième partie, on propose le regroupement correspondant à des transformées en base 4, c'est-à-dire :

$$\left\{\widehat{f}[2k+1] \setminus k = 0,\dots,N/2-1\right\} = \left\{\widehat{f}[4k+1] \setminus k = 0,\dots,N/4-1\right\} \bigcup \left\{\widehat{f}[4k+3] \setminus k = 0,\dots,N/4-1\right\}$$

Montrer que ceci mène aux formules de transformation suivantes :

$$\widehat{f}[4k+2j+1] = \sum_{n_1=0}^{\frac{N}{4}-1} \omega_{N/4}^{-kn_1} \omega_N^{-n_1(2j+1)} \sum_{n_2=0}^{3} f[n_1 + n_2 N/4]\, \omega_4^{-n_2(2j+1)},$$

pour $j = 0,1$ et $k = 0,\dots,\frac{N}{4}-1$. Les sommes intérieures sont-elles compliquées à calculer ? Identifier les « twiddle factors ».

2. Trouver d'autres schémas de regroupement. Pourquoi le regroupement par 4 est avantageux ? Calculer le nombre d'opérations nécessaires à chaque fois, et comparer avec celui du schéma DIF classique, en base 2.
3. Transformer les algorithmes décrits plus haut pour obtenir un schéma de décimation temporelle (regroupement des entrées). Décrire une implémentation itérative des algorithmes, en n'oubliant pas les procédures de renversement de bits permettant d'économiser des mémoires temporaires.

Pour plus d'informations, et une généralisation aux TFD de longueur p^r, on pourra consulter [75].

Exercice III.4 (Convolution optimisée). On souhaite calculer la convolution acyclique de deux suites finies f et g de tailles respectives N et M. On suppose que M est beaucoup plus petit que N. Pour simplifier, on suppose que les indices des deux suites commencent à 0.

1. On prend $N = pM$. On note $f_j \overset{\text{déf.}}{=} \{f[k+jM]\}_{k=0}^{M-1}$, pour $j = 0, \ldots, p-1$. Montrer que l'on a
$$f \star g[k] = \sum_{j=0}^{p-1} f_j \star g[k - jM].$$
2. En déduire un moyen rapide de calculer $f \star g$ sans avoir à ajouter $N - M - 1$ zéros à la fin de g. Quelle est la complexité de l'algorithme obtenu ?
3. Dans le cas où N n'est pas un multiple de M, que peut-on faire ?

Exercice III.5 (Matrice circulante). Cet exercice présente de fortes similitudes avec l'exercice I.1 sur les déterminants circulants, avec une présentation utilisant cette fois le produit de convolution. Soit $c \overset{\text{déf.}}{=} (c_0, \ldots, c_{N-1})^{\mathrm{T}} \in \mathbb{C}^N$ un vecteur de taille N. On définit la matrice circulante C qui est associée à ce vecteur par

$$C \overset{\text{déf.}}{=} \begin{pmatrix} c_0 & c_{N-1} & c_{N-2} & \cdots & c_1 \\ c_1 & c_0 & c_1 & \cdots & c_2 \\ \vdots & \vdots & \vdots & & \vdots \\ c_{N-1} & c_{N-2} & c_{N-3} & \cdots & c_0 \end{pmatrix}.$$

1. Soit $\{e_1, \ldots, e_N\}$ la base canonique de $\mathbb{C}^N$. On considère la matrice R dont les colonnes sont $\{e_2, e_3, \ldots, e_N, e_1\}$. On rappelle que Ω_N désigne la matrice de Fourier, qui est définie à l'équation (2.16). Montrer que l'on a
$$\Omega_N R \Omega_N^{-1} = D \qquad \text{avec} \qquad D = \operatorname{diag}(1, \omega_N^{-1}, \ldots, \omega_N^{-(N-1)}).$$
2. Montrer alors que l'on a
$$\Omega_N C \Omega_N^{-1} = \Delta \qquad \text{avec} \qquad \Delta = \operatorname{diag}(\widehat{c}[0], \widehat{c}[1], \ldots, \widehat{c}[N-1]).$$
En déduire que pour $x \in \mathbb{C}^N$, on peut calculer le produit Cx de la manière suivante :
$$Cx = \Omega_N^{-1} \left((\Omega_N c) \cdot (\Omega_N x) \right),$$
où l'on a noté $\cdot$ le produit composante par composante des matrices.
3. Montrer que pour $x \in \mathbb{C}$, on a $Cx = c * x$. En utilisant le théorème de convolution 3.4, en déduire une démonstration immédiate de la question précédente.

Exercice III.6 (Interpolation trigonométrique). Soit $f \in \mathbb{C}^N$ un échantillon de taille $N = 2N_0 + 1$. On définit un vecteur f_0 de taille $P = \eta N$ (avec $\eta \in \mathbb{N}$ suffisamment grand) de la façon suivante :

$$\widehat{f_0} \stackrel{\text{déf.}}{=} \eta \left\{ \widehat{f}[0], \widehat{f}[1], \ldots, \widehat{f}[N_0], 0, \ldots, 0, \widehat{f}[N_0+1], \ldots, \widehat{f}[N-1] \right\}.$$

Montrer que l'on a

$$\forall k \in \{0, \ldots, N-1\}, \quad f[k] = f_0[\eta k].$$

En déduire un algorithme rapide pour interpoler une fonction par des polynômes trigonométriques. On peut voir cet algorithme en action à la figure 3.10.

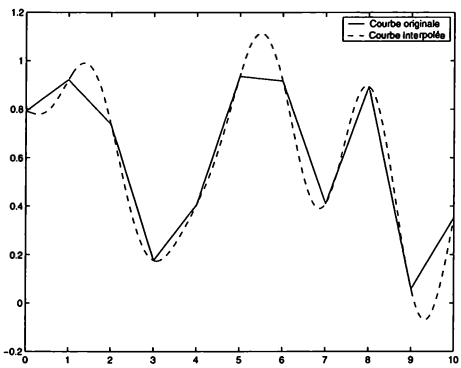

FIG. 3.10 – *Interpolation trigonométrique*

Exercice III.7 (Interpolation de Chebyshev). On définit les polynômes de *Chebyshev* par

$$T_k(X) \stackrel{\text{déf.}}{=} \cos(k \arccos(X)).$$

La figure 3.11 montre les représentations graphiques des polynômes T_k pour des petites valeurs de k. Ce sont des cas particuliers de figures de Lissajous (qui sont utilisées pour étudier les phénomènes ondulatoires), c'est-à-dire des courbes paramétrées du type $(x = a\cos(kt + c), y = b\cos(t))$. On considère une fonction continue $f : [-1,1] \to \mathbb{R}$. On souhaite l'interpoler en N points $\{x_k\}_{k=0}^{N-1}$ par un polynôme P_{N-1} de degré $N-1$, où les x_k sont définis par

$$\forall k \in \{0, \ldots, N-1\}, \quad x_k \stackrel{\text{déf.}}{=} \cos\left((k+1/2)\frac{\pi}{N}\right).$$

1. Montrer que T_N est bien un polynôme, en déterminant une relation de récurrence entre T_k et T_{k-1}. Montrer que les racines de T_N sont les x_k, pour $k \in \{0, \ldots, N-1\}$.
2. Montrer que P_{N-1} peut se mettre sous la forme

$$P_{N-1} = \sum_{k=0}^{N-1} \alpha_k T_k.$$

3. On considère deux types de transformées discrètes en cosinus (souvent notées DCT-2 et DCT-3 en anglais, car il en existe d'autres) d'un échantillon $\{f[k]\}_{k=0}^{N-1} \in \mathbb{R}^N$:

$$\mathscr{C}_2(f)[j] \stackrel{\text{déf.}}{=} \sum_{k=0}^{N-1} f[k] \cos\left((k+1/2)\frac{j\pi}{N}\right)$$

$$\mathscr{C}_3(f)[j] \stackrel{\text{déf.}}{=} \frac{1}{2} f[0] + \sum_{k=1}^{N-1} f[k] \cos\left((j+1/2)\frac{k\pi}{N}\right).$$

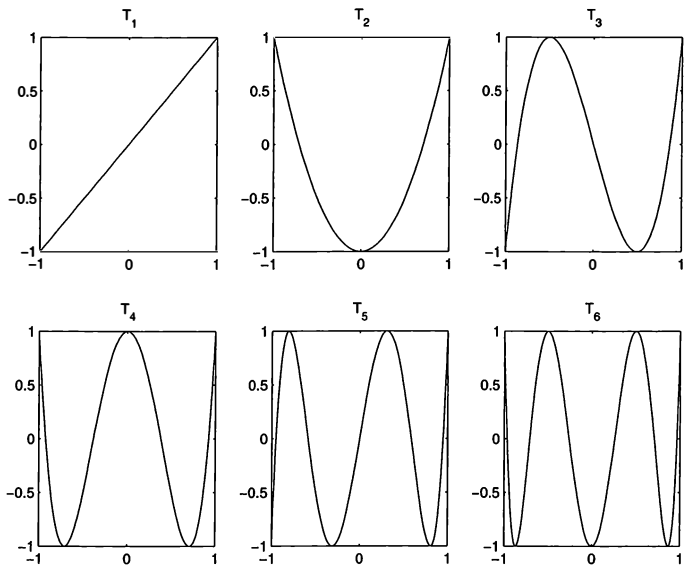

FIG. 3.11 – *Polynômes T_k pour $k = 1, \ldots, 6$*

Montrer que l'inverse de $\mathscr{C}_2$ est $\frac{2}{N}\mathscr{C}_3$. Implémenter en MATLAB ces transformées en utilisant une TFD de taille $4N$ (on pourra penser à rendre le signal pair, puis à insérer des 0 aux indices impairs). Pour plus d'informations sur la transformée en cosinus, on pourra consulter l'article de STRANG [69]. On pourra noter que c'est la transformée $\mathscr{C}_2$ qui est utilisée pour la compression d'images JPEG.

4. Comment calculer les $\{\alpha_k\}_{k=0}^{N-1}$ à partir des $\{f[k] = f(x_k)\}_{k=0}^{N-1}$? En conclure un algorithme rapide d'interpolation polynomiale aux points x_k.

La figure 3.12 montre une comparaison entre l'interpolation de Lagrange (points équidistants) et l'interpolation de Chebyshev sur une fonction en apparence anodine :

$$f(x) \overset{\text{déf.}}{=} \frac{1}{\alpha^2 + x^2} \quad \text{pour} \quad \alpha \in \mathbb{R}_+^*.$$

Pour la figure, on a pris $N = 11$, et $\alpha = 0.3$. Essayer, expérimentalement, de déterminer la valeur α_0 à partir de laquelle le polynôme de Lagrange converge uniformément vers f quand $n \to +\infty$. L'interpolation de Chebyshev est un cas simple de méthode spectrale. Ces méthodes utilisent des décompositions selon des polynômes orthogonaux pour approcher les solutions d'équations aux dérivées partielles. Il s'agit d'une extension des décompositions en séries de Fourier adaptée aux fonctions non périodiques. Tout ceci est très bien décrit dans le livre de BOYD [9].

Exercice III.8 (Dérivation fractionnaire). Soit $f : \mathbb{R} \to \mathbb{R}$ une fonction de classe $\mathbb{C}^\infty$ décroissant rapidement à l'infini. Montrer que la transformée de Fourier (définie à l'équation (1.1), chap. IV) de $f^{(n)}$ est

$$\mathscr{F}(f^{(n)})(\xi) = (-\mathrm{i}\xi)^{-n}\mathscr{F}(f)(\xi).$$

Expliquer en quoi cette propriété permet de définir une *dérivation fractionnaire*, c'est-à-dire que l'on peut définir une dérivation pour des valeurs réelles de n. Implémenter une routine MATLAB qui réalise un calcul approché de dérivée fractionnaire à l'aide de

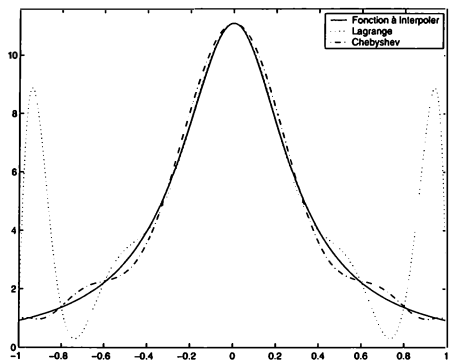

FIG. 3.12 – *Interpolation de Lagrange et de Chebyshev*

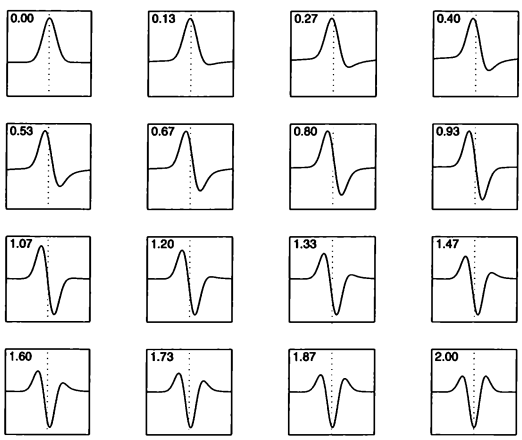

FIG. 3.13 – *Dérivées fractionnaires successives d'une gaussienne*

l'algorithme FFT. La figure 3.13 montre la dérivée fractionnaire d'une gaussienne obtenue par cette méthode, et ceci pour différentes valeurs de n entre 0 et 2.

Exercice III.9 (Transformée de Fourier intermédiaire). On rappelle que l'on note Ω_N la matrice de Fourier, qui est définie à l'équation (2.16). C'est une matrice autoadjointe, donc comme tout endomorphisme normal (c'est-à-dire qui commute avec son adjoint), elle diagonalise en base orthonormée de $\mathbb{C}^N$ (ce qui est faux dans $\mathbb{R}^N$). Ceci signifie qu'il existe une matrice P unitaire et une matrice D diagonale telle que

$$\Omega_N = PDP^*.$$

1. Quelles sont les entrées de D? Vérifier ceci avec MATLAB, en employant la commande `eig` qui fournit les valeurs propres ainsi qu'une décomposition selon les vecteurs propres. On remarquera que comme le nombre de valeurs propres distinctes est inférieur à N, le choix de la base orthonormée de vecteurs propres est totalement arbitraire.
2. On définit la matrice Ω_N^α, pour $\alpha \in \mathbb{R}$, par

$$\Omega_N^\alpha \stackrel{\text{déf.}}{=} PD^\alpha P^*,$$

où D^α est *une* puissance $\alpha^{\text{ième}}$ de D. On définit alors des transformées de Fourier intermédiaires :

$$\forall f \in \mathbb{C}^N, \quad \mathscr{F}^\alpha(f) \stackrel{\text{déf.}}{=} \Omega_N^\alpha f.$$

Montrer que l'on a

$$\forall (\alpha,\beta) \in \mathbb{R}^2, \quad \mathscr{F}^\alpha \circ \mathscr{F}^\beta = \mathscr{F}^{\alpha+\beta} \quad \text{et} \quad \mathscr{F}^1 = \mathscr{F}.$$

3. La figure 3.14 montre le module de la matrice $\mathscr{F}^\alpha$ pour un paramètre α variant entre 0.3 et 1. Que représentent les deux diagonales blanches que l'on peut distinguer (on pourra s'aider du calcul de la matrice Ω_N^2) ?

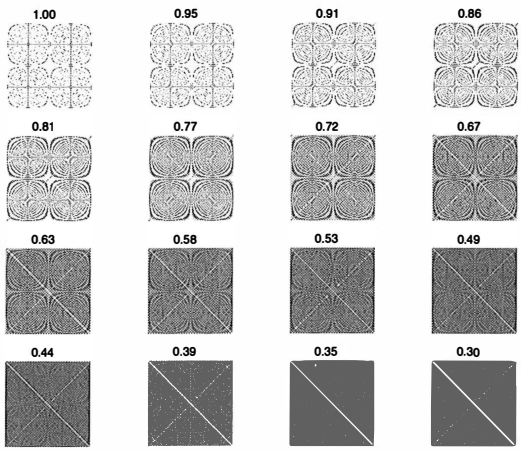

FIG. 3.14 – *Module de différentes matrices de transformées de Fourier partielles*

4. Expliquer pourquoi on peut construire une infinité de transformées intermédiaires. En laissant MATLAB décider d'une factorisation de Ω_N, implémenter la transformée obtenue, puis la tester avec différentes valeurs de α et différents signaux.

La figure 3.15 montre un panel de transformées intermédiaires pour une gaussienne. Le paramètre de transformation α varie entre 0 et 2. Bien sûr, pour $\alpha = 2$, on retrouve le signal d'origine (car la gaussienne est symétrique).

Exercice III.10 (Diagonalisation de la TFD). Dans l'exercice précédent, on a utilisé MATLAB pour diagonaliser en base orthonormée la matrice de la TFD. La construction théorique d'une base orthonormée n'est pas simple, principalement parce que les valeurs propres ont une multiplicité plus grande que 1, ce qui laisse un choix potentiellement infini de décompositions. Le but est donc de construire un procédé canonique pour déterminer une base de diagonalisation. Cet exercice est inspiré de l'article de DICKINSON [27]. On pourra aussi lire l'article de CANDAN [13] qui fait la relation entre la matrice S et la discrétisation d'une équation différentielle.

1. On définit une matrice $S \in M_N(\mathbb{R})$ de la façon suivante :

$$S \stackrel{\text{déf.}}{=} \begin{pmatrix} C_0 & 1 & 0 & & 1 \\ 1 & C_1 & 1 & & 0 \\ 0 & 1 & C_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \dots & C_{N-1} \end{pmatrix} \quad \text{où} \quad C_k \stackrel{\text{déf.}}{=} 2\left(\cos\left(\frac{2k\pi}{N}\right) - 2\right).$$

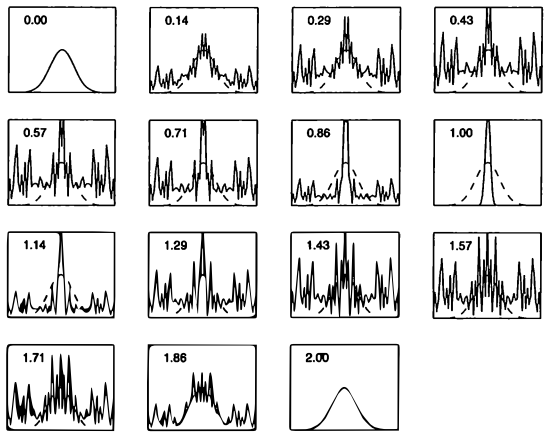

FIG. 3.15 – *Transformées de Fourier partielles successives d'une gaussienne*

Expliquer pourquoi S diagonalise en base orthonormée.

2. Montrer que S et Ω_N, la matrice de Fourier, commutent, c'est-à-dire que $S\Omega_N = \Omega_N S$. On pourra décomposer S en $S = \Gamma + \Delta$, où Γ est une matrice circulante astucieusement choisie de façon à ce que $\Omega_N \Gamma = \Delta \Omega_N$.
3. Montrer que si f et g sont deux endomorphismes diagonalisables de $\mathbb{C}^N$ qui commutent, alors il existe une base commune de diagonalisation.
4. On souhaite montrer que les valeurs propres de S sont distinctes. Soit P la matrice de l'endomorphisme unitaire de $\mathbb{C}^N$ qui envoie un élément f de $\mathbb{C}^N$ sur

$$\forall n \in \{1, \ldots, \lfloor (N-1)/2 \rfloor\}, \quad Pf[n] \overset{\text{déf.}}{=} \frac{f[n] + f[-n]}{\sqrt{2}},$$
$$\forall n \in \{\lceil (N+1)/2 \rceil, \ldots, N-1\}, \quad Pf[n] \overset{\text{déf.}}{=} \frac{f[n] - f[-n]}{\sqrt{2}}.$$

et $Pf[0] = f[0]$. Dans le cas où N est pair, il faut de plus ajouter $Pf[N/2] = f[N/2]$. Montrer que cet opérateur est symétrique et orthogonal, et qu'il correspond à la décomposition de f en ses parties symétrique et antisymétrique. Montrer ensuite que PSP^{-1} est une matrice tridiagonale symétrique.
5. Montrer que les valeurs propres d'une matrice tri-diagonale symétrique à éléments diagonaux non nuls sont distinctes. On pourra s'aider du livre de CIARLET [16] qui décrit la méthode de Givens-Householder pour calculer les valeurs propres d'une matrice symétrique. En conclure que les valeurs propres de S sont bien distinctes.
6. En déduire que l'on a ainsi construit de façon canonique une base orthonormée de vecteurs propres de Ω_N.

Sur la figure 3.16 on peut voir le module des premiers vecteurs propres de la TFD (c'est-à-dire ceux qui ont le moins de changements de signe) construits à l'aide de la méthode que nous venons d'exposer. Sur la figure 3.17 on peut voir la matrice des modules des vecteurs propres de la TFD (les grands coefficients sont noirs).

Exercice III.11 (Orthogonalisation sur un groupe cyclique). Cet exercice étudie dans un cas particulier la notion d'orthogonalisation introduite à l'exercice I.8. Il est cependant indépendant. On considère le groupe fini $G = \mathbb{Z}/n\mathbb{Z}$, ainsi que l'espace vectoriel $\mathbb{C}[G]$ des

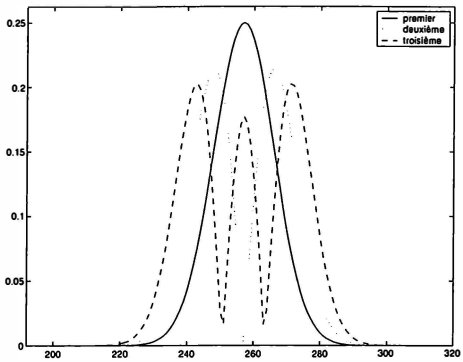

FIG. 3.16 – *Modules de quelques vecteurs propres de* Ω_N

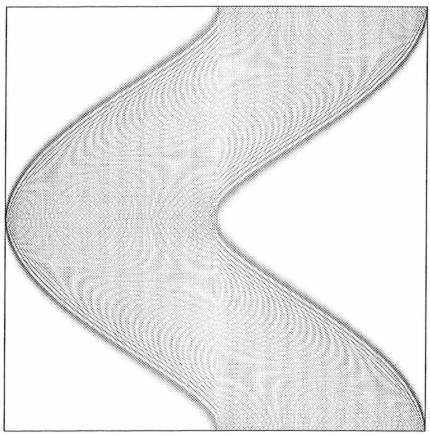

FIG. 3.17 – *Matrice des modules des vecteurs propres orthogonaux*

fonctions de G dans $\mathbb{C}$. Pour $f \in \mathbb{C}[G]$ et $k \in G$, on définit deux actions de G sur $\mathbb{C}[G]$ en posant

$$k \top f : x \mapsto f(x-k) \qquad \text{et} \qquad k \perp f : x \mapsto \omega^{-kx} f(x),$$

où l'on a noté $\omega_n \overset{\text{déf.}}{=} e^{\frac{2i\pi}{n}}$.

1. Montrer que les opérations $\top$ et $\perp$ sont reliées par

$$\mathscr{F}(k \top f) = k \perp \mathscr{F}(f) \qquad \text{et} \qquad \mathscr{F}(k \perp f) = k \top \mathscr{F}(f).$$

2. On rappelle que f est dite orthonormée sous l'action de $\top$ si $\{k \top f\}_{k \in G}$ est une base orthonormée de $\mathbb{C}[G]$. En utilisant la question précédente, expliquer comment sont reliées les bases orthonormées pour $\top$ et les bases orthonormées pour $\perp$.
3. Montrer que f est orthonormée pour $\top$ si et seulement si $\forall k \in G, |\widehat{f}[k]| = 1$.
4. Soit $f \in \mathbb{C}[G]$ telle que $\widehat{f}$ ne s'annule pas. On définit alors $f_0 \in \mathbb{C}[G]$ par

$$\forall k \in G, \quad \widehat{f_0}[k] = \frac{\widehat{f}[k]}{|\widehat{f}[k]|}.$$

Montrer que f_0 est orthonormée pour $\top$. Proposer une construction similaire pour $\perp$.

5. On suppose maintenant que g est orthonormée pour $\top$. Soit $\varphi \in \mathbb{C}[G]$ quelconque. On note, pour $k \in G$, $\mathscr{G}(\varphi)[k] \overset{\text{déf.}}{=} \langle \varphi, k\top g \rangle$ les coefficients de décomposition de φ dans la base orthonormée $\{k\top g\}_{k\in G}$. Montrer que $\mathscr{G}(\varphi) = \frac{1}{n} f * \widetilde{g} =: \frac{1}{n}\mathrm{Corr}(\varphi, g)$, où $\widetilde{g}[k] \overset{\text{déf.}}{=} \overline{g[-k]}$, et Corr est par définition la corrélation de deux vecteurs (voir aussi l'exercice IV.7 pour la corrélation de deux images). En déduire un algorithme rapide de calcul de $\mathscr{G}(\varphi)$ en $\mathrm{O}(n\log(n))$ opérations.

La figure 3.18 montre deux exemples d'orthogonalisation. La fonction du haut, qui est plus proche de l'orthogonalité que celle du bas (on le voit sur les modules des coefficients de Fourier qui sont loin de 1), donne naissance à une fonction g moins oscillante. Intuitivement, pour orthogonaliser une fonction quelconque, il faut la « faire osciller ».

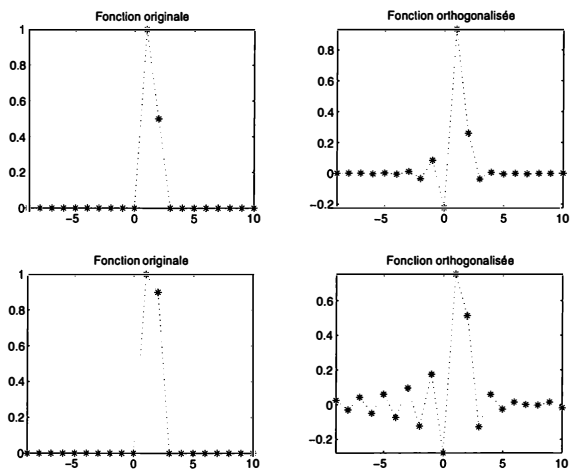

FIG. 3.18 – *Exemples d'orthogonalisation*

Chapitre IV
Applications de la transformée de Fourier discrète

> A paper by Cooley and Tukey [20] described a recipes for computing Fourier coefficients of a time series that used many fewer operations than did the straightforward procedure ... What lies over the horizon in digital signal processing is anyone's guess, but I think it will surprise us all.
>
> B.P. BOGERT (1967)

Nous avons donc vu, au chapitre précédent, que l'on dispose d'un algorithme efficace, l'algorithme FFT, pour calculer la transformée de Fourier discrète. Dès lors, toutes les applications utilisant de près où de loin la théorie de Fourier vont pouvoir bénéficier de cette « trouvaille » algorithmique. Mais ce phénomène va même plus loin, puisque de nombreux autres problèmes, pourtant fort éloignés de l'analyse harmonique, vont être résolus de manière rapide grâce à l'algorithme FFT. Nous verrons ainsi que l'on peut calculer rapidement des produits de grands entiers, ou bien approcher la solution de l'équation de Poisson, ce qui peut paraître quelque peu déconnecté des préoccupations que nous avions jusqu'alors !

1 Lien avec la transformée de Fourier sur $\mathbb{R}$

> To a considerable extent the continuous case can be obtained through a limiting process from the discrete case by dividing the continuum of messages and signals into a large but finite number of small regions and calculating the various parameters involved on a discrete basis.
>
> C. E. SHANNON [66] (1948)

Ce chapitre est avant tout utile pour mieux comprendre de façon intuitive la transformée de Fourier discrète, grâce à de nombreuses analogies avec la transformée de Fourier continue. Il n'est pas là pour donner à la TFD une nature numérique, car il faut avant tout concevoir la transformée discrète comme une transformation algébrique, avec une formule de reconstruction exacte (la transformée de Fourier discrète inverse). Cependant, il est vrai que l'algorithme FFT est souvent employé pour calculer de manière approchée des coefficients de Fourier, même si nous allons vite voir que la formule de quadrature correspondante n'est pas très précise.

1.1 Transformée de Fourier continue

Nous venons de définir, de façon que l'on pourrait qualifier d'abstraite, la transformée de Fourier discrète. On peut donc fort naturellement se demander si cette dernière a un quelconque rapport avec la transformée de Fourier usuelle sur $\mathbb{R}$. Cette dernière, pour une fonction $f \in L^1(\mathbb{R})$ est définie par l'équation

$$\forall \xi \in \mathbb{R}, \quad \widehat{f}(\xi) \stackrel{\text{déf.}}{=} \int_{-\infty}^{+\infty} f(t) e^{-\mathrm{i}\xi t} \mathrm{d}t. \tag{1.1}$$

Cette transformation fonctionnelle a une signification très importante, particulièrement dans le domaine du traitement du signal. Si l'on considère que la fonction f représente un signal continu qui se propage dans le temps, la transformée de Fourier permet de passer d'une représentation temporelle à une représentation fréquentielle. La quantité $\widehat{f}(\xi)$ représente intuitivement combien il y a de variations à la fréquence ξ dans f.

De plus, on peut étendre par densité la transformée de Fourier aux fonctions $f \in L^2(\mathbb{R})$ d'énergie finie, c'est-à-dire telles que $\int_{\mathbb{R}} |f(x)|^2 \mathrm{d}x < +\infty$. Ainsi, la célèbre formule de *Parseval* :

$$\forall f \in L^2(\mathbb{R}), \quad \|\widehat{f}\|_{L^2} = 2\pi \|f\|_{L^2}$$

peut s'interpréter comme une conservation de l'énergie lors du passage du domaine temporel au domaine fréquentiel. Pour plus de détails sur la construction de la transformée de Fourier sur $\mathbb{R}$, on pourra consulter le livre de RUDIN [62].

De même que pour la transformée de Fourier sur un groupe fini, on a aussi un théorème d'inversion, sous des hypothèses un peu restrictives.

Proposition 1.1 (Formule d'inversion). *Lorsque $f \in L^2$ et $\widehat{f} \in L^1$, on a presque partout*

$$f(x) = \frac{1}{2\pi} \int_{\mathbb{R}} \widehat{f}(\xi) e^{\mathrm{i}x\xi} \mathrm{d}\xi. \tag{1.2}$$

En fait, on pourrait refaire une théorie des caractères sur le groupe $(\mathbb{R},+)$ telle qu'elle l'a été faite sur les groupes abéliens finis. Voici par exemple la détermination des caractères de la droite réelle :

Proposition 1.2 (Caractères de $\mathbb{R}$). *On nomme caractère de $(\mathbb{R},+)$ les morphismes continus de $\mathbb{R}$ dans $\Gamma \stackrel{\text{déf.}}{=} \{z \in \mathbb{C} \setminus |z| = 1\}$. Comme d'habitude, on note $\widehat{\mathbb{R}}$ le groupe formé des caractères. Pour $\gamma \in \mathbb{R}$, soit*

$$e_\gamma : \left\{ \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{C}^* \\ t & \longmapsto & e^{\mathrm{i}\gamma t} \end{array} \right. .$$

Alors on a $\widehat{\mathbb{R}} = \{e_\gamma\}_{\gamma \in \mathbb{R}}$ et l'application $\gamma \mapsto e_\gamma$ est un isomorphisme entre $\mathbb{R}$ et $\widehat{\mathbb{R}}$.

Démonstration. Les e_γ sont bien sûr des éléments de $\widehat{\mathbb{R}}$. Soit donc χ un morphisme continu de $\mathbb{R}$ dans Γ. Nous allons commencer par montrer que χ est une fonction dérivable. Pour ce faire, il suffit d'intégrer la propriété d'homomorphisme au voisinage de 0 :

$$\int_0^h \chi(s+t) \mathrm{d}t = \chi(s) \int_0^h \chi(t) \mathrm{d}t.$$

Comme $\chi(0) = 1$, pour h suffisamment petit, on a $\int_0^h \chi(t) \mathrm{d}t \neq 0$. On a donc, pour un certain h,

$$\chi(s) = \frac{\int_s^{h+s} \chi(t) \mathrm{d}t}{\int_0^h \chi(t) \mathrm{d}t},$$

ce qui définit bien une fonction continûment dérivable. Notons $\lambda = \chi'(0)$. En factorisant le taux d'accroissement, on obtient

$$\forall s \in \mathbb{R}, \quad \chi'(s) = \lim_{t\to 0} \frac{\chi(s+t)-\chi(t)}{h} = \chi(s) \lim_{t\to 0} \frac{\chi(t)-\chi(0)}{h} = \lambda\chi(s).$$

On remarque que la seule fonction f qui vérifie $f' = \lambda f$ et $f(0) = 1$ est $s \mapsto e^{\lambda s}$. Il ne reste plus qu'à montrer que $\lambda \in i\mathbb{R}$. Il suffit d'utiliser $\chi(s)\overline{\chi(s)} = \chi(0) = 1$. En différenciant cette égalité en 0, on trouve $\lambda + \overline{\lambda} = 0$. □

Ainsi, la formule d'inversion (1.2) est à rapprocher la formule (4.4), chap. I, sur un groupe fini : on s'est en quelque sorte contenté de remplacer la somme finie par une intégrale. De même, on pourrait analyser la décomposition d'une fonction 2π-périodique en série de Fourier. Cette fois-ci, il s'agirait d'utiliser les caractères du cercle $S^1 \overset{\text{déf.}}{=} \mathbb{R}/2\pi\mathbb{Z}$, qui sont les fonctions

$$\forall n \in \mathbb{Z}, \forall t \in \mathbb{R}, \quad e_n(t) \overset{\text{déf.}}{=} e^{int}.$$

La formule de décomposition d'une fonction périodique en série de Fourier (sous de bonnes hypothèses, et en précisant le sens de la convergence) est une fois encore une formule d'inversion, avec cette fois-ci une somme dénombrable. Pour une introduction aux séries et intégrales de Fourier sur un groupe, on pourra consulter le livre de DYM et MACKEAN [29], chapitre 4.

1.2 Calcul approché de la transformée de Fourier sur $\mathbb{R}$

Avant de se lancer dans des calculs approchés d'intégrales, il faut être conscient que la transformée de Fourier discrète ne se résume pas à de telles approximations. On peut en partie expliquer les propriétés « intuitives » de la transformée de Fourier discrète en invoquant des approximations de la transformée de Fourier continue. Cependant, ce qui fait que la transformée de Fourier discrète marche si bien ne vient pas de sa capacité à approcher fidèlement la transformée continue (c'est même le contraire), mais vient du fait qu'elle transpose les propriétés algébriques que l'on utilise pour la droite réelle (convolution, inversion, translation, ...) au cas d'un domaine fini et cyclique. Ce sont donc bien les propriétés algébriques de la TFD qui en font un outil puissant en analyse numérique, et qui permettent d'avoir des algorithmes simples et rapides. Dans ce paragraphe, nous allons néanmoins expliquer les connexions qui relient, en termes de calculs approchés, les deux transformées, discrète et continue.

Bien sûr, la bonne méthode pour calculer de façon approchée la transformée continue est de chercher la valeur de $\widehat{f}$ en certains points. En effet, on ne connaît en pratique le signal f que sous la forme d'un échantillon $\{f[n]\}_{n=0}^{N-1}$, chaque valeur $f[n]$ étant mesurée pour une valeur du paramètre x égale à $x_n \overset{\text{déf.}}{=} n\Delta$, pour $n = 0, \ldots, N-1$. Δ est le pas de discrétisation, c'est-à-dire l'intervalle (de temps si on considère un signal variant dans le temps) entre deux mesures du signal f.

Dans la suite, en vue de simplifier les explications, on suppose que N est un entier pair. Il est clair qu'étant donné un échantillon de N valeurs du signal f, il est vain de vouloir calculer plus de N valeurs indépendantes de la transformée de Fourier $\widehat{f}$. Cette remarque intuitive va être confirmée par le calcul approché suivant :

$$\forall \xi \in \mathbb{R}, \quad \widehat{f}(\xi) \approx \Delta \sum_{n=0}^{N-1} f[n] e^{-i\xi x_n}.$$

Cette approximation est obtenue en utilisant la méthode des rectangles à gauche pour calculer de manière approchée l'intégrale (1.1). Pour que cette approximation ait un sens, il faut bien sûr qu'en dehors de l'intervalle $[0, N\Delta]$ la fonction f soit sinon nulle, du moins à décroissance très rapide.

On voit alors qu'en calculant les valeurs de $\widehat{f}$ pour des valeurs du paramètre ξ de la forme $\xi_k \overset{\text{déf.}}{=} \frac{2k\pi}{N\Delta}$ on obtient une écriture particulièrement agréable pour le calcul approché de $\widehat{f}(\xi_k)$:

$$\widehat{f}(\xi_k) \approx \Delta \sum_{n=0}^{N-1} f[n] e^{\frac{-2i\pi}{N} kn}. \tag{1.3}$$

Comme nous l'avons déjà mentionné, il est logique de ne calculer que N valeurs de la transformée de Fourier, donc nous allons appliquer le calcul précédent aux points ξ_k pour k variant dans $\{-N/2+1, \ldots, N/2\}$. On pourrait se demander pourquoi on ne commence pas à l'indice $k = -N/2$, mais on voit que l'on obtient le même résultat pour $\xi_{-N/2}$ et pour $\xi_{N/2}$ (ce qui est conforme à notre idée : inutile de calculer plus de N coefficients). En rapprochant l'expression (1.3) et la définition de la transformée de Fourier discrète (1.1), chap. III, on obtient

$$\forall k \in \{0, \ldots, N-1\}, \quad \widehat{f}(\xi_k) \approx \Delta \widehat{f}[k], \tag{1.4}$$

où l'on a noté $\widehat{f}[k]$ la $k^{\text{ième}}$ entrée de la transformée de Fourier discrète du vecteur $\{f[0], \ldots, f[N-1]\} \in \mathbb{C}^N$ (comme défini par l'équation (1.1), chap. III).

Cependant, il y a une légère astuce dans l'équation (1.4). En effet, l'indice k varie dans $\{-N/2+1, \ldots, N/2\}$, alors que le vecteur de transformée discrète $\{\widehat{f}[0], \ldots, \widehat{f}[N-1]\}$ a ses indices dans $\{0, \ldots, N-1\}$. Il est néanmoins facile de voir que l'on n'a pas commis d'erreur en écrivant l'égalité (1.4), puisque le vecteur de transformée discrète peut être vu comme une fonction périodique de période N. On peut donc remplacer les fréquences négatives $\{-N/2+1, \ldots, -1\}$ par les fréquences $\{N/2+1, \ldots, N-1\}$: on obtient bien un vecteur dont les indices varient entre 0 et $N-1$.

Ainsi, la formule (1.4) nous dit qu'à un facteur Δ près, le vecteur de la transformée de Fourier discrète $\{\widehat{f}[0], \ldots, \widehat{f}[N-1]\}$ représente une approximation de la transformée de Fourier du signal f, mais prise à des fréquences bien particulières : les indices n entre 0 et $N/2-1$ correspondent aux fréquences positives entre 0 et $f_c \overset{\text{déf.}}{=} \frac{\pi}{\Delta}$ (exclue), les indices $n = N/2+1, \ldots, N-1$ correspondent aux fréquences négatives entre $-f_c$ (exclue) et 0 (exclue), alors que l'indice $N/2$ correspond à la fois à la fréquence f_c et $-f_c$ (ce qui est normal, puisque le signal discret transformé est périodique de période N). Il faut donc faire attention aux deux points suivants.

- Le vecteur transformé $\{\widehat{f}[0], \ldots, \widehat{f}[N-1]\}$ est considéré comme une donnée périodique de période N (c'est le propre de la transformée de Fourier sur un groupe abélien, en l'occurrence $\mathbb{Z}/N\mathbb{Z}$). Ce n'est bien sûr pas le cas de la fonction $f : \mathbb{R} \to \mathbb{C}$ qui n'a aucune raison d'être périodique de période $2f_c$.
- Par rapport à la transformée de Fourier sur $\mathbb{R}$ du signal continu f, les fréquences sont rangées dans le désordre : fréquences négatives, puis fréquences positives.

1.3 Ajout de zéros

Il y a deux techniques de base pour influer sur la façon dont on peut calculer, à l'aide de la transformée discrète, différentes valeurs de la transformée continue.

- On peut échantillonner le signal avec plus ou moins de précision. Nous avons vu que plus l'échantillonnage est précis (c'est-à-dire plus on prend de points pour représenter la fonction analysée), plus le spectre de la transformée est large. Ainsi, si l'on désire couvrir un spectre deux fois plus grand, il suffit de diviser par deux l'intervalle d'interpolation. Bien sûr, cela rallonge aussi le temps nécessaire pour faire le calcul.
- On peut ajouter des zéros à la fin du vecteur. Si l'on est satisfait du spectre sur lequel on calcule la transformée (plus précisément des fréquences maximales et minimales que l'on peut calculer), on peut ensuite vouloir calculer la transformée avec plus de précision, par exemple si on veut tracer une courbe pour représenter graphiquement la transformée. Dans ce cas, la marche à suivre est simple : il suffit de rajouter des zéros à la fin du vecteur, pour rajouter autant de fréquences intermédiaires calculées.

En jouant sur ces deux paramètres (précision d'interpolation et ajout de zéros), on peut calculer des transformées discrètes « sur mesure », pour avoir une certaine taille de vecteur fixée (ceci peut être utilisé pour créer des filtres, cf. section 2), mais aussi pour la représentation de la transformée. La figure 4.1 montre les différents résultats que l'on peut obtenir en jouant à la fois sur le nombre de points d'échantillonnage et sur l'ajout de zéros. Les fonctions représentées sont les modules de la TFD de la fonction indicatrice de $[0.5, 1]$, échantillonnée sur $[0, 1]$ à intervalles réguliers. Chaque ligne utilise un même nombre de points d'échantillonnage, mais avec plus où moins de zéros ajoutés. L'exercice

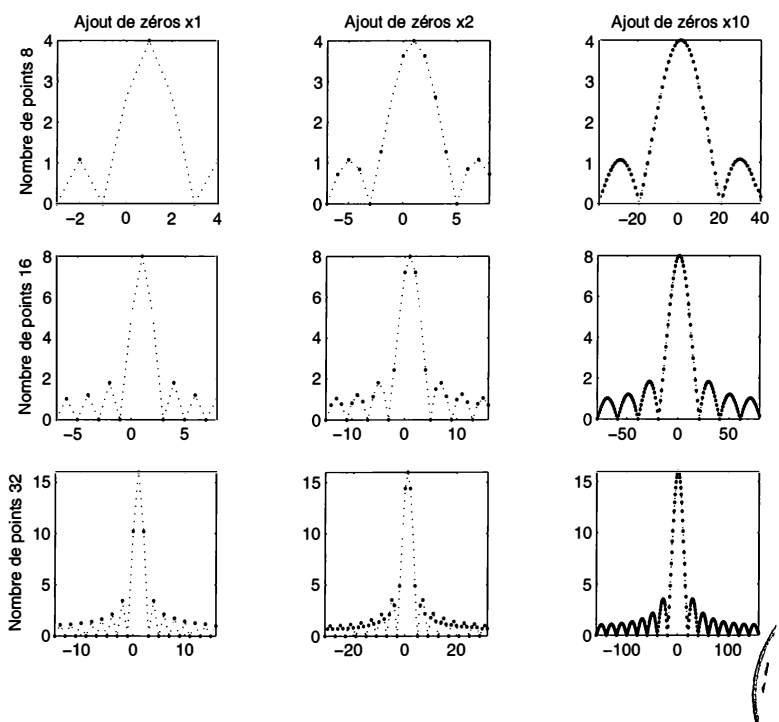

FIG. 4.1 – *Influence de l'échantillonnage et de l'ajout de zéros*

IV.9 est instructif à cet égard, puisqu'il réinvestit tout ceci dans le but de créer et tester un filtre passe-bas.

1.4 Dualité temps/fréquence

> Dans le *temps* à une dimension, la répétition à intervalle égaux est le principe du *rythme*. Pendant qu'une pousse effectue sa croissance, on pourrait dire qu'elle traduit un rythme temporel lent en un rythme spatial.
>
> H. WEYL [77] (1952)

Le calcul approché que nous venons de faire nous permet, via l'utilisation de la transformée de Fourier discrète de calculer la valeur de la transformée de Fourier d'un signal pour certaines fréquences uniquement, précisément celles de la forme $\frac{2k\pi}{N\Delta}$ pour $k \in \{-N/2+1, \ldots, N/2\}$. On remarque donc que plus la précision avec laquelle on réalise le calcul est grande (c'est-à-dire le pas de discrétisation Δ est petit), plus le spectre sur lequel on calcule la transformée est étalé. Ceci n'est pas un phénomène isolé, et résulte d'une dualité très forte entre la fonction de départ et sa transformée de Fourier.

Le résultat suivant illustre bien cette dualité entre les propriétés temporelles d'un signal (c'est-à-dire les propriétés d'une fonction f) et ses propriétés fréquentielles (c'est-à-dire celles de la fonction transformée $\widehat{f}$).

Proposition 1.3. *Soit f une fonction de $L^2(\mathbb{R})$. Alors f et $\widehat{f}$ ne peuvent être simultanément à support compact.*

La démonstration de ce résultat est proposée à l'exercice IV.1. Cette propriété a un analogue discret immédiat, que l'on peut voir en considérant l'impulsion discrète δ_0 (le vecteur qui vaut 1 en 0, et qui est nul partout ailleurs). Sa transformée de Fourier discrète a un spectre qui couvre tout l'intervalle considéré, puisqu'il s'agit de la fonction exponentielle $t \mapsto e^{2i\pi t}$ échantillonnée à intervalles de temps réguliers.

2 Filtrage

La notion de filtrage d'un signal échantillonné est simple à définir, mais est utilisée d'une manière intensive et variée. Nous ne ferons donc pas le tour de la question, cependant, il est intéressant de relier la technique du filtrage avec des outils tels l'algorithme FFT et la convolution linéaire.

2.1 Filtres linéaires

Commençons par définir le filtrage d'un signal théorique infini, avant de nous intéresser au calcul pratique sur des signaux finis. Le filtrage d'un signal $f \in \mathbb{C}^{\mathbb{Z}}$ consiste à convoler (de manière acyclique bien sûr) ce signal avec un autre signal $g \in \mathbb{C}^{\mathbb{Z}}$ fixé à l'avance. On obtient ainsi un opérateur de filtrage

$$\Phi^g : \left\{ \begin{array}{ccc} \mathbb{C}^{\mathbb{Z}} & \longrightarrow & \mathbb{C}^{\mathbb{Z}} \\ f & \longmapsto & f \star g \end{array} \right. .$$

On dit que g est la fonction de transfert du filtre Φ^g.

Une des propriétés remarquables des filtres linéaires est l'obtention de la *réponse impulsionnelle* (c'est-à-dire la façon dont le système que représente le filtre réagit à une impulsion), qui se calcule très simplement :

$$\Phi^g(\delta_0) = \delta_0 \star g = g,$$

où on a noté δ_0 l'impulsion en 0, c'est-à-dire la suite qui vaut 1 en 0, et qui est nulle partout ailleurs (on parle de *Dirac discret*). Autrement dit, la fonction de transfert du filtre n'est rien d'autre que la réponse impulsionnelle.

Une fois exposées toutes ces définitions, on est amené à se poser la question du calcul pratique du filtre. Si l'on veut pouvoir calculer la convolution $f \star g$ en un temps fini, une hypothèse naturelle est d'imposer à la réponse impulsionnelle g d'être finie (plus précisément à support fini). C'est un choix assez radical, mais l'intérêt est qu'il va nous permettre d'appliquer un filtre linéaire à des signaux finis de façon très simple. Il va en effet suffire d'utiliser les techniques de convolution acyclique déjà présentées à la section 3.3, chap. III. Rappelons brièvement la marche à suivre. On suppose que le signal à filtrer est du type : $f = \{f[0], \ldots, f[N-1]\}$, et que la réponse impulsionnelle, elle, est peut être définie pour des indices négatifs, mais est, dans tous les cas, de taille finie P. On doit :

- ajouter assez de zéros au deux vecteurs pour qu'ils atteignent la taille de $N+P-1$;
- déplacer les entrées d'indices négatifs de g à la fin du vecteur (c'est la coutume pour les convolutions cycliques) ;
- calculer la convolution *cyclique* par FFT puis FFT inverse ;
- extraire du résultat les indices qui nous intéressent et les remettre dans l'ordre (si on souhaite récupérer les entrées d'indices négatifs).

L'une des caractéristiques principales des filtres de convolution que nous venons de décrire est qu'ils possèdent une réponse impulsionnelle finie, dans le sens où elle devient nulle en dehors du support de g. C'est pour cela que l'on nomme souvent les filtres de convolution *filtres à réponse impulsionnelle finie*, en anglais *FIR* (pour **F**inite **I**mpulse **R**esponse). Nous verrons au paragraphe 2.2, chap. V, que l'on peut construire de façon simple des filtres qui ne possèdent pas cette propriété, mais qui peuvent quand même être calculés en un temps fini.

Dans la plupart des cas, la fonction de transfert g a un support limité au voisinage de 0 (c'est-à-dire que seules les entrées au voisinage de 0 sont non nulles). Un bon exemple est le filtre gaussien, donné par l'équation

$$\forall k \in \{-N/2, \ldots, N/2\}, \quad g[k] = \frac{1}{M} \exp\left(-\frac{(2k/N)^2}{2t}\right).$$

Le paramètre M est ajusté de sorte que $\|g\|_1 \overset{\text{déf.}}{=} \sum_k |g[k]| = 1$. Le paramètre t représente la variance de la gaussienne. Plus t est petit, plus la gaussienne est « ramassée », donc plus g tend vers le Dirac δ_0 (et le filtre Φ^g tend vers l'identité). Au contraire, plus t devient grand, plus la gaussienne est « étalée », et plus le filtre lisse le signal. Il faut faire attention au fait que les indices de g ont été pris dans $[-N/2, N/2]$, car on souhaite réaliser un filtre acyclique. Pour réaliser un filtre cyclique et/ou calculer le filtre par FFT, il faut bien sûr reporter les fréquences négatives *à la fin* du vecteur.

La figure 4.2 montre différents noyaux gaussiens (en haut). Le noyau gauche a une variance t tellement faible qu'il est égal au Dirac. La ligne du bas montre le filtrage d'un

signal irrégulier (représenté à gauche, puisque le filtrage par un Dirac n'a pas d'effet). On voit que plus la variance est élevée, plus le signal filtré est régulier. Ceci rejoint la théorie classique de la convolution continue : lorsque l'on filtre par un noyau de classe $\mathscr{C}^\infty$ (par exemple une gaussienne), la fonction devient instantanément de classe $\mathscr{C}^\infty$, donc extrêmement lisse !

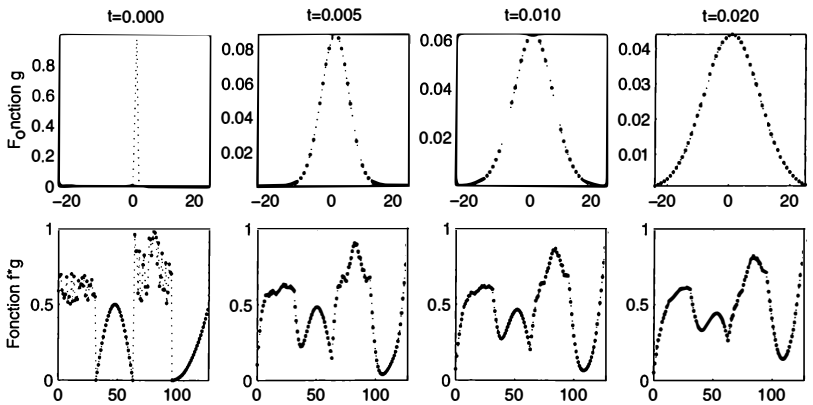

FIG. 4.2 – *Lissage par une gaussienne*

2.2 Types de réponses et stabilité

Nous avons donc vu que la réponse impulsionnelle définit un filtre. Cependant, on peut aussi caractériser ce filtre par d'autres types de réponses, dont notamment :

- *la réponse fréquentielle.* C'est simplement la transformée de Fourier de la fonction de transfert, c'est-à-dire $\sum_{k\in\mathbb{Z}} g[k]e^{ikx}$, pour $x \in [-\pi, \pi]$. C'est une fonction 2π-périodique, puisque c'est la série de Fourier associée aux coefficients $g[k]$. Puisque la réponse impulsionnelle g est finie, on peut la calculer par transformée discrète (FFT), en ajoutant beaucoup de zéros à la fin de g pour avoir une très bonne précision (un tracé quasi-continu de la fonction, cf. paragraphe 1.3). Elle représente la façon dont le filtre opère dans le domaine fréquentiel. En effet, en utilisant le théorème de convolution 3.4, chap. III, on voit que la réponse fréquentielle indique par quelle quantité le filtrage va multiplier l'amplitude de chaque harmonique (c'est-à-dire chaque composante du vecteur transformé) du signal d'origine.
- *la réponse indicielle.* Il s'agit du vecteur obtenu en filtrant un échelon, c'est-à-dire la suite qui est nulle pour les indices négatifs, et qui vaut 1 pour les indices égaux ou supérieurs à 0. Cette réponse indique comment le filtre va réagir face à une discontinuité. Pour un esprit humain normalement constitué, c'est la réponse qui a le plus de sens, puisque l'œil humain est avant tout entraîné à repérer les discontinuités. Ainsi, en observant la réponse indicielle d'un filtre 2D, nous aurons des indications sur la façon dont le filtre va transformer les contours de l'image (là où il y a une variation forte de l'intensité).

D'une façon pragmatique, la réponse fréquentielle se calcule très simplement en employant l'algorithme FFT (en prenant soin d'ajouter suffisamment de zéros pour obtenir

une précision suffisante, et en remettant les fréquences négatives à leur place). Pour la réponse indicielle, il y a au moins deux manières de procéder.

- On peut donner un échelon en entrée du filtre. Pour un filtrage linéaire, il suffit de donner en entrée un vecteur constant égal à 1, l'algorithme étant censé ajouter suffisamment de zéros pour éviter la convolution circulaire.
- Si on connaît la réponse impulsionnelle y_0 et que l'on souhaite calculer la réponse indicielle y_1, il suffit de remarquer que la fonction échelon est la primitive discrète (c'est-à-dire la somme partielle), et donc en utilisant la linéarité du filtre, il en sera de même des deux réponses. On obtient donc la formule simple

$$\forall n \geqslant 0, \quad y_1[n] = \sum_{k=-\infty}^{n} y_0[k].$$

La figure 4.3 montre les trois différents types de réponses pour 3 fonctions de transfert différentes. La première ligne représente une gaussienne, et on constate bien que la réponse impulsionnelle est aussi une gaussienne (ce qui est logique, puisque la transformée d'une gaussienne est gaussienne, voir le lemme 3.10, chap. II). Les deux autres lignes montrent des filtres avec une décroissance moins rapide, et on constate quelques oscillations dans la réponse fréquentielle. Il est à noter que les filtres ne sont pas causaux, c'est-à-dire que les fonctions de transfert sont définies pour des indices négatifs.

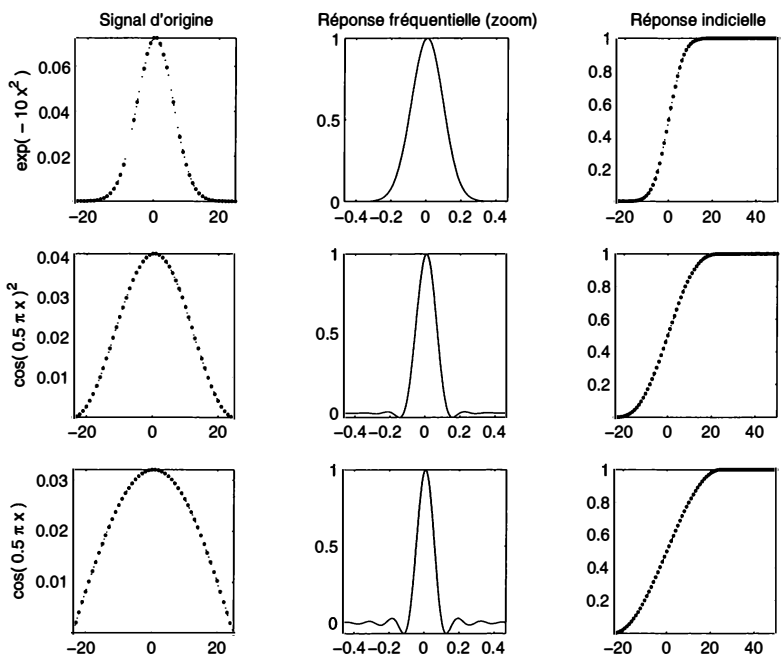

FIG. 4.3 – *Réponses fréquentielles et indicielles*

***Remarque* 2.1. (Domaine temporel et fréquentiel).** Suivant la réponse qui nous intéresse, on peut voir un filtre comme opérant dans le domaine temporel (même si le signal n'est pas échantillonné dans le temps mais par exemple dans l'espace) où bien dans le domaine fréquentiel. Dans certaines applications, il est naturel de créer le filtre en fonction des propriétés temporelles que l'on veut donner au filtre (par exemple pour lisser

des images dans une direction privilégiée). Dans d'autres applications, nous allons nous intéresser au comportement fréquentiel du filtre (à la façon dont le filtre va supprimer les hautes fréquences, pour un filtre passe-bas à la sortie d'un micro). Les remarques sur la dualité temps/fréquence faites au paragraphe 1.4 expliquent que l'on ne peut pas gagner sur les deux tableaux. Par exemple, si l'on souhaite créer un filtre *passe bande* très précis (c'est-à-dire avec un support le plus compact possible), il sera nécessairement peu utilisable dans le domaine temporel, car il aura une réponse impulsionnelle très étendue.

Avant de passer à l'étude des filtres bidimensionnels, présentons brièvement une notion importante qui sera reprise par la suite (lors de l'étude de la transformée en Z, au paragraphe 2, chap. V). Il s'agit de la notion de *stabilité* d'un filtre, qui se définit d'une façon très intuitive.

Définition 2.2 (Stabilité). Un filtre Φ^g est dit stable si pour tout signal borné $f \in \mathbb{C}^{\mathbb{Z}}$, la sortie $\Phi^g(\Phi)$ est aussi bornée.

Un calcul simple montre que

$$\forall n \in \mathbb{Z}, \quad |\Phi^g(f)[n]| \leqslant \sup_{n\in\mathbb{Z}}(|f[n]|) \sum_{k=-\infty}^{+\infty} |g[k]|.$$

En conséquence, pour qu'un filtre soit stable, il suffit que $g \in \ell^1(\mathbb{Z})$, où l'on a noté $\ell^1(\mathbb{Z})$ l'espace des suites absolument sommables. On peut vérifier que cette condition est également nécessaire, en prenant une suite f telle que $f[k]g[-k] = |g[k]|$ (pour les k tels que $g[k] \neq 0$). On a alors, si $g \notin \ell^1(\mathbb{Z})$,

$$\Phi^g(f)[0] = \sum_{k=-\infty}^{+\infty} f[k]g[-k] = \sum_{k=-\infty}^{+\infty} |g[k]| = +\infty.$$

Si $g \in \ell^1(\mathbb{Z})$, l'opérateur de filtrage Φ^g est un endomorphisme continu de l'espace des suites bornées, et sa norme est exactement $\|g\|_{\ell^1}$.

Les filtres linéaires à réponse impulsionnelle finie que l'on a considérés jusqu'à présent sont donc toujours stables. Nous verrons au paragraphe 2.2, chap. V, qu'il est possible de construire des filtres qui ne possèdent pas cette propriété sympathique.

2.3 Filtrage 2D et analyse d'image

On peut bien sûr considérer la convolution bidimensionnelle telle qu'elle est expliquée au paragraphe 4.2, chap. III. Ceci donne naissance à un filtre Φ^g, qui agit sur des signaux bidimensionnels $f : \{0,\dots,N-1\} \times \{0,\dots,N-1\} \to \mathbb{R}$. Ces signaux peuvent être représentés comme des images, puisqu'en chaque *pixel* (i,j) de l'image, on a une intensité lumineuse $f[i,j]$. De façon plus précise, on restreint le plus souvent l'ensemble d'arrivée à un ensemble fini, par exemple $\{0,\dots,255\}$. Ces 256 valeurs représentent les fameux *niveaux de gris* qui seront affichés à l'écran.

Pour « lisser » une image, nous allons une fois de plus considérer des fonctions de transfert g resserrées autour de 0, par exemple une gaussienne :

$$\forall (k,l) \in \{-N/2,\dots,N/2\}^2, \quad g[k,l] = \frac{1}{M} \exp\left(-\frac{(2k/N)^2 + (2l/N)^2}{2t}\right).$$

Le paramètre M est comme dans le cas 1D choisi de tel sorte que $\|g\|_1 = 1$, et t est ajusté en fonction de la puissance du lissage souhaité. La figure 4.4 montre différents noyaux gaussiens (ligne du haut) ainsi qu'une image lissée par ces mêmes noyaux gaussiens (ligne du bas).

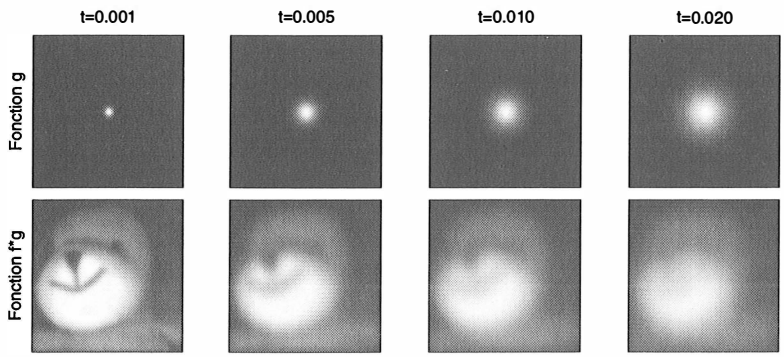

FIG. 4.4 – *Lissage d'une image par une gaussienne 2D*

Une application très courante du filtrage d'image est d'adoucir le bruit présent dans une image naturelle dégradée. C'est ce que montre la figure 4.5, où l'on peut voir *Maurice* dont l'image, de mauvaise qualité, a été améliorée par un filtre gaussien. L'exercice IV.7

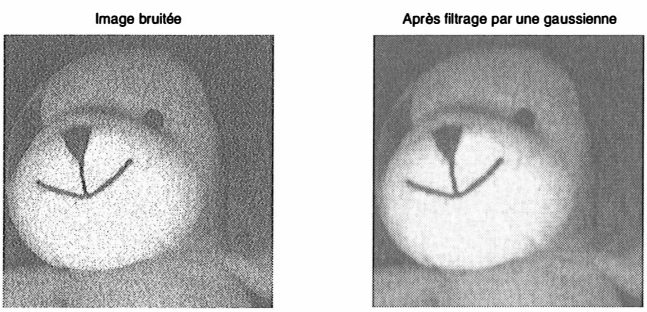

FIG. 4.5 – *Exemple de filtrage d'image*

propose d'aller plus loin dans l'analyse d'image, en appliquant le calcul de corrélation à la recherche de sous-images dans une image plus grande.

3 Aspects géométriques du filtrage

Dans cette section, nous allons aborder le problème du filtrage d'un signal sous un angle original et simple, celui de la géométrie plane. Plutôt que de considérer le signal étudié comme une suite de valeurs espacées dans le temps, on va l'utiliser pour décrire un polygone dessiné dans le plan complexe. Nous allons alors nous intéresser à l'action d'un filtre sur la forme de ce polygone. Plus précisément, nous allons voir comment se comportent les itérés successifs du polygone filtré.

3.1 Filtrage de polygones

Dans la suite de cet exposé, nous allons étudier un polygone à N sommets, Π, que nous allons considérer comme un vecteur $\Pi \in \mathbb{C}^N$. Ainsi, $\Pi[0]$ représentera le premier sommet du polygone, $\Pi[1]$ le deuxième, etc. De façon équivalente, on peut aussi considérer un polygone comme une fonction $\Pi : \mathbb{Z}/N\mathbb{Z} \to \mathbb{C}$. Cette description est très commode, puisqu'elle s'adapte bien à la notion de polygone fermé. En effet, on considère que le sommet $\Pi[i]$ est relié au sommet $\Pi[i+1]$, et on a naturellement envie de relier le sommet $\Pi[N-1]$ au sommet $\Pi[0]$. Ceci revient à considérer un signal N-périodique.

Nous nous intéressons à l'action d'un filtre circulaire Φ^g sur un polygone Π. Nous allons donc considérer les polygones itérés $\Pi^{(k)}$, pour $k \geqslant 0$, qui sont définis de la manière suivante :

$$\begin{cases} \Pi^{(0)} = \Pi \\ \Pi^{(k)} = g * \Pi^{(k-1)} \quad \forall k > 0 \end{cases}, \tag{3.1}$$

où g est la fonction de transfert du filtre. La question naturelle est de savoir si $\Pi^{(k)}$ va tendre vers un certain polygone limite, $\Pi^{(\infty)}$, si les polygones itérés vont rester bornés, où si au contraire ils vont « exploser ». Pour mener à bien cette étude, il suffit de calculer la transformée de Fourier de la relation d'itération (3.1), et on s'aperçoit que

$$\forall k \geqslant 0, \quad \widehat{\Pi^{(k)}} = (\widehat{g})^n \cdot \widehat{\Pi}.$$

L'étude de la convergence de $\Pi^{(k)}$ se fait donc de façon très simple dans le domaine de Fourier. De plus, grâce à la formule d'inversion de la transformée, une convergence dans le domaine de Fourier est équivalente à une convergence du polygone. Voici donc les différents cas qui peuvent se présenter.

- Si $\exists i \in \{0,\dots,N-1\}$ tel que $|\widehat{g}[i]| > 1$: alors les polygones itérés vont exploser. Cela correspond aux cas (a) et (b) de la figure 4.6.
- Si pour tout i, on a soit $|\widehat{g}[i]| < 1$ soit $\widehat{g}[i] = 1$, alors les polygones itérés vont converger vers un polygone $\Pi^{(\infty)}$ qui est défini par

$$\forall i = 0,\dots,N-1, \quad \widehat{\Pi^{(\infty)}}[i] = \begin{cases} 0 & \text{si} \quad |\widehat{g}[i]| < 1 \\ \widehat{\Pi}[i] & \text{si} \quad \widehat{g}[i] = 1 \end{cases}.$$

  Cela correspond au cas (c) de la figure 4.6.
- Si $\forall i \in \{0,\dots,N-1\}$, $|\widehat{g}[i]| \leqslant 1$, mais s'il existe un i tel que $|\widehat{g}[i]| = 1$ et $\widehat{g}[i] \neq 1$, alors les polygones itérés ne vont pas converger, mais ils vont rester bornés. Ceci correspond au cas (d) de la figure 4.6. On peut remarquer que si $\widehat{g}[i]$ est une racine de l'unité, alors le mouvement est périodique (même si le phénomène est difficile à étudier numériquement à cause des erreurs d'arrondi).

On peut donner deux exemples typiques de filtrage de polygones. Ils sont représentés à la figure 4.7.

Exemple 3.1. Le premier correspond au filtre

$$g = \{1/2, 1/2, 0, \dots, 0\}.$$

Ceci consiste à remplacer le polygone Π par la polygone $\Pi^{(1)}$ tel que

$$\Pi^{(1)}[i] = \frac{1}{2}(\Pi[i] + \Pi[i+1]).$$

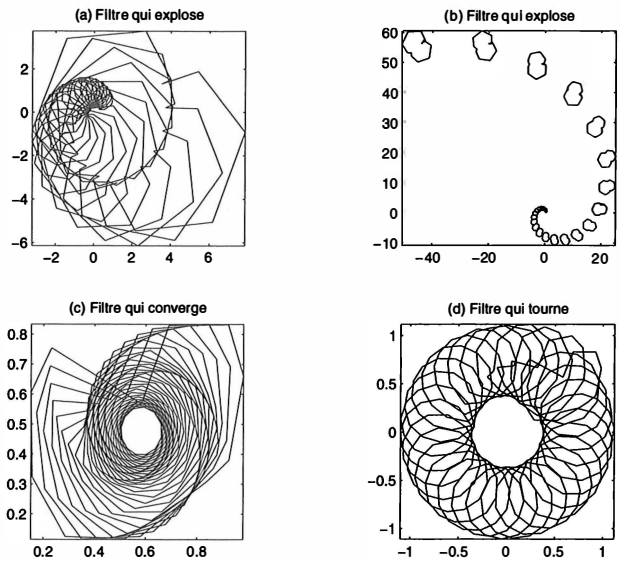

FIG. 4.6 – *Différents cas de filtrages de polygones* .

En quelque sorte, ceci revient à joindre les milieux consécutifs de chaque côté du polygone. De façon intuitive (et sur le dessin de la figure 4.7, à gauche), on à l'impression que les polygones itérés convergent vers le centre du polygone. Confirmons ceci par le calcul :

$$\widehat{g}[k] = \frac{1}{2}\left(1 + e^{-\frac{2i\pi}{N}k}\right) = \cos\left(\frac{k\pi}{N}\right) e^{-\frac{i\pi}{N}k}.$$

Comme on a $\widehat{g}[0] = 1$ et pour $k \geqslant 1$, $|\widehat{g}[k]| < 1$, on conclut donc que les polygones itérés vont converger vers le point $\widehat{\Pi}[0]$, qui correspond au centre de gravité du polygone.

Exemple 3.2. Pour le deuxième exemple, il s'agit d'un filtre qui agit dans le domaine de Fourier de la manière suivante :

$$\widehat{g} = \{0.8, 1, 0.8, \ldots, 0.8\}.$$

Les polygones itérés vont donc converger vers un polygone $\Pi^{(\infty)}$ tel que

$$\widehat{\Pi^{(\infty)}} = \{0, \widehat{\Pi}[1], 0, \ldots, 0\}.$$

On vérifie que ceci correspond à un polygone régulier inscrit dans un cercle de rayon $|\widehat{\Pi}[1]|$, comme on peut le voir à la figure 4.7 (droite).

3.2 Inégalités polygonales

Ce paragraphe est tiré du livre de TERRAS [72]. J'ai souhaité l'insérer dans une étude plus générale de l'analyse de Fourier géométrique, qui fait l'objet de ce paragraphe. Il s'agit d'utiliser la transformée de Fourier afin de démontrer des inégalités de nature euclidienne sur les polygones. L'outil principal sera l'égalité de Plancherel 1.6, chap. III, qui va permettre de démontrer les inégalités en passant dans le domaine de Fourier.

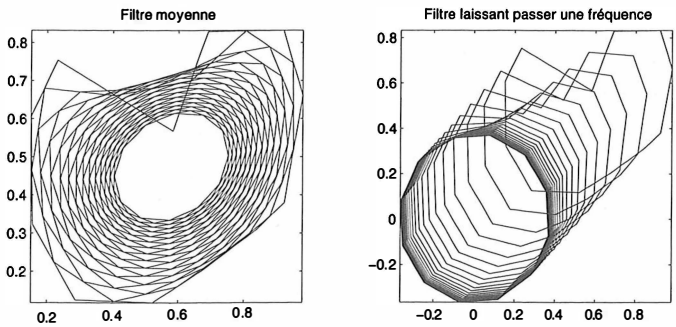

FIG. 4.7 – *Filtrage moyenne et filtrage passe fréquence*

Comme au paragraphe précédent, nous considérons un polygone Π à N côtés, qui peut être vu comme une fonction $\Pi : \mathbb{Z}/N\mathbb{Z} \to \mathbb{C}$. Nous allons définir plusieurs quantités liées à ce polygone. Tout d'abord la somme des carrés des longueurs des côtés :

$$S(\Pi) \stackrel{\text{déf.}}{=} \sum_{i=0}^{N-1} |\Pi[i+1] - \Pi[i]|^2.$$

Ensuite, la somme des carrés des distances au centre de gravité du polygone :

$$T(\Pi) \stackrel{\text{déf.}}{=} \sum_{i=0}^{N-1} |\Pi[i] - \widehat{\Pi}[0]|^2.$$

Enfin, l'aire orientée du polygone :

$$A(\Pi) \stackrel{\text{déf.}}{=} \frac{1}{2} \sum_{i=0}^{N-1} \mathfrak{Im}\left(\overline{\Pi[i]}\Pi[i+1]\right).$$

Proposition 3.3. *On a les inégalités suivantes :*

(i) $A(\Pi) \leqslant \frac{1}{2} T(\Pi)$

(ii) $4\sin^2\left(\frac{\pi}{N}\right) T(\Pi) \leqslant S(\Pi)$.

Démonstration. **Inégalité (i)** : On introduit l'opérateur de décalage T défini par la relation $T\Pi[k] = \Pi[k+1]$, ce qui permet d'écrire

$$A(\Pi) = \frac{1}{2}\mathfrak{Im}\left(\sum_{k=0}^{N-1} \Pi[i]\overline{T\Pi[i]}\right).$$

Il suffit d'utiliser ensuite la formule de Plancherel pour calculer $A(\Pi)$ dans le domaine de Fourier :

$$A(\Pi) = \frac{1}{2}\mathfrak{Im}\left(\sum_{k=0}^{N-1} \widehat{\Pi}[k]\overline{\mathscr{F}(T\Pi)[k]}\right).$$

On utilise ensuite le fait que $\mathscr{F}(T\Pi)[k] = e^{\frac{2i\pi}{N}k}\widehat{\Pi}[k]$ pour obtenir

$$A(\Pi) = \frac{1}{2}\sum_{k=0}^{N-1} \sin\left(\frac{2k\pi}{N}\right) |\widehat{\Pi}[k]|^2.$$

En remarquant que $\sin\left(\frac{2k\pi}{N}\right) \leqslant 1$, on obtient bien l'inégalité voulue, après avoir utilisé une fois de plus la formule de Plancherel.

Inégalité (ii) : Pour ce faire, introduisons le filtre dont la fonction de transfert est égale à $g \stackrel{\text{déf.}}{=} \{-1,1,0,\ldots,0\}$. On peut réécrire la quantité $S(\Pi)$ de la manière suivante :

$$S(\Pi) = \sum_{k=0}^{N-1} |g * \Pi[k]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |\mathscr{F}(g * \Pi)[k]|^2.$$

Pour la dernière égalité, on a utilisé la formule de Plancherel. Calculons le module de la transformée de Fourier de g :

$$\forall k = 1,\ldots,N-1, \quad |\widehat{g}[k]|^2 = |e^{-\frac{2i\pi}{N}k} - 1|^2 = 4\sin^2\left(\frac{k\pi}{N}\right) \geqslant 4\sin^2\left(\frac{\pi}{N}\right).$$

Il ne reste plus qu'à utiliser la propriété de convolution pour obtenir, comme $\widehat{g}[0] = 0$,

$$S(\Pi) = \frac{1}{N} \sum_{k=1}^{N-1} |\widehat{g}[k]|^2 |\widehat{\Pi}[k]|^2 \geqslant \frac{1}{N} 4\sin^2\left(\frac{\pi}{N}\right) \sum_{k=1}^{N-1} |\widehat{\Pi}[k]|^2.$$

On conclut ensuite en utilisant une fois de plus la formule de Plancherel :

$$\frac{1}{N} \sum_{k=1}^{N-1} |\widehat{\Pi}[k]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} \left|\mathscr{F}\left(\Pi[k] - \widehat{\Pi}[0]\right)\right|^2 = T(\Pi).$$

Ce qui termine la démonstration. □

3.3 Descripteurs de Fourier

Pour terminer ce chapitre sur les applications de la théorie de Fourier à la géométrie, nous allons aborder le problème de la *reconnaissance de formes*. Pour être plus précis, on souhaite savoir si deux polygones Π_1 et Π_2 représentent la même forme, à translation, rotation et homothétie près. Nous allons bien sûr essayer de comparer les deux transformées de Fourier $\widehat{\Pi}_1$ et $\widehat{\Pi}_2$.

Il s'agit donc, à partir d'un vecteur transformé, de créer une quantité $\mathscr{D}(\Pi)$ caractérisant un polygone Π à translation et similitude près. Voici les trois opérations à effectuer :

- pour la translation : on sait que seule la composante $\widehat{\Pi}[0]$ est modifiée par une translation. En fait, $\widehat{\Pi}[0]$ représente précisément le centre de gravité du polygone. Nous allons donc purement et simplement ignorer la première entrée du vecteur transformé.
- pour la rotation et l'homothétie : on considère une similitude plane de centre ω, d'angle θ, et de rapport r. Ceci correspond à la transformation $z \mapsto \omega + re^{i\theta}(z-\omega)$. On vérifie que cette transformation change les coefficients de Fourier $\widehat{\Pi}[k]$ (pour $k > 0$) en les multipliant par $re^{i\theta}$. Supposons que $\widehat{\Pi}[1] \neq 0$ (sinon, on prend un autre indice $k_0 > 0$ tel que $\widehat{\Pi}[k_0] \neq 0$). Pour annuler l'effet de l'homothétie, il suffit de considérer les quantités $\frac{\widehat{\Pi}[2]}{\widehat{\Pi}[1]}, \frac{\widehat{\Pi}[3]}{\widehat{\Pi}[1]}, \ldots, \frac{\widehat{\Pi}[N-1]}{\widehat{\Pi}[1]}$.
- pour l'invariante par décalage circulaire des données : on veut que le polygone $\Pi' \stackrel{\text{déf.}}{=} T\Pi$ défini par $\Pi' = \{\Pi[1], \Pi[2], \ldots, \Pi[N-1], \Pi[0]\}$ soit indiscernable du polygone Π, ce qui signifie que $\mathscr{D}(\Pi) = \mathscr{D}(\Pi')$. On constate que $\widehat{\Pi'}[k] = e^{\frac{2i\pi}{N}k}\widehat{\Pi}[k]$. Pour avoir cette invariance, nous allons donc considérer le module des quantités déjà calculées.

Après toutes ces constatations, nous sommes donc amenés à assigner à chaque polygone Π un descripteur de Fourier, que l'on définit de la manière suivante.

Définition 3.4 (Descripteur de Fourier). Soit Π un polygone à N côtés tel que $\widehat{\Pi}[1] \neq 0$. Son descripteur de Fourier $\mathscr{D}(\Pi)$ est un vecteur complexe de taille $N-2$ défini de la manière suivante :

$$\mathscr{D}(\Pi) \stackrel{\text{déf.}}{=} \left\{ \frac{|\widehat{\Pi}[2]|}{|\widehat{\Pi}[1]|}, \frac{|\widehat{\Pi}[3]|}{|\widehat{\Pi}[1]|}, \ldots, \frac{|\widehat{\Pi}[N-1]|}{|\widehat{\Pi}[1]|} \right\}.$$

On définit alors la distance entre deux polygones (on suppose qu'ils vérifient $\widehat{\Pi}_i[1] \neq 0$) Π_1 et Π_2 de la façon suivante :

$$d(\Pi_1, \Pi_2)^2 \stackrel{\text{déf.}}{=} \sum_{k=0}^{N-3} (\mathscr{D}(\Pi_1)[k] - \mathscr{D}(\Pi_2)[k])^2.$$

Deux polygones Π_1 et Π_2 qui sont images l'un de l'autre par une similitude plane vérifient donc $d(\Pi_1, \Pi_2) = 0$. Cependant, on prendra garde au fait que la réciproque est fausse. En effet, si on choisit des nombres $\theta_0, \ldots, \theta_{N-1}$ arbitraires, alors le polygone Π_2 défini par $\widehat{\Pi}_2[k] \stackrel{\text{déf.}}{=} e^{\mathrm{i}\theta_k} \widehat{\Pi}_1[k]$ vérifie $d(\Pi_1, \Pi_2) = 0$. Dans la pratique cependant, la valeur de d donne une bonne idée sur la similitude entre deux formes. C'est ce qu'illustre la figure 4.8, où le deuxième polygone est proche du premier.

4 Résolution numérique d'équations aux dérivées partielles

> On voit donc que les valeurs particulières
>
> $$ae^{-x}\cos y, be^{-3x}\cos 3y, ce^{-5x}\cos 5y, \ldots$$
>
> prennent leur origine dans la question physique elle même, et ont une relation avec les phénomènes de la chaleur. Chacun d'eux exprime un mode simple suivant lequel la chaleur s'établit et se propage dans une lame rectangulaire, dont les côtés, infinis, conservent une température constante.
>
> JOSEPH FOURIER, *Théorie analytique de la chaleur* (1822)

L'une des principales utilités de la transformée de Fourier continue est la résolution d'équations aux dérivées partielles. D'un point de vue purement formel (presque totalement algébrique), elle permet de remplacer un problème complexe (une équation différentielle linéaire) en un problème beaucoup plus simple, une équation polynomiale. Ceci provient du fait que la transformée de Fourier remplace la dérivation par rapport à x en multiplication par $\mathrm{i}x$. Bien sûr, il faut se soucier des problèmes de convergence ainsi que des conditions aux bords, mais la transformée de Fourier s'avère un outil théorique très puissant pour démontrer l'existence de solutions pour de nombreuses équations.

D'un point de vue pratique, le calcul numérique de la solution d'une équation peut, à divers moments du procédé de discrétisation, mener à l'utilisation de la transformée de

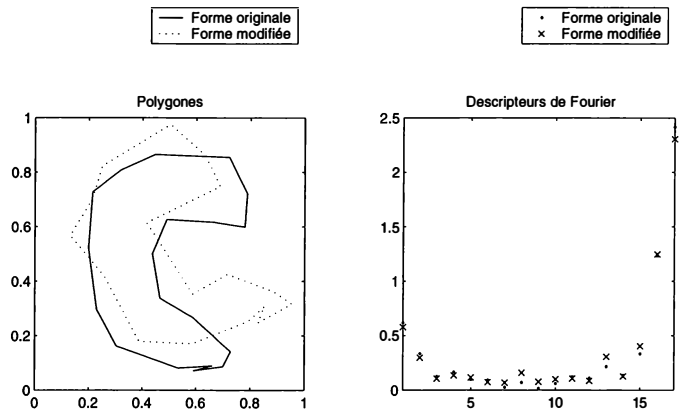

FIG. 4.8 – *Descripteurs de Fourier pour deux formes proches*

Fourier discrète. Ceci peut provenir d'une discrétisation pure et simple de la transformée de Fourier continue (par exemple pour *l'équation de la chaleur*, paragraphe 4.2), ou bien d'une manière plus ingénieuse pour simplifier et accélérer les calculs (par exemple pour *l'équation de Poisson*, paragraphe 4.3). Dans les paragraphes qui suivent, nous allons nous concentrer sur ces aspects numériques ; l'utilisation théorique de la transformée continue est détaillée notamment à l'exercice IV.3.

4.1 Calcul de coefficients de Fourier

Comme nous l'avons vu au paragraphe 1.2, l'algorithme FFT permet de calculer de manière approchée la valeur de la transformée de Fourier continue à certaines fréquences ξ_k. Mais en réalité, le calcul de $\widehat{f}[\xi_k]$ que l'on a effectué correspond à l'approximation de l'intégrale :

$$\widehat{f}(\xi_k) \approx \Delta \int_0^{N\Delta} f(t) e^{-\frac{2i\pi}{N\Delta} kt} \mathrm{d}t. \tag{4.1}$$

Or, si on note f_1 la fonction périodique de période $N\Delta$ qui coïncide avec f sur l'intervalle $[0, N\Delta]$, l'équation (4.1) correspond au calcul de $c_k(f_1)$, le k^{ime} coefficient de Fourier de la fonction f_1.

Résumons tous ces résultats par une formule permettant de calculer de manière approchée N coefficients de Fourier pour une fonction f périodique de période 1 :

$$\forall n \in \{-N/2+1, \ldots, 0, \ldots, N/2\}, \quad c_n(f) \overset{\text{déf.}}{=} \int_0^1 f(t) e^{-2i\pi nt} \mathrm{d}t \approx \frac{1}{N} \widehat{f}[n],$$

où l'on a noté $\widehat{f}$ le vecteur transformé de Fourier du vecteur $\{f(k/N)\}_{k=0}^{N-1}$.

L'algorithme FFT va donc permettre de calculer d'un seul coup N coefficients de Fourier d'une fonction échantillonnée en N points. De plus, les techniques d'ajout de zéros et de raffinement de l'échantillon permettent de moduler le nombre de coefficients calculés pour un même échantillonnage. Le seul problème potentiel réside dans le manque de précision (la méthode des rectangles n'est que d'ordre 1), ce qui peut s'avérer problématique lorsque les coefficients de Fourier décroissent rapidement vers 0, comme c'est le cas pour une fonction très régulière. Deux solutions sont alors possibles :

– augmenter le nombre de points d'interpolation.

– utiliser une méthode d'intégration plus élevée. Il est possible de faire apparaître une transformée de Fourier discrète avec d'autres formules que celle des rectangles (par exemple celle de *Simpson*). L'exercice V.10, question 2, détaille cette technique.

4.2 Application à l'équation de la chaleur

Dans ce paragraphe, nous allons appliquer la méthode décrite au paragraphe précédent, qui permet de calculer d'un seul coup un très grand nombre de coefficients de Fourier (certes avec une précision discutable). Il s'agit de résoudre l'équation de la chaleur, qui historiquement a eu un rôle très important, puisque c'est elle qui a poussé JOSEPH FOURIER à élaborer sa théorie, dans son article *Théorie Analytique de la Chaleur* (1822).

On veut résoudre de manière approchée l'équation de la chaleur sur le cercle $S^1 \overset{\text{déf.}}{=} \mathbb{R}/\mathbb{Z}$:

$$\begin{cases} \forall (t,x) \in \mathbb{R}_*^+ \times S^1, & \dfrac{\partial u}{\partial t} = \kappa \dfrac{\partial^2 u}{\partial x^2} \\ \forall x \in S^1, & u(0,x) = f(x) \end{cases}, \tag{4.2}$$

où la solution cherchée u est supposée suffisamment régulière sur $\mathbb{R}_*^+ \times S^1$, et continue sur $\mathbb{R}^+ \times S^1$. Dans ce paragraphe, nous n'allons pas utiliser de méthode de différence finie, contrairement à ce que nous ferons au paragraphe 4.3 pour résoudre l'équation de Poisson. L'exercice IV.2 étudie la stabilité d'une telle méthode pour l'équation de la chaleur. Nous allons plutôt résoudre de façon explicite l'équation continue, et calculer des approximations de la solution par FFT.

Cette équation traduit l'évolution de la chaleur dans un cercle de longueur 1, totalement isolé, et dont on connaît la répartition initiale de la température. La constante κ traduit la conductivité du matériau, et sans perte de généralité, elle sera prise égale à $\frac{1}{2}$ dans la suite. En effet, on peut remplacer la fonction u par $(t,x) \mapsto u\left(\frac{t}{2\kappa}, x\right)$, ce qui ne modifie pas le problème. Pour un exposé sur les différentes applications des séries et de la transformée de Fourier aux équations différentielles (et en particulier à l'équation de la chaleur), le livre de DYM et MACKEAN [29] est une excellente source. Dans la suite, nous nous contenterons d'énoncer les principaux résultats. En particulier, le résultat d'unicité est détaillé à l'exercice IV.3.

Pour débuter, nous recherchons une solution formelle sous la forme

$$u(t,x) \overset{\text{déf.}}{=} \sum_{n \in \mathbb{Z}} c_n(t) e_n(x) \qquad \text{avec} \quad e_n(x) \overset{\text{déf.}}{=} e^{2i\pi nx},$$

puisque intuitivement, la solution doit être périodique à t fixé. Les coefficients $c_n(t)$ sont définis par

$$\forall t > 0, \quad c_n(t) \overset{\text{déf.}}{=} \int_0^1 u(t,x) e_n(-x) \mathrm{d}x.$$

En faisant deux intégrations par parties, on obtient une équation différentielle vérifiée par c_n :

$$\begin{aligned} \frac{dc_n}{dt}(t) &= \int_0^1 \frac{\partial u}{\partial t}(t,x) e_n(-x) \mathrm{d}x = \int_0^1 \frac{1}{2} \frac{\partial^2 u}{\partial x^2}(t,x) e_n(-x) \mathrm{d}x \\ &= -2\pi^2 n^2 \int_0^1 u(t,x) e_n(-x) \mathrm{d}x = -2\pi^2 n^2 c_n(t). \end{aligned}$$

Comme $c_n(0) = \widehat{f}(n)$ (le $n^{\text{ième}}$ coefficient de Fourier de f), on obtient une solution formelle de l'équation de la chaleur (4.2) :

$$\forall (t,x) \in \mathbb{R}_*^+ \times S^1, \quad u(t,x) = \sum_{n\in\mathbb{Z}} \widehat{f}(n) \exp(-2\pi^2 n^2 t) e_n(x). \tag{4.3}$$

Le fait que la fonction u ainsi définie soit bien solution du problème posé pour $t > 0$ vient du fait que l'on peut dériver sous le signe somme car la série de terme $\exp(-2\pi^2 n^2 t)$ est normalement convergente pour $t \geqslant \varepsilon > 0$, ainsi que toutes ses dérivées par rapport à t. La seule chose difficile à montrer est que l'on a bien

$$\|u(t,\cdot) - f\|_\infty \xrightarrow[t\to 0]{} 0,$$

c'est-à-dire que les conditions initiales sont bien respectées. On rappelle que $\|\cdot\|_\infty$ désigne la norme uniforme sur S^1. Tout ceci est détaillé dans l'exercice IV.3 en même temps que la démonstration de l'unicité de la solution.

***Remarque* 4.1. (Filtrage continu).** De façon intuitive, le passage de $u(0,\cdot) = f$ à $u(t,\cdot)$ correspond à la multiplication des coefficients de Fourier $\widehat{f}(n)$ de f par $\exp(-2\pi^2 n^2 t)$. Ceci revient à filtrer f par une gaussienne, c'est-à-dire à lisser la fonction de départ. Plus t est grand, plus la variance de la gaussienne est forte, donc plus l'effet de flou induit par le filtrage est prononcé. A la limite, lorsque $t \to +\infty$, le filtrage correspond tout simplement à moyenner la fonction, donc la répartition de la chaleur est uniforme.

Dans le but de calculer de façon approchée la solution u de l'équation de la chaleur, nous allons, pour un certain N assez grand fixé (que l'on suppose être une puissance de 2), calculer

$$u_N(t,x) \overset{\text{déf.}}{=} \sum_{n=-N/2}^{N/2-1} c_n(t) e_n(x).$$

Bien sûr, nous allons utiliser la technique développée au paragraphe 4.1 et donc échantillonner la fonction f selon un vecteur $\widetilde{f} \overset{\text{déf.}}{=} \{f(k/n)\}_{k=0}^{N-1}$. Le calcul de la FFT de ce vecteur nous permet, à un facteur $\frac{1}{N}$ près, de calculer de façon approchée N coefficients de Fourier de la fonction f, et donc de construire la fonction u_N. Tout ceci est détaillé dans les programmes MATLAB présentés au paragraphe 6, annexe A. La seule difficulté technique est que les coefficients de Fourier calculés par la FFT ne sont pas rangés dans le bon ordre, mais via l'indexation $\{0,\ldots,N/2-1,-N/2,\ldots,-1\}$. On peut voir l'évolution de la solution dans le temps à la figure 4.9, où la donnée initiale est une fonction indicatrice (donc discontinue). On voit bien l'effet régularisant de l'équation de la chaleur : pour $t > 0$, la solution devient lisse, et tend vers une fonction constante quand t tend vers $+\infty$.

***Remarque* 4.2. (Filtrage discret).** La résolution de l'équation de la chaleur par TFD revient donc à réaliser un filtrage discret avec un filtre passe bas de plus en plus fort. En substance, le fait de filtrer par un filtre passe bas symétrique revient à résoudre l'équation de la chaleur pour un certain temps t. Plus le filtre est régularisant, plus t est grand.

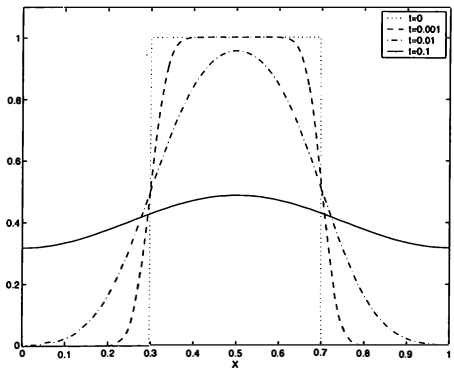

FIG. 4.9 – *Evolution de la solution de l'équation de la chaleur*

4.3 Résolution de l'équation de Poisson par différences finies

On souhaite trouver une fonction $u : [0,1] \times [0,1] \to \mathbb{R}$ suffisamment régulière (de classe $\mathscr{C}^2$) qui satisfasse l'équation de *Poisson* :

$$\begin{cases} \forall (x,y) \in [0,1] \times [0,1], \quad \Delta u(x,y) \overset{\text{déf.}}{=} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} = f(x,y) \\ \forall x \in [0,1], \quad u(x,0) = u_{x0}(x) \quad \text{et} \quad u(x,1) = u_{x1}(x) \\ \forall y \in [0,1], \quad u(0,y) = u_{0y}(y) \quad \text{et} \quad u(1,y) = u_{1y}(y) \end{cases} \tag{4.4}$$

où $f : \mathbb{R}^2 \to \mathbb{R}$ est une fonction continue connue à l'avance, et les fonctions u_{x0}, u_{x1}, u_{0y} et u_{1y} sont aussi des fonctions continues supposées connues (on fixe les valeurs de la solution sur les bords).

L'équation de *Poisson* a des interprétations très importantes, notamment en physique. On peut noter :

- *équation d'une membrane élastique* : la surface de la membrane est représentée par l'équation $z = u(x,y)$. La fonction f représente la quantité surfacique de forces appliquées verticalement à la surface. Le fait d'imposer la valeur de la fonction u sur les bords correspond à fixer les bords de la surface à une armature.
- *équation d'un potentiel électrique* : la quantité $u(x,y)$ représente la valeur d'un potentiel électrique surfacique en un point (x,y), et la fonction f prend en compte une répartition surfacique de charges électriques.

Dans le cas particulier où la fonction f est nulle, on parle d'équation de *Laplace*, et la fonction u est alors appelée *fonction harmonique*. De façon évidente, on montre que la partie réelle d'une fonction holomorphe est harmonique. On peut même montrer que localement, la réciproque est vraie (et donc une fonction harmonique possède des dérivées partielles à tout ordre !). Pour plus de détails sur la théorie des fonctions harmoniques, on pourra consulter le livre de RUDIN [62, p.275].

On se propose d'approcher la solution u de l'équation (4.4) par la méthode des *différences finies*. Pour ce faire, nous allons discrétiser le carré $[0,1] \times [0,1]$ selon $N+1$ points sur les deux directions. On cherche donc une solution approchée $\{U(i,j)\}_{0 \leqslant i,j \leqslant N}$, où $U(i,j)$ représente une approximation de $u(ih, jh)$ (on a noté $h \overset{\text{déf.}}{=} \frac{1}{N}$ le pas de la subdivision).

En replaçant le Laplacien Δu par une approximation discrète, on obtient, pour les points intérieurs au carré les équations suivantes, pour i et j dans $\{1,\ldots,N-1\}$,

$$\frac{1}{h^2}\{U(i-1,j)+U(i+1,j)+U(i,j+1)+U(i,j-1)-4U(i,j)\}=F(i,j), \qquad (4.5)$$

où l'on a noté $F(i,j) \overset{\text{déf.}}{=} f(ih,jh)$ le membre de droite de l'équation. On a bien sûr fait attention aux termes de bord ($i,j=0,N$), qui ne font pas partie du système puisqu'ils sont fixés une fois pour toutes par les conditions au bord.

On serait ainsi tenté d'écrire l'équation (4.5) comme une convolution par un filtre Φ :

$$U * \Phi = F$$

où $*$ désigne l'opérateur de convolution circulaire 2D, défini au paragraphe 4.2, chap. III, et la fonction de transfert s'écrit

$$\Phi \overset{\text{déf.}}{=} \frac{1}{h^2}\begin{pmatrix} 1 & & & \\ & & \vdots & \\ & \cdots & 0 & \cdots \\ & & \vdots & \\ 1 & & & \\ -4 & 1 & & 1 \end{pmatrix}, \qquad (4.6)$$

(il faut se rappeler que les fonctions considérées sont N périodiques, donc que les fréquences négatives sont repoussées à l'autre bout du tableau). Cependant, il y a au moins deux objections à faire à cette écriture.

- Les filtres opèrent sur un signal périodique, or ce n'est pas le cas ici : les valeurs des bords n'ont aucune raison de se « recoller ». De plus, l'équation (4.5) n'est valide qu'à l'intérieur du domaine, c'est-à-dire pour $0<i,j<N$. Elle ne décrit pas une convolution circulaire.
- Les valeurs du bord sont prises en compte dans le filtrage, donc font partie des inconnues : on veut au contraire qu'elles soient fixes.

Pour contourner ce problème, il suffit de rendre nulle U aux extrémités, c'est-à-dire pour $i,j=0,N$, tout simplement en faisant passer dans le membre de droite les termes de bord. Voici par exemple une équation obtenue pour $i=1$ et $1<j<N-1$:

$$\frac{1}{h^2}\{0+U(2,j)+U(1,j+1)+U(1,j-1)-4U(1,j)\}=F(i,j)-\frac{1}{h^2}u_{0y}(jh) \overset{\text{déf.}}{=} \widetilde{F}(i,j),$$

(on a utilisé le fait que $U(0,j)=u_{0y}(jh)$ la valeur fixée sur le bord $x=0$). En faisant de même pour les quatre bords de la matrice F, on crée une nouvelle matrice $\widetilde{F}$ nulle sur les bords, et grâce à cette manipulation, on peut remplacer l'inconnue U par une inconnue $\widetilde{U}$ qui est nulle sur les bords. Il reste justement à régler le problème de l'application du filtre sur les bords (l'équation (4.5)) n'est valable qu'à l'intérieur). Pour pouvoir le faire, il suffit de prolonger les fonctions considérées par imparité selon les deux axes. En effet, la nullité de la fonction sur les bords ainsi que l'équation (4.5) du filtre montre que les termes symétriques par rapport à un bord doivent être de signes opposés. Par exemple, on a l'équation

$$U(1,j)+U(-1,j)=\widetilde{F}(0,j)=0,$$

d'où $U(1,j) = -U(-1,j)$. On étend donc les matrices $\widetilde{U}$ et $\widetilde{F}$ pour obtenir des matrices (toujours notées de la même manière) de taille $2N$ impaires. Ainsi, dans le cas (simpliste, mais instructif) où $N = 3$, la matrice $\widetilde{F}$ va s'écrire

$$\widetilde{F} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \widetilde{F}(1,1) & \widetilde{F}(2,1) & 0 & -\widetilde{F}(2,1) & -\widetilde{F}(1,1) \\ 0 & \widetilde{F}(1,2) & \widetilde{F}(2,2) & 0 & -\widetilde{F}(2,2) & -\widetilde{F}(1,2) \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\widetilde{F}(1,2) & -\widetilde{F}(2,2) & 0 & \widetilde{F}(2,2) & \widetilde{F}(1,2) \\ 0 & -\widetilde{F}(1,1) & -\widetilde{F}(2,1) & 0 & \widetilde{F}(2,1) & \widetilde{F}(1,1) \end{pmatrix}.$$

Avec toutes ces nouvelles conventions, l'équation (4.5), qui s'étend à $0 \leqslant i, j \leqslant 2N$ et est aussi vraie en tant que convolution périodique, s'écrit

$$\widehat{U} * \widetilde{\Phi} = \widetilde{F},$$

où $\widetilde{\Phi}$ est encore définie comme à l'équation (4.6), mais est cette fois de taille $2N$.

L'idée est ensuite de prendre la transformée de Fourier 2D des deux membres, puisque cette dernière transforme le produit de convolution en produit simple, comme énoncé à la proposition 3, chap. III. On obtient l'équation très agréable

$$\mathscr{F}\left(\widetilde{U}\right) \cdot \mathscr{F}\left(\widetilde{\Phi}\right) = \mathscr{F}\left(\widetilde{F}\right),$$

où l'on a noté $\cdot$ le produit terme à terme des deux matrices. Cette équation est maintenant très simple à résoudre, puisque l'on sait calculer la transformée de $\widetilde{\Phi}$, comme le montre le lemme suivant.

Lemme 4.3. *La transformée de Fourier de la fonction de transfert Φ est donnée par*

$$\mathscr{F}\left(\widetilde{\Phi}\right) = -\frac{4}{h^2}\left\{\sin\left(\frac{i\pi}{2N}\right)^2 + \sin\left(\frac{j\pi}{2N}\right)^2\right\}_{0 \leqslant i,\, j \leqslant 2N-1}.$$

Démonstration. Il suffit d'appliquer la définition et de faire un regroupement de termes :

$$\begin{aligned} \mathscr{F}\left(\widetilde{\Phi}\right)[i,j] &= \sum_{k,l} \Phi[k,l] e^{\frac{2i\pi}{2N}ki} e^{\frac{2i\pi}{2N}lj} \\ &= \frac{1}{h^2}\left\{e^{\frac{2i\pi}{2N}i} + e^{\frac{2i\pi}{2N}j} + e^{-\frac{2i\pi}{2N}i} + e^{-\frac{2i\pi}{2N}j} - 4\right\} \\ &= -\frac{4}{h^2}\left\{\sin\left(\frac{i\pi}{2N}\right)^2 + \sin\left(\frac{j\pi}{2N}\right)^2\right\}. \end{aligned}$$

□

On obtient donc

$$G(i,j) = \mathscr{F}\left(\widetilde{U}\right)(i,j) = \frac{\mathscr{F}(\widetilde{F})(i,j)}{\sin^2\left(\frac{i\pi}{2N}\right) + \sin^2\left(\frac{j\pi}{2N}\right)},$$

(il faut faire attention à la division 0/0 pour $i = j = 0$, mais on sait que le résultat est 0). On termine grâce au calcul de la transformée inverse, $\widetilde{U} = \mathscr{F}^{-1}(G)$. Il ne reste plus qu'à

extraire la partie de la matrice $\widetilde{U}$ qui nous intéresse (c'est-à-dire $0 \leqslant i, j \leqslant N$), et à ajouter les termes de bord que l'on avait retranchés au début.

L'intégralité de cette méthode pour résoudre l'équation de Poisson est reprise de façon algorithmique en MATLAB au paragraphe 5, annexe A. Pour pouvoir observer la qualité de l'approximation, on a choisi une équation dont on connaît la solution (en fait, on choisit d'abord la solution, puis on calcule le membre de droite), et on crée les fonctions de bord en conséquence. La figure 4.10 montre la résolution d'une équation de Poisson dont on connaît explicitement une solution, à savoir $u(x,y) = e^{xy}$. La figure 4.11 reprend la même équation, mais modifie les conditions sur deux des bords.

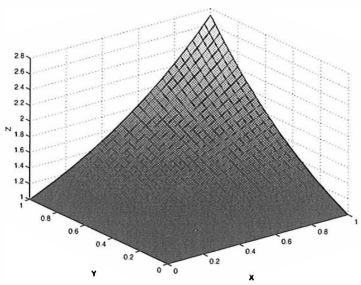

FIG. 4.10 – *Solution de l'équation de Poisson*

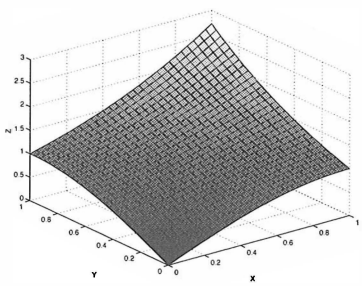

FIG. 4.11 – *Solution modifiée*

***Remarque* 4.4. (Cas d'une fonctionnelle quadratique).** Si l'on considère une solution exacte quadratique, par exemple la fonction $u(x,y) = x^2 + y^2$, qui satisfait l'équation de Poisson :

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 4,$$

on remarque que l'erreur obtenue lors de la résolution par une méthode de différences finies est quasi-nulle. Ceci s'explique par le fait que l'approximation du laplacien par l'équation (4.5) est en fait exacte pour un polynôme de degré inférieur à deux (c'est une formule d'interpolation d'ordre deux).

L'exercice IV.5 reprend cette méthode de résolution en expliquant son fonctionnement par des décompositions matricielles.

5 Calculs de produits

> Our interest in multiplying large numbers is part theoretical, however, because it is interesting to explore the ultimate limits of computational complexity.
>
> D. E. KNUTH [39] (1997)

Contrairement à ce que l'on pourrait penser après toutes ces application dédiées au calcul numérique, l'algorithme FFT possède des applications de nature nettement plus algébriques. C'est principalement la propriété de convolution qui est utilisée. Cet algorithme a permis des avancées significatives pour certains calculs arithmétiques, par exemple pour les calculs intensifs de grands nombres (la recherche des décimales de π en est le meilleur exemple). On pourra lire à ce sujet [5] qui explique quelques anecdotes instructives.

5.1 Présentation théorique

Avant de décrire des méthodes utilisant l'algorithme FFT présenté au paragraphe 2.1, chap. III, pour calculer des produits, on peut expliquer de manière un peu théorique la démarche que nous allons suivre.

Considérons, pour $(\xi_0,\dots,\xi_{N-1}) \in \mathbb{C}^N$, le morphisme d'évaluation

$$\Phi : \begin{cases} \mathbb{C}[X] & \longrightarrow & \mathbb{C}^N \\ P & \longmapsto & (P(\xi_0),\dots,P(\xi_{N-1})) \end{cases}.$$

Le noyau de ce morphisme est l'idéal de $\mathbb{C}[X]$ engendré par le polynôme $\prod_{k=0}^{N-1}(X-\xi_k)$. Dans le cas où les points ξ_k sont distincts, on obtient, par passage au quotient, un isomorphisme linéaire (puisque les deux espaces ont même dimension N)

$$\widetilde{\Phi} : \begin{cases} \mathbb{C}[X]/\prod_{k=0}^{N-1}(X-\xi_k) & \xrightarrow{\simeq} & \mathbb{C}^N \\ \overline{P} & \longmapsto & (\overline{P}(\xi_0),\dots,\overline{P}(\xi_{N-1})) \end{cases}.$$

C'est même à l'évidence un isomorphisme d'algèbre, si on munit $\mathbb{C}[X]$ du produit des polynômes, et $\mathbb{C}^N$ du produit composante par composante.

Grâce à ce morphisme d'évaluation, et son morphisme inverse (interpolation), on peut tracer le diagramme suivant :

$$\begin{array}{ccc} (P_0,\dots,P_{N-1}),(Q_0,\dots,Q_{N-1}) & \xrightarrow[O(N^2)]{mult.polynômes} & (R_0,\dots,R_{N-1}) \\ \Big\downarrow \text{évaluation} & & \text{interpolation} \Big\uparrow \\ \begin{array}{l}(P(\xi_0),\dots,P(\xi_{N-1})),\\ (Q(\xi_0),\dots,Q(\xi_{N-1}))\end{array} & \xrightarrow[O(N)]{mult.ponctuelle} & (R(\xi_0),\dots,R(\xi_{N-1})) \end{array},$$

qui nous suggère une nouvelle façon de multiplier deux polynômes, en passant « par en bas », à savoir en utilisant l'application

$$\Psi : \begin{cases} \mathbb{C}[X]\times\mathbb{C}[X] & \longrightarrow & \mathbb{C}[X]/\prod(X-\xi_k) \\ (P,Q) & \longmapsto & \Phi^{-1}(\Phi(\overline{P})\cdot\Phi(\overline{Q})) \end{cases},$$

où l'on a noté $\overline{P}$ la classe de P dans l'algèbre $\mathbb{C}[X]/\prod(X-\xi_k)$. D'après ce que nous venons de dire, ce morphisme calcule donc le produit des deux polynômes modulo le polynôme $\prod(X-\xi_k)$. Les questions naturelles sont donc les suivantes.

- Quel choix faire pour les points $\{\xi_0,\dots,\xi_{N-1}\}$ afin que cette nouvelle manière de calculer un produit soit rapide ?
- Comment faire pour récupérer vraiment le produit des deux polynômes, et pas seulement le produit modulo un certain polynôme ?

En réalité, nous avons déjà répondu à ces deux questions au chapitre III. Il suffit de relier la TFD à l'évaluation de polynômes pour réinvestir les algorithmes déjà construits.

***Remarque* 5.1.** L'isomorphisme $\widetilde{\Phi}$ est en fait l'isomorphisme canonique donné par le théorème chinois :

$$\mathbb{C}[X]/\prod_{k=0}^{N-1}(X-\xi_k) \xrightarrow{\simeq} \prod_{k=0}^{N-1}\mathbb{C}[X]/(X-\xi_k) \simeq \mathbb{C}^N.$$

En effet, les $X-\xi_k$ sont premiers entre eux (car les ξ_k sont distincts), donc l'application du théorème est licite, et la réduction modulo $X-\xi_k$ envoie un polynôme P sur $P(\xi_k)$.

5.2 Multiplication de polynômes modulo $X^N - 1$

Le but de ce paragraphe est de présenter la transformée de Fourier comme une transformation (en fait un morphisme) sur les polynômes de degrés fixés. Nous pourrons alors utiliser les propriétés algébriques de la transformée de Fourier discrète ainsi que l'algorithme FFT pour effectuer des opérations sur les polynômes de façon très rapide. Derrière ces procédés particulièrement efficaces se cache un problème plus subtil que la simple utilisation de la transformée de Fourier, puisqu'il s'agit de la question de la représentation des polynômes. En effet, la transformée de Fourier permet de jongler entre deux types de représentations, et ainsi d'exploiter les points forts de chacune.

La finalité de cette approche étant de réinvestir les algorithmes déjà présentés au chapitre III (principalement FFT et convolution rapide), nous nous empressons de choisir judicieusement les points d'évaluation/interpolation ξ_k, en l'occurrence, les N racines $N^{\text{ièmes}}$ de l'unité, c'est-à-dire $\xi_k = \omega_N^{-k} = e^{-\frac{2ik\pi}{N}}$ pour $k = 0, \ldots, N-1$. On constate alors que le morphisme $\widetilde{\Phi}$ est en fait la transformée de Fourier discrète. Plus précisément on peut le réécrire de la manière suivante :

$$\widetilde{\Phi} : \begin{cases} \mathbb{C}[X]/(X^N - 1) & \xrightarrow{\simeq} & \mathbb{C}^N \\ \overline{P} & \mapsto & \widetilde{\Phi}(P) = \mathscr{F}(P_0, \ldots, P_{N-1}) \end{cases},$$

où l'on a noté $P_0, \ldots, P_{N-1}$ les coefficients du polynôme $\overline{P}$ (on a choisi le représentant de degré inférieur à N). Bien sûr, on a utilisé ici l'identité $\prod_{k=0}^{N-1}(X - \xi_k) = X^N - 1$. On s'aperçoit donc que le calcul de la transformée de Fourier discrète de P (en tant que vecteur de $\mathbb{C}^N$) n'est rien d'autre que le calcul des valeurs que prend P en les N racines $N^{\text{ièmes}}$ de l'unité, et on retrouve exactement le morphisme Φ du paragraphe précédent.

La transformée de Fourier discrète permet ainsi de jongler entre deux représentations des polynômes de degré $N-1$ (en fait modulo $X^N - 1$) :

- *la représentation par coefficients* : ceci revient à considérer un polynôme comme un vecteur de $\mathbb{C}^N$. Bien que très couramment utilisée, cette représentation a un point faible : elle n'est pas du tout appropriée pour calculer le produit de deux polynômes. Alors que la somme de deux polynômes P et Q de degré au plus N se calcule en N opérations (comme le montre l'égalité $(P+Q)_k = P_k + Q_k$), le produit, calculé de façon naïve, nécessite N^2 opérations.
- *la représentation par valeurs* : on se donne le polynôme par les valeurs qu'il prend en N points distincts (ici pris de façon bien particulière, les racines $N^{\text{ièmes}}$ de l'unité). Cette représentation est bien plus adaptée au calcul du produit, puisqu'il suffit de faire le produit des valeurs de chaque polynôme.

Ceci est à prendre au pied de la lettre : l'algorithme FFT fournit un moyen rapide et simple de passer de l'une à l'autre des représentations.

Pour terminer, rappelons donc l'équation obtenue pour le calcul du produit de deux polynômes modulo $X^N - 1$:

$$P * Q = \mathscr{F}^{-1}(\mathscr{F}(P) \cdot \mathscr{F}(Q)). \tag{5.1}$$

Dans cette équation, on a confondu vecteur et polynômes modulo $X^N - 1$, et le produit $P * Q$ peut aussi être vu comme un produit de convolution circulaire entre deux vecteurs. On peut représenter graphiquement l'opération effectuée. Les deux premiers graphiques de la figure 4.12 montrent les polynômes définis par $P = 1 + 2X + X^3 - X^4 + X^5$ ainsi que

$Q = X - X^2 + 2X^3 + 2X^5$. En abscisse, on a mis les degrés des monômes $1, \ldots, X^{10}$. On choisit donc de travailler dans $\mathbb{Z}/11\mathbb{Z}$. Ceci est judicieux, car le degré du produit $P * Q$ est justement 10. Ainsi, lorsque l'on représente dans le graphique de droite le produit des deux vecteurs, on trouve bien la représentation graphique du produit $P * Q$, puisque la réduction modulo $X^{11} - 1$ n'a pas eu d'effet.

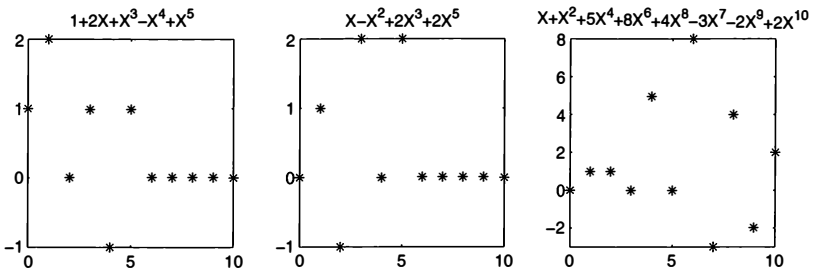

FIG. 4.12 – *Représentation graphique du produit de polynômes par convolution*

***Remarque* 5.2. (Interpolation de Lagrange).** Des algorithmes permettent de calculer les polynômes d'interpolation dans le cas où les points ξ_k ne sont pas nécessairement des racines de l'unité. Il y a bien sûr le résultat de *Lagrange*, qui donne une base explicite de $\mathbb{C}_{N-1}[X]$ (espace des polynômes de degré au plus $N-1$) dans lequel le calcul s'effectue simplement. En effet, si on cherche le polynôme P qui prend les valeurs y_i aux points x_i, pour $i = 0, \ldots, N-1$, alors $P = \sum_{i=0}^{N-1} y_i P_i$, où P_i est le $i^{\text{ème}}$ *polynôme de Lagrange* associé aux points $\{x_i\}_{i=0}^{N-1}$:

$$P_i \overset{\text{déf.}}{=} \frac{\prod_{j=0}^{N-1}(X - x_j)}{\prod_{j \neq i}(x_i - x_j)}.$$

Cependant, le calcul numérique du polynôme d'interpolation dans la base des $\{P_i\}_{i=0}^{N}$ n'est pas utilisé dans la pratique, car il conduit à une accumulation des erreurs numériques. On préfère utiliser la technique des *différences divisées*, qui est expliquée par exemple dans [23].

5.3 Multiplication de polynômes

La difficulté à laquelle on est confronté lors du calcul du produit de deux polynômes P et Q (de degré $N-1$) par la méthode présentée plus haut est qu'a priori, le produit PQ est un polynôme de degré $2N-2$. Ce polynôme sera donc réduit modulo $X^N - 1$, ce qui a souvent un effet plus qu'indésirable ... De façon plus précise, les coefficients du produit sont donnés par l'équation

$$\forall n \in \{0, \ldots, 2N-1\}, \quad (PQ)_n = \sum_{k=0}^{n} P_k Q_{n-k}. \tag{5.2}$$

Il s'agit en fait d'un produit de convolution acyclique (à comparer au produit cyclique de l'équation (5.1)), et nous allons pouvoir utiliser la technique déjà présentée à la section 3.3, chap. III, pour le calculer.

Cette démarche (ajout de zéros à la fin du vecteur pour rendre le produit cyclique) est très intuitive, puisqu'elle consiste à considérer les deux polynômes comme des polynômes de degré $2N-1$ (en ajoutant des coefficients nuls). On peut alors appliquer l'approche présentée au paragraphe précédent (c'est-à-dire employer un produit de convolution cyclique, où si on préfère, calculer le produit modulo $X^{2N}-1$). Bien heureusement, ceci ne change rien au résultat, puisque le polynôme PQ n'est pas affecté par la réduction modulo $X^{2N}-1$. D'une façon plus théorique, ceci revient à utiliser la bijectivité de l'application

$$\begin{cases} \mathbb{C}_{2N-1}[X] & \longrightarrow & \mathbb{C}[X]/(X^{2N}-1) \\ P & \longmapsto & P \mod X^{2N}-1 \end{cases},$$

où l'on a noté $\mathbb{C}_{2N-1}[X]$ l'espace des polynômes de degré inférieur ou égal à $2N-1$.

5.4 Multiplication de grands entiers

On note $(a_0,\ldots,a_{N-1})$ la représentation en base b d'un grand entier a, c'est-à-dire

$$a = a_0 + a_0 b + \cdots + a_{N-1} b^{N-1}.$$

On remarque que la multiplication de deux entiers a et a' s'apparente au calcul d'un produit de polynômes, à une exception près : les entiers a_k et a'_k, pour $k=0,\ldots,N-1$, doivent appartenir à l'ensemble $\{0,\ldots,b-1\}$. Si on veut calculer rapidement la multiplication de deux grands entiers, on pourra donc utiliser la technique de produit de polynômes que nous venons d'exposer au paragraphe précédent, suivie d'une phase de « propagation des retenues ». Des programmes MATLAB permettant de réaliser tout ceci sont rassemblés au paragraphe 4, annexe A.

Cette approche souffre néanmoins de quelques points faibles, principalement liés à l'utilisation de calculs en virgule flottante (pour l'algorithme FFT classique), qui sont soumis à des erreurs d'arrondi (alors que les calculs entiers sont à la fois plus rapides et exempts d'erreurs). Ce problème sera résolu au chapitre VI, chap. VI, grâce à l'introduction de la transformée de Fourier sur les corps finis et les anneaux.

6 Exercices

Exercice IV.1 (Transformée à support compact). Il s'agit de démontrer la proposition 1.3. Soit f une fonction telle que $\mathrm{Supp}(\widehat{f}) \subset [-A,A]$. Expliquer pourquoi f est de classe $\mathscr{C}^\infty$, et calculer ses dérivées successives $f^{(n)}(t_0)$, pour $t_0 \in \mathbb{R}$, sous forme d'intégrale entre $-A$ et A. En utilisant un développement limité de $t \mapsto e^{it}$ en t_0, en déduire un développement de f. En déduire que $f \neq 0$ ne peut s'annuler sur tout un intervalle non vide.

Exercice IV.2 (Résolution de l'équation de la chaleur par différences finies). On souhaite résoudre de manière approchée l'équation de la chaleur sur le cercle, dont on rappelle la formulation, pour $\kappa = 1$:

$$\begin{cases} \forall (t,x) \in \mathbb{R}_*^+ \times S^1, & \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \\ \forall x \in S^1, & u(0,x) = f(x) \end{cases}, \tag{6.1}$$

où la solution cherchée u est supposée suffisamment régulière sur $\mathbb{R}_*^+ \times S^1$, et continue sur $\mathbb{R}^+ \times S^1$. Pour se faire, on considère une discrétisation de pas $d = \frac{1}{N}$ en espace, et de

pas h en temps. Ceci nous amène à considérer les vecteurs $u^n \in \mathbb{R}^N$, pour $n \geqslant 0$, censés approcher la fonction u :

$$\forall n \geqslant 0, \forall k \in \{0,\dots,N-1\}, \quad u^n[k] \approx u(nh,kd).$$

1. Montrer que l'on peut, suite à une discrétisation de l'équation (6.1), considérer l'équation aux différences

$$u^{n+1}[k] - u^n[k] = s\left(u^n[k+1] + u^n[k-1] - 2u^n[k]\right),$$

pour $n \geqslant 0$ et $k = 1,\dots,N-1$. On a noté $s \overset{\text{déf.}}{=} \frac{h}{d^2}$, et par convention, on définit $u^n[-1] \overset{\text{déf.}}{=} u^n[N-1]$ et $u^n[N] \overset{\text{déf.}}{=} u^n[0]$.

2. Montrer que ce schéma explicite peut s'écrire sous la forme d'une convolution. Est-il avantageux de calculer u^n par convolutions itérées en utilisant l'algorithme FFT ? Proposer une implémentation MATLAB de l'algorithme choisi.
3. On dit que le schéma numérique choisi est stable si, pour tout u_0 tel que $\|u_0\|_\infty \leqslant 1$, alors la solution approchée u^n reste bornée quel que soit n. Donner, en utilisant la transformée de Fourier discrète, une condition nécessaire et suffisante pour que le schéma que nous venons de construire soit stable.
4. On souhaite maintenant envisager des schémas non explicites, c'est-à-dire tels que u^{n+1} ne soit pas donné directement en fonction de u^n. On considère le schéma

$$u^{n+1} - u^n = A * \left(\theta u^{n+1} + (1-\theta)u^n\right),$$

où θ est un paramètre variant dans $[0,1]$, et A est le vecteur tel que $A[0] = -2s$, $A[1] = A[-1] = s$, et dont toutes les autres entrées sont nulles. En particulier, on remarque que pour $\theta = 0$ on retrouve le schéma explicite déjà construit, et que pour $\theta = 1$, on obtient un schéma implicite. Expliquer comment on peut résoudre cette équation en utilisant la transformée de Fourier. Etudier ensuite le problème de la stabilité du schéma obtenu.

5. Reprendre les questions précédentes dans le cas de l'équation de la chaleur en dimension 2, c'est-à-dire sur $\mathbb{R}^+ \times S^1 \times S^1$. En particulier, on proposera une implémentation des algorithmes implicites utilisant des calculs de FFT bidimensionnelles.

La figure 4.13 montre la résolution de l'équation de la chaleur 2D par différents schémas. Horizontalement, chaque image représente un pas de l'algorithme. La première ligne correspond à $\theta = 0$, et un pas $h_1 = 4 \times 10^{-5}$: le schéma est complètement instable (il faut prendre h de l'ordre de 10^{-6} pour que le schéma soit stable). La deuxième ligne correspond à $\theta = 0.5$ et $h_2 = 4 \times 10^{-4} = 10h_1$: on commence à appercevoir des instabilités. La dernière ligne correspond à $\theta = 1$ et $h_3 = 4 \times 10^{-3} = 100h_1$: même avec un pas temporel aussi grand, le schéma est complètement stable, l'image devenant totalement floue.

Exercice IV.3 (Unicité pour l'équation de la chaleur). Cette démonstration est tirée du livre de DYM et MCKEAN [29]. On considère l'équation de la chaleur sur le cercle (4.2). On souhaite montrer que sous l'hypothèse f continue, l'équation (4.3) définit bien une solution de l'équation, et que c'est en fait la seule.

1. Montrer que pour $t > 0$, la solution u peut s'écrire sous la forme d'une convolution :

$$u(t,x) = p_t * f(x) \quad \text{avec} \quad p_t(x) \overset{\text{déf.}}{=} \sum_{n\in\mathbb{Z}} e^{-2\pi^2 n^2 t} e_n(x).$$

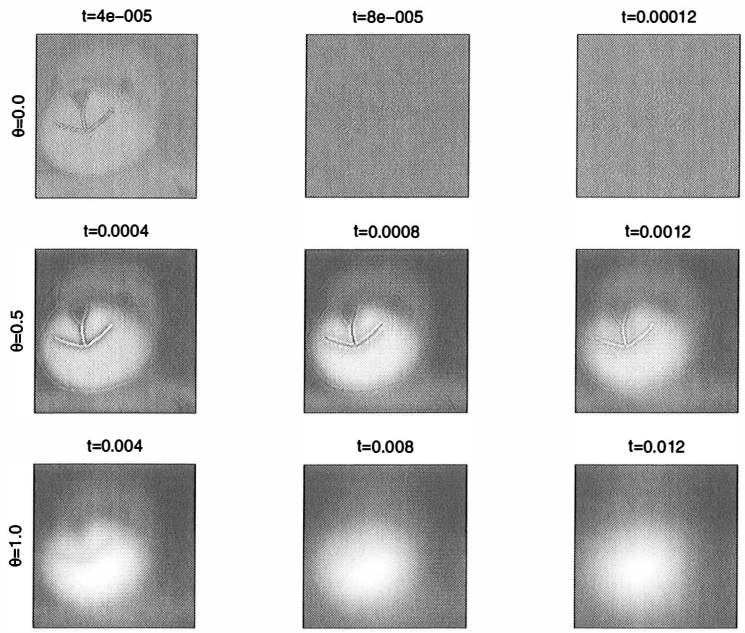

FIG. 4.13 – *Résolution de l'équation de la chaleur par différences finies*

2. Dans le cas où $f \in \mathscr{C}^2(S^1)$, montrer que l'on a bien $\|u(t,\cdot) - f\|_\infty \xrightarrow[t\to 0]{} 0$.
3. On désire montrer que si $f \geqslant 0$, alors $u \geqslant 0$. On considère la fonction v définie $v(t,x) = e^{\beta t}u(t,x)$, pour un certain paramètre β. Montrer que si on suppose que $u(t_0,x_0) < 0$, la fonction v atteint son minimum $\alpha < 0$ sur $[0,t_0] \times S^1$ pour un certain temps $t_1 > 0$ et une position x_1. Montrer alors que l'on a

$$0 \geqslant \frac{\partial v}{\partial t}(t_1,x_1) = \alpha\beta + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(t_1,x_1) \geqslant \alpha\beta$$

   En déduire une contradiction en prenant $\beta < 0$.
4. En utilisant le produit de convolution qui définit u, en déduire que p_t est positif. En déduire le *principe du maximum* pour l'équation de la chaleur :

$$\forall t > 0, \quad \|u(t,\cdot)\|_\infty \leqslant \|f\|_\infty.$$

   Montrer que ceci assure l'unicité de la solution de l'équation de la chaleur.
5. En utilisant une suite de fonction $f_n \in \mathscr{C}^2(S^1)$ qui converge uniformément vers f, en déduire que dans le cas où f est simplement continue, on a quand même la convergence $\|u(t,\cdot) - f\|_\infty \xrightarrow[t\to 0]{} 0$.

Exercice IV.4 (Potentiel électrique 3D). On souhaite généraliser l'algorithme de résolution de l'équation de Poisson décrit au paragraphe 4.3 pour des problèmes tridimensionnels. Par exemple on souhaite déterminer un potentiel électrique u qui satisfasse l'équation de Poisson :

avec des conditions aux bords ainsi qu'une fonction f spécifiée par l'utilisateur. Il faudra bien sûr utiliser une transformée de Fourier tridimensionnelle, et penser à rendre impairs les tableaux 3D rencontrés. Pour représenter la solution, on pourra dessiner des surfaces équipotentielles, c'est-à-dire les surfaces d'équation $f(x,y,z) = \lambda$ pour certaines valeurs du paramètre λ.

Exercice IV.5 (Formulation matricielle pour l'équation de Poisson). Cet exercice, qui propose une explication plus calculatoire de l'algorithme de résolution de l'équation de Poisson, est en partie inspiré de l'article de SWARZTRAUBER et SWEET [70]. On reprend les notations du paragraphe 4.3, et on considère notamment une matrice carrée U de taille $N-1$ (les indices variant de 1 à $N-1$) qui est la solution de l'équation aux différences finies (4.5) à l'intérieur du carré $[0,1] \times [0,1]$ (c'est-à-dire sans les termes de bord).

1. Sans prendre en compte les termes de bord, montrer que l'on peut écrire l'équation aux différences sous la forme
$$T_{N-1}U + UT_{N-1} = F,$$
où T_{N-1} est la matrice de taille $N-1$ avec $-2/h^2$ sur la diagonale et $1/h^2$ sur la sous-diagonale et la sur-diagonale.
2. Puisque la valeur de la solution sur les bords est supposée connue, utiliser la même approche que celle employée au paragraphe 4.3 pour obtenir une équation modifiée du type
$$T_{N-1}\widetilde{U} + \widetilde{U}T_{N-1} = \widetilde{F}. \tag{6.2}$$
3. Montrer que les vecteurs propres de T_{N-1} sont les
$$V_j \overset{\text{déf.}}{=} \left\{ \sin\left(\frac{ij\pi}{N}\right) \right\}_{i=1}^{N-1}.$$
Déterminer les valeurs propres associées. On note V la matrice de changement de base, ce qui signifie que ses colonnes sont les V_j, et D la matrice diagonale dont les entrées sont les valeurs propres calculées. En notant $U_0 \overset{\text{déf.}}{=} V^{-1}\widetilde{U}V$ et $\widetilde{F}_0 = V^{-1}\widetilde{F}V$, en déduire que U_0 vérifie l'équation
$$DU_0 + U_0D = \widetilde{F}_0.$$
Résoudre cette équation.
4. Montrer alors que la matrice V est en fait orthogonale, et que le calcul de Vx, où x est un vecteur de taille $N-1$, est équivalent à un calcul de *transformée en sinus* (c'est-à-dire la partie imaginaire d'une certaine transformée de Fourier discrète). En déduire que le calcul de $\widetilde{U} = VU_0V^{-1}$ est en fait équivalent au calcul d'une transformée en sinus bidimensionnelle (c'est-à-dire une transformée sur les lignes suivie d'une transformée sur les colonnes d'une matrice).
5. Expliquer comment on peut calculer des transformées en sinus (unidimensionnelles puis en 2D) grâce à une TFD de taille double. Comment calculer la transformée inverse ? Enfin, faites le rapprochement entre la démarche matricielle proposée dans cet exercice et le calcul de convolution qui soutenait l'algorithme proposé au paragraphe 4.3.

Exercice IV.6 (Lissage d'image). Ecrire un programme qui permet de lisser une image 2D en niveau de gris, comme le montre la figure 4.5. On pourra utiliser une fonction de transfert gaussienne, et adapter les paramètres à la taille de l'image.

Exercice IV.7 (Corrélation et détection d'image). Cet exercice est inspiré d'un article de LEWIS [47], qui remet au goût du jour l'utilisation de la corrélation pour la détection d'images. Soit $f \in \mathbb{R}^{N\times N}$ une image de taille N, et $g \in \mathbb{R}^{P\times P}$ une autre image, dont la taille est typiquement beaucoup plus petite que celle de f. La question est de déterminer si l'image g est une sous-image de f, et si c'est le cas, de repérer son emplacement. A cet effet, on définit la distance entre f et g

$$\forall (u,v) \in \{0,\dots,N-1\}^2,\ d(f,g)[u,v]^2 \overset{\text{déf.}}{=} \sum_{(x,y)\in D(u,v)} (f(x,y)-g(x-u,y-v))^2,$$

où $D(u,v)$ désigne le sous-ensemble de $\{0,\dots,N-1\}^2$ formé des couples (x,y) tels que $(x-u,y-v) \in \{0,\dots,P-1\}^2$.

1. Quelle est la signification intuitive de $d(f,g)$? En quelle circonstance $d(f,g)$ est-elle voisine de zéro? Dans le cas où la quantité

$$P_{u,v}(f) = \sum_{(x,y)\in D(u,v)} f(x,y)^2$$

est presque constante, montrer que la recherche des points où $d(f,g)$ est petite revient à maximiser la *corrélation* entre f et g

$$\mathrm{Corr}(f,g)[u,v] \overset{\text{déf.}}{=} \sum_{(x,y)\in D(u,v)} f(x,y)\, g(x-u,y-v).$$

2. Montrer que $\mathrm{Corr}(f,g)$ peut s'écrire comme un produit de convolution acyclique. En déduire que l'on peut calculer cette corrélation de façon rapide en utilisant l'algorithme FFT.
3. On souhaite corriger le défaut que l'on a introduit en supposant que $P_{u,v}(f)$ est presque constante. On note $\widetilde{f}_{u,v}$ la moyenne de f sur $D(u,v)$, et $\widetilde{g}$ la moyenne de g. On définit alors la *corrélation normalisée*

$$\overline{\mathrm{Corr}}(f,g)[u,v] \overset{\text{déf.}}{=} \frac{\sum_{(x,y)} (f(x,y)-\widetilde{f}_{u,v})(g(x-u,y-v)-\widetilde{g})}{\left\{\sum_{(x,y)} (f(x,y)-\widetilde{f}_{u,v})^2 \sum_{(x,y)} (g(x,y)-\widetilde{g})^2\right\}^{1/2}},$$

où les sommes portent sur $(x,y) \in D(u,v)$. Expliquer en quoi cette quantité apporte bien une correction. Dispose-t-on toujours d'un algorithme de calcul rapide par FFT?
4. Montrer que le numérateur de $\overline{\mathrm{Corr}}(f,g)$ s'écrit comme une convolution. On note, pour $k=1,2$, les « sommes glissantes »

$$\forall (u,v) \in \{0,\dots,N-1\}^2, \quad s_k(u,v) \overset{\text{déf.}}{=} \sum_{(x,y)\in D(u,v)} f[x,y]^k,$$

avec par convention $s_k(u,v)=0$ pour $u \geqslant N$ ou $v \geqslant N$. Montrer que s_k vérifie l'équation de récurrence

$$\begin{aligned} s_k(u,v) =& s_k(u+1,v)+s_k(u,v+1)-s_k(u+1,v+1) \\ &+ f(u,v)+f(u+P,v+P)-f(u,v+P)-f(u+P,v). \end{aligned}$$

En déduire un algorithme de calcul rapide de s_k (évaluer sa complexité), puis de $\overline{\mathrm{Corr}}(f,g)$.

La figure 4.14 présente un exemple d'application de cette méthode. On voit bien que la corrélation normalisée (image (d)) présente un maximum beaucoup plus franc que la corrélation non normalisée (image (c)). On pourra noter que [47] propose un algorithme de calcul rapide qui a été employé entre autres pour effectuer les recalages dans le film *Forest Gump* (1994).

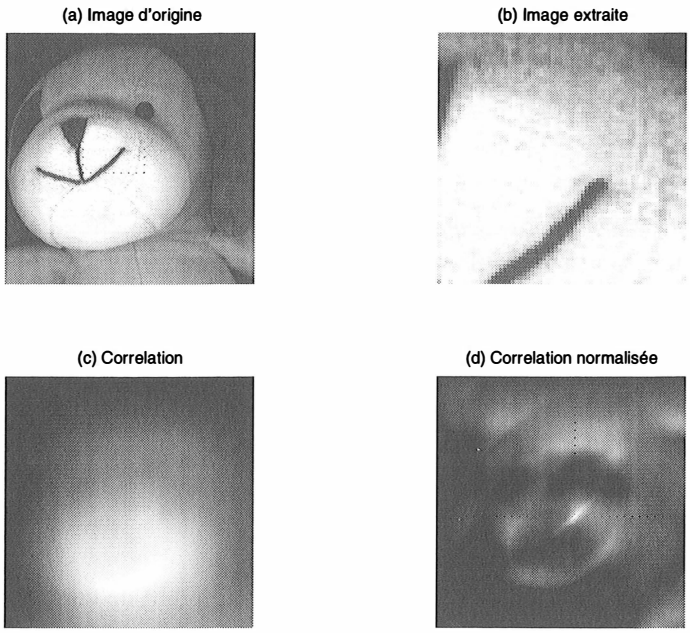

FIG. 4.14 – *Corrélation entre deux images*

Exercice IV.8 (Rotation par FFT). Soit $f \in \mathbb{C}^{N\times N}$ un signal bidimensionnel. On définit, pour $v = (v_1, v_2) \in \mathbb{R}^2$ et $\lambda \in \mathbb{R}$,

$$T_v(f)[k,l] = f[k-v_1, l-v_2], \quad S_\lambda^{(x)}(f)[k,l] = f[k-\lambda l, l], \quad S_\lambda^{(y)}(f)[k,l] = f[k, l-\lambda k].$$

1. Exprimer $\mathscr{F}(T_v(f))$ en fonction de $\mathscr{F}(f)$. En déduire un algorithme rapide pour réaliser une translation quelconque d'une image. En translatant chaque ligne (resp. chaque colonne) de f, écrire un algorithme rapide pour calculer $S_\lambda^{(x)}(f)$ (respectivement $S_\lambda^{(y)}(f)$).
2. Montrer qu'une rotation d'angle θ autour de l'origine peut s'écrire sous la forme

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \sin(\theta) \end{pmatrix} = \begin{pmatrix} 1 & \lambda_1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \lambda_2 & 1 \end{pmatrix} \begin{pmatrix} 1 & \lambda_3 \\ 0 & 1 \end{pmatrix}.$$

   Avec la question précédente, en déduire un algorithme rapide pour effectuer une rotation à une image $f \in \mathbb{C}^{N\times N}$ autour de l'origine.
3. Quels sont les avantages et les inconvénients de cet algorithme ? Comment les résoudre ? Comment faire tourner une image autour de son centre ?

La figure 4.15 montre plusieurs rotations d'une image autour de son centre.

FIG. 4.15 – *Rotation d'une image par FFT*

Exercice IV.9 (Filtre passe bas). On souhaite réaliser un filtre de convolution passe bas.

1. Ecrire un programme MATLAB qui construit un vecteur f, de taille N, tel que le filtre Φ^f conserve les $N/2$ basses fréquences, et supprime les $N/2$ hautes fréquences.
2. Représenter avec une grande précision la transformée de Fourier continue de la réponse impulsionnelle f (autrement dit la réponse fréquentielle du filtre). Que constate-t-on ?
3. En considérant une coupure moins brutale dans les fréquences conservées/rejetées, reprendre les questions précédentes et commenter les résultats. En particulier, imaginer une famille de filtres f_ε, $\varepsilon \in [0,1]$, avec une coupure brutale pour $\varepsilon = 0$ et douce pour $\varepsilon = 1$.

La figure 4.16 montre trois filtres différents, avec des coupures de plus en plus douces. On peut aussi voir les transformées discrètes des filtres, et leurs transformées de Fourier continues.

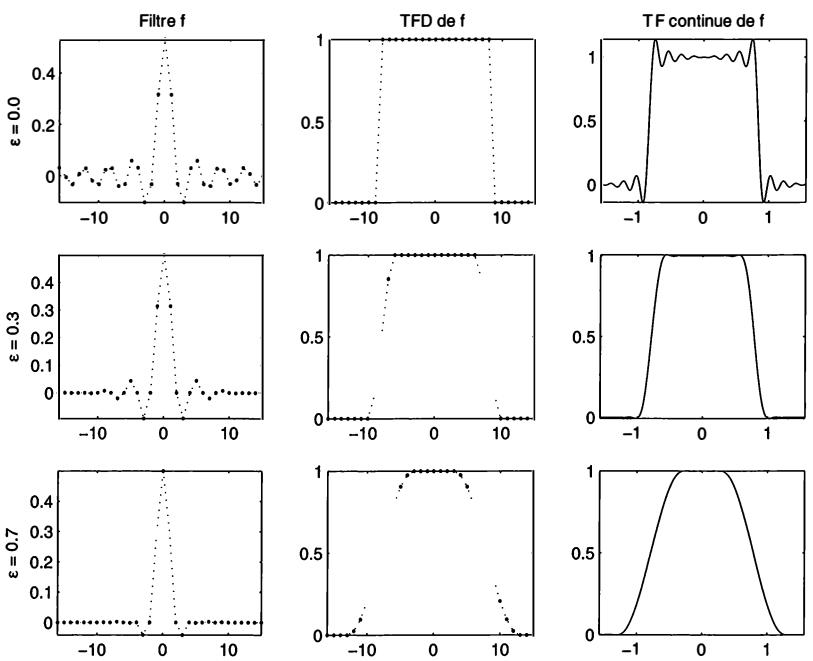

FIG. 4.16 – *Filtres passe bas pour différentes valeurs de* ε

Exercice IV.10 (Itérations entières). On considère l'expérience suivante : on dispose n enfants en cercle, et on leur donne à chacun un nombre pair, arbitraire, de bonbons. Le

jeu consiste à ce que chaque enfant donne à son voisin de droite la moitié de ses bonbons, et à itérer le procédé.

1. On s'arrange pour que les enfants aient à chaque partage un nombre pair de bonbons. Pour cela, une personne extérieure distribue, après chaque itération, un bonbon à chaque enfant ayant un nombre impair de bonbons. Montrer qu'après un nombre fini d'itérations, tous les enfants ont le même nombre de bonbons.
2. Si on autorise des parties fractionnaires de bonbons, montrer comment on peut traduire cette expérience par un calcul de convolution. En déduire qu'après un nombre potentiellement infini d'itérations, tous les enfants ont le même nombre de bonbons.
3. Etudier les deux premières questions pour des règles de partage différentes. Par exemple, que se passe-t-il si chaque enfant donne la moitié de ses bonbons à son voisin de gauche, et l'autre moitié à son voisin de droite ?

Exercice IV.11 (Algorithme de Karatsuba). Nous allons expliciter la construction d'un algorithme récursif de multiplication de polynômes. Il utilise une technique dite *diviser pour régner*, souvent employée en algorithmique, voir par exemple le livre de CORMEN [21] pour d'autres exemples. On considère deux polynômes P et Q de degré n sur un corps K. On note $k \stackrel{\text{déf.}}{=} \lfloor (n+1)/2 \rfloor$.

1. On écrit les polynômes P et Q sous la forme
$$P(X) = P_0(X) + X^k P_1(X) \quad \text{et} \quad Q(X) = Q_0(X) + X^k Q_1(X),$$
où les polynômes P_0, Q_0 sont de degré inférieur à k, et les polynômes P_1, Q_1 sont de degrés inférieurs à k ou $k+1$, selon la parité de n. Montrer que le produit $P(X)Q(X)$ peut se mettre sous la forme
$$P(X)Q(X) = R_0(X) + X^k R_1(X) + X^{2k} R_2(X).$$
Préciser la valeur des polynômes qui interviennent dans cette égalité.
2. Montrer que le polynôme R_1 peut se calculer à l'aide de seulement une multiplication, mais par contre 4 additions.
3. Implémenter un algorithme récursif en utilisant à chaque étape la décomposition que nous venons d'effectuer.
Prouver que la complexité de cet algorithme est $O(n^{\log_2(3)})$. Pour quelles valeurs de n cet algorithme est préférable à l'algorithme utilisant la FFT décrit au paragraphe 5.3 ?

Exercice IV.12 (Spline et filtrage). Cet exercice nécessite quelques connaissances sur les séries de Fourier. Si $\{f[k]\}_{k\in\mathbb{N}}$ est une suite de $\ell^2(\mathbb{Z})$, on définit sa transformée de Fourier par
$$\forall x \in \mathbb{R}, \quad \widehat{f}(x) \stackrel{\text{déf.}}{=} \sum_{k\in\mathbb{Z}} f[k] e^{-\mathrm{i}kx}.$$
C'est une fonction 2π-périodique, que l'on peut assimiler à une fonction de $L^2(\mathbb{R}/2\pi\mathbb{Z})$. Soit $u : \mathbb{R} \to \mathbb{R}$ une fonction continue qui décroît suffisamment vite en $\pm\infty$. On suppose que l'on connaît en fait des valeurs échantillonnées de u, notées $u_d[k] \stackrel{\text{déf.}}{=} u(k)$, pour $k \in \mathbb{Z}$. On souhaite interpoler ces valeurs sous une des deux formes suivantes :
$$v(x) \stackrel{\text{déf.}}{=} \sum_{k\in\mathbb{Z}} u_d[k] \varphi(x-k) \tag{6.3}$$
$$v(x) \stackrel{\text{déf.}}{=} \sum_{k\in\mathbb{Z}} a[k] \psi(x-k), \tag{6.4}$$

les fonctions φ et ψ étant données à l'avance, avec un petit support. On suppose bien sûr que l'interpolation est exacte, c'est-à-dire $\forall k \in \mathbb{Z}$, $v(k) = u(k)$. La suite $a[k]$, $k \in \mathbb{Z}$, est inconnue, et il va falloir la déterminer. Le schéma d'interpolation (6.3) correspond à une interpolation directe, alors que (6.4) correspond à une interpolation indirecte.

1. On note $\psi_d[k] \overset{\text{déf.}}{=} \psi(k)$ la suite échantillonnée de ψ. On suppose que

$$\forall \xi \in \mathbb{R}, \quad \widehat{\psi_d}(\xi) \neq 0.$$

Montrer alors que le problème d'interpolation indirecte admet une solution c unique, donnée par la relation

$$\forall \xi \in \mathbb{R}, \quad \widehat{c}(\xi) = \frac{\widehat{u}(\xi)}{\widehat{\psi_d}(\xi)}.$$

Comment peut-on ramener cette interpolation à une interpolation directe ? Quel problème rencontre-t-on ?

2. On définit la fonction B-spline d'ordre n, notée β^n par

$$\beta^0 = 1_{[-\frac{1}{2},\frac{1}{2}]} \quad \text{et} \quad \forall n > 0, \quad \beta^n = \beta^0 * \beta^{n-1}.$$

Quel est le support de β^n ? Ces fonctions permettent-elles de définir un schéma d'interpolation direct ? Indirect ? La figure 4.17 montre les 4 premières fonctions splines β^n.

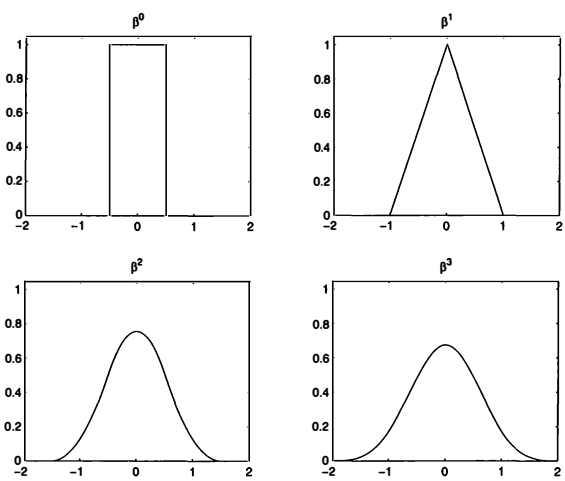

FIG. 4.17 – *Fonctions splines de base*

3. Calculer la valeur de $\widehat{\beta^n}$ (transformée de Fourier continue). On note β_d^n la suite échantillonnée à partir de β^n. Calculer la valeur de $\widehat{\beta_d^n}$ (série de Fourier), et montrer que cette fonction ne s'annule pas (on aura à distinguer selon la parité de n).
4. En déduire une expression de la fonction β_{card}^n qui permet de réaliser une interpolation indirecte à partir des fonctions splines. Quel est son support ? Montrer que l'on a la convergence suivante, dans $L^2(\mathbb{R})$:

$$\beta_{card}^n \xrightarrow[n\to+\infty]{} \text{sinc}, \quad \text{où} \quad \text{sinc}(x) \overset{\text{déf.}}{=} \frac{\sin(\pi x)}{\pi x}.$$

Quand $n \to \infty$, quel type d'interpolation réalise-t-on ? La figure 4.18 montre une comparaison entre les fonctions cardinales correspondant à l'interpolation spline de degré 3 (c'est-à-dire β^3_{card}) et à l'interpolation Shannon (c'est-à-dire sinc). On voit que la fonction spline a beaucoup moins de « rebonds ».

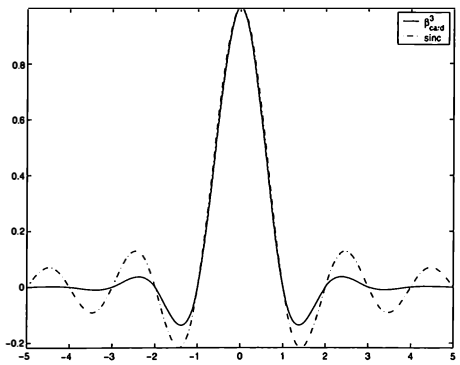

FIG. 4.18 – *Comparaison entre spline et sinus cardinal*

5. En écrivant c comme une convolution, expliquer comment on peut en calculer une valeur approchée à l'aide de l'algorithme FFT. Quel est l'inconvénient de cette méthode ? A l'exercice V.8, nous verrons comment on peut calculer c par des calculs récursifs bien plus efficaces.

Il existe des méthodes plus classiques pour calculer l'interpolation par des splines cubiques. Par exemple, dans [16], CIARLET décompose la fonction cherchée dans une base de polynômes adaptée, et résout un système linéaire tridiagonal. Comparer cette méthode avec celle par filtrage proposée dans cet exercice. Dans la pratique, on considère seulement un nombre fini de valeurs $u_d[k]$, pour $k \in \{0, \ldots, K-1\}$. On peut alors montrer que l'on perd l'unicité de la solution, mais que l'on peut imposer des conditions au bord pour remédier au problème. L'exercice V.8 propose d'étudier ce problème pour les splines cubiques. La figure 4.19 montre l'interpolation par des splines cubiques, avec deux types de conditions aux bords :

– Splines libres : on impose que la dérivée de la fonction interpolante s'annule au bords.
– Splines « not-a-knot » : on n'impose pas de conditions sur les points du bord, mais on impose que la dérivée troisième de v soit continue en 1 et $K-2$.

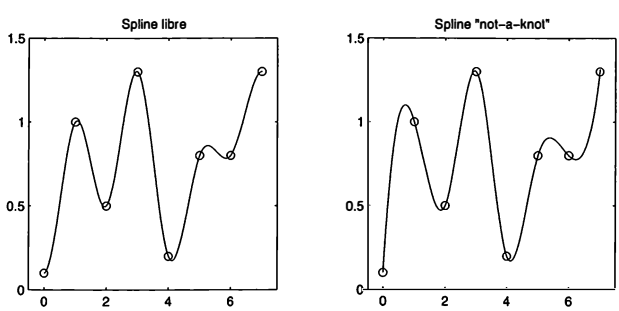

FIG. 4.19 – *Interpolation par des splines cubiques*

Chapitre V
Extension de la notion de transformée de Fourier

> The truth is that the digital computer has totaly defeated the anlog computer. [...] The question is wether the special ideas of Fourier analysis still have a part to play, and the answer is absolutly *yes*.
>
> G. STRANG [68] (1986)

Ce chapitre regroupe de nombreuses notions voisines de la transformée de Fourier, ainsi que des applications directes de ces développements. Nous allons ainsi être amenés à définir de nouvelles transformations, entre autres *la transformée de Hartley*, la *transformée en Z*, et la *transformée de Fourier fractionnaire*. Le plus souvent, il s'agit de trouver des algorithmes pour pallier certaines faiblesses de la TFD (par exemple la *transformée de Hartley*), ou bien d'étendre de façon naturelle certaines notions (la *transformée en Z* par exemple). Nous allons étudier dans quels cas ces transformations sont plus efficaces ou plus adaptées que la transformée de Fourier discrète, et quelles applications peuvent tirer bénéfice des algorithmes rapides obtenus. Les deux points importants qu'il faut garder à l'esprit lorsque l'on travaille avec de telles transformations sont les suivants :

- Elles ne correspondent pas à des calculs approchés. Il s'agit de formules exactes, qui bien souvent possèdent une formule d'inversion. Ces transformées peuvent être utiles pour certains calculs numériques (par exemple le calcul approché d'une transformée de Fourier ou d'une convolution infinie), mais il s'agit avant tout de calculs de nature algébrique.
- Elles disposent d'algorithmes de calcul rapides. Ce sont ces algorithmes qui donnent toute sa valeur à une transformée, et qui font qu'elle sera utilisable de façon intensive. Ces algorithmes sont des conséquences directes de la nature algébrique des transformées, et ne font que refléter certaines symétries et invariances algébriques.

1 Transformée de Hartley

L'un des désavantages de la transformée de Fourier discrète est qu'elle nécessite des calculs avec des nombres complexes, ce qui n'est pas adapté au calcul des convolutions avec des signaux réels. En effet, d'inutiles multiplications et additions complexes (plus coûteuses que les multiplications et additions réelles) sont effectuées, et les erreurs d'arrondi n'en sont qu'amplifiées. Nous allons définir une transformée, appelée *transformée de Hartley*, qui permet, à l'instar de la TFD, de calculer des produits de convolution, mais qui ne fait intervenir que des calculs avec des nombres réels. Une référence très complète sur la *transformée de Hartley* et ses applications est le livre de BRACEWELL [10].

1.1 Définition et premières propriétés

Définition 1.1 (Transformée de Hartley). Soit $f = \{f[n]\}_{n=0}^{N-1} \in \mathbb{C}^N$. On définit sa *transformée de Hartley* discrète $\mathscr{H}(f) \in \mathbb{C}^N$ par

$$\forall k \in \{0,\dots,N-1\}, \quad \mathscr{H}(f)[k] \overset{\text{déf.}}{=} \sum_{n=0}^{N-1} f[n] \operatorname{cas}\left(\frac{2\pi}{N} nk\right), \tag{1.1}$$

où l'on a noté $\operatorname{cas}(x) \overset{\text{déf.}}{=} \sin(x) + \cos(x) = \sqrt{2}\cos\left(x - \frac{\pi}{4}\right)$.

***Remarque* 1.2.** La transformée de Hartley discrète a son analogue continu, à savoir, pour une fonction $f \in L^1(\mathbb{R})$, la fonction

$$\mathscr{H}(f) : s \to \int_{\mathbb{R}} f(x) \operatorname{cas}(2\pi s x) \mathrm{d}x.$$

La plupart des énoncés valables dans le cas discret ont une formulation analogue pour le cas continu, et nous laissons au lecteur le soin de les énoncer.

Proposition 1.3 (Formule d'inversion). *$\mathscr{H}$ est un isomorphisme de $\mathbb{R}^N$ dans $\mathbb{R}^N$. Plus précisément, pour $f \in \mathbb{R}^N$, on a $\mathscr{H}^2(f) = Nf$, ce qui signifie que l'inverse de la transformée de Hartley est $\mathscr{H}^{-1} = \frac{1}{N}\mathscr{H}$.*

Démonstration. Nous allons utiliser, pour n et $n' \in \{0,\dots,N-1\}$, la relation d'orthogonalité

$$\sum_{k=0}^{N-1} \operatorname{cas}\left(\frac{2\pi}{N} nk\right) \operatorname{cas}\left(\frac{2\pi}{N} n'k\right) = N\delta_n^{n'}, \tag{1.2}$$

où $\delta_n^{n'}$ vaut 1 si $n = n'$ et 0 sinon. On note $\omega = e^{\frac{2i\pi}{N}}$, d'où

$$v_n[k] \overset{\text{déf.}}{=} \operatorname{cas}\left(\frac{2\pi}{N} nk\right) = \frac{1}{2}\left(\omega^{nk}(1-\mathrm{i}) + \omega^{-nk}(1+\mathrm{i})\right).$$

On calcule donc

$$\langle v_n, v_{n'}\rangle = \frac{(1-\mathrm{i})^2}{4}\sum_k \omega^{(n+n')k} + \frac{(1+\mathrm{i})^2}{4}\sum_k \omega^{(n+n')k} + \frac{(1-\mathrm{i})(1+\mathrm{i})}{2}\sum_k \omega^{(n-n')k}. \tag{1.3}$$

Les deux premières sommes sont opposées, et la dernière vaut $N\delta_n^{n'}$. Pour obtenir la formule d'inversion, on note que $\mathscr{H}(f)[n] = \langle f, v_n\rangle$, d'où

$$\mathscr{H}(\mathscr{H}(f))[n] = \sum_{k=0}^{N-1} \mathscr{H}(f)[k] v_n[k] = \sum_{k=0}^{N-1} f[k] \langle v_k, v_n\rangle = Nf[n]. \qquad \square$$

Proposition 1.4. *On a les relations suivantes entre la transformée de Fourier discrète et la transformée de Hartley d'un vecteur $f \in \mathbb{R}^N$:*

$$\begin{aligned} \mathscr{H}(f) &= \mathfrak{Re}(\mathscr{F}(f)) + \mathfrak{Im}(\mathscr{F}(f)) \\ \mathscr{F}(f) &= \mathscr{H}(f)_s - \mathrm{i}\mathscr{H}(f)_a. \end{aligned} \tag{1.4}$$

On a noté, pour $g \in \mathbb{R}^N$, g_s et g_s les parties symétrique et anti-symétrique de g, introduites à la définition 5.2, chap. III.

***Remarque* 1.5.** Les relations (1.4) impliquent, dans le cas où l'on se restreint à des vecteurs réels, une correspondance bijective entre transformée de Fourier discrète et transformée de Hartley. Comment expliquer que N nombres complexes (pour la TFD) puissent contenir autant d'informations que seulement N nombres réels ? En réalité, il n'y a pas de contradiction, il faut simplement se rappeler que dans le cas d'un signal réel, le vecteur $\mathscr{F}(f)$ est le conjugué de $\mathscr{F}(f^\natural)$ (où $f^\natural$ est défini à l'équation (5.1), chap. III), il y a donc une redondance d'informations (exactement deux fois trop d'informations). Pour un signal réel, la transformée de Hartley est nettement plus économique (que ce soit en matière de temps de calcul ou d'espace mémoire) que la transformée de Fourier, puisque l'on va manipuler exactement deux fois moins d'informations. C'est cette qualité que nous allons exploiter au paragraphe suivant.

1.2 Transformée de Hartley rapide

Comme pour la TFD, on dispose d'un algorithme rapide pour calculer la transformée de Hartley. Cet algorithme a été décrit en détail par ULLMANN [73]. Pour le comprendre, nous allons effectuer un découpage de la transformée, comme nous l'avons déjà fait lors de l'étude de l'algorithme FFT. Il s'agit bien sûr d'exploiter les symétries (algébriques) de la transformation, ainsi que les propriétés de la fonction cas. Une fois tout ceci mis en place, nous verrons tout naturellement apparaître un algorithme récursif.

Cet algorithme utilise une propriété de décimation temporelle. Pour obtenir l'équation de récurrence correspondante, nous allons procéder comme nous l'avons déjà fait pour la transformée de Fourier discrète. If faut décomposer la somme qui définit $\mathscr{H}(f)$, pour $f \in \mathbb{C}^N$, de la manière suivante :

$$\mathscr{H}(f)[k] = \sum_{n=0}^{N/2-1} f[2n]\cas\left(\frac{2\pi}{N}2nk\right) + \sum_{n=0}^{N/2-1} f[2n+1]\cas\left(\frac{2\pi}{N}(2n+1)k\right). \tag{1.5}$$

Utilisons les notations de (2.4), chap. III. On reconnaît dans la somme de gauche une transformée de Hartley de longueur $N/2$, plus précisément $\mathscr{H}(f^0)$. La somme de droite pose problème, mais on peut lever cette difficulté en utilisant la propriété suivante de la fonction cas.

Proposition 1.6. *On a, pour* $(\alpha,\beta) \in \mathbb{R}^2$,

$$\cas(\alpha+\beta) = \cas(\alpha)\cos(\beta) + \cas(-\alpha)\sin(\beta). \tag{1.6}$$

Démonstration. Cette propriété se démontre très simplement en utilisant les identités trigonométriques bien connues des fonctions cos et sin. □

En utilisant cette propriété, on peut réécrire la deuxième somme de l'équation (1.5) pour obtenir

$$\mathscr{H}(f)[k] = \mathscr{H}(f^0) + \cos\left(\frac{2\pi}{N}k\right)\mathscr{H}(f^1)[k] + \sin\left(\frac{2\pi}{N}k\right)\mathscr{H}(f^1)[-k].$$

On définit alors l'opérateur χ_N^x, pour $x \in \mathbb{R}$, de la manière suivante :

$$\forall a \in \mathbb{C}^N, \quad \chi_N^x a \overset{\text{déf.}}{=} \left\{a[j]\cos\left(\frac{2\pi jx}{N}\right) + a^\natural[j]\sin\left(\frac{2\pi jx}{N}\right)\right\}_{j=0}^{N-1} \in \mathbb{C}^N. \tag{1.7}$$

On rappelle que $a^\sharp$ est le vecteur symétrisé, équation (5.1), chap. III. Nous allons maintenant découper le vecteur $\mathscr{H}(f)$ en ses parties gauche et droite, notées $\mathscr{H}(f)_g$ et $\mathscr{H}(f)_d$. La relation

$$\operatorname{cas}\left(\frac{2\pi}{N}(n+N/2)\right) = -\operatorname{cas}\left(\frac{2\pi}{N}n\right)$$

permet d'obtenir une écriture très simple de l'équation de récurrence cherchée :

$$\mathscr{H}(f)_g = \mathscr{H}(f^0) + \chi_{N/2}^{1/2}\mathscr{H}(f^1), \tag{1.8}$$

$$\mathscr{H}(f)_d = \mathscr{H}(f^0) - \chi_{N/2}^{1/2}\mathscr{H}(f^1). \tag{1.9}$$

Ces équations permettent d'implémenter de façon immédiate un algorithme de calcul rapide que l'on nomme *FHT* pour *Fast Hartley Transform*. La procédure `fht` réalise cet algorithme par des appels récursifs, et son programme se trouve à la section 2, annexe A.

***Remarque* 1.7.** Les équations de récurrence (1.8) et (1.9) montrent que le calcul de $\mathscr{H}(f)$ nécessite en fait le calcul de deux transformées de taille moitié. Cependant, à cause du terme inversé $a^\sharp$ présent dans l'opérateur $\chi_N^x(a)$, il est difficile d'utiliser un schéma papillon comme pour l'algorithme FFT. Dans le but d'écrire un algorithme itératif, ULLMANN, dans [73], montre comment on peut faire le calcul par un double schéma papillon, en utilisant quatre entrées.

On peut montrer qu'une transformation papillon de l'algorithme FHT nécessite quatre multiplications et six additions réelles. En ce qui concerne l'algorithme FFT, on trouve une addition et deux multiplications complexes, soit quatre additions et six multiplications. Cependant, la boucle effectuée pour calculer les transformées papillons de l'algorithme FFT court de 0 à $N-1$, alors que pour l'algorithme FHT, il s'agit d'une boucle entre 0 et $N/2-1$. Au final, l'algorithme FHT présente le double avantage de nécessiter deux fois moins d'opérations, et d'utiliser deux fois moins de mémoire (on ne manipule pas de nombres complexes).

***Remarque* 1.8. (Zero padding)**. Nous avons déjà expliqué au paragraphe 1.3, chap. IV, qu'il était possible de représenter assez fidèlement la TFD continue d'un signal fini par *zero padding*. Il est bien sûr possible de faire de même avec la transformée de Hartley. L'algorithme FHT, allié à une procédure de zéro-padding permet donc de calculer simplement une transformée de Hartley continue. Plus on ajoute de zéros, plus on calcule de valeurs intermédiaires de la transformée, et plus la précision de calcul est bonne. La figure 5.1 montre ce procédé sur un signal simple (trianglulaire), et permet de comparer le spectre de Fourier (en fait sa partie réelle) et son spectre de Harley. Le spectre de Hartley étant la différence entre les parties réelle et imaginaire du spectre de Fourier, les ressemblances ne sont donc pas fortuites !

1.3 Calcul de convolution par transformée de Hartley

Les équations (1.4) montrent qu'il est possible de calculer une TFD d'un signal réel au moyen d'un calcul de transformée de Hartley, ce qui évite d'avoir recours à des multiplications et additions complexes. Dans le même ordre d'idée, on peut établir une formule qui permet de calculer un produit de convolution de deux signaux réels en utilisant uniquement des transformées de Hartley.

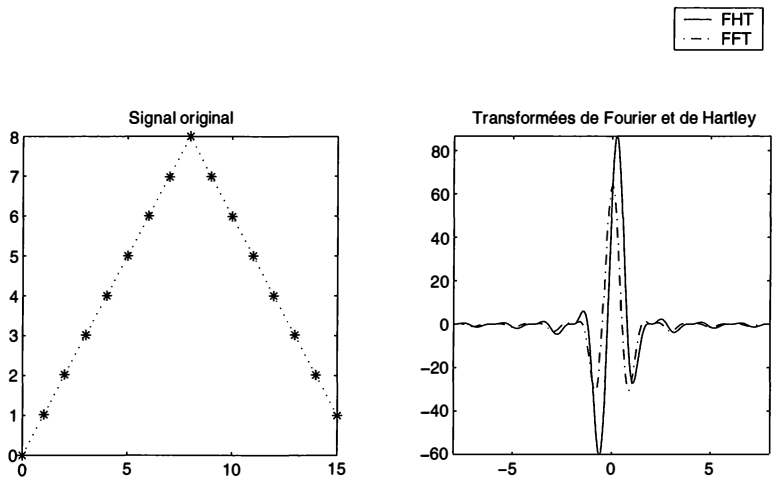

FIG. 5.1 – *Comparaison entre le spectre de Hartley et le spectre de Fourier*

Proposition 1.9 (Convolution et transformée de Hartley). *Soient a et b deux vecteurs réels de taille N. On a*

$$\mathscr{H}(a*b) = \frac{1}{2}\{\mathscr{H}(a)\mathscr{H}(b) - \mathscr{H}(a)^{\sharp}\mathscr{H}(b)^{\sharp} + \mathscr{H}(a)\mathscr{H}(b)^{\sharp} + \mathscr{H}(a)^{\sharp}\mathscr{H}(b)\},$$

soit, pour $n = 0,\ldots,N-1$,

$$\mathscr{H}(a*b)[n] = \frac{1}{2}\{c[n](d[n]+d[-n]) + c[-n](d[n]-d[-n])\},$$

où l'on a noté $c \overset{\text{déf.}}{=} \mathscr{H}(a)$ *et* $d \overset{\text{déf.}}{=} \mathscr{H}(b)$. *Il faut bien sûr considérer l'indice* $-n$ *comme pris modulo N.*

Démonstration. Après interversion des sommations dans l'expression de $\mathscr{H}(a*b)[k]$ on trouve

$$\mathscr{H}(a*b)[n] = \sum_{l=0}^{N-1} a[l] \sum_{k=0}^{N-1} b[k] \operatorname{cas}\left(\frac{2\pi}{N} n(k+l)\right).$$

Il suffit ensuite d'utiliser la relation (1.6) pour obtenir

$$\mathscr{H}(a*b)[n] = \mathscr{H}(b)[n] \sum_{l=0}^{N-1} a[l] \cos\left(\frac{2\pi}{N} nk\right) + \mathscr{H}(b)^{\sharp}[n] \sum_{l=0}^{N-1} a[l] \sin\left(\frac{2\pi}{N} nk\right).$$

Pour conclure, il ne reste plus qu'à exprimer les fonctions cos et sin à l'aide de la fonction cas de la manière suivante :

$$\begin{cases} \cos(x) = \operatorname{cas}(x) + \operatorname{cas}(-x) \\ \sin(x) = \operatorname{cas}(x) - \operatorname{cas}(-x) \end{cases}. \qquad \square$$

Le programme `fht_convol`, que l'on peut trouver à la section 2, annexe A, réalise cette convolution de signaux réels à l'aide de l'algorithme FHT. Il est plus économique, à la fois en termes de temps de calcul, et en termes de place mémoire utilisée.

***Remarque* 1.10. (Auto corrélation).** Dans le cas où les signaux a et b sont égaux, on peut optimiser légèrement l'implémentation de l'algorithme de calcul de convolution en écrivant

$$\mathscr{H}(a*a)[k] = c[k]c[-k] + \frac{1}{2}(c[k]^2 - c[-k]^2).$$

2 Transformée en Z et applications

Notre objectif ici est de donner un cadre relativement général pour étudier des transformées qui possèdent des propriétés similaires à celles de la transformée de Fourier discrète, et en quelque sorte, la généralisent (comme par exemple la transformée en Z vectorielle à la section 3, et la transformée de Fourier fractionnaire à la section 4). Pour ce faire, nous allons nous intéresser aux fonctions génératrices, que l'on appellera transformée en Z. Après la définition de cette transformée, nous présenterons une application immédiate de la transformée en Z à la construction de filtres définis par des équations de récurrence.

L'ouvrage de WICH [78] regroupe quantité d'informations sur la transformée en Z. Le lien avec les séries génératrices et les fonctions holomorphes est longuement discuté dans le livre de DEMENGEL [25].

2.1 Définition et propriétés formelles

Notre but est de faire le lien entre transformée en Z et séries entières. Nous allons donc définir la notion de transformée en Z dans le cadre de signaux de taille potentiellement infinie. Dans la pratique, nous utiliserons la transformée sur des signaux de taille finie, ce qui permet de ne pas avoir à se soucier d'éventuels problèmes de convergence. Nous considérerons donc une suite $f = \{f[n]\}_{n\in\mathbb{Z}} \in \mathbb{C}^{\mathbb{Z}}$. Seule l'étude de la fonction de transfert d'un filtre récursif (au paragraphe 2.2) requiert l'utilisation d'une série infinie. Cependant, même dans ce cas, nous aurons une expression exacte de la transformée (sous forme de fraction rationnelle), ce qui nous évitera tout problème.

Définition 2.1 (Transformée en Z). Soit $f \in \mathbb{C}^{\mathbb{Z}}$. On appelle *transformée en Z* de f la fonction

$$\mathscr{Z}(f) : \left\{ \begin{array}{ccc} D & \longrightarrow & \mathbb{C} \\ z & \longmapsto & \sum_{n=-\infty}^{+\infty} f[n] z^{-n} \end{array} \right. , \tag{2.1}$$

où D est l'ensemble (éventuellement vide) des points où la série converge.

***Remarque* 2.2. (Transformée en Z et fonctions holomorphes).** La théorie des *séries de Laurent* montre que la transformée en Z est en fait définie à l'intérieur d'une couronne du type

$$C_\alpha^\beta \overset{\text{déf.}}{=} \{z \in \mathbb{C};\ \alpha < |z| < \beta\} \qquad \text{pour } 0 \leqslant \alpha < \beta,$$

où β peut éventuellement valoir $+\infty$. On pourra se référer par exemple au livre de CARTAN [15] pour une étude complète de la décomposition d'une fonction holomorphe en série entière et en série de Laurent. La fonction $\mathscr{Z}(f)$ est donc holomorphe à l'intérieur de son disque de convergence. On dit aussi souvent que $\mathscr{Z}(f)$ est la série génératrice associée à la suite f. Comme nous le verrons plus tard (en considérant les filtres récursifs), cette notion permet de représenter de façon élégante certaines suites qui sont définies par récurrence. Pour un exposé intéressant sur la résolution de récurrences par séries génératrices, on pourra regarder l'ouvrage de référence de DONALD KNUTH [37]. Un exposé plus simple se trouve dans [34]. Le livre [79] constitue un ouvrage original sur le sujet.

Exemple 2.3. Voici quelques exemples simples.

1. Soit la suite $f \in \mathbb{C}^{\mathbb{Z}}$ définie par $f[n] \overset{\text{déf.}}{=} 0$ pour $n < 0$ et $f[n] \overset{\text{déf.}}{=} 1$ sinon. On a

$$\mathscr{Z}(f)(z) = \sum_{n\geqslant 0} z^{-n} = \frac{1}{1-z^{-1}},$$

la série étant convergente dans $C_1^{+\infty}$.

2. Pour $(a,b) \in \mathbb{C}^2$, on définit la suite $f \in \mathbb{C}^{\mathbb{Z}}$ par $f[n] \overset{\text{déf.}}{=} a^n$ pour $n \geq 0$ et $f[n] \overset{\text{déf.}}{=} b^n$ sinon. On a

$$\mathcal{Z}(f)(z) = \sum_{n \geqslant 0} \left(\frac{a}{z}\right)^n + \sum_{n \geqslant 0} \left(\frac{z}{b}\right)^n - 1 = \frac{1}{1-a/z} + \frac{1}{1-z/b} - 1,$$

la somme étant convergente dans $C_{|a|}^{|b|}$.

Proposition 2.4 (Propriétés de la transformée en Z). *On note f et g deux suites de $\mathbb{C}^{\mathbb{Z}}$. Outre la linéarité, voici les propriétés importantes de la transformée en Z :*

(i) *convolution linéaire :*

$$\mathcal{Z}(f \star g) = \mathcal{Z}(f)\mathcal{Z}(g),$$

la série définissant $\mathcal{Z}(f \star g)$ étant au moins convergente sur l'intersection des couronnes de convergence de $\mathcal{Z}(f)$ et $\mathcal{Z}(g)$ (si elle est non vide).

(ii) *dérivation :*

$$\mathcal{Z}\left(\{nf[n]\}_{n\in\mathbb{Z}}\right)(z) = -z\frac{d}{dz}\mathcal{Z}(f)(z).$$

Démonstration. Nous allons faire la preuve de la propriété la plus importante, la propriété de convolution (i). La convergence absolue sur l'intersection des domaines de convergence nous permet d'écrire, pour z dans cette intersection :

$$\begin{aligned}\mathcal{Z}(f\star g)(z) &= \sum_{n=-\infty}^{+\infty} (f\star g[n])z^{-n} = \sum_{n=-\infty}^{+\infty}\sum_{m=-\infty}^{+\infty} f[n]g[n-m]z^{-n} \\ &= \sum_{m=-\infty}^{+\infty} f[m]z^{-m}\left(\sum_{n=-\infty}^{+\infty} g[n-m]z^{-(n-m)}\right) \\ &= \mathcal{Z}(f)(z)\mathcal{Z}(g)(z).\end{aligned}$$

La convergence absolue justifie l'interversion des deux sommations. □

***Remarque* 2.5.** Il existe aussi des formules permettant d'intégrer la fonction $\mathcal{Z}(f)$. Elles nécessitent l'utilisation d'intégrales curvilignes, ainsi que d'éventuelles précautions sur le comportement à l'infini de f.

2.2 Filtres récursifs

Nous avons déjà vu à la section 2, chap. IV, la définition des filtres linéaires, ainsi que leur utilisation pour modifier de façon adéquate des signaux. Nous allons définir ici un nouveau type de filtres, nommés *filtres récursifs*, qui permettent eux aussi de réaliser des opérations intéressantes sur des signaux. La transformée en Z est très souvent utilisée pour représenter la fonction de transfert de filtres récursifs. Elle est en effet très pratique pour au moins deux raisons :

- la représentation sous la forme d'une fonction va permettre d'utiliser des opérations algébriques pour modifier le filtre (somme, produit, dérivée, etc.).
- la représentation complexe sous forme de module et argument (c'est-à-dire en coordonnées polaires) va permettre de créer de toutes pièces des filtres de façon intuitive.

D'excellents livres sont disponibles sur le traitement du signal digital en général, et l'utilisation de la transformée en Z pour la création de filtres. On peut citer entre autres [67] ainsi que [17].

Définition 2.6 (Filtre récursif). Soient $a = \{a_0, \ldots, a_k\}$ et $b = \{b_1, \ldots, b_l\}$ deux vecteurs de nombres complexes. Ils permettent de définir le *filtre récursif* Φ_a^b opérant sur des suites $x \in \mathbb{C}^{\mathbb{N}}$ en définissant $y = \Phi_a^b(x) \in \mathbb{C}^{\mathbb{N}}$ par

$$\begin{aligned} \forall n \in \mathbb{Z}, \quad y[n] = a_0 x[n] + a_1 x[n-1] + \cdots + a_k x[n-k] \\ + b_1 y[n-1] + \cdots + b_l y[n-l]. \end{aligned} \tag{2.2}$$

***Remarque* 2.7.** L'équation (2.2) définit donc le filtre par récurrence. Le filtre n'utilise, pour le calcul de $y[n]$, que des valeurs déjà calculées de y ainsi que des entrées du vecteur x d'indice inférieur à n. Conformément à la terminologie déjà employée (pour les filtres de convolution), on dit que le filtre est causal. On obtient ainsi un algorithme simple pour évaluer l'action du filtre sur un signal x. Dans la suite, on ne considérera que des filtres x à support fini, et tous les signaux mis en jeu (en particulier x et y) seront indexés à partir de zéro. On remarque que le calcul des premières entrées du vecteur y demande la connaissance de $y[-1], y[-2], \ldots, y[-l]$, ainsi que $x[n-1], x[n-2], \ldots, x[-k]$. Par convention, on supposera, sauf mention explicite du contraire, que ces entrées sont nulles.

***Remarque* 2.8. (Lien avec les filtres de convolution).** Nous avons déjà défini, à la section 2, chap. IV, des filtres de convolution finis. On remarque que dans le cas où le vecteur b est nul, le filtre récursif est en fait un filtre de convolution fini. Les filtres récursifs peuvent être vus, d'un point de vue théorique, comme une généralisation des filtres de convolution linéaire. En fait, nous verrons même dans l'étude qui suit, par le calcul de la transformée en Z, que les filtres récursifs sont des filtres de convolution, mais dont la réponse impulsionnelle est infinie. C'est pour cela qu'on les nomme filtres *IIR* dans la littérature anglo-saxonne (abréviation d'**I**nfinite **I**mpulse **R**esponse). D'un point de vue pragmatique, il n'en va pas de même : les usages de ces deux types de filtrages sont différents, principalement à cause des propriétés inhérentes à la fois à leur implémentation, et à leurs réponses impulsionnelles. Contrairement à un filtre de convolution linéaire qui peut être calculé de façon rapide par FFT, un filtre récursif se limitera généralement à quelques termes de récurrence seulement (ce qui signifie que k et l seront souvent supposés petits). Ces filtres permettent donc, sans avoir à calculer de convolutions, d'obtenir des réponses très longues (en théorie infinies) pour des coûts très faibles, de l'ordre de $O(N)$ (où N désigne la taille des vecteurs filtrés), si on suppose que k et l sont petits devant N. Par contre, le fait de n'avoir qu'un petit nombre de coefficients à sa disposition pour créer ces filtres les rend moins maniables. Enfin, il est à noter que le calcul d'un filtre par récurrence facilite la propagation des erreurs d'arrondi (puisqu'elles s'ajoutent au fur et à mesure des calculs). Pour enrayer ce phénomène, on est souvent obligé de faire des calculs en double précision.

L'idée de ce paragraphe est d'utiliser la transformée en Z pour définir un filtre récursif d'une façon plus agréable que par l'équation de récurrence (2.2). Pour ce faire, calculons la transformée en Z de cette équation. Après regroupement des termes, on obtient

$$\mathcal{Z}(y)(z) = H(z)\mathcal{Z}(x)(z),$$

avec

$$H(z) \overset{\text{déf.}}{=} \frac{a_0 + a_1 z^{-1} + \cdots + a_k z^{-k}}{1 - b_1 z^{-1} - \cdots - b_l z^{-l}}.$$

On souhaite affirmer que H est la transformée en Z d'une certaine fonction de transfert h, dans le but d'écrire le filtre Φ_a^b comme étant le filtre de convolution Φ_h. En effet, supposons que l'on ait réussi à trouver un tel h. En utilisant le résultat de convolution 2.4 (i), on obtient

$$\mathcal{Z}(y) = H \cdot \mathcal{Z}(x) = \mathcal{Z}(h \star x) = \mathcal{Z}(h) \cdot \mathcal{Z}(x).$$

On voit donc que $H = \mathcal{Z}(h)$. Le problème est de savoir si la connaissance de H permet de déterminer un tel h, autrement dit, s'il est possible de calculer l'inverse de la transformée en Z. La réponse est loin d'être évidente. On sait, grâce aux *formules de Cauchy*, qu'une fonction holomorphe définie sur une couronne C_α^β admet un unique développement en série de Laurent à l'intérieur de cette couronne. Cependant, il reste à savoir à quelle couronne nous avons à faire. Selon le choix du domaine de convergence, on obtient une suite h différente. Nous avons donc besoin de plus d'informations sur le filtre. En prenant la transformée en Z de l'équation de récurrence (2.2), nous avons en quelque sorte « oublié » les conditions aux limites, c'est-à-dire les valeurs de $x[-1], \ldots, x[-k]$ ainsi que $y[-1], \ldots, y[-l]$. Dans la pratique, on dispose en fait d'informations simples qui permettent de retrouver le domaine de convergence qui correspond à l'équation de récurrence, et ainsi retrouver la suite h à partir de H.

- Si on considère l'équation de récurrence écrite en (2.2), on voit que le filtre considéré est causal (c'est-à-dire que la détermination de $y[n]$ ne dépend que des entrées d'indices inférieurs à n de $x[n]$). Ceci implique que la réponse impulsionnelle est nulle pour les indices négatifs, ce qui se traduit par le fait que le domaine de convergence est l'extérieur d'un cercle (il suffit d'utiliser le critère de convergence pour une série entière, puisque l'on a en fait une série entière en $1/z$).
- Le plus souvent, on veut que le filtre soit stable en plus d'être causal, ce qui implique, comme nous allons le voir à la proposition 2.10, que le domaine de convergence doit contenir le cercle unité.

Voici un exemple qui montre l'importance de la spécification du domaine de convergence.

Exemple 2.9 (Causalité et domaine de convergence). On considère un signal y qui satisfait à l'équation aux différences

$$y[n] = \alpha y[n-1] + x[n], \tag{2.3}$$

où $x \in \mathbb{C}^{\mathbb{Z}}$ est le signal d'entrée. On obtient donc pour la réponse impulsionnelle la transformée en Z suivante :

$$H(z) = \frac{1}{1 - \alpha z^{-1}}.$$

De façon naturelle, l'équation (2.3) définit un filtre causal, et on peut essayer de trouver sa réponse impulsionnelle par la méthode que nous venons d'expliquer. La fonction H doit donc être considérée comme définie à l'extérieur du cercle de rayon $|\alpha|$. On obtient alors, par développement de la fraction en série entière en fonction de z^{-1},

$$\forall z \in C_{|\alpha|}^{+\infty}, \quad H(z) = \sum_{n \geqslant 0} \alpha^n z^{-n}.$$

D'où la valeur de la réponse impulsionnelle h :

$$\forall n < 0, \quad h[n] = 0,$$
$$\forall n \geqslant 0, \quad h[n] = \alpha^n.$$

C'est donc bien un filtre causal. Si l'on veut qu'il soit stable, on vérifie qu'il faut en outre supposer que $|\alpha| < 1$. Cependant, un autre cas peut se produire, si on considère que la fonction H est définie sur $C_0^{|\alpha|}$. Il faut, cette fois faire la manipulation suivante :

$$\forall z \in C_0^{|\alpha|}, \quad H(z) = \frac{-z/\alpha}{1 - z/\alpha} = \sum_{n<0} -\alpha^n z^{-n}.$$

On obtient ainsi l'expression de h :

$$\begin{aligned} &\forall n < 0, \quad h[n] = -\alpha^n, \\ &\forall n \geqslant 0, \quad h[n] = 0. \end{aligned}$$

Le filtre est donc anti-causal, et la stabilité impose que $|\alpha| > 1$. Le fait qu'il soit anti-causal s'explique simplement en réécrivant l'équation de récurrence (2.3) sous une forme inverse (propagation des calculs dans l'autre sens) :

$$y[n-1] = \frac{1}{\alpha} y[n] - \frac{1}{\alpha} x[n].$$

Une fois que l'on a réussi à obtenir la valeur de h, on voit que le filtre Φ_a^b est bien un filtre de convolution, mais qui reste un peu spécial, puisque sa réponse impulsionnelle, dans le domaine de la transformée en Z, peut être mise sous la forme d'une fraction rationnelle.

Nous venons de voir la façon dont on peut représenter la fonction de transfert d'un filtre récursif grâce à sa transformée en Z. Cette représentation a l'avantage d'offrir une écriture compacte et simple. Ceci permet de faire des calculs de façon efficace sur les filtres, plus simplement que ne l'autorise la représentation sous forme de convolution exposée au paragraphe 2.1, chap. IV. Avant d'étudier l'utilisation de la transformée en Z pour la création de nouveaux filtres, voyons le rapport entre la fraction rationnelle représentant un filtre, et la stabilité de ce filtre.

Proposition 2.10 (Pôles et stabilité). *Soit $H(z) = \frac{A(1/z)}{B(1/z)}$ la transformée en Z de la fonction de transfert d'un filtre Φ_a^b (qui est donc causal). Alors, ce filtre est stable si et seulement si tous les pôles de H sont situés à l'intérieur du cercle unité Γ.*

Démonstration. Nous avons déjà dit que pour un filtre causal, la série qui définit H converge à l'extérieur d'un certain cercle, et la réciproque est vraie (il suffit de considérer le développement en $1/z$ d'une fonction holomorphe définie au voisinage de l'infini). Comme Φ_a^b est bien sûr causal, cette remarque s'applique. De plus, nous avons vu au paragraphe 2.2, chap. IV, qu'un filtre de réponse impulsionnelle h était stable si et seulement si $\|h\|_{\ell^1} < +\infty$. Si h est absolument sommable, on a la majoration

$$\forall z \in \Gamma, \quad |H(z)| \leqslant \sum_{n=-\infty}^{+\infty} |h[n] z^{-n}| = \sum_{n=-\infty}^{+\infty} |h[n]| =: \|h\|_{\ell^1}.$$

On voit donc que la condition $\|h\|_{\ell^1} < +\infty$ implique que la série qui définit H est absolument convergente sur le cercle Γ. La réciproque est vraie. Le fait que $\|h\|_{\ell^1} < +\infty$ est donc équivalent au fait que la région de convergence contienne tout l'extérieur du cercle Γ. Or, la région de convergence ne saurait contenir de pôle. Tous les pôles de la fraction rationnelle H doivent donc être de module strictement inférieur à 1. Encore une fois, la réciproque est vraie. □

2.3 Application à la construction de filtres

Dans ce paragraphe, nous allons nous concentrer sur l'utilisation de la transformée en Z pour la création de filtres récursifs digitaux. La démarche pour créer un filtre est de décider de l'emplacement des zéros et des pôles de la fonction de transfert. Comme nous l'avons déjà vu, les pôles doivent être contenus dans le cercle unité pour que le filtre soit causal et stable. Par exemple, on peut choisir les emplacements repérés sur la figure 5.2, ce qui conduit à l'expression de la fonction de transfert :

$$H(z) = \frac{(z-e^{i\pi/4})(z-e^{-i\pi/4})}{(z-0.9e^{i\pi/4})(z-0.9e^{-i\pi/4})} \approx \frac{1-1.414z+z^2}{0.810-1.273z-z^2}.$$

Cette transformée en Z est représentée à la figure 5.3. De cette expression, on déduit

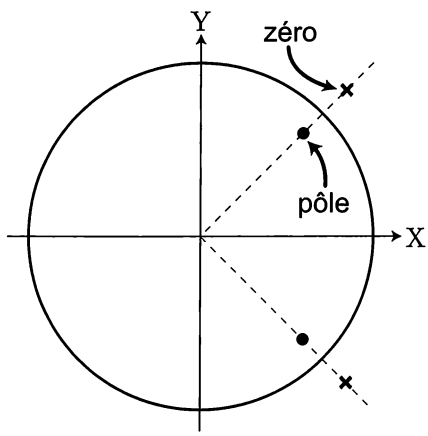

FIG. 5.2 – *Positionnement des pôles et des zéros*

immédiatement les coefficients du filtre récursif associé :

$$a_0 = 1 \quad a_1 \approx -1.414 \quad a_2 = 1$$
$$b_0 \approx 1.273 \quad b_1 \approx -0.810.$$

Dès lors, si l'on désire calculer la réponse fréquentielle, deux méthodes s'offrent à nous :

- calculer la réponse fréquentielle directement à partir de H : il suffit de considérer la valeur de la fonction de transfert sur le cercle unité, c'est-à-dire la fonction $\xi \mapsto H(e^{2i\pi\xi})$, pour $\xi \in [0,1]$. Par transformée de Fourier inverse, on en déduit la réponse impulsionnelle (le calcul approché se fait en échantillonnant la réponse fréquentielle, puis par FFT inverse).
- calculer la réponse impulsionnelle en utilisant l'équation de récurrence du filtre et en l'appliquant pour l'impulsion δ_0. On peut ensuite utiliser une transformée de Fourier (discrète) pour approcher la réponse fréquentielle. Pour avoir suffisamment de précision, il faudra calculer une réponse impulsionnelle approchée assez longue.

La figure 5.4 montre les réponses impulsionnelles et fréquentielles du filtre. Elles ont été calculées directement à partir de la fonction de transfert H représentée à la figure 5.3. Au paragraphe 3.1, nous présenterons un algorithme de calcul rapide pour déterminer la valeur de la transformée en Z sur certains contours, et nous calculerons la réponse impulsionnelle à partir de l'équation de récurrence (2.2).

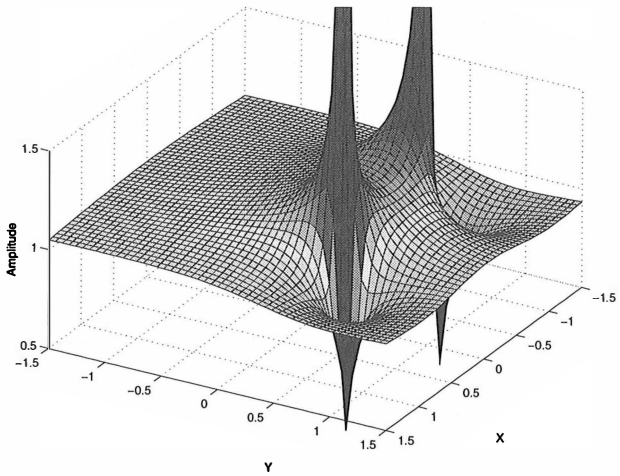

FIG. 5.3 – *Transformée en Z du filtre*

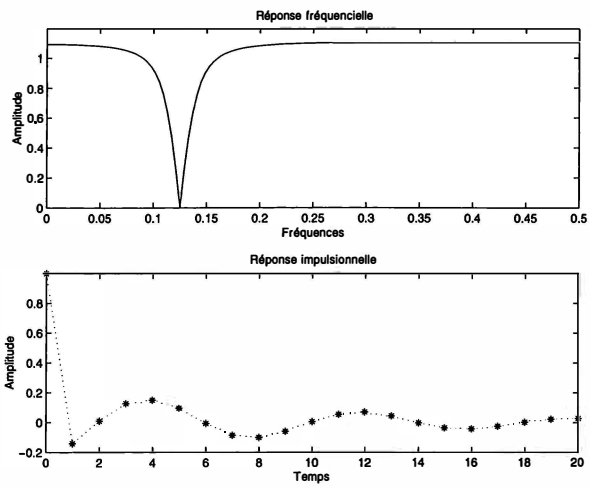

FIG. 5.4 – *Réponse fréquentielle et impulsionnelle du filtre*

L'exemple précédent est loin d'être aussi anecdotique qu'il en a l'air. En effet, en décomposant la fraction rationnelle H, on va pouvoir se ramener au cas de filtres simples, c'est-à-dire avec au plus deux pôles et deux zéros. Voici deux démarches que l'on peut suivre.

– **Décomposition en produits.** On peut factoriser les numérateurs et les dénominateurs en polynômes de degré 1 ou 2 sur $\mathbb{R}[X]$ (respectivement de degré 1 sur $\mathbb{C}[X]$). On obtient ainsi l'écriture du filtre Φ_a^b sous la forme d'une cascade de filtres :

$$\Phi_a^b = \Phi_{\alpha_1}^{\beta_1} \circ \cdots \circ \Phi_{\alpha_p}^{\beta_p},$$

où chaque α_i et β_i représente les coefficients d'un polynôme de degré au plus 2 (respectivement au plus 1). Le filtre Φ_a^b correspond donc à la mise en série d'une suite de filtres récursifs d'ordre au plus 2 (respectivement au plus 1).

– **Décomposition en éléments simples.** On peut décomposer la fraction H en somme d'éléments simples sur $\mathbb{R}[X]$ (respectivement $\mathbb{C}[X]$). On obtient alors la décomposition

$$\Phi_a^b = \Phi_{\alpha_1}^{\beta_1} + \cdots + \Phi_{\alpha_p}^{\beta_p},$$

où chaque α_i et β_i représente les coefficients d'un polynôme de degré au plus 2 (respectivement au plus 1).

Dans le cas où l'on réalise des décomposition sur $\mathbb{C}[X]$, même si les signaux sont réels, il faudra faire les calculs de convolution avec des nombres complexes. Chacune de ces décompositions fournit une nouvelle façon d'implémenter le filtre récursif, en plus de l'implémentation naïve de l'équation (2.2). L'exercice V.8 applique ces deux méthodes pour calculer les coefficients d'une interpolation par splines.

2.4 Rapprochement avec le filtrage analogique

Avant d'entrer dans les détails de l'implémentation d'une transformée en Z discrète, nous allons essayer d'établir une connexion entre les filtres récursifs digitaux et les filtres analogiques. Les filtres analogiques sont en quelque sorte les ancêtres des filtres digitaux modernes, mais sont encore utilisés dans de nombreuses situations. Il est donc intéressant de comprendre pourquoi les filtres récursifs (qui effectuent des transformations discrètes) permettent de replacer les filtres analogiques (qui effectuent des transformations continues) dans le cadre « moderne » du traitement digital du signal. Sans entrer dans la description du filtrage analogique, disons simplement qu'il s'agit de faire passer un signal continu par un ensemble de composants électroniques, de sorte que le signal de sortie soit relié au signal d'entrée par une équation différentielle linéaire. Le filtre digital se comporte alors comme un système dynamique régi par une équation différentielle.

L'équation aux différences (2.2) est en fait l'analogue discret des équations différentielles que doivent satisfaire les systèmes dynamiques. On peut prendre l'exemple d'un *circuit RLC* (c.f. le schéma de la figure 5.5). On a alors l'équation différentielle suivante qui relie V_e et V_s, les tensions d'entrée et de sortie :

$$2\lambda \frac{\mathrm{d}V_e}{\mathrm{d}t} = \omega_0^2 V_s + 2\lambda \frac{\mathrm{d}V_s}{\mathrm{d}t} + \frac{\mathrm{d}^2 V_s}{\mathrm{d}t^2}, \tag{2.4}$$

où l'on a noté $\omega_0^2 = \frac{1}{LC}$ et $\lambda = \frac{R}{2L}$. On peut considérer ce circuit comme un filtre analogique. Comme nous avons développé la transformée en Z pour étudier les filtres discrets, nous allons introduire une autre généralisation de la transformée de Fourier, continue cette fois, pour étudier les filtres analogiques. Il s'agit de la *transformée de Laplace*, qui est définie de la manière suivante.

Définition 2.11 (Transformée de Laplace). Pour une fonction $f : \mathbb{R} \to \mathbb{C}$, on définit formellement sa *transformée de Laplace* par

$$\forall s \in D, \quad \mathscr{L}(f)(s) \overset{\text{déf.}}{=} \int_{t\in\mathbb{R}} f(t) e^{-st} \mathrm{d}t.$$

La fonction $\mathscr{L}(f)$ est définie sur un domaine D où l'intégrale converge.

Moyennant des précautions sur les domaines de définition des fonctions considérées, on peut définir la fonction de transfert du filtre analogique, dans le domaine de Laplace :

$$K(s) \overset{\text{déf.}}{=} \frac{\mathscr{L}(V_s)(s)}{\mathscr{L}(V_e)(s)} = \frac{2\lambda s}{\omega_0^2 + 2\lambda s + s^2}.$$

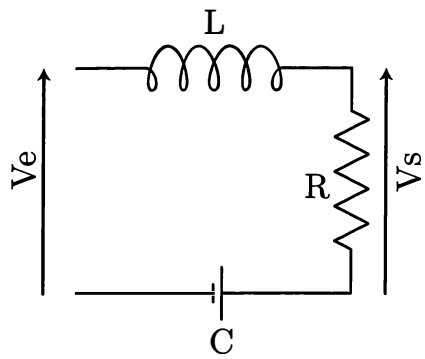

FIG. 5.5 – *Circuit RLC*

On a calculé ici simplement la transformée de Laplace des deux membres de l'équation (2.4). On a utilisé le fait que la transformée de Laplace transforme la dérivation en la multiplication par le paramètre s.

Nous nous intéressons maintenant au problème résultant de la discrétisation, à intervalles de temps régulier Δ, des signaux étudiés. On obtient des signaux discrets $\widetilde{V}_e$ et $\widetilde{V}_s$, qui vérifient l'équation aux différences

$$\frac{2\lambda}{\Delta}\left(\widetilde{V}_e[n]-\widetilde{V}_e[n-1]\right)=\omega_0^2\widetilde{V}_s[n]+\frac{2\lambda}{\Delta}\left(\widetilde{V}_s[n]-\widetilde{V}_s[n-1]\right)+\left(\widetilde{V}_s[n-1]+\widetilde{V}_s[n+1]-2\widetilde{V}_s[n]\right).$$

La résolution de l'équation différentielle d'origine est ainsi remplacée par un schéma aux différences finies. Bien sûr, on aurait pu choisir d'autres méthodes pour calculer de façon approchée les dérivées mises en jeu. Cela aurait conduit à une équation légèrement différente. L'exercice V.7 propose de calculer quelques équations aux différences finies pour un circuit analogique intégrateur.

D'un point de vue purement discret, on obtient un filtre récursif, qui peut être calculé à l'aide d'un ordinateur (et non plus d'un circuit électrique comme c'était le cas pour le filtre RLC). On peut ensuite calculer la fonction de transfert dans le domaine de la transformée en Z pour étudier ce filtre :

$$H(z)\overset{\text{déf.}}{=}\frac{\mathscr{Z}(\widetilde{V}_s)(z)}{\mathscr{Z}(\widetilde{V}_e)(z)}=\frac{2\lambda(1-z)}{2\lambda z^{-1}+\left(\Delta\omega_0+2\lambda-\frac{2}{\Delta}\right)+\left(\frac{1}{\Delta}-2\lambda\right)z}.$$

La réponse fréquentielle du filtre analogique est la fonction $\theta\mapsto K(\mathrm{i}\theta)$, pour $\theta\in\mathbb{R}$. La réponse fréquentielle du filtre digital est $\theta\mapsto H\left(e^{\mathrm{i}\theta}\right)$, pour $\theta\in[0,2\pi[$. La figure 5.6 montre que dans les deux cas, on obtient des filtres passe-bande (il laissent passer une petite gamme de fréquences et diminuent beaucoup les autres). La transformée en Z est en quelque sorte l'outil qui permet d'étudier les filtres récursifs, alors que la transformée de Laplace permet elle d'étudier leurs cousins continus, les filtres analogiques. Ainsi, les principes de construction de filtres digitaux par l'utilisation de la transformée en Z (placement des pôles et des zéros, mise en série de filtres, etc.) s'appliquent aussi à la création de filtres analogiques, à condition d'utiliser la transformée de Laplace.

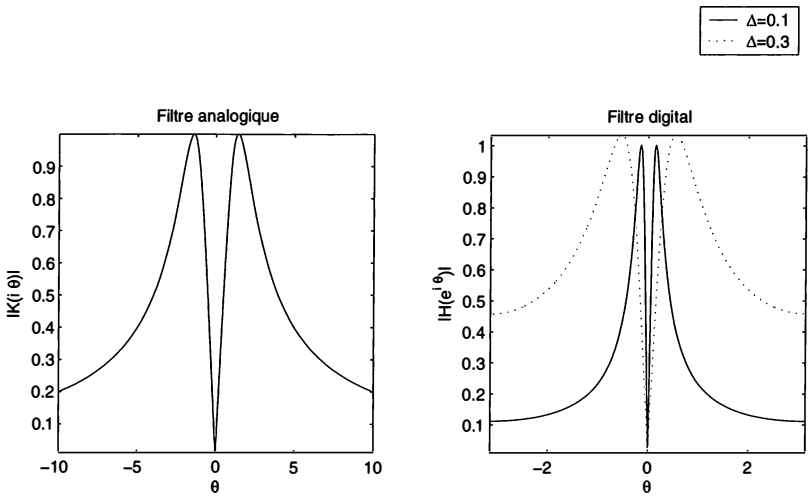

FIG. 5.6 – *Réponses fréquentielles des filtres analogique et digital*

3 Transformée en Z vectorielle

La présentation que nous venons de faire de la transformée en Z est avant tout théorique. Dans le but de calculer effectivement les valeurs d'une fonction transformée en Z, il nous faut échantillonner et faire un nombre fini de calculs. C'est ce que nous allons faire dans ce paragraphe, en définissant une nouvelle transformée, que l'on appelle transformée en Z *vectorielle*. L'algorithme qui en résulte s'appelle algorithme *chirp*. Il a été découvert pour la première fois, dans le cadre (plus restreint) de la TFD par BLUESTEIN, et est bien expliqué dans l'article [71]. Quelques aspects concernant la programmation de la transformée en Z sont abordés dans [2].

3.1 Algorithme de calcul discret

La transformée en Z, même opérant sur des échantillons discrets et finis, n'en demeure pas moins une fonction de la variable complexe z. Le fait est qu'un ordinateur ne sait pas travailler directement avec de telles fonctions (si l'on excepte des logiciels tels MAPLE qui savent faire certaines manipulations formelles). Il nous faut donc un moyen d'évaluer de façon numérique la valeur de la transformée en Z en certains points, et ceci de façon rapide. Pour construire cet algorithme, nous allons introduire une transformée dédiée au calcul de $\mathscr{Z}(f)$ en un nombre de points suffisant (autant que de points dans l'échantillon d'origine).

Définition 3.1 (Transformée en Z vectorielle). On se fixe $z \in \mathbb{C}$. Pour un vecteur $f \in \mathbb{C}^N$, on définit la *transformée en Z vectorielle* (au point z) par

$$\mathscr{G}_z(f) \stackrel{\text{déf.}}{=} \{\mathscr{Z}(f)(z^n)\}_{n=0}^{N-1} = \left\{ \sum_{k=0}^{N-1} f[k] z^{-kn} \right\}_{n=0}^{N-1}. \tag{3.1}$$

***Remarque* 3.2.** Le vecteur obtenu peut être vu comme le calcul de la valeur que prend la transformée en Z le long d'une courbe tracée dans le plan complexe. Si le point z est pris de module 1, cette courbe sera le cercle unité, sinon, il s'agira d'une spirale.

Soit $z \in \mathbb{C}$ fixé. Pour construire un algorithme de calcul efficace de $\mathscr{G}_z(f)$, nous allons utiliser la relation

$$\forall (n,k), \quad nk = \frac{1}{2}\left(n^2 + k^2 - (n-k)^2\right).$$

En l'appliquant à l'équation de définition (3.1), on obtient

$$\mathscr{G}_z(f)[n] = z^{-\frac{n^2}{2}} \sum_{k=0}^{N-1} f[k] z^{-\frac{k^2}{2}} z^{\frac{(n-k)^2}{2}} = z^{-\frac{n^2}{2}} (\widetilde{f} \star g)[n],$$

où l'on a noté g le vecteur défini par

$$\forall k \in \{-N+1, N-1\}, \quad g[k] \stackrel{\text{déf.}}{=} z^{-\frac{k^2}{2}},$$

et f le vecteur

$$\forall k \in \{0, \ldots, N-1\}, \quad \widetilde{f}[n] \stackrel{\text{déf.}}{=} f[k] z^{-\frac{k^2}{2}}.$$

Il faut faire attention au fait que la convolution est une convolution linéaire entre un vecteur de taille N et un vecteur de taille $2N-1$. En utilisant la méthode décrite à la section 3.3, chap. III, on peut calculer une convolution acyclique très rapidement en la remplaçant par une convolution cyclique de taille plus grande. Plus précisément, les vecteurs à convoler étant de taille N et $2N-1$, il faut en théorie calculer une convolution cyclique de taille $3N-2$.En réalité, pour utiliser un l'algorithme FFT de Cooley-Tukey « classique », on ajoute des zéros pour atteindre une taille $M = 2^k$ juste après $3N-2$. On peut cependant faire beaucoup mieux (taille $2N-1$) en exploitant le fait que $g[k] = g[-k]$. Ceci est expliqué à l'exercice V.6 et donne naissance à la procédure MATLAB `czt` (pour **C**hirp **Z** **T**ransform) (voir la correction de l'exercice V.6). On peut ainsi calculer la transformée en Z vectorielle en un temps de l'ordre de $O(N\log(N))$.

L'approche « chirp » consiste donc à remplacer un calcul de transformée par un calcul de convolution. Une autre astuce (l'utilisation d'un corps fini) permet d'arriver à un résultat similaire (lorsque N est un nombre premier). Ceci est l'objet de l'exercice V.9.

Pour terminer, utilisons l'algorithme de calcul que nous venons de construire pour dessiner des transformées en Z vectorielles d'un filtre récursif. On a choisi le filtre dont les pôles et les zéros sont placés sur la figure 5.2. On a calculé la réponse impulsionnelle du filtre en utilisant directement l'équation de récurrence (2.2). On a choisi deux contours, qui correspondent respectivement à $z = e^{\frac{2i\pi}{N}}$ (cercle unité) et $z = 1.001e^{\frac{2i\pi}{N}}$ (spirale). Le premier contour permet de calculer la réponse impulsionnelle (on retrouve la figure 5.4). En effet, calculer la transformée en Z pour $z = e^{\frac{2i\pi}{N}}$ revient à calculer une TFD (avec un gain de temps substantiel si N n'est pas une puissance de 2). Pour le deuxième contour en revanche, on constate que le deuxième « saut » est moins marqué, car la spirale est plus éloignée du deuxième pôle que ne l'est le cercle unité.

3.2 Applications à la transformée de Fourier discrète

Ce paragraphe fait la liaison entre la transformée en Z vectorielle, et la TFD. En particulier, nous allons voir comment ce rapprochement permet de réaliser des calculs de TFD dans le cas où la longueur N des signaux n'est pas un nombre composite du type $N = 2^p$ (cas où l'algorithme FFT est très efficace).

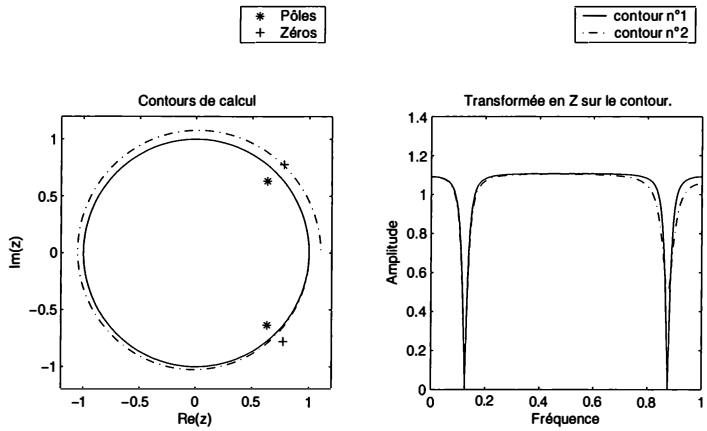

FIG. 5.7 – *Transformée en Z suivant deux contours*

On peut en effet voir la transformée de Fourier discrète comme un cas particulier de transformée en Z vectorielle. Pour cela, on choisit $z = \omega_N \overset{\text{déf.}}{=} e^{\frac{2i\pi}{N}}$ et on obtient, pour un vecteur $f \in \mathbb{C}^N$,

$$\mathscr{F}(f) = \mathscr{G}_{\omega_N}(f).$$

Or une des forces de l'algorithme *chirp transform* présenté au paragraphe précédent est qu'il peut s'appliquer à n'importe quel entier positif N. Contrairement à l'algorithme FFT, il n'est pas restreint aux seuls entiers N dont on connaît une « bonne » factorisation (l'exemple le plus simple est $N = 2^p$), comme cela est expliqué au paragraphe 2.4, chap. III. On peut même appliquer l'algorithme *chirp transform* pour des transformées dont la longueur N est un nombre premier, alors que dans ce cas il est impossible de réduire le temps de calcul par une approche FFT ! Bien sûr, cet algorithme nécessite un certain nombre de calculs supplémentaires, entre autres :

- ajout de zéros pour transformer la convolution acyclique en convolution circulaire. En fait, nous allons calculer des FFT de longueur $M = 2^k$ juste après $2N - 1$.
- calcul de deux FFT (voire trois en prenant en compte le vecteur g) pour calculer une convolution circulaire.

Cependant, dans le cas où l'on doit calculer une TFD de longueur N (et où on ne peut pas remplacer ces calculs par une transformée plus grande), cet algorithme constitue une alternative avantageuse par rapport au calcul naïf. Toute fois, il faut garder à l'esprit que dans bon nombre d'applications, on peut se contenter de calculer une transformée aux fréquences $\{k/N'\}_{k=-N'/2}^{N'/2-1}$ plutôt que $\{k/N\}_{k=-N/2}^{N/2-1}$, et donc que cette approche est à proscrire !

***Remarque* 3.3.** Le pire des cas pouvant se présenter pour l'algorithme *chirp* pour le calcul d'une TFD est $2N - 1 = 2^p + 1$. On doit en effet calculer 3 FFT de taille $2^{p+1} \approx 4N$ pour le calcul de la convolution (on a $N' = 2^{p+1}$ et il faut doubler la taille car la convolution n'est pas circulaire). On voit donc que l'on réalise environ 12 fois plus de calculs que pour une FFT de taille 2^p ...

4 Transformée de Fourier fractionnaire

Dans cette section, nous allons étudier la *transformée de Fourier fractionnaire.* Il s'agit simplement de considérer des fréquences intermédiaires lors de l'évaluation de la somme qui définit la TFD. Certes, on perd de nombreuses propriétés de la transformée de Fourier discrète (convolution, inversion, etc.), puisque l'on n'utilise plus les caractères exponentiels $e_n : k \mapsto e^{\frac{2i\pi}{N}kn}$. Cependant, nous allons voir que l'on dispose d'un algorithme de calcul rapide, ce qui rend cette transformée simple à utiliser. Une présentation relativement complète de la transformée de Fourier fractionnaire est faite dans [6].

4.1 Définition et algorithme de calcul

Voici la définition, très naturelle, de cette nouvelle transformée.

Définition 4.1 (Transformée de Fourier fractionnaire). Soit $\alpha \in \mathbb{R}$. On définit la *transformée de Fourier fractionnaire* $G(f,\alpha)$ d'un vecteur $f \in \mathbb{C}^N$ par

$$\forall k \in \{0,\ldots,N-1\}, \quad G(f,\alpha)[k] \stackrel{\text{déf.}}{=} \sum_{n=0}^{N-1} f[n] e^{-\alpha \frac{2i\pi}{N}kn}.$$

***Remarque* 4.2. (Lien avec la TFD).** On constate que si $\alpha = 1$, on retrouve la transformée de Fourier discrète. Pour $\alpha = -1$, on retrouve la transformée de Fourier discrète inverse (à un facteur $1/N$ près). C'est dans ce sens que la transformée fractionnaire généralise la TFD usuelle.

Dans le but de construire un algorithme de calcul, nous allons faire le lien avec la transformée en Z, définie par l'équation (3.1). On constate en effet que dans le cas où $z = e^{\frac{2i\alpha\pi}{N}}$, les deux transformées coïncident. En utilisant l'algorithme chirp de transformation en Z, on va donc pouvoir calculer la transformée de Fourier fractionnaire de façon rapide.

La transformée de Fourier fractionnaire n'a pas vraiment de signification intuitive simple. Nous verrons au prochain paragraphe qu'elle permet de calculer des valeurs intermédiaires de la transformée de Fourier d'un signal discret. Elle va ainsi se révéler efficace pour analyser des signaux dont la périodicité est inconnue. Cependant, la transformée de Fourier fractionnaire n'est pas ce que l'on pourrait appeler une transformée partielle, comme nous avons pu en définir au paragraphe 5.2, chap. III, et à l'exercice III.9. En effet, la composée de deux transformées fractionnaires avec $\alpha = \frac{1}{2}$ ne redonne pas la transformée de Fourier classique.

La figure 5.8 montre les transformées de Fourier fractionnaires $G(f,\alpha)$ d'une fonction en escalier pour plusieurs valeurs de α. L'exercice V.11 propose de calculer les transformées de Fourier fractionnaires d'une image.

4.2 Analyse de signaux à périodicité non entière

La transformée de Fourier discrète est un outil puissant pour analyser des données physiques récoltées par des capteurs ou d'autres méthodes plus ou moins sophistiquées. Cependant, toute cette analyse faite à l'aide de la transformée de Fourier suppose que le

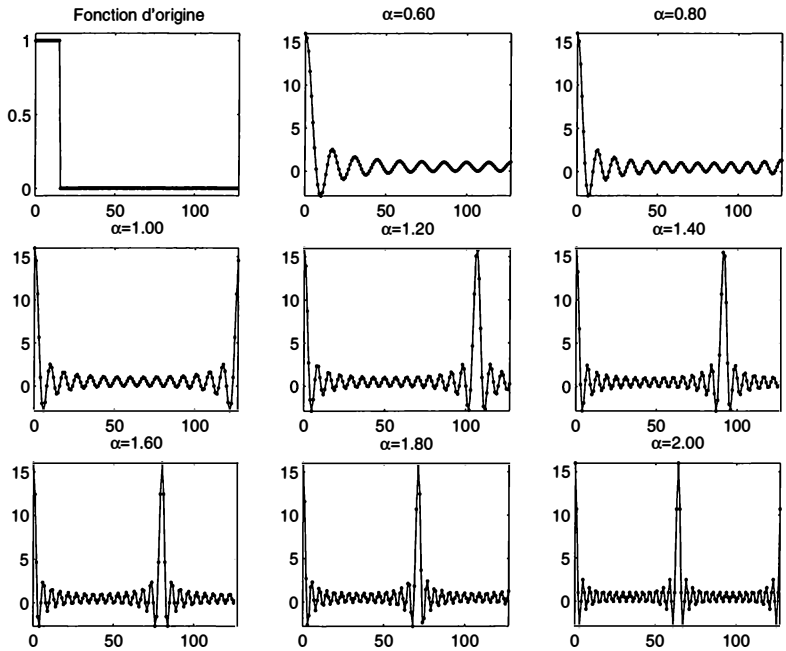

FIG. 5.8 – *Transformées de Fourier fractionnaires d'une fonction 1D*

signal enregistré est périodique, et surtout que sa période est un diviseur de la longueur sur laquelle on a échantillonné le signal. Cependant, dans la majeure partie des applications, on est loin d'être en mesure de connaître *a priori* cette période. Dans certains cas, on dispose d'informations sur la valeur de cette période. Un bon exemple est celui des données météorologiques. On sait que la période de rotation de la terre autour du soleil est de 365,2422 jours. Cependant, même dans ce cas favorable, les données acquises le sont au rythme d'une fois par jour, soit 365 données par an. En conséquence, le spectre du signal va être anormalement compliqué, en tout cas beaucoup plus que si on avait pu obtenir exactement 365,2422 échantillons par an.

On est donc confronté à un double problème.

- Comment déterminer, à partir d'un spectre donné, la vraie période du signal ?
- Une fois cette période connue, comment modifier le spectre d'origine pour qu'il corresponde à des données échantillonnées suivant la bonne période ?

Il nous faut construire un algorithme qui automatise ces deux tâches, et fasse les calculs de façon rapide. Nous allons voir que l'utilisation de la transformée de Fourier fractionnaire et de son algorithme rapide permet de résoudre ce problème.

Pour avoir une idée sur la façon de rechercher la période, il est intéressant d'étudier ce qui se passe sur un échantillon monochromatique, c'est-à-dire sur une sinusoïde. Soit le signal

$$f = \left\{ e^{\beta k \frac{2i\pi}{N}} \right\}_{k=0}^{N-1},$$

où β est un nombre réel. Si β n'est pas un entier, on est en présence d'un signal que l'on sait périodique (de période N/β), mais dont l'échantillonnage ne reflète pas la périodicité. La figure 5.9 montre le spectre obtenu pour $\beta = 4.63$. On y a tracé à la fois la transformée

de Fourier discrète et la transformée de Fourier continue (qui est approchée en ajoutant un nombre conséquent de zéros à la fin du signal avant de calculer la TFD). Bien que le

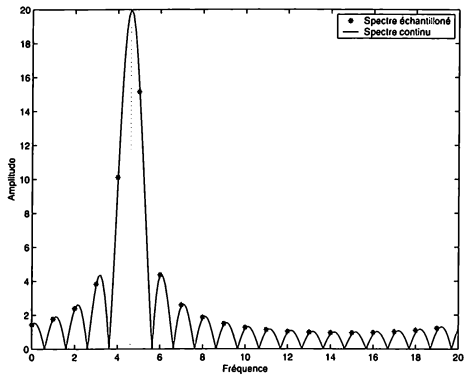

FIG. 5.9 – *Spectre d'une sinusoïde mal échantillonée*

spectre ne soit pas égal à celui que devrait avoir une onde monochromatique (on devrait obtenir une seule raie placée à la fréquence 4.63), il présente un maximum à partir duquel les valeurs de la transformée décroissent. Sur le dessin, il est facile de déterminer la valeur exacte de l'abscisse de ce maximum, qui correspond à la fréquence cherchée et qui nous permet de déterminer la période. Cependant, il est beaucoup trop coûteux de calculer la transformée de Fourier continue pour avoir une bonne valeur approchée de la période. Sans connaître cette transformée de Fourier continue, on est néanmoins en mesure de déterminer la période à une unité près. Ici nous voyons que cette fréquence cherchée est comprise entre 4 et 5. Notons b l'entier immédiatement inférieur à β. Pour calculer avec plus de précision β, nous allons calculer de façon plus fine le spectre de f dans l'intervalle $[b, b+1]$. Soit donc δ un pas de subdivision. On cherche la valeur de la transformée de Fourier de f aux fréquences intermédiaires $b, b+\delta, \ldots, b+m\delta \geq b+1$, ce qui revient à faire le calcul :

$$\forall k \in \{0, \ldots, m\}, \quad \widehat{f}(b+k\delta) = \sum_{n=0}^{N-1} f[n] e^{-\frac{2i\pi}{N} n(b+k\delta)} = G(\widetilde{f}, \delta)[k],$$

où $\widetilde{f}$ est le vecteur

$$\widetilde{f} = \left\{ f[n] e^{-\frac{2i\pi}{N} nb} \right\}_{n=0}^{N-1}.$$

La figure 5.10 (a) montre un signal périodique (bruité) dont la longueur d'échantillonnage n'est malheureusement pas choisie multiple de la période. A la figure (b), on peut voir le spectre de ce signal, qui présente un pic en $b = 2$. La figure (c) réalise un zoom sur la fenêtre fréquentielle $[2, 3]$, et l'on peut voir précisément que $\beta = 2.23$.

Maintenant que nous avons calculé avec précision la fréquence β qui détermine la période, nous aimerions modifier le spectre de manière à ce que cette fréquence β prenne la place d'une fréquence effectivement calculée par la TFD, en l'occurrence la fréquence b. On espère ainsi que les coefficients vont décroître plus vite, comme c'est le cas avec une onde monochromatique bien échantillonnée. Posons $\alpha = \frac{b}{\beta}$, qui est légèrement plus petit que 1. Notons r l'entier le plus proche de $N\alpha$. Pour que la fréquence β devienne la fréquence b, il nous faut calculer une transformée de Fourier en multipliant les fréquences par $\frac{1}{\alpha}$, ce

qui conduit à calculer

$$\forall k \in \{0,\dots,r-1\}, \quad F_k \overset{\text{déf.}}{=} \widehat{f}\left(\frac{k}{\alpha}\right) = \sum_{n=0}^{r-1} f[n] e^{-\frac{2i\pi}{N\alpha}kn} = G\left(f, \frac{r}{N\alpha}\right)[k],$$

où il faut faire attention à compléter le vecteur f par des zéros pour qu'il atteigne la longueur r. Dans le cas où $N\alpha = r$ est un entier, et que le signal f est monochromatique comme nous l'avons déjà défini, on obtient

$$\forall k \in \{0,\dots,r-1\}, \quad \widehat{f}\left(\frac{k}{\alpha}\right) = \sum_{n=0}^{r-1} e^{\frac{2i\pi}{N}\beta n} e^{-\frac{2i\pi}{r}nk} = r\delta_b^k.$$

On obtient donc bien le résultat voulu, à savoir une correction parfaite du spectre. Si $N\alpha$ n'est pas un entier, la correction n'est toute fois pas parfaite ; on peut calculer l'erreur obtenue pour une onde monochromatique :

$$\forall k \in \{0,\dots,r-1\}, k \neq b, \quad F_k \overset{\text{déf.}}{=} \left|\widehat{f}\left(\frac{k}{\alpha}\right)\right| = \left|\frac{1 - e^{-\frac{2i\pi}{N\alpha}n(b-k)}}{1 - e^{-\frac{2i\pi}{N\alpha}(b-k)}}\right| = \left|\frac{sin\left(\frac{\pi s(b-k)}{N\alpha}\right)}{\sin\left(\frac{\pi(b-k)}{N\alpha}\right)}\right|,$$

où l'on a noté $s \overset{\text{déf.}}{=} r - N\alpha$. En notant que $|s| < \frac{1}{2}$, on voit que

- on a $F_b = r$, comme dans le cas où $N\alpha$ est entier,
- lorsque k est voisin de b, au premier ordre, on a $|F_k| \equiv |s|$ qui est borné par $\frac{1}{2}$, donc nettement plus petit que F_b,
- Lorsque k est loin de b en revanche, la situation est moins favorable (le dénominateur de $|F_k|$ peut devenir petit). Cependant, en utilisant le fait que

$$\forall x \in [0,\pi/2], \quad \frac{2x}{\pi} \leqslant \sin(x) \leqslant x,$$

on montre que $|F_k|$ reste majoré par $\frac{N}{\pi\beta}$ ou $\frac{N}{\pi(N-\beta)}$.

Dans le cas où le signal n'est pas monochromatique, la décroissance sera bien entendu plus faible, et l'amélioration moins visible. La figure 5.10 (d) montre l'effet de cet ajustement de fréquence, et on observe une nette diminution des valeurs de la transformée hors du pic à la fréquence 2. On a obtenu une représentation fréquencielle de la fonction analysée beaucoup plus efficace et apte à être utilisée pour des traitements ultérieurs.

5 Exercices

Exercice V.1 (Vecteurs propres et transformée de Hartley). Quelles sont les valeurs propres de la transformée de Hartley définie à la section 1 ? En s'inspirant de la construction du paragraphe 5.2, chap. III, proposer une méthode pour obtenir des vecteurs propres pour chacune des valeurs propres. En s'inspirant de ce qui a été fait au paragraphe 5.2, chap. III, en déduire la construction d'une transformée de Hartley intermédiaire $\mathscr{H}^\lambda$. La figure 5.11 montre différentes transformées intermédiaires. On pourra faire la comparaison avec la figure 3.9, chap. III.

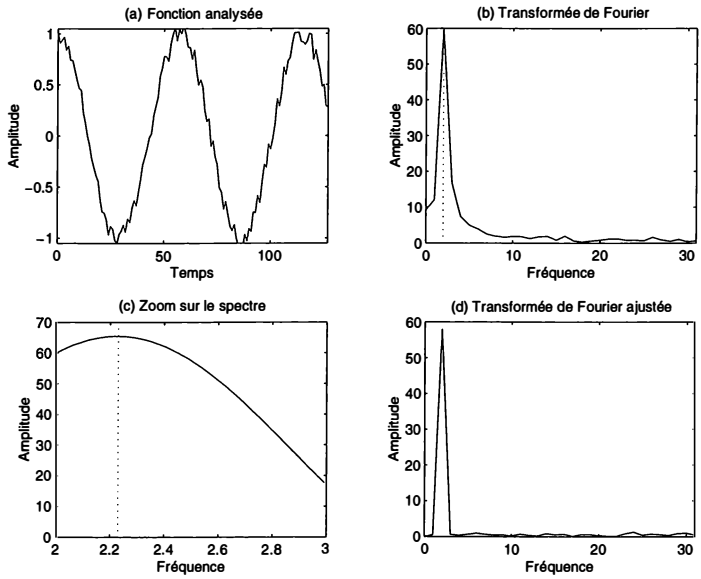

FIG. 5.10 – *Ajustement du spectre par transformée de Fourier fractionnaire*

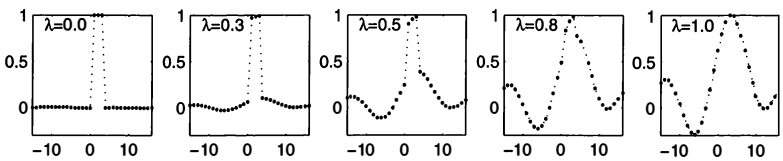

FIG. 5.11 – *Parties réelles des transformées intermédiaires $\mathscr{H}^{\lambda}(f)$ pour $\lambda \in [0,1]$*

Exercice V.2 (Transformée de Hartley généralisée). En remarquant que

$$\mathrm{cas}(x) = \sqrt{2}\cos\left(x - \frac{\pi}{4}\right), \tag{5.1}$$

on peut définir, pour $f \in \mathbb{R}^N$, une transformée de Hartley généralisée par

$$\forall n \in \{0,\ldots,N-1\}, \quad \mathscr{H}_{\lambda}(f)[n] \stackrel{\text{déf.}}{=} \sum_{k=0}^{n-1} f[k]\cos\left(\frac{2\pi}{N}nk + \lambda\right),$$

où λ est un paramètre réel dans $[0,\pi[$. Pour $\lambda \notin \left\{0, \frac{\pi}{2}\right\}$, montrer que cette transformation est bijective, et que son inverse est donné par

$$(\mathscr{H}_{\lambda})^{-1} = \frac{2}{\sin(2\lambda)}\mathscr{H}_{\frac{\pi}{2}-\lambda}. \tag{5.2}$$

La figure 5.12 montre la transformée généralisée du signal triangulaire de la figure 5.1 (gauche), pour λ variant entre 0 et 2π.

Exercice V.3 (Interpolation et transformée de Hartley). A l'exercice III.6, nous avons vu comment on peut interpoler un signal échantillonné grâce à la transformée de Fourier discrète. Toujours en utilisant la technique de zero padding, expliquer pourquoi la transformée de Hartley discrète permet aussi d'interpoler des valeurs échantillonnées. Montrer qu'en fait les interpolations obtenues par Fourier et par Hartley sont les mêmes.

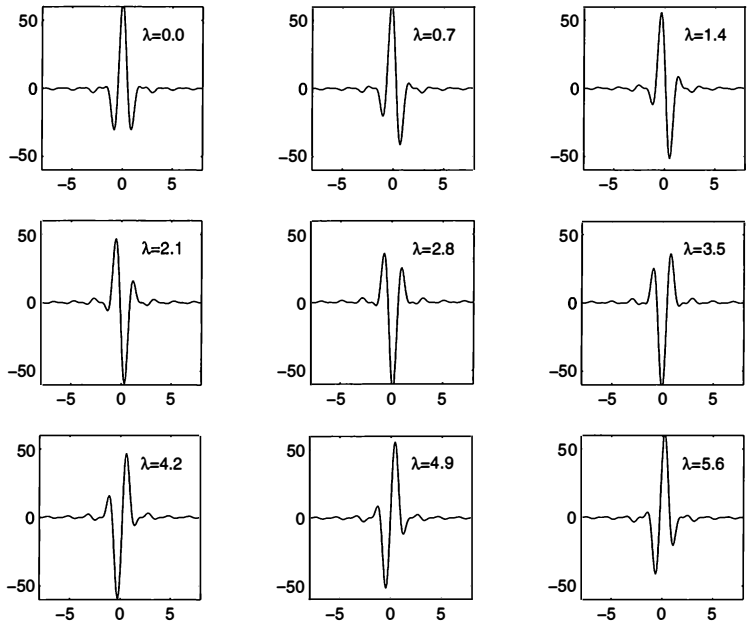

FIG. 5.12 – *Transformée de Hartley généralisée pour différentes valeurs de* λ

Exercice V.4 (Transformée de Hartley 2D). Soit $f \in \mathbb{R}^{N_1 \times N_2}$. On définit sa transformée de Hartley bidimensionnelle $\mathscr{H}(f)$ par

$$\mathscr{H}(f)[n_1,n_2] \overset{\text{déf.}}{=} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} f[k_1,k_2] \operatorname{cas}\left(\frac{2\pi}{N_1} k_1 n_1\right) \operatorname{cas}\left(\frac{2\pi}{N_2} k_2 n_2\right),$$

où $n_1 \in \{0,\ldots,N_1-1\}$ et $n_2 \in \{0,\ldots,N_2-1\}$. Montrer que l'on a la formule d'inversion suivante :

$$f = \frac{1}{N_1 N_2} \mathscr{H}(\mathscr{H}(f)).$$

Comment calculer rapidement cette transformée à l'aide de l'algorithme FHT ?

Exercice V.5 (Transformée de Hartley et corps finis). Etendre la définition de la *transformée de Hartley* dans le cadre des corps finis (puis des anneaux dans lesquels on dispose d'une racine $n^{\text{ième}}$ principale). Si ζ représente une racine $n^{\text{ième}}$ de l'unité, on pourra utiliser $\frac{\zeta^2+1}{2\zeta}$ pour remplacer $\cos\left(\frac{2\pi}{n}\right)$. Ecrire l'algorithme *FHT* correspondant.

Exercice V.6 (Transformation chirp et matrices de Toeplitz). Dans cet exercice, on reprend la description de l'algorithme chirp pour le calcul de la transformée en Z, dans le but de trouver une formulation matricielle. Rappelons la définition de l'algorithme chirp. Il consister à calculer la transformée en Z d'un vecteur $f \in \mathbb{C}^N$ par l'intermédiaire d'un vecteur $\{y[n]\}_{n=0}^{N-1}$ défini par

$$\forall n \in \{0,\ldots,N-1\}, \quad y[n] \overset{\text{déf.}}{=} z^{\frac{n^2}{2}} \mathscr{G}_z(f)[n] = \sum_{n=0}^{N-1} h[k] g[n-k],$$

où l'on a noté

et

$$\forall k \in \{-N+1, N-1\}, \quad g[k] \stackrel{\text{déf.}}{=} z^{-\frac{k^2}{2}}.$$

1. Montrer que y peut se calculer comme un produit matriciel $y = Gh$, où G est ce que l'on nomme une matrice de *Toeplitz*, c'est-à-dire une matrice constante le long de chacune de ses diagonales. Par exemple, montrer que pour $N = 3$, on a

$$G = \begin{pmatrix} g[0] & g[1] & g[2] \\ g[1] & g[0] & g[1] \\ g[2] & g[1] & g[0] \end{pmatrix}.$$

2. En toute généralité, une matrice de Toeplitz T de taille $m \times n$ se note

$$T \stackrel{\text{déf.}}{=} \begin{pmatrix} t_{m-1} & t_m & \cdots & t_{m+n-2} \\ t_{m-2} & t_{m-1} & \cdots & t_{m+n-3} \\ \vdots & & & \vdots \\ t_0 & & & t_{m-1} \end{pmatrix}.$$

   Elle est donc entièrement déterminée par sa première colonne, que l'on note sous la forme $t_c \stackrel{\text{déf.}}{=} (t_0, \ldots, t_{n-1})^{\mathrm{T}}$ et sa première ligne notée $t_l \stackrel{\text{déf.}}{=} (t_m, \ldots, t_{m+n-2})^{\mathrm{T}}$ (on ne prend pas l'élément t_{m-1}). On considère le vecteur

$$c \stackrel{\text{déf.}}{=} (t_c, 0, \ldots, 0, t_l)^{\mathrm{T}} \in \mathbb{C}^M.$$

   où M est la puissance de deux immédiatement après $n + m - 1$. On note C matrice circulante associée à c, comme définie à l'exercice III.5. Où peut-on retrouver la matrice T à l'intérieur de la matrice C ? Comment calculer un produit Tx, où $x \in \mathbb{C}^n$, en utilisant la matrice C ? En déduire un algorithme permettant de calculer Cx rapidement, à l'aide de l'algorithme FFT.

3. Appliquer la construction précédente à la matrice G. Montrer par exemple que dans le cas $N = 3$, on obtient la matrice C suivante :

$$C = \begin{pmatrix} g[0] & g[1] & g[2] & 0 & 0 & 0 & g[2] & g[1] \\ g[1] & g[0] & g[1] & g[2] & 0 & 0 & 0 & g[2] \\ g[2] & g[1] & g[0] & g[1] & g[2] & 0 & 0 & 0 \\ 0 & g[2] & g[1] & g[0] & g[1] & g[2] & 0 & 0 \\ 0 & 0 & g[2] & g[1] & g[0] & g[1] & g[2] & 0 \\ 0 & 0 & 0 & g[2] & g[1] & g[0] & g[1] & g[2] \\ g[2] & 0 & 0 & 0 & g[2] & g[1] & g[0] & g[1] \\ g[1] & g[2] & 0 & 0 & 0 & g[2] & g[1] & g[0] \end{pmatrix}.$$

   En déduire un algorithme permettant de calculer y, puis $\mathscr{G}_z(f)$ rapidement.

Exercice V.7 (Méthodes de quadrature et filtres récursifs). On considère un circuit intégrateur, qui relie les tensions d'entrée $x(t)$ et de sortie $y(t)$ par l'équation

$$y(t) = \int_0^t x(\tau) \mathrm{d}\tau.$$

Quelle est la fonction de transfert de ce filtre dans le domaine de Laplace ? On souhaite, à partir de ce filtre analogique, créer un filtre récursif digital. On envisage les méthodes de

quadrature suivantes, pour une fonction $f : \mathbb{R} \to \mathbb{R}$ continue :

$$(M_0) \qquad \int_0^1 f(t)\mathrm{d}t \approx f(0), \tag{5.3}$$

$$(M_1) \qquad \int_0^1 f(t)\mathrm{d}t \approx \frac{1}{2}(f(0)+f(1)), \tag{5.4}$$

$$(M_2) \qquad \int_0^1 f(t)\mathrm{d}t \approx \frac{1}{6}f(0)+\frac{2}{3}f(1/2)+\frac{1}{6}f(1). \tag{5.5}$$

Ces méthodes sont appelées respectivement méthode des *rectangles à gauche*, méthode des *trapèzes*, et méthode de *Simpson*. En considérant une discrétisation des signaux x et y de pas Δ, donner les équations de récurrence obtenues en employant chacune des trois méthodes. Quelles sont les fonctions de transfert associées (pour la transformée en Z) ?

Exercice V.8 (Spline et filtres récursifs). On reprend les notations de l'exercice IV.12. Il s'agit d'expliquer comment on peut calculer les coefficients d'interpolation $c[k]$ par l'intermédiaire d'un filtre récursif. Nous allons essentiellement nous concentrer sur l'exemple des splines cubiques, et nous laissons au lecteur le soin de s'entraîner sur d'autres exemples, puis de généraliser la méthode. On pourra pour se faire s'aider d'une expression de β_d^n, et se référer à l'article de UNSER [74].

1. Calculer β_d^3, puis montrer que sa transformée en Z vaut
$$\mathscr{Z}(\beta_d^3)(z) = \frac{z+4+z^{-1}}{6}.$$
2. On rappelle que l'on a $c = \Phi_d^3 * u_d$, où Φ_d^3 est défini par $\widehat{\Phi_d^3} = 1/\widehat{\beta_d^3}$. Décomposer la fraction rationnelle $\mathscr{Z}(\Phi_d^3)$ en éléments simples. Comment peut-on calculer les coefficients c de l'interpolation indirecte en utilisant deux filtres récursifs ?
3. On suppose que l'on connaît le signal $u_d[k]$ pour $k \in \{0,\dots,K-1\}$. Montrer que l'on peut organiser les calculs de c de la manière suivante :
$$\begin{cases} \forall k \in \{1,\dots,K-1\}, & c^+[k] = u_d[k] + b_1 c^+[k-1] \\ \forall k \in \{0,\dots,K-2\}, & c^-[k] = u_d[k] + b_1 c^-[k+1] \\ \forall k \in \{0,\dots,K-1\}, & c[k] = b_0(c^+[k] + c^-[k] - u_d[k]) \end{cases}$$
On précisera les valeurs des constantes b_0 et b_1. Quelles valeurs donner à $c^+[0]$ et $c^-[K-1]$? On pourra par exemple faire en sorte que le signal produit puisse être prolongé par symétrie miroir en $K-1$ et 0 (pour éviter des discontinuités).
4. Reprendre les deux questions précédentes en exploitant une décomposition en produit de la forme
$$\mathscr{Z}(\Phi_d^3) = \frac{A}{(1-\alpha z^{-1})(1-\alpha z)}.$$

Exercice V.9 (Transformation Chirp et corps fini). Soit p un nombre premier. On note g un générateur du groupe $\mathbb{F}_p^*$.
Une fonction $f : \{0,\dots,p-1\} \to \mathbb{C}$ sera considérée comme une fonction définie sur $\mathbb{F}_p$. Montrer que l'on peut écrire la transformée de Fourier de f de la manière suivante :

$$\forall b \in \{0,\dots,p-2\}, \quad \widehat{f}(g^{-b}) = f(0) + \sum_{a=0}^{p-2} f(g^a)\omega_p^{-g^{a-b}},$$

où $\omega_p = e^{\frac{2i\pi}{p}}$. En déduire que l'on peut calculer la transformée de Fourier de f à l'aide d'une convolution sur $\mathbb{Z}/(p-1)\mathbb{Z}$. Préciser les vecteurs mis en jeu, et implémenter cet algorithme.

Exercice V.10 (Calculs approchés par transformée de Fourier fractionnaire). Soit une fonction continue $f : \mathbb{R} \to \mathbb{R}$ que l'on suppose à support dans $[-a/2, a/2]$. On dispose d'un échantillonnage régulier $f[k] = f(x_k)$ avec $x_k = f((k - N/2)a/N)$ pour $k = 0, \ldots, N-1$.

1. On souhaite évaluer la transformée de Fourier continue de f autour d'une fréquence ζ, plus précisément aux points $y_k = \zeta + \frac{2\pi}{a}(k - N/2)\gamma$, où γ représente la précision voulue. Si l'on souhaite effectuer le calcul des $\widehat{f}(x_k)$ à l'aide de l'algorithme FFT, quelle est la taille de la transformée à calculer (on supposera que $\gamma \in \mathbb{Q}$) ? Montrer que l'on peut effectuer ce calcul de façon plus économique à l'aide de la transformée de Fourier fractionnaire.
2. On souhaite maintenant augmenter la précision des calculs. En utilisant la méthode de quadrature de Simpson (exercice V.7, méthode (M_2)), expliquer comment il faut modifier la méthode de la question précédente.

Exercice V.11 (Transformée de Fourier fractionnaire d'une image). Proposer une transformée de Fourier fractionnaire 2D qui étend la transformée 1D par produit tensoriel (on pourra s'aider de la construction de la transformée de Hartley 2D, exercice V.4). Ecrire la fonction MATLAB qui correspond, et tester la transformée sur plusieures images. La figure 5.13 présente les transformées de la fonction indicatrice d'un disque.

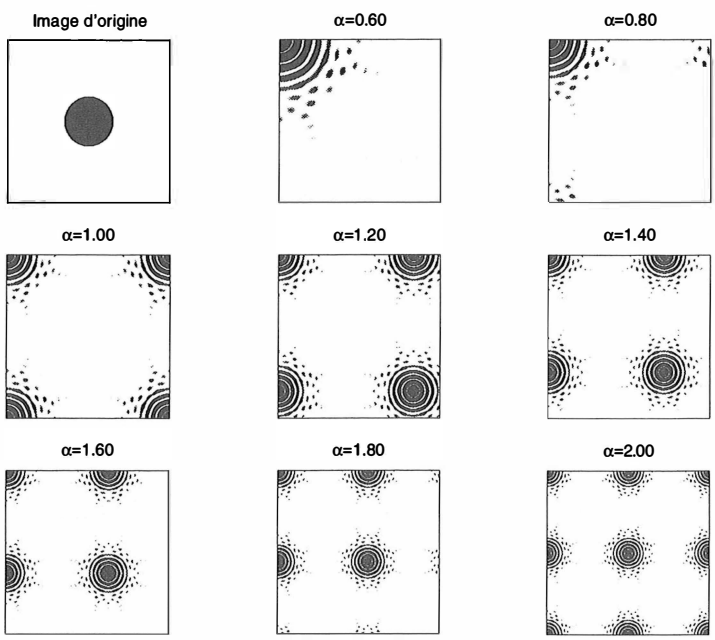

FIG. 5.13 – *Transformées de Fourier fractionnaires d'une fonction 2D*

Chapitre VI
Transformée de Fourier à valeurs dans un corps fini

> Many people regard arithmetic as a trivial thing that children can learn and computer do, but we will see that artihmetic is a fascinating topic with many interesting facets. It is important to make a thorough study of efficient methods for calculating with numbers, since arithmetic underlines so many computer application.
>
> D. E. KNUTH [39] (1997)

Nous avons déjà rencontré à de nombreuses reprises les corps finis, en particulier au chapitre II. Cependant, nous nous sommes bornés à les exploiter en tant que domaines de *départ* des fonctions que l'on souhaitait analyser. Pourtant, il est très fréquent de manipuler des données à valeurs dans un ensemble fini, que l'on peut souvent munir d'une structure de corps fini. L'exemple le plus frappant est celui des données binaires, qui peuvent être modélisées par des fonctions à valeurs dans $\mathbb{F}_2 \simeq \{0, 1\}$. Dans ce chapitre, nous allons présenter de nombreuses situations similaires, et nous verrons comment les outils de Fourier s'étendent naturellement à de telles structures de corps.

1 Calculs sur un corps fini

Depuis le début de l'exposé, nous nous sommes limités à l'étude des fonctions à valeurs dans le corps $\mathbb{C}$ des complexes, et nous nous sommes particulièrement intéressés aux morphismes de G (groupe fini abélien) dans le groupe multiplicatif $\mathbb{C}^*$. Cependant, la majeure partie des résultats reste valide si on considère les morphismes de G dans un corps commutatif quelconque. Dans cette section, nous allons étudier de plus près le cas des corps finis. Non seulement nous allons reprendre les résultats déjà énoncés aux chapitres précédents, mais nous allons les particulariser et expliquer pourquoi et comment effectuer les calculs dans un corps fini.

Il s'agit d'effectuer nos calculs modulo un nombre premier p. Il faut se garder de croire que nous allons nous restreindre au seul corps $\mathbb{F}_p \stackrel{\text{déf.}}{=} \mathbb{Z}/p\mathbb{Z}$. Par exemple, au paragraphe 1.4, nous allons nous placer dans un corps plus grand, du type $\mathbb{F}_{p^r}$, où $r \geqslant 1$, pour définir une transformée d'une longueur arbitraire.

1.1 Transformée de Fourier sur un corps fini

On se fixe donc un entier premier p. Tous nos calculs seront effectués modulo p. Pour construire une transformée à valeurs dans un corps fini, nous avons besoin d'une *racine primitive* $n^{\text{ième}}$ de l'unité. Précisons un peu ce que cela signifie.

Définition 1.1 (Racine primitive). Soit K un corps. Un élément $\zeta \in K$ est appelé *racine de l'unité* si c'est un élément d'ordre fini de K^*, c'est-à-dire s'il existe un entier $s > 0$ tel que $\zeta^s = 1$.
Un élément d'ordre n de K^* est appelé racine primitive $n^{\text{ième}}$ de l'unité, ce qui signifie que $\zeta^n = 1$ et que pour tout s tel que $0 < s < n$, on a $\zeta^s \neq 1$.

Dans un corps fini K de cardinal $q = p^r$, tout élément de K^* est nécessairement une racine $(q-1)^{\text{ième}}$ de l'unité. Bien sûr, l'existence d'une racine $n^{\text{ième}}$, pour un n quelconque, n'est pas assurée. En effet, l'ordre de tout élément de K^* est un diviseur de $q-1$. Réciproquement, si $n|q-1$, c'est-à-dire $-1 = nk$, alors, si on note ω un générateur de K^*, $\zeta = \omega^k$ est une racine $n^{\text{ième}}$ primitive.

Pour simplifier, nous allons supposer que $q = p$ est un nombre premier. On admet aussi que l'on dispose de ζ, une racine $n^{\text{ième}}$ primitive de l'unité sur $\mathbb{F}_p$. Il est alors très simple de définir la transformée de Fourier à valeurs dans le corps $\mathbb{F}_p$.

Définition 1.2 (Transformée sur $\mathbb{F}_p$). Pour un vecteur $f \in (\mathbb{F}_p)^n$, on définit la transformée de Fourier :

$$\forall j \in \{0,\dots,n-1\}, \quad \mathscr{F}(f)[j] = \widehat{f}[j] \stackrel{\text{déf.}}{=} \sum_{k=0}^{n-1} f[k]\zeta^{-kj}. \tag{1.1}$$

La transformée de Fourier sur $\mathbb{F}_p$ possède exactement les mêmes propriétés que la transformée de Fourier classique ; les voici brièvement rappelées.

Proposition 1.3 (Propriétés). *$\mathscr{F}$ est un isomorphisme d'algèbre de $((\mathbb{F}_p)^n, *)$ dans $((\mathbb{F}_p)^n, \cdot)$, où l'on a noté $*$ le produit de convolution circulaire et $\cdot$ le produit composante par composante. Son inverse est donné par*

$$\mathscr{F}^{-1}(f)[n] = n^{-1} \sum_{k=0}^{n-1} f[k]\zeta^{kn}. \tag{1.2}$$

La transformée de Fourier modulo p présente un avantage certain : tous les calculs se font avec des entiers (certes modulo p, mais au final, on réalise toujours des additions et multiplications d'entiers). Il n'y a donc pas d'erreur numérique susceptible d'entacher les résultats des calculs. En revanche, dans des calculs nécessitant une grande précision, l'utilisation d'une FFT classique peut conduire à des erreurs. Lorsque l'on veut par exemple calculer le produit de deux grands entiers (en utilisant la technique présentée au paragraphe 5.4, chap. IV), il est très important de minimiser les erreurs numériques, puisque l'on veut, au final, retrouver des valeurs entières (nous allons en fait arrondir à l'entier le plus proche). Par exemple, dans [5], l'auteur explique qu'en double précision, passé 10 millions de décimales, l'algorithme de FFT, utilisé pour des calculs de produits d'entiers, donnait de mauvais résultats à cause des erreurs d'arrondi. C'est pourquoi les algorithmes de calculs sur les corps finis (et plus généralement sur $\mathbb{Z}/m\mathbb{Z}$) sont au cœur des systèmes informatiques nécessitant des calculs entiers de haute précision.

Au delà de tous ces avantages, il faut garder à l'esprit que le résultat obtenu par transformée sur un corps fini n'a plus aucune signification « physique ». En effet, la transformée

de Fourier à valeurs dans $\mathbb{C}$ représente une approximation de la transformée de Fourier continue sur $\mathbb{R}$ (se référer aux calculs effectués au paragraphe 1.2, chap. IV), ce qui est bien sûr loin d'être le cas pour la transformée sur un corps fini. On peut considérer la transformation effectuée dans les complexes comme un moyen de passer d'une représentation temporelle des données à une représentation fréquentielle, alors qu'il n'y a pas d'interprétation aussi évidente pour la transformation réalisée sur $\mathbb{F}_p$.

1.2 Un cas particulier

Le problème auquel nous sommes confrontés pour construire cette transformée est la recherche d'une racine primitive $n^{\text{ième}}$ de l'unité. Pour bien comprendre les difficultés que l'on rencontre, commençons par montrer un cas où tout se passe bien, mais qui, comme nous allons le voir, est très restrictif.

On suppose que l'on a choisi $n = p-1$. On sait que le groupe multiplicatif $\mathbb{F}_p^* = \mathbb{F}_p - \{0\}$ est un groupe fini cyclique. En conséquence, il possède un élément générateur : notons le ζ. Par définition, on a donc

$$\mathbb{F}_p^* = \{1, \zeta, \zeta^2, \ldots, \zeta^{n-1}\} \quad \text{et } \zeta^n = 1.$$

Nous avons donc exhibé une racine $n^{\text{ième}}$ primitive de l'unité, mais au prix d'un choix particulier pour la valeur de n. Un algorithme naïf pour déterminer un générateur du groupe multiplicatif $(\mathbb{Z}/n\mathbb{Z})^*$ consiste à essayer un par un tous les éléments du groupe. Un algorithme plus efficace est présenté dans le livre de COHEN [18].

1.3 Corps cyclotomiques

Dans le paragraphe précédent, nous avons construit une transformée de Fourier de taille n sur $\mathbb{F}_p$ dans un cas bien précis, ce qui imposait une relation entre les entiers n et p. Cependant, on peut souhaiter choisir indépendamment ces deux paramètres, par exemple dans le cas fréquent où l'on veut réaliser des TFD de taille très grande, beaucoup plus que p. Le problème est qu'il n'y a aucune raison que le corps $\mathbb{F}_p \overset{\text{déf.}}{=} \mathbb{Z}/p\mathbb{Z}$ contienne des racines $n^{\text{ièmes}}$ primitives. Nous allons donc devoir travailler dans une extension de $\mathbb{F}_p$, c'est-à-dire un certain $\mathbb{F}_{p^r}$ pour $r \in \mathbb{N}^*$.

Pour mener à bien la recherche de cette extension, on a besoin d'étudier les facteurs irréductibles du polynôme $X^n - 1$, puisque ce seront les polynômes minimaux d'une éventuelle racine primitive. Rappelons, sans démontration, quels sont ces facteurs irréductibles sur le corps $\mathbb{Q}$.

Théorème et définition 1.4 (Polynômes cyclotomiques). On note Φ_n le $n^{\text{ième}}$ polynôme cyclotomique, défini de la façon suivante :

$$\Phi_n(X) \overset{\text{déf.}}{=} \prod_{k \in (\mathbb{Z}/n\mathbb{Z})^*} \left(X - e^{\frac{2i\pi k}{n}}\right).$$

Les polynômes cyclotomiques vérifient

$$X^n - 1 = \prod_{d|n} \Phi_d(X). \tag{1.3}$$

Ils sont à coefficients dans $\mathbb{Z}$, et de plus, ils sont irréductibles dans $\mathbb{Q}[X]$.

Pour une preuve de ce théorème, ainsi qu'une construction très claire des corps finis, on pourra regarder le livre de DEMAZURE [24].

Exemple 1.5. L'équation (1.3) fournit un algorithme qui permet de déterminer par récurrence les polynômes cyclotomiques, dont voici quelques exemples :

$$\begin{aligned}\Phi_1(X) &= X-1,\\ \Phi_2(X) &= X+1,\\ \Phi_3(X) &= X^2+X+1,\\ \Phi_6(X) &= X^2-X-1.\end{aligned}$$

Le programme MAPLE 1.1 permet de calculer un polynôme cyclotomique. Voir à ce sujet l'exercice VI.1.

Programme 1.1 Calcul d'un polynôme cyclotomique

```
with(numtheory): cyclotomic(105,X);
```

$$1+X-X^8+X^2-X^5-2X^7+X^{35}-X^{28}+X^{48}+X^{46}-X^{43}-2X^{41}-X^{40}-X^{39}+X^{36}+X^{34}$$
$$+X^{33}+X^{31}-X^{26}-X^{24}-X^{22}-X^{20}+X^{17}+X^{16}+X^{15}+X^{14}+X^{12}-X^9-X^6+X^{47}$$
$$-X^{42}+X^{32}+X^{13}$$

Les polynômes Φ_n étant à coefficients dans $\mathbb{Z}$, on peut donc les réduire modulo p et les regarder comme éléments de $\mathbb{F}_p[X]$. Bien qu'ils soient irréductibles sur $\mathbb{Q}$, il n'y a aucune raison qu'ils le soient encore sur $\mathbb{F}_p$. Plus précisément, c'est le théorème suivant qui va nous permettre de construire l'extension de corps dont on a besoin.

Proposition 1.6 (Cyclotomie sur $\mathbb{F}_p$). *On suppose* $\operatorname{pgcd}(n,p)=1$. *Soit r l'ordre de p dans le groupe $(\mathbb{Z}/n\mathbb{Z})^*$, c'est-à-dire le plus petit entier t tel que $p^t=1$ modulo n. Alors le polynôme cyclotomique $\Phi_n(X)$ se décompose dans $\mathbb{F}_p[X]$ en produit de polynômes irréductibles de degré r, tous différents.*

Démonstration. Comme n est premier avec p, le polynôme X^n-1 est premier avec sa dérivée nX^{n-1}, donc il n'a pas de racine multiple dans une extension. Avec la relation (1.3), on en déduit donc que le polynôme Φ_n n'a pas de facteur multiple. Il suffit donc de montrer que tout facteur irréductible de Φ_n est de degré r.

Soit donc P un facteur irréductible de degré s. On note $K=F_p[X]/(P)$ le corps résiduel. Son cardinal est $|K|=p^s$. L'image ζ de l'indéterminée X dans K est une racine primitive de l'unité, puisque P est un facteur irréductible de X^n-1 et que ζ est racine de P. Comme tout élément x de K^* vérifie $x^{p^s-1}=1$, en particulier, $\zeta^{p^s-1}=1$, et la définition d'une racine primitive $n^{\text{ième}}$ implique que $n|p^s-1$. Par définition de l'ordre r de p dans $(\mathbb{Z}/n\mathbb{Z})^*$, on a donc $r\leqslant s$.

Montrons l'inégalité inverse. Comme $n|p^r-1$, on a $p^r-1=\lambda n$, d'où $\zeta^{p^r-1}=(\zeta^n)^\lambda=1$, donc $\zeta^{p^r}=\zeta$. Notons alors k le sous-corps de K formé des racines de l'équation $X^{p^r}=X$. Il contient ζ qui engendre K^*, donc il est en fait égal à K tout entier. Comme $X^{p^r}-X$ a au plus p^r racines distinctes, on obtient que $|K|=|k|\leqslant p^r$, donc $s\leqslant r$ qui est l'inégalité cherchée. □

***Remarque* 1.7.** Dans le cas où les entiers n et p ne sont pas premiers entre eux, n est un multiple de p, soit $n=mp^t$, où cette fois m est premier avec p. On écrit alors

$$X^n-1=(X^m)^{p^t}-1^{p^t}=(X^m-1)^{p^t}.$$

On peut maintenant appliquer la proposition précédente au polynôme $X^m - 1$ pour trouver une décomposition de $X^n - 1$.

***Remarque* 1.8.** En fait, la proposition 1.6 est encore valable sur un corps du type $\mathbb{F}_q$ où $q = p^t$ (sans modification de la démonstration). Le polynôme Φ_n se décompose dans $\mathbb{F}_q[X]$ en produit de polynômes irréductibles de degré r, où r est l'ordre de q dans $(\mathbb{Z}/n\mathbb{Z})^*$. Il faut faire attention : le fait de regarder le polynôme Φ_n sur $\mathbb{F}_p$ revient à réduire ses coefficients modulo p (et pas modulo q !). Cependant, comme il y a plus d'éléments dans $\mathbb{F}_q$ que dans $\mathbb{F}_p$, il est possible que certains facteurs irréductibles de Φ_n sur $\mathbb{F}_p$ se décomposent sur $\mathbb{F}_q$.

Si on note P un facteur irréductible de Φ_n, le corps $K \overset{\text{déf.}}{=} \mathbb{F}_p[X]/(P)$ est donc l'extension de $\mathbb{F}_p$ que l'on cherche. C'est un corps de cardinal p^r, et on peut le voir comme un espace vectoriel de dimension r sur $\mathbb{F}_p$, simplement en considérant ses éléments comme des polynômes de degré au plus $r-1$. On note généralement $\mathbb{F}_{p^r}$ ce corps, mais il ne faut pas le confondre avec $\mathbb{Z}/p^r\mathbb{Z}$: en effet, dans K, on continue à effectuer l'addition modulo p, et la multiplication est plus complexe puisqu'elle correspond à la multiplication des polynômes modulo P. Pour finir, notons que l'on a bien ainsi trouvé une racine primitive de l'unité. Il suffit de considérer la classe de l'indéterminée X dans K : c'est en fait une racine que l'on notera ζ de P dans K. On peut alors voir le corps K comme $\mathbb{F}_p[\zeta]$, le corps engendré par ζ sur $\mathbb{F}_p$. On nomme ce corps le *corps cyclotomique d'indice n* sur $\mathbb{F}_p$.

Exemple 1.9. On considère le polynôme

$$\Phi_{15}(X) = 1 - X + X^3 - X^4 + X^5 - X^7 + X^8 \in \mathbb{Z}[X].$$

Comme l'ordre de 2 dans $(\mathbb{Z}/15\mathbb{Z})^*$ est 4, Φ_{15} se décompose sur $\mathbb{F}_2$ en produit de 2 polynômes de degré 4 irréductibles :

$$\Phi_{15}(X) = (1 + X + X^4)(1 + X^3 + X^4) \in \mathbb{F}_2[X].$$

Un corps de décomposition de Φ_{15} est donc par exemple

$$\mathbb{F}_2[X]/(1 + X + X^4) \simeq \mathbb{F}_{2^4} \simeq \mathbb{F}_2[\alpha],$$

où l'on a noté α une racine de $1 + X + X^4$ (c'est-à-dire la classe de l'indéterminée X dans le corps quotient $\mathbb{F}_2[X]/(1 + X + X^4)$).

D'une façon pratique, pour construire ce corps, il faut dans un premier temps calculer le polynôme Φ_n (par exemple en utilisant par récurrence la relation (1.3)), puis rechercher un facteur irréductible de Φ_n sur $\mathbb{F}_p$. Pour ce faire, on pourra utiliser MAPLE, toute la démarche étant détaillée dans le premier programme du paragraphe 1, annexe B. Un algorithme permettant de factoriser des polynômes sur un corps fini est l'algorithme de *Berlekamp*. Il est détaillé dans le livre de DEMAZURE [24].

1.4 Transformée sur un corps cyclotomique

Grâce à la décomposition du polynôme Φ_n sur le corps $\mathbb{F}_p$, on a réussi à construire une extension de $\mathbb{F}_p$, notée $K \overset{\text{déf.}}{=} \mathbb{F}_{p^r}$, dans laquelle se trouve une racine $n^{\text{ième}}$ primitive de

l'unité, α. De plus, la connaissance de son polynôme minimal P nous permet de calculer avec les éléments de K (qui sont des vecteurs de r éléments de $\mathbb{F}_p$), puisqu'il suffit d'utiliser la multiplication polynomiale modulo P.

En se plaçant dans le corps K, on définit donc la transformée d'un vecteur $f \in K^n$ par l'équation (1.1). Cette fois, $\mathscr{F}(f)$ est lui aussi un vecteur de K^n. Dans la pratique, on utilise souvent cette transformée pour des données qui sont des éléments de $(\mathbb{F}_p)^n$, mais il faut bien sûr garder en tête que le résultat n'a aucune raison de rester dans $\mathbb{F}_p$. On retrouve le même phénomène que pour le calcul de la TFD (classique) d'un vecteur réel, pour laquelle on est obligé de se placer dans l'extension $\mathbb{C}$ du corps des réels. Rappelons que si on note les éléments de $\mathbb{F}_{p^r}$ comme des vecteurs (plutôt que des polynômes), un élément du corps de base $\mathbb{F}_p$ se distingue par le fait que seule la première composante du vecteur est non nulle.

Cette construction d'un algorithme de transformée rapide pour une longueur N arbitraire sera utilisée à la section 3. Il sera en effet question de décoder certains codes correcteurs à l'aide de la transformée de Fourier. Nous verrons que l'utilisation d'un sur-corps (qui peut paraître pour l'instant un peu artificielle) devient alors très naturelle pour rechercher l'ensemble des codes vérifiant certaines propriétés (on les appelle *codes cycliques*).

1.5 Calculs effectifs

Nous allons pouvoir utiliser la transformée de Fourier à valeurs dans un corps fini pour faire des calculs de convolution. Comme tous les calculs sont effectués modulo p, nous allons devoir choisir un entier p suffisamment grand pour pouvoir récupérer des résultats de calculs dans $\mathbb{Z}$, et non pas modulo p.

Le meilleur exemple de l'utilisation de cette transformée de Fourier est le calcul de produits de grands entiers, écrits en base b. Le paragraphe 5.4, chap. IV, explique comment calculer le produit de deux entiers à l'aide d'une convolution acyclique. On peut déterminer facilement une borne sur la valeur des entrées d'une convolution linéaire de deux entiers représentés en base b :

$$|f \star g[k]| = \left| \sum_{l=0}^{n-1} f[l] g[k-l] \right| \leqslant \sum_{l=0}^{n-1} |f[l]||g[k-l]| \leqslant n(b-1)^2.$$

Pour pouvoir calculer de façon exacte le produit de deux entiers, il faut que $|f \star g[k]| < p$. On aura donc intérêt à choisir $p > n(b-1)^2$.

Pour terminer, il convient de dire quelques mots sur l'implémentation pratique de l'algorithme de calcul de la transformée. Le plus souvent, on va considérer que n est de la forme 2^k, et on va utiliser l'algorithme dichotomique de FFT présenté à la section 2, chap. III. Les détails de l'algorithme restent inchangés. Entre autres, si ζ est la racine d'ordre 2^k cherchée, alors ζ^2 sera la racine d'ordre 2^{k-1}, etc. Bien sûr, il faut remplacer les opérations dans le corps des complexes par des opérations dans le corps fini $K = \mathbb{F}_{p^r}$. Il faut donc choisir une façon de représenter les éléments du corps K, et savoir comment les multiplier. Si on a recherché la racine primitive ζ comme il est indiqué au paragraphe 1.3, on a un moyen très naturel de le faire. En effet, on dispose par la même occasion du polynôme minimal P de ζ, de degré r, et on peut donc représenter les éléments du corps comme des polynômes de degré au plus $r-1$ (c'est-à-dire par des vecteurs de taille r dans la mémoire de l'ordinateur). L'addition s'effectue alors composante par composante, alors

que la multiplication s'effectue comme une multiplication de polynômes modulo P. Cette multiplication se calcule de façon classique, puis en prenant le reste de la division euclidienne par P.

Le programme MAPLE de la section 1, annexe B, effectue pas à pas les différentes étapes menant à la construction d'une transformée sur un corps fini. Il débute par la factorisation de Φ_n sur $\mathbb{F}_p$, et contient un algorithme FFT récursif. Il utilise pleinement les facilités de MAPLE, ce qui permet de ne pas avoir à se soucier des opérations dans le corps $\mathbb{F}_{p^r}$. On notera enfin qu'il précède le programme sur les *codes correcteurs BCH* (paragraphe 4, annexe B), ce dernier utilisant les procédures de calcul de transformée sur un corps fini.

2 Calculs sur un anneau

Il est naturel, après avoir présenté la transformée de Fourier à valeurs dans un corps fini, d'essayer d'étendre cette notion à un anneau commutatif quelconque. Cette généralisation n'est pas du tout gratuite, puisqu'elle va permettre de calculer dans un anneau $\mathbb{Z}/m\mathbb{Z}$ quelconque à moindre frais (le calcul des sommes et des produits se fait tout simplement modulo m). Nous allons ainsi obtenir un algorithme FFT nettement plus simple que celui construit au paragraphe précédent, qui nécessitait le passage dans un sur-corps et le calcul de divisions euclidiennes pour effectuer les produits dans ce corps. Bien sûr, tout comme dans l'exemple simpliste du paragraphe 1.2, où nous avions $n = p - 1$, nous allons encore une fois avoir des limitations sur le choix de n ou de p. Cependant, nous allons voir qu'en considérant des anneaux du type $\mathbb{Z}/2^r\mathbb{Z}$, on peut construire un algorithme FFT très simple, où la recherche de racine primitive devient triviale.

2.1 Racines principales de l'unité

Dans ce paragraphe, nous considérons des structures plus générales, à savoir des anneaux commutatifs quelconques. Dans la pratique, on utilise surtout l'anneau $\mathbb{Z}/m\mathbb{Z}$, où m est un entier positif, ce qui permet de faire des calculs de façon simple à l'aide d'un ordinateur (addition et multiplication modulo m). Un problème se pose lorsque l'on souhaite définir une transformée de Fourier à l'aide de l'équation (1.2) sur un anneau A quelconque : la transformation ainsi définie n'a aucune raison d'être bijective. Ceci est dû à la présence de diviseurs de zéro dans l'anneau A. On rappelle qu'un diviseur de zéro $x \neq 0$, est un élément de A tel qu'il existe un $y \neq 0$ vérifiant $xy = 0$. Par exemple, dans l'anneau $\mathbb{Z}/6\mathbb{Z}$, on a l'égalité $2 \cdot 3 = 0$, donc les éléments 2 et 3 sont diviseurs de zéro. Pour pallier ce problème, nous allons devoir imposer des restrictions supplémentaires sur le choix de la racine $n^{\text{ième}}$ ζ. En suivant les notations de DEMAZURE [24], nous allons introduire le concept de *racine principale de l'unité*.

Définition 2.1 (Racine principale de l'unité). Soit A un anneau commutatif quelconque. Un élément $\zeta \in A$ est appelé racine $n^{\text{ième}}$ principale de l'unité si :

(i) on a $\zeta^n = 1$ (en particulier, ζ est inversible).

(ii) pour tout $i \in \{1, \ldots, n-1\}$, l'élément $1 - \zeta^i$ n'est pas un diviseur de zéro dans A. Ceci signifie que si $a\zeta^i = a$, alors $a = 0$.

***Remarque* 2.2.** Le fait que $\zeta^i - 1$ ne soit pas diviseur de zéro pour $i = 1, \ldots, n-1$ implique en particulier que $\zeta^i \neq 1$, donc une racine principale est une racine primitive. On

constate que si l'anneau est intègre (et a fortiori si c'est un corps), les notions de racine $n^{\text{ième}}$ primitive et de racine $n^{\text{ième}}$ principale coïncident. Par contre, si on considère par exemple l'anneau $\mathbb{Z}/15\mathbb{Z}$, on voit que l'élément 2 est une racine primitive $4^{\text{ième}}$ de l'unité, mais elle n'est pas principale, puisque $2^2 - 1 = 3$ est un diviseur de zéro.

Nous allons donc rechercher une racine principale de l'unité, par exemple en menant une recherche exhaustive parmi les éléments de A^*. Comme ce calcul est effectué une fois pour toutes, il n'est pas important de disposer d'un algorithme rapide. On peut alors définir la transformée de Fourier discrète à valeurs dans A de la façon habituelle.

Définition 2.3 (Transformée de Fourier dans un anneau). Soit A un anneau commutatif, et $\zeta \in A^*$ une racine $n^{\text{ième}}$ principale. Pour un vecteur $f \in A^n$, on définit la transformée de Fourier $\mathscr{F}$ ainsi qu'une autre application notée $\widetilde{\mathscr{F}}$ par

$$\forall j \in \{0,\dots,n-1\}, \quad \mathscr{F}(f)[j] \stackrel{\text{déf.}}{=} \sum_{k=0}^{n-1} f[k]\zeta^{-kj}, \tag{2.1}$$

$$\forall j \in \{0,\dots,n-1\}, \quad \widetilde{\mathscr{F}}(f)[j] \stackrel{\text{déf.}}{=} \sum_{k=0}^{n-1} f[k]\zeta^{kj}. \tag{2.2}$$

Dans le but d'étudier les relations entre ces deux transformées, nous avons besoin du lemme suivant.

Lemme 2.4. *Soit A un anneau commutatif et ζ une racine $n^{\text{ième}}$ principale de l'unité. On a alors*

$$\sum_{i=0}^{n-1} \zeta^{ki} = \begin{cases} n & \text{si } k = 0 \mod n \\ 0 & \text{sinon} \end{cases}.$$

De plus, n n'est pas un diviseur de zéro dans l'anneau.

Démonstration. Nous allons démontrer l'égalité polynomiale suivante :

$$X^n - 1 = \prod_{i=0}^{n-1} (X - \zeta^i).$$

Notons $P(X) = X^n - 1$. On a $P(1) = 0$, donc P s'écrit $(X-1)Q(X)$, où Q est un polynôme unitaire (en effet, comme le polynôme $X - 1$ est unitaire, on peut réaliser la division euclidienne de P par $X - 1$ et voir que le reste est nul). Comme ζ est aussi racine de P, on voit que $(\zeta - 1)Q(\zeta) = 0$, et le fait que $\zeta - 1$ ne soit pas diviseur de zéro permet de conclure que $Q(\zeta) = 0$. On recommence avec Q, que l'on écrit sous la forme $(X - \zeta)R(X)$. On a alors $P(\zeta^2) = (\zeta^2 - 1)\zeta(\zeta - 1)R(\zeta^2)$. Le fait que $\zeta^2 - 1$ ne soit pas diviseur de zéro permet d'affirmer que $R(\zeta^2) = 0$. On continue ainsi jusqu'à trouver la factorisation annoncée. En enlevant le facteur $X - 1$, on trouve

$$X^{n-1} + \cdots + X + 1 = \prod_{i=1}^{n-1} (X - \zeta^i).$$

D'où, en évaluant l'égalité précédente en $X = 1$,

$$n1_A = \prod_{i=1}^{n-1} (1 - \zeta^i).$$

Ceci montre que n n'est pas diviseur de zéro dans A. En évaluant une autre fois la même égalité, mais pour $X = \zeta^k$, on obtient l'égalité annoncée. □

On peut alors énoncer le résultat principal.

Théorème 2.5 (Propriétés de la transformée sur un anneau). *On note $*$ le produit de convolution circulaire et $\cdot$ le produit composante par composante. $\mathscr{F}$ est un morphisme d'algèbre de $(A^n,*)$ dans $(A^n,\cdot)$. On a les relations*

$$\forall f \in A^n, \quad \mathscr{F}(\widetilde{\mathscr{F}}(f)) = nf \qquad et \qquad \widetilde{\mathscr{F}}(\mathscr{F}(f)) = nf.$$

Les morphismes $\mathscr{F}$ et $\widetilde{\mathscr{F}}$ sont injectifs. De plus, si $n1_A$ est inversible, alors l'application $\mathscr{F}$ est un isomorphisme d'inverse $n^{-1}\widetilde{\mathscr{F}}$.

Démonstration. La propriété de morphisme ne présente pas de difficulté, la démonstration est identique à celle effectuée pour la transformée à valeur dans $\mathbb{C}$, théorème 4.15, chap. I.

Nous allons calculer $\mathscr{F}(\widetilde{\mathscr{F}}(f))$:

$$\mathscr{F}(\widetilde{\mathscr{F}}(f))[n] = \sum_i \zeta^{-in} \sum_j f[j]\zeta^{ij} = \sum_j f[j] \sum_i \zeta^{i(j-n)}.$$

Il suffit ensuite d'utiliser le lemme précédent 2.4 avec $k = j-n$ pour conclure.

L'injectivité de $\mathscr{F}$ résulte simplement du fait que n n'est pas diviseur de zéro dans A, car si $\mathscr{F}(f) = 0$, alors $\widetilde{\mathscr{F}}(\mathscr{F}(f)) = nf = 0$, donc $f = 0$. □

Dans le cadre de l'anneau $A \overset{\text{déf.}}{=} \mathbb{Z}/m\mathbb{Z}$, les conditions sous lesquelles on peut construire une transformée de Fourier deviennent plus simples. L'exercice VI.2 détaille les étapes qui permettent de démontrer la proposition suivante :

Proposition 2.6. *Soit $m = p_1^{k_1} \times \cdots \times p_r^{k_r}$ où les p_i sont des nombres premiers distincts. On peut construire une transformée de Fourier de taille n inversible sur l'anneau $A \overset{\text{déf.}}{=} \mathbb{Z}/m\mathbb{Z}$ si les conditions suivantes sont satisfaites.*

(i) $\operatorname{pgcd}(n,m) = 1$.

(ii) *Si m s'écrit sous la forme , alors n divise* $\operatorname{pgcd}(p_1 - 1, \ldots, p_r - 1)$.

La condition (i) permet d'inverser n dans A, et la condition (ii) permet de construire une racine $n^{\text{ième}}$ principale. Dans le paragraphe suivant, nous allons particulariser cette étude à certaines classes d'entiers m, dans le but de construire un algorithme de calcul rapide de type FFT.

2.2 Implémentation d'un algorithme FFT

On souhaite implémenter un algorithme FFT dichotomique sur un anneau A quelconque. On suppose donc que $n = 2^s$, et le problème, pour pouvoir implémenter l'algorithme, est tout d'abord de trouver une racine $2^{s\text{ième}}$ principale de l'unité. Mais lorsque le deuxième appel récursif sera lancé, il faudra trouver une racine $(2^{s-1})^{\text{ième}}$ principale, puis une racine $(2^{s-2})^{\text{ième}}$, etc. Si on veut que l'algorithme FFT soit réellement utile, il faut que cette recherche d'une racine principale prenne le moins de temps possible.

La proposition suivante, dont la démonstration fait l'objet de l'exercice VI.3, va nous permettre de construire cette racine principale ζ.

Proposition 2.7. *Soit A un anneau commutatif. Pour que ζ soit une racine principale $(2^k)^{ième}$, il faut et il suffit que 2 ne soit pas diviseur de zéro dans l'anneau, et de plus que $\zeta^{2^{k-1}} = -1$.*

On va choisir un entier m impair pour que 2 soit inversible dans $A \overset{\text{déf.}}{=} \mathbb{Z}/m\mathbb{Z}$ (en particulier, 2 ne sera pas un diviseur de zéro). De plus, en prenant simplement $\zeta = 2$, on voit que $\zeta^{2^{k-1}} = -1$ dans A en assignant à m la valeur $2^{2^{k-1}} + 1$. On a bien trouvé une racine $(2^k)^{\text{ième}}$ de l'unité, en l'occurrence $\zeta = 2$, de plus, comme on le voit en reprenant l'exercice VI.3, ζ^2 sera alors une racine $(2^{k-1})^{\text{ième}}$ de l'unité principale, puis on utilisera ζ^4, etc.

***Remarque* 2.8.** Les nombres de la forme $2^{2^n} + 1$ sont d'une grande importance, et sont appelés *nombres de Fermat*. On note $\text{Fer}_n \overset{\text{déf.}}{=} 2^{2^n} + 1$ le $n^{\text{ième}}$ nombre de Fermat. On voit facilement que tout nombre premier de la forme $2^k + 1$ est en fait un nombre de Fermat. On peut voir que $\text{Fer}_0 = 3$, $\text{Fer}_1 = 5$, $\text{Fer}_2 = 17$, $\text{Fer}_3 = 257$ et $\text{Fer}_4 = 65537$ sont tous premiers, malheureusement, Fer_5 ne l'est pas ...

En conclusion, cette méthode, bien que moins souple au niveau du choix de n, est nettement plus simple à mettre en œuvre que la transformée de Fourier dans un corps cyclotomique. De plus, elle demande nettement moins de calculs. Comme un programme complet vaut mieux que de longs discours, on pourra se référer au paragraphe 2, annexe B, où l'algorithme FFT de longueur n sur l'anneau $\mathbb{Z}/m\mathbb{Z}$ avec $m = 2^{2^{s-1}}$ est implémenté en MAPLE.

3 Application aux codes correcteurs

> The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have meaning ; that is they refer to or are correlated according to some system with certain physical or conceptual entities.
>
> C. E. SHANNON [66] (1948)

Ce paragraphe est une modeste introduction à la théorie des *codes correcteurs* ; il s'agit avant tout de donner envie au lecteur de chercher des informations complémentaires dans la bibliographie proposée. Le but est d'appliquer les outils introduits depuis le début de ce chapitre, d'une part pour mieux comprendre les conditions mises en jeu dans la construction des codes, et d'autre part pour obtenir des algorithmes de décodage rapides et efficaces. Dans un premier temps, les définitions des notions principales sont données, en mettant en avant la théorie des codes cycliques. Puis, il s'agit de réinvestir au mieux les connaissances que l'on possède sur les corps cyclotomiques et sur la transformée de Fourier discrète, pour arriver, à la fin de l'exposé, à construire des codes et des algorithmes.

La théorie classique des codes correcteurs d'erreurs aborde une grande variété de sujets, ce qui la rend particulièrement attractive (par exemple pour illustrer une leçon d'agrégation). Tout d'abord, elle traite de la géométrie finie (puisqu'il s'agit au fond de manipuler des boules dans un espace fini). Ensuite, l'aspect combinatoire des codes présente bon nombre de propriétés remarquables, principalement autour des relations liant entre eux les différents paramètres des codes. Enfin, il sera beaucoup question de théorie des corps, qui constitue le cœur de cet exposé.

Une très bonne référence sur la théorie des codes correcteurs est le livre de PAPINI et WOLFMAN [56]. Celui de DEMAZURE [24] constitue un très bel exposé, avec entre autres un algorithme de décodage des codes *BCH* très efficace.

3.1 Notion de code correcteur

La notion de *codage* d'une information n'est pas nouvelle : elle est même intimement liée à l'activité humaine, au besoin naturel de communiquer par des moyens divers mais pas toujours très fiables. Le meilleur exemple est le langage humain, très complexe, et qui répond parfaitement au besoin de correction des erreurs. Après tout, pourquoi utiliser des langues naturelles si compliquées, alors que l'on pourrait utiliser des mots certainement beaucoup plus courts, avec une syntaxe beaucoup plus simple ? Une des explications consiste à dire que ces langues naturelles permettent de mieux se comprendre (à condition d'en maîtriser les rudiments). En effet, la diversité des mots employés diminue le risque d'erreur lors d'une conversation, et la rigidité des règles de grammaire rend cette communication moins sensible aux aléas (bruit ambiant, mauvaise prononciation, etc.). En quelques sorte, tout ceci contribue à rendre les mots qui composent la langue très différents les uns des autres, de façon à ce qu'on puisse facilement les distinguer. Le cas échéant, si une erreur de communication se produit, il sera relativement simple pour le locuteur de retrouver le sens du message orignal.

Ce premier exemple donne tous les points clefs d'une théorie des codes correcteurs d'erreurs. Il s'agit de trouver un moyen de coder une certaine information de façon à la rendre moins sensible aux erreurs lors d'une communication. On peut découper le processus de codage de la manière suivante :

- la transformation de l'information (qui peut être une pensée, un son, une séquence d'ADN, etc.) sous la forme d'une suite de symboles. Dans la théorie qui va suivre, nous allons nous désintéresser du sens de cette suite de symboles. Il s'agit en quelque sorte d'une couche d'abstraction qui va nous permettre de fixer une façon de représenter l'information que l'on veut traiter.
- le codage mathématique proprement dit. C'est cette partie du processus de codage qui va nous intéresser. Il s'agit de modifier de façon adéquate la suite de symboles de façon à la rendre le moins sensible possible aux erreurs de transmission. Cette tranformation va exiger l'ajout d'information redondante.

Le but de la théorie des codes correcteurs est donc d'utiliser des structures algébriques et des algorithmes pour que :

- l'information redondante ajoutée au message lors du codage soit la plus faible possible, pour un nombre d'erreurs corrigées fixé.
- les algorithmes de codage et surtout de décodage soient rapides et efficaces.

Pour y parvenir, nous allons imposer des structures (algébriques) plus ou moins rigides sur l'ensemble des mots manipulés (que l'on nomme le code).

Le premier choix à faire est celui de l'alphabet que nous allons utiliser pour écrire les mots des messages. L'information d'origine (par exemple un message que l'on veut transmettre sur un réseau) sera ainsi transformée, lors de la première étape du codage, en une suite de symboles puisés dans cet alphabet. Un exemple capital d'alphabet est celui utilisé pour coder les informations génétiques du code humain. Dans ce cas, l'information à coder est

l'ADN (**A**cide **D**ésoxyribo**n**ucléique), et les symboles sont 4 « bases azotées », que l'on note symboliquement A,T,G et C. Ce premier exemple de codage est symptomatique de la construction mathématique que nous allons effectuer. Il s'agit de remplacer l'information (qui possède un certain sens pour celui qui la transmet) en une suite de symboles fixés. Des études récentes discutent d'ailleurs de l'existence de codes correcteurs dans la suite de séquences azotées de l'ADN, voir par exemple [49].

De façon à rendre agréable l'analyse mathématique qui va suivre, nous allons prendre pour symboles des éléments de certains ensembles algébriques. Le choix que nous allons faire, et qui peut paraître arbitraire, est celui d'un corps fini. En pratique, ce choix n'est pas si restrictif que cela. Par exemple, le codage binaire des éléments du corps $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$ convient parfaitement à la façon dont l'information est stockée dans la mémoire d'un ordinateur. Dans le même ordre d'idée, les éléments d'un corps du type $\mathbb{F}_{2^r}$ pourront être stockés sous la forme de vecteurs de r bits. Le deuxième choix à faire est celui des mots que nous allons effectivement envoyer sur un réseau. Ils auront tous la même longueur, que l'on notera n. Ainsi nous allons regrouper les éléments de l'alphabet envoyés par paquets de n pour obtenir des n-uplets $(x_0,\dots,x_{n-1})$, avec $x_k \in \mathbb{F}_q$. Dans la suite, nous supposerons donc que ces mots sont des éléments de l'espace vectoriel $(\mathbb{F}_q)^n$, où $q \overset{\text{déf.}}{=} p^r$ avec p un nombre premier.

On souhaite exploiter certaines structures algébriques pour donner naissance à des codes de plus en plus efficaces. Cependant, il existe de nombreux codes utilisés très fréquemment et qui ne nécessitent pas des constructions algébriques complexes. En voici deux exemples.

Exemple 3.1 (Bit de parité). Il s'agit sans doute du code le plus simple et le plus utilisé. L'information que l'on souhaite coder consiste en une suite de n bits, que l'on notera donc sous la forme $a = a_0a_1\dots a_{n-1}$, où $a_k \in \{0,1\}$. L'opération de codage consiste à ajouter un $(n+1)^{\text{ième}}$ bit a_n défini de la manière suivante :

$$a_n \overset{\text{déf.}}{=} a_0 + \cdots + a_{n-1} \mod 2.$$

En fait, on s'arrange pour que la somme de tous les bits du mot transmis soit toujours un nombre pair. On constate alors que ce code permet, à la réception, de déceler une erreur due à la modification d'uniquement un bit, puisque alors la somme des bits sera impaire. Cependant, si deux erreurs simultanées ont lieu, le code sera incapable de les remarquer. En outre, il est vain de vouloir corriger une erreur avec un code aussi simple. Il va donc falloir construire des codes plus complexes (avec plus d'informations redondantes ajoutées), pour obtenir des performances qui satisfassent nos exigences. Cependant, ce premier exemple est fondamental, d'une part parce que dans bien des cas, il est suffisant (si la communication est très fiable, comme c'est le cas pour le stockage en mémoire centrale d'un ordinateur), et d'autre part parce qu'il peut être superposé à un code déjà existant pour le rendre plus efficace.

Exemple 3.2 (Code ISBN). Le code *ISBN* (pour **I**nternational **S**tandard **B**ook **N**umber) est un identificateur unique attribué à chaque livre. Il s'agit d'une suite de 9 chiffres décimaux suivis d'un autre chiffre compris entre 0 et 10 (on note alors 10 sous la forme du chiffre romain X). Si on note l'ISBN d'un livre $d_0\dots d_9$, le dixième chiffre est calculé de façon à satisfaire l'égalité

$$10d_0 + 9d_1 + \cdots + 2d_8 + d_9 = 0 \mod 11.$$

En général, on regroupe les d_i en 4 paquets séparés par des tirets, comme par exemple dans l'ISBN 0-201-89683-4, dont les significations sont les suivantes.

- Le premier groupe est composé d'un seul chiffre, qui caractérise le pays de l'éditeur (0 pour l'Angleterre, 2 pour la France, 3 pour l'Allemagne, etc.).
- Le deuxième groupe caractérise l'éditeur, et sa longueur peut varier de 2 à 7 chiffres (moins il y a de chiffres, plus l'éditeur est important).
- Le troisième groupe caractérise le livre, et sa longueur peut varier de 1 à 6 chiffres.
- Le quatrième et dernier groupe correspond au chiffre d_9 censé détecter d'éventuelles erreurs.

On voit facilement que ce code permet de détecter une erreur, ainsi qu'un échange entre deux chiffres consécutifs. Cependant, il ne permet pas de corriger les erreurs, ni de détecter certaines inversions de chiffres. Ce codage est très astucieux, car la plupart du temps, les codes ISBN sont saisis par des humains, qui ont souvent tendance à inverser deux chiffres consécutifs.

Il existe de nombreux autres exemples de codes correcteurs dans la vie courante. On pourrait citer les numéros de *Traveler's Checks American Express*, les *codes barres*, et bien sûr les *disques compacts*, qui utilisent des codes beaucoup plus complexes (pour contrer d'éventuelles rayures sur leur surface).

Nos mots sont donc des vecteurs de l'espace $(\mathbb{F}_q)^n$. Pour étudier les propriétés de notre code $\mathscr{C} \subset (\mathbb{F}_q)^n$, il est intéressant d'introduire une notion de proximité entre les mots qui le composent, pour pouvoir considérer ce code sous un angle géométrique. On est amené à considérer une distance sur l'ensemble des mots. Voici celle qui est la plus couramment employée.

Définition 3.3 (Distance de Hamming). On définit le *poids* d'un mot $x \in (\mathbb{F}_q)^n$, que l'on note $w(x)$, comme le nombre d'entrées non nulles dans x. La *distance de Hamming* entre deux mots x et y de $(\mathbb{F}_q)^n$ est définie par $d(x,y) = w(x-y)$.

On voit facilement que d est bien une distance, c'est-à-dire que pour des éléments x, y, et z de $(\mathbb{F}_p)^n$:

(i) $x \neq y \Longrightarrow d(x,y) > 0$;
(ii) $d(x,y) = d(y,x)$;
(iii) $d(x,y) \leqslant d(x,z) + d(z,y)$.

Cette distance correspond donc au nombre d'entrées qui diffèrent entre deux mots. Par exemple, l'illustration 6.1 permet de visualiser la distance de Hamming sur $(\mathbb{F}_2)^3$, chaque arête du cube reliant deux points à distance 1. On aurait pu choisir une autre distance, mais il s'avère que la distance de Hamming est à la fois suffisamment simple pour permettre de faire des calculs efficaces, et assez précise pour bien rendre compte de l'efficacité des codes.

3.2 Présentation des codes linéaires

Après le choix de l'alphabet (qui est donc un corps fini $\mathbb{F}_q$), la seconde hypothèse que nous allons faire concerne l'ensemble des mots que l'on peut coder. En effet, il est évident que l'opération de codage va ajouter de l'information au message original, donc l'ensemble des mots « valides » (c'est-à-dire les mots que nous serons en mesure de produire par

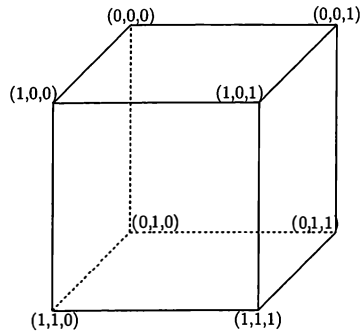

FIG. 6.1 – *Distance de Hamming sur* $(\mathbb{F}_2)^3$

codage) n'occupera pas l'espace $(\mathbb{F}_q)^n$ en entier. De façon intuitive, on peut voir que plus il y aura d'espace entre les mots valides (c'est-à-dire les mots du code), plus nous serons en mesure de les discerner les uns des autres, et donc plus il sera facile de repérer d'éventuelles erreurs.

La recherche d'un ensemble $\mathscr{C} \subset (\mathbb{F}_q)^n$ présentant, pour une taille $|\mathscr{C}|$ donnée, la meilleure répartition de poids (dans un sens à préciser, mais intuitivement, tel que les mots soient les plus éloignés possibles des uns des autres) est un problème extrêmement difficile. Ainsi, cette recherche est intimement liée à celle d'empilement de sphères les plus compacts possibles, et au fameux problème du *Kissing Number*. On renvoie pour plus de détails à l'article de ELKIES [30]. On peut définir un ensemble de constantes caractérisant avec plus ou moins de finesse la répartition des mots d'un code.

Définition 3.4 (Fonctions de répartition). On note $\{A_i\}_{i=0}^n$ la *répartition des poids* d'un code $\mathscr{C} \subset (\mathbb{F}_q)^n$:

$$\forall i = 0,\ldots,n, \quad A_i \stackrel{\text{déf.}}{=} \#\{x \in \mathscr{C} \setminus w(x) = i\}. \tag{3.1}$$

On note $\{B_i\}_{i=0}^n$ la *répartition de distance* de $\mathscr{C}$:

$$\forall i = 0,\ldots,n, \quad B_i \stackrel{\text{déf.}}{=} \frac{1}{|\mathscr{C}|}\#\{(x,y) \in \mathscr{C} \setminus d(x,y) = i\}. \tag{3.2}$$

Il est à noter que les couples (x,y) sont considérés ordonnés c'est-à-dire $(x,y) \neq (y,x)$.

***Remarque* 3.5.** On constate que l'on a $B_0 = 1$, et que

$$A_0 + \cdots + A_n = B_0 + \cdots + B_n = |\mathscr{C}|.$$

De plus, si u est un vecteur quelconque de $(\mathbb{F}_q)^n$, les codes $\mathscr{C}$ et $\mathscr{C} + u$ ont la même répartition de poids. C'est pourquoi dans la pratique on suppose que $0 \in \mathscr{C}$, même si $\mathscr{C}$ n'est pas linéaire.

Nous reviendrons sur le calcul et l'étude de ces répartitions à la section 4. Pour quantifier de façon plus simple cette répartition, nous introduisons une notion très utile, celle de *distance minimale*.

Définition 3.6 (Distance minimale). On note d la *distance minimale* du code $\mathscr{C}$ considéré, qui est définie de la façon suivante :

$$d \stackrel{\text{déf.}}{=} \min\{d(x,y) \setminus x \neq y \in \mathscr{C}\}.$$

Intuitivement, on se rend compte que plus cette distance minimale est grande, plus le code va être efficace, puisque les mots seront plus éloignés les uns des autres. Ce choix est très partial, et en quelque sorte très pessimiste, puisqu'il ne prend en compte que la plus petite distance. Cependant, il nous donne des conditions sous lesquelles on est assuré de corriger une erreur, ce qui est précisé par la définition suivante.

Définition 3.7 (Capacité de correction). La *capacité de correction t* d'un code $\mathscr{C}$ est le nombre maximum d'erreurs qu'il peut corriger. De façon plus précise, si l'émetteur envoie un mot codé $x \in \mathscr{C}$ à travers un réseau, et que le récepteur reçoit un mot x' (qui est peut être différent du mot envoyé), alors il doit être en mesure de retrouver le mot original si $d(x,x') \leqslant t$.

Ceci signifie que les boules de rayon t (pour la distance de Hamming) dont le centre est un mot du code doivent être disjointes les unes des autres, ou, en d'autres termes, que les mots doivent être à une distance d'au moins $2t+1$ les uns des autres. On obtient donc le résultat suivant.

Proposition 3.8. *Un code $\mathscr{C}$ est t-correcteur (c'est-à-dire qu'il a une capacité de correction d'au moins t) si la distance minimale entre deux mots distincts est supérieure ou égale à $2t+1$. La capacité de correction du code est $t = \lfloor (d-1)/2 \rfloor$, où l'on a noté $\lfloor x \rfloor$ la partie entière d'un réel x.*

Pour simplifier grandement la recherche de codes efficaces (par exemple avec une grande distance minimale d), nous allons imposer une structure très restrictive aux mots du code.

Définition 3.9 (Code linéaire). Un code linéaire $\mathscr{C}$ de taille n et de dimension m sur $\mathbb{F}_q$ est un sous-espace vectoriel de dimension m de $(\mathbb{F}_q)^n$. Si on considère une matrice G (appelée *matrice génératrice*) dont les colonnes forment une base de $\mathscr{C}$, on a

$$\mathscr{C} = \left\{ Gx \setminus x \in (\mathbb{F}_q)^m \right\}.$$

Notons bien qu'il n'y a pas unicité dans le choix de G. Certes, un code linéaire, même optimal, sera dans le meilleur des cas aussi bon que le meilleur des codes non linéaires (c'est-à-dire un code quelconque), et, dans la pratique, il sera beaucoup moins bon. Cependant, le fait de se restreindre à des sous-espaces vectoriels va rendre notre recherche beaucoup plus fructueuse, et va aussi amener son lot d'algorithme de décodage efficace. Par exemple, la propriété de linéarité de $\mathscr{C}$ nous permet de calculer la distance minimale d beaucoup plus simplement :

$$d \overset{\text{déf.}}{=} \min \{ d(x,y) \setminus x \neq y \in \mathscr{C} \} = \min \{ w(x) \setminus x \neq 0 \in \mathscr{C} \}.$$

De même, dans le cas linéaire, on constate que les répartitions de poids et de distance coïncident. Sauf mention explicite du contraire, on suppose maintenant que le code $\mathscr{C}$ est un code linéaire, de taille n et de dimension m.

La première phase de l'opération de codage consiste à transformer le message d'origine contenant une certaine information en un mot du code $\mathscr{C}$. Il faut que cette opération soit bijective. Dans le cas d'un code linéaire, il est très facile d'y parvenir. La façon la plus simple d'opérer est tout simplement de considérer que nos messages sont des « petits » vecteurs de $(\mathbb{F}_q)^m$, et qu'on les envoie sur des « grands » vecteurs de $(\mathbb{F}_q)^n$, simplement en les multipliant à gauche par la matrice G. La matrice G n'est pas choisie de façon canonique, mais le choix d'une autre base conduit à un code ayant sensiblement les mêmes propriétés (on parle de code isomorphe).

***Remarque* 3.10. (Matrice de contrôle).** Un code de taille n et de dimension m sur $\mathbb{F}_q$ peut être vu comme le noyau d'une matrice H de taille $(n-m) \times n$. On appelle cette

matrice une *matrice de contrôle* du code $\mathscr{C}$; elle permet de vérifier si un vecteur $x \in (\mathbb{F}_q)^n$ appartient au code puisque $x \in \mathscr{C} \Leftrightarrow Hx = 0$. Il n'y a pas unicité dans le choix de G.

Exemple 3.11 (Code de répétitions). Prenons l'exemple simple du code de répétition. Il consiste à répéter, par exemple, 4 fois un symbole $x \in \mathbb{F}_2$. Les deux seuls mots du code sont (0000) et (1111). Des matrices génératrices G et de contrôle H sont par exemple

$$G = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad H = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Le code dual (notion précisée un peu plus tard, définition 4.1) est constitué des vecteurs $x \in (\mathbb{F}_2)^4$ tels que $\langle x, [1111] \rangle = 0$. C'est donc simplement le code consistant à ajouter à $x \in \mathbb{F}_2^3$ un bit de parité. On constate qu'il suffit, à une transposition près, d'échanger les matrices génératrices et de contrôle. On pourra voir à ce sujet l'exercice VI.6.

Exemple 3.12 (Code de Hamming de longueur 7). On considère le code de taille 7 et de dimension 4 sur $\mathbb{F}_2$ dont la matrice génératrice est

$$G \overset{\text{déf.}}{=} \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

On peut expliciter les 16 éléments qui composent le code :

x^{T}	$(Gx)^{\mathrm{T}}$	$w(Gx)$	x^{T}	$(Gx)^{\mathrm{T}}$	$w(Gx)$
(0000)	(0000000)	0	(1000)	(1101000)	3
(0100)	(0110100)	3	(0010)	(0011010)	3
(0001)	(0001101)	3	(1100)	(1011100)	4
(1010)	(1110010)	4	(1001)	(1100101)	4
(0110)	(0101110)	4	(0101)	(0111001)	4
(0011)	(0010111)	4	(1110)	(1000110)	3
(1101)	(1010001)	3	(1011)	(1111111)	7
(0111)	(0100011)	3	(1111)	(1001011)	4

Comme on peut le constater, chaque mot non nul du code est de poids supérieur à 3, donc la capacité de correction de ce code est 1. De plus, il possède une propriété intéressante : les 4 vecteurs lignes de la matrice G se déduisent les uns des autres par permutation circulaire. Par conséquent, l'ensemble du code est invariant par permutation circulaire. Dans la suite, nous nous intéresserons aux propriétés algébriques de tels codes, et nous verrons que l'on dispose d'outils fort pratiques pour les construire, et, dans certains cas, les décoder. L'exercice VI.7 propose de généraliser la construction qui vient d'être faite, pour donner naissance à une famille de codes très utilisée, les *codes de Hamming*.

Pour terminer ce paragraphe, voici une relation importante entre les différents paramètres d'un code, qui montre bien le choix à faire entre capacité de correction et redondance de l'information transmise.

Proposition 3.13 (Borne de Singleton). *Soit $\mathscr{C}$ un code correcteur linéaire de longueur n, de dimension m, et de distance minimale d. On a alors*

$$d \leqslant n + 1 - m. \tag{3.3}$$

Démonstration. Soit E le sous-espace de $(\mathbb{F}_q)^n$ formé des vecteurs dont les $m-1$ dernières composantes sont nulles. C'est un espace de dimension $n-m+1$.
On a $\dim(\mathscr{C})+\dim(E)=n+1$ ce qui implique que $\mathscr{C}\cap E\neq\{0\}$. Il existe donc un élément x non nul dans $\mathscr{C}$ dont les $m-1$ dernières composantes sont nulles, donc qui vérifie $w(x)\leqslant n+1-m$. On a donc la relation voulue de part la définition de la distance minimale. □

***Remarque* 3.14.** Les codes qui vérifient $d=n+1-m$ sont appelés *codes MDS* (pour **M**aximum **D**istance **S**eparable, en Anglais dans le texte), ils sont donc optimaux pour la borne de Singleton. L'exercice VI.12 étudie en détail ces codes.

3.3 Codes cycliques

Le problème des codes linéaires est que dans le cas général, on ne dispose pas d'algorithmes rapides de décodage, ou alors ces derniers nécessitent le stockage de tables de décodage qui deviennent vite énormes pour des codes de tailles respectables. Pour remédier à ce problème, nous allons imposer une structure supplémentaire aux codes linéaires.

Définition 3.15 (Code cyclique). Un code $\mathscr{C}$ de taille n sur $\mathbb{F}_p$ est dit cyclique s'il est stable par décalage circulaire, c'est-à-dire

$$\forall a=(a_0,\dots,a_{n-1})^{\mathrm{T}}\in\mathscr{C},\quad \widetilde{a}\overset{\text{déf.}}{=}(a_{n-1},a_0,\dots,a_{n-2})^{\mathrm{T}}\in\mathscr{C}.$$

Une façon très commode de représenter les mots d'un code cyclique consiste à les considérer comme des éléments de la $\mathbb{F}_p$-algèbre $\mathscr{A}$ de dimension n qu'est $\mathbb{F}_p[X]/(X^n-1)$. On considère donc qu'un mot a est en fait un polynôme de degré au plus $n-1$ (on choisit un représentant modulo X^n-1), noté $a_0+a_1X+\cdots+a_{n-1}X^{n-1}$. On remarque alors que

$$\widetilde{a}=a_{n-1}+a_0X+\cdots+a_{n-2}X^{n-1}=Xa+a_{n-1}(X^n-1).$$

Donc modulo X^n-1 (c'est-à-dire dans $\mathscr{A}$), on a $\widetilde{a}=Xa$. Le code $\mathscr{C}$ est stable par multiplication par X. Comme c'est aussi un espace vectoriel, par linéarité, on en déduit qu'il est en fait stable par multiplication par tout polynôme $P\in\mathscr{A}$. Ceci signifie que c'est un idéal de l'anneau $\mathscr{A}$. On sait que les idéaux de $\mathscr{A}$ sont en bijection avec les idéaux de $\mathbb{F}_p[X]$ qui contiennent l'idéal engendré par X^n-1. Comme l'anneau de polynômes $\mathbb{F}_p[X]$ est principal, un idéal de $\mathbb{F}_p[X]/(X^n-1)$ est engendré par un unique polynôme unitaire qui doit en plus être un diviseur de X^n-1.

Si on note P le polynôme générateur du code $\mathscr{C}$, on a, en notant $s\overset{\text{déf.}}{=}\deg(P)$,

$$\mathscr{C}=\left\{PQ \mod X^n-1 \setminus Q\in\mathbb{F}_p[X]\right\}=\{PQ\setminus \deg(Q)\leqslant n-s-1\}.$$

Le code $\mathscr{C}$ est donc de longueur n et de dimension $n-s$. L'opération de codage est encore plus simple que pour un code linéaire. L'information que l'on veut transmettre, au lieu d'être contenue dans un vecteur de taille $n-s$, est cette fois représentée par un polynôme de degré au plus $n-s-1$ (mais c'est la même chose). Pour obtenir un mot du code que nous allons envoyer sur un réseau, il suffit de multiplier ce polynôme par le polynôme générateur P.

Nous pouvons dès maintenant faire un rapprochement avec les idées que nous avons introduites lors de l'étude de la transformée de Fourier sur un groupe cyclique, et plus particulièrement lors de la recherche de techniques de multiplication rapide de polynômes

(section 5, chap. IV). Nous avons déjà remarqué que la multiplication par un polynôme modulo $X^n - 1$ correspond au calcul d'une convolution cyclique de vecteurs de taille n (l'équation (5.1), chap. IV, montre très clairement le rapprochement). On peut écrire l'opération de codage d'un vecteur $x \in (\mathbb{F}_p)^{n-s}$ comme la convolution circulaire $x * y$ où on a noté y le vecteur des coefficients du polynôme P. Il convient d'ajouter des zéros à la fin de chacun des vecteurs pour qu'ils atteignent la taille n. La matrice génératrice G du code correspond donc à une matrice circulante comme nous l'expliquons à l'exercice III.5.

Nous ne nous étions intéressés, à la section 5, chap. IV, qu'à des calculs pour des polynômes de $\mathbb{C}[X]$. Mais cette limitation n'était justifiée que parce que nous ne disposions pas de transformée de Fourier sur un corps autre que $\mathbb{C}$. Grâce à la construction du paragraphe 1.4, cette limitation est levée, et nous sommes capables de réaliser une transformée de Fourier à valeurs dans le corps $\mathbb{F}_p$ (même si cette dernière, rappelons-le, nécessite de passer dans un corps plus grand, que l'on a noté $\mathbb{F}_{p^r}$). Donc en reprenant la formule de calcul du produit de convolution cyclique (5.1), chap. IV, on voit que nous pouvons effectuer de façon rapide l'opération de codage, moyennant bien sûr l'utilisation d'un algorithme FFT pour réaliser la transformée de Fourier.

3.4 Construction des codes BCH

Dans ce paragraphe, nous allons présenter une classe de codes cycliques nommés *codes BCH*, du nom de leurs inventeurs, BOSE, CHAUDHURI et HOCQUENGHEM. La construction utilise pleinement la décomposition des polynômes cyclotomiques expliquée au paragraphe 1.3. L'intérêt majeur de ces codes, outre leur description simple à l'aide de la transformée de Fourier, est que l'on dispose explicitement d'un minorant de leur capacité de correction. Ceci permet d'ajuster les paramètres du code en fonction des besoins. De plus, nous allons voir au paragraphe 3.5 que l'on dispose d'un algorithme de décodage efficace, ce qui rend ces codes utilisables de façon pratique.

Nous allons construire un polynôme générateur d'un code cyclique à l'aide des corps cyclotomiques présentés au paragraphe 1.3. En effet, puisque nous nous intéressons aux diviseurs du polynôme $X^n - 1$, il est naturel de considérer le comportement modulo p des polynômes cyclotomiques Φ_n. Dans la suite, on suppose que $\operatorname{pgcd}(n, p) = 1$ (voir la remarque 1.7 dans le cas contraire). Rappelons que si on note r l'ordre de p dans le groupe multiplicatif $(\mathbb{Z}/n\mathbb{Z})^*$, alors $K \overset{\text{déf.}}{=} \mathbb{F}_{p^r}$ est un corps de rupture du polynôme Φ_n. Ceci nous permet donc de choisir $\alpha \in K$ une racine $n^{\text{ième}}$ primitive de l'unité (un tel choix, rappelons-le, résulte du choix d'un facteur irréductible de Φ_n modulo p, qui est de degré r). Nous allons alors choisir le polynôme générateur P sous la forme

$$P = \prod_{i \in I} (X - \alpha^i), \tag{3.4}$$

où I est un sous-ensemble de $\{1, \ldots, n-1\}$ à déterminer. En effet, pour que l'on obtienne un code cyclique sur $\mathbb{F}_p$, encore faut-il que le polynôme P soit à coefficients dans $\mathbb{F}_p$, et non pas simplement dans K. Voici un lemme simple mais important qui nous donne un moyen pour voir si un polynôme appartient à $\mathbb{F}_p[X]$.

Lemme 3.16. *Un polynôme $Q \in K[X]$ appartient à $\mathbb{F}_p[X]$ si et seulement s'il vérifie $Q(X^p) = Q(X)^p$.*

Démonstration. On sait qu'un élément $y \in K$ appartient au sous-corps premier $\mathbb{F}_p$ si et seulement s'il vérifie $y^p = y$ (reprendre les arguments de la preuve de la proposition 1.6 en considérant les racines de $X^p - X$). En utilisant le *morphisme de Frobenius* $x \mapsto x^p$ de K dans K, on voit que si on note $Q = a_0 + \cdots + a_k X^k$, on a

$$Q(X)^p = (a_0 + a_1 X + \cdots + a_k X^k)^p = a_0^p + a_1^p X^p + \cdots + a_k^p (X^p)^k. \tag{3.5}$$

Ainsi, dire que $a_i^p = a_i$ pour $i = 0, \ldots, k$ revient donc à dire que $Q(X)^p = Q(X^p)$. □

La proposition suivante nous donne alors un critère qui va nous permettre de choisir de façon effective l'ensemble I.

Proposition 3.17. *Le polynôme P appartient à $\mathbb{F}_p[X]$ si et seulement si I est stable par multiplication par p modulo n.*

Démonstration. Si $P \in \mathbb{F}_p[X]$ et si β est une racine de P, on voit avec le lemme précédent que $P(\beta^p) = P(\beta)^p = 0$, en conséquence de quoi l'ensemble I est stable par multiplication par p.
Réciproquement, si I est stable par multiplication par p, alors

$$P(X)^p = \prod_{i \in I} (X - \alpha^i)^p = \prod_{i \in I} (X^p - \alpha^{ip}) = \prod_{i \in I} (X^p - \alpha^i) = P(X^p),$$

donc on a bien $P \in \mathbb{F}_p[X]$. □

On a enfin en mains tous les outils nécessaires à la définition des codes BCH.

Définition 3.18 (Codes BCH). On appelle *code BCH* de distance assignée δ un code cyclique dont le polynôme générateur est obtenu par l'équation (3.4), où I désigne la plus petite classe cyclotomique (c'est-à-dire le plus petit ensemble stable par multiplication par p modulo n) contenant l'ensemble d'indices $\{1, \ldots, \delta - 1\}$.

Avant d'aller plus loin dans l'investigation des propriétés de ces codes, nous allons donner un moyen simple de calculer le polynôme générateur une fois que l'on connaît la décomposition sur $\mathbb{F}_p$ du polynôme cyclotomique Φ_n. Pour ce faire, nous allons considérer les classes cyclotomiques les plus simples qui soient, c'est-à-dire les classes engendrées par un seul élément $k \in \{0, \ldots, n-1\}$:

$$I_k \overset{\text{déf.}}{=} \{k, kp, \ldots, kp^{s-1}\}, \tag{3.6}$$

où on a noté s le plus petit entier tel que $kp^s = k$ (dans le cas où $k = 1$, on a bien sûr $s = r$ degré de P). De façon plus élégante, on peut considérer la relation d'équivalence $\sim$ sur $\mathbb{Z}/n\mathbb{Z}$:

$$\forall (x, y) \in (\mathbb{Z}/n\mathbb{Z})^2, \quad (x \sim y) \Leftrightarrow \left(\exists i, x = yq^i\right).$$

Les classes cyclotomiques sont alors les classes d'équivalence de $\mathbb{Z}/n\mathbb{Z}$ pour cette relation, et forment une partition de $\{0, \ldots, n-1\}$. On remarque que le polynôme

$$P_k \overset{\text{déf.}}{=} \prod_{i \in I_k} (X - \alpha^i) \tag{3.7}$$

est, d'après la proposition 3.17, un polynôme de $\mathbb{F}_p[X]$ irréductible de degré s admettant α^k comme racine. En conséquence, c'est le polynôme minimal de α^k. On obtient alors facilement la description suivante du polynôme générateur du code.

Proposition 3.19. *Le polynôme P générateur d'un code BCH de distance assigné δ est le PPCM des polynômes $P_1, \ldots, P_{\delta-1}$ définis par l'équation* (3.7).

Exemple 3.20. Voici un exemple de l'utilisation des classes cyclotomiques, dans le cas de codes sur le corps $\mathbb{F}_2$. On souhaite construire des codes de longueur $n \stackrel{\text{déf.}}{=} 2^5 - 1 = 31$, donc on a ici $r = 5$. Voici la liste des facteurs irréductibles de $X^n - 1$ sur $\mathbb{F}_2$:

Polynôme :	Classe cyclotomique :
$P_0 = X + 1$	$I_0 = \{0\}$
$P_1 = 1 + X + X^2 + X^4 + X^5$	$I_1 = \{1,2,4,8,16\}$
$P_3 = 1 + X^3 + X^5$	$I_3 = \{3,6,12,24,17\}$
$P_5 = 1 + X + X^2 + X^3 + X^5$	$I_5 = \{5,10,20,9,18\}$
$P_{11} = 1 + X + X^2 + X^3 + X^4 + X^5$	$I_{11} = \{11,22,13,26,21\}$
$P_{14} = 1 + X^2 + X^5$	$I_{14} = \{14,28,25,19,7\}$
$P_{15} = 1 + X + X^3 + X^4 + X^5$	$I_{15} = \{15,30,29,27,23\}$

Conformément à ce que nous avons conseillé de faire pour construire un corps cyclotomique, on choisit arbitrairement un facteur irréductible de plus haut degré, par exemple P_1, et on note α une de ses racines, qui sera donc une racine primitive de l'unité, et que l'on peut voir comme un élément de $K \stackrel{\text{déf.}}{=} \mathbb{F}_{2^5}$. Le polynôme P_0 est ainsi le polynôme minimal de 1 associé à la classe I_0, le polynôme P_3 le polynôme minimal de α^3 associé à la classe I_3, etc. Pour construire un code BCH, il suffit de choisir de façon judicieuse certaines classes cyclotomiques, et de multiplier entre eux les polynômes correspondants, pour obtenir le polynôme générateur du code. Par exemple, si on choisit les classes I_1 et I_3, on obtient la classe cyclotomique $\{1,2,3,4,6,8,12,16,17,24\}$, et on voit que c'est la plus petite classe contenant $\{1,2,3,4\}$. Par conséquent, le polynôme

$$P_1(X)P_3(X) = 1 + X + X^2 + X^3 + X^5 + X^6 + X^8 + X^9 + X^{10}$$

engendre un code BCH de distance assignée 5. Des exemples plus conséquents sont facilement constructibles à partir du programme MAPLE présenté à la section 4, annexe B, et qui permet de construire des codes BCH de paramètres arbitraires. Avant d'étudier les relations qui peuvent exister entre δ et la capacité de correction du code, on va introduire un formalisme matriciel qui va naturellement conduire à l'utilisation de la transformée de Fourier.

Le polynôme P que nous venons de construire est le polynôme de $\mathbb{F}_p[X]$ de plus bas degré ayant pour racine $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$. En conséquence, les polynômes $Q \in \mathbb{F}_p[X]$ qui constituent le code sont donc les polynômes de degré inférieur à n qui vérifient

$$\forall i \in \{1, \ldots, \delta - 1\}, \quad Q(\alpha^i) = 0. \tag{3.8}$$

Matriciellement, si on note $q = \{q_0, \ldots, q_{n-1}\}$ le vecteur constitué des coefficients de Q, l'équation précédente est équivalente à

$$Aq = 0 \qquad \text{avec} \qquad A \stackrel{\text{déf.}}{=} \left\{\alpha^{ij}\right\}_{i=0,\ldots,\delta-1}^{j=0,\ldots,n-1} \in M_{\delta-1,n-1}(K).$$

On peut aussi utiliser le langage de la transformée de Fourier discrète. On note $\mathscr{F}$ la transformée obtenue avec la racine $n^{\text{ième}}$ primitive α^{-1}, comme définie à l'équation (1.2) (en prenant $\zeta = \alpha^{-1}$). La condition (3.8) devient alors

$$\forall i \in \{1, \ldots, \delta - 1\}, \quad Q(\alpha^i) = \widehat{q}[i] \stackrel{\text{déf.}}{=} \mathscr{F}(q)[i] = 0.$$

Autrement dit, le code est maintenant défini en termes de conditions spectrales, de façon plus précise, par la nullité de $\delta - 1$ composantes du vecteur transformé. En employant la notion de matrice de contrôle introduite à la remarque 3.10, on peut donc dire que les $\delta - 1$ lignes de la matrice de Fourier constitue une matrice de contrôle pour le code BCH considéré.

***Remarque* 3.21. (Dualité et distance minimale).** Intuitivement, on commence à comprendre quel effet peut avoir l'utilisation d'un δ élevé. En forçant le vecteur transformé à avoir un support très réduit (par les conditions de nullité que nous venons d'expliciter), on force aussi le vecteur d'origine (c'est-à-dire un mot du code) à avoir beaucoup de composantes non nulles. Ceci est en accord avec le principe d'incertitude discret, tel qu'il est énoncé à l'exercice I.11 (le résultat s'étend sans difficulté au cas de la transformée de Fourier à valeur dans un corps fini). De ce fait, nous allons obtenir une grande distance minimale pour le code, qui va donc avoir un taux de correction élevé. Encore une fois, nous voyons apparaître le principe de dualité que nous avons mentionné au paragraphe 1.4, chap. IV. Précisons un peu tout ceci en indiquant un résultat qui nous donne un minorant sur le taux de correction du code.

Proposition 3.22 (Distance minimale d'un code BCH). *La distance minimale du code $\mathscr{C}$ construit est au moins δ.*

Démonstration. Il s'agit de montrer que la boule de centre 0 et de rayon $r \overset{\text{déf.}}{=} \delta - 1$ ne contient que 0. Soit $Q \in \mathscr{C}$, qui est donc un polynôme de degré au plus $n-1$. On suppose qu'il est dans cette boule, donc il a au plus r coefficients non nuls. Si on l'écrit sous la forme

$$Q(X) = a_1 X^{b_1} + \cdots + a_r X^{b_r},$$

le fait qu'il appartienne à $\mathscr{C}$ implique

$$\forall i \in \{1, \ldots, r\}, \quad Q(\alpha^i) = a_1 \alpha^{ib_1} + \cdots + a_r \alpha^{ib_r}.$$

Ceci signifie que le vecteur $a \overset{\text{déf.}}{=} \{a_1, \ldots, a_r\} \in \mathbb{C}^r$ satisfait le système linéaire $Ma = 0$, où $M \overset{\text{déf.}}{=} \left\{\alpha^{ib_j}\right\}_{1 \leqslant i,j \leqslant r}$. On constate que c'est une matrice de *Vandermonde*, et comme les α^{b_j} sont distincts, elle est inversible. Ceci implique donc que $a = 0$, ce qu'il fallait démontrer. □

Dans la suite, pour simplifier les explications, on supposera que $\delta = 2t + 1$, de sorte que le code est au moins t-correcteur, puisque alors $\lfloor (\delta - 1)/2 \rfloor = t$. On peut remarquer que dans le cas où $p = 2$, on peut toujours supposer que $\delta = 2t + 1$, puisque la partie $\{1, \ldots, \delta - 1\}$ peut être supposée stable par multiplication par $p = 2$.

3.5 Décodage par transformée de Fourier

L'un des intérêts des codes BCH que nous venons de construire est que l'on dispose d'algorithmes simples et rapides pour les décoder. Par exemple une méthode utilisant *l'algorithme d'Euclide étendu* est présentée dans le livre de DEMAZURE [24]. Dans ce paragraphe, nous allons présenter un autre algorithme, fondé sur la description du code en termes de transformée de Fourier discrète.

On suppose que l'on vient de recevoir un mot codé $x' = x + \varepsilon$, où x représente le mot d'origine, et ε l'erreur de transmission. Notre but est de retrouver le mot x, ou de façon équivalente, de déterminer l'erreur ε, que l'on écrit sous la forme

$$\varepsilon(X) \stackrel{\text{déf.}}{=} \varepsilon_0 + \varepsilon_1 X + \cdots + \varepsilon_{n-1} X^{n-1}.$$

ε est inconnu, mais pour avoir une chance de pouvoir résoudre ce problème, on suppose tout de même qu'il vérifie $w(\varepsilon) \leqslant t$. On rappelle que l'on a supposé que $\delta = 2t + 1$, de sorte que la proposition 3.22 nous assure que le problème est bien posé. Mais il nous faut trouver un moyen de le résoudre de manière efficace.

Nous savons, par la définition du code en termes de transformée de Fourier, que

$$\forall i \in \{1, \ldots, 2t\}, \quad \widehat{\varepsilon}[i] = \widehat{x'}[i] - \widehat{x}[i] = \widehat{x'}[i].$$

Nous savons donc calculer $2t = \delta - 1$ coefficients du vecteur $\widehat{\varepsilon}$. Il reste encore à calculer les autres, pour pouvoir, par transformée de Fourier inverse, retrouver l'erreur ε. Pour ce faire, on introduit une inconnue intermédiaire, un autre polynôme.

Définition 3.23 (Polynôme localisateur d'erreurs). On note

$$J \stackrel{\text{déf.}}{=} \{i \in \{0, \ldots, n-1\} \setminus \varepsilon_i \neq 0\}.$$

Nous avons déjà supposé que $\text{Card}(J) \leqslant t$, puisque $w(x) \leqslant t$. On appelle *polynôme localisateur d'erreurs*, et on note σ, le polynôme

$$\sigma(Z) \stackrel{\text{déf.}}{=} \prod_{i \in J} (1 - \alpha^i Z) \stackrel{\text{déf.}}{=} 1 + \sigma_1 Z + \cdots + \sigma_t Z^t.$$

Le polynôme localisateur d'erreurs est donc un polynôme de degré au plus t, comportant t coefficients a priori inconnus. Les inverses des racines de σ correspondent aux α^i, où i est la position d'une erreur dans le mot transmis.

On remarque que les polynômes ε et $\widehat{\sigma}$ possèdent une propriété d'orthogonalité, au sens que

– si $s \in J$, $\widehat{\sigma}[s] \stackrel{\text{déf.}}{=} \sigma(\alpha^{-s}) = 0$ et $\varepsilon[s] \neq 0$.
– si $s \notin J$, $\widehat{\sigma}[s] \stackrel{\text{déf.}}{=} \sigma(\alpha^{-s}) \neq 0$ et $\varepsilon[s] = 0$.

On peut résumer ceci par l'équation $\widehat{\sigma} \cdot \varepsilon = 0$, où on a noté $\cdot$ la multiplication coefficient par coefficient des polynômes. En utilisant le théorème de convolution (qui est encore valable pour une transformée sur un corps fini, comme l'explique la proposition 1.3), on obtient par passage à la transformée de Fourier l'équation de convolution

$$\mathscr{F}^{-1}(\widehat{\sigma} \cdot \varepsilon) = \sigma * \mathscr{F}^{-1}(\varepsilon) = \frac{1}{N} \sigma * \widehat{\varepsilon}^{\natural} = 0.$$

On rappelle que $\widehat{\varepsilon}^{\natural}$, considéré comme un vecteur, est obtenu conformément à la définition 5.1, chap. III. C'est donc le vecteur $\{\widehat{\varepsilon}[0], \widehat{\varepsilon}[n-1], \ldots, \widehat{\varepsilon}[1]\}$. Il suffit ensuite de remplacer la convolution de vecteurs par la multiplication modulo $Z^n - 1$ des polynômes, pour obtenir une équation polynomiale assez complexe :

$$\left(1 + \sigma_1 Z + \cdots + \sigma_t Z^t\right)\left(\widehat{\varepsilon}_0 + \widehat{\varepsilon}_{n-1} Z + \cdots + \widehat{\varepsilon}_1 Z^{n-1}\right) \mod Z^n - 1 = 0.$$

On peut remplacer cette équation polynomiale en deux systèmes d'équations linéaires :

$$(\mathscr{S}_1)\begin{cases} \widehat{\varepsilon}_0 + \widehat{\varepsilon}_1\sigma_1 + \widehat{\varepsilon}_2\sigma_2 + \cdots + \widehat{\varepsilon}_t\sigma_t = 0 \\ \widehat{\varepsilon}_{n-1} + \widehat{\varepsilon}_0\sigma_1 + \widehat{\varepsilon}_1\sigma_2 + \cdots + \widehat{\varepsilon}_{t-1}\sigma_t = 0 \\ \vdots \\ \widehat{\varepsilon}_{n-i} + \widehat{\varepsilon}_{n-i+1}\sigma_1 + \widehat{\varepsilon}_{n-i+2}\sigma_2 + \cdots + \widehat{\varepsilon}_{t-i}\sigma_t = 0 \\ \vdots \\ \widehat{\varepsilon}_{t+1} + \widehat{\varepsilon}_{t+2}\sigma_1 + \widehat{\varepsilon}_{t+3}\sigma_2 + \cdots + \widehat{\varepsilon}_{2t+1}\sigma_t = 0 \end{cases}$$

$$(\mathscr{S}_2)\begin{cases} \widehat{\varepsilon}_t + \widehat{\varepsilon}_{t+1}\sigma_1 + \widehat{\varepsilon}_{t+2}\sigma_2 + \cdots + \widehat{\varepsilon}_{2t}\sigma_t = 0 \\ \vdots \\ \widehat{\varepsilon}_2 + \widehat{\varepsilon}_3\sigma_1 + \widehat{\varepsilon}_4\sigma_2 + \cdots + \widehat{\varepsilon}_{t+2}\sigma_t = 0 \\ \widehat{\varepsilon}_1 + \widehat{\varepsilon}_2\sigma_1 + \widehat{\varepsilon}_3\sigma_2 + \cdots + \widehat{\varepsilon}_{t+1}\sigma_t = 0 \end{cases}$$

Comme on connaît les valeurs de $\widehat{\varepsilon}_1, \ldots, \widehat{\varepsilon}_{2t}$, le système $(\mathscr{S}_2)$ nous permet de calculer $\sigma_1, \ldots, \sigma_t$ de façon très simple (le système est en fait triangulaire). On peut maintenant utiliser le système $(\mathscr{S}_1)$ pour trouver $\widehat{\varepsilon}_0, \widehat{\varepsilon}_{2t}, \ldots, \varepsilon_{n-1}$, puisque l'on dispose de $n-t$ équations pour seulement $n-2t$ inconnues.

***Remarque* 3.24.** Une fois que l'on a calculé $\sigma_1, \ldots, \sigma_t$ (en résolvant le système $\mathscr{S}_2$), une autre alternative s'offre à nous. En effet, puisque l'on connaît σ, il est possible de tester, pour $i = 0, \ldots, n-1$, si $\sigma(\alpha^{-i}) = 0$, et ainsi de détecter les positions des erreurs. Le système $\mathscr{S}_1$ étant très simple à résoudre, les deux méthodes sont cependant équivalentes.

Cette méthode de décodage est implémentée à l'aide de MAPLE au paragraphe 4, annexe B. Elle utilise les routines définies au paragraphe 1, annexe B, pour réaliser des transformées de Fourier à valeur dans $\mathbb{F}_p$.

4 Codes correcteurs et dualité sur un groupe abélien fini

Dans cette dernière partie, nous allons voir comment les outils développés dans les chapitres précédents peuvent être utiles pour étudier les caractéristiques d'un code. Ainsi, vont tour à tour intervenir les notions d'orthogonalité, de dualité sur un groupe fini, et bien sûr les diverses transformations qui sont liées à toutes ces notions (transformée de Walsh et de Fourier entre autres).

Le livre fondamental sur l'étude combinatoire des codes correcteurs est celui de MACWILLIAMS et SLOANE [50]. Ce paragraphe reprend les principaux résultats sur la dualité, en les exposant à travers le langage qui nous est maintenant familier, celui des algèbres de groupe $\mathbb{C}[G]$ et des caractères. Il est important de jeter un coup œil aux exercices proposés, l'exercice VI.11 par exemple, propose une construction de codes non linéaires très efficaces.

4.1 Polynômes énumérateurs de poids

Dans la suite, nous allons nous restreindre à l'étude des codes binaires, mais tous les résultats donnés s'étendent au cas des codes définis sur un corps fini $\mathbb{F}_q$ quelconque. Il

faut utiliser les caractères additifs appropriés, et on se reportera à l'exercice II.8 pour obtenir la formule de MacWilliams correspondante. Dans les pages suivantes, nous allons considérer $\mathscr{C}$, un code linéaire sur $\mathbb{F}_2$, de taille n et de dimension m.

Définition 4.1 (Orthogonal d'un code). On rappelle que l'on dispose d'une forme bilinéaire symétrique non dégénérée canonique sur $(\mathbb{F}_2)^n$, déjà introduite au paragraphe 3.2, chap. II :

$$\forall (x,y) \in (\mathbb{F}_2)^n \times (\mathbb{F}_2)^n, \quad \langle x,y \rangle \stackrel{\text{déf.}}{=} \sum_{i=0}^{n-1} x_i y_i. \tag{4.1}$$

On note $\mathscr{C}^\perp$ le *code orthogonal* de $\mathscr{C}$ pour cette forme bilinéaire, c'est-à-dire :

$$\mathscr{C}^\perp \stackrel{\text{déf.}}{=} \{x \in (\mathbb{F}_2)^n \setminus \forall y \in \mathscr{C},\ \langle x,y \rangle = 0\}.$$

C'est donc un code de taille n et de dimension $n-m$.

***Remarque* 4.2. (Code dual).** On parle aussi de *code dual* pour désigner $\mathscr{C}^\perp$. Cette dénomination est très naturelle, puisque nous avons déjà vu à la section 3, chap. II les similitudes (et même l'identité dans le cas de $(\mathbb{F}_2)^n$) entre les notions d'orthogonalité, de dualité sur un espace vectoriel, et de dualité sur un groupe abélien fini.

***Remarque* 4.3. (Code auto-dual).** Il est important d'insister sur le fait que, même si on a toujours $\dim(\mathscr{C}) + \dim(\mathscr{C}^\perp) = n$, un code et son dual ne sont en général pas supplémentaires. Il arrive parfois que l'on ait $\mathscr{C} \subset \mathscr{C}^\perp$, et l'on parle de code *auto-orthogonal*. Lorsque l'on a $\mathscr{C}^\perp = \mathscr{C}$, on dit que le code est *auto-dual*. Cette notion est étudiée plus en détail à l'exercice VIII.9. Le fait que la matrice génératrice puisse en même temps servir de matrice de contrôle permet de simplifier les procédures de décodage.

Définition 4.4 (Polynôme énumérateur). On note $W_{\mathscr{C}} \in \mathbb{Z}[X,Y]$ le *polynôme énumérateur de poids* de $\mathscr{C}$, qui est défini par

$$W_{\mathscr{C}}(X,Y) \stackrel{\text{déf.}}{=} \sum_{i=0}^{n} A_i X^{n-i} Y^i,$$

où on a noté $\{A_i\}_{i=0}^n$ la répartition de poids de $\mathscr{C}$, définie par l'équation (3.1).

Le résultat fondamental pour la détermination du polynôme énumérateur est l'identité de *MacWilliams*, déjà démontrée au théorème 3.6, chap. II, et que l'on rappelle ici.

Théorème 4.5 (Identité de MacWilliams). *On a*

$$W_{\mathscr{C}^\perp}(X,Y) = \frac{1}{2^m} W_{\mathscr{C}}(X+Y, X-Y).$$

Plusieurs exercices proposent d'utiliser tous ces outils dans le but d'obtenir des informations de nature combinatoire sur les codes correcteurs. L'exercice VI.8 propose de calculer les polynômes énumérateurs pour les codes de Hamming. L'exercice VI.12 étudie la répartition des poids des mots dans les codes *MDS* (c'est-à-dire ceux qui sont optimaux pour la borne de Singleton). Enfin, l'exercice VIII.9 étudie les codes auto-duaux, en employant les techniques de la théorie des invariants pour exploiter les identités de MacWilliams.

4.2 Algèbre d'un groupe et codes correcteurs

Toutes ces techniques combinatoires sont donc très utiles pour analyser la structure d'un code linéaire. Elles tombent cependant en défaut dès qu'il s'agit d'étudier un code non linéaire, c'est-à-dire d'étudier la répartition des mots d'un ensemble $\mathscr{C} \subset (\mathbb{F}_2)^n$. On rappelle que la notion essentielle pour étudier un code non linéaire n'est pas la répartition de poids $\{A_i\}_{i=0}^n$ mais la répartition de distance $\{B_i\}_{i=0}^n$, définie à l'équation (3.2). Dans le cas non linéaire, ces deux répartitions ne coïncident pas, et il peut être très complexe de les déterminer. Nous allons cependant voir qu'en utilisant la transformée de Fourier sur l'algèbre $\mathbb{C}[(\mathbb{F}_2)^n]$, on peut obtenir de nombreuses informations sur $\mathscr{C}$. Pour plus de commodité, on notera $G \overset{\text{déf.}}{=} (\mathbb{F}_2)^n$, qui peut être vu comme un groupe additif. On rappelle que $\mathbb{C}[G]$ désigne l'algèbre des fonctions de G dans $\mathbb{C}$. Les caractères de G sont notés, pour $a \in G$,

$$\chi_a : \left\{ \begin{array}{ccc} G & \longrightarrow & \mathbb{C}^* \\ x & \longmapsto & (-1)^{\langle a,x \rangle} \end{array} \right. .$$

On rappelle la définition de la transformée de Fourier de $f \in \mathbb{C}[G]$, déjà donnée en (4.1), chap. I :

$$\widehat{f} : \left\{ \begin{array}{ccc} G & \longrightarrow & \mathbb{C} \\ a & \longmapsto & \sum_{x \in G} \chi_a(x) f(x) \end{array} \right. .$$

Ces définitions bien en tête, expliquons comment on peut représenter un ensemble de vecteurs de $(\mathbb{F}_2)^n = G$ comme une fonction de $\mathbb{C}[G]$.

Définition 4.6 (Fonction indicatrice). Soit $\mathscr{C} \subset G$ (qui n'est pas nécessairement un code linéaire). On définit la fonction indicatrice de $\mathscr{C}$ par

$$f_{\mathscr{C}} \overset{\text{déf.}}{=} \sum_{x \in \mathscr{C}} \delta_x.$$

C'est la fonction qui vaut 1 sur $\mathscr{C}$, et zéro partout ailleurs. On peut ainsi identifier les sous-ensembles de G (c'est-à-dire les codes quelconques) à des fonctions de $\mathbb{C}[G]$.

Ces fonctions indicatrices sont étudiées en détail à l'exercice I.4. Cet exercice fait en particulier le lien entre les propriétés spectrales de la fonction $f_{\mathscr{C}}$ et la « régularité » de $\mathscr{C}$. Attardons-nous un instant sur le cas des codes linéaires. La question qui se pose naturellement est de savoir s'il y a un rapport entre la fonction indicatrice de $\mathscr{C}$ et celle de $\mathscr{C}^\perp$. Nous avons déjà vu à l'équation (3.6), chap. II, que

$$x \in \mathscr{C}^\perp \quad \Leftrightarrow \quad \forall t \in \mathscr{C},\ \langle x,t \rangle = 0 \quad \Leftrightarrow \quad \forall t \in \mathscr{C},\ \chi_x(t) = 1. \tag{4.2}$$

Cette propriété est fondamentale ; elle va nous permettre de calculer la transformée de Fourier de la fonction $f_{\mathscr{C}}$.

Proposition 4.7 (Transformée d'une fonction indicatrice). *Soit $\mathscr{C}$ un code linéaire. On a alors*

$$\widehat{f_{\mathscr{C}}} = |\mathscr{C}| f_{\mathscr{C}^\perp}.$$

Démonstration. Si $x \in \mathscr{C}^\perp$, alors l'équation (4.2) nous dit que

$$\widehat{f_{\mathscr{C}}}(x) = \sum_{t \in \mathscr{C}} \chi_x(t) = \sum_{t \in \mathscr{C}} 1 = |\mathscr{C}|.$$

De même, si $x \notin \mathscr{C}^\perp$, l'équation (4.2) nous dit qu'il existe $t_0 \in \mathscr{C}$ tel que

$$\chi_x(t_0) \neq 1, \quad \text{c'est-à-dire} \quad \chi_x(t_0) = -1.$$

On obtient donc

$$-\widehat{f_{\mathscr{C}}}(x) = \sum_{t\in\mathscr{C}} \chi_x(t_0)\chi_x(t) = \sum_{t\in\mathscr{C}} \chi_x(t+t_0) = \widehat{f_{\mathscr{C}}}(x),$$

la dernière égalité provenant du fait que $\mathscr{C}$ est un sous-espace vectoriel de G (ou un sous-groupe additif, en changeant de vocabulaire). On a donc bien

$$x \notin \mathscr{C}^\perp \Longrightarrow \widehat{f_{\mathscr{C}}}(x) = 0. \qquad \square$$

Notre but est d'étendre la notion de dualité à des ensembles $\mathscr{C}$ quelconques. Il serait tentant d'étudier l'espace formé des vecteurs orthogonaux à $\mathscr{C}$. Cependant, la construction qui va permettre d'obtenir des informations sur $\mathscr{C}$ est plus complexe. En effet, il s'agit de construire une fonction duale de la fonction indicatrice $f_{\mathscr{C}}$. Dans un souci de généralité, nous allons définir la fonction duale d'un élément quelconque de $\mathbb{C}[G]$, à partir du moment où sa moyenne n'est pas nulle.

Définition 4.8 (Fonction duale). Soit $f \in \mathbb{C}[G]$ telle que $M_f \overset{\text{déf.}}{=} \sum_{x\in G} f(x) \neq 0$. On définit la *fonction duale* de f, notée $f^\perp$ par $f^\perp \overset{\text{déf.}}{=} \frac{1}{M_f}\widehat{f}$.

On peut donc énoncer le résultat important suivant.

Proposition 4.9. *Si $\mathscr{C}$ est un code linéaire, on a $(f_{\mathscr{C}})^\perp = f_{\mathscr{C}^\perp}$.*

Démonstration. Il suffit de remarquer que pour $f = f_{\mathscr{C}}$, on a $M_f = |\mathscr{C}|$. Il ne reste plus qu'à appliquer la proposition 4.7. $\square$

Toujours dans l'idée d'étendre les notions propres aux codes linéaires à des codes plus généraux, définissons le polynôme énumérateur de poids d'une fonction.

Définition 4.10 (Polynôme énumérateur d'une fonction). Soit $f \in \mathbb{C}[G]$.
Pour $i = 0, \ldots, n$, on définit

$$A_i \overset{\text{déf.}}{=} \sum_{w(x)=i} f(x),$$

ainsi que le polynôme énumérateur de poids de f :

$$W_f(X,Y) \overset{\text{déf.}}{=} \sum_{i=0}^{n} A_i X^{n-i}Y^i.$$

On constate que ce polynôme généralise celui défini en 4.4, puisque l'on a, pour $\mathscr{C}$ un code linéaire,

$$W_{\mathscr{C}}(X,Y) = W_{f_{\mathscr{C}}}(X,Y).$$

La question est de savoir si les identités de MacWilliams sont encore valides. La réponse est positive, et on peut reformuler ces identités avec le vocabulaire des codes correcteurs.

Théorème 4.11 (Identité de MacWilliams pour des fonctions). *Soit $f \in \mathbb{C}[G]$ telle que $M_f \neq 0$. On a alors*

$$W_{f^\perp}(X,Y) = \frac{1}{M_f} W_f(X+Y, X-Y). \tag{4.3}$$

Démonstration. On a, en utilisant la définition de $f^\perp$,

$$\begin{aligned} W_{f^\perp}(X,Y) &= \sum_{x\in G} \frac{1}{M_f} \left(\sum_{y\in G} \chi_x(y) f(y) \right) X^{n-w(x)}Y^{w(x)} \\ &= \frac{1}{M_f} \sum_{y\in G} f(y) \sum_{x\in G} \chi_x(y) X^{n-w(x)}Y^{w(x)}. \end{aligned}$$

Or nous avons déjà calculé la somme interne lors de la preuve du théorème de MacWilliams 3.6, chap. II :

$$\sum_{x\in G} \chi_x(y) X^{n-w(x)} Y^{w(x)} = (X+Y)^{n-w(y)}(X-Y)^{w(y)},$$

et on arrive bien au résultat voulu. □

On voit que si l'on applique l'identité (4.3) à la fonction indicatrice d'un code linéaire, on retrouve l'identité de MacWilliams pour les codes linéaires.

4.3 Etude combinatoire de codes quelconques

Nous allons maintenant voir comment on peut appliquer toutes ces constructions effectuées sur l'algèbre $\mathbb{C}[G]$ à l'étude d'un code. Soit donc $\mathscr{C} \subset (\mathbb{F}_2)^n$ un ensemble quelconque. On peut voir $\mathscr{C}$ comme un code correcteur quelconque, non nécessairement linéaire.

Définition 4.12 (Fonction de distance). La *fonction de distance* $D_{\mathscr{C}} \in \mathbb{C}[G]$ est définie par :

$$D_{\mathscr{C}} \overset{\text{déf.}}{=} \frac{1}{|\mathscr{C}|} f_{\mathscr{C}} * f_{\mathscr{C}},$$

où $*$ désigne le produit de convolution des fonctions sur G, comme nous l'avons défini à l'équation (4.6), chap. I.

Cette fonction peut se calculer de façon explicite :

$$D_{\mathscr{C}} = \frac{1}{|\mathscr{C}|} \sum_{x\in\mathscr{C}} \sum_{y\in\mathscr{C}} \delta_{x+y}. \tag{4.4}$$

En particulier, on voit que si $\mathscr{C}$ est un code linéaire, on a $D_{\mathscr{C}} = f_{\mathscr{C}}$. Dans la suite, on note le polynôme énumérateur de poids de $D_{\mathscr{C}}$ sous la forme

$$W_{D_{\mathscr{C}}} = \sum_{i=0}^{n} D_i X^{n-i} Y^i.$$

On a alors la proposition suivante, qui permet d'obtenir simplement des informations sur la répartition des mots de $\mathscr{C}$ les uns par rapport aux autres.

Proposition 4.13 (Répartition de distance). $\{D_i\}_{i=0}^n$ *représente la répartition de distance de* $\mathscr{C}$*, c'est-à-dire :*

$$D_i = B_i \overset{\text{déf.}}{=} \frac{1}{|\mathscr{C}|} \#\left\{(x,y) \in \mathscr{C}^2 \setminus d(x,y) = i\right\},$$

où $d(x,y)$ *désigne la distance de Hamming entre* x *et* y*.*

Démonstration. L'équation (4.4) peut se mettre sous la forme

$$D_{\mathscr{C}} = \sum_{i=0}^{n} \frac{1}{|\mathscr{C}|} \sum_{d(x,y)=i} \delta_{x-y}. \tag{4.5}$$

Or on a, par définition, $D_i = \sum_{w(z)=i} D_{\mathscr{C}}(z)$. Ce qui donne donc, en utilisant (4.5),

$$D_i = \frac{1}{|\mathscr{C}|} \sum_{d(u,v)=i} 1,$$

ce qui est exactement le résultat voulu. □

***Remarque* 4.14.** On peut remarquer que le raisonnement effectué pour démontrer la proposition 4.13 utilise le fait que, pour $(x,y) \in (\mathbb{F}_2^n)^2$, on a $x+y=x-y$. Pour étendre cette proposition à un corps fini $\mathbb{F}_q$ quelconque, il faut introduire, pour $f \in \mathbb{C}[(\mathbb{F}_q)^n]$ la fonction symétrisée $\widetilde{f} : x \mapsto f(-x)$. On doit alors définir la fonction de distance comme suit :

$$D_{\mathscr{C}} \overset{\text{déf.}}{=} \frac{1}{|\mathscr{C}|} f_{\mathscr{C}} * \widetilde{f_{\mathscr{C}}}.$$

On vérifie sans problème que la proposition 4.13 est encore valide, tout comme le reste des résultats de ce paragraphe.

Dans le cas linéaire, on sait donc que le polynôme énumérateur de $D_{\mathscr{C}}^{\perp} = D_{\mathscr{C}^{\perp}}$ va représenter la répartition de distance du code dual $\mathscr{C}^{\perp}$. Il est donc naturel d'étudier la généralisation de ce procédé aux codes non linéaires. Ceci signifie donc étudier le polynôme énumérateur de la fonction $D_{\mathscr{C}}^{\perp}$, qui n'a a priori aucune raison d'avoir des propriétés intéressantes. Pour calculer cette fonction, remarquons que, d'après (4.4),

$$M_{D_{\mathscr{C}}} = \sum_{x \in \mathscr{C}} \sum_{y \in \mathscr{C}} \frac{1}{|\mathscr{C}|} = |\mathscr{C}|. \tag{4.6}$$

Le théorème 4.11 nous permet de calculer, à partir de $W_{D_{\mathscr{C}}}$, le polynôme de la fonction duale. Pour l'instant, contentons nous de l'écrire sous la forme

$$W_{D_{\mathscr{C}}^{\perp}} = \sum_{i=0}^{n} B_i' X^{n-i} Y^i. \tag{4.7}$$

On peut remarquer que, par définition de la fonction duale et en utilisant (4.6), les B_i' valent

$$B_i' = \frac{1}{|\mathscr{C}|} \sum_{w(x)=i} \widehat{D_{\mathscr{C}}}(x),$$

même s'il est moins simple d'utiliser cette expression plutôt que le résultat du théorème 4.11. A priori, les nombres B_i' ne possèdent aucune propriété particulière. En particulier, il n'y a aucune raison pour que les B_i' représentent la répartition de distance d'un code. En effet, $\mathscr{C}$ n'étant pas linéaire, le code dual $\mathscr{C}^{\perp}$ n'est pas défini : tout repose donc sur la *fonction* duale $f_{\mathscr{C}}^{\perp}$. Par exemple, B_i' n'a aucune raison d'être entier ! Voici cependant un résultat simple qui précise les choses.

Proposition 4.15. *Les B_i' sont des nombres rationnels positifs.*

Démonstration. Il suffit d'utiliser le théorème de convolution 4.15, chap. I pour réécrire les B_i' sous la forme

$$B_i' = \frac{1}{|\mathscr{C}|^2} \sum_{w(x)=i} \mathscr{F}(f_{\mathscr{C}} * f_{\mathscr{C}})(x) = \frac{1}{|\mathscr{C}|^2} \sum_{w(x)=i} \widehat{f_{\mathscr{C}}}(x)^2 \geqslant 0. \qquad \square$$

Cette propriété, en apparence anodine, permet de démontrer une inégalité très fine sur la taille maximum des codes de taille n et de distance minimale d. Ce développement nécessite l'introduction des polynômes de *Krawtchouk*, qui sont définis au début de l'exercice VI.12. L'exercice VI.13 détaille les étapes qui permettent de démontrer cette inégalité.

Exemple 4.16. Les *codes de Hadamard* sont définis à l'exercice VI.11. On considère l'exemple du code $\mathscr{A}_8$, calculé grâce aux résidus quadratiques modulo 7. Les vecteurs qui le composent sont donnés à l'équation (5.2). C'est un code simplexe contenant le vecteur nul, donc sa répartition de poids est égale à sa répartition de distance :

$$A_0 = B_0 = 1, \quad A_4 = B_4 = 7 \quad \text{et} \quad \forall i \notin \{0,4\}, \quad A_i = B_i = 0.$$

On obtient donc le polynôme énumérateur de poids suivant :

$$W_{D_{\mathscr{A}_8}}(X,Y) = X^7 + 7X^3Y^4.$$

La répartition de distance duale se calcule de la façon suivante :

$$\begin{aligned} W_{D^\perp_{\mathscr{A}_8}} &= \frac{1}{8} W_{D_{\mathscr{A}_8}}(X+Y, X-Y) = \frac{1}{8}\left((X+Y)^7 + 7(X+Y)^3(X-Y)^4\right) \\ &= X^7 + 7X^4Y^3 + 7X^3Y^4 + Y^7. \end{aligned}$$

Ce qui donne donc

i	1	3	4	7
B'_i	1	7	7	1

.

Pour un code de Hadamard H_{12}, on obtient

$$\begin{aligned} W_{D^\perp_{\mathscr{A}_{12}}}(X,Y) &= \frac{1}{12} W_{D_{\mathscr{A}_{12}}}(X+Y, X-Y) \\ &= \frac{1}{12}\left((X+Y)^{11} + 11(X+Y)^5(X-Y)^6\right) \\ &= X^{11} + \frac{55}{3}Y^3X^8 + \frac{110}{3}Y^4X^7 + \frac{88}{3}Y^5X^6 + \frac{88}{3}Y^6X^5 \\ &\quad + \frac{110}{3}Y^7X^4 + \frac{55}{3}Y^8X^3 + Y^{11}. \end{aligned}$$

Ce qui donne donc

i	1	3	4	5	6	7	8	11
B'_i	1	$18\frac{1}{3}$	$39\frac{2}{3}$	$29\frac{1}{3}$	$29\frac{1}{3}$	$36\frac{2}{3}$	$18\frac{1}{3}$	1

On constate que l'on a bien $B'_i \geqslant 0$ et que

$$\sum_{i=0}^{n} B'_i = \frac{512}{3} = \frac{2^{11}}{|\mathscr{A}_{12}|}.$$

Ceci est tout à fait normal, puisque $\mathscr{C}$ est un code qui contient 0 et que

$$\mathscr{W}_{D^\perp_{\mathscr{C}}}(1,1) = \frac{1}{|\mathscr{C}|}\mathscr{W}_{D_{\mathscr{C}}}(2,0) = \frac{2^n}{|\mathscr{C}|}.$$

5 Exercices

Exercice VI.1 (Polynômes cyclotomiques). En utilisant MAPLE, montrer que les plus petites valeurs de n pour lesquelles Φ_n possède un coefficient égal à $\pm 1, \pm 2, \pm 3, \ldots$ sont

0, 105, 385, 1365, 1785, 2805, 3135, 6545, 6545, 10465, 10465,
10465, 10465, 10465, 11305, 11305, 11305, 11305, 11305,
11305, 11305, 15015, 11305, 17255, 17255, 20615, 20615, ...

Exercice VI.2 (Condition d'existence d'une racine principale). Cet exercice détaille les étapes de la démonstration de la proposition 2.6. Il s'agit de trouver une condition pour que l'on puisse construire une transformée de Fourier sur $\mathbb{Z}/m\mathbb{Z}$, où m s'écrit sous la forme $m = p_1^{k_1} \times \cdots \times p_r^{k_r}$ et les p_i sont des nombres premiers distincts. On rappelle le théorème d'Euler : si x est inversible dans $\mathbb{Z}/s\mathbb{Z}$, alors $x^{\Phi(s)} = 1 \mod s$. On a noté $\Phi(s)$ la fonction d'Euler, c'est-à-dire le nombre d'inversibles dans $\mathbb{Z}/s\mathbb{Z}$.

1. Expliquer pourquoi il existe une racine $n^{\text{ième}}$ principale dans $\mathbb{F}_p$, pour p premier, si et seulement si $n|p-1$. Soit alors ζ une telle racine, que l'on assimile à son représentant dans $\{0,\ldots,p-1\}$, vu comme sous-ensemble de $\mathbb{Z}/p^r\mathbb{Z}$.
2. On se place dans $\mathbb{Z}/p^r\mathbb{Z}$, et on note $\zeta_0 \stackrel{\text{déf.}}{=} \zeta^{p^{r-1}}$. Montrer que ζ est inversible dans $\mathbb{Z}/p^r\mathbb{Z}$ puis que $\zeta_0^{p-1} = 1$.
3. Montrer que dans $\mathbb{F}_p$, et pour $s = 1,\ldots,n-1$,

$$\zeta^s - 1 = \left(\zeta^{p^{r-1}}\right)^s - 1 = \zeta_0^s - 1.$$

   En déduire que $\zeta_0^s - 1$ est premier avec p^r et donc que ζ_0 est une racine $n^{\text{ième}}$ principale de l'unité dans $\mathbb{Z}/p^r\mathbb{Z}$.
4. En utilisant le théorème chinois, conclure.

Exercice VI.3 (Racines dyadiques principales). On souhaite démontrer la proposition 2.7, qui donne un critère simple pour trouver une racine $(2^s)^{\text{ième}}$ principale dans un anneau commutatif A.

1. Montrer que pour que $\zeta \in A$ soit une racine $(nm)^{\text{ième}}$ principale de l'unité, il faut et il suffit que ζ^m soit une racine $n^{\text{ième}}$ principale, et que ζ^n soit une racine $m^{\text{ième}}$ principale.
2. Quelles sont les racines carrées principales de l'unité ? Préciser à quelle condition elles existent.
3. Montrer par récurrence sur k la proposition 2.7

Exercice VI.4 (Fonctions booléennes). Cet exercice utilise la transformée de Walsh définie à la section 2, chap. II. Une fonction $\widetilde{f} : (\mathbb{F}_2)^n \to \mathbb{F}_2$ est appelée fonction booléenne à n arguments. D'une façon pratique, on peut aussi se représenter une telle fonction par la fonction réelle $f \stackrel{\text{déf.}}{=} (-1)^{\widetilde{f}}$ à valeurs dans $\{-1,1\}$, ce qui permet de calculer la transformée de Walsh $\mathscr{W}(f)$:

$$\forall k \in (\mathbb{F}_2)^n, \quad \mathscr{W}(f)(k) \stackrel{\text{déf.}}{=} \sum_{t\in(\mathbb{F}_2)^n} f(t)(-1)^{\langle t,k\rangle}.$$

Dans la suite, on jonglera entre ces deux types de représentations f et $\widetilde{f}$. Une fonction booléenne $\widetilde{f}$ est dite affine si elle s'écrit

$$\forall x \in (\mathbb{F}_2)^n, \quad \widetilde{f}(x) = \widetilde{f}_{a,b}(x) \stackrel{\text{déf.}}{=} \langle x,a\rangle + b,$$

où $a \in (\mathbb{F}_2)^n$, $b \in \mathbb{F}_2$, et $\langle\cdot,\cdot\rangle$ désigne la forme bilinéaire canonique sur $(\mathbb{F}_2)^n$, déjà rencontrée à l'équation (4.1). On va utiliser la distance $d(f,g)$ entre deux fonctions booléennes, qui est, par définition, la distance de Hamming (définition 3.3) entre les vecteurs $V(f) = \{\widetilde{f}(x)\}_{x\in\mathbb{F}_2^n}$ et $V(g) = \{\widetilde{g}(x)\}_{x\in\mathbb{F}_2^n}$ On définit alors la non-linéarité de $\widetilde{f}$ par

$$N(f) \stackrel{\text{déf.}}{=} \inf\left\{d(f,f_{a,b}) \setminus a \in (\mathbb{F}_2)^n, b \in \mathbb{F}_2\right\}.$$

1. Expliquer pourquoi toute fonction booléenne $\widetilde{f}$ peut se mettre de façon unique sous la forme polynomiale suivante :

$$\forall x \in (\mathbb{F}_2)^n, \quad \widetilde{f}(x) = b + a_0 x_0 + \cdots + a_{n-1} x_{n-1} + a_{01} x_0 x_1 + a_{02} x_0 x_2 + \cdots + a_{0\cdots n-1} x_0 \cdots x_{n-1}.$$

   Justifier de façon intuitive le terme de non-linéarité.

2. Montrer que l'on a

$$N(f) = 2^{n-1} - \frac{1}{2} \max \{ |\mathscr{W}(f)(k)| \setminus k \in (\mathbb{F}_2)^n - \{0\} \}.$$

   En déduire une méthode rapide de calcul de $N(f)$ qui utilise l'algorithme FWT.

3. Montrer que l'on a

$$N(f) \leqslant 2^{n-1} - 2^{\frac{n}{2}-1}.$$

   On suppose que n est pair. Montrer qu'une fonction $\widetilde{f}$ atteint la borne précédente si et seulement si pour tout $k \in (\mathbb{F}_2)^n$, $|\mathscr{W}(f)(k)| = 2^{\frac{n}{2}}$. Dans la littérature anglo-saxonne, on les appelle « bent functions ». Elles ont été introduites pour la première fois par ROTHAUS dans [61].

4. Pour $u \in (\mathbb{F}_2)^n$ et $v \in (\mathbb{F}_2)^m$, on pose $w = (u,v) \in (\mathbb{F}_2)^{n+m}$. Soient $\widetilde{f}$ et $\widetilde{g}$ des fonctions de n et m variables. On définit $\widetilde{h}$ une fonction de $n+m$ variables par $\widetilde{h}(w) = \widetilde{f}(u) + \widetilde{g}(v)$. Montrer que $\widetilde{h}$ est bent si et seulement si $\widetilde{f}$ et $\widetilde{g}$ le sont.
   Montrer que $\widetilde{f_0}(u_1,u_2) = u_1 u_2$ est bent. En déduire l'exsitence de fonctions bent pour n pair.

5. On nomme code de *Reed-Muller* d'ordre 1 en n variables (noté $R(1,n)$) le sous-espace vectoriel de l'espace des fonctions booléennes formé des $f_{a,b}$, pour $a \in \mathbb{F}_2^n$ et $b \in \mathbb{F}_2$. Quelle sont la dimension et la distance minimale de ce code ? La procédure de codage consiste, à partir du couple (a,b), à produire la table de vérité $V(f_{a,b})$, c'est-à-dire le vecteur $\{f_{a,b}(u)\}_{u \in (\mathbb{F}_2)^n} = F_{a,b}$. Proposer un algorithme de codage rapide. Pour $F = V(f) \in (\mathbb{F}_2)^{2^n}$, quel est le couple (a,b) tel que $d(f, f_{a,b})$ soit minimal ? En déduire un algorithme de décodage rapide.

La détermination des fonctions les moins linéaires dans le cas où n est impair est un problème ouvert. Les fonctions fortement non-linéaires sont très utilisées en cryptographie. Un exposé complet sur ce sujet est l'article de PASALIC et JOHANSSON [57].

Exercice VI.5 (Apprentissage de fonctions booléennes). Dans cet exercice, on conserve les notations de l'exercice VI.4. On se propose, en utilisant quelques notions de probabilité, d'effectuer des prédictions booléennes sur une fonction f en n'utilisant qu'une connaissance approchée de sa transformée de Walsh. Cette théorie a été développée initialement par KUSHILEVITZ et MANSOUR dans [40]. On note $\mathbb{P}$ la distribution de probabilité uniforme sur $(\mathbb{F}_2)^n$, c'est-à-dire $\forall x \in (\mathbb{F}_2)^n$, $\mathbb{P}(x) = 2^{-n}$. En représentant une fonction booléenne $\widetilde{f}$ par la fonction réelle $f = (-1)^{\widetilde{f}}$, on peut alors calculer l'espérance de f :

$$E[f] \overset{\text{déf.}}{=} \frac{1}{2^n} \sum_{x \in (\mathbb{F}_2)^n} f(x) = \frac{1}{2^n} \mathscr{W}(f)(0).$$

Enfin, on rappelle la borne de *Chernoff-Hoeffding*, que l'on pourra trouver dans [55]. Soient $X_1, \ldots, X_m$ des variables indépendantes identiquement distribuées telles que X_i soit

à valeurs dans $[-1,1]$, $E[X_i] = p$ et $S_m = \sum_{i=1}^m X_i$. Alors on a

$$\mathbb{P}\left[\left|\frac{S_m}{m} - p\right| \geqslant \lambda\right] \leqslant 2e^{-\frac{\lambda^2 m}{2}}.$$

Dans la suite, on considère f une fonction booléenne inconnue, dont on suppose seulement pouvoir accéder à des échantillons $f(x_k)$, où les x_k sont tirés au hasard de façon indépendante selon une loi uniforme. Notre but est de donner, avec aussi peu d'informations que possible, une bonne estimation de f. On dit que l'on « apprend » la fonction f.

1. Supposons qu'un des coefficients $a_\beta = \langle f, \chi_\beta \rangle = 2^{-n}\mathcal{W}(f)(\beta)$ soit très grand. Quelle est l'erreur quadratique $E[(f-h)^2]$ que l'on commet en remplaçant f par $h \stackrel{\text{déf.}}{=} a_\beta \chi_\beta$?
2. A priori, h n'a aucune raison d'être une fonction booléenne. On remplace donc h par $h_0 = \text{Signe}(h)$. Montrer que l'on a alors

$$\mathbb{P}(f(x) \neq h_0(x)) \leqslant E[(f-h)^2].$$

3. On est donc intéressé par le calcul approché de $a_\beta = E[f\chi_\beta]$. On propose d'utiliser m échantillons $f(x_k)$ et de calculer la valeur moyenne :

$$\widetilde{a_\beta} \stackrel{\text{déf.}}{=} \frac{1}{m}\sum_{k=1}^{m} f(x_k)\chi_\beta(x_k). \tag{5.1}$$

On souhaite donc approcher la fonction inconnue f par $\varphi_0 \stackrel{\text{déf.}}{=} \text{Signe}(\widetilde{a_\beta}\chi_\beta)$. Montrer que si $m \geqslant \frac{2}{\lambda^2}\ln\left(\frac{2}{\delta}\right)$ alors

$$\mathbb{P}\left(\left|a_\beta - \widetilde{a_\beta}\right| \geqslant \lambda\right) \leqslant \delta.$$

En déduire que l'on a, avec une probabilité d'au moins $1-\delta$, la majoration suivante de la probabilité d'erreur :

$$\mathbb{P}(f(x) \neq \varphi_0(x)) \leqslant 1 - a_\beta^2 + \lambda^2.$$

4. On souhaite maintenant approcher toute une classe de fonctions possédant des coefficients « hautes fréquences » faibles. On dit qu'une fonction booléenne f a un degré (α, d) si

$$\sum_{w(s)>d} a_s^2 \leqslant \alpha,$$

où $w(s)$ est le poids de Hamming d'un mot $s \in (\mathbb{F}_2)^n$. Pour tous les s tels que $w(s) \leqslant d$, on calcule $\widetilde{a_s}$ par l'équation (5.1). On considère ensuite la fonction

$$\varphi_0 \stackrel{\text{déf.}}{=} \text{Signe}(\varphi) \quad \text{où} \quad \varphi \stackrel{\text{déf.}}{=} \sum_{w(s)\leqslant d} \widetilde{a_s}\chi_s.$$

Montrer que si l'on choisit $m \geqslant \frac{2n^d}{\varepsilon}\ln\left(\frac{2n^d}{\delta}\right)$, alors on a

$$\mathbb{P}(f(x) \neq \varphi_0(x)) \leqslant \alpha + \varepsilon$$

avec une probabilité d'au moins $1-\delta$.

On pourra noter que l'exercice VIII.7 utilise la théorie des représentations pour trouver une base orthonormée de $\mathbb{C}[(\mathbb{F}_2)^n]$ qui diffère de la base de Walsh. Ceci permet d'envisager d'autres méthodes pour apprendre une fonction booléenne.

Exercice VI.6 (Matrices génératrice et de contrôle). Soit $\mathscr{C}$ un code linéaire de dimension m et de taille n sur $\mathbb{F}_q$.

1. Quel liens y a-t-il entre les matrices de contrôle et les matrices génératrices de $\mathscr{C}$ et de $\mathscr{C}^\perp$?
2. On suppose maintenant que la matrice génératrice de $\mathscr{C}$ est sous *forme systématique*, c'est-à-dire que

$$G = \begin{pmatrix} \mathrm{Id}_m \\ A \end{pmatrix} \quad \text{avec} \quad A \in (\mathbb{F}_q)^{(n-m)\times m}.$$

   Quels sont les avantages d'une telle fome ? Comment s'écrit une matrice de contrôle de $\mathscr{C}$?
3. Montrer que tout code linéaire $\mathscr{C}$ est équivalent à un code systématique. On dit que deux codes sont équivalents si ils ne diffèrent que par l'ordre des symboles formant chaque mot du code (ils possèdent les mêmes caractéristiques, en particulier, la même répartition de poids).

Exercice VI.7 (Codes de Hamming). On appelle *code de Hamming* sur $\mathbb{F}_2$ tout code $\mathscr{C}$ de longueur $n = 2^k - 1$ admettant comme matrice de contrôle une matrice H définie comme suit : les colonnes de H sont tous les vecteurs de $(\mathbb{F}_2)^k - \{0\}$.

1. Montrer que sa dimension est $m = 2^k - 1 - k$ et que sa distance minimale est 3. Comment décoder une erreur ?
2. En reprenant la construction des codes BCH dans le cas où $q = 2$ et $n = 2^k - 1$, montrer que l'on peut définir ainsi un code cyclique qui est un code de Hamming (on considérera le corps cyclotomiques $K = \mathbb{F}_{2^k}$, et on utilisera le fait que si α est une racine $n^{\text{ième}}$ primitive, alors α^i parcourt tout K^*).
3. Prouver que le code ainsi construit est parfait, dans le sens où les boules de rayon 1 (la capacité de correction), dont le centre est un mot du code, forment une partition de $(\mathbb{F}_q)^n$ (ici avec $q = 2$ et $n = 2^k - 1$). Expliquer pourquoi le code défini à l'exemple 3.12 est bien un code de Hamming.
4. Montrer que le code dual de $\mathscr{C}$ est un code simplexe, c'est-à-dire que la distance entre deux mots quelconques du code est constante. Combien vaut cette distance ?
5. Comment généraliser la construction des codes de Hamming à un corps fini quelconque (on pensera à utiliser des vecteurs représentant les droites vectorielles) ? Montrer en particulier que sa dimension est $\frac{q^k-1}{q-1} - k$, sa taille $\frac{q^k-1}{q-1}$, et sa distance minimale 3.

Exercice VI.8 (Polynômes énumérateurs et codes de Hamming). On note H la matrice de taille $k \times 2^k - 1$ ayant pour colonnes toutes les représentations binaires des entiers entre 1 et $2^k - 1 = n$.

1. Expliquer pourquoi le code $\mathscr{C}$, dont H est une matrice de contrôle, est un code de Hamming, comme défini à l'exercice VI.7. Calculer le polynôme énumérateur de poids du code de Hamming de taille 7 décrit à l'exemple 3.12.
2. Quelle est la matrice génératrice du code $\mathscr{C}^\perp$? Montrer que chaque colonne de cette matrice a un poids de Hamming égal à 2^{k-1}.

3. En déduire que le polynôme énumérateur de poids de $\mathscr{C}$ s'écrit
$$W_{\mathscr{C}} = \frac{1}{2^k}\left((X+Y)^n + n(X-Y)^{\frac{n+1}{2}}(X+Y)^{\frac{n-1}{2}}\right).$$
4. Quel est le nombre de mots de poids 1 et 2 ? Ceci est-il en accord avec les résultats de l'exercice VI.7 ? Montrer que le nombre de mots de poids 3 est $\frac{1}{6}n(n-1)$.
5. Montrer que la répartition des poids est symétrique, c'est-à-dire que $A_{n-i} = A_i$.

Exercice VI.9 (Code de Hamming étendu). On note $\mathscr{C}$ le code de Hamming étendu de taille 8. C'est le code obtenu en ajoutant un bit de parité au code de Hamming de taille 7 présenté à l'exemple 3.12. Ceci signifie que tous les vecteurs $x \in \mathscr{C}$ vérifient, modulo 2, $\sum_{k=0}^{7} x_i = 0$. Quels sont les paramètres de ce code ? Dresser la liste des mots qui le composent. Calculer son polynôme énumérateur, et montrer que ce code est auto-dual.

Exercice VI.10 (Code de répétition). On note $\mathscr{C}$ le code de répétition pure. Il consiste à remplacer un élément $x \in \mathbb{F}_q$ par le vecteur de $(\mathbb{F}_q)^n$ dont les entrées sont x.

1. Quels sont ses paramètres (dimension, distance minimale) ? Quel est son polynôme énumérateur ?
2. Identifier le code dual $\mathscr{C}^\perp$. Quel est son polynôme énumérateur ? Dans quel(s) cas ce code est-il auto-dual, c'est-à-dire $\mathscr{C} = \mathscr{C}^\perp$?

Exercice VI.11 (Codes de Hadamard). Les matrices de Hadamard sont définies à l'exercice II.6. Soit H_n une telle matrice, de taille $n \times n$. On la suppose normalisée, c'est-à-dire que les entrées de la première ligne et de la première colonne sont égales à 1. On définit $\widetilde{H_n}$ la matrice obtenue à partir de H_n en remplaçant les 1 par des 0 et les -1 par des 1. On définit alors deux codes :

– Le code $\mathscr{A}_n$, dont les mots sont les lignes de $\widetilde{H_n}$ auxquelles on a enlevé la première colonne.
– Le code $\mathscr{B}_n$, qui est constitué de l'union des mots de $\mathscr{A}_n$ et de leurs compléments (on met à 1 les entrées égales à 0, et vice et versa).

1. Ces codes sont-ils linéaires (on pourra distinguer selon la construction de H_n) ?
2. Montrer que deux lignes distinctes de $\widetilde{H_n}$ ont $\frac{n}{2}$ entrées communes et $\frac{n}{2}$ entrées différentes. Quels sont les paramètres de ces deux codes (taille, nombre d'éléments, et distance minimale) ? En déduire que $\mathscr{A}_n$ est un code simplexe, c'est-à-dire que la distance entre deux mots quelconques du code est constante.

Voici par exemple les mots de deux codes $\mathscr{A}_8$ générés par la méthode des résidus quadratiques (dite de Paley) et des matrices de Walsh (chaque ligne représente un mot) :

Résidus quadratiques :	Matrice de Walsh :
(0000000)	(0000000)
(1001011)	(1010101)
(1100101)	(0110011)
(1110010)	(1100110)
(0111001)	(0001111)
(1011100)	(1011010)
(0101110)	(0111100)
(0010111)	(1101001)

(5.2)

Exercice VI.12 (Codes MDS). On considère un code de taille n et de dimension m sur $\mathbb{F}_2$. On note d sa distance minimale, et on rappelle que le code est dit MDS s'il y a égalité dans la borne de Singleton (3.3), c'est-à-dire $d = n+1-m$. On note A_i le nombre de mots de poids i dans $\mathscr{C}$, et A'_i le nombre de mots de poids i dans $\mathscr{C}^\perp$.

1. On définit les polynômes de *Krawtchouk* P_k, par
$$\forall k = 0,\dots,n, \quad P_k(x) \overset{\text{déf.}}{=} \sum_{j=0}^{k} (-1)^j C_x^j C_{n-x}^{k-j}, \tag{5.3}$$
où le coefficient binomial C_x^j, pour $j \in \mathbb{N}$, est défini par
$$C_x^j \overset{\text{déf.}}{=} \frac{x(x-1)\cdots(x-j+1)}{j!}.$$
Montrer que l'on a
$$A'_k = \frac{1}{|\mathscr{C}|} \sum_{i=0}^{n} A_i P_k(i).$$
2. Montrer que l'on a les égalités suivantes :
$$\forall k = 0,\dots,n, \quad \sum_{i=0}^{n-k} C_{n-i}^k A_i = 2^{m-k} \sum_{i=0}^{k} C_{n-i}^{n-k} A'_i.$$
On pourra penser à dériver formellement le polynôme $P(1,Y)$ par rapport à Y.
3. On suppose maintenant que le code $\mathscr{C}$ est MDS. Expliquer pourquoi on a $A_i = 0$ pour $1 \leqslant i \leqslant n-m$, ainsi que $A'_i = 0$ pour $1 \leqslant i \leqslant m$. En déduire que l'on a
$$\forall k = 0,\dots,m-1, \quad \sum_{i=n-m+1}^{n-k} C_{n-i}^k A_i = C_n^k (2^{m-k} - 1).$$
4. Expliquer pourquoi les identités précédentes déterminent de façon unique la répartition des poids de $\mathscr{C}$. Par exemple, donner le nombre de mots de poids non nul minimal d'un code MDS.

Exercice VI.13 (Borne de la programmation linéaire). On note $R(n,d)$ le cardinal maximal d'un code de taille n et de distance minimale d sur $\mathbb{F}_2$. Cette quantité est extrêmement difficile à estimer en général, et nous allons voir qu'en utilisant les résultats de MacWilliams, on peut donner un majorant, appelé borne de la programmation linéaire (car cette quantité apparaît comme solution d'un problème d'optimisation d'une forme linéaire sous contraintes linéaires).

1. Soit $\mathscr{C}$ un code binaire de taille n. On note P_k le $k^{\text{ième}}$ polynôme de Krawtchouk, qui est défini à l'équation (5.3). On note B_i la répartition de $\mathscr{C}$, montrer que l'on a
$$\forall k \in \{0,\dots,n\}, \quad \sum_{i=0}^{n} B_i P_k(i) \geqslant 0.$$
2. En déduire que l'on a
$$R(n,d) \leqslant \max\left\{ \sum_{i=0}^{n} \widetilde{B}_i \;\middle\backslash\; (\widetilde{B_0},\dots,\widetilde{B_n}) \in E_n^d \right\}.$$
L'ensemble E_n est défini de la manière suivante :
$$E_n^d \overset{\text{déf.}}{=} \left\{ (1,0,\dots,0,x_{d+1},\dots,x_n) \in \mathbb{R}_+^{n+1} \;\middle\backslash\; \forall k = 0,\dots,n, \sum_{i=0}^{n} x_i P_k(i) \geqslant 0 \right\}.$$

Chapitre VII
Représentations linéaires des groupes finis

> Quand vous vous habillez, l'ordre dans lequel vous effectuez les phases successives de l'opération n'est pas sans importance : vous commencez par votre chemise et finissez par votre manteau. Et pour vous deshabiller, vous suivez l'ordre inverse : vous ôtez d'abord le manteau, la chemise en dernier lieu.
>
> H. WEYL [77] (1952)

Ce chapitre traite de la théorie des représentations, qui permet d'étendre la notion de caractère et de transformée de Fourier aux groupes non commutatifs. Cette théorie parvient à faire la liaison entre plusieurs domaines des mathématiques et à utiliser des outils spécifiques à une discipline pour résoudre des problèmes formulés dans le langage d'une autre :

- algèbre générale : le problème initial est celui de l'étude d'un groupe abstrait.
- algèbre linéaire : le but est de réaliser « géométriquement » notre groupe comme un groupe de transformations linéaires. L'utilisation des outils matriciels va permettre d'effectuer des calculs sur notre groupe, et d'obtenir des informations précises (résolubilité, simplicité, classes de conjugaison, etc.).
- géométrie : l'étude abstraite d'une géométrie revient à l'étude des invariants pour une action de groupe donnée. La plupart des actions considérées sont linéaires et la théorie des représentations rentre alors naturellement en jeu.

La notion de représentation linéaire, bien qu'assez complexe au premier abord, est en fait au centre de nombreux problèmes au programme de l'agrégation : actions de groupes, matrices équivalentes, groupes finis, espaces hermitiens (l'espace $\mathbb{C}[G]$ est naturellement muni d'une telle structure), dimension des espaces vectoriels, dénombrement, groupes de permutations, sous-espaces stables, dualité, sous-groupes distingués (étude de la simplicité).

En ce qui concerne la théorie des représentations, la référence principale en français est le livre de J.P.SERRE [64]. La démonstration de l'orthogonalité des caractères est assez calculatoire, et ne sera abordée qu'au chapitre suivant. On pourra regarder le livre de référence en langue anglaise de FULTON et HARRIS [35] pour retrouver celle qui est faite ici. En complément, [36] explicite de façon complète la théorie de Fourier sur $\mathbb{C}[G]$, [1] donne bon nombre de tables de caractères des groupes classiques, tout comme [19]. L'histoire de la théorie des représentations des groupes finis est expliquée dans les deux articles de LAM [41] et [42].

1 Premières définitions

Cette première partie est consacrée à la définition même de la notion de représentation, mais aussi (et surtout) à la mise en place de la problématique déjà énoncée : la recherche, à isomorphisme près, des représentations irréductibles. Nous donnons donc la définition de la notion d'isomorphisme de représentations, ainsi que les détails des notions connexes à celle d'irréductibilité. Pour mettre en relief ces définitions, de nombreux exemples sont exposés, qu'ils soient fondamentaux (dans le sens où ils mènent à des constructions utilisées par la suite) ou seulement instructifs (permettant de faire des calculs « à la main » sans utiliser les outils développés par la suite).

1.1 Représentations linéaires

Définition 1.1 (Représentation linéaire). Soit V un K-espace vectoriel de dimension finie n. Une *représentation linéaire* d'un groupe G dans V est la donnée d'un morphisme $\rho : G \to GL(V)$. Ceci correspond à la donnée d'une *action linéaire* du groupe G sur V, en notant $\forall (g,v) \in G \times V$, $g.v = \rho(g)(v)$. On dit aussi que V est un G-module (cette terminologie sera expliquée par la suite).

Définition 1.2 (Représentation fidèle). Une représentation ρ sur un espace vectoriel V est dite *fidèle* si ρ est injective. On dit aussi que G agit fidèlement sur V.

Exemple 1.3. Les exemples qu'il faut avoir en tête sont de nature géométrique : une action de groupe fidèle permet de réaliser un groupe abstrait comme un groupe de transformations (le plus souvent unitaires ou orthogonales, cf. la proposition 1.27) d'un espace vectoriel. Par exemple, on peut voir le groupe symétrique $\mathfrak{S}_4$ comme l'ensemble des isométries qui conservent un cube. Cette identification établit une représentation du groupe abstrait $\mathfrak{S}_4$ sur l'espace vectoriel $\mathbb{R}^3$ comme un groupe de transformations orthogonales.

Nous avons déjà vu au paragraphe 4.2, chap. I, la définition ainsi que les principales propriétés de l'algèbre d'un groupe abélien (sur le corps $\mathbb{C}$ des complexes). Ces définitions s'étendent sans difficulté à un groupe non commutatif et à un corps K quelconque. Nous allons voir que cette algèbre permet de définir la notion de représentation d'une autre manière, en utilisant le langage des modules.

Définition 1.4 (Algèbre d'un groupe). On suppose donné un espace vectoriel V de dimension $|G|$ sur un corps K dont une base est indexée par G, c'est-à-dire une base de la forme $\{e_g\}_{g \in G}$. On peut alors définir une structure de K-algèbre sur V par l'égalité $e_g * e_h = e_{gh}$ qui s'étend par bilinéarité à tout l'espace.On note $K[G]$ cette algèbre, et on identifie souvent $g \in G$ et $e_g \in K[G]$ de sorte que l'on parle de *l'algèbre du groupe G*, et que G s'injecte canoniquement dans $K[G]$ par $g \mapsto e_g$.

L'utilité principale de l'algèbre $K[G]$ qui englobe notre groupe G est qu'elle vérifie une *propriété universelle*, dans le sens où toute représentation sur G s'étend de manière unique en une représentation sur $K[G]$. Définissons d'abord la notion de représentation d'algèbre.

Définition 1.5 (Représentation d'algèbre). Une représentation d'une K-algèbre associative A est la donnée d'un espace vectoriel V sur K de dimension finie et d'un morphisme de K-algèbre $\rho : A \to \mathrm{End}(V)$.

On voit facilement qu'une représentation d'un groupe $\rho : G \to GL(V)$ s'étend par linéarité de façon unique en une représentation d'algèbre $\widetilde{\rho} : K[G] \to \mathrm{End}(V)$. Donc les représentations de $K[G]$ correspondent exactement aux représentations de G. En fait, tout énoncé

concernant des représentations de G a un équivalent en terme d'algèbre du groupe G. Il s'agit simplement d'un choix de langage, chacun ayant des préférences pour telle ou telle formulation.

***Remarque* 1.6. (Représentations et K[G]-modules).** La définition d'une représentation ρ de l'algèbre $K[G]$ correspond exactement à la définition d'un $K[G]$-module à gauche. La multiplication externe d'un vecteur $v \in V$ par un « scalaire » $\lambda \in K[G]$ est alors donnée par $\lambda \cdot v \overset{\text{déf.}}{=} \rho(\lambda)(v)$. Réciproquement, la donnée d'une telle structure définit sans ambiguïté une représentation d'algèbre. Ainsi, les notions de représentation de groupe, de représentation d'algèbre, et de $K[G]$-module sont totalement équivalentes. Dans la suite, on rencontrera la notion de G-morphisme, qui correspondra aux morphismes pour la structure de G-module.

Comme nous l'avons déjà remarqué au paragraphe 4.2, chap. I, dans le cas d'un groupe fini, l'algèbre d'un groupe s'identifie de façon naturelle (et canonique) à l'espace des fonctions de G dans le corps K considéré, muni d'un produit appelé produit de convolution. Rappelons ces constructions dans le cadre d'un groupe fini G quelconque et d'un corps K quelconque.

***Remarque* 1.7. (Fonctions sur G et algèbre du groupe).** Si on note, pour $g \in G$, δ_g la fonction qui vaut 1 en g et 0 ailleurs, alors on peut décomposer une fonction $f : G \to K$ dans la base $\{\delta_g\}_{g\in G}$:

$$f = \sum_{g\in G} f(g)\delta_g. \tag{1.1}$$

Ceci permet d'identifier f avec l'élément

$$\sum_{g\in G} f(g)e_g \quad \in K[G].$$

On identifiera donc $K[G]$ avec l'espace des fonctions de G dans K, dont une base est donnée par $\{\delta_g\}_{g\in G}$.

Rappelons la formule qui donne la multiplication sur l'algèbre $\mathbb{C}[G]$, et que l'on nomme, en utilisant le vocabulaire de l'analyse fonctionnelle, *produit de convolution*.

Définition 1.8 (Produit de convolution). La multiplication sur $K[G]$ est définie par extension de la multiplication dans G. En quelque sorte, on sait comment multiplier entre eux les éléments de la base $\{\delta_g\}_{g\in G}$ (en se rappelant qu'un élément $\delta_g \in K[G]$ s'identifie à $g \in G$), et par bilinéarité de la multiplication, on peut ainsi calculer le produit de deux éléments en les décomposant comme en (1.1). Ainsi, pour $(f, g) \in K[G]^2$ on obtient

$$\forall s \in G, \quad (f * g)(s) \overset{\text{déf.}}{=} \sum_{hk=s} f(h)g(k) = \sum_{h\in G} f(h)g(h^{-1}s).$$

Cette multiplication est nommée produit de convolution. L'élément neutre pour cette opération est δ_1 (qui s'identifie à l'élément neutre 1 de G), et il ne faut donc pas confondre $*$ avec la multiplication composante par composante des fonctions de G dans K (notée habituellement $\cdot$), pour laquelle l'élément neutre est la fonction constante 1.

1.2 Exemples fondamentaux

La représentation régulière

Cette représentation est la plus élémentaire, et aussi la plus importante. C'est en décomposant la représentation régulière en sous-représentations que l'on obtiendra bon nombre d'informations sur le groupe G, comme on le verra au paragraphe 4.2.

Définition 1.9. La *représentation régulière à gauche* est la représentation de G sur l'espace vectoriel $K[G]$ définie par le morphisme

$$\forall s \in G, \quad \rho_r(s) : \begin{cases} K[G] & \longrightarrow & K[G] \\ f & \longmapsto & \delta_s * f \end{cases}.$$

***Remarque* 1.10.** On peut, comme cela est expliqué à la définition 1.5 prolonger la représentation régulière en une représentation d'algèbre définie sur $K[G]$ tout entière. On voit que l'on obtient tout simplement la structure d'algèbre de $K[G]$ (pour le produit de convolution). Ceci signifie que le $K[G]$-module donné par $K[G]$ lui même correspond à la représentation régulière.

Proposition 1.11. *La représentation régulière est fidèle.*

Démonstration. Si $\rho(g) = \mathrm{Id}$, alors $\rho(g)(\delta_e) = \delta_e$, c'est-à-dire $\delta_g * \delta_e = \delta_g = \delta_e$, donc $g = e$. □

La représentation somme

L'opération la plus simple que l'on puisse faire entre deux représentations est la somme directe, qui est définie de la manière suivante.

Définition 1.12 (Représentation somme). Pour deux représentations ρ_V et ρ_W respectivement sur V et W, on définit une représentation $\rho_{V\oplus W}$ sur $V \oplus W$ par

$$\forall g \in G, \forall (v,w) \in V \times W, \quad \rho_{V\oplus W}(g)((v,w)) \overset{\text{déf.}}{=} \rho_V(g)(v) + \rho_W(g)(w).$$

La notion de somme est à rapprocher de la notion de décomposabilité qui est abordée au paragraphe 1.3. La question est de savoir à quelle condition une représentation peut s'écrire comme la somme de deux autres. Nous verrons (au théorème 1.29) que ceci est simplement lié au fait que la représentation admet ou non des sous-représentations.

La représentation des morphismes

La représentation produit (au sens du *produit tensoriel* d'espaces vectoriels) ne sera pas abordée. Cependant, on peut utiliser, à la place, la notion de représentation sur l'espace vectoriel des morphismes.

Définition 1.13 (Représentation des morphismes). Pour deux représentations ρ_V et ρ_W respectivement sur V et W, on définit une représentation $\rho_{\mathscr{L}(V,W)}$ sur $\mathscr{L}(V,W)$ (espace des applications linéaires de V dans W) par

$$\forall g \in G, \forall f \in \mathscr{L}(V,W), \quad \rho_{\mathscr{L}(V,W)}(g)(f) \overset{\text{déf.}}{=} \rho_W(g) \circ f \circ \rho_V(g^{-1}).$$

Proposition 1.14. *On définit bien ainsi une représentation.*

Démonstration. On note $\rho \overset{\text{déf.}}{=} \rho_{\mathscr{L}(V,W)}$. Tout d'abord, constatons que $\rho(g)$ est bien une application linéaire. La proposition résulte simplement du calcul, pour $f \in \mathscr{L}(V,W)$ et pour $\forall (g,h) \in G^2$:

$$\begin{aligned}\rho(gh)(f) &= \rho_W(gh) \circ f \circ \rho_V((gh)^{-1}) \\ &= \rho_W(g) \left\{\rho_W(h) \circ f \circ \rho_V(h^{-1})\right\} \rho_V(g^{-1}) \\ &= \rho(g) \left\{\rho(h)(f)\right\}.\end{aligned}$$

□

La représentation duale

Dans le même ordre d'idées que pour la représentation des morphismes, on peut définir une représentation duale.

Définition 1.15 (Représentation duale). Pour une représentation ρ sur V, on définit une représentation ρ^* sur V^* le dual de V par

$$\forall g \in G, \quad \rho^*(g) \overset{\text{déf.}}{=} \rho(g^{-1})^t,$$

où l'on a noté $\varphi^{\mathrm{T}} \in \mathscr{L}(V^*)$ l'opérateur transposé de $\varphi \in \mathscr{L}(V)$.

***Remarque* 1.16.** On peut voir que cette définition correspond à la représentation des morphismes $\rho_{\mathscr{L}(V,K)}$ (où K est considéré comme un espace de dimension 1), puisque $V^* = \mathscr{L}(V,K)$, avec la représentation triviale $\rho_K(g) = \mathrm{Id}$.

Définition 1.17 (Crochet de la dualité). On note, pour $f \in V^*$ et pour $x \in V$,

$$\langle x, f \rangle_{(E,E^*)} \overset{\text{déf.}}{=} f(x).$$

On nomme cette application le *crochet de la dualité*, qui est une forme bilinéaire sur $E \times E^*$.

On montre facilement que la représentation duale que nous venons de construire a un comportement intéressant vis-a-vis du crochet de la dualité.

Proposition 1.18. *La représentation duale sur V^* conserve le crochet de la dualité, c'est-à-dire :*

$$\forall g \in G, \forall (f,x) \in E^* \times E, \quad \langle x, \rho_{V^*}(g)(f) \rangle_{(E,E^*)} = \left\langle \rho_V(g^{-1})(x), f \right\rangle_{(E,E^*)}.$$

Une action sur les polynômes

Définition 1.19. Soit G un sous-groupe fini de $GL_n(K)$. Si on note, pour $A \in G$, A^{-1} sous la forme $(a_{i,j})_{1 \leqslant i,j \leqslant n}$, on définit une action linéaire de G sur $K[X_1, \ldots, X_n]$ en définissant $\rho(A)(P)$ le polynôme obtenu par la substitution de X_i par $\sum_{j=1}^n a_{j,i} X_j$. On note symboliquement $\rho(A)(P)(X) = P(A^{-1} \cdot X)$.

Si on considère cette action sur $K[X_1,\dots,X_n]$ tout entier, on obtient une représentation de dimension infinie. Cependant, il est facile de voir que cette action respecte le degré des polynômes. On peut donc restreindre cette action au sous-espace $K_s[X_1,\dots,X_n]$ constitué des polynômes de degré inférieur ou égal à s. C'est un espace de dimension finie, et ceci donne naissance à une représentation linéaire de dimension finie. Mais on peut aussi considérer l'espace $K_s^0[X_1,\dots,X_n]$ des polynômes homogènes de degré s (en incluant le polynôme nul), ce qui fournit une deuxième famille de représentations de dimension finie. C'est ce point de vue qui sera adopté pour prouver le *théorème de Molien*, à l'exercice VII.6.

Enfin, notons que la théorie des polynômes invariants sous cette action est très importante. L'exercice VII.5 propose de démontrer un résultat fondamental de cette théorie. Dans le cadre de l'étude des codes correcteurs, ce sont ces outils de théorie des représentations qui permettent de classifier les codes *auto-duaux*. On trouvera une instance de cette approche dans l'exercice VIII.9.

Représentation de degré 1

On se place dans le cas où $K = \mathbb{C}$. Une représentation de degré 1 est simplement un morphisme de G dans le groupe multiplicatif $\mathbb{C}^*$ (on identifie $GL_1(\mathbb{C})$ et $\mathbb{C}^*$). C'est donc un caractère de G comme défini au chapitre I. On retrouve ainsi la théorie classique de dualité sur un groupe fini. Nous savons déjà que si G est abélien, les caractères forment une base de l'espace $\mathbb{C}[G]$ des fonctions de G dans $\mathbb{C}$. Dans la suite, nous étendrons la notion de caractère, et nous verrons que cette notion a bien les propriétés espérées.

- Dans le cas où G est commutatif, cette nouvelle notion n'apporte rien de nouveau : on retrouve uniquement les caractères déjà définis. Intuitivement, on sait que la dimension 1 suffit pour étudier les groupes commutatifs.
- Dans le cas non commutatif, l'ajout de « nouveaux » caractères permet de développer une théorie de Fourier généralisant la théorie déjà développée dans le cas commutatif.

Dans un premier temps, contentons nous de la remarque suivante :

***Remarque* 1.20.** Si ρ est une représentation de G sur un espace vectoriel V, alors l'application qui à $s \in G$ associe $\det(\rho(s))$ est une représentation de degré 1 sur $\mathbb{C}$ (c'est-à-dire un caractère au sens où nous l'avons déjà défini).

1.3 Représentations irréductibles

La notion de représentation irréductible est très intuitive. Comme dans toute construction, on cherche les « briques de bases », celles avec lesquelles nous allons pouvoir reconstruire tout l'édifice (ici toutes les représentations). Notre outil est, comme nous l'avons défini au paragraphe 1.2, la somme de représentations. La définition intuitive d'une « brique de base » est qu'elle doit être minimale (au sens de l'inclusion des sous-espaces non nuls). Cette définition est-elle bien compatible avec la construction par somme de nouvelles représentations ? C'est ce que précisera le théorème 1.29, dans le cas d'un corps algébriquement clos. En effet, nous allons petit à petit quitter la généralité des constructions du paragraphe précédent, pour nous restreindre au cas du corps $K = \mathbb{C}$, pour lequel

il y a déjà beaucoup à faire. Cependant, nous essaierons, dans la mesure du possible, de mentionner les résultats qui restent valables sous des hypothèses plus faibles.

Définition 1.21 (Représentations isomorphes). Deux représentations ρ et ρ' d'un même groupe G respectivement sur deux K-espaces vectoriels V et V' sont dites *isomorphes* s'il existe un isomorphisme d'espaces vectoriels $\tau : V \to V'$ tel que pour tout $s \in G$, $\tau \circ \rho(s) = \rho'(s) \circ \tau$, ce qui permet d'identifier les deux représentations.

La notion d'isomorphisme de représentations définit une relation d'équivalence sur les représentations d'un groupe G donné. Dans la suite, nous allons nous intéresser aux classes d'équivalence pour cette relation. Nous allons maintenant donner les définitions qui permettent d'expliciter les notions de « briques de bases ».

Définition 1.22 (Sous-représentations). Si une représentation ρ de G sur V admet un sous-espace vectoriel $W \subset V$ stable par tous les $\rho(s) \in GL(V)$, elle induit une représentation ρ_W sur W appelée *sous-représentation.*

***Remarque* 1.23.** En utilisant le langage des $K[G]$-modules, on voit qu'une sous-représentation n'est rien d'autre qu'un sous $K[G]$-module, et qu'un isomorphisme de représentations est un isomorphisme de $K[G]$-modules.

Définition 1.24 (Représentations irréductibles). Une représentation sur un espace V est dite *irréductible* si elle admet exactement deux sous-représentations : $\{0\}$ et V tout entier.

Définition 1.25 (Représentation indécomposable). Une représentation sur un espace V est dite *indécomposable* si à chaque fois que l'on a un isomorphisme de représentations $V \simeq W_1 \oplus W_2$, alors $W_1 = \{0\}$ ou $W_2 = \{0\}$.

***Remarque* 1.26. (Irréductibilité et indécomposabilité).** Il est évident qu'une représentation irréductible est en particulier indécomposable, puisqu'une décomposition de V sous la forme $V \simeq W_1 \oplus W_2$ non triviale donne naissance à deux sous-représentations. Nous allons maintenant nous intéresser à la question réciproque. Pour résoudre ce problème, il faut savoir si, étant donnée une sous-représentation non triviale W_1 de V, on peut trouver une autre sous-représentation W_2 telle que $V \simeq W_1 \oplus W_2$. Ceci signifie exactement trouver un supplémentaire de W_1 stable sous l'action de G. L'exercice VII.1 montre qu'en général cet aspect réciproque n'a aucune raison d'être vrai. Cependant, sous certaines hypothèses restrictives sur le corps de base, nous allons voir que l'on peut démontrer l'équivalence entre irréductibilité et indécomposabilité.

Important : À partir de maintenant, sauf mention explicite du contraire, nous travaillerons dans le corps des complexes $K = \mathbb{C}$.

Proposition 1.27 (Représentation unitaire). *Soit ρ une représentation d'un groupe fini G sur un espace V. Alors ρ laisse invariant le produit hermitien suivant :*

$$\langle x, y \rangle_G \overset{\text{déf.}}{=} \sum_{s \in G} \langle \rho(s)(x), \rho(s)(y) \rangle,$$

où l'on a noté $\langle \cdot, \cdot \rangle$ un produit hermitien quelconque sur V.

Démonstration. Le fait que $\langle \cdot, \cdot \rangle_G$ est invariant par G est trivial :

$$\langle \rho(g)(x), \rho(g)(y) \rangle_G \overset{\text{déf.}}{=} \sum_{s \in G} \langle \rho(sg)(x), \rho(sg)(y) \rangle = \langle x, y \rangle_G.$$

Le seul point important est que $\langle \cdot, \cdot \rangle_G$ est bien un produit hermitien, comme somme de produits hermitiens. Ceci est valide car on travaille sur le corps $\mathbb{C}$ des complexes. □

***Remarque* 1.28. (Représentation unitaire).** Ce résultat est équivalent au fait que les matrices M_s des $\rho(s)$ sont unitaires dans une base orthonormée pour $\langle \cdot, \cdot \rangle_G$, c'est-à-dire vérifient $M_s M_s^* = \mathrm{Id}$, où M_s^* est la matrice adjointe. On dit que M_s est une *représentation matricielle unitaire*.

Théorème 1.29 (Irréductibilité et indécomposabilité). *Une représentation ρ sur V est irréductible si et seulement si elle est indécomposable.*

Démonstration. Soit W_1 une sous-représentation non triviale de V. Comme nous l'avons déjà fait remarquer, pour démontrer l'équivalence, il suffit de trouver un supplémentaire de W_1 stable par G. On peut alors considérer le produit hermitien invariant $\langle \cdot, \cdot \rangle_G$, et choisir W_2 l'orthogonal de W_1. Par conservation du produit scalaire, l'image par G d'un vecteur orthogonal à W_1 est encore orthogonal à W_1 : W_2 est bien stable sous l'action de G. □

***Remarque* 1.30.** La démonstration qui précède utilise l'existence d'un produit hermitien stable. Elle n'est donc pas valable sur un corps autre que $\mathbb{C}$. On peut cependant proposer une autre démarche, qui permet de démontrer le théorème 1.29 dans le cas d'un corps K tel que sa caractéristique ne divise pas $|G|$. Voici donc une deuxième démonstration :

Démonstration. Il existe une autre façon de construire un supplémentaire stable de W_1. En effet, considérons un supplémentaire W_2 quelconque, et notons π la projection sur W_1 associée à la décomposition $V = W_1 \oplus W_2$. On peut alors définir un endomorphisme π_0 de la manière suivante :

$$\pi_0 \overset{\text{déf.}}{=} \frac{1}{|G|} \sum_{g \in G} \rho(g) \circ \pi \circ \rho(g^{-1}) \; ;$$

ceci est valide car $\mathrm{Car}(K)$ ne divise pas $|G|$. On peut alors constater que π_0 est un projecteur d'image W_1, et même mieux, que son noyau $W_1 \overset{\text{déf.}}{=} \ker(\pi_0)$ est stable par l'action de G (on le vérifie à la main aisément). En vertu des propriétés des projecteurs, on a $V = W_1 \oplus W_2$. Cette construction, qui peut paraître un peu magique, est en fait très naturelle, et deviendra claire une fois définies les notions de G-morphisme (définition 2.3), et surtout d'*opérateur de Reynolds* R_G (définition 2.10), puisque l'on a en fait construit $\pi_0 = R_G(\pi)$. □

***Remarque* 1.31.** Le théorème 1.29 signifie que si ρ est réductible, les matrices des $\rho(s)$ s'écrivent comme une diagonale de deux blocs dans une base bien choisie, ce qui correspond bien à la notion de réductibilité (et à la représentation somme). Voici maintenant le résultat qui nous assure que les « briques de base » que sont les représentations irréductibles permettent de reconstruire toutes les représentations.

Proposition 1.32. *Toute représentation peut s'écrire comme somme de représentations irréductibles.*

Démonstration. On raisonne par récurrence sur la dimension de V l'espace vectoriel de notre représentation. Un espace de dimension 1 est irréductible. Si V est un espace de dimension plus grande que 1 irréductible, et la démonstration est terminée. Sinon, V admet un sous-espace stable non trivial W, et avec le corollaire 1.29, on peut trouver W_0 stable

tel qu'on ait $V = W \oplus W_0$. En appliquant l'hypothèse de récurrence à W et W_0, on prouve ce qui était demandé. □

***Remarque* 1.33.** Cette écriture n'est pas unique, mais nous allons voir qu'elle est unique « à isomorphisme près », au sens que si l'on a deux décompositions de W sous la forme $W_1 \oplus \cdots \oplus W_r$ et $W'_1 \oplus \cdots \oplus W'_{r'}$, alors $r = r'$, et quitte à réordonner les indices, il existe des isomorphismes $W_i \simeq W'_i$.

1.4 Le groupe symétrique

Avant d'attaquer le vif du sujet, à savoir l'introduction d'outils utiles pour étudier les représentations, intéressons nous à un groupe de première importance, le groupe symétrique $\mathfrak{S}_n$. Nous allons essayer de dégager « à la main » les principales caractéristiques de ses représentations, à défaut de pouvoir en donner une description exhaustive.

Définition 1.34 (Représentation par permutation). Soit $\mathfrak{S}_n$ le groupe des permutations d'un ensemble à n éléments, identifié à l'ensemble $\{1,\ldots,n\}$. Soit V un espace vectoriel de dimension n, dont on choisit une base $\mathscr{B} = \{e_1,\ldots,e_n\}$. $\mathfrak{S}_n$ agit sur V par permutation des éléments de la base $\mathscr{B}$, ce qui permet de définir un morphisme $\rho_p : \mathfrak{S}_n \to GL(V)$. On notera donc, pour $\sigma \in \mathfrak{S}_n$, $\rho_p(\sigma)$ l'endomorphisme correspondant, c'est-à-dire tel que $\rho_p(e_i) = e_{\sigma(i)}$. On note M_σ la matrice de permutation associée, qui est la matrice de $\rho_p(\sigma)$ dans la base $\mathscr{B}$. Cette matrice ne comporte qu'un seul 1 par ligne et par colonne, et des 0 partout ailleurs. De plus, seule $M_{\mathrm{Id}} = \mathrm{Id}$ ne comporte que des 1 sur la diagonale (ce fait sera utilisé pour l'exemple 3.5).

***Remarque* 1.35. (Lien avec la représentation régulière).** Tout groupe G de cardinal n s'injecte dans le groupe $\mathfrak{S}_n$ des permutations de l'ensemble $\{1,\ldots,n\}$. Pour le voir, il suffit de numéroter les éléments de $G = \{g_1,\ldots,g_n\}$ et de considérer, pour $h \in G$, la permutation

$$j(h) : \left\{ \begin{array}{ccc} \{1,\ldots,n\} & \longrightarrow & \{1,\ldots,n\} \\ k & \longmapsto & k' \end{array} \right. \qquad \text{où } k' \text{ est tel que } hg_k = g_{k'}.$$

On a alors le diagramme commutatif suivant :

$$\begin{array}{ccc} G & \xrightarrow{\rho_r} & GL(V) \\ \downarrow{\scriptstyle j} & & \| \\ \mathfrak{S}_n & \xrightarrow{\rho_p} & GL(V) \end{array}$$

Dans le même ordre d'idées, lorsque l'on étudie la représentation ρ_p de $\mathfrak{S}_n$ sur un espace de dimension n (qu'on peut supposer sur un corps K quelconque), on peut s'intéresser à la détermination des classes de similitude des matrices des $\rho_p(\sigma)$, $\sigma \in \mathfrak{S}_n$. Commençons par caractériser les classes de conjugaison (c'est-à-dire de similitude) dans le groupe symétrique $\mathfrak{S}_n$.

Lemme 1.36 (Classes de conjugaison dans $\mathfrak{S}_n$). *Deux éléments de $\mathfrak{S}_n$ sont dans la même classe de conjugaison si et seulement si leur décomposition en cycles disjoints possède le même nombre de cycles d'une longueur donnée.*
En particulier, il y a autant de classes de conjugaison dans $\mathfrak{S}_n$ que de partitions de n du type

$$n = k_1 + k_2 + \cdots + k_p \quad avec \quad k_1 \geqslant k_2 \geqslant \cdots \geqslant k_p > 0.$$

Démonstration. Tout d'abord, notons que deux cycles de même longueur sont conjugués. En effet, si on considère $c \overset{\text{déf.}}{=} (c_1, \ldots, c_k)$ et $c' \overset{\text{déf.}}{=} (c'_1, \ldots, c'_k)$, il suffit d'utiliser une permutation $\sigma : c_i \mapsto c'_i$ pour voir que $c = \sigma^{-1} \circ c' \circ \sigma$. Donc si deux permutations possèdent des décompositions avec des cycles de même longueur, elles sont conjuguées.
Réciproquement, il est évident que la conjugaison d'une permutation conjugue aussi les cycles qui la composent, et donc conserve leurs longueurs. □

Théorème 1.37 (Théorème de Brauer). *On se place sur un corps K de caractéristique 0. Deux matrices M_σ et $M_{\sigma'}$ sont semblables si et seulement si σ et σ' sont conjuguées dans $\mathfrak{S}_n$, c'est-à-dire :*

$$\exists \tau \in \mathfrak{S}_n, \quad \sigma\tau = \tau\sigma'.$$

Démonstration. Cette démonstration m'a été communiquée par DANIEL FERRAND.
Le sens réciproque découle directement de la définition d'une représentation.
En effet, si $\exists \tau \in \mathfrak{S}_n$, $\sigma' = \tau^{-1}\sigma\tau$, alors $M_{\sigma'} = M_{\tau^{-1}\sigma\tau} = M_\tau^{-1} M_\sigma M_\tau$.
Pour le sens direct, on note $c_k(\sigma)$ le nombre de cycles d'ordre k dans la décomposition de σ en cycles disjoints. En utilisant le lemme 1.36, il suffit de montrer que $\forall k \in \{1, \ldots, n\}$, $c_k(\sigma) = c_k(\sigma')$. Or tout cycle α d'ordre k vérifie $\alpha^k = \mathrm{Id}$, donc le polynôme caractéristique de M_α est $X^k - 1$. Comme M_σ et $M_{\sigma'}$ ont le même polynôme caractéristique (car elles sont semblables), on a

$$\prod_{k \geqslant 1} (X^k - 1)^{c_k(\sigma)} = \prod_{k \geqslant 1} (X^k - 1)^{c_k(\sigma')}. \tag{1.2}$$

Pour $m \in \mathbb{N}$, on prend ζ une racine $m^{\text{ième}}$ de l'unité (dans une clôture algébrique $\overline{K}$ de K) d'ordre m (c'est-à-dire primitive). Comme on est en caractéristique 0, $P = X^k - 1$ et $P' = kX^{k-1}$ sont premiers entre eux, ce qui signifie que P est scindé à racines simples dans $\overline{K}$. Donc la multiplicité de ζ dans P est 1 si ζ est racine de P (c'est-à-dire si $m|k$), 0 sinon. En égalant les multiplicités dans l'égalité (1.2), on obtient

$$\sum_{m|k} c_k(\sigma) = \sum_{m|k} c_k(\sigma').$$

Supposons maintenant qu'il existe m tel que $c_m(\sigma) \neq c_m(\sigma')$. On choisit m le plus grand possible, notons-le m_0 (ceci est possible car $k \mapsto c_k(\sigma)$ est une fonction à support fini). On a alors

$$0 = \sum_{m_0|k} c_k(\sigma) - \sum_{m_0|k} c_k(\sigma') = c_{m_0}(\sigma) - c_{m_0}(\sigma'),$$

ce qui est une contradiction, car $c_{m_0}(\sigma) \neq c_{m_0}(\sigma')$. □

Après avoir répondu à ces questions sur les classes de similitudes liées à la représentation par permutation, se pose le problème de la détermination des représentations de $\mathfrak{S}_n$. Comme il est vain de vouloir toutes les déterminer (il est facile d'en créer des nouvelles par sommes directes), le vrai problème est en fait la détermination des représentations *irréductibles* de $\mathfrak{S}_n$. Cette question est difficile, et dans un premier temps, nous allons nous contenter de donner des représentations « évidentes ». L'exemple du groupe $\mathfrak{S}_4$ sera traité complètement au paragraphe 1.4, chap. VIII. Il existe cependant des descriptions précises des représentations irréductibles de $\mathfrak{S}_n$, qui sont fondées sur l'action de ce groupe sur les *tableaux de Young*. Une description complète se trouve dans le livre de FULTON et HARRIS [35].

Tout d'abord, est-ce que la représentation de permutation, pour $n \geqslant 2$, est irréductible ? On voit aisément que non, en regardant le sous-espace

$$H_0 \overset{\text{déf.}}{=} \{\lambda(1,\dots,1) \setminus \lambda \in \mathbb{C}\} = \text{Vect}((1,\dots,1)). \tag{1.3}$$

On voit facilement que ce sous-espace est stable par toute permutation des coordonnées, et qu'il admet un supplémentaire également stable par G :

$$H_1 \overset{\text{déf.}}{=} \{(x_1,\dots,x_n) \in V \setminus x_1 + \cdots + x_n = 0\}. \tag{1.4}$$

La représentation par permutation ρ_p induit la représentation triviale (c'est-à-dire constante égale à l'identité) sur H_0. La question de l'irréductibilité de la représentation H_1 est abordée à l'exercice VII.7. Nous verrons au paragraphe 1.4, chap. VIII, que dans l'exemple de $\mathfrak{S}_4$, cette représentation est bien irréductible. On nomme *représentation standard* la représentation induite par ρ_p sur H_1. Outre la représentation triviale (qui est bien sûr irréductible), il reste une autre représentation de degré un, donnée par l'équation

$$\forall \sigma \in \mathfrak{S}_n, \quad \rho_\varepsilon(\sigma) = \varepsilon(\sigma),$$

où l'on a noté $\varepsilon(\sigma)$ la signature de la permutation σ (comme c'est une représentation de degré 1, on a noté $\rho_\varepsilon(\sigma)$ comme un scalaire, alors que c'est en réalité une matrice de taille 1). On nomme cette représentation la représentation alternée. De plus, nous avons déjà vu au chapitre I que c'étaient les deux seules représentations de degré 1 de $\mathfrak{S}_n$.

2 Invariance et représentations

Un moyen très simple pour créer des sous-espaces globalement stables sous l'action d'un groupe G est de regarder l'ensemble des vecteurs qui ne sont pas modifiés par G, qui forme bien un sous-espace. Sur ce sous-espace invariant, G induit la représentation triviale. L'intérêt capital de ce sous-espace est qu'on dispose d'une description complète, et d'un moyen très simple d'en générer des éléments. L'idée fondamentale qui se cache derrière la construction faite dans ce paragraphe (et derrière *l'opérateur de Reynolds*, qui est présenté au paragraphe 2.3) est que l'on se trouve sur un groupe fini, et donc que l'on est en mesure de *moyenner* l'action de notre groupe. Ce principe très simple, que nous avons déjà utilisé pour construire des supplémentaires stables, sera d'un usage constant dans la suite de l'exposé, et c'est pour cela qu'il est important de le formaliser.

2.1 Sous-représentation invariante

Définition 2.1 (Sous-représentation invariante). Soit ρ une représentation sur V. On note V^G le sous-espace des vecteurs *invariants*, c'est-à-dire :

$$V^G = \left\{ v \in V \setminus \forall s \in G, \rho(s)(v) \overset{\text{déf.}}{=} s.v = v \right\}.$$

C'est une sous-représentation de V.

Exemple 2.2. On considère l'action du groupe symétrique $\mathfrak{S}_n$ par permutation des coordonnées, comme définie au paragraphe 1.2. Cette action permet de définir une autre action, sur $K[X_1,\dots,X_n]$ cette fois-ci, via la construction effectuée au paragraphe 1.2. La

représentation invariante $K[X_1,\dots,X_n]^{\mathfrak{S}_n}$ est formée des polynômes symétriques, ce qui signifie que $P \in K[X_1,\dots,X_n]^{\mathfrak{S}_n}$ si et seulement si

$$\forall \sigma \in \mathfrak{S}_n, \quad P(X_{\sigma(1)},\dots,X_{\sigma(n)}) = P(X_1,\dots,X_n).$$

Ces polynômes, ainsi que l'action des groupes finis sur les polynômes sont étudiés en détail aux exercices VII.5 et VII.6.

Définition 2.3 (Opérateurs d'entrelacement). Dans le cas de la représentation des morphismes $\rho_{\mathscr{L}(V,W)}$ sur $\mathscr{L}(V,W)$ de deux représentations ρ_V et ρ_W respectivement sur V et W, on note $\mathrm{Hom}_G(V,W)$ l'espace des invariants. On nomme ces éléments des *opérateurs d'entrelacement* ou des *G-morphismes*.

***Remarque* 2.4.** Dire que $f \in \mathscr{L}(V,W)$ est un opérateur d'entrelacement correspond au fait que f vérifie $\forall s \in G$, $f \circ \rho_V(s) = \rho_W(s) \circ f$, c'est-à-dire f fait commuter, pour tout $s \in G$, le diagramme suivant :

$$\begin{array}{ccc} V & \xrightarrow{f} & W \\ \downarrow{\scriptstyle \rho_V(s)} & & \downarrow{\scriptstyle \rho_W(s)} \\ V & \xrightarrow{f} & W \end{array}$$

Si f est bijectif, ceci correspond au fait que f est un isomorphisme de représentations. Dans le cas général, on parle de G-morphisme, ou d'opérateur d'entrelacement. En reprenant le langage des $K[G]$-modules, un opérateur d'entrelacement est simplement un morphisme de $K[G]$-modules.

2.2 Lemme de Schur

Ce lemme, à l'apparence très simple, est en fait la pierre angulaire de la plupart des démonstrations qui seront faites dans la suite de l'exposé.

Lemme 2.5 (Lemme de Schur). *Soient $\rho_V : G \to GL(V)$ et $\rho_W : G \to GL(W)$ deux représentations irréductibles d'un groupe G. Soit $f \in \mathscr{L}(V,W)$ un opérateur d'entrelacement, c'est-à-dire $f \in \mathrm{Hom}_G(V,W)$. Alors*

(i) *si ρ_V et ρ_W ne sont pas isomorphes, $f = 0$.*
(ii) *si $f \neq 0$, alors f est un isomorphisme, les représentations sont isomorphes, et si on suppose $V = W$, $\rho_V = \rho_W$, alors f est une homothétie.*

Démonstration. Si on suppose que $f \neq 0$, alors les hypothèses montrent que $\ker(f)$ est stable par tous les $\rho_V(s)$. En effet,

$$\forall x \in \ker(f), \quad f(\rho_V(s)(x)) = \rho_W(s)(f(x)) = \rho_W(s)(0) = 0,$$

d'où $\rho_V(x) \in \ker(f)$. Donc comme ρ_V est irréductible et $f \neq 0$, $\ker(f) = \{0\}$. De même, on montre que $\mathrm{Im}(f)$ est stable par tous les $\rho_W(s)$, et comme ρ_W est irréductible et $f \neq 0$, $\mathrm{Im}(f) = W$. Au final, f est un isomorphisme et ρ_V et ρ_W sont isomorphes.
Pour montrer (ii), comme on travaille sur des $\mathbb{C}$-espaces vectoriels, f a au moins une valeur propre λ. En posant $f' = f - \lambda \mathrm{Id}$, on voit que $\ker(f') \neq \{0\}$. En appliquant la première partie de la démonstration à f' qui est encore un G-morphisme, on a $f' = 0$. □

***Remarque* 2.6.** Dans le cas où on travaille dans un corps K non nécessairement algébriquement clos, on garde le fait que si $f \neq 0$, f est un isomorphisme. En particulier, si V est irréductible, alors $\mathrm{Hom}_G(V,V) \overset{\text{déf.}}{=} \mathrm{End}_G(V)$ est un corps non nécessairement commutatif.

***Remarque* 2.7.** On peut démontrer le lemme de Schur en employant le langage des $K[G]$-modules. En effet, dire que f est un opérateur d'entrelacement signifie que f est un morphisme de $K[G]$-modules. Or on vérifie que dans ce cas, $\ker(f)$ et $\mathrm{Im}(f)$ sont des sous-$K[G]$-modules respectivement de V et W. L'irréductibilité de ces deux modules permet de conclure de la même manière.

Corollaire 2.8. *On considère toujours deux représentations irréductibles de G sur V et W. On a* $\dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W)) = 1$ *si V et W sont isomorphes, et* $\dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W)) = 0$ *sinon.*

Démonstration. Si $V = W$ ou si les représentations ne sont pas isomorphes, le lemme de Schur nous donne le résultat. Dans le cas où les deux représentations sont isomorphes (mais non définies sur le même espace), il suffit de se fixer g un isomorphisme entre les deux espaces vectoriels. On peut alors considérer $\rho'_W(s) = g^{-1} \circ \rho_W(s) \circ g \in \mathscr{L}(V)$. En appliquant le lemme de Schur à ρ_V et ρ'_W, on voit que tout opérateur d'entrelacement entre ces deux représentations s'écrit $\lambda\,\mathrm{Id}$. Donc tout opérateur d'entrelacement entre V et W s'écrit λg, et on a bien $\dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W)) = 1$. □

***Remarque* 2.9.** **(Cas des groupes commutatifs)**. Une fois définie la notion d'opérateur d'entrelacement, une question naturelle est de savoir si, pour $g \in G$, $\rho(g)$ est un opérateur d'entrelacement. Or $\rho(g) \in \mathrm{Hom}_G(V)$ est équivalent à

$$\forall h \in G, \quad \rho(g)\rho(h) = \rho(h)\rho(g), \quad \text{c'est-à-dire} \quad \rho(ghg^{-1}h^{-1}) = 1.$$

Donc si $g \in Z(G)$, le centre de G, alors $\rho(g) \in G$.
En particulier, si G est commutatif, alors $Z(G) = G$, et donc avec le lemme de Schur 2.5, pour une représentation irréductible ρ de G sur V, comme les $\rho(g)$, pour $g \in G$ sont des opérateurs d'entrelacement, ce sont des multiples de l'identité, ce qui signifie que ρ est une représentation de degré 1. On retombe donc dans la théorie classique de la dualité sur un groupe fini commutatif (ce qui est rassurant), et on peut utiliser pleinement la théorie développée au chapitre précédent. Cette constatation sera redémontrée à l'aide de la théorie des caractères au corollaire 5.15.

2.3 Opérateur de Reynolds

Nous allons maintenant définir l'opérateur qui va nous permettre de moyenner l'action de G sur un espace vectoriel.

Définition 2.10 (Opérateur de Reynolds). Soit ρ une représentation de G sur V. On définit l'opérateur $R_G \in \mathscr{L}(V,V)$ par

$$R_G \overset{\text{déf.}}{=} \frac{1}{|G|} \sum_{s \in G} \rho(s) \qquad \in \mathscr{L}(V,V).$$

On l'appelle *opérateur de Reynolds* associé à ρ.

Théorème 2.11 (Propriétés de l'opérateur de Reynolds). *R_G est un projecteur sur V^G. En particulier :*

(i) $V^G = \mathrm{Im}(R_G) = \ker(R_G - \mathrm{Id})$.
(ii) $\dim_{\mathbb{C}}(V^G) = \mathrm{tr}(R_G)$.

Démonstration.

(i) Soit $y = R_G(x) \in \mathrm{Im}(R_G)$, alors pour $s \in G$, on a

$$\rho(s)(y) = \frac{1}{|G|} \sum_{g \in G} \rho(s)\rho(g)(x) = \frac{1}{|G|} \sum_{g \in G} \rho(sg)(x) = R_G(x) = y,$$

donc $y \in V^G$. La réciproque est évidente :

$$\text{si } x \in V^G, \forall s \in G, \rho(s)(x) = x, \text{ alors } x = R_G(x) \in \mathrm{Im}(R_G).$$

Pour montrer que R_G est un projecteur, il faut montrer que $R_G^2 = R_G$, ce qui est évident car $\mathrm{Im}(R_G)$ est stable par G. Enfin, en vertu de la théorie des projecteurs, $\mathrm{Id} - R_G$ est aussi un projecteur de noyau $\mathrm{Im}(R_G) = V^G$.

(ii) La propriété (i) nous montre que R_G est un projecteur sur V^G, donc en particulier $V = \ker(R_G) \oplus \mathrm{Im}(R_G)$ et en écrivant la matrice de R_G dans une bonne base, on en déduit (ii). □

2.4 Application moyennée

En appliquant le lemme de Schur à la représentation des morphismes, nous allons pouvoir calculer la dimension de l'espace des G-morphismes. Pour clarifier les notations, nous allons introduire la définition suivante.

Définition 2.12 (Application moyennée). Soient ρ_V et ρ_W deux représentations respectivement sur V et W. On note $\rho_{\mathscr{L}(V,W)}$ la représentation des morphismes sur $\mathscr{L}(V,W)$. Pour $f \in \mathscr{L}(V,W)$, on note $\widetilde{f} \overset{\text{déf.}}{=} R_G(f) \in \mathscr{L}(V,W)$, ce qui correspond à l'application moyennée :

$$\widetilde{f} \overset{\text{déf.}}{=} \frac{1}{|G|} \sum_{s \in G} \rho_W(s) \circ f \circ \rho_V(s^{-1}).$$

Proposition 2.13 (Application aux G-morphismes). *On reprend les notations de la définition précédente. On suppose que les représentations sur V et W sont irréductibles. On a*

$$\dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W)) = \mathrm{tr}(R_G) = \begin{cases} 1 & \textit{si les représentations sont isomorphes,} \\ 0 & \textit{sinon.} \end{cases}$$

Démonstration. Nous avons vu au corollaire 2.8 que $\dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W))$ vaut bien le membre de droite de l'égalité cherchée. De plus, le théorème 2.11 (ii), nous dit que l'on a $\mathrm{tr}(R_G) = \dim_{\mathbb{C}}(\mathrm{Hom}_G(V,W))$. □

Au final, pour tout $f \in \mathscr{L}(V,W)$, $\widetilde{f}$ est une application G-invariante pour la représentation linéaire $\rho_{\mathscr{L}(V,W)}$, c'est-à-dire un G-morphisme, $\widetilde{f} \in \mathrm{Hom}_G(V,W)$. On a en quelque sorte un moyen de « fabriquer » des opérateurs d'entrelacement. Nous verrons à l'exercice VII.5 une application importante de cette technique.

3 Caractères

Dans cette partie, nous allons définir et utiliser l'outil principal qui sert à analyser des représentations, mais aussi à trouver des représentations de certains groupes abstraits.

3.1 Définition et premières propriétés

Définition 3.1 (Caractères). Soit ρ une représentation d'un groupe G sur V un $\mathbb{C}$-espace vectoriel de dimension n. On lui associe son *caractère* χ_ρ défini par $\chi_\rho(s) = \mathrm{tr}(\rho(s))$ où tr désigne la trace. C'est une fonction de G dans $\mathbb{C}$, c'est-à-dire $\chi_\rho \in \mathbb{C}[G]$.

***Remarque* 3.2.** Il faut faire attention au fait que les caractères tels que nous venons de les définir ne sont pas, en général, des morphismes de G dans $\mathbb{C}^*$. Ce ne sont donc pas des caractères au sens du chapitre I.

La connaissance du caractère χ d'une représentation ρ permet de connaître, pour tout $g \in G$ et tout $k \in \mathbb{N}$, la valeur de $\mathrm{tr}(\rho(g)^k) = \mathrm{tr}(\rho(g^k))$, et si on note $\{\lambda_1, \ldots, \lambda_n\}$ les valeurs propres de $\rho(g)$, cela revient donc à connaître, pour tout $k \in \mathbb{N}$, la valeur de $S_k = \sum_{i=1}^n \lambda_i^k$. S_k est la $k^{\text{ième}}$ somme de Newton associée aux valeurs propres de $\rho(g)$, et donc, en vertu des relations de Newton, ces sommes permettent (moyennant tout de même la résolution d'un système linéaire triangulaire) de calculer la valeurs des $\sigma_i(\lambda_1, \ldots, \lambda_n)$, où les σ_i sont les polynômes symétriques élémentaires. Grâce aux relations coefficients/racines, on connaît donc le polynôme caractéristique P_g de notre endomorphisme $\rho(g)$, et donc (moyennant la recherche des racines de ce polynôme tout de même) les valeurs propres $\{\lambda_1, \ldots, \lambda_n\}$. En conclusion, la connaissance du caractère d'une représentation est en fait équivalente à la connaissance de toutes les valeurs propres de tous les morphismes associés aux éléments de G. Est-ce suffisant pour caractériser une représentation (tout du moins à isomorphisme près) ? C'est à cette question que nous allons essayer de répondre. Mais avant toute chose, voici un lemme classique qui sera utile pour étudier la représentation des morphismes.

Lemme 3.3. *Soit $u \in \mathscr{L}(W)$ et $v \in \mathscr{L}(V)$ deux applications linéaires.*
On définit $\Phi \in \mathscr{L}(\mathscr{L}(V), \mathscr{L}(W))$ par l'égalité $\Phi(f) = u \circ f \circ v$.
On a alors $\mathrm{tr}(\Phi) = \mathrm{tr}(u)\,\mathrm{tr}(v)$.

Démonstration. On se donne des bases $\{e_i\}_{i\in I}$ de V et $\{f_j\}_{j\in J}$ de W, ainsi que les bases duales $\{e_i^*\}_{i\in I}$ et $\{f_j^*\}_{j\in J}$. On peut construire une base $\{F_{i,j}\}_{(i,j)\in I\times J}$ de $\mathscr{L}(V,W)$ par

$$\forall x \in V, \quad F_{i,j}(x) \overset{\text{déf.}}{=} \langle e_i^*, x\rangle f_j \in W.$$

Si les endomorphismes de $\mathscr{L}(V,W)$ sont écrits sous forme matricielle dans les bases (e_i) et (f_j), alors $F_{k,l} = (\delta_{ik}\delta_{jl})_{(i,j)\in I\times J}$. L' élément $F_{k,l}^*$ de la base duale associe à une matrice $(a_{i,j})$ la valeur $a_{k,l}$. La base duale est ainsi définie par la propriété :

$$\forall f \in \mathscr{L}(V,W), \quad \left\langle F_{i,j}^*, f\right\rangle = \left\langle f_j^*, f(e_i)\right\rangle.$$

On a donc

$$\begin{aligned}\mathrm{tr}(\Phi) &\overset{\text{déf.}}{=} \sum_{(i,j)\in I\times J} \left\langle F_{i,j}^*, \Phi(F_{i,j})\right\rangle = \sum_{(i,j)\in I\times J} \left\langle f_j^*, u\circ F_{i,j}\circ v(e_i)\right\rangle \\ &= \sum_{(i,j)\in I\times J} \left\langle f_j^*, u(\langle e_i^*, v(e_i)\rangle f_j)\right\rangle = \sum_{(i,j)\in I\times J} \left\langle f_j^*, u(f_j)\right\rangle \langle e_i^*, v(e_i)\rangle \\ &= \mathrm{tr}(u)\,\mathrm{tr}(v).\end{aligned}$$

□

Dans la suite, si ρ_U est une représentation sur un espace U, nous abrégerons la notation χ_{ρ_U} en χ_U. Commençons par donner les propriétés évidentes des caractères.

Proposition 3.4 (Propriétés des caractères). *On a les propriétés suivantes.*

(i) $\chi_\rho(1) = n$.

(ii) $\forall s \in G, \quad \chi_\rho(s^{-1}) = \overline{\chi_\rho(s)}$.

(iii) $\forall (s,t) \in G^2, \chi_\rho(tst^{-1}) = \chi_\rho(s)$: *on dit que χ_ρ est une fonction centrale (voir paragraphe 5.2) sur G.*

(iv) *Si ρ se décompose en une somme directe de deux représentations ρ_V et ρ_W, alors* $\chi_\rho \overset{\text{déf.}}{=} \chi_{V\oplus W} = \chi_V + \chi_W$.

(v) *Si on note $\rho_{\mathscr{L}(V,W)}$ la représentation des morphismes sur $\mathscr{L}(V,W)$ de deux représentations ρ_V et ρ_W, alors* $\chi_{\mathscr{L}(V,W)} = \overline{\chi_{\rho_V}}\chi_{\rho_W}$.

(vi) *Si on note ρ^* la représentation duale d'une représentation ρ , alors* $\chi_{\rho^*} = \overline{\chi_\rho}$.

(vii) *Deux représentations isomorphes ont même caractère.*

Démonstration.

(i) C'est évident car $\mathrm{tr}(\mathrm{Id}_V) = \dim(V) = n$.

(ii) Ceci vient du fait que l'on peut prendre une matrice unitaire pour $\rho(s)$ et du calcul $\chi_\rho(s^{-1}) = \mathrm{tr}(\rho(s)^{-1}) = \mathrm{tr}(\rho(s)^*) = \overline{\mathrm{tr}(\rho(s))}$.

(iii) Ceci vient du fait que $\forall (A,B) \in GL_n(\mathbb{C})$, $\mathrm{tr}(BAB^{-1}) = \mathrm{tr}(A)$.

(iv) Si on note $\mathscr{B}_V$ une base de V et $\mathscr{B}_W$ une base de W, la matrice de $\rho_{V\oplus W}(s)$ s'écrit dans la base $\mathscr{B} \overset{\text{déf.}}{=} \mathscr{B}_V \cup \mathscr{B}_W$:

$$M(s) = \begin{pmatrix} M_V(s) & 0 \\ 0 & M_W(s) \end{pmatrix},$$

où $M_V(s)$ est la matrice de ρ_V(s) dans la base $\mathscr{B}_V$ et $M_W(s)$ celle de $\rho_W(s)$ dans $\mathscr{B}_W$. D'où

$$\chi_{V\oplus W}(s) = \mathrm{tr}(M(s)) = \mathrm{tr}(M_V(s)) + \mathrm{tr}(M_W(s)) = \chi_V(s) + \chi_W(s).$$

(v) Ceci provient du lemme 3.3, appliqué à $u = \rho_W(s)$ et $v = \rho_V(s^{-1})$.

(vi) Ceci provient du fait que la représentation V^* est isomorphe à la représentation des morphismes $\mathscr{L}(V,K)$, ce qui permet d'utiliser (v), et du fait que $\mathrm{tr}(f^{\mathrm{T}}) = \mathrm{tr}(f)$.

(vii) Même démonstration que pour (iii). □

Exemple 3.5 (Représentation régulière). On note ρ_r la représentation régulière à gauche d'un groupe G, sur un espace de dimension $|G| = n$. Cette représentation correspond à une représentation par permutation des éléments de la base $\{\delta_g\}_{g\in G}$ de $\mathbb{C}[G]$. La présence d'entrées non nulles sur la diagonale de la matrice associée à $\rho_r(g)$ correspond à des points fixes pour la permutation induite par g. Or la permutation induite par un élément différent de l'élément neutre n'a pas de point fixe, puisque

$$\rho_r(g)(\delta_h) = \delta_h \quad \Leftrightarrow \quad gh = h \quad \Leftrightarrow \quad g = e.$$

On a donc $\chi_r(1) = n$ et $\forall s \neq 1, \chi_r(s) = 0$.

3.2 Relations d'orthogonalité

Les caractères sont des éléments (certes un peu particuliers) de l'espace $\mathbb{C}[G]$ des fonctions de G dans $\mathbb{C}$. Une idée importante est que l'on peut munir l'espace vectoriel $\mathbb{C}[G]$ d'une structure d'espace hermitien, et le produit scalaire associé va s'avérer un outil à la fois calculatoire et théorique très efficace.

Définition 3.6 (Produit hermitien). Si φ et ψ sont deux fonctions de G dans $\mathbb{C}$, on pose

$$\langle \varphi, \psi \rangle \overset{\text{déf.}}{=} \frac{1}{|G|} \sum_{t \in G} \varphi(t) \overline{\psi(t)}.$$

$\langle \cdot, \cdot \rangle$ est un *produit hermitien* sur l'espace vectoriel $\mathbb{C}[G]$ des fonctions de G dans $\mathbb{C}$.

Nous allons maintenant réinvestir les propriétés de l'opérateur de Reynolds pour démontrer le premier résultat important de ce chapitre, à savoir l'orthogonalité des caractères irréductibles.

Théorème 3.7 (Relations d'orthogonalité). *Une famille de caractères de représentations irréductibles deux à deux non isomorphes forme une famille orthonormale de l'espace des fonctions de G dans* $\mathbb{C}$*, ce qui signifie que*

- *si* χ *est le caractère d'une représentation irréductible, on a* $\langle \chi, \chi \rangle = 1$.
- *si* χ *et* χ' *sont deux caractères de représentations irréductibles non isomorphes, on a* $\langle \chi, \chi' \rangle = 0$.

Démonstration. Soient ρ_1 et ρ_2 deux représentations de la famille considérée, respectivement sur des espaces vectoriels V et W. Avec la proposition 2.13, on a donc $\mathrm{tr}(R_G) = \delta$, où $\delta = +1$ si les deux représentations sont isomorphes (donc en fait égales), et 0 sinon. Or

$$\mathrm{tr}(R_G) = \frac{1}{|G|} \sum_{s \in G} \mathrm{tr}(\rho_{\mathscr{L}(V,W)})(s) = \frac{1}{G} \sum_{s \in G} \chi_{\rho_{\mathscr{L}(V,W)}}(s).$$

Nous avons vu à la proposition 3.4, (v), que $\chi_{\mathscr{L}(V,W)}(s) = \overline{\chi_V(s)} \chi_W(s)$, donc on a bien

$$\mathrm{tr}(R_G) = \frac{1}{|G|} \sum_{s \in G} \overline{\chi_V(s)} \chi_W(s) \overset{\text{déf.}}{=} \langle \chi_W, \chi_V \rangle = \delta. \qquad \square$$

Corollaire 3.8. *Il y a un nombre fini de classes de représentations irréductibles (sous-entendu de classes pour la relation « être isomorphe »).*

Démonstration. Les caractères des représentations irréductibles non isomorphes forment une famille libre, car orthogonale, de $\mathbb{C}[G]$, qui est un espace de dimension finie sur $\mathbb{C}$. En conséquence, il y a un nombre fini de caractères, donc un nombre fini de représentations irréductibles. Leur nombre est borné par $\dim(\mathbb{C}[G]) = |G|$. $\square$

4 Représentations et dénombrement

Avant d'aller plus loin dans l'étude des caractères, nous pouvons tirer bon nombre de conclusions intéressantes en utilisant seulement l'orthogonalité des caractères, que nous venons de démontrer. En particulier, nous allons pouvoir répondre au problème de l'unicité de la décomposition en représentations irréductibles.

4.1 Décomposition d'une représentation

Dans la suite de l'exposé, on se donne une famille de représentants $(V_i)_{i=1}^p$ de l'ensemble des représentations irréductibles sur G, chaque G-module V_i étant implicitement lié à une

représentation $\rho_i \overset{\text{déf.}}{=} \rho_{V_i}$. Ceci signifie que les V_i sont deux à deux non G-isomorphes, et que toute représentation irréductible W est G-isomorphe à un unique V_i.

Définition 4.1 (Dual d'un groupe fini). On note $\widehat{G}$ l'ensemble des classes d'équivalence des représentations irréductibles sur G pour la relation d'isomorphisme. Par abus de langage, on notera souvent $\widehat{G}$ l'ensemble $(\rho_i)_{i=1}^p$, ce qui correspond à choisir un représentant dans chaque classe. De même, on notera souvent $\sum_{\rho\in\widehat{G}}$ à la place de $\sum_{i=1}^p$.

Proposition 4.2 (Unicité de la décomposition). *Soit une représentation sur V, de caractère χ_V. Alors elle se décompose (c'est-à-dire est G-isomorphe) en*

$$V \simeq \bigoplus_{i=1}^{p} V_i^{\oplus a_i}, \qquad \text{avec} \quad a_i = \langle \chi_V, \chi_i \rangle \tag{4.1}$$

Dans cette relation, on a noté $V_i^{\oplus a_i} \overset{\text{déf.}}{=} V_i \oplus \cdots \oplus V_i$ (a_i fois).
De plus, on a $\langle \chi_V, \chi_V \rangle = \sum_{i=1}^p a_i^2$.

Démonstration. On sait, d'après la proposition 1.32 que la représentation sur V se décompose en somme de q représentations irréductibles $(W_j)_{j=1}^q$:

$$V = W_1 \oplus \cdots \oplus W_q, \tag{4.2}$$

chaque espace W_i étant associé à un morphisme $\rho_{W_i} : G \to GL(W_i)$. Donc d'après le proposition 3.4, (iv), on a $\chi_V = \chi_{W_1} + \cdots + \chi_{W_q}$. Comme on a $\langle \chi_V, \chi_i \rangle = \sum_{j=1}^q \langle \chi_{W_j}, \chi_i \rangle$, et que $\langle \chi_{W_i}, \chi_i \rangle$ vaut 1 si W_i est G-isomorphe à W_i, et 0 sinon, on en déduit que $\langle \chi_V, \chi_i \rangle$ représente le nombre de W_j, pour $j = 1, \ldots, q$, qui sont isomorphes à V_i. Or par définition, c'est a_i.
Au final, dans l'écriture (4.2), on peut regrouper les W_j isomorphes à V_i, et donc écrire $V_i^{\oplus a_i}$ à la place. □

***Remarque* 4.3.** C'est en ce sens que la décomposition d'une représentation V est unique. De plus, si on considère une représentation irréductible W, elle est isomorphe à un certain V_i, et le nombre de fois que W intervient dans la décomposition (c'est-à-dire le nombre de W_i isomorphes à W) est indépendant de la décomposition et vaut $\langle \chi_W, \chi_V \rangle \overset{\text{déf.}}{=} a_i$. En particulier, si l'on dispose de deux décompositions $W = W_1 \oplus \cdots \oplus W_r$ et $W = W_1' \oplus \cdots \oplus W_{r'}'$, alors $r = r'$, et quitte à réordonner les indices, il existe des isomorphismes $W_i \simeq W_i'$.

Corollaire 4.4. *Deux représentations sont isomorphes si et seulement si elles ont le même caractère. De plus, une représentation sur V de caractère χ_V est irréductible si et seulement si $\langle \chi_V, \chi_V \rangle = 1$.*

Démonstration. Le caractère détermine entièrement la décomposition (4.1) en fonction des éléments de $\widehat{G}$, donc détermine la classe d'isomorphisme.
De plus, un caractère est irréductible si et seulement si sa décomposition ne possède qu'un seul terme, c'est-à-dire s'il existe un $j \in \{1, \ldots, p\}$ tel que $a_j = 1$ et si $i \neq j$, alors $a_i = 0$. Ceci est équivalent à $\sum_{i=1}^p a_i^2 = 1$. □

4.2 Résultats de dénombrement

Le point central pour démontrer les relations liant les degrés des représentations irréductibles est l'utilisation de la représentation régulière, puisque nous allons voir qu'elle

contient toutes les autres représentations, et que l'on peut même expliciter sa décomposition :

Proposition 4.5 (Décomposition de la représentation régulière). *On note ρ_r la représentation régulière d'un groupe G, sur un espace vectoriel V de dimension n. Soit χ_r son caractère (cf. exemple 3.5). On reprend les notations du paragraphe 4.1. La décomposition de la représentation régulière sur V (c'est-à-dire en termes de G-isomorphisme) s'écrit*

$$V \simeq \bigoplus_{i=1}^{p} V_i^{\oplus n_i} \qquad \textit{avec} \qquad n_i \overset{\text{déf.}}{=} \dim_{\mathbb{C}}(V_i) = \langle \chi_r, \chi_i \rangle .$$

Démonstration. D'après la proposition 4.2, le nombre de fois que V_i intervient dans la représentation régulière vaut

$$a_i \overset{\text{déf.}}{=} \langle \chi_r, \chi_i \rangle = \frac{1}{|G|} \sum_{s \in G} \chi_r(s^{-1}) \chi_i(s) = \frac{1}{|G|} \chi_r(1) \chi_i(1) = \chi_i(1) = n_i. \qquad \square$$

Corollaire 4.6. *On a les relations :*

(i) $\sum_{i=1}^{p} n_i^2 = |G|$.
(ii) *Pour* $s \neq 1$, $\sum_{i=1}^{p} n_i \chi_i(s) = 0$.

Démonstration. D'après la proposition 4.5, on a $\chi_r(s) = \sum n_i \chi_i(s)$. On en déduit (i) en prenant $s = 1$ et (ii) en prenant $s \neq 1$. $\square$

***Remarque* 4.7.** La relation (i) permet de déterminer si on a, ou non, trouvé toutes les représentations d'un groupe donné, et, le cas échéant, de déterminer la dimension d'une éventuelle représentation manquant à l'appel. Dans ce cas, on peut utiliser la relation (ii) pour déterminer la valeur de ce caractère (voir l'exemple du groupe $\mathfrak{S}_4$, paragraphe 1.4, chap. VIII, pour une application).

5 Théorie de Fourier

Avant d'essayer de développer une théorie des séries de Fourier semblable à celle des caractères sur un groupe commutatif (ceci sera fait au paragraphe 5.3), commençons par définir le morphisme de transformée de Fourier. Cette construction est identique à celle faite dans le cadre des groupes abéliens au paragraphe 4.1, chap. I, puisqu'elle consiste à étendre une représentation sur un groupe G à l'espace des fonctions de G dans $\mathbb{C}$. En effet, nous avons déjà dit qu'une représentation de G s'étend de manière unique en une représentation d'algèbre sur $\mathbb{C}[G]$. Tout naturellement, nous sommes en fait en train de construire la transformée de Fourier, et nous allons voir que, comme dans le cas des fonctions de carré intégrable sur un intervalle réel, cette transformée de Fourier est un morphisme d'algèbre.

5.1 Transformée de Fourier

Définition 5.1 (Transformée de Fourier). Soit $f \in \mathbb{C}[G]$ une fonction de G dans $\mathbb{C}$. On définit, pour ρ une représentation de G sur un espace V, l'application $\pi(f)(\rho)$, par

$$\pi(f)(\rho) = \sum_{s \in G} f(s) \rho(s) \in \mathscr{L}(V,V).$$

Ceci permet de définir l'application transformée de Fourier :

$$\mathscr{F} : \left\{ \begin{array}{ccc} \mathbb{C}[G] & \longrightarrow & \bigoplus_{i=1}^{p} \mathrm{End}(V_i) \\ f & \longmapsto & \{\pi(f)(\rho_i)\}_{i=1}^{p} \end{array} \right. , \tag{5.1}$$

où l'on a noté $\mathrm{End}(V_i) \overset{\text{déf.}}{=} \mathscr{L}(V_i, V_i)$ l'espace des applications linéaires de V_i dans V_i. Par abus de notation, on notera $\mathscr{F}(f)(\rho_i)$ à la place de $\pi(f)(\rho_i) = (\mathscr{F}f)_i$ (la $i^{\text{ième}}$ composante de $\rho_i(f)$).

***Remarque* 5.2. (Transformée de Fourier et représentation sur $\mathbb{C}[G]$).** Cette définition est en fait très naturelle, puisque nous nous sommes contentés de prolonger la représentation ρ définie sur G à une représentation sur l'algèbre $\mathbb{C}[G]$ (c'est-à-dire en un morphisme d'algèbres de $\mathbb{C}[G]$ dans $\mathrm{End}(V)$). En effet, on peut identifier $s \in G$ à l'élément δ_s (c'est l'identification canonique), et on remarque que

$$\forall s \in G, \quad \pi(\delta_s)(\rho) = \rho(s).$$

On veut prolonger ρ en $\widetilde{\rho}$ sur $\mathbb{C}[G]$. Pour définir, pour $f \in \mathbb{C}[G]$, la valeur de $\widetilde{\rho}(f)$, il suffit d'écrire f sous la forme $f = \sum_{s\in G} f(s)\delta_s$. L'unique façon d'effectuer ce prolongement est d'utiliser la linéarité que doit avoir la fonction $\widetilde{\rho}$, et de poser

$$\widetilde{\rho}(f) \overset{\text{déf.}}{=} \sum_{s\in G} f(s)\rho(s).$$

Comme par hasard, c'est la définition que l'on a prise pour $\pi(f)(\rho)$! De cette remarque, on tire immédiatement la proposition suivante.

Proposition 5.3 (Convolution et transformée de Fourier). *L'application transformée de Fourier est un morphisme d'algèbres de* $(\mathbb{C}[G], *)$ *dans* $\bigoplus_{i=1}^{p} (\mathrm{End}(V_i), \circ)$*, autrement dit,*

$$\forall \rho \in \widehat{G}, \forall (f,g) \in \mathbb{C}[G]^2, \quad \mathscr{F}(f*g)(\rho) = \mathscr{F}(f)(\rho) \circ \mathscr{F}(g)(\rho).$$

Démonstration. Il suffit d'utiliser le fait que l'application $f \in \mathbb{C}[G] \mapsto \mathscr{F}f(\rho)$ est l'unique représentation qui étend la représentation ρ à l'espace $\mathbb{C}[G]$. Or une représentation d'une algèbre est un morphisme d'algèbres, d'où la proposition. □

Comme pour la transformée de Fourier sur un groupe abélien, l'application que nous venons de définir est en fait un isomorphisme d'algèbre. C'est ce que nous allons démontrer, en utilisant une fois de plus la représentation régulière.

Proposition 5.4 (Bijectivité de la transformée de Fourier). *L'application transformée de Fourier est un isomorphisme d'algèbres de* $(\mathbb{C}[G], *)$ *sur* $\bigoplus_{i=1}^{p} (\mathrm{End}(V_i), \circ)$.

Démonstration. **Injectivité :** Soit $f \in \mathbb{C}[G]$ telle que $\forall i = 1, \ldots, p$, $\mathscr{F}f(\rho_i) = 0$. Soit alors ρ une représentation quelconque. On peut décomposer ρ en somme des représentations $(\rho_i)_{i=1}^{p}$. Ceci signifie que dans de bonnes bases, la matrice de $\rho(f)$ est formée de tableaux diagonaux des matrices $\mathscr{F}f(\rho_i)$, $i = 1, \ldots, p$, donc qu'elle est nulle.
En appliquant ce résultat à la représentation régulière ρ_r sur l' espace $V = \mathbb{C}[G]$, on obtient $\rho_r(f_r) = 0$, d'où

$$0 = \mathscr{F}f(\rho_r)(\delta_e) \overset{\text{déf.}}{=} f * \delta_e = f.$$

Surjectivité : Nous avons déjà vu au corollaire 4.6 que $\sum_{i=1}^{p} n_i^2 = |G|$. Or on sait que $\dim_{\mathbb{C}}(\mathbb{C}[G])$ est égal à $|G|$ (car les $(\delta_s)_{s\in G}$ forment une base canonique de $\mathbb{C}[G]$), et que

l'on a de plus $\dim_{\mathbb{C}}(\mathrm{End}(V_i)) = n_i^2$ (c'est une algèbre de matrices). Par égalité des dimensions, on en déduit que $\mathscr{F}$, qui est injective, est bijective. □

On peut même aller plus loin en explicitant l'application inverse, grâce à une formule d'inversion qui généralise la formule déjà prouvée dans le cas d'un groupe abélien, à la proposition 4.4, chap. I.

Théorème 5.5 (Formule d'inversion). *Pour $f \in \mathbb{C}[G]$, on a la formule d'inversion suivante :*

$$\forall g \in G, \quad f(g) = \frac{1}{|G|} \sum_{\rho_i \in \widehat{G}} n_i \operatorname{tr}\left(\rho_i(g^{-1})\, \mathscr{F}f(\rho_i)\right),$$

où n_i est le degré de la représentation ρ_i et tr *désigne la trace.*

Démonstration. En utilisant la linéarité des deux membres de l'égalité, il suffit de démontrer la proposition dans le cas où $f = \delta_h$. Le membre de droite de l'égalité se résume alors à

$$\frac{1}{|G|} \sum_{i=1}^{p} n_i \operatorname{tr}(\rho_i(g^{-1})\rho_i(h)) = \frac{1}{|G|} \sum_{i=1}^{p} n_i \chi_i(g^{-1}h).$$

Or, d'après la proposition 4.6, cette dernière quantité vaut 1 si $g^{-1}h = 1$ (c'est-à-dire $g = h$), et 0 sinon. En regardant le membre de droite de l'égalité, qui vaut $\delta_h(g)$, on voit que c'est ce qu'il fallait démontrer. □

5.2 Espace des fonctions centrales

On suppose comme précédemment que l'on dispose d'une famille de représentants $(\chi_i)_{i\in I}$ des caractères des représentations irréductibles sur V, c'est-à-dire de $\widehat{G}$. Nous avons vu que les caractères sur un groupe G ont une propriété importante, puisqu'il sont dans le centre de $\mathbb{C}[G]$ pour le produit de convolution. Ils partagent cette propriété avec une classe de fonctions plus grande, que l'on nomme les fonctions centrales, et que l'on va étudier dans ce paragraphe. Nous allons voir en particulier le résultat primordial de ce chapitre, qui dit que les caractères forment en fait une base de cet espace, et que cette base est même orthonormée.

Commençons par rappeler les définitions liées à l'action de G sur lui-même par conjugaison.

Définition 5.6 (Classes de conjugaison). G agit sur lui même par conjugaison : pour un élément $g \in G$, l'action envoie $h \in G$ sur ghg^{-1}. Les orbites pour cette action sont appelées les *classes de conjugaison* de G. Ainsi, la classe d'un élément $h \in G$ est

$$C_h \overset{\text{déf.}}{=} \left\{ghg^{-1} \setminus g \in G\right\}.$$

Deux éléments sont dits conjugués s'ils appartiennent à la même classe de conjugaison.

Définition 5.7 (Espace des fonctions centrales). Une fonction $\varphi : G \to \mathbb{C}$ est dite centrale si elle vérifie

$$\forall (s,g) \in G^2, \quad \varphi(sgs^{-1}) = \varphi(g).$$

On note $\mathbb{C}[G]^G$ l'ensemble des fonctions centrales sur G : c'est un sous-espace vectoriel de l'espace $\mathbb{C}[G]$ des fonctions de G dans $\mathbb{C}$. Chaque fonction étant constante sur chaque classe de conjugaison, la dimension de $\mathbb{C}[G]^G$ est égale au nombre de ces mêmes classes

(égale, donc, au nombre d'orbites pour l'action de conjugaison de G sur G). De façon plus précise, si on note $\{C_1, \dots, C_q\}$ les différentes classes de conjugaison de G, une base de l'espace $\mathbb{C}[G]^G$ est donnée pas les fonctions $f_{C_1}, \dots, f_{C_q}$, définies par

$$\forall g \in G, \quad f_{C_i}(g) = \begin{cases} 1 & \text{si } g \in C_i \\ 0 & \text{sinon} \end{cases}. \tag{5.2}$$

En fait, les fonctions centrales forment un sous-espace très important de l'algèbre $\mathbb{C}[G]$.

Proposition 5.8. *Les fonctions centrales sont les fonctions $f \in \mathbb{C}[G]$ qui vérifient $\forall \varphi \in \mathbb{C}[G]$, $f * \varphi = \varphi * f$. En d'autres termes, $\mathbb{C}[G]^G$ est le centre de $\mathbb{C}[G]$ pour le produit de convolution $*$.*

Démonstration. Il suffit d'écrire la définition du produit de convolution :

$$(f*\varphi)(g) \overset{\text{déf.}}{=} \sum_{h\in G} f(h)\varphi(h^{-1}g) = \sum_{h'\in G} \varphi(h')f(gh'^{-1}),$$

où l'on a effectué le changement de variable $h' = h^{-1}g$ dans la sommation. On conclut en utilisant, puisque f est centrale, le fait que $f(gh'^{-1}) = f(h'^{-1}g)$. □

***Remarque* 5.9.** La notation $\mathbb{C}[G]^G$ est cohérente avec la théorie des actions de groupes, car les fonctions centrales peuvent être vues comme les éléments invariants de $\mathbb{C}[G]$ sous l'action de conjugaison par G. En effet, G agit par conjugaison sur $\mathbb{C}[G]$ par

$$\forall g \in G, \forall f \in \mathbb{C}[G], \quad g \cdot f : x \mapsto f(gxg^{-1}).$$

Les fonctions centrales forment donc la sous-représentation invariante de $\mathbb{C}[G]$ sous cette action de G sur $\mathbb{C}[G]$.

Lemme 5.10. *Si $f \in \mathbb{C}[G]^G$ est une fonction centrale de G dans $\mathbb{C}$, alors, pour toute représentation irréductible ρ sur un espace V de dimension n, $\mathscr{F}f(\rho)$ est une homothétie de rapport $\frac{|G|}{n}\langle f, \overline{\chi_\rho}\rangle$.*

Démonstration. Commençons par remarquer que $\mathscr{F}f(\rho)$ est un opérateur d'entrelacement pour ρ :

$$\forall s \in G, \quad \rho(s)^{-1}\mathscr{F}f(\rho)\rho(s) = \sum_{t\in G} f(t)\rho(s^{-1})\rho(t)\rho(s) = \sum_{t\in G} f(t)\rho(s^{-1}ts).$$

Donc, en utilisant le changement de variable $u = s^{-1}ts$ et en utilisant le fait que la fonction f est centrale, il vient

$$\forall s \in G, \quad \rho(s)^{-1}\mathscr{F}f(\rho)\rho(s) = \sum_{u\in G} f(sus^{-1})\rho(u) = \sum_{u\in G} f(u)\rho(u) \overset{\text{déf.}}{=} \mathscr{F}f(\rho).$$

On applique alors le cas (ii) du lemme de Schur 2.5 pour voir que $\mathscr{F}f(\rho)$ est une homothétie de rapport λ. Comme sa trace vaut $n\lambda$, on a

$$n\lambda = \sum_{t\in G} f(t)\operatorname{tr}(\rho(t)) = \sum_{t\in G} f(t)\chi_\rho(t) \overset{\text{déf.}}{=} |G|\langle f, \overline{\chi_\rho}\rangle. \qquad \square$$

***Remarque* 5.11.** Cette propriété, démontrée de façon quelque peu calculatoire, traduit simplement le fait que le morphisme d'algèbres $\mathscr{F}$ fait correspondre le centre de $\mathbb{C}[G]$ (c'est-à-dire les fonctions centrales) avec le centre de l'algèbre $\bigoplus_{i=1}^p \operatorname{End}(V_i)$ (c'est-à-dire les éléments qui induisent sur chaque V_i des homothéties). Toutes ces propriétés vont permettre d'affiner le résultat d'orthogonalité des caractères, démontré au théorème 3.7.

Théorème 5.12. *$(\chi_\rho)_{\rho\in\widehat{G}} = (\chi_i)_{i=1}^p$ forme une base orthonormale de l'espace $\mathbb{C}[G]^G$ des fonctions centrales sur G.*

Démonstration. Les (χ_i) forme une famille orthonormale, donc libre, il suffit de montrer qu'elle est génératrice. Dire que les (χ_i) engendrent $\mathbb{C}[G]^G$ est équivalent (car $f \in \mathbb{C}[G]^G \Leftrightarrow \overline{f} \in \mathbb{C}[G]^G$) à dire que les $\overline{(\chi_i)}$ engendrent $\mathbb{C}[G]^G$. Autrement dit, si on prend $f \in \mathbb{C}[G]^G$ orthogonale à $H \overset{\text{déf.}}{=} \text{Vect}\{\overline{\chi_i} \setminus i = 1,\dots,p\}$, on veut montrer que $f = 0$. Or avec le lemme 5.10, on sait que $\mathscr{F}f(\rho_i)$ est une homothétie de rapport $\langle f, \overline{\chi_i}\rangle$, donc est nulle, car f est orthogonale à H. Ceci signifie que la transformée de Fourier de f est nulle, donc $f = 0$ grâce à la proposition 5.4. □

Corollaire 5.13. *Le nombre p de représentations irréductibles sur G non isomorphes (c'est-à-dire le cardinal de $\widehat{G}$) est égal au nombre de classes de conjugaison de G.*

Démonstration. Comme les fonctions de $\mathbb{C}[G]^G$ sont les fonctions constantes sur les classes de conjugaison de G, la dimension de $\mathbb{C}[G]^G$ est égale au nombre de ces classes. On termine en utilisant le fait que les $(\chi_i)_{i=1}^p$ forment une base de $\mathbb{C}[G]^G$. □

***Remarque* 5.14.** Même si on sait que le nombre de représentations irréductibles à isomorphisme près est le même que le nombre de classes de conjugaison, on n'a, a priori, aucun moyen de mettre en relation ces deux types d'objets. Par exemple, étant donnée une classe, on aimerait disposer d'un moyen de construire une représentation irréductible. Dans le cadre du groupe $\mathfrak{S}_n$, on sait le faire, mais la construction est compliquée (se référer au livre de FULTON et HARRIS [35]).

Corollaire 5.15. *G est commutatif si et seulement si toutes ses représentations irréductibles sont de degré 1.*

Démonstration. Si on note p le nombre de classes de conjugaisons, G est commutatif si et seulement si $p = |G|$. Or avec le corollaire 4.6, on a $\sum_{i=1}^p n_i^2 = |G|$, donc G est commutatif si et seulement si $\forall i = 1,\dots,p, n_i = 1$. □

5.3 Séries de Fourier

Ce paragraphe se contente de synthétiser les résultats précédents sous la forme d'une formule de décomposition d'une fonction centrale en *série de Fourier*. On retrouve exactement les mêmes énoncés que pour les séries de Fourier sur un groupe abélien, en se restreignant bien sûr aux fonctions centrales. Dans le chapitre suivant, nous étendrons ces séries de Fourier aux fonctions quelconques de $\mathbb{C}[G]$, mais ceci demandera de laisser de côté nos caractères, pourtant si utiles !

Définition 5.16 (Coefficients de Fourier). Pour $f \in \mathbb{C}[G]$ et pour ρ_i une représentation irréductible, on définit le *coefficient de Fourier* de f en ρ_i par

$$c_f(\rho_i) \overset{\text{déf.}}{=} \langle f, \chi_i\rangle \overset{\text{déf.}}{=} \frac{1}{|G|}\sum_{t\in G} f(t)\overline{\chi_i(t)}, \tag{5.3}$$

où l'on a noté χ_i le caractère de ρ_i.

Proposition 5.17 (Décomposition en série de Fourier). *Soit $f \in \mathbb{C}[G]^G$ une fonction centrale sur G. On a la décomposition de f en série de Fourier :*

$$f = \sum_{\rho\in\widehat{G}} c_f(\rho)\chi_\rho.$$

Démonstration. Ceci provient immédiatement du fait que les (χ_i) forment une base orthonormale de $\mathbb{C}[G]^G$. □

Proposition 5.18 (Formule de Plancherel). *Soient f et g deux fonctions centrales sur G. On a l'identité de Plancherel :*

$$\frac{1}{|G|}\sum_{s\in G} f(s)\overline{g(s)} = \sum_{\rho\in\widehat{G}} c_f(\rho)\overline{c_g(\rho)}.$$

Démonstration. En écrivant $f = \sum_{\chi\in\widehat{G}} c_f(\chi)\chi$ ainsi que $g = \sum_{\chi\in\widehat{G}} c_g(\chi)\chi$, il vient

$$\sum_{s\in G} f(s)\overline{g(s)} = |G|\,\langle f,g\rangle = |G| \sum_{(\chi_1,\chi_2)\in\widehat{G}^2} c_f(\chi_1)\overline{c_g(\chi_2)}\,\langle\chi_1,\chi_2\rangle.$$

On obtient donc l'égalité voulue grâce aux relations d'orthogonalité entre les caractères. □

En conclusion, observons comment cette décomposition d'une fonction centrale s'explique en termes de changement de base. Nous avons déjà vu que la base « naturelle » dans laquelle on représente volontiers une fonction centrale est la base $\{f_{C_1},\dots,f_{C_p}\}$ des fonctions « plateau ». La décomposition en série de Fourier permet de passer de cette base, peu pratique du point de vue calculatoire à la base des caractères, qui a des propriétés beaucoup plus intéressantes vis-a-vis de la convolution. C'est précisément de cette utilisation des caractères qu'il va être question dans le paragraphe suivant.

5.4 Transformée de Fourier et caractères

Dans ce paragraphe, nous nous intéressons à présent aux propriétés des caractères en tant qu'éléments centraux de l'algèbre $\mathbb{C}[G]$. Mais pour étudier les *projecteurs* d'une telle algèbre, nous sommes amenés à utiliser certains concepts d'une portée plus générale.

Définition 5.19 (Idempotents centraux). Soit $\mathscr{A}$ une algèbre associative de dimension finie sur le corps $\mathbb{C}$ des complexes. Un élément $e\in\mathscr{A}$ est appelé *idempotent central* s'il vérifie $e^2 = e$ et $\forall x\in\mathscr{A}$, $e*x = x*e$. Une famille $\{e_\lambda\}_{\lambda\in L}$ (où L est un ensemble fini) est un système d'idempotents orthogonaux si elle vérifie

$$\sum_{\lambda\in L} e_\lambda = 1_{\mathscr{A}} \qquad \forall(\lambda,\mu)\in L^2,\quad e_\lambda * e_\mu = \begin{cases} e_\lambda & \text{si } \lambda=\mu \\ 0 & \text{sinon}\end{cases}.$$

Supposons que l'on dispose d'un isomorphisme d'algèbre

$$\Phi : \mathscr{A} \xrightarrow{\simeq} \bigoplus_{\lambda\in L} \mathrm{End}(V^\lambda) = \mathscr{B},$$

où les V^λ sont des espaces vectoriels de dimension finie. Alors, notons

$$E_\lambda \overset{\text{déf.}}{=} 0\oplus\cdots\oplus 0\oplus \mathrm{Id}_{V^\lambda}\oplus 0\oplus\cdots\oplus 0\in\mathscr{B}.$$

On voit facilement que la famille $\{E_\lambda\}_{\lambda\in L}$ forme un système minimal d'idempotents orthogonaux, puisque l'on dispose d'une description complète du centre de $\mathscr{B}$ (les morphismes qui sont des homothéties sur chaque V^λ). En conséquence, grâce à Φ, il est très

simple de déterminer un système d'idempotents sur $\mathscr{A}$, il suffit de considérer les $\{e_\lambda\}_{\lambda\in L}$ avec $e_\lambda \overset{\text{déf.}}{=} \Phi^{-1}(E_\lambda)$.

Si on se place dans le cas où $\mathscr{A} = \mathbb{C}[G]$, la transformée de Fourier $\mathscr{F}$ définie par l'équation (5.1) va donc nous permettre de construire notre système formé par les $\{e_\lambda\}_{\lambda\in\widehat{G}}$. Ici, l'ensemble L a été pris égal à $\widehat{G}$, puisque à chaque classe de représentation irréductible λ correspond un idempotent e_λ. Le point important est que l'on peut calculer explicitement ces idempotents, en utilisant la formule d'inversion de Fourier, proposition 5.5 :

$$\forall \lambda \in \widehat{G}, \forall g \in G, \quad e_\lambda(g) = \mathscr{F}^{-1}E_\lambda(g) = \frac{n_\lambda}{|G|}\chi_\lambda(g^{-1}),$$

où on a noté n_λ la dimension de la représentation λ. Rappelons les deux propriétés essentielles de nos e_λ :

$$\sum_{\lambda\in\widehat{G}} e_\lambda = \delta_1 \quad \text{et} \quad \forall(\lambda,\mu)\in\widehat{G}^2, \quad e_\lambda * e_\mu = \begin{cases} e_\lambda & \text{si } \lambda = \mu \\ 0 & \text{sinon} \end{cases}.$$

Ces idempotents orthogonaux permettent en particulier de calculer des projections sur des sous-représentations. Soit ρ_U une représentation de G sur un espace U. On étend de façon naturelle cette représentation en une représentation d'algèbre, que l'on note encore ρ_U. En utilisant le langage de la transformée de Fourier, on peut même écrire, pour $f \in \mathbb{C}[G]$, que $\rho_U(f) = \mathscr{F}(f)(\rho_U)$. Pour tout $\lambda \in \widehat{G}$, on définit alors un endomorphisme de U, noté P_λ de la façon suivante :

$$P_\lambda \overset{\text{déf.}}{=} \rho_U(e_\lambda) = \mathscr{F}(e_\lambda)(\rho_U).$$

Par ailleurs, on sait, avec la proposition 4.2, que l'espace U se décompose en somme directe de représentations irréductibles, et même plus précisément :

$$U = \bigoplus_{\lambda\in\widehat{G}} V_\lambda^{\oplus a_\lambda} \quad \text{avec} \quad a_\lambda = \langle\chi_\lambda, \chi_U\rangle \in \mathbb{N},$$

où les V_λ sont les espaces associés aux représentations irréductibles $\lambda \in \widehat{G}$.

Définition 5.20 (Espaces isotypiques). On appelle $U_\lambda \overset{\text{déf.}}{=} V_\lambda^{\oplus a_\lambda}$ la *composante isotypique* de U associée à la représentation irréductible λ.

On peut maintenant énoncer le théorème important de ce paragraphe.

Proposition 5.21. *P_λ est le projecteur sur U_λ associé à la décomposition $U = \bigoplus U_\lambda$.*

Démonstration. Soit V une sous-représentation irréductible. La construction de e_λ montre que $\mathscr{F}(e_\lambda)(\rho_U)$, restreint à V_μ est l'identité si $V \simeq V_\lambda$, et est nul sinon. Ceci veut bien dire que P_λ est le projecteur cherché. □

6 Exercices

Exercice VII.1 (Irréductibilité et indécomposabilité). Soit K un corps. On considère la représentation

$$\begin{cases} K & \longrightarrow & GL_2(K) \\ x & \longmapsto & \begin{pmatrix} 1 & x \\ 0 & 1 \end{pmatrix} \end{cases}.$$

Montrer que si $K = \mathbb{C}$ ou si K est un corps fini, l'espace K^2 est indécomposable mais réductible pour la représentation considérée. En déduire que le théorème 1.29 est faux, si le groupe de départ est infini, ou si le corps de l'espace d'arrivée n'est pas algébriquement clos.

Exercice VII.2 (Opérateurs stationnaires). Soit G un groupe fini, et un opérateur linéaire $A : \mathbb{C}[G] \to \mathbb{C}[G]$. On note, pour $h \in G$, τ_h l'opérateur de translation, c'est-à-dire :

$$\forall f \in \mathbb{C}[G], \quad \tau_h f : g \mapsto f(h^{-1}g).$$

On suppose que A commute avec les translations, c'est-à-dire $A\tau_h = \tau_h A$. Montrer qu'il existe $\varphi \in \mathbb{C}[G]$ telle que l'on ait

$$\forall f \in \mathbb{C}[G], \quad Af = f * \varphi,$$

où $*$ désigne le produit de convolution.

Exercice VII.3 (Représentation irréductible). Soit χ_U le caractère d'une représentation de dimension 1 non triviale, et χ_V celui d'une représentation irréductible. Montrer que $\chi_U \chi_V$ est le caractère d'une représentation irréductible différente de ρ_U et ρ_V.

Exercice VII.4 (Représentation d'un groupe produit). Soient G et H deux groupes finis. Donner des représentants des représentations irréductibles du groupe $G \times H$ en fonction des représentations de G et de H. Quels sont les caractères correspondants ?

Exercice VII.5 (Action sur les polynômes). Soit G un sous-groupe fini de $GL_n(K)$, où K désigne un corps de caractéristique 0. Au paragraphe 1.2, on a défini une représentation de G sur l'espace vectoriel des polynômes en n indéterminées, $K[X_1, \ldots, X_n]$. On rappelle que l'on note $K[X_1, \ldots, X_n]^G$ le sous-espace des polynômes invariants sous cette action. C'est aussi un sous-anneau. On souhaite montrer que ce sous-anneau est engendré par un nombre fini de polynômes.
Dans la suite, on aura besoin des notations suivantes :

$$\forall \alpha = (\alpha_1, \ldots, \alpha_n) \in (\mathbb{N}^+)^n, \quad X^\alpha \overset{\text{déf.}}{=} X_1^{\alpha_1} \cdots X_n^{\alpha_n}.$$

On note alors $|\alpha| = |\alpha_1| + \cdots + |\alpha_n|$ le degré du monôme obtenu. Par commodité, on note aussi

$$(A \cdot X)^\alpha \overset{\text{déf.}}{=} (A \cdot X)_1^{\alpha_1} \cdots (A \cdot X)_n^{\alpha_n}, \text{ avec } (A \cdot X)_i \overset{\text{déf.}}{=} a_{i1}X_1 + \cdots + a_{in}X_n.$$

Le but de cet exercice est de trouver un ensemble de polynômes $\{P_1, \ldots, P_s\}$ générateur de l'anneau $K[X_1, \ldots, X_n]^G$. Ceci signifie que

$$\forall P \in K[X_1, \ldots, X_n]^G, \exists Q \in K[Y_1, \ldots, Y_s], \quad P = Q(P_1, \ldots, P_s). \tag{6.1}$$

Ce théorème est souvent appelé théorème de Noether.

1. Dans le cas où le groupe G est le groupe symétrique $\mathfrak{S}_n$, on fait agir G sur l'espace $K[X_1, \ldots, X_n]$ par permutation des indéterminées. Expliquer pourquoi cette action rentre dans le cadre de cet exercice. Donner alors des générateurs de l'anneau des invariants.
2. On considère les sous-groupes de $GL_2(\mathbb{C})$ suivants :

$$V_4 \overset{\text{déf.}}{=} \left\{ \begin{pmatrix} \pm 1 & 0 \\ 0 & \pm 1 \end{pmatrix} \right\} \quad \text{et} \quad C_2 \overset{\text{déf.}}{=} \{\mathrm{Id}, -\mathrm{Id}\}.$$

Pour chacun d'eux, déterminer l'anneau des invariants, et donner un ensemble de générateurs minimal. Est-ce que la décomposition d'un polynôme de $K[X_1, X_2]$ en fonction de ces générateurs est unique ?

3. On rappelle que l'opérateur de Reynolds pour l'action de G sur $K[X_1,\dots,X_n]$ est défini par
$$\forall f \in K[X_1,\dots,X_n],\quad R_G(f)(X) \overset{\text{déf.}}{=} \frac{1}{|G|}\sum_{A\in G} f(A\cdot X),$$
où on note symboliquement $A\cdot X$ l'action de A sur les indéterminées $X_1,\dots,X_n$. On souhaite montrer le résultat suivant :
$$K[X_1,\dots,X_n]^G \overset{\text{déf.}}{=} K\left[R_G(X^\beta)\ ;\ |\beta|\leqslant|G|\right].$$
Expliquer pourquoi il suffit de montrer que pour tout exposant α, $R_G(X^\alpha)$ s'exprime comme un polynôme en les $R_G(X^\beta)$, $|\beta|\leqslant|G|$.
4. On note
$$(X_1+\cdots+X_n)^k = \sum_{|\alpha|=k} a_\alpha X^\alpha,$$
où les a_α sont des entiers positifs. Soient alors $u_1,\dots,u_n$ de nouvelles indéterminées. On note
$$\forall A\in G,\quad U_A \overset{\text{déf.}}{=} u_1 A_1\cdot X+\cdots+u_nA_n\cdot X.$$
Montrer que l'on a
$$S_k(U_A\ ;\ A\in G)\overset{\text{déf.}}{=}\sum_{A\in G}(U_A)^k = \sum_{|\alpha|=k}|G|a_\alpha R_G(X^\alpha)u^\alpha.$$
On rappelle que tout polynôme symétrique de $K[Y_1,\dots,Y_p]$ s'écrit en fonction des p premières sommes de Newton S_k définies par
$$\forall k\in\{1,\dots,p\},\quad S_k(Y_1,\dots,Y_p)=\sum_{i=1}^{p} Y_i^k.$$
En utilisant cette propriété pour les sommes de Newton $S_k(U_A\ ;\ A\in G)$, montrer qu'il existe un polynôme F à coefficients dans K tel que
$$\sum_{|\alpha|=k} a_\alpha R_G(X^\alpha)u_\alpha = F\left(\sum_{|\beta|=1}|G|a_\alpha R_G(X^\beta)u^\beta,\dots,\sum_{|\beta|=|G|}|G|a_\alpha R_G(X^\beta)u^\beta\right).$$
En déduire le résultat voulu.
5. On considère le groupe
$$C_4\overset{\text{déf.}}{=}\{\mathrm{Id},A,A^2,A^3\},\quad \text{où}\quad A\overset{\text{déf.}}{=}\begin{pmatrix}0&-1\\1&0\end{pmatrix}.$$
Utiliser l'opérateur de Reynolds pour déterminer l'anneau des invariants.

L'exercice VIII.9, propose d'utiliser MAPLE pour calculer l'anneau des invariants par la méthode exposée ici.

Exercice VII.6 (Théorème de Molien). On considère G un sous-groupe fini de $GL_n(\mathbb{C})$, et on souhaite étudier l'action de G sur les polynômes, comme définie au paragraphe 1.2. Plus précisément, pour obtenir une représentation en dimension finie, on considère la restriction de cette action à l'espace $V_s \overset{\text{déf.}}{=} K_s^0[X_1,\dots,X_n]$ des polynômes homogènese de degré s (dans lequel on inclut bien sûr le polynôme nul). On note d_s la dimension de V_s.

On note $\rho_s : G \to GL(V_s)$ l'action ainsi définie. On rappelle qu'une base de V_s est donnée par l'ensemble des monômes de degré s. Pour $i \geqslant 0$, on note a_i le nombre maximum de polynômes de V_s qui sont homogènes, invariants, et linéairement indépendants. Pour étudier ces nombres a_i, on introduit la série formelle de Molien :

$$\Phi(\lambda) \overset{\text{déf.}}{=} \sum_{n=0}^{\infty} a_i \lambda^i.$$

On souhaite montrer le théorème de Molien, qui affirme que :

$$\Phi(\lambda) = \sum_{A \in G} \frac{1}{\det(\mathrm{Id} - \lambda A)} \tag{6.2}$$

1. Montrer que si un polynôme $P \in V_s$ est invariant sous l'action de G, alors chacune de ses composantes homogènes est invariante sous-G.
   Expliquer pourquoi $a_s = \dim(V_s^G)$, où on rappelle que V_s^G désigne le sous-espace vectoriel des invariants.
2. Pour $A \in G$, on note $A^{[s]}$ la matrice de $\rho_s(A)$ dans la base de V_s constituée des monômes homogènes de degré s. On indexera les éléments de cette base dans l'ordre lexicographique $X_1 < \cdots < X_n$. Par exemple, pour $n = 2$:

$$\text{si} \quad A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad \text{alors} \quad A^{[2]} = \begin{pmatrix} a^2 & ac & c^2 \\ 2ab & ad+bc & 2cd \\ b^2 & bd & d^2 \end{pmatrix}.$$

   Montrer que

$$a_s = \frac{1}{|G|} \sum_{A \in G} \mathrm{tr}\left(A^{[s]}\right).$$

3. On note $\omega_1, \ldots, \omega_n$ les valeurs propres de $A \in G$. Quelles sont les valeurs propres de $A^{[s]}$? En déduire que le coefficient en λ^s dans $\det(\mathrm{Id} - \lambda A)^{-1}$ est égal à la trace de $A^{[s]}$. En déduire l'expression (6.2).
4. Dans le cas des groupes G_1 et G_2 rencontrés à l'exercice VIII.9, quelle est l'expression de $\Phi(\lambda)$? En quoi ce résultat permet de simplifier la recherche de générateurs (au sens de (6.1)) de $K[X_1, \ldots, X_n]^G$?

Exercice VII.7 (Lemme de Cauchy-Frobenius). Soit G un groupe fini agissant sur un ensemble fini X. Pour $g \in G$, on note X_g l'ensemble des points fixes de g, c'est-à-dire :

$$X_g \overset{\text{déf.}}{=} \{x \in X \setminus g \cdot x = x\}.$$

1. On note χ_1 le caractère de la représentation triviale sur G. Soit V un espace vectoriel de dimension $|X|$ dont une base est $\{e_x\}_{x \in X}$. Ceci permet de définir une représentation par permutation $\pi : G \to GL(V)$ par les relations

$$\forall g \in G, \forall x \in X, \quad \pi(g)(e_x) \overset{\text{déf.}}{=} e_{g \cdot x}.$$

   Calculer $\langle \chi_1, \chi_\pi \rangle$ à l'aide des $|X_g|$.
2. Démontrer le lemme de Cauchy-Frobenius (parfois attribué à Burnside), à savoir que $\langle \pi, \chi_1 \rangle$ est égal au nombre d'orbites de X sous l'action de G.
3. A partir de deux types de pierres précieuses, combien de colliers différents de 6 pierres un joaillier peut-il construire ? On pourra utiliser une action du groupe diédral D_6 sur l'ensemble $X = \{0,1\}^6$.

4. On suppose que $|X| \geqslant 2$ et que l'action de G sur X est doublement transitive. On rappelle que $\mathrm{Hom}_G(V)$ désigne l'espace des opérateurs d'entrelacement, c'est-à-dire les $f \in \mathscr{L}(V,V)$ tels que $f \circ \pi(g) = \pi(g) \circ f$. Montrer que $\dim(\mathrm{Hom}_G(V)) = 2$. En déduire la décomposition de G-modules $V = \mathbb{C}\mathbf{1} \oplus W$, où W est irréductible. Conclure que la représentation standard de $\mathfrak{S}_n$ est irréductible pour $n \geqslant 2$.

Exercice VII.8 (Représentation et théorie des nombres). Soit G un groupe fini. On souhaite montrer que les dimensions des représentations irréductibles de G sont des diviseurs de G. Cet exercice nécessite quelques connaissances sur les entiers algébriques, que l'on pourra trouver au début du livre de SAMUEL [63].

1. Soit $\rho : G \to V$ une représentation irréductible, et K une classe de conjugaison de G. On définit
$$f \overset{\text{déf.}}{=} \sum_{g \in K} \rho(g) \quad \in GL(V).$$
Montrer que f est une homothétie de rapport $r(\rho,K)$. Déterminer $r(\rho,K)$ en fonction de $\chi_\rho(K)$ (la valeur de χ_ρ sur K), et d_ρ, la dimension de V.
2. Démontrer la relation
$$\frac{|G|}{d_\rho} = \sum_K r(\rho,K)\chi(K^{-1}),$$
la somme portant sur l'ensemble des classes de conjugaison de G.
3. Soit K, K' et K'' trois classes de conjugaison de G. On définit, pour $x \in K''$,
$$a(K,K',x) \overset{\text{déf.}}{=} \mathrm{Card}\left\{(k_1,k_2) \in K \times K' \setminus x = k_1 k_2\right\}.$$
Montrer que $a(K,K',x)$ prend une valeur constante pour $x \in K''$.
On note $a(K,K',K'')$ cette valeur.
4. Montrer la relation
$$r(\rho,K)\, r(\rho,K') = \sum_{K''} a(K,K',K'') r(\rho,K''),$$
la somme portant sur les classes de conjugaison de G.
5. En déduire que $\sum_K r(\rho,K)\mathbb{Z}$ est un sous-anneau de $\mathbb{C}$ de type fini sur $\mathbb{Z}$. Montrer alors que les $r(\rho,K)$ sont des entiers algébriques, puis que $\frac{|G|}{d_\rho}$ est un entier algébrique. Conclure.

Exercice VII.9 (Déterminant d'un groupe). Le lecteur pourra faire le rapprochement entre cet exercice et l'exercice I.1 qui étudie les déterminants circulants. Il s'agit en quelque sorte de généraliser cette notion à un groupe quelconque. Cet exercice est tiré de l'exposé de LAM [41], qui traduit en des termes modernes la découverte de la théorie des représentations par FROBENIUS.
Soit G un groupe fini, et $\rho : G \to GL_n(\mathbb{C})$ une représentation. On rappelle qu'elle s'étend de manière unique en un morphisme d'algèbres $f \mapsto \pi f$ encore noté $\rho : \mathbb{C}[G] \to M_n(\mathbb{C})$. On considère un ensemble d'indéterminées $\{X_g\}_{g \in G}$. Le déterminant du groupe G est noté $\Theta(G)$, et c'est le déterminant de la matrice A de taille $|G| \times |G|$, dont les entrées, indexées par les éléments de G, sont $A_{g,h} \overset{\text{déf.}}{=} X_{gh^{-1}}$. Pour simplifier les notations, on note
$$X \overset{\text{déf.}}{=} \sum_{g \in G} X_g \delta_g,$$

que l'on peut considérer comme un élément générique de l'algèbre $\mathbb{C}[G]$. De même, on définit la valeur de ρ en X :

$$\rho(X) \stackrel{\text{déf.}}{=} \sum_{g \in G} X_g \rho(g),$$

que l'on peut voir comme une matrice à coefficients dans $\mathbb{C}\left[\{X_g \setminus g \in G\}\right]$. Ceci permet de définir le déterminant du groupe G en ρ :

$$\Theta_\rho(G) \stackrel{\text{déf.}}{=} \det(\rho(X)),$$

qui est donc un polynôme à $|G|$ indéterminées.

1. Soit ρ_r la représentation régulière de G. Montrer que $\Theta_{\rho_r}(G) = \Theta(G)$.
2. Soit ρ_U et ρ_V deux représentations de G sur des espaces vectoriels U et V. On note $\rho_{U\oplus V}$ la représentation somme. Montrer que
$$\Theta_{\rho_{U\oplus V}}(G) = \Theta_{\rho_u}(G)\Theta_{\rho_V}(G).$$
3. On considère un système de représentants des représentations irréductibles, $\rho_i : G \to GL_{n_i}(\mathbb{C})$, pour $i = 1, \ldots, p$. Expliquer pourquoi les morphismes d'algèbres associés $\rho_i : \mathbb{C}[G] \to M_{n_i}(\mathbb{C})$ sont surjectifs. Si on note $\rho_i(X) \stackrel{\text{déf.}}{=} \{\lambda_{jk}(X)\}$ (sous forme matricielle), en déduire que les formes linéaires (en chaque X_g, $g \in G$) $\lambda_{jk}(X)$, pour $1 \leqslant j, k \leqslant n_i$ sont indépendantes.
4. Démontrer que le déterminant des matrices de $M_n(\mathbb{C})$, vu comme un polynôme en n^2 variables, est irréductible.
5. Expliquer pourquoi on peut compléter la famille $\{\lambda_{jk}(X)\}$ en une base de l'espace des formes linéaires en les variables $\{X_g \setminus g \in G\}$. En déduire que $\Theta_{\rho_i}(G)$ est irréductible.
6. En remarquant que X_1 n'apparaît que sur la diagonale de $\rho_i(X)$, en déduire si l'on regarde $\Theta_{\rho_i}(G)$ comme un polynôme en X_1, alors son terme de degré $X_1^{n_i-1}$ s'écrit
$$\sum_{g \neq 1} \chi_{\rho_i}(g) X_1^{n_i-1} X_g.$$
En déduire que la connaissance de $\Theta_{\rho_i}(G)$ détermine ρ_i, puis que les $\Theta_{\rho_i}(G)$, pour $i = 1, \ldots, p$ sont deux à deux non proportionnels.
7. Conclure que la décomposition de $\Theta(G)$ en facteurs irréductibles sur $\mathbb{C}$ s'écrit
$$\Theta(G) = \prod_{i=1}^{p} \Theta_{\rho_i}(G)^{n_i}.$$

Exercice VII.10 (Groupe affine sur un corps fini). Cet exercice introduit des notions importantes que l'on utilisera à l'exercice suivant. Soit p un nombre premier. On considère le groupe des transformations affines inversibles de $\mathbb{F}_p$, qui sont de la forme

$$\Phi_{a,b} : \left\{ \begin{array}{ccc} \mathbb{F}_p & \longrightarrow & \mathbb{F}_p \\ x & \longmapsto & ax+b \end{array} \right. ,$$

où $a \in \mathbb{F}_p^*$ et $b \in \mathbb{F}_p$. On note G_p ce groupe.

1. Montrer que l'identification de $\Phi_{a,b}$ avec le couple $(b, a) \in \mathbb{F}_p \times \mathbb{F}_p^*$ permet de définir G_p comme un produit semi-direct. Quel est l'élément neutre ? Donner les formules définissant le produit de deux éléments de ce groupe ainsi que l'inverse d'un élément.

2. On définit une application
$$\pi : \left\{ \begin{array}{ccc} G_p & \longrightarrow & \mathbb{C}[\mathbb{F}_p] \\ (b,a) & \longmapsto & \left(f_{(b,a)} : x \mapsto f(a^{-1}(x-b))\right) \end{array} \right. .$$
Montrer qu'il s'agit en fait d'une représentation unitaire (pour le produit hermitien usuel sur $\mathbb{C}[\mathbb{F}_p]$) du groupe G_p.
3. On considère le sous-espace $E \subset \mathbb{C}[\mathbb{F}_p]$:
$$E \stackrel{\text{déf.}}{=} \left\{ f \in \mathbb{C}[\mathbb{F}_p] \;\middle\backslash\; \sum_{x \in \mathbb{F}_p} f(x) = 0 \right\}. \tag{6.3}$$
Montrer que E est un sous-espace invariant sous l'action de π, et que la restriction de π à E définit une représentation irréductible.

Exercice VII.11 (Transformée en ondelettes sur $\mathbb{F}_p$). Cet exercice est tiré d'un article de FLORNES et de ses collaborateurs [32]. Il s'agit de construire une transformée en ondelettes sur le corps $\mathbb{F}_p$, en utilisant le langage de la théorie de représentations. On reprend les notations de l'exercice précédent.
Soit $\psi \in \mathbb{C}[\mathbb{F}_p]$, que l'on nommera *ondelette*. On définit, pour $f \in \mathbb{C}[\mathbb{F}_p]$ la transformée en ondelettes $\mathscr{W}(f) \in \mathbb{C}[G_p]$:
$$\mathscr{W}(f) : \left\{ \begin{array}{ccc} G_p & \longrightarrow & \mathbb{C} \\ (b,a) & \longmapsto & p \left\langle f, \psi_{(b,a)} \right\rangle = \sum_{x \in \mathbb{F}_p} f(x) \overline{\psi(a^{-1}(x-b))} \end{array} \right. .$$
1. Exprimer $\mathscr{W}(f)(b,a)$ en fonction de $\widehat{f}$ (la transformée de Fourier de f sur le groupe additif $\mathbb{F}_p$).
2. On suppose que $\psi \in E$. Montrer alors que si $f \in E$, on a la formule d'inversion :
$$\forall x \in \mathbb{F}_p, \quad f(x) = \frac{1}{c_\psi} \sum_{(b,a) \in G_p} \mathscr{W}(f)(b,a) \psi_{(b,a)}(x),$$
où l'on a noté $c_\psi \stackrel{\text{déf.}}{=} p^2 \langle \psi, \psi \rangle$. On pourra penser à calculer la transformée de Fourier des deux membres.
3. Soit maintenant une ondelette ψ telle que
$$(p-1)|\widehat{\psi}(0)|^2 = \sum_{k=1}^{p-1} |\widehat{\psi}(k)|^2.$$
Montrer que $\mathscr{W}$ est, à une constante d_ψ près, une isométrie de $\mathbb{C}[\mathbb{F}_p]$ sur $\mathbb{C}[G_p]$. Montrer que son inverse est donné par la formule
$$\forall x \in \mathbb{F}_p, \quad f(x) = \frac{1}{d_\psi} \sum_{(b,a) \in G_p} \mathscr{W}(f)(b,a) \psi_{(b,a)}(x),$$
avec $d_\psi = (p-1)|\widehat{\psi}(0)|^2$.
4. Ecrire un algorithme de transformée en ondelettes sur $\mathbb{F}_p$, ainsi que la transformée inverse. Représenter graphiquement les résultats obtenus pour diverses ondelettes et fonctions de test.

La figure 7.1 représente quelques transformées. La colonne de droite représente le module de $\mathscr{W}(f)(b,a)$ (translation b en abscisse, dilatation a en ordonnée), plus la couleur est noire, plus le coefficient est grand.

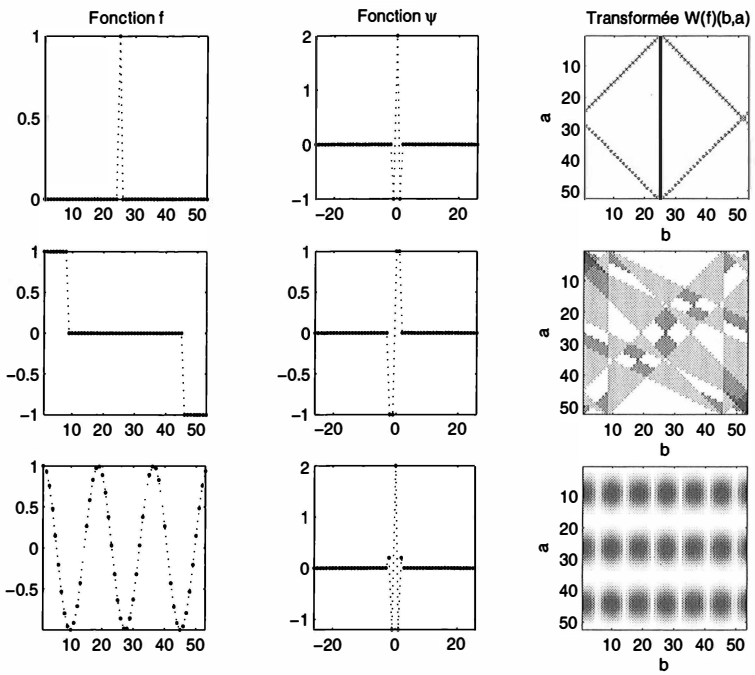

FIG. 7.1 – *Transformée en ondelettes sur* $\mathbb{F}_{53}$

Chapitre VIII
Applications des représentations linéaires

> C'est pourquoi les balles de tennis et les étoiles sont des sphères ; la terre serait également une sphère si elle ne tournait pas autour d'un axe. [...] Le phénomène qui demande une explication n'est donc pas cette symétrie de rotation mais bien les écarts par rapport à cette symétrie ...
>
> H. WEYL [77] (1952)

Les représentations linéaires ont de nombreuses applications, principalement en algèbre théorique. Même dans le cadre simple des groupes finis, cette théorie permet de démontrer des résultats difficiles. Sans aller très loin dans cette direction, le deuxième paragraphe montre comment, à partir de la connaissance des caractères d'un groupe (c'est-à-dire d'informations sur la façon dont notre groupe agit sur des objets extérieurs), on peut déduire des informations sur les sous-groupes qui le composent. Avant toute chose, et pour fournir un peu de matériel d'étude, le premier paragraphe étudie certains groupes finis importants. Enfin, le dernier paragraphe, qui clôt ce livre, transpose le problème de l'analyse de données dans le cadre des groupes non commutatifs.

1 Représentation de groupes classiques

La mise en pratique de la théorie développée dans ce chapitre passe par l'étude de groupes élémentaires mais qui interviennent de façon constante aussi bien en physique théorique ou en cristallographie qu'en mathématiques. Nous allons donc déterminer la liste de représentations irréductibles de ces groupes, leurs caractères, en essayant de retrouver les différentes significations géométriques de nos groupes (groupes d'isométries d'une figure, action sur les faces, les arêtes, etc.).

1.1 Table des caractères

Comme les caractères sont constants sur les classes de conjugaison $C_1, \ldots, C_p$ de G, il nous suffit de dresser un tableau des valeurs des caractères $(\chi_i)_{i=1}^p$ sur ces classes. Nous allons donc considérer les quantités $\chi_i(g_j)$, où g_j est un représentant de la classe C_j. Dans la suite, on place toujours en première position la représentation triviale, de sorte que $\chi_1 = 1$. Par commodité, on indique aussi les cardinaux k_j des différentes classes C_j.

Enfin, on utilise le fait que $\chi_i(1) = n_i$, pour établir une table, qui est une matrice carrée de taille p :

	1	k_2	$\dots$	k_p
	1	g_2	$\dots$	g_p
χ_1	1	1	$\dots$	1
χ_2	n_2	$\chi_2(g_2)$	$\dots$	$\chi_2(g_p)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
χ_p	n_p	$\chi_p(g_2)$	$\dots$	$\chi_p(g_p)$

Nous avons vu, au paragraphe 5.2, chap. VII, que les caractères forment une base orthonormée de l'espace des fonctions centrales. Ceci se traduit, sur la table de caractères, par des relations d'orthogonalité entre les lignes de la table, en prenant bien soin d'affecter chaque colonne j du poids k_j. En fait, on a aussi des relations similaires sur les colonnes de la matrice comme le précise la proposition suivante :

Proposition 1.1 (Orthogonalité des colonnes). *Si on note $\chi_\lambda(C_1)$ la valeur du caractère χ_λ sur la classe de conjugaison C_1, on a*

$$\sum_{\lambda\in\widehat{G}} \chi_\lambda(C_1)\chi_\lambda(C_2) = \begin{cases} \frac{|G|}{|C_1|} & \text{si } C_1 = C_2 \\ 0 & \text{sinon} \end{cases}.$$

Démonstration. Soit C une classe de conjugaison. On rappelle que l'on note f_C la fonction caractéristique de cette classe (cf. équation (5.2), chap. VII). Calculons ses coefficients de Fourier en utilisant la formule de définition (5.3), chap. VII :

$$\forall \lambda \in \widehat{G}, \quad c_{f_C}(\lambda) = \frac{1}{|G|}\sum_{g\in G} f_C(g)\overline{\chi_\lambda(g)} = \frac{|C|}{|G|}\overline{\chi_\lambda(C)}.$$

En prenant successivement $C = C_1$ puis $C = C_2$ dans cette formule, puis en utilisant la formule de *Plancherel*, équation (5.18), chap. VII, on obtient le résultat voulu. □

1.2 Les groupes cycliques

Un groupe cyclique étant commutatif, d'après le corollaire 5.15, chap. VII, il n'a que des représentations de dimension 1, c'est-à-dire des caractères au sens premier du terme (des morphismes de G dans le groupe multiplicatif $\mathbb{C}^*$). Soit $G = \{1, g_0, g_0^2, \dots, g_0^{n-1}\}$ un groupe cyclique fini de cardinal n et de générateur g_0. Soit $\omega_n = e^{\frac{2i\pi}{n}}$. Nous avons déjà vu que tous les éléments de $\widehat{G}$ sont alors de la forme, pour $i \in \{0, \dots, n-1\}$,

$$\chi_i : \begin{cases} G & \longrightarrow & \mathbb{C}^* \\ g = g_0^k & \longmapsto & (\omega_n^i)^k = e^{\frac{2i\pi ik}{n}} \end{cases}.$$

En particulier, on a $G \simeq \widehat{G}$. On peut donc écrire la table de $\widehat{\mathbb{Z}/n\mathbb{Z}}$, qui est une matrice de *Vandermonde* :

	1	$k_2 = 1$	$\dots$	$k_n = 1$
	$g_1 = 0$	$g_2 = 1$	$\dots$	$g_n = n-1$
χ_1	1	1	$\dots$	1
χ_2	1	ω_n		ω_n^{n-1}
χ_3	1	ω_n^2		$\omega_n^{2(n-1)}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
χ_n	1	ω_n^{n-1}	$\dots$	$\omega_n^{(n-1)(n-1)}$

1.3 Les groupes diédraux

Définition 1.2 (Groupe diédral). On appelle groupe diédral D_n le groupe des isométries du plan qui conservent un polygone régulier à n côtés. Il contient n rotations d'angle $\frac{k\pi}{n}$, $k = 0, \dots, n-1$ qui forment un sous-groupe isomorphe à C_n, ainsi que n symétries. Si on note r la rotation d'angle $\frac{2\pi}{n}$ et s une des symétries, alors on a les relations

$$r^n = 1 \qquad s^2 = 1 \qquad (sr)^2 = 1.$$

Selon qu'un élément de D_n appartient ou non à C_n, un élément de D_n s'écrit de manière unique sous la forme $s^i r^k$ avec $k = 0, \dots, n-1$ et $i = 0, 1$. De plus on a $x \in C_n \Leftrightarrow i = 0$.

Notons tout d'abord qu'il nous suffit de donner les valeurs des différentes représentations et des différents caractères pour les deux générateurs r et s.

Cas où n est pair :
Une représentation ρ de degré un (ou son caractère, puisque c'est la même chose) doit vérifier $\psi(s)^2 = 1$, c'est-à-dire $\psi(s) = \pm 1$. Elle doit aussi vérifier $\psi(sr)^2 = 1$, donc $\psi(r) = \pm 1$ et $\psi(r)^n = 1$. Comme n est pair, la condition sur r s'écrit $\psi(r) = \pm 1$. Au final, on obtient les 4 représentations suivantes :

	r^k	sr^k
ψ_1	1	1
ψ_2	1	-1
ψ_3	$(-1)^k$	$(-1)^k$
ψ_4	$(-1)^k$	$(-1)^{k+1}$

Pour les représentations de degré deux, posons $\omega_n = e^{\frac{2i\pi}{n}}$. Nous allons définir pour $h \in \mathbb{N}$ une représentation sur D_n par les formules

$$\rho_h(r^k) = \begin{pmatrix} \omega_n^{hk} & 0 \\ 0 & \omega_n^{-hk} \end{pmatrix} \qquad \rho_h(s^k) = \begin{pmatrix} 0 & \omega_n^{-hk} \\ \omega_n^{hk} & 0 \end{pmatrix}.$$

On vérifie que ces formules définissent bien une représentation. De plus, on peut prendre $h \in \{0, \dots, n-1\}$, et les représentations ρ_h et ρ_{n-h} sont isomorphes, puisque

$$\forall g \in G, \quad \rho_h(g) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \rho_{n-h}(g) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^{-1}$$

On en vient donc à ne considérer que les représentations ρ_h pour $h = 0, \dots, n/2$. La représentation correspondant au cas $h = 0$ est réductible, puisque les droites $\mathbb{C}(e_1 + e_2)$ et $\mathbb{C}(e_1 - e_2)$ sont stables. Il en est de même pour le cas $h = n/2$. On peut aussi constater que $\chi_{\rho_0} = \psi_1 + \psi_2$ et que $\chi_{\rho_{n/2}} = \psi_3 + \psi_4$, ce qui prouve que les représentations ρ_0 et $\rho_{n/2}$ sont réductibles, et permet de connaître leur décomposition. Pour les autres valeurs de h, la représentation ρ_h est bien irréductible. En effet, si ρ_h admettait une sous-représentation non triviale, il s'agirait d'une droite, et on voit qu'une droite stable par $\rho_h(r)$ est nécessairement un axe de coordonnée, qui n'est pas laissé stable par $\rho_h(sr)$. On peut calculer les caractères de ces $n/2 - 1$ représentations irréductibles :

	r^k	sr^k
χ_h	$2\cos\left(\frac{2\pi hk}{n}\right)$	0

On voit donc que ces représentations ne sont pas isomorphes (car leurs caractères sont différents). Pour vérifier que l'on a bien ainsi construit toutes les représentations, il suffit de calculer la somme des carrés des degrés des représentations. Au total, on obtient bien $4 \times 1 + (n/2 - 1) \times 4 = 2n = |D_n|$.

Cas où n est impair :
Cette fois-ci, on ne peut avoir que deux représentations de degré un :

	n r^k	n sr^k
ψ_1	1	1
ψ_2	1	-1

On définit les représentations ρ_h comme dans le cas où n est pair. Pour $1 \leqslant h \leqslant (n-1)/2$, ces représentations sont irréductibles et deux à deux non isomorphes. Leurs caractères ont déjà été calculés dans le cas n pair. En calculant la somme des carrés des degrés, on obtient $2 \times 1 + (n-1)/2 \times 4 = 2n = |D_n|$. On a ainsi énuméré toutes les représentations irréductibles.

1.4 Le groupe $\mathfrak{S}_4$

La première chose à faire est de déterminer les classes de conjugaison de $\mathfrak{S}_4$, le groupe des permutations d'un ensemble à 4 éléments, identifié à $\{1,2,3,4\}$. On utilise pour ce faire le lemme 1.36, chap. VII, et on obtient donc

- la classe de l'identité, qui correspond à la décomposition $4 = 1+1+1+1$, c'est-à-dire en quatre 1-cycles. Elle possède 1 élément.
- la classe des transpositions, par exemple de l'élément (12), qui correspond à la décomposition $4 = 2+1+1$. Elle possède 6 éléments (choix de 2 éléments parmi 4 sans ordre, ce qui fait C_4^2).
- la classe des trois cycles, par exemple de l'élément (123), qui correspond à la décomposition $4 = 3+1$. Elle comporte 8 éléments (4 choix possibles de 3 éléments parmi 4, et 2 cycles possibles par choix).
- la classe des quatre cycles, par exemple de l'élément (1234), qui correspond à la décomposition $4 = 4$. Elle comporte 6 éléments (24 permutations que l'on regroupe par paquets de 4 4-cycles identiques).
- la classe des couples de 2-cycles disjoints, par exemple de l'élément $(12)(34)$, qui correspond à la décomposition $4 = 2+2$. Elle comporte 3 éléments (6 choix possibles pour la première transposition, et le choix de la deuxième divise par deux le nombre de possibilités).

Par le corollaire 5.13, chap. VII, nous savons que $\mathfrak{S}_4$ admet, à isomorphisme près, 5 représentations irréductibles. Nous avons déjà déterminé un certain nombre de représentations au paragraphe 1.4, chap. VII :

- la représentation triviale, sur un espace U (de dimension 1), de caractère $\chi_1 = (1,1,1,1,1)$ (on note ainsi la ligne correspondante dans le tableau des caractères. On indexe les colonnes dans le même ordre que celui utilisé pour les classes de conjugaison).
- la représentation alternée, sur un espace V (de dimension 1), qui correspond à la signature et a pour caractère $\chi_\varepsilon = (1,-1,1,-1,1)$.

– La représentation standard, sur un espace V_s (de dimension 3), dont le caractère χ_s, d'après la décomposition trouvée au paragraphe 1.4, chap. VII, vérifie $\chi_p = \chi_s + \chi_1$ (on a noté χ_p le caractère de la représentation par permutation des éléments d'une base). Or la valeur $\chi_p(\sigma)$ correspond au nombre d'éléments laissés fixes par σ, ce qui donne $\chi_p = (4,2,1,0,0)$. Au final, on a donc $\chi_s = (3,1,0,-1,-1)$. On remarque que l'on a

$$|G|\langle\chi_s,\chi_s\rangle = 1\chi_s(\mathrm{Id})^2 + 6\chi_s((12))^2 + 8\chi_s((123))^2 + 6\chi_s((1234))^2 + 3\chi_s((12)(34))^2 = 24.$$

D'où $\langle\chi_s,\chi_s\rangle = 1$, donc d'après le corollaire 4.4, chap. VII, la représentation standard de $\mathfrak{S}_4$ est irréductible.

On obtient pour l'instant une table des caractères partielle :

	1	6	8	6	3
	Id	(12)	(123)	(1234)	(12)(34)
χ_1	1	1	1	1	1
χ_ε	1	−1	1	−1	1
χ_s	3	1	0	−1	−1

Il reste encore deux représentations à déterminer, et en utilisant la relation (i) du corollaire 4.6, chap. VII, on a $n_4^2 + n_5^2 = 13$, où l'on a noté n_4 et n_5 les degrés des deux représentations. On a donc nécessairement une représentation de degré 3 et l'autre de degré 2. La première représentation peut s'obtenir par l'intermédiaire de la représentation des morphismes sur $W \overset{\text{déf.}}{=} \mathscr{L}(V_s,V_\varepsilon)$ des représentations standard et alternée. Elle est de degré 3, et son caractère est $\chi_{\mathscr{L}(W,V)} = \chi_W\overline{\chi_V} = (3,-1,0,1,-1)$. On remarque qu'il est bien différent des caractères déjà déterminés, et que $\langle\chi_{\mathscr{L}(W,V)},\chi_{\mathscr{L}(W,V)}\rangle = 1$, donc cette représentation est bien irréductible (voir l'exercice VII.3 pour généralisation). Pour déterminer la dernière représentation, sur un espace noté W' (de dimension 2), on utilise la relation (ii) du corollaire 4.6, chap. VII, et on trouve $\chi_{W'} = (2,0,-1,0,2)$. Au final, on obtient la table des caractères :

	1	6	8	6	3
	Id	(12)	(123)	(1234)	(12)(34)
χ_1	1	1	1	1	1
χ_ε	1	−1	1	−1	1
χ_s	3	1	0	−1	−1
χ_W	3	−1	0	1	−1
$\chi_{W'}$	2	0	−1	0	2

Une des réalisations concrètes du groupe $\mathfrak{S}_4$ est le groupe des isométries directes conservant un cube. On peut voir cette réalisation par l'action du groupe sur les quatre grandes diagonales du cube. En conséquence, le groupe agit aussi en permutant les faces du même cube, ce qui donne naissance à une représentation par permutation du groupe $\mathfrak{S}_4$, c'est-à-dire $\rho_E : \mathfrak{S}_4 \to GL(E)$, où E est un espace vectoriel de dimension 6. Comme pour toute représentation par permutation, la valeur de $\chi_E(\sigma)$, pour $\sigma \in \mathfrak{S}_4$ est égale au nombre de faces fixées par l'action de σ. Identifions les différentes valeurs de ce caractère :

– une rotation de 180° sur un axe reliant les milieux de deux côtés opposés : cette permutation échange seulement deux diagonales. Elle correspond à la classe de (12). Aucune face n'est fixée.

– une rotation de 120° selon une grande diagonale : seule la diagonale en question est invariante, les autres permutant circulairement. Elle correspond à la classe de (123). Aucune face n'est fixée.
– une rotation de 90° selon un axe de coordonnées : permute en cercle les quatre diagonales. Elle correspond à la classe de (1234). Deux faces sont fixées.
– une rotation de 180° selon un axe de coordonnées : permute deux par deux les diagonales. Elle correspond à la classe de (12)(34). Deux faces sont fixées.

Le caractère de notre représentation est donc donné par

	Id	(12)	(123)	(1234)	(12)(34)
χ_E	6	0	0	2	2

On a $\langle \chi_\rho, \chi_\rho \rangle = 3$, donc notre représentation s'écrit comme somme de trois représentations irréductibles. Pour calculer la décomposition de cette représentation, il suffit de calculer les différents produits scalaires :

$$\begin{array}{lll} \langle \chi_E, \chi_1 \rangle = 1, & \langle \chi_E, \chi_\varepsilon \rangle = 0, & \langle \chi_E, \chi_s \rangle = 0, \\ \langle \chi_E, \chi_W \rangle = 1, & \langle \chi_E, \chi_{W'} \rangle = 1. & \end{array}$$

On obtient ainsi la décomposition $E = \mathbb{C} \oplus W \oplus W'$, en tant que somme de G-modules.

2 La question de la simplicité

Dans ce paragraphe, nous allons utiliser la théorie des caractères pour obtenir des informations sur la structure de notre groupe. Nous allons nous intéresser à la recherche de sous-groupes distingués.

2.1 Noyau des caractères

Commençons par une proposition, qui va permettre de caractériser le noyau des représentations.

Proposition 2.1. *Soit G un groupe fini, et $\rho : G \to GL(V)$ une représentation, de caractère χ_V sur un espace V de dimension d. On note $g \in G$ un élément d'ordre k. Alors :*

(i) *$\rho(g)$ est diagonalisable.*
(ii) *χ_V est somme de $\chi_V(1) = \dim(V) = d$ racines $k^{\text{ièmes}}$ de l'unité.*
(iii) *$|\chi_V(g)| \leqslant \chi_V(1) = d$.*
(iv) *$K_{\chi_V} \overset{\text{déf.}}{=} \{x \in G \setminus \chi_V(x) = \chi_V(1)\}$ est un sous-groupe distingué de G. On le nomme le noyau de la représentation.*

Démonstration.

(i) Comme $g^k = 1$, on a $\rho(g)^k = \mathrm{Id}$. Donc le polynôme minimal de $\rho(g)$ divise $X^k - 1$, qui est scindé à racine simple.
(ii) Soient $\omega_1, \ldots, \omega_d$ les valeurs propres de $\rho(g)$, qui sont des racines $k^{\text{ièmes}}$ de l'unité. On a $\chi_V(g) = \omega_1 + \cdots + \omega_d$.
(iii) $|\chi_V(g)| \leqslant |\omega_1| + \cdots + |\omega_d| = d$.

(iv) Si $|\chi_V(g)| = d$, on a égalité dans l'inégalité triangulaire précédente. Ceci signifie que les nombres complexes ω_i sont positivement liés sur $\mathbb{R}$. Comme ils sont de module 1, ils sont tous égaux. Si $\chi_V(g) = d$, on a nécessairement $\omega_i = 1$, donc $\rho(g) = \mathrm{Id}$. Donc $K_{\chi_V} = \ker(\rho)$, est bien un sous-groupe distingué. □

Dans la suite, nous aurons besoin du lemme suivant.

Lemme 2.2. *Soit $N \lhd G$ un sous-groupe distingué de G. Soit ρ_U une représentation de G/N sur un espace vectoriel U. Alors il existe une représentation canonique de G sur U telle que les sous-représentations de U sous l'action de G/N soient exactement celles de U sous l'action de G.*

Démonstration. Il suffit de poser

$$\forall g \in G, \quad \widetilde{\rho}_U(g) \overset{\text{déf.}}{=} \rho_U \circ \pi(g),$$

où $\pi : G \to G/N$ est la projection canonique. $\widetilde{\rho}_U$ définit bien la représentation cherchée. □

2.2 Utilisation de la table des caractères

Soit G un groupe fini. On note $\widehat{G} = \{\rho_1, \ldots, \rho_r\}$ son dual, formé de représentants des représentations irréductibles non isomorphes. Voici le résultat qui va nous permettre de déterminer l'ensemble des sous-groupes distingués d'un groupe donné.

Proposition 2.3. *Les sous-groupes distingués de G sont exactement du type*

$$\bigcap_{i \in I} K_{\chi_i} \qquad \text{où} \quad I \subset \{1, \ldots, r\}.$$

Démonstration. Soit $N \lhd G$ un sous-groupe distingué. On note ρ_U la représentation régulière de G/N. Ceci signifie donc que U est un espace vectoriel de dimension égale à $|G/N| = |G|/|N|$, de base $\{e_g\}_{g \in G/N}$, et l'on a $\rho_U(h)(e_g) = e_{hg}$.
Nous avons déjà vu à la proposition 1.11, chap. VII, que la représentation régulière est fidèle, donc ρ_U est injective. En utilisant le lemme 2.2, on étend cette représentation en une représentation $\widetilde{\rho}_U : G \to U$. Notons χ le caractère de la représentation $\widetilde{\rho}_U$. On a alors l'égalité $\ker(\widetilde{\rho}_U) = \ker(\rho_U \circ \pi) = N$, d'où $N = K_\chi$.
Il ne reste plus qu'à décomposer la représentation $\widetilde{\rho}_U$ en fonction des représentations irréductibles, pour obtenir

$$\chi = a_1\chi_1 + \cdots + a_r\chi_r.$$

On a donc, en utilisant le point (iii) de la proposition 2.1,

$$\forall g \in G, \quad |\chi(g)| \leqslant \sum_{i=1}^{r} a_i |\chi_i(g)| \leqslant \sum_{i=1}^{r} a_i |\chi_i(1)| = \chi(1).$$

On a donc l'égalité $\chi(g) = \chi(1)$ (c'èst-à-dire $g \in K_\chi$) si et seulement si on a une égalité dans l'inégalité triangulaire précédente. Il s'en suit que $\chi(g) = \chi(1)$ si et seulement si $\forall i, a_i\chi_i(g) = a_i\chi_i(1)$. Ceci est finalement équivalent à

$$\forall i, \quad a_i > 0 \Longrightarrow g \in K_{\chi_i}.$$

On a donc bien le résultat voulu, avec $I \overset{\text{déf.}}{=} \{i \setminus a_i > 0\}$.

Enfin, la réciproque est évidente : en effet, comme les K_{χ_i} sont distingués, tout sous-groupe du type $\cap_{i\in I} K_{\chi_i}$ l'est aussi. □

Corollaire 2.4. *G est simple si et seulement si $\forall i \neq 1, \forall g \in G, \chi_i(g) \neq \chi_i(1)$.*

Démonstration. Si on suppose qu'il existe $g \in G$, avec $g \neq 1$, tel que $\chi_i(g) = \chi_i(1)$, alors $K_i \subset G$ est un sous-groupe distingué non trivial, donc G n'est pas simple.
Réciproquement, si G est non simple, alors il existe $g \neq 1$ dans un certain sous-groupe distingué $N \lhd G$ non trivial. Avec la proposition précédente, $N = \cap_{i\in I} K_i$, donc $g \in K_i$ pour $i \in I \subset \{2, \ldots, r\}$. Ceci signifie bien que $\chi_i(g) = \chi_i(1)$. □

Grâce à la table des caractères, on est donc en mesure de dresser la liste de tous les sous-groupes distingués d'un groupe G donné, et même de déterminer les relations d'inclusion entre ces sous-groupes. Par exemple, on peut considérer le groupe $\mathfrak{S}_4$, dont la table a été établie au paragraphe 1.4. On voit qu'il possède deux sous-groupes distingués non triviaux : $\ker(\chi_\varepsilon) = \mathfrak{A}_4$ ainsi que $\ker(\chi_{W'}) = \langle (12)(34) \rangle$ (la classe de la permutation $(12)(34)$). De plus, on voit que $\ker(\chi'_W) \subset \ker(\chi_\varepsilon)$.

3 Analyse spectrale

Nous avons vu au chapitre précédent que la famille des caractères d'un groupe fini constituait une base orthogonale de l'espace des fonctions centrales. Le résultat fondamental de ce paragraphe est la généralisation de ce résultat à l'espace des fonctions de G dans $\mathbb{C}$ tout entier. Bien sûr, il va falloir considérer une autre famille de fonctions, qui intervient de manière naturelle lorsque l'on essaie de calculer de façon matricielle la transformée de Fourier. Cette méthode pour trouver des bases orthonormées d'un espace fonctionnel est à la base de l'analyse spectrale sur un groupe fini quelconque, qui a de nombreuses applications, notamment en statistiques.

3.1 Orthogonalité des fonctions coordonnées

Les caractères sont avant tout des objets théoriques pour la recherche des représentations d'un groupe G (grâce aux relations d'orthogonalité des lignes et des colonnes de la table des caractères), et pour l'étude du groupe G lui même (étude de sa simplicité, résolubilité, etc.). D'une manière pratique, le fait qu'ils forment une base uniquement de l'espace des fonctions centrales les rend peu utiles pour analyser une fonction de G dans $\mathbb{C}$ quelconque. Pour résoudre cette difficulté, nous préférerons utiliser la transformée de Fourier telle qu'elle est définie au paragraphe précédent. Nous allons même voir que, grâce à une certaine formulation matricielle, cette transformée correspond aussi au calcul d'une décomposition dans une base orthogonale.

On considère comme d'habitude un groupe fini G, et on note $\widehat{G} = \{\rho_1, \ldots, \rho_p\}$ les représentants des classes de représentations irréductibles. Chaque représentation ρ_k est liée à un espace V_k de dimension n_k, et ces différentes représentations sont bien sûr deux à deux non isomorphes. Nous avons vu, à la proposition 1.27, chap. VII, que l'on pouvait, pour chaque représentation ρ_k, trouver une base de V_k dans laquelle les matrices $R_k(g)$ des endomorphismes $\rho_k(g)$ sont unitaires. Nous allons noter ces matrices sous la forme $R_k(g) = \{r^k_{ij}(g)\}$. On obtient ainsi une série d'applications :

$$\forall k \in \{1, \ldots, p\}, \forall (i,j) \in \{1, \ldots, n_k\}^2, \quad r^k_{ij} : G \to \mathbb{C}.$$

Plus précisément, on obtient ainsi $\sum_{k=1}^{p} n_k^2 = n$ éléments de $\mathbb{C}[G]$. La proposition suivante, qui est le cœur des développements qui vont suivre, nous dit que ces éléments ne sont pas quelconques.

Théorème 3.1 (Orthogonalité des fonctions coordonnées). *Les r_{ij}^k pour $k \in \{1,\dots,p\}$ et pour $(i,j) \in \{1,\dots,n_k\}^2$, forment une base orthogonale de $\mathbb{C}[G]$. D'une façon plus précise, on a*

$$\forall (k,l) \in \{1,\dots,p\}^2, \forall (a,b,c,d) \in \{1,\dots,n_k\}^4, \quad \left\langle r_{ab}^k, r_{cd}^l \right\rangle = \delta_a^c \delta_b^d \delta_k^l \frac{1}{n_k}.$$

Démonstration. Il s'agit en fait de reformuler le résultat du paragraphe 2.4, chap. VII. Soit en effet ρ_k et ρ_l deux représentations irréductibles. On sait, que pour $f \in \mathscr{L}(U_k,U_l)$, l'application $\widetilde{f} \overset{\text{déf.}}{=} R_G(f) \in \mathscr{L}(U_k,U_l)$ est un opérateur d'entrelacement. D'après le lemme de Schur, c'est soit une homothétie de rapport $\frac{\operatorname{tr}(f)}{n_k}$ (si $k = l$), soit le morphisme nul (si $k \neq l$).

Dans les bases que l'on a choisies pour U_k et U_l, le morphisme f s'écrit sous forme matricielle $\{x_{ij}\}_{i=1\dots d_l}^{j=1\dots n_k}$. De même, on écrit la matrice de $\widetilde{f}$ sous la forme $\{\widetilde{x}_{ij}\}_{i=1\dots d_l}^{j=1\dots n_k}$. On peut calculer explicitement la valeur des $\widetilde{x}_{ij}$:

$$\widetilde{x}_{i_2 i_1} \overset{\text{déf.}}{=} \frac{1}{|G|} \sum_{j_1,j_2,g \in G} r_{i_2 j_2}^l(g) x_{j_2 j_1} r_{j_1 i_1}^k(g^{-1}). \tag{3.1}$$

Commençons par le cas où les représentations ne sont pas isomorphes, c'est-à-dire $k \neq l$. Le fait que $\widetilde{f} = 0$ est équivalent à $\widetilde{x}_{i_2 i_1} = 0$, et ceci quels que soient les $x_{j_2 j_1}$. L'expression de $\widetilde{x}_{i_2 i_1}$ définit une forme linéaire en $x_{j_2 j_1}$, qui est nulle. Ceci veut donc dire que ses coefficients sont nuls. En remarquant que $r_{i_1 j_1}(g^{-1}) = \overline{r_{j_1 i_1}(g)}$, on obtient ainsi, dans le cas où $k \neq l$,

$$\forall (i_1,j_1) \in \{0,\dots,n_k\}^2, \forall (i_2,j_2) \in \{0,\dots,n_l\}^2,$$
$$\frac{1}{|G|} \sum_{g \in G} \overline{r_{j_1 i_1}^k(g)} r_{j_2 i_2}^l(g) \overset{\text{déf.}}{=} \left\langle r_{j_1 i_1}^k, r_{j_2 i_2}^l \right\rangle = 0.$$

Il reste maintenant le cas où $k = l$. On a cette fois-ci $\widetilde{f} = \frac{\operatorname{tr}(f)}{n_i} \mathrm{Id}$, d'où

$$\forall (i_1,j_1) \in \{0,\dots,n_k\}^2, \forall (i_2,j_2) \in \{0,\dots,n_l\}^2 \quad \widetilde{x}_{i_2 i_1} \overset{\text{déf.}}{=} \frac{1}{d_i} \left(\sum_{j_1,j_2} \delta_{j_1}^{j_2} x_{j_2 j_1} \right) \delta_{i_1}^{i_2}.$$

En réutilisant l'expression de $\widetilde{x}_{i_2 i_1}$ obtenue à l'équation (3.1), et en égalant les coefficients de la forme linéaire obtenue, on a la formule

$$\frac{1}{|G|} \sum_{g \in G} \overline{r_{j_1 i_1}^k(g)} r_{j_2 i_2}^l(g) \overset{\text{déf.}}{=} \left\langle r_{j_1 i_1}^k, r_{j_2 i_2}^l \right\rangle = \frac{1}{n_i} \delta_{i_1}^{i_2} \delta_{j_1}^{j_2}. \qquad \square$$

***Remarque* 3.2.** Comme les caractères des représentations irréductibles sont des sommes des fonctions coordonnées différentes, ce résultat affirme en même temps l'orthogonalité des caractères, que nous avons déjà démontrée au théorème 3.7, chap. VII.

On note $I \overset{\text{déf.}}{=} \{(k,i,j) \setminus k = 1,\dots,p \text{ et } i,j = 1,\dots,n_k\}$. Le résultat que nous venons de démontrer affirme l'existence d'une base orthonormée de l'espace $\mathbb{C}[G]$, que l'on note sous la forme $\{\Delta_{kij}\}_{(k,i,j) \in I}$. On remarque que l'on a bien sûr $|I| = |G|$, qui est la dimension de $\mathbb{C}[G]$. Ces fonctions sont définies de la manière suivante :

$$\forall (k,i,j) \in I, \quad \Delta_{(k,i,j)} \overset{\text{déf.}}{=} \sqrt{n_k} r_{ij}^k.$$

3.2 Séries de Fourier généralisées

Le résultat fondamental du paragraphe précédent met donc à notre disposition une base orthogonale de l'espace des fonctions de G dans $\mathbb{C}$. On ne peut s'empêcher de faire la comparaison avec le résultat déjà obtenu grâce à la théorie des caractères au théorème 5.12, chap. VII. Or, il est important de comprendre que ces deux constructions n'ont strictement rien à voir. Les caractères sont définis de façon canonique. Ils ne dépendent pas du choix d'une quelconque écriture matricielle de nos représentations. Il s'agit avant tout d'un outil théorique pour obtenir des informations sur les représentations (par exemple savoir si une représentation est irréductible) ou sur le groupe lui même (pour déterminer les sous-groupes distingués). En revanche, on peut construire une quantité de bases orthonormées de $\mathbb{C}[G]$ grâce aux fonctions coordonnées. Il suffit d'appliquer aux matrices des différentes représentations unitaires un changement de base unitaire. Il s'agit donc d'un outil calculatoire. Le seul cas où ces deux constructions coïncident est celui des groupes finis commutatifs. En effet, les représentations irréductibles d'un tel groupe sont de dimension 1, et l'unique entrée des matrices correspondantes est égale (à une constante près) au caractères de la représentation. On voit d'ailleurs que dans ce cas particulier, la construction des fonctions coordonnées, non canonique dans le cas général, devient canonique.

Nous souhaitons maintenant appliquer la construction que nous venons d'effectuer à l'analyse d'une fonction $f \in \mathbb{C}[G]$. On suppose donc que l'on dispose d'une base orthonormée $\{\Delta_{kij}\}_{(k,i,j)\in I}$. On définit alors les coefficients de Fourier par rapport à cette base.

Définition 3.3 (Coefficients de Fourier). Pour $f \in \mathbb{C}[G]$, on appelle *coefficients de Fourier* par rapport à la base $\{\Delta_{kij}\}_{(k,i,j)\in I}$, et on note $c_f(k,i,j)$ les quantités

$$\forall (k,i,j) \in I, \quad c_f(k,i,j) \stackrel{\text{déf.}}{=} \langle f, \Delta_{kij} \rangle .$$

On a donc le développement de Fourier suivant :

$$f = \sum_{(k,i,j)\in I} c_f(k,i,j)\Delta_{kij}.$$

On peut ensuite se demander quel lien il existe entre les coefficients de Fourier que nous venons d'introduire, et la transformée de Fourier définie en 5.1, chap. VII. Le calcul de la transformée de Fourier d'une fonction $f \in \mathbb{C}[G]$ est équivalent au calcul, pour toute représentation irréductible ρ_k, de chaque coefficient de $\mathscr{F}(f)(\rho_k)$, c'est-à-dire de

$$\forall (k,ij) \in I, \quad \mathscr{F}(f)(\rho_k)_{ij} = \sum_{g\in G} f(s)\,(\rho_k(g))_{ij} = c_h(k,i,j), \tag{3.2}$$

où l'on a noté $h \stackrel{\text{déf.}}{=} \frac{1}{\sqrt{n_k}}\overline{f}$. On voit donc que le calcul des coefficients de Fourier est totalement équivalent à celui du calcul de la transformée de Fourier. En continuant à exploiter les analogies entre ces deux notions, on peut aussi dire que le calcul de la transformée s'apparente à un calcul de changement de bases. On s'aperçoit en effet qu'à condition de remplacer f par son conjugué, puis de normaliser le résultat (en le multipliant par $\sqrt{n_i}$), le calcul de la transformée de Fourier (sous forme matricielle) revient en fait à passer de la base canonique des δ_g à la base orthonormée des Δ_{kij}.

Une des questions est de savoir si l'on dispose, à l'instar de l'algorithme FFT sur les groupes abéliens, d'un algorithme de calcul rapide de la transformée de Fourier sur un

groupe non commutatif. On peut en effet constater qu'une implémentation naïve des équations (3.2) nécessite $O(|G|^2)$ opérations. L'article de synthèse de ROCKMORE [52] explique qu'il existe de tels algorithmes pour de larges classes de groupes, notamment les groupes symétriques dont il est question au prochain paragraphe.

3.3 La représentation du signal

Le problème fondamental du traitement du signal est celui de la représentation (au sens premier du terme) des données étudiées. Le langage de l'algèbre linéaire permet de formaliser ce problème de façon concise et élégante. Les signaux que l'on souhaite analyser peuvent en effet être vus comme des fonctions $f : D \to \mathbb{C}$ où D est un domaine a priori quelconque (par exemple un carré dans le cas d'une image). Dans le cadre d'un traitement informatique, on est amené à considérer des domaines D finis. Le problème de la représentation d'un signal fini peut alors se résumer en la recherche d'une « bonne » base de l'espace vectoriel de dimension finie formé des fonctions de D dans $\mathbb{C}$. D'un point de vue pratique, la qualité de notre base se mesurera en sa capacité de simplifier notre façon d'appréhender les données à analyser. En particulier, il faudra que la représentation des données dans la nouvelle base soit plus simple, plus *creuse* que dans la base d'origine.

La première propriété importante que l'on souhaite pour la base cherchée est d'être orthonormée. Ceci permet d'avoir des formules d'analyse et de reconstruction simples, et plus robustes d'un point de vue numérique. C'est exactement ce que nous avons fait lors des différents calculs de transformées de Fourier déjà rencontrés. En second lieu, la recherche d'un bonne base nécessite d'exploiter les symétries du domaine D. Même si ce point peut paraître sans rapport avec l'efficacité de la base (a priori, il n'y a aucune raison pour que les signaux étudiés suivent les symétries du domaine), l'exploitation des symétries est essentielle pour obtenir des algorithmes de calcul rapides. Par exemple, si l'algorithme FFT est si rapide, c'est parce qu'il exploite totalement la symétrie (périodicité) de l'ensemble $\mathbb{Z}/n\mathbb{Z}$, ce qui permet d'éviter au maximum tout calcul superflu. Dans la pratique, cette propriété de respect de symétrie est en fait également importante pour la représentation des fonctions, car la majeure partie des signaux « naturels » respectent les régularités du domaine d'origine. L'exemple le plus frappant est l'étude de signaux musicaux stationnaires par décomposition en série de Fourier. On observe, après quelques harmoniques fondamentales, des coefficients qui décroissent très rapidement : la représentation fréquentielle d'un tel signal est beaucoup plus compacte que sa représentation temporelle.

Pour essayer d'exploiter les idées développées au paragraphe précédent, il semble naturel de vouloir munir D d'une structure de groupe fini. Ceci laisse le plus souvent une grande latitude pour le choix d'une base orthonormée. D'une part, il existe une multitude de structures, qui peuvent être non isomorphes, et même si deux structures sont isomorphes, l'une peut être mieux adaptée que l'autre au signal étudié. D'autre part, nous avons déjà expliqué que le choix de différentes bases pour le calcul des matrices des représentations irréductibles donnait naissance à des bases orthonormées différentes. Ainsi, l'exercice VIII.7 propose d'utiliser la théorie des représentations pour trouver une base orthonormée de l'espace des fonctions de $\{0,1\}^n$ dans $\mathbb{C}$. Ceci fait écho aux exercices VI.4 et VI.5 qui utilisent la base de Walsh (c'est-à-dire les caractères abéliens) pour étudier les fonctions booléennes. Nous allons maintenant voir sur un exemple concret comment effectuer ces choix de structures et de bases, et s'en servir pour analyser un ensemble de données.

L'exemple que nous allons mentionner maintenant est tiré du livre de DIACONIS [26], qui a été le premier à appliquer la théorie des représentations aux statistiques. Pour un panorama complet des algorithmes de calculs rapides en théorie des représentations, on pourra se reporter à l'article de ROCKMORE [52]. On considère le résultat d'un sondage où l'on a demandé à un nombre conséquent de personnes de classer par ordre de préférence les trois lieux d'habitation suivants : *ville* (proposition 1), *banlieue* (proposition 2), *campagne* (proposition 3). Chaque personne répond à l'enquête en donnant une permutation des trois propositions. Par exemple, la permutation $(2,3,1)$ correspond au classement banlieue, puis campagne, puis ville. Voici les résultats de l'enquête :

ville	banlieue	campagne	résultat
1	2	3	242
2	1	3	170
3	2	1	359
1	3	2	28
2	3	1	12
3	1	2	628

L'ensemble des résultats peut ainsi être vu comme une fonction $f : \mathfrak{S}_3 \to \mathbb{N}$, qui à chaque permutation de $(1,2,3)$ assigne le nombre de personnes ayant donné pour réponse cette permutation. Le problème qui se pose maintenant est celui de l'analyse de ces résultats. La permutation avec le plus fort résultat (en l'occurrence $(3,1,2)$) nous donne quelques informations sur les préférences des personnes interrogées. Mais pour analyser les interactions entre les différentes permutations, il faut utiliser une analyse plus fine.

Nous allons donc effectuer un changement de base, et calculer la façon dont f se décompose dans une base orthogonale obtenue grâce aux représentations irréductibles du groupe $\mathfrak{S}_3$. Outre les représentations ρ_1, triviale, et ρ_2, alternée, il y a une représentation irréductible de dimension 2, la représentation standard ρ_3. L'exercice VIII.3 propose une méthode géométrique pour trouver les matrices orthogonales associées. Nous proposons ici un autre choix de base. En l'occurrence, si on note $\{e_1, e_2, e_3\}$ la base canonique de $\mathbb{C}^3$, on choisit $\{(e_1 - e_2)/\sqrt{2}, (e_1 + e_2 - 2e_3)/\sqrt{6}\}$ pour base orthonormée de l'orthogonal de $e_1 + e_2 + e_3$. Les matrices de la représentation ρ_3 s'écrivent dans cette base :

$$\rho_3((1,2,3)) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad \rho_3((2,1,3)) = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix},$$

$$\rho_3((3,2,1)) = \frac{1}{2}\begin{pmatrix} 1 & -\sqrt{3} \\ -\sqrt{3} & -1 \end{pmatrix}, \qquad \rho_3((1,3,2)) = \frac{1}{2}\begin{pmatrix} 1 & \sqrt{3} \\ \sqrt{3} & -1 \end{pmatrix},$$

$$\rho_3((2,3,1)) = \frac{1}{2}\begin{pmatrix} -1 & \sqrt{3} \\ -\sqrt{3} & -1 \end{pmatrix}, \qquad \rho_3((3,1,2)) = \frac{1}{2}\begin{pmatrix} -1 & -\sqrt{3} \\ \sqrt{3} & -1 \end{pmatrix}.$$

En calculant les produits scalaires entre la fonction f et les 6 fonctions coordonnées de ces représentations, on peut décomposer les fonctions f comme suit :

$$f = \frac{1}{6}(1439\rho_1 + 325\rho_2 - 109\rho_{311} - 1640.2\rho_{312} + 493.6\rho_{321} - 203\rho_{322}),$$

où l'on a noté ρ_{3ij} la fonction coordonnée (i,j) de la représentation matricielle ρ_3. Le premier coefficient, le plus important, correspond à la valeur moyenne de la fonction. Il n'est donc pas très informatif. Par contre, on constate que le coefficient de ρ_{312} est

nettement plus important que tous les autres. Il correspond à la composante de f sur la fonction

$$\rho_{321} = (0, 0, -0.87, 0.87, 0.87, -0.87),$$

où l'on a énuméré les valeurs de ρ_{312} sur les éléments de $\mathfrak{S}_3$ dans le même ordre que celui des résultats du sondage. On constate que cette fonction effectue en fait un groupement des réponses en 3 paquets de 2 permutations (suivant que la valeur est -0.87, 0 ou 0.87), chaque paquet étant caractérisé par le choix du lieu classé en dernier. Le meilleur estimateur après la moyenne correspond donc ici au choix du lieu de résidence le moins apprécié.

4 Exercices

Exercice VIII.1 (Orthogonalité des caractères). On note $\Phi \overset{\text{déf.}}{=} \{\chi_i(C_j)\}_{1 \leqslant i,j \leqslant p}$ la table des caractères. On note $K = \mathrm{diag}(k_1, \ldots, k_p)$ la matrice diagonale dont les entrées sont les cardinaux des classes de conjugaison. Montrer que l'on a $\Phi K \Phi^* = |G|\mathrm{Id}$, c'est-à-dire que la matrice $\frac{1}{\sqrt{|G|}} K^{1/2} \Phi$ est unitaire. En déduire une autre démonstration de la formule d'orthogonalité des colonnes, proposition 1.1.

Exercice VIII.2 (Représentation du groupe diédral). On considère le groupe diédral D_n. Montrer qu'on peut le réaliser géométriquement comme le groupe formé des transformations suivantes :

- les rotations autour de l'axe Oz et d'angles $\frac{2k\pi}{n}$, pour $k = 0, \ldots, n-1$.
- les symétries par rapport aux droites du plan Oxy formant des angles $\frac{k\pi}{n}$ avec l'axe Ox, pour $k = 0, \ldots, n-1$.

On obtient ainsi une représentation $\rho : D_n \to O_3(\mathbb{R})$. Est-elle irréductible ? Calculer son caractère, et en déduire la décomposition de cette représentation.

Exercice VIII.3 (Représentations de $\mathfrak{S}_3$). On considère le triangle dont les trois sommets ont pour affixes $1, e^{\frac{2i\pi}{3}}, e^{-\frac{2i\pi}{3}}$, et on fixe la base $\{1, \mathrm{i}\}$ du plan complexe. Le groupe $\mathfrak{S}_3$ agit sur les sommets du triangle en les permutant. Calculer les matrices de deux générateurs de ce groupe (par exemple (12) et (123)). En déduire la table des caractères du groupe $\mathfrak{S}_3$.

Exercice VIII.4 (Action sur les faces d'un cube). Comme indiqué au paragraphe 1.4, le groupe $\mathfrak{S}_4$ peut être considéré comme le groupe des isométries directes du cube. Il agit donc en permutant l'ensemble à 8 éléments formé par les sommets du cube, ce qui donne naissance à une représentation de dimension 8. Calculer le caractère de cette représentation. En utilisant la table des caractères de $\mathfrak{S}_4$, en déduire une décomposition de cette représentation. Faire de même avec la permutation des arêtes.

Exercice VIII.5 (Caractère de $\mathfrak{S}_4$). On considère le caractère $\chi_{W'}$ de $\mathfrak{S}_4$ dont la table est donnée par

	Id	(12)	(123)	(1234)	(12)(34)
$\chi_{W'}$	2	0	-1	0	2

1. On note $\rho_{W'}$ la représentation associée. Montrer que $\rho_{W'}((12)(34)) = \mathrm{Id}$.

2. Montrer que si $H \subset G$ est un sous-groupe distingué, alors une représentation $\rho : G \to GL(V)$ est triviale sur H si et seulement si elle se factorise par G/H en $\widehat{\rho}$:

$$G \xrightarrow{\pi} G/H \xrightarrow{\widehat{\rho}} GL(V)$$

c'est-à-dire que l'on peut identifier les représentations de G/H avec les représentations triviales sur H.

3. On note H le sous-groupe de $\mathfrak{S}_4$ engendré par $(12)(34)$. Montrer que $\mathfrak{S}_4/H \simeq \mathfrak{S}_3$. Par exemple, on pourra considérer l'action de $\mathfrak{S}_4$ sur les faces opposées d'un cube.
4. Conclure en montrant que $\rho_{W'}$ est en fait la représentation standard de $\mathfrak{S}_3$.

Exercice VIII.6 (Représentation d'un groupe simple). Soit G un groupe fini simple non abélien. On souhaite montrer que G ne possède pas de représentation irréductible de dimension 2.

1. Commencer par montrer le lemme de Cauchy : si p est un nombre premier qui divise $|G|$, alors G possède un élément d'ordre p. Pour se faire, on pourra considérer l'ensemble $X = G^p$, ainsi que l'action du groupe $\mathbb{Z}/p\mathbb{Z}$ sur X :

$$\begin{cases} (\mathbb{Z}/p\mathbb{Z}, G) & \longrightarrow & X \\ (k, (x_{\overline{0}}, \ldots, x_{\overline{p-1}})) & \longmapsto & (x_{\overline{k}}, \ldots, x_{\overline{p-1+k}}) \end{cases},$$

à laquelle on appliquera l'équation aux classes (se reporter au livre de PERRIN [58]).

2. On suppose que G possède une représentation irréductible $\rho : G \to GL_2(\mathbb{C})$. En admettant le résultat de l'exercice VII.8, en déduire que G possède un élément t d'ordre 2.
3. Montrer que ρ est en fait à valeur dans $SL_2(\mathbb{C})$. Montrer ensuite que $\rho(t) \in SL_2(\mathbb{C})$ doit être égal à $-\mathrm{Id}$. En déduire que t est dans le centre de G. Conclure.

Exercice VIII.7 (Groupe quaternionique). On note H_8 le groupe quaternionique, qui est formé des 8 éléments $\{\pm 1, \pm i, \pm j, \pm k\}$ dont les multiplications sont données par la règle des signes et les formules

$$i^2 = j^2 = k^2 = -1, \quad jk = -kj = i, \quad ki = -ik = j, \quad ij = -ji = k.$$

On nomme $H = \mathbb{R}[H_8]$ l'algèbre des quaternions. Pour plus d'informations sur les quaternions, on pourra consulter [58].

1. On note $q = a1 + bi + cj + dk$ un élément générique de H. Montrer que l'application :

$$q \mapsto \begin{pmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{pmatrix}$$

permet d'identifier H à une sous-algèbre de $M_4(\mathbb{R})$. En déduire que H est bien une algèbre associative. En déduire aussi une représentation de H_8.

2. Montrer que l'application

$$\varphi : q \mapsto M(a + \mathrm{i}b, c - \mathrm{i}d) \quad \text{avec} \quad M(\alpha, \beta) \overset{\text{déf.}}{=} \begin{pmatrix} \alpha & -\overline{\beta} \\ \beta & \overline{\alpha} \end{pmatrix}$$

permet d'identifier H à une sous-algèbre de $M_2(\mathbb{C})$. En déduire une représentation de dimension 2 de H_8 sur le corps des complexes. Est-elle unitaire ?

3. Calculer les 4 représentations irréductibles de dimension 1 de H_8. Montrer qu'avec la représentation obtenue à la question précédente, on dispose de toutes les représentations irréductibles. Donner alors la base orthonormée correspondante de l'espace des fonctions de H_8 dans $\mathbb{C}$.
4. Expliquer comment H_8 permet de définir une structure de groupe non-commutatif sur l'espace $\{0,1\}^3$. En utilisant le résultat de l'exercice VII.4 décrire les représentations du groupe $\{0,1\}^{3n}$ vu comme produit du groupe $\{0,1\}^3$. En déduire une base orthonormée de l'espace des fonctions de $\{0,1\}^{3n}$ dans $\mathbb{C}$.

On pourra rapprocher cette construction de celle de la base de Walsh rencontrée à la section 2, chap. II, qui consistait à utiliser la structure de groupe additif abélien de $\{0,1\}^n$. On a, en quelque sorte, raffiné la construction pour utiliser une structure non-commutative. Il existe des applications importantes de ce type de constructions, par exemple de telles bases orthonormées permettent de généraliser la technique d'apprentissage de fonctions booléennes présentée à l'exercice VI.5. C'est BONEH, dans [8] qui a le premier introduit ce procédé.

Exercice VIII.8 (Anneau des invariants). On considère G le groupe des isométries directes de $\mathbb{R}^3$ conservant un cube centré en l'origine et dont les arêtes sont alignées avec les axes de coordonnées. Cet exercice ne suppose pas connu l'isomorphisme entre G et $\mathfrak{S}_4$. On garde les notations de l'exercice VII.5, et on souhaite déterminer géométriquement des éléments de $K[X,Y,Z]^G$.

1. Expliquer pourquoi $X^2+Y^2+Z^2 \in K[X,Y,Z]^G$.
2. Montrer que, si on note $f \overset{\text{déf.}}{=} XYZ$, alors

$$\forall A \in G, \quad f(A\cdot(X,Y,Z)) = af(X,Y,Z), \quad \text{pour } a \in \mathbb{R}.$$

   Montrer ensuite que l'on a nécessairement $a = \pm 1$.
   En conclure que $(XYZ)^2 \in K[X,Y,Z]^G$.
3. De même, montrer que les polynômes

$$f = (X+Y+Z)(X+Y-Z)(X-Y-Z)(X-Y-Z)$$
$$\text{et} \qquad g = (X^2-Y^2)(X^2-Z^2)(Y^2-Z^2)$$

   sont de carré invariant sous G.

Exercice VIII.9 (Codes correcteurs auto-duaux). Soit $\mathscr{C}$ un code linéaire sur $\mathbb{F}_2$ de taille n et de dimension k. On note $\mathscr{C}^\perp$ son code dual et $W_{\mathscr{C}}(X,Y)$ le polynôme énumérateur de poids de $\mathscr{C}$. On suppose que $\mathscr{C}$ est auto-dual, c'est-à-dire $\mathscr{C} = \mathscr{C}^\perp$.

1. Quelle relation doivent vérifier n et k?
2. On note A la matrice 2×2 définie par

$$A \overset{\text{déf.}}{=} \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

   En utilisant les identités de MacWilliams 4.5, chap. VI, montrer que $W_{\mathscr{C}}(X,Y)$ est invariant sous l'action de A (comme définie au paragraphe 1.2, chap. VII).
3. On note G_1 le groupe engendré par A. Ecrire les éléments qui le composent. Expliquer pourquoi $W_{\mathscr{C}}(X,Y) \in K[X,Y]^{G_1}$. En utilisant le résultat de l'exercice VII.5, montrer que $K[X,Y]^{G_1}$ est engendré, au sens de (6.1), chap. VII, par les polynômes

$$X+(\sqrt{2}-1)Y \quad \text{et} \quad Y(X-Y).$$

4. Montrer que pour tout $x \in \mathscr{C}$, $w(x)$ est pair.
   En déduire que $W_{\mathscr{C}}(X,Y) \in K[X,Y]^{G_2}$, où l'on a noté G_2 le groupe engendré par A et $-\mathrm{Id}_2$.
5. Ecrire un programme MAPLE qui calcule des générateurs de l'anneau des invariants sous l'action d'un groupe donné. Utiliser ce programme pour calculer des générateurs pour $K[X,Y]^{G_2}$. Quels sont les problèmes de cette méthode ?

Il existe des méthodes plus performantes pour calculer l'anneau des invariants d'un groupe donné. Le livre de COX [22] présente les bases de Gröbner, et leurs applications à la théorie des invariants.

Correction des exercices

1 Correction des exercices du chapitre 1

Correction de l'exercice I.1 :

1. Soit $\chi \in \widehat{G}$. Le théorème de convolution 4.15, chap. I, nous dit que $\mathscr{F}(f*\chi) = \widehat{f}\cdot\widehat{\chi}$. On vérifie facilement que $\widehat{\chi} = |G|\delta_{\chi^{-1}}$, où $\delta_{\chi^{-1}} \in \mathbb{C}[\widehat{G}]$ est la fonction qui vaut 1 en χ^{-1} et 0 sinon. Ceci permet d'écrire que $\mathscr{F}(f*\chi) = |G|\widehat{f}(\chi^{-1})\delta_{\chi^{-1}}$. Il ne reste plus qu'à appliquer la transformée de Fourier inverse en utilisant le fait que $\mathscr{F}^{-1}(\delta_{\chi^{-1}}) = \frac{1}{G}\chi$. On obtient ainsi que $\Phi^f(\chi) = f*\chi = \widehat{f}(\chi^{-1})\chi$. Donc χ est un vecteur propre pour Φ^f, de valeur propre associée $\widehat{f}(\chi^{-1})$.
2. On a $\Phi^f(\delta_g) = f*\delta_g = \sum_{h\in G} f(g^{-1}h)\delta_h$. Si on note $\{0,\dots,n-1\}$ les éléments de $G \simeq \mathbb{Z}/n\mathbb{Z}$, on écrit donc la matrice $A = \{a_{ij}\}$ sous la forme $a_{ij} = f(i-j)$.
   De plus, avec la question précédente, on a l'expression du déterminant
$$\det(A) = \prod_{\chi\in\widehat{G}} \widehat{f}(\chi^{-1}) = \prod_{\chi\in\widehat{G}} \widehat{f}(\chi).$$
3. Il suffit de choisir $G = \mathbb{Z}/n\mathbb{Z}$, ainsi que $\forall i \in \mathbb{Z}/n\mathbb{Z}$, $f(i) = a_i$.
4. On peut calculer $\widehat{f}$ en $O(n\log(n))$ opérations avec l'algorithme FFT, et on obtient le déterminant en multipliant entre elles les entrées de $\widehat{f}$. Le programme MATLAB 1.1 réalise ceci, sur un vecteur f de taille 10 tiré au hasard.

Programme 1.1 Calcul de déterminant circulant

```
n = 10; f = rand(n,1);
prod(fft(f))
```

Correction de l'exercice I.2 :

1. On considère R_1 (resp. R_2) une rotation d'axe unitaire u_1 (resp. u_2), ainsi que v_1, w_1 (resp. v_2, w_2) une base orthonormée du plan $(u_1)^\perp$ (resp. $(u_2)^\perp$). On prend Ω l'isométrie qui envoie (u_1, v_1, w_1) sur (u_2, v_2, w_2). Si les deux rotations ont le même angle, on a $R_1 = \Omega^* R_2 \Omega$.
2. Comme $\chi(a^{-1}ba) = \chi(a)^{-1}\chi(b)\chi(a) = \chi(b)$, χ est constant sur les classes de conjugaison. Avec la question précédente, $\chi(R)$ ne dépend que de l'angle de la rotation R.
3. On utilise le fait que $\mathrm{tr}(t_\beta) = 1 + 2\cos(\beta)$. En faisant le calcul (avec MAPLE, comme le montre le programme 1.2), on trouve que $\mathrm{tr}(r_\alpha s_\alpha^{-1}) = 2\cos(\alpha) + \cos(\alpha)^2$.

Programme 1.2 Calcul de $\mathrm{tr}(t_\beta)$

```
with(linalg):
r1 := matrix(3,3,[1,0,0,0,cos(t),sin(t),0,-sin(t),cos(t)]);
r2 := matrix(3,3,[cos(t),0,-sin(t),0,1,0,sin(t),0,cos(t)]);
tr := trace(r1&*r2);
plot(tr, theta=0..pi);
```

L'étude de la fonction $\alpha \mapsto \frac{1}{2}(2\cos(\alpha)+\cos(\alpha)^2-1)$ montre immédiatement qu'elle est strictement décroissante sur $[0,\pi]$ et prend toutes les valeurs entre 1 et -1.

4. On a donc, $\forall \beta \in [0,\pi]$, $\chi(t_\beta) = \chi(r_\alpha)\chi(s_\alpha)^{-1} = 1$. Comme une rotation d'angle $-\beta$ et d'axe v peut être vue comme une rotation d'angle β et d'axe $-v$, l'égalité précédente est encore vraie pour $\beta \in [-\pi,0]$. Donc $\chi = 1$.

Correction de l'exercice I.3 : Il suffit d'utiliser la relation $\delta_0 = \frac{1}{|G|}\sum_{\chi\in\widehat{G}}\chi$. On écrit alors que

$$\begin{aligned} N(h) &= \sum_{(x_1,\ldots,x_n)\in G^n} \delta_0(\varphi(x_1,\ldots,x_n)-h) \\ &= \sum_{(x_1,\ldots,x_n)\in G^n}\sum_{\chi\in\widehat{G}} \chi(\varphi(x_1,\ldots,x_n)-h). \end{aligned}$$

On trouve bien l'égalité demandée en remarquant que

$$\chi(\varphi(x_1,\ldots,x_n)-h) = \chi(\varphi(x_1,\ldots,x_n))\overline{\chi(h)}.$$

Correction de l'exercice I.4 : Dans la suite, on note $n = |G|$.

1. On a $\|f_A\|_2^2 = \frac{1}{G}\sum_{x\in A} 1 = \frac{|A|}{|G|}$. De même, $\widehat{f_A}(\chi_0) = \sum_{x\in G} f_A(x) = |A|$.
2. En utilisant la formule de Plancherel, proposition 4.7, chap. I, on obtient l'égalité $\|\widehat{f_A}\|_2^2 = n\|f_A\|_2^2 = |A|$. On peut majorer $\|\widehat{f_A}\|_2^2$ de la façon suivante :

$$n\|\widehat{f_A}\|_2^2 \leqslant |\widehat{f_A}(\chi_0)|^2 + (n-1)\Phi(A)^2 = |A|^2 + (n-1)\Phi(A)^2.$$

On obtient ainsi $\Phi(A)^2 \geqslant \frac{1}{n-1}|A|(n-|A|) \geqslant |A|/2$. L'autre inégalité est triviale, puisque $|\widehat{f_A}(\chi)| \leqslant \sum_{x\in A}|\chi(x)| \leqslant |A|$.

3. On note $B = G\backslash A$. On a $f_B = 1 - f_A$, donc $\widehat{f_B} = \widehat{1} - \widehat{f_A} = |G|\delta_{\chi_0} - \widehat{f_A}$. On a donc, pour $\chi \neq \chi_0$, $|\widehat{f_B}(\chi)| = |\widehat{f_A}(\chi)|$, donc $\Phi(A) = \Phi(B)$. Au final, on a les inégalités $|G| - |A| \geqslant \Phi(A) \geqslant \sqrt{\frac{|G|-|A|}{2}}$
4. Si $\chi \neq \chi_0$, alors $\widehat{f_{\alpha(A)}}(\chi) = \sum_{x\in A}\chi\circ\alpha(x)$. Or, l'application $\chi \mapsto \chi\circ\alpha$ est une permutation des caractères non triviaux. Donc $\Phi(A) = \Phi(\alpha(A))$.

Correction de l'exercice I.5 : Dans la suite, on note $n = |G|$.

1. En remplaçant A_1 par $A_1\backslash a = \{x - a \setminus x \in A_1\}$, on se ramène à l'équation homogène $x_1 + \cdots + x_n = 0$, avec $x_1 \in A_1\backslash a$, qui possède le même nombre de solutions que l'équation de départ.

En appliquant le résultat de l'exercice I.3, avec la fonction $\varphi(x_1,\dots,x_k)=x_1+\cdots x_k$ et $h=0$, on obtient

$$N=\frac{1}{|G|}\sum_{\chi\in\widehat{G}}\sum_{x_i\in A_i}\chi(x_1+\cdots+x_k). \tag{1.1}$$

La dernière somme peut s'écrire

$$\sum_{x_i\in A_i}\chi(x_1+\cdots+x_k)=\prod_{i=1}^{k}\sum_{x_i\in A_i}\chi(x_i)=\prod_{i=1}^{k}\widehat{f_{A_i}}(\chi).$$

Le terme de la somme (1.1) correspondant à χ_0 donne $\frac{|A_1|\cdots|A_k|}{|G|}$. Les autres termes donnent R.

2. Pour $k=3$, on a

$$R=\frac{1}{|G|}\sum_{\chi\neq\chi_0}\widehat{f_{A_1}}(\chi)\cdot\widehat{f_{A_2}}(\chi)\cdot\widehat{f_{A_3}}(\chi).$$

D'où l'inégalité

$$|R|\leqslant\frac{\Phi(A_3)}{|G|}\sum_{\chi\in\widehat{G}}|\widehat{f_{A_1}}(\chi)|\cdot|\widehat{f_{A_2}}(\chi)|.$$

En utilisant l'inégalité de Cauchy-Schwartz, on obtient

$$\begin{aligned}\sum_{\chi\in\widehat{G}}|\widehat{f_{A_1}}(\chi)|\cdot|\widehat{f_{A_2}}(\chi)|&\leqslant\left\{\left(\sum_{\chi\in\widehat{G}}|\widehat{f_{A_1}}(\chi)|^2\right)\left(\sum_{\chi\in\widehat{G}}|\widehat{f_{A_2}}(\chi)|^2\right)\right\}^{1/2}\\&\leqslant\left(n\|\widehat{f_{A_1}}\|^2 n\|\widehat{f_{A_2}}\|^2\right)^{1/2}\\&=n^2\|f_{A_1}\|\|f_{A_2}\|=n\sqrt{|A_1||A_2|}.\end{aligned}$$

La translation par a étant un automorphisme de G, en utilisant le résultat de l'exercice I.4, question 4, l'inégalité trouvée est invariante par translation par a.

3. Grâce à la question 2, l'inégalité proposée est une réécriture de $R<\frac{|A_1||A_2||A_3|}{|G|}$. En utilisant l'égalité prouvée à la question 1, on obtient $N>0$.

Correction de l'exercice I.6 :

1. On vérifie l'associativité de l'opération sur la formule donnée. L'élément neutre est $(1,1,\chi_0)$, et $(\lambda,x,\chi)^{-1}=(\lambda^{-1}\chi(x),x^{-1},\chi^{-1})$.
2. Il faut montrer les différents axiomes d'une action de groupe (voir [58]), en particulier, avec un léger abus de notation,

$$\begin{aligned}(\lambda,x,\chi)\cdot[(\mu,y,\tau)\cdot f](z)&=(\lambda,x,\chi)\cdot[\mu\tau(z)f(yz)]\\&=\lambda\mu\tau(xz)\chi(z)f(xyz)\\&=[(\lambda,x,\chi)(\mu,y,\tau)]\cdot f(z).\end{aligned}$$

L'action de $\mathscr{H}(G)$ sur $\mathbb{C}[G]$ est linéaire (c'est-à-dire que l'action commute avec les lois de l'espace vectoriel). C'est ce que l'on appelle une représentation linéaire, (se reporter au chapitre VII).

3. On a
$$(\lambda,x,\chi)\cdot f = D_\lambda \circ M_\chi \circ T_x \circ f.$$
On peut d'ailleurs, avec ce formalisme, redémontrer facilement le résultat de la question précédente. Il suffit de remarquer que $T_x M_\tau = D_{\tau(x)} M_\tau T_x$, ce qui permet d'écrire
$$(D_\lambda M_\chi T_x)(D_\mu M_\tau T_y) = D_\lambda D_\mu M_\chi (T_x M_\tau) T_y = (D_\lambda D_\mu D_{\tau(x)})(M_\chi M_\tau)(T_x T_y).$$
On peut ensuite simplifier les termes en utilisant le fait que $\lambda \mapsto D_\lambda$, $\chi \mapsto M_\chi$ et $x \mapsto T_x$ sont des morphismes de groupes.
4. Pour simplifier les notations, on introduit, pour $(\lambda,\chi,x) \in \mathscr{H}(\widehat{G})$, les opérateurs de dilatation, translation, et modulation par
$$\widetilde{D}_\lambda(\varphi)(\tau) = \lambda\varphi(\tau), \qquad \widetilde{T}_\chi(\varphi)(\tau) = \varphi(\chi\tau), \qquad \widetilde{M}_x(\varphi)(\tau) = \tau(x)\varphi(\tau),$$
où $\tau \in \widehat{G}$ et $\varphi \in \mathbb{C}[\widehat{G}]$. On montre alors facilement les relations suivantes :
$$\mathscr{F}(T_x f) = \widetilde{M}_{x^{-1}}\mathscr{F}(f), \qquad \mathscr{F}(M_\chi) = \widetilde{T}_\chi \mathscr{F}(f), \qquad \mathscr{F}(D_\lambda) = \widetilde{D}_\lambda \mathscr{F}(f).$$
Par analogie avec l'action de $\mathscr{H}(G)$ sur $\mathbb{C}[G]$, on définit une action de $\mathscr{H}(\widehat{G})$ (dont il reste encore à définir la multiplication !) sur $\mathbb{C}[\widehat{G}]$ par
$$(\lambda,\chi,x)\cdot f = \widetilde{D}_\lambda \widetilde{M}_x \widetilde{T}_\chi f.$$
Pour obtenir la loi de multiplication qui nous convienne, on se contente de composer l'action de groupe avec elle même :
$$(\widetilde{D}_\lambda \widetilde{M}_x \widetilde{T}_\chi)(\widetilde{D}_\mu \widetilde{M}_y \widetilde{T}_\tau) = (\widetilde{D}_\lambda \widetilde{D}_\mu \widetilde{D}_{\chi(y)})(\widetilde{M}_x \widetilde{M}_y)(\widetilde{T}_\chi \widetilde{T}_\tau).$$
Une fois de plus, c'est le calcul de commutation translation/dilation qui permet d'arriver au résultat $\widetilde{T}_\chi \widetilde{M}_y = \widetilde{D}_{\chi(y)} \widetilde{M}_y \widetilde{T}_\chi$. Au final, on obtient la loi suivante sur $\mathscr{H}(\widehat{G})$:
$$(\lambda,\chi,x)\cdot(\mu,\tau,y) = (\lambda\mu\chi(y), \chi\tau, xy).$$
En quelque sorte, cette loi a été « construite » pour que l'on obtienne bien une action de groupe.
5. Le fait que α soit un morphisme résulte du calcul suivant
$$\begin{aligned}\alpha((\lambda\mu\tau(x),xy,\chi\tau)) &= (\lambda\mu\tau(x)(\chi\tau)(x^{-1}y^{-1}, \chi\tau, x^{-1}y^{-1}) \\ &= \big((\lambda\chi^{-1}(x))(\mu\tau^{-1}(y))\chi^{-1}(y), \chi\tau, x^{-1}y^{-1}\big) \\ &= (\lambda\chi^{-1}(x), \chi, x^{-1})\cdot(\mu\tau^{-1}(y), \tau, y^{-1}).\end{aligned}$$
Avec la question précédente, on a
$$\mathscr{F}(D_\lambda M_\chi T_x f) = \widetilde{D}_\lambda \widetilde{T}_\chi \widetilde{M}_{x^{-1}} \mathscr{F}(f) = D_{\lambda\chi(x^{-1})} M_{x^{-1}} T_\chi \mathscr{F}(f).$$
En traduisant cette égalité en terme d'action de groupe, on obtient le résultat souhaité.

6. On note, pour $g \in \mathscr{H}(G)$, $\rho(g) \in \mathscr{L}(\mathbb{C}[G])$ l'application linéaire $f \mapsto g \cdot f$. De même, on note, pour $\widetilde{g} \in \mathscr{H}(\widehat{G})$, $\widetilde{\rho}(\widetilde{g}) \in \mathscr{L}(\mathbb{C}[\widehat{G}])$ l'application linéaire $\varphi \mapsto \widetilde{g} \cdot \varphi$. ρ et $\widetilde{\rho}$ sont des représentations linéaires des deux groupes finis $\mathscr{H}(G)$ et $\mathscr{H}(\widehat{G})$. L'hypothèse faite sur Φ signifie que $\Phi \circ \rho = \rho \circ \Phi$. En utilisant le lemme de Schur 2.5, chap. VII, on a donc que Φ est une homothétie. Si on suppose maintenant que Φ commute avec les actions de $\mathscr{H}(G)$ et $\mathscr{H}(\widehat{G})$, alors Φ entrelace les deux représentations. En utilisant le corollaire 2.8, chap. VII, comme la dimension de l'espace vectoriel des morphismes d'entrelacement est 1, Φ s'écrit $r\Psi$, où Ψ est un isomorphisme (non canonique) entre $\mathbb{C}[G]$ et $\mathbb{C}[\widehat{G}]$ fixé (ces deux espaces ont bien sûr même dimension) et $r \in \mathbb{C}$. Comme $G \simeq \widehat{G}$ (non canonique), il est même facile de construire un isomorphisme d'algèbres entre $\mathbb{C}[G]$ et $\mathbb{C}[\widehat{G}]$ (toujours non canonique).

Correction de l'exercice I.7 :

1. On a $\langle \tau_n(f), \tau_p(f) \rangle = f * \widetilde{f}(n-p)$, où on a noté $\widetilde{f}(x) = \overline{f}(-x)$. La famille $\{\tau_n(f)\}$ est donc orthonormée si et seulement si $f * \widetilde{f}(n) = \delta_0(0)$. Ceci s'écrit, avec les distributions,
$$(f * \widetilde{f}) \cdot \Pi_1 = \delta_0, \tag{1.2}$$
où δ_0 est le Dirac en 0. On sait que la transformée de Fourier de $f * \widetilde{f}$ est $|\widehat{f}|^2$. En prenant la transformée de Fourier de l'équation (1.2), et en utilisant la propriété de convolution de la transformée de Fourier (voir [62]) on trouve donc $\frac{1}{2\pi}|\widehat{f}|^2 * \widehat{\Pi_1} = 1$. En utilisant le calcul de $\widehat{\Pi_1}$ fait à l'exercice II.9, on trouve le résultat voulu.
2. On a $|\widehat{\varphi}| \leqslant A|\widehat{f}|$, donc $\varphi \in L^2(\mathbb{R})$.
La fonction φ vérifie bien $\sum_{k\in\mathbb{Z}} |\widehat{\varphi}(\omega + 2k\pi)|^2 = 1$, donc la famille $\{\tau_n(\varphi)\}$ est orthonormée.

Correction de l'exercice I.8 :

1. Le fait que b soit orthonormé est équivalent au fait que $\psi_b = \delta_e$, où e est l'élément neutre de G. En prenant les transformées de Fourier des deux membres, on trouve $\widehat{\psi_b} = \widehat{\delta_e} = \mathbf{1}$, où on a noté $\mathbf{1}$ la fonction constante égale à 1. Il ne reste plus qu'à remarquer que $\widehat{\psi_b} = |G| \langle \mathscr{U}_\chi b, b \rangle$
2. Il s'agit de montrer trois choses :
   (i) l'orthogonalité de $\mathscr{U}_{\chi_1}$ et $\mathscr{U}_{\chi_2}$.
   (ii) l'idempotence de $\mathscr{U}_{\chi_1}$.
   (iii) que $\mathscr{U}_{\chi_1}$ est auto-adjoint.
   Montrons (i) :
$$\begin{aligned}
\mathscr{U}_{\chi_1}\mathscr{U}_{\chi_2} &= \frac{1}{|G|^2} \sum_{(U,V)\in G^2} UV\chi_1(U)\chi_2(V) \\
&= \frac{1}{|G|^2} \sum_{(U,R)\in G^2} R\chi_1(U)\overline{\chi_2(U)}\chi_2(R) \\
&= \frac{1}{|G|^2} \sum_{U\in G} \chi_1(U)\overline{\chi_2(U)} \sum_{R\in G} R\chi_2(R) = \delta_{\chi_1}^{\chi_2}\mathscr{U}_{\chi_2}.
\end{aligned}$$
   Pour la deuxième égalité, on a utilisé le changement de variable $R = UV$, et pour la dernière égalité, on a utilisé l'orthogonalité des caractères χ_1 et χ_2.

Pour montrer (ii), le calcul est identique, il suffit de prendre $\mathscr{U}_{\chi_1} = \mathscr{U}_{\chi_2}$.
Le fait que $\mathscr{U}_{\chi_1}$ soit auto-adjoint est immédiat :

$$(\mathscr{U}_{\chi_1})^* = \sum_{U\in G} U^*\overline{\chi_1(U)} = \sum_{U\in G} U^{-1}\chi_1(U^{-1}) = \mathscr{U}_{\chi_1}.$$

Pour la dernière égalité, on a simplement utilisé le fait que $U \mapsto U^{-1}$ était une permutation sur l'indice de sommation.

3. Pour montrer l'orthogonalité de $\widetilde{b}$, il faut mener le calcul suivant :

$$\begin{aligned}\left\langle \widetilde{b}, \mathscr{U}_\chi(\widetilde{b})\right\rangle &= \left\langle \sum_{\tau\in\widehat{G}} \mathscr{U}_\tau b\frac{1}{\sqrt{\Phi(\tau)}}, \sum_{\tau\in\widehat{G}} \mathscr{U}_\tau\mathscr{U}_\chi b\frac{1}{\sqrt{\Phi(\tau)}}\right\rangle \\ &= \left\langle \mathscr{U}_\chi b\frac{1}{\sqrt{\Phi(\chi)}}, \mathscr{U}_\chi\mathscr{U}_\chi b\frac{1}{\sqrt{\Phi(\chi)}}\right\rangle \\ &= \frac{1}{\Phi(\chi)}\left\langle \mathscr{U}_\chi b, \mathscr{U}_\chi b\right\rangle = \frac{1}{\Phi(\chi)}\left\langle b, \mathscr{U}_\chi b\right\rangle = 1.\end{aligned}$$

Pour la deuxième égalité, on a développé les sommes, et on a utilisé les relations d'orthogonalité entre les $\mathscr{U}_\tau$ démontrées à la question précédente. Pour la dernière égalité, on a utilisé le fait que $\mathscr{U}_\chi$ est auto-adjoint.

4. L'exercice I.7 se place dans le cas continu et utilise le groupe $\mathbb{R}$ agissant de façon unitaire sur $L^2(\mathbb{R})$ par translation. Il s'agit aussi d'une méthode d'orthogonalisation dans le domaine de Fourier.

Correction de l'exercice I.9 :

1. On a $\widehat{P}(\chi_0) = 1$ et, pour $\chi \neq \chi_0$, $\widehat{U}(\chi) = 0$. En utilisant la formule de Plancherel, proposition 4.7, chap. I, on obtient

$$\|P-U\|_2^2 = \frac{1}{|G|}\|\widehat{P}-\widehat{U}\|_2^2 = \frac{1}{|G|^2}\sum_{\chi\neq\chi_0}|\widehat{P}(\chi)|^2.$$

2. Il suffit de remarquer que $\left|P(g) - \frac{1}{|G|}\right| \leqslant |G|\|P-U\|_2^2$.

Correction de l'exercice I.10 :

1. On a la relation

$$\mathbb{P}(X_{k+1} = i) = \sum_{j=0}^{n-1}\mathbb{P}(X_k = j)\mathbb{P}(X_{k+1} = i|X_k = j),$$

ce qui signifie exactement $p^{(k+1)} = Pp^{(k)}$.
En itérant cette relation, on obtient $p^{(k)} = P^k p^{(0)}$.

2. Dans ce cas particulier, on a $(Px)[i] = (1-p)x[i-1] + px[i+1]$. On reconnaît une formule de convolution, et on a $Px = \nu * x$. Il est aussi possible de considérer des variables aléatoires Y_i, indépendantes, de vecteur densité ν: On a alors $X_k = \sum_{i=1}^k Y_i$. En utilisant le fait que $P_{Y_i+Y_j} = P_{Y_i} * P_{Y_j} = \nu * \nu$, on retrouve $p^{(k)} = \nu * \cdots * \nu * p^{(0)}$ (k produits).

3. Grâce au théorème de convolution 4.14, chap. I, on obtient $\widehat{p^{(k)}} = \widehat{v} \cdot \ldots \cdot \widehat{v} \cdot \widehat{p^{(0)}}$. On peut calculer explicitement $\widehat{v}[i] = pe^{\frac{2i\pi}{n}} + (1-p)e^{-\frac{2i\pi}{n}}$. Comme $0 < p < 1$ et que k est impair, on a, pour $i \neq 0$, $|\widehat{v}[i]| < 1$, et donc $\widehat{p^{(k)}} \to \delta_{\chi_0}$ lorsque $k \to +\infty$. En prenant la transformée inverse de cette relation, on obtient $p^{(k)} \to u$ lorsque $k \to +\infty$. Si k est pair, on a $\widehat{v}[n/2] = -1$ et la probabilité ne converge pas. Intuitivement, on voit que si on écrit $\{0,\ldots,n-1\} = P \cup I$ (partition entre pairs et impairs), alors $p^{(2s)}$ va être porté par P, et $p^{(2s+1)}$ par I, ce qui exclut toute convergence (les deux ensembles ne se « mélangent » pas).
4. On a $p^{(k+1)}[i] = \sum_{j=0}^{n-1} v_{i-j} p^{(k)}[i] = p^{(k)} * v[i]$. En écrivant cette équation dans le domaine de Fourier, et en l'itérant, on voit qu'il peut se présenter plusieurs situations :
   - Si $\exists i$, $|\widehat{c}[i]| > 1$, alors $p^{(k)}$ va exploser. Ceci ne peut pas arriver pour une distribution de probabilité, puisque $|\widehat{c}[i]| \leqslant 1$.
   - Si $\exists i$, $|\widehat{c}[i]| = 1$ et $\widehat{c}[i] \neq 1$, alors $p^{(k)}$ ne va pas converger.
   - Sinon, $p^{(k)} \to p^{\infty}$ lorsque $k \to +\infty$, où on a défini p^{∞} par $\widehat{p^{\infty}}[i] = \widehat{p^{(0)}}[i]$ si $\widehat{v}[i] = 1$, et $\widehat{p^{\infty}}[i] = 0$ sinon.

   On pourra comparer ceci avec l'étude des polygones (paragraphe 3.1, chap. IV) qui est en tout point identique (sauf que les polygones peuvent exploser !). Le code MATLAB 1.3 permet, à partir d'un vecteur de probabilité initiale `p0`, et du vecteur de transition `v`, de calculer la probabilité `pk` à la $k^{\text{ième}}$ itération.

Programme 1.3 Calcul de $p^{(k)}$

```
pk = p0;
for i=1:k
  pk = real( ifft( fft(pk).*fft(v) ) );
end
```

Correction de l'exercice I.11 : Dans la suite on identifie $\widehat{\mathbb{Z}/n\mathbb{Z}}$ à $\mathbb{Z}/n\mathbb{Z}$ et on note $\widehat{f}(k)$ pour $\widehat{f}(\chi_k)$.

1. La méthode est exactement celle utilisée pour démontrer la borne sur la distance minimale des codes BCH, proposition 3.22, chap. VI.
   On suppose que $\mathrm{Supp}(f) = \{a_1,\ldots,a_p\}$. Pour l'instant, on suppose simplement que $\widehat{f}(0) = \cdots = \widehat{f}(p-1) = 0$. En notant $\omega = e^{\frac{2i\pi}{n}}$, on obtient le système

$$\begin{pmatrix} 1 & 1 & & 1 \\ \omega^{a_1} & \omega^{a_2} & & \omega^{a_p} \\ \vdots & \vdots & & \vdots \\ \omega^{a_1(p-1)} & \omega^{a_2(p-1)} & \ldots & \omega^{a_p(p-1)} \end{pmatrix} \begin{pmatrix} f(a_1) \\ f(a_2) \\ \vdots \\ f(a_p) \end{pmatrix} = 0.$$

   La matrice du système est de Vandermonde, elle est inversible, ce qui est absurde car les $f(a_i)$ ne sont pas tous nuls. Dans le cas général, si on suppose que p entrées consécutives de $\widehat{f}$ sont nulles, on se ramène au cas précédent en effectuant une translation sur $\widehat{f}$ (ce qui revient à une modulation sur f et ne change pas le support). Il est maintenant simple de voir que ceci implique le principe d'incertitude. Supposons d'abord que $p|N$. On partitionne $\{0,\ldots,n-1\}$ en n/p blocs de taille p. D'après ce que nous venons de montrer, sur chacun de ces blocs, $\widehat{f}$ ne peut être

nulle. Ainsi, chaque bloc contient au moins un k tel que $\widehat{f}(k) \neq 0$ et $|\mathrm{Supp}(\widehat{f})| \geqslant n/p$.

Si p ne divise pas n, on note $d = \lceil n/p \rceil$. Il est impossible de distribuer moins de d éléments parmi n places sur un cercle sans laisser deux éléments avec un trou de p places entre eux.

En conséquence, on a $|\mathrm{Supp}(\widehat{f})| \geqslant d$ et $|\mathrm{Supp}(f)| \times |\mathrm{Supp}(\widehat{f})| \geqslant dp \geqslant n$.

2. La première inégalité est triviale. Pour la deuxième, il suffit d'utiliser la formule d'inversion de Fourier 4.4, chap. I, et d'écrire

$$|f(x)| \leqslant \frac{1}{|G|} \sum_{\chi \in \widehat{G}} |\widehat{f}(\chi)||\overline{\chi}(x)|,$$

ce qui mène à l'inégalité voulue en prenant le maximum de toutes ces inégalités, pour $x \in G$.

L'inégalité précédente s'écrit $M^2 \leqslant \left\langle \widehat{f}, g \right\rangle$, où l'on a noté g la fonction indicatrice de $\mathrm{Supp}(f)$. Avec Cauchy-Schwartz, on obtient $M^2 \leqslant \|f\|^2 \|g\|^2$, ce qui est la première inégalité demandée. La deuxième inégalité s'obtient en utilisant simplement la formule de Plancherel 4.7, chap. I. En combinant la première égalité de la question 2, et l'égalité que l'on a démontrée, on trouve l'inégalité finale.

3. La transformée de f_H est étudiée à la proposition 4.7, chap. VI.
On a donc $|\mathrm{Supp}(f_H)| = |H|$ et $|\mathrm{Supp}(f_{H^\sharp})| = |H^\sharp| = |G|/|H|$. La fonction f_H atteint bien la borne établie.

2 Correction des exercices du chapitre 2

Correction de l'exercice II.1 :

1. Chacun des résidus $[(-1)^b b]_p$ est clairement pair. S'il y a un doublon parmi ces résidus, alors $(-1)^{ra} ra = (-1)^{ra'} ra' \mod p$, donc $a = \pm a' \mod p$, et au final on a $a + a' = 0 \mod p$. Ceci est impossible car $0 < a + a' < 2p$ et $a + a' \neq p$ (car $a + a'$ est pair).
2. En faisant le produit des éléments de B on trouve $\prod_{b \in B} b = r^{\frac{p-1}{2}} \prod_{a \in A} a \mod p$, puisque $\mathrm{Card}(A) = \frac{p-1}{2}$. De même, en faisant le produit des éléments de l'ensemble $\{[(-1)^b b]_p\}$, on trouve $\prod_{a \in A} a = (-1)^{\sum_{b \in B} b} \prod_{b \in B} b \mod p$. Avec le critère d'Euler, lemme 1.1, chap. II, on obtient $\left(\frac{r}{p}\right) = r^{\frac{p-1}{2}}$, ce qui conduit à l'égalité souhaitée.
3. On remarque que $\left\lfloor \frac{ra}{p} \right\rfloor$ est le quotient de la division euclidienne de ra par p, et $[ra]_p$ le reste. On a donc $ra = \left\lfloor \frac{ra}{p} \right\rfloor p + [ra]_b$. En sommant sur tous les $a \in A$, on obtient l'égalité demandée. Comme tous les a sont pairs et que $p = 1 \mod 2$, on a $\sum_{b \in B} b = \sum_{a \in A} \left\lfloor \frac{ra}{p} \right\rfloor \mod 2$. On traduit cette inégalité immédiatement en termes de puissance de -1.
4. S'il y avait un point (x, y) sur $]AB[$, avec $x \leqslant r$ et $y \leqslant p$, alors on aurait $py = rx$, ce qui n'est pas, puisque p et r sont des premiers distincts. Sur chaque droite horizontale $x = a$, avec a pair, c'est-à-dire pour $a \in A$, le nombre de points situés en dessous de $[AB]$ est $\left\lfloor \frac{ra}{p} \right\rfloor$. En sommant pour $a \in A$, on compte tous les points d'abscisses paires en dessous de $[AB]$.

5. On note $C_p(F)$ le nombre de points d'abscisses paires dans une figure F, $C_i(F)$ le nombre de points d'abscisses impaires et $C(F)$ le nombre total de points.
On note n_1 le nombre de points d'abscisse a situés au-dessous de $[AB]$, et n_2 le nombre de points situés au-dessus. On a $n_1+n_2=r-1$ qui est pair, donc $n_1=n_2 \mod 2$. Ceci signifie donc que $C_p(HKBD)=C_p(MHB) \mod 2$.
Par symétrie par rapport à H, n_2 est égal au nombre de points d'abscisse $p-a$ situés en dessous de $[AB]$. Ceci signifie donc que $C_p(MHB)=C_i(ALH) \mod 2$.
On a $C_p(ABD)=C_p(AKH)+C_p(HKDB) \mod 2$ (décomposition de ABD en deux).
Donc $C_p(ABD)=C_p(AKH)+C_i(ALH) \mod 2$. Par symétrie par rapport à $[AB]$, on a $C_i(ALH)=C_i(AKH)$.
Au final, on a $C_p(ABD)=C_p(AKH)+C_i(AKH)=C(AKH)$.
6. La question précédente nous dit que $\left(\frac{r}{p}\right)=(-1)^{C(AKH)}$. En échangeant les rôles de r et p, on a aussi $\left(\frac{p}{r}\right)=(-1)^{C(AHL)}$. Au final, on obtient

$$\left(\frac{r}{p}\right)\left(\frac{p}{r}\right)=(-1)^{C(AKH)+C(AHL)}=(-1)^{C(AKHL)}=(-1)^{\frac{(p-1)(r-1)}{4}}.$$

Correction de l'exercice II.2 :

1. Cette question traduit la structure des sous-groupes de $\mathbb{F}_q^*$, qui est un groupe cyclique, voir par exemple [24]. On montre que le seul groupe d'indice k de $\mathbb{F}_q^*$ est formé des racines du polynôme $X^{\frac{q-1}{k}}-1$, c'est donc H_k.
2. On a

$$\sum_{i=0}^{k-1} G(\chi_i,\psi)=\sum_{x\in\mathbb{F}_q^*}\psi(x)\sum_{i=0}^{k-1}\chi_i(x).$$

Par la proposition d'orthogonalité 1.9, chap. II, appliquée au caractères multiplicatifs du groupe $\mathbb{F}_q^*/H_k$, $\sum_{i=0}^{k-1}\chi_i(x)$ vaut 0 si $x\notin H_k$, et k si $x\in H_k$. On obtient ainsi

$$\sum_{i=0}^{k-1} G(\chi_i,\psi)=k\sum_{x\in H_k}\psi(x)=k\widehat{f_{H_k}}(\psi).$$

Soit ψ un caractère additif non trivial. Grâce à la proposition 1.17, chap. II, on a la majoration

$$\left|\widehat{f_{H_k}}(\psi)\right|\leqslant\frac{1}{k}\sum_{i=0}^{k-1}|G(\chi_i,\psi)|=\frac{1}{k}(1+(k-1)\sqrt{q})<\sqrt{q}.$$

3. On note $A_3=H_k$, d'où $|A_3|=\frac{q-1}{k}$. Comme $\mathbb{F}_q$ possède k racines de l'unité, l'équation $z^k=u$ possède k racines, et donc $N=kN'$. En utilisant le résultat de l'exercice I.5, question 2, on a

$$\left|N'-\frac{|A_1||A_2||H_k|}{q}\right|<\Phi(H_k)\sqrt{|A_1||A_2|},$$

ce qui donne bien l'inégalité voulue.
4. Sous l'hypothèse $q\geqslant k^2l_1l_2+4$, on peut majorer le membre de droite de l'équation (4.4), chap. II :

$$k\sqrt{|A_1||A_2|q}=k|A_1||A_2|\sqrt{\frac{l_1l_2q}{(q-1)^2}}\leqslant|A_1||A_2|\sqrt{\frac{(q-4)q}{(q-1)^2}}.$$

On montre ensuite, par une étude de fonction, que, pour $q \geqslant 0$, $\frac{(q-4)q}{(q-1)^2} \leqslant \frac{(q-1)^2}{q^2}$, ce qui permet d'avoir l'inégalité

$$\left| N - \frac{|A_1||A_2|(q-1)}{q} \right| < \frac{|A_1||A_2|(q-1)}{q},$$

et montre que $N > 0$.
Dans le cas où k ne divise pas $k-1$, on note $d = \text{pgcd}(q-1,k)$. Soit $A = \{x^k\}$ et $B = \{x^d\}$ Comme $d|k$, $k = \lambda d$ pour un certain λ, d'où $x^k = (x^\lambda)^d$ et $A \subset B$. Réciproquement, avec le théorème de Bezout, $\exists (u,v)$ tels que $d = u(q-1) + vk$. On a donc $x^d = (x^{q-1})^u (x^v)^k = (x^v)^k$, donc $B \subset A$. Donc si $q \geqslant k^2 l_1 l_2 + 4$, on a $q \geqslant d^2 l_1 l_2 + 4$, ce qui permet d'appliquer le raisonnement précédent au groupe H_d et montre que l'équation considérée a encore une solution.

5. En considérant $A_1 = A_2 = H_d$, où $d = \text{pgcd}(k, q-1)$, on a $|A_i| = \frac{q-1}{d} \geqslant \frac{q-1}{k}$. Ceci implique $l_i \leqslant k$ donc $k^2 l_1 l_2 + 4 \leqslant k^4 + 4$.

Correction de l'exercice II.3 : Un exemple d'interaction d'ordre 2 :

$$\alpha_{ab} = \frac{1}{4}(\alpha_{+++} + \alpha_{++-} + \alpha_{--+} + \alpha_{---}) - \frac{1}{4}(\alpha_{+-+} + \alpha_{-++} + \alpha_{+--} + \alpha_{-+-}).$$

L'interaction d'ordre 3 :

$$\alpha_{abc} = \frac{1}{4}(\alpha_{+++} + \alpha_{+--} + \alpha_{-+-} + \alpha_{+--}) - \frac{1}{4}(\alpha_{++-} + \alpha_{+-+} + \alpha_{-++} + \alpha_{---}).$$

En réordonnant les interactions (pour les mettre dans l'ordre « de Yates »), on obtient une écriture matricielle

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} \alpha_{+++} \\ \alpha_{-++} \\ \alpha_{+-+} \\ \alpha_{--+} \\ \alpha_{++-} \\ \alpha_{-+-} \\ \alpha_{+--} \\ \alpha_{---} \end{pmatrix} = \begin{pmatrix} 8\mu \\ 4\mu_a \\ 4\mu_b \\ 4\mu_{ab} \\ 4\mu_c \\ 4\mu_{ac} \\ 4\mu_{bc} \\ 4\mu_{abc} \end{pmatrix},$$

où, de façon plus compacte $W_8 \widetilde{\alpha} = \widetilde{\mu}$, avec des conventions évidentes (W_8 est la matrice de Walsh, équation (2.2), chap. II). La multiplication peut être effectuée de façon rapide par l'algorithme FWT.

Correction de l'exercice II.4 :

1. On vérifie que $h \mapsto 2^{\frac{j}{2}} h(2^j \cdot - k)$ est une isométrie de $L^2([0,1])$, donc en particulier $\|\psi_{j,k}\|_2^2 = \|\psi\|_2^2 = 1$.
Si $k_1 \neq k_2$, ψ_{j,k_1} et ψ_{j,k_2} ont des supports disjoints donc $\langle \psi_{j,k_1}, \psi_{j,k_2} \rangle = 0$. Si $j_1 < j_2$, alors ψ_{j_1,k_1} est constante sur le support de ψ_{j_2,k_2}, et comme $\int \psi_{j_2,k_2} = 0$, on a encore $\langle \psi_{j_1,k_1}, \psi_{j_2,k_2} \rangle = 0$.
Comme $\dim(E_j) = 2^j$, la famille $\{\psi_n\}_{n=0}^{2^j-1}$ est une base orthonormée de E_j.
2. Comme f est continue sur $[0,1]$ qui est compact, il existe un module de continuité uniforme C tel que $|f(x) - f(y)| \leqslant C(|x-y|)$, avec $C(t) \to 0$ lorsque $t \to 0$. Pour

conclure à la convergence uniforme de f_j, il suffit de remarquer qu'il existe $x_k \in I_k$ tel que $f_j(I_k) = f(x_k)$ (théorème des valeurs intermédiaires). Pour $x \in I_k$, on a donc

$$|f_j(x) - f(x)| \leqslant |f_j(x) - f_j(x_k)| + |f(x_k) - f(x)| \leqslant C(|x - x_k|) \xrightarrow[j\to+\infty]{} 0.$$

Il y a encore un peu de travail à faire pour en déduire la convergence de $\widetilde{f_n}$. Il faut contrôler le terme complémentaire

$$R_j(x) \overset{\text{déf.}}{=} \sum_{k=0}^{2^j-1} |\langle f, \psi_{j,k}\rangle|\,|\psi_{j,k}(x)|.$$

Il faut en fait majorer finement les produits scalaires.
On note $A_1 \overset{\text{déf.}}{=} [k2^{-j}, (k+1/2)2^{-j}[$ et $A_2 \overset{\text{déf.}}{=} [(k+1/2)2^{-j}, (k+1)2^{-j}[$. On a

$$|\langle f, \psi_{j,k}\rangle| = 2^{j/2}\left|\int_{A_1} f - \int_{A_2} f\right| = 2^{j/2}2^{-j+1}|f(x_1) - f(x_2)| \leqslant 22^{-j/2}C(2^{-j}).$$

où $x_1 \in A_1$ et $x_2 \in A_2$. Au final, si on prend $x \in I_k$, on obtient la majoration

$$R_j(x) = |\langle f, \psi_{j,k}\rangle|\,2^{j/2} \leqslant 2C(2^{-j}) \xrightarrow[j\to+\infty]{} 0,$$

ce qui montre la convergence uniforme de $|\widetilde{f_n}| \leqslant |f_J| + |R_J|$ pour un certain J.
On a $\|\widetilde{f_n} - f\|_2 \leqslant \|\widetilde{f_n} - f\|_\infty$, en conséquence, la suite $\widetilde{f_n}$ converge aussi en norme L^2, et $\{\psi_n\}$ forme une base hilbertienne.

3. Les fonctions $\varphi_{j,k}$ sont les fonctions indicatrices des intervalles I_k multipliées par $2^{j/2}$. Cette famille forme donc une base de F_J. Leurs supports étant disjoints elles sont deux à deux orthogonales.
La transformation $h \mapsto 2^{j/2}h(2^j \cdot -k)$ conservant l'énergie, cette base est orthonormée.
Si on pose $g_{j-1} = f_j - f_{j-1}$, on voit que $g_{j-1} = \sum_{k=0}^{2^{j-1}-1} \langle f, \psi_{j,k}\rangle$. Ceci signifie que si on écrit $F_j = F_{j-1} \oplus G_{j-1}$, on a

$$F_{j-1} = \text{Vect}\{\varphi_{j-1,k} \setminus k = 0, \dots, 2^{j-1}-1\}$$
$$\text{et} \quad G_{j-1} = \text{Vect}\{\psi_{j,k} \setminus k = 0, \dots, 2^{j-1}-1\}.$$

On a les relations suivantes, pour $k = 0, \dots, 2^{j-1}-1$:

$$\psi_{j,k} = \frac{\varphi_{j,2k} - \varphi_{j,2k+1}}{\sqrt{2}} \quad \text{et} \quad \varphi_{j-1,k} = \frac{\varphi_{j+1,2k} + \varphi_{j+1,2k+1}}{\sqrt{2}}. \tag{2.1}$$

4. La quantité $x^{(0)}[k]$ est simplement la valeur que prend f sur l'intervalle I_k, multipliée par $2^{j/2}$.
Pour étudier l'opérateur $\Phi_i : x^{(i)} \mapsto (x^{(i+1)}, d^{(i+1)})$, il faut considérer
– pour l'ensemble de départ, la base canonique de $\mathbb{R}^N$, $N = 2^{j-i}$.
– pour l'ensemble d'arrivée, la base « alternée » $\{e_0, f_0, \dots, e_{N/2-1}, f_{N/2-1}\}$, où on a noté $\{e_0, \dots, e_{N/2-1}, f_0, \dots, f_{N/2-1}\}$ la base canonique de $\mathbb{R}^{N/2} \times \mathbb{R}^{N/2}$.
Dans ces bases, la matrice de Φ_i est une diagonale de blocs $\frac{1}{\sqrt{2}}\left(\begin{smallmatrix} 1 & 1 \\ -1 & 1 \end{smallmatrix}\right)$, donc c'est une rotation d'angle $\pi/2$ sur chaque sous-espace $\text{Vect}(e_i, f_i)$.
On vérifie que $d^{(i)}$ est de moyenne nulle, donc $x^{(i)}$ a la même moyenne que $x^{(i+1)}$. De proche en proche, la moyenne de $x^{(0)}$ est celle de $x^{(i)}$ et donc au final vaut $x^{(j)}[0]$.

5. On définit $\widetilde{x}^{(i)}[k] \overset{\text{déf.}}{=} \langle f, \varphi_{j-i,k} \rangle$ et $\widetilde{d}^{(i)}[k] \overset{\text{déf.}}{=} \langle f, \psi_{j-i+1,k} \rangle$. On a $\widetilde{x}^{(0)} = x^{(0)}$. Il s'agit de montrer que $\widetilde{x}^{(i)}$ et $\widetilde{d}^{(i)}$ vérifient les mêmes équations de récurrence que $x^{(i)}$ et $d^{(i)}$. Ceci impliquera que $\widetilde{x}^{(i)} = x^{(i)}$ et $\widetilde{d}^{(i)} = d^{(i)}$ Ces relations sont évidentes en prenant les produits scalaires des équations (2.1) avec la fonction f.
6. Comme Φ_i est une isométrie, on a $\|x^{(i)}\|_2^2 = \|x^{(i+1)}\|_2^2 + \|d^{(i+1)}\|_2^2$. En itérant, on trouve

$$\|f\|_2^2 = \|x^{(0)}\|_2^2 = \sum_{i=1}^{j} \|d^{(i)}\|_2^2 + |x^{(j)}[0]|^2 = \|\Gamma(x^{(0)})\|_2^2.$$

Ce qui signifie que Γ est une isométrie. Ceci revient à décomposer le signal discret $x^{(0)}$ sur la base de $\mathbb{R}^n$ formée des Γe_i, où e_i est la base canonique de $\mathbb{R}^n$. Cette base correspond aux fonctions $\psi_{j,k}$ échantillonnées avec un pas de 2^{-j}. On nomme cette base la base de Haar discrète.
Par rapport à la base de Walsh, cette base est formée de fonctions à support très compact. Plus j devient grand, plus le support est réduit.
Comme le vecteur $x^{(i)}$ est de taille $n/2^i$, l'application de l'opérateur Φ_i nécessite $cn2^{-i}$ opérations (c est une constante, représentant une addition, une soustraction, et deux divisions par $\sqrt{2}$). L'algorithme nécessite donc $\sum_{i=0}^{j} cn2^{-i} = 2cn$, c'est-à-dire $O(n)$ opérations. C'est beaucoup plus rapide que la transformée de Walsh, qui nécessite $O(n\log(n))$ opérations !

Le programme 2.1 implémente une fonction MATLAB qui réalise l'opérateur Γ.

Programme 2.1 Procédure `haar`

```
function y = haar(x)
n = length(x); j = log2(n); y = [];
for i=0:j-1
  d = 1/sqrt(2)*( x(1:2:2^(j-i))-x(2:2:2^(j-i)) );
  y = [y;d];
  x = 1/sqrt(2)*( x(1:2:2^(j-i))+x(2:2:2^(j-i)) );
end
y = [y;x];
```

Correction de l'exercice II.5 :

1. Soit $n = 2^k$. La transformée de Walsh 2D s'écrit simplement, pour $f \in \mathbb{C}^{n \times n}$,

$$\mathscr{W}_k(f)[i,j] \overset{\text{déf.}}{=} \sum_{s=0}^{n-1} \sum_{t=0}^{n-1} f[s,t]\chi_{i,j}(s,t).$$

La transformée inverse est $\mathscr{W}_k^{-1} = \frac{1}{n^2}\mathscr{W}_k$. La transformée de Walsh 2D correspond à appliquer une transformée 1D sur les colonnes, puis une transformée 1D sur les lignes. En utilisant la fonction `fwt` écrite à la section 1, annexe A, on peut définir une fonction MATLAB réalisant la transformée, comme le montre le programme 2.2.
2. Pour plus de clarté, on note $\chi_i^{(k)}$ le caractère χ_i sur $(\mathbb{Z}/2\mathbb{Z})^k$, que l'on peut voir comme un vecteur de $\{\pm 1\}^n$. On note $n_i^{(k)}$ le nombre de changements de signe de $\chi_i^{(k)}$, c'est-à-dire

Programme 2.2 Procédure `fwt2d`

```
function y = fwt2d(x)
n = length(x); y = zeros(n,n);
for(i=1:n) y(i,:) = fwt(x(i,:)')'; end;
for(j=1:n) y(:,j) = fwt(y(:,j)); end;
```

Nous allons montrer, par récurrence sur k, que $n_i^{(k)}$ fournit une nouvelle numérotation des $\chi_i^{(k)}$, c'est-à-dire que $i \mapsto n_i^{(k)}$ est une permutation de $\{0,\dots,2^k-1\}$. Pour $k=1$, on a $n_0^{(1)}=0$ et $n_1^{(1)}=1$ donc la propriété est vraie. On suppose la propriété vraie pour $i \mapsto n_i^{(k)}$. Pour $i=0,\dots,2^k-1$, on a, en notant (a,b) la concaténation des vecteurs a et b,

$$\chi_i^{(k+1)} = \left(\chi_i^{(k)}, \chi_i^{(k)}\right) \qquad \text{et} \qquad \chi_{i+2^k}^{(k+1)} = \left(\chi_i^{(k)}, -\chi_i^{(k)}\right).$$

Ceci implique les relations suivantes sur les changements de signe :

$$n_i^{(k+1)} = 2n_i^{(k)} + \varepsilon_i^{(k)} \qquad \text{et} \qquad n_{i+2^k}^{(k+1)} = 2n_i^{(k)} + (1-\varepsilon_i^{(k)}), \tag{2.2}$$

où $\varepsilon_i^{(k)}=1$ s'il y a une discontinuité au milieu de $\chi_i^{(k+1)}$, ce qui revient à dire que $\chi_i^{(k)}[2^k-1] = -1$. Avec les relations trouvées, il est facile de voir que les $n_i^{(k+1)}$ couvrent tout $\{0,\dots,2^{k+1}-1\}$.

3. Le spectre de Walsh se calcule grâce à la fonction `fwt` (voir section 1, annexe A). On peut ensuite classer le spectre par nombre de changements de signe. Ceci peut être fait rapidement en calculant en même temps que la transformée le nombre de changements de signe grâce aux équations (2.2). En effet, les quantités $\varepsilon_i^{(k)}$ vérifient aussi une équation de récurrence, $i=0,\dots,2^k-1$, on a

$$\varepsilon_i^{(k+1)} = \varepsilon_i^{(k)} \qquad \text{et} \qquad \varepsilon_{i+2^k}^{(k+1)} = 1-\varepsilon_i^{(k)}.$$

Par exemple, la routine MATLAB 2.3 calcule le vecteur $n^{(k)}$.

Programme 2.3 Procédure `nbr_chgt_signe`

```
function nk = nbr_chgt_signe(n)
p = log2(n); nk = 0; ek = 0;
for k=1:p
  ek = [ek; 1-ek]; nk = 2*[nk;nk]+ek;
end
```

4. En gardant seulement les coefficients correspondant aux fonctions avec peu de changements de signe, on reconstruit une fonction avec moins de détails. Ceci a pour effet de conserver moins d'information, et donc permet une compression du signal.
   La compression par transformée de Walsh est rapide à calculer (seulement des additions et des soustractions). Par contre, elle introduit des discontinuités dans le signal souvent inacceptables.
5. Le nombre de changements de signe n'est pas bien défini pour une fonction 2D. Pour une fonction $\chi_{i,j}$, on note n_i le nombre de changements de signe sur l'axe des x, et n_j le nombre de changements de signe sur l'axe des y. On peut par exemple classer les fonctions par ordre de n_i+n_j croissant, et on règle les cas d'égalité par ordre de n_j croissant.

Correction de l'exercice II.6 :

1. Comme on a $\mathscr{W}_k^{-1} = \frac{1}{2^k}\mathscr{W}_k = \frac{1}{2^k}\mathscr{W}_k{}^{\mathrm{T}}$, on a bien $W_n W_n{}^{\mathrm{T}} = n\mathrm{Id}_n$, avec $n = 2^k$.
2. La première chose à remarquer est que si on permute les lignes et les colonnes d'une matrice de Hadamard, alors elle reste de Hadamard. Il en est de même si l'on multiplie une ligne ou une colonne par -1. En multipliant par -1 les colonnes puis les lignes qui commencent par -1, on met ainsi la matrice sous forme normalisée. Par permutation sur les lignes, on peut réarranger les trois premières lignes pour qu'elles soient de la forme annoncée. Par les propriétés d'orthogonalité entre ces trois lignes, on obtient les équations
$$i+j+j+l=n, \quad i+j-k-l=0, \quad i-j+k-l=0, \quad i-j-k+l=0.$$
Ceci mène à $i=j=k=l=\frac{n}{4}$, et donc n est divisible par 4.
3. Pour commencer, $\eta(i-j)=\eta(-1)\eta(j-i)=-\eta(j-i)$ puisque $p=4k-1$ (utiliser la formule d'Euler 1.1, chap. II, pour le voir). Donc Q est anti-symétrique. Soit $c\neq 0$, on calcule ensuite
$$\sum_{b=0}^{p-1}\eta(b)\eta(b+c)=\sum_{b=1}^{p-1}\eta(b)\eta(bz)=\sum_{z\neq 1}\eta(z)=0-\eta(1)=-1.$$
On a fait le changement de variable $z=\frac{b+c}{b}$, en utilisant le fait que $b\mapsto z$ est une bijection de $\mathbb{F}_q^*$ sur $\mathbb{F}_q\backslash\{1\}$.
Si $i=j$, on a $(QQ^{\mathrm{T}})_{ii}=\sum_{b=0}^{p-1}\eta(b)=p-1$. Si $i\neq j$, on a
$$(QQ^{\mathrm{T}})_{ij}=\sum_{k=0}^{p-1}\eta(k-i)\eta(k-j)=\sum_{b=0}^{p-1}\eta(b)\eta(b+c)=-1,$$
avec $b=k-i$ et $c=i-j$.
Comme $\mathbb{F}_p^*$ contient $\frac{1}{2}(p-1)$ résidus quadratiques et autant de non résidus, chaque ligne de Q contient autant de $+1$ que de -1 et $QJ=JQ=0$.
4. On a
$$H_n H_n{}^{\mathrm{T}}=\begin{pmatrix}1 & v\\ v^{\mathrm{T}} & Q-\mathrm{Id}_p\end{pmatrix}\begin{pmatrix}1 & v^{\mathrm{T}}\\ v & Q^{\mathrm{T}}-\mathrm{Id}_p\end{pmatrix}=\begin{pmatrix}p+1 & 0\\ 0 & J+(Q-\mathrm{Id}_p)(Q^{\mathrm{T}}-\mathrm{Id}_p)\end{pmatrix},$$
et $J+(Q-\mathrm{Id}_p)(Q^{\mathrm{T}}-\mathrm{Id}_p)=J+p\mathrm{Id}_p-J-Q-Q^{\mathrm{T}}+\mathrm{Id}_p=(p+1)\mathrm{Id}_p$.
5. La quantité $|\det(A)|$ mesure le volume du parallélépipède engendré par les vecteurs colonnes de A. Ce volume est plus petit que le produit des normes de ces vecteurs. Si on a $|a_{ij}|\leqslant 1$, alors la norme d'un vecteur colonne est majorée par $\sqrt{n}$ est on trouve bien la majoration de Hadamard. Si A est une matrice de Hadamard, on a $\det(A)^2=\det(AA^{\mathrm{T}})=\det(n\mathrm{Id}_n)=n^n$.

La procédure MAPLE 2.4 construit la matrice de Paley H_n, et on peut la tester avec le programme 2.5.

Correction de l'exercice II.7 :

1. On vérifie que
$$(A\otimes A)(A\otimes A)^*=\begin{pmatrix}\langle A_1,A_1\rangle AA^* & \cdots & \langle A_1,A_s\rangle AA^*\\ \vdots & & \vdots\\ \langle A_s,A_1\rangle AA^* & \cdots & \langle A_s,A_s\rangle AA^*\end{pmatrix}=s^2\mathrm{Id}_{s^2},$$
où l'on a noté A_i la $i^{\text{ième}}$ colonne de A. Par récurrence, on obtient le résultat voulu.

Programme 2.4 Procédure `MatricePaley`

```
MatricePaley := proc(p)
local H,Q,i,j;
with(numtheory):
Q := Matrix(1..p,1..p);
for i from 1 to p do
for j from 1 to p do
  Q[i,j] := legendre(i-j,p);
end do:
end do:
H := Matrix(1..p+1,1..p+1);
H[2..p+1,2..p+1] := Q-Matrix(1..p,1..p, shape=identity);
H[1..p+1,1] := 1; H[1,1..p+1] := 1;
return H;
end proc:
```

Programme 2.5 Test de la construction de Paley

```
H := MatricePaley(31);
evalm(H&*transpose(H));
```

2. Pour la transformée de Walsh, on a $W_{2^k} = A^{\otimes k}$ avec $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$.
3. La procédure MATLAB 2.6 réalise le calcul de la transformée de façon rapide (de l'ordre de $O(N \log_s(N))$ opérations, où N est la taille du vecteur à transformer). On a $x = (A^{\otimes n})^{-1} y = \frac{1}{s^n}(A^*)^{\otimes n}$. La transformée inverse s'obtient donc en passant A^* à la place de A à l'algorithme, et en divisant le résultat par s^n.

Programme 2.6 Procédure `decompose_tensoriel`

```
function y = decompose_tensoriel(x,A)
s = length(A); m = length(x); m0 = m/s;
if(m==1) y = x; return; end;
B = zeros(m0,s); % résulats temporaires
for j=1:s
  sel = ((j-1)*m0+1):(j*m0);
  B(:,j) = decompose_tensoriel(x(sel),A);
end
y = zeros(m,1);
for i=1:s
  sel = ((i-1)*m0+1):(i*m0);
  for j=1:s
    y(sel) = y(sel) + A(i,j)*B(:,j);
  end
end
```

4. Pour $\alpha = \pi/4$, on obtient la transformée de Walsh ordinaire. Pour $\alpha = \pi/2$, la transformée réalise une symétrie.

Correction de l'exercice II.8 :

1. $\langle \cdot, \cdot \rangle$ est la forme bilinéaire canonique sur $E \times E$. On identifie un élément $x \in E$ a la forme linéaire $\langle x, \cdot \rangle$. Cette identification marche car la forme est non-dégénérée, et correspond au crochet de la dualité.

2. On considère l'application $\Phi : a \mapsto \chi_a$. C'est un morphisme de groupes. On montre facilement qu'il est injectif. En effet, si $\Phi(a) = 0$, ceci signifie que pour tout $x \in E$ on a $\chi_1(\langle x, a\rangle) = 1$, et comme χ_1 est injectif, $\forall x \in E$, $\langle x, a\rangle = 0$. Comme la forme bilinéaire $\langle \cdot, \cdot \rangle$ est non-dégénérée, on a $a = 0$. Enfin, comme E et $\widehat{E}$ ont même dimension, le morphisme est bijectif.
3. Si $x \in H$, par la propriété de groupe, $x + \cdots + x = kx \in H$, donc H est stable pour la loi externe, donc c'est un espace vectoriel.
   On a $H^\sharp = \{\chi_a \setminus \forall x \in H, \chi_a(x) = 0\}$, qui est en correspondance par Φ avec l'ensemble $\{a \setminus \forall x \in H, \langle x, a\rangle = 0\} = H^\perp$.
4. La formule de Poisson, 3.3, chap. II, est encore valable sur E. La preuve de la formule de MacWilliams, théorème 3.6, chap. II, est encore la même. Seul le calcul de $\widehat{f}(\chi_a)$ est légèrement changé, puisque l'on a

$$\widehat{f}(\chi_a) = (x + (q-1)y)^{k-w(a)}(x-y)^{w(a)}.$$

Correction de l'exercice II.9 : La formule de Poisson s'écrit, pour une fonction f de la classe de Schwartz $\mathscr{S}(\mathbb{R})$:

$$\sum_{n\in\mathbb{Z}} \widehat{f}(n) = \frac{2\pi}{s} \sum_{n\in\mathbb{Z}} f\left(\frac{2n\pi}{s}\right).$$

On vérifie qu'au sens des distributions, le membre de gauche est égal à $\left\langle \widehat{f}, \Pi_1 \right\rangle = \left\langle f, \widehat{\Pi_1} \right\rangle$, et que le membre de droite est égal à $\left\langle \frac{2\pi}{s}\Pi_{\frac{2\pi}{s}}, f \right\rangle$. L'égalité étant valide pour tout f dans $\mathscr{S}(\mathbb{R})$, on en déduit la formule demandée.

Correction de l'exercice II.10 :

1. On a, avec la formule d'inversion de Fourier,

$$f(x) = \frac{1}{2\pi} \int_{I_T} \widehat{f}(\omega) e^{\mathrm{i}\omega x} \mathrm{d}\omega.$$

   Avec le théorème de dérivation sous le signe d'intégration, on voit que f est de classe $\mathscr{C}^\infty$.
2. En appliquant le résultat de l'exercice II.9, on voit facilement que

$$\widehat{f_d}(\omega) = \frac{1}{T} \sum_{k\in\mathbb{Z}} \widehat{f}\left(\omega - \frac{2k\pi}{T}\right).$$

   En effet, on a $f_d = f \cdot \Pi_T$, et en prenant la transformée de Fourier des deux membres (au sens des distributions), on trouve $\widehat{f_d}(\omega) = f * \widehat{\Pi_T}(\omega)$. Si $n \neq 0$, le support de $\widehat{f}\left(\cdot - \frac{n\pi}{T}\right)$ et celui de $\widehat{f}$ sont d'intersection vide. Ceci implique donc le résultat demandé.
3. Le résultat de la question précédente peut s'écrire $\widehat{f}(\omega) = T h_T \widehat{f_d}$, où h_T est l'indicatrice de l'intervalle I_T. Sa transformée de Fourier inverse est $\mathscr{F}^{-1}(h_T) = T \operatorname{sinc}_T$. En prenant la transformée de Fourier inverse de la relation $\widehat{f}(\omega) = T h_T \widehat{f_d}$, on trouve le théorème d'échantillonnage.
4. On note $g = \operatorname{sinc}_T$. Le fait que les fonctions $\{g(. - nT)\}$ soient orthogonales se vérifie immédiatement en utilisant la formule de Plancherel :

$$\langle g, g(. - nT)\rangle = \frac{1}{2\pi} \langle \mathscr{F}(g), \mathscr{F}(g(. - nT))\rangle = \frac{T^2}{2\pi} \left\langle h_T, h_T(\cdot) e^{-\mathrm{i}nT\cdot} \right\rangle = T\delta_0^n.$$

Le fait que cette famille soit totale résulte du théorème d'échantillonnage, puisque la formule de reconstruction (4.7), chap. II, converge en norme L^2.
La projection sur cette base correspond à l'échantillonnage, puisque l'on a la relation $\langle f, g(.-nT)\rangle = f(nT)$.

3 Correction des exercices du chapitre 3

Correction de l'exercice III.1 :

1. On a $u^{(1)} = (0,1)$, $u^{(2)} = (0,2,1,3)$ et $u^{(3)} = (0,4,2,6,1,5,3,7)$.
2. Soit $k^{(n+1)} = \sum_{i=0}^{n} k_i 2^i$ un entier de $n+1$ bits. On note $\widetilde{k}^{(n+1)}$ l'entier avec les bits renversés. On a
$$\widetilde{k}^{(n+1)} = \sum_{i=0}^{n} k_i 2^{n-i} = \sum_{i=0}^{n-1} k_i 2^{n-i} + k_n = 2\widetilde{k}^{(n)} + k_n,$$
ce qui est exactement l'équation de récurrence vérifiée par $u^{(n)}$.
3. $\widetilde{f}$ est utile pour construire un algorithme FFT n'utilisant pas de mémoire temporaire, comme expliqué au paragraphe 2.5, chap. III.
4. On a $\widetilde{f}_g = \widetilde{f^0}$ et $\widetilde{f}_d = \widetilde{f^1}$.
5. La question précédente donne naissance à une fonction MATLAB récursive, programme 3.1. Cette procédure nécessite $O(N\log(N))$ opérations, ce qui est la même complexité que la procédure `rev_bits`.

Programme 3.1 Procédure `rev_bits_rec`

```
function y = rev_bits_rec(x)
n = length(x);
if(n>1) y = [rev_bits_rec(x(1:2:n)) ; rev_bits_rec(x(2:2:n))];
else y = x; end
```

Correction de l'exercice III.2 :

1. φ est un morphisme d'anneaux, et on peut expliciter son inverse. Avec le théorème de Bezout, $\exists(u\,v)$, $up+vq=1$. On prend alors $\psi(k_1,k_2) = k_2 up + k_1 vq \mod N$.
2. On note $\omega_r = e^{\frac{2i\pi}{r}}$. On calcule tout d'abord la transformée $\widehat{2\mathrm{D}}$:
$$\widehat{F}[s_1,s_2] = \sum_{k_1,k_2} f[\psi(k_1,k_2)]\omega_p^{-s_1k_1}\omega_q^{-s_2k_2} = \sum_{k_1,k_2} f[\psi(k_1,k_2)]\omega_N^{-(qs_1k_1+ps_2k_2)}.$$
Calculons la quantité A suivante :
$$A \overset{\text{déf.}}{=} \psi(k_1,k_2)(s_1q+s_2p) = qk_1s_1(qv) + pk_2s_2(pv) \mod N.$$
On veut montrer que $A = qs_1k_1 + ps_2k_2 \mod N$. Avec le théorème Chinois, il suffit de montrer que cette égalité est vraie modulo p et modulo q, ce qu'on vérifie sans problème. On a donc
$$\begin{aligned}\widehat{F}[s_1,s_2] &= \sum_{k_1,k_2} f[\psi(k_1,k_2)]\omega_N^{-\psi(k_1,k_2)(s_1q+s_2p)} \\ &= \sum_k f[k]\omega_N^{-k(s_1q+s_2p)} = \widehat{f}[s_1q+s_2p],\end{aligned}$$
où l'on s'est contenté de faire le changement d'indice $k = \psi(k_1,k_2)$.

3. On a $0 \leqslant s_1 q + s_2 p \leqslant N-1$. De plus, par le lemme chinois, le système de deux équations $\{n = s_1 q \mod p \; ; \; n = s_2 p \mod q\}$ a une unique solution modulo N, et le représentant de cette solution est donc n.
   Pour calculer la FFT de f, il suffit de calculer la FFT 2D de F, et de réordonner les indices selon $(s_1, s_2) \mapsto s_1 q + s_2 p \mod N$.
4. L'étape de l'algorithme de Good-Thomas présentée permet de remplacer l'étape de l'algorithme de Cooley-Tukey expliquée au paragraphe 2.4, chap. III. Ce remplacement permet d'éviter les multiplications internes par ω_N^{-bd} dans l'équation 2.14, chap. III, qui correspondent à l'opérateur $\mathscr{S}_N^x$. Le lemme chinois a en quelque sorte éliminé les « twiddle factors ».
   Le programme 3.2 montre une procédure MATLAB naïve. En réalité, il faudrait

Programme 3.2 Procédure `fft_gt`

```
function y = fft_gt(x,p)
n = length(x); q = n/p;
y = zeros(n,1); m = zeros(p,q);
for i=0:n-1
  m(mod(i,p)+1, mod(i,q)+1) = x(i+1);
end
m = fft2(m);
for(s1=0:p-1) for(s2=0:q-1)
  y( mod(s1*q+s2*p,n)+1 ) = m(s1+1,s2+1);
end; end;
```

appeler, plutôt que `fft2(m)`, une procédure de FFT sur les lignes (longueur q) puis sur les colonnes (longueur p) qui exploite les décompositions de p et q.

Correction de l'exercice III.3 :

1. La première partie du regroupement ne génère pas de « twiddle factors », il n'est donc pas la peine de la décomposer.
   En utilisant le fait que $(n_1, n_2) \mapsto n_1 + n_2 N/4$ est une bijection de l'ensemble produit $\{0, \ldots, N/4-1\} \times \{0, \ldots, 3\}$ sur $\{0, \ldots, N-1\}$, on peut écrire

$$\widehat{f}[4k+2j+1] = \sum_{n_1=0}^{N/4-1} \sum_{n_2=0}^{3} \omega_N^{-(n_1+n_2N/4)(4k+2j-1)} f[n_1 + n_2 N/4].$$

   Pour conclure à l'expression proposée, il suffit de remarquer que

$$\omega_N^{-(n_1+n_2N/4)(4k+2j-1)} = \omega_{N/4}^{-kn_1} \omega_N^{-n_1(2j+1)} \omega_4^{-n_2(2j+1)}.$$

   Les sommes intérieures sont des TFD de longueur 4, et elles sont triviales à calculer puisque les racines complexes utilisées sont $\{\pm 1, \pm \mathrm{i}\}$.
2. On peut effectuer des regroupements des fréquences par paquets de 2^l (le cas $l = 1$ correspond à la décimation fréquentielle classique, et $l = 2$ à la question 1). Ceci conduit à la décomposition suivante de la TFD :

$$\widehat{f}[2^l k + 2j + 1] = \sum_{n_1=0}^{N/2^l-1} \omega_{N/2^l}^{-kn_1} \omega_N^{-n_1(2j+1)} \sum_{n_2=0}^{2^l} f\left[n_1 + n_2 N/2^l\right] \omega_{2^l}^{-n_2(2j+1)},$$

   pour $j = 0, \ldots, 2^{l-1} - 1$ et $k = 0, \ldots, N/2^l - 1$. On peut montrer que le split-radix de taille 2^l optimal correspond à celui de la question 1 (voir [75]). Ceci provient du fait que les TFD de taille 4 ne nécessitent aucune multiplication complexe.

3. Pour un schéma de décimation temporelle, il s'agit de regrouper non pas les fréquences du vecteur transformé, mais les entrées du vecteur à transformer. Ceci donne naissance à l'équation de décomposition suivante :

$$\widehat{f}[n_1+n_2N/4] = (-1)^{n_2}\sum_{k=0}^{N/2-1} f[2k]\omega_{N/2}^{n_1k} + \sum_{j=0}^{1}\omega_N^{n_1(2j+1)}\omega_4^{n_2(2j+1)}\sum_{k=0}^{N/4} f[4k+2j+1]\omega_{N/4}^{kn_1},$$

pour $n_1 = 0,\ldots,N/4$ et $n_2 = 0,\ldots,3$. La TFD de longueur N est ainsi décomposée en la somme d'une TFD de longueur $N/2$ et deux TFD de longueur $N/4$. L'analyse de l'optimalité de cette décomposition est identique à celle de la version décimation fréquentielle de la question 1.

Correction de l'exercice III.4 :

1. On a $f = \sum_{j=0}^{p-1} f_j[\cdot - jM]$. Par bilinéarité du produit de convolution, on a donc $f \star g = \sum_{j=0}^{p-1} f_j \star g[\cdot - jM]$
2. Il faut donc calculer les p produits de convolution $f_j \star g$. Comme les deux vecteurs ont pour taille M, ce produit peut se calculer par FFT en ajoutant seulement $M-1$ zéros. En supposant que l'algorithme FFT nécessite $cM\log(M)$, le calcul de convolution nécessite $2cM\log(M)+M$ opérations, et ce calcul est effectué p fois.
3. Si N n'est pas un multiple de M, il convient d'ajouter des zéros à f pour atteindre une taille égale au multiple de M juste après N.

La procédure MATLAB `convol`, programme 3.3, met en place cette méthode. Il est à noter que la FFT du vecteur g (auquel on a ajouté $M-1$ zéros) est stockée une fois pour toutes dans la variable `fg`.

Programme 3.3 Procédure `convol`

```
function y = convol(f,g)
N = length(f); M = length(g); p = N/M;
y = zeros(M+N-1,1);
fg = fft( [g; zeros(M-1,1)] );
for j=0:p-1
  fj = [f( (1:M)+j*M ); zeros(M-1,1)];
  sel = (j*M+1):((j+2)*M-1); % les indices concernés
  y(sel) = y(sel) + ifft( fft(fj).*fg );
end
```

Correction de l'exercice III.5 :

1. On a, pour $x = (x_0,\ldots,x_{N-1})^{\mathrm{T}}$,

$$\Omega_N(Dx)[k] = (\Omega_N(x_{N-1},x_0,\ldots,x_{N-2})^{\mathrm{T}})[k] = \sum_{i=0}^{N-1} x_{i-1}\omega_N^{-ki}$$
$$= \omega_N^{-k}\sum_{i=0}^{N-1} x_i\omega_N^{-ki} = \omega_N^{-k}(\Omega_N x)[k],$$

ce qui est le résultat demandé.

2. On utilise la décomposition $C = \sum_{i=0}^{N} c_i R^i$, d'où

$$\Omega_N C \Omega_N^{-1} = \sum_{i=0}^{N} c_i \Omega_N R^i \Omega_N^{-1} = \sum_{i=0}^{N} c_i D^i = \Delta.$$

On remarque que si $y \in \mathbb{C}^N$, $\Delta y = (\Omega_N c) \cdot y$. On a donc $\Delta \Omega_N x = (\Omega_N c) \cdot (\Omega_N x)$.

3. Le fait que $Cx = c * x$ se vérifie immédiatement. Avec le théorème de convolution, on a $\mathscr{F}(Cx) = \mathscr{F}(c*x) = \widehat{c} \cdot \widehat{x}$. Comme $\widehat{x} = \Omega_N x$, on obtient bien la même formule.

Correction de l'exercice III.6 : En utilisant la formule d'inversion de Fourier 1.4, chap. III, on a

$$\begin{aligned} f_0[\eta k] &= \frac{1}{P} \sum_{s=0}^{P-1} \widehat{f_0}[s] \omega_P^{s\eta k} \\ &= \frac{1}{N} \sum_{s=0}^{N_0} \widehat{f}[s] \omega_N^{sk} + \frac{1}{N} \sum_{s=N_0+1}^{N-1} \widehat{f}[s] \omega_N^{sk-P+N} = f[k]. \end{aligned}$$

Le programme 3.4 implémente la technique d'interpolation exposée.

Programme 3.4 Procédure `interp_trigo`

```
function y = interp_trigo(x,eta)
N = length(x); N0 = (N-1)/2; P =N*eta;
f = fft(x);
f = eta*[f(1:N0+1); zeros(P-N,1); f(N0+2:N)];
y = real( ifft(f) );
```

Correction de l'exercice III.7 :

1. En utilisant la relation trigonométrique

$$\cos((k+1)\theta) + \cos((k-1)\theta) = \cos(k\theta)\cos(\theta), \quad \text{avec} \quad \theta = \arccos(X),$$

on obtient la relation de récurrence $T_{k+1} = 2XT_k - T_{k-1}$. Cette relation montre que T_k est un polynôme à coefficients entiers de degré k.
Comme on a $T_N(x_j) = \cos\left(N(j+1/2)\frac{\pi}{N}\right) = 0$, on a trouvé les N racines de T_N qui est de degré N.

2. Comme les $\{T_k\}_{k=0}^{N-1}$ ont des degrés différents, ils forment une famille libre de l'espace des polynômes de degré inférieur à $N-1$. Comme ils sont au nombre de N, c'est une base de cet espace.

3. La formule d'inversion peut se vérifier à la main, mais c'est assez pénible. Il faut mieux se ramener à des vecteurs orthogonaux. On peut en effet montrer assez facilement que les vecteurs

$$v_k = \left\{ \lambda_k \sqrt{\frac{2}{N}} \cos\left(\frac{k\pi}{N}\left(n+\frac{1}{2}\right)\right) \right\}_{k=0}^{N-1} \quad \text{avec} \quad \lambda_k = \begin{cases} 2^{-1/2} & \text{si } k=0 \\ 1 & \text{sinon} \end{cases}$$

forment une base orthonormée de $\mathbb{C}^N$. L'article de STRANG [69] propose une jolie preuve qui utilise le fait que les v_k sont vecteurs propres d'une certaine matrice symétrique.
Le programme MATLAB 3.5 implémente la transformée $\mathscr{C}_2$ par l'intermédiaire d'une FFT de taille $4N$. Le programme 3.6 implémente lui la transformée $\mathscr{C}_3$, toujours par l'intermédiaire d'une FFT de taille $4N$.

Programme 3.5 Procédure `dct2`

```
function y = dct2(x)
n = length(x);
y(2:2:2*n,1) = x;
y = [y ; zeros(2*n,1)];
y = real( fft(y) ); y = y(1:n);
```

Programme 3.6 Procédure `dct3`

```
function y = dct3(x)
n = length(x);
y = [x ; zeros(3*n,1)];
y = real( fft(y) ); y = y(2:2:2*n) - x(1)/2;
```

4. Les coefficients $\alpha = (\alpha_0, \ldots, \alpha_{N-1})^{\mathrm{T}}$ du polynôme P_{N-1} vérifient

$$f[j] = \sum_{k=0}^{N-1} \alpha_k T_k(x_j) = \sum_{k=0}^{N-1} \alpha_k \cos\left(k(j+1/2)\frac{\pi}{N}\right) = \mathscr{C}_3(\alpha) + \frac{\alpha_0}{2}. \tag{3.1}$$

Il y a donc une légère difficulté à cause du terme « parasite » $\frac{\alpha_0}{2}$. Si on note $\mathbf{1}$ la fonction constante égale à 1, on a $\mathscr{C}_2(\mathbf{1}) = N\delta_0$. On peut donc inverser l'égalité 3.1 :

$$\frac{2}{N}\mathscr{C}_2(f) = \frac{2}{N}\mathscr{C}_2(\mathscr{C}_3(\alpha)) + \frac{1}{N}\mathscr{C}_2(\alpha_0\mathbf{1}) = \alpha + \alpha_0\delta_0 = (2\alpha_0, \alpha_1, \ldots, \alpha_{N-1}).$$

Le programme 3.7 montre comment, avec MATLAB on peut utiliser la procédure `dct3` pour effectuer une interpolation de Chebyshev. Ici, on interpole la fonction $f(x) = \frac{1}{\alpha^2+x^2}$ connue en `n` points (les points x_k). Le programme dessine la courbe interpolée en l'évaluant en `nn` points espacés régulièrement.

Programme 3.7 Interpolation de Chebyshev

```
n = 16; nn = 200; alpha = 0.3;
x = cos( ((0:n-1)+1/2)*pi/n )';
f = 1./(alpha^2+x.^2);
coef = 2/n*dct2(f);
coef(1) = coef(1)*1/2;
xx = (-1:2/(nn-1):1)';
ff = zeros(nn,1);
for k=0:n-1
  ff = ff + coef(k+1)*cos(k*acos(xx));
end
plot(x,f,'o',xx,ff);
```

Correction de l'exercice III.8 : On prouve la formule demandée simplement par n intégrations par parties.

On remarquera que la multiplication par $(i\xi)^n$ correspond à un filtre qui amplifie les hautes fréquences. C'est tout à fait logique, puisque la dérivation fait perdre de la régularité, tout comme un filtre amplifiant les hautes fréquences (décroissance moins rapide de la transformée, que l'on peut comparer à une amplification du « bruit »).

Le programme MATLAB 3.8 propose une fonction qui réalise une dérivée fractionnaire à l'ordre `alpha`. La procédure utilise l'algorithme FFT, donc, pour qu'elle approche avec justesse la dérivée de la fonction d'origine, il faut que l'échantillonnage soit assez fin, et

Programme 3.8 Procédure `der_frac`

```
function y = der_frac(f, alpha)
n = length(f)/2;
fce = pi*[(0:n)/n, ((-n+1):-1)/n];
f = fft( f );
f = (-i*fce).^alpha.*f; f = real(ifft(f));
```

il faut que $\widehat{h}$ soit à support dans $[-\pi,\pi]$ (sinon, attention à l'aliasing, car on ne remplit plus les hypothèses du théorème de Shannon, exercice II.10).

Correction de l'exercice III.9 :

1. Comme $(\Omega_N)^4 = N^2 \mathrm{Id}_N$, les valeurs propres de Ω_N sont $\{\pm\sqrt{N}, \pm \mathrm{i}\sqrt{N}\}$.
2. On a $\mathscr{F}^\alpha \circ \mathscr{F}^\beta = PD^\alpha P^* PD^\beta P^* = PD^{\alpha+\beta}P^* = \mathscr{F}^{\alpha+\beta}$.
3. On remarque que $\Omega_N^2 = \Delta$ où Δ est la matrice telle que $\Delta e_i = e_{-i}$ (inversion du signal). Cette matrice correspond à la diagonale inversée. Plus α (modulo 4) est proche de 2, plus cette diagonale inversée est prépondérante dans $\mathscr{F}^\alpha$, et plus α est proche de 0, plus la diagonale est dominante.
4. Pour $N > 4$, il existe une infinité de bases orthogonale de diagonalisation de Ω_N (les espaces propres sont de dimension supérieure à 1). Chaque base différente donne naissance à une transformée intermédiaire.

La procédure MATLAB 3.9 calcule une transformée intermédiaire pour une valeur de α donnée.

Programme 3.9 Procédure `tfd_interm`

```
function y = tfd_interm(x,alpha)
n = length(x);
f = (0:n-1)'*(0:n-1);
Omega = exp(-2i*f*pi/n);
[V,D] = eig(Omega,'nobalance');
y = V*D^alpha*ctranspose(V)*x;
```

Correction de l'exercice III.10 :

1. S est une matrice symétrique *réelle*, qui diagonalise en base orthonormée.
2. On note $\omega =$. Le plus simple est d'exploiter la décomposition suivante de S :

$$\begin{pmatrix} -2 & 1 & 0 & \dots & 1 \\ 1 & -2 & 1 & \dots & 0 \\ 0 & 1 & -2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \dots & -2 \end{pmatrix} + \mathrm{diag}\left(2\left(\cos\left(k\frac{2\pi}{N}\right)-1\right) \;\middle\backslash\; k = 0,\dots,N-1\right),$$

que l'on note $S = \Gamma + \Delta$. Γ est une matrice circulante (voir exercice III.5), la multiplication par cette matrice correspond à la convolution par $v = (-2,1,0,\dots,0,1)^{\mathrm{T}}$. Comme la transformée de Fourier de v est la diagonale de Δ, on a $\Omega_N\Gamma = \Delta\Omega_N$ (avec le théorème de convolution). En utilisant la symétrie de S et Ω_N, il vient

$$S\Omega_N = (\Omega_N S)^{\mathrm{T}} = (\Delta\Omega_N)^{\mathrm{T}} + (\Omega_N\Delta)^{\mathrm{T}} = \Delta\Omega_N + \Omega_N\Delta = \Omega_N S.$$

3. On pourra trouver la démonstration de cette propriété classique dans [59]. Il s'agit essentiellement d'utiliser le fait que les espaces propres de f sont stables par g, ce qui résulte du fait que si φ et ψ commutent, alors $\ker(\varphi)$ est stable par ψ. C'est exactement ceci que l'on utilise pour prouver le *lemme de Schur* 2.5, chap. VII.
4. Un simple calcul montre que cet opérateur est symétrique et orthogonal. Par exemple pour $N = 5$, on a
$$P = \frac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \end{pmatrix}.$$
Pour simplifier les explications, supposons que $N = 2p$. Pour voir que PSP^{-1} est tridiagonal, il faut utiliser la base canonique de $\mathbb{C}^N$, notée $\{\delta_0, \ldots, \delta_{N-1}\}$, et considérer la base $\mathscr{B} \overset{\text{déf.}}{=} \{e_0 = \delta_0, e_1, \ldots, e_p, f_1, \ldots, f_p\}$.
On a noté $e_i = \frac{1}{\sqrt{2}}(\delta_i + \delta_{-i})$ et $f_i = \frac{1}{\sqrt{2}}(\delta_i + \delta_{-i})$. Alors on a
$$S(e_0) = S(\delta_0) = C_0\delta_0 + \delta_1 + \delta_{-1} = C_0 e_0 + e_1,$$
$$\forall 1 \leqslant i < p, \quad S(e_i) = \frac{1}{\sqrt{2}}S(\delta_i) + \frac{1}{\sqrt{2}}S(\delta_{-i}) = \frac{1}{\sqrt{2}}(C_i + C_{N-i})e_i + e_{i+1} + e_{i-1},$$
$$\forall 1 \leqslant i < p, \quad S(f_i) = \frac{1}{\sqrt{2}}S(\delta_i) - \frac{1}{\sqrt{2}}S(\delta_{-i}) = \frac{1}{\sqrt{2}}(C_i - C_{N-i})f_i + f_{i+1} + f_{i-1}.$$
Il faut faire attention que les indices sont exprimés modulo N, et que pour $i = p$, les inégalités sont valables à condition de prendre la convention $e_{p+1} = e_p$ et $f_{p+1} = f_p$. Donc l'opérateur S, exprimé dans la base $\mathscr{B}$ est tridiagonal, ce qui revient à dire que PSP^{-1} est tridiagonal.
5. La démonstration utilise une procédure de séparation des valeurs propres grâce aux *suites de Sturm*. Voir [16].
6. Cette construction est totalement intrinsèque, elle ne repose pas sur un choix arbitraire des vecteurs de diagonalisation. Pour construire une TFD partielle, il faut faire un choix dans l'ordre des vecteurs propres. Dans [13] le nombre de changements de signes est utilisé.

La procédure MATLAB 3.10 construit la base de vecteurs propres exposée dans cette exercice, pour une taille N donnée en paramètre.

Programme 3.10 Procédure `vect_propres_tfd`

```
function V = vect_propres_tfd(n)
x = (0:n-1)'*(0:n-1);
Omega = 1/sqrt(n)*exp(2i*x*pi/n);
d = 2*( cos(2*pi/n*(0:n-1)) - 2 );
S = diag(d,0) + diag(ones(n-1,1),1) + diag(ones(n-1,1),-1);
S(1,n) = 1; S(n,1) = 1;
[V,D] = eig(S);
```

Correction de l'exercice III.11 :

1. Faisons uniquement l'un des deux calculs :
$$\mathscr{F}(k \top f)[t] = \sum_{s=0}^{n-1} \omega_n^{-st} f[s-k] = \omega_n^{-kt} \sum_{r=0}^{n-1} \omega_n^{-rt} f[r] = (k \perp f)[t],$$

où l'on a fait le changement d'indice de sommation $r = s - k$.

2. Si $\mathscr{B} = \{e_i\}_{i=0}^{n-1}$ est orthonormée pour $\perp$, alors $\mathscr{F}(\mathscr{B}) = \{\mathscr{F}(e_i)\}_{i=0}^{n-1}$ est orthonormée pour $\top$. Bien sûr, $\perp$ et $\top$ sont interchangeables.
3. f est orthonormée pour $\top$ si et seulement si, pour tout k

$$\frac{1}{n} f * \widetilde{f}[k] = \frac{1}{n} \sum_{s=0}^{n-1} f[s]\overline{f[s-k]} = \delta_0[k],$$

où on a noté $\widetilde{f}[s] = \overline{f[-s]}$. En prenant la transformée de Fourier de $f * \widetilde{f} = \delta_0$, et en utilisant le théorème de convolution, on trouve $|\widehat{f}|^2 = \mathbf{1}$, où on a noté $\mathbf{1}$ la fonction constante égale à 1.
4. Il est évident que $|\widehat{f_0}|^2 = 1$, donc f_0 est orthonormée pour $\top$.
Si on veut que g soit orthonormée pour $\perp$, alors on applique la construction précédente à $f = \widehat{g}$, et on considère $g_0 = \mathscr{F}^{-1}(f_0)$.
5. On a $\langle \varphi, k \top g \rangle = \frac{1}{n} \sum_{s=0}^{n-1} g[s-k]\varphi[s] = \frac{1}{n} f * \widetilde{f}[k]$. Comme $\mathscr{G}$ s'exprime comme une convolution, on peut le calculer de façon rapide par FFT.

La procédure MATLAB 3.11 permet d'orthogonaliser un vecteur donné.

Programme 3.11 Procédure `fft_orthog`

```
function y = fft_orthog(x)
y = fft(x);
if( min(abs(y))==0 ) error('La TFD de x s''annule.'); return; end;
y = y./abs(y); y = real( ifft(y) );
```

4 Correction des exercices du chapitre 4

Correction de l'exercice IV.1 : Avec la formule d'inversion de Fourier, on a

$$f(t) = \frac{1}{2\pi} \int_{-A}^{A} \widehat{f}(\omega) e^{-\mathrm{i}\omega t} \mathrm{d}\omega.$$

Par dérivation sous le signe intégral, on montre que f est de classe $\mathscr{C}^\infty$.
Si $f(t) = 0$ pour $t \in [c,d]$, on aurait, au point $t_0 = \frac{1}{2}(c+d)$,

$$f^{(n)}(t_0) = \frac{1}{2\pi} \int_{-A}^{A} (-\mathrm{i}\omega)^n \widehat{f}(\omega) e^{-\mathrm{i}\omega t_0} \mathrm{d}\omega = 0.$$

En développant $t \mapsto \exp(-\mathrm{i}(t - t_0))$ au voisinage de 0, on trouve e

$$f(t) = \frac{1}{2\pi} \sum_{n=0}^{+\infty} \frac{[-\mathrm{i}(t-t_0)]^n}{n!} \int_{-A}^{A} \widehat{f}(\omega) \omega^n e^{\mathrm{i}\omega t_0} \mathrm{d}\omega = 0,$$

ce qui est absurde. L'interversion entre $\sum$ et $\int$ est justifiée par le théorème de Fubini.

Correction de l'exercice IV.2 :

1. On utilise les schémas de différences finies $\frac{\partial u}{\partial t}(t,x) \approx \frac{1}{h}(u(t+h,x) - u(t+h,x))$ et $\frac{\partial^2 u}{\partial x^2}(t,x) \approx \frac{1}{d^2}(u(t,x+d) + u(t,x-d) - 2u(t,x))$, ce qui mène à l'équation annoncée. Bien sûr, les vecteurs sont considérés comme cycliques.

2. On a $u^{n+1} = g * u^n$, avec $g = \{1-2s, s, 0, \ldots, 0, s\}$. Comme le support de g est très petit, on n'a pas intérêt à utiliser l'algorithme FFT.
3. Comme la transformée de Fourier est une isométrie, u^n reste borné si et seulement si $\widehat{u^n}$ reste borné. Comme on a $\widehat{u^n} = \mathscr{F}(g * \cdots * g * u^0) = (\widehat{g})^n \cdot \widehat{u^0}$, cette condition est équivalente à $|\widehat{g}| \leqslant 1$. On peut calculer cette transformée de Fourier, et on trouve

$$1-4s \leqslant \widehat{g}[k] = 1+2s\left(\cos\left(\frac{2\pi}{N}\right)-1\right) \leqslant 1$$

La condition de stabilité s'écrit donc $-1 \leqslant 1-4s$, c'est-à-dire $s \leqslant \frac{1}{2}$. Cette condition est appelée condition de *Courant-Friedrichs-Levy* ou *CFL*. La procédure MATLAB 4.1 calcule la solution u^t après `t` itérations, en lui spécifiant la condition initiale (variable `x`) et la précision (variable `s`).

Programme 4.1 Procédure `resolution_explicite`

```
function y = resolution_explicite(x,s,t)
n = length(x); y = x;
for i=1:t
  for k=1:n
    y1(k) = s*y( mod(k-2,n)+1 ) + s*y( mod(k,n)+1 ) + (1-2*s)*y(k);
  end
  y = y1;
end
```

4. En utilisant les propriétés de convolution, on obtient la solution

$$\widehat{u^{n+1}}[k] = \frac{1+(1-\theta)\widehat{A}[k]}{1-\theta\widehat{A}[k]}\widehat{u^n}[k] = h\left(\frac{2k\pi}{N}\right)\cdot\widehat{u^n},$$

qu'il est donc possible de résoudre en Fourier. On remarque que

$$\frac{1-4(1-\theta)s}{1+4\theta s} \leqslant h(\omega) = \frac{1+2s(1-\theta)(\cos(\omega)-1)}{1-2s\theta(\cos(\omega)-1)} \leqslant 1,$$

ces inégalités étant faciles à trouver par une étude de fonction, et ce sont les meilleures possibles. La condition de stabilité s'écrit donc $-1 \geqslant \frac{1-4(1-\theta)s}{1+4\theta s}$, ce qui est équivalent à $s(1-2\theta) \leqslant \frac{1}{2}$. Ainsi, si $\theta \geqslant \frac{1}{4}$, le schéma est toujours stable. Si $\theta < \frac{1}{4}$, il faut que $s \leqslant \frac{1}{2(1-2\theta)}$. On retrouve la condition CFL pour $\theta = 0$. Le programme MATLAB 4.2 permet de résoudre l'équation de la chaleur par cette méthode implicite. Par rapport à la procédure 4.1, il prend un argument supplémentaire, `theta`.

Programme 4.2 Procédure `resolution_implicite`

```
function y = resolution_implicite(x,s,theta,t)
n = length(x); y = x;
A = zeros(n,1); A(1) = -2*s; A(2) = s; A(n) = s;
fA = fft(A); y = fft(x);
mult = ( ones(n,1)+(1-theta)*fA )./( ones(n,1)-theta*fA );
for(i=1:t) y = y.*mult; end;
y = real( ifft(y) );
```

Programme 4.3 Procédure `resolution_implicite_2d`

```
function y = resolution_implicite_2d(x,s,theta,t)
n = length(x); y = x;
A = zeros(n,n); A(1,1)=-4*s; A(2,1)=s; A(1,2)=s; A(n,1)=s; A(1,n)=s;
fA = fft2(A); y = fft2(x);
mult = ( ones(n,n)+(1-theta)*fA )./( ones(n,n)-theta*fA );
for(i=1:t) y = y.*mult; end;
y = real( ifft2(y) );
```

5. Les équations 2D sont les mêmes à condition de considérer le filtre A tel que seules les entrées $A[0,0] = -4s$, $A[\pm 1,0] = s$ et $A[0,\pm 1] = s$ soient non nulles. Les conditions de stabilité sont les mêmes. Le programme MATLAB 4.3 résout l'équation de la chaleur pour une fonction 2D donnée.

Correction de l'exercice IV.3 :

1. On a

$$u(t,x) = \sum_{n\in\mathbb{Z}} \widehat{f}(n) e^{-2\pi^2 n^2 t} e^{2i\pi nx} = \int_0^1 \left\{ \sum_{n\in\mathbb{Z}} e^{-2\pi^2 n^2 t} e_n(x-y) \right\} f(y)\mathrm{d}y = p_t * f(x),$$

où la convolution est celle de $L^1([0,1])$. On a noté $p_t(x) = \sum_{n\in\mathbb{Z}} e^{-2\pi^2 n^2 t} e_n(x)$.
Par convergence normale des dérivées pour $t > 0$, on voit que $u \in \mathscr{C}^\infty(S^1 \times \mathbb{R}_*^+)$. De plus, par dérivation sous le signe intégrale, on voit que pour $t > 0$, u vérifie l'équation aux dérivées partielles de la chaleur.

2. Si $f \in \mathscr{C}^2(S^1)$, on a $|\widehat{f}(n)| = \mathrm{O}\left(\frac{1}{n^2}\right)$ (voir [80] par exemple), et donc par convergence dominée, il vient

$$\|u(t,\cdot) - f\|_\infty \leqslant \sum_{n\in\mathbb{Z}} |\widehat{f}(n)||1 - e^{-2\pi^2 n^2 t}| \xrightarrow[n\to+\infty]{} 0.$$

3. On suppose $u(t_0,x_0) < 0$. Sur $[0,t_0] \times S^1$ qui est compact, v atteint son minimum α en (t_1,x_1). Sur l'axe des x, puisque x_1 est un point intérieur, on a $\frac{\partial u}{\partial x}(t_1,x_1) = 0$ ainsi que $\frac{\partial^2 u}{\partial x^2}(t_1,x_1) \geqslant 0$. Sur l'axe des t, on peut éventuellement avoir $t_1 = t_0$, mais dans tous les cas, on a $\frac{\partial v}{\partial t}(t_1,x_1) \leqslant 0$. On a donc

$$0 \geqslant \frac{\partial v}{\partial t}(t_1,x_1) = \beta v(x_1,t_1) + e^{\beta t}\frac{\partial u}{\partial t}(t_1,x_1) \geqslant \alpha\beta + \frac{1}{2}e^{\beta t}\frac{\partial^2 u}{\partial x^2}(x_1,t_1) \geqslant \alpha\beta,$$

ce qui est absurde si on prend β tel que que $\alpha\beta > 0$.
Si u et $\widetilde{u}$ sont solution du même problème de la chaleur, alors $u - \widetilde{u}$ est solution de l'équation de la chaleur avec pour condition initiale $f = 0$. Par le principe du maximum, il vient $\|u(\cdot,t) - \widetilde{u}(\cdot,t)\|_\infty \leqslant \|f\|_\infty = 0$, donc $u = \widetilde{u}$.

4. Si on n'avait pas $p_t \geqslant 0$, on pourrait trouver un voisinage $]a,b[$ sur lequel $p_t < 0$. On note $x_0 = \frac{1}{2}(a+b)$, et $b - a = 2m$. En choisissant f régulière, à support dans $[-m,m]$, et $f > 0$ sur $]-m,m[$, on a

$$f * p_t(x_0) = \int_a^b p_t(y) f(x_0 - y)\mathrm{d}y < 0,$$

ce qui est absurde d'après la question précédente. Ainsi, il vient

$$\|u(\cdot,t)\|_\infty = \|p_t * f\|_\infty \leqslant \|f\|_\infty \int_0^1 p_t = \|f\|_\infty,$$

puisque $\int_0^1 p_t = \sum e^{-2\pi^2 n^2 t} \int_0^1 e_n = 1$.

5. On a

$$\|u(t,\cdot) - f\|_\infty \leqslant \|p_t * f_n - p_t * f\|_\infty + \|p_t * f_n - f_n\|_\infty + \|f_n - f\|_\infty.$$

Soit $\varepsilon > 0$. On se fixe N tel que $\|f_n - f\|_\infty \leqslant \varepsilon/4$.
On a aussi $\|p_t * f_n - p_t * f\|_\infty \leqslant \|f_n - f\|_\infty \leqslant \varepsilon/4$. Puis, comme f est de classe $\mathscr{C}^2$, on fixe t_0 tel que si $t \leqslant t_0$, on a $\|p_t * f_n - f_n\|_\infty \leqslant \varepsilon/2$.

Correction de l'exercice IV.4 : Les équations sont les mêmes, sauf qu'il faut prendre pour Φ le filtre 3D tel que $\Phi[\pm 1,0,0] = \Phi[0,\pm 10] = \Phi[0,0\pm 1] = 1$, $\Phi[0,0,0] = 6$, et les autres entrées sont nulles. La seule chose difficile à coder est la fonction qui rend la matrice de données antisymétrique. Ceci est réalisé par la procédure MATLAB 4.4.

Programme 4.4 Procédure `antisymetrise`

```
function ff = antisymetrise(f)
n = length(f)+1; ff = zeros(2*n,2*n,2*n);
for(x=1:2*n) for(y=1:2*n) for(z=1:2*n)
  if mod(x-1,n)==0 | mod(y-1,n)==0 | mod(z-1,n)==0
    ff(x,y,z) = 0;
  else
    signe = 1; nx = x; ny = y; nz = z;
    if(x>n) signe = -signe; nx = 2*n-x+2; end
    if(y>n) signe = -signe; ny = 2*n-y+2; end
    if(z>n) signe = -signe; nz = 2*n-z+2; end
    ff(x,y,z) = signe*f(nx-1,ny-1,nz-1);
  end;
end; end; end;
```

Correction de l'exercice IV.5 :

1. L'équation aux différences finies (4.5), chap. IV, s'écrit comme la somme de deux convolutions acycliques, l'une sur les lignes et l'autre sur les colonnes. La multiplication à gauche par T_{N-1} réalise cette convolution sur les lignes, et la multiplication à droite réalise celle sur les lignes.
2. En retranchant les valeurs aux bords (entrées $U_{i,j}$ avec $i,j \in \{1,N-1\}$), on vérifie que les convolutions acycliques évoquées à la question précédente s'écrivent comme des produits matriciels $T_{N-1}\widetilde{U}$ (colonnes) et $\widetilde{U}T_{N-1}$ (lignes).
3. En utilisant les identités trigonométriques, on a, en notant $\omega = \pi/N$, pour $2 \leqslant i \leqslant N-2$,

$$\begin{aligned} h^2(T_{N-1}V_j)[i] &= \sin((i-1)j\omega) - 2\sin(ij\omega) + \sin((i+1)j\omega) \\ &= -4\sin^2\left(\frac{j\pi}{2N}\right) V_j[i]. \end{aligned}$$

On vérifie que ce résultat est encore valable pour $i = 1$ et $i = N-1$. On peut donc diagonaliser T_{N-1}, $V^{-1}T_{N-1}V = D$, avec $D = \left\{-4\sin^2\left(\frac{j\pi}{2N}\right)\right\}_{1\leqslant j\leqslant N-1}$.
En multipliant l'équation (6.2), chap. IV, à gauche par V^{-1} et à droite par V, on obtient l'équation demandée.

4. Comme T_{N-1} est symétrique, ses vecteurs propres sont orthogonaux et donc V est une matrice orthogonale. On note $\mathscr{S} : \mathbb{C}^{N-1} \to \mathbb{C}^{N-1}$ la transformée en sinus, définie par
$$\mathscr{S}(f)[i] = (Vf)[i] = \sum_{j=1}^{N-1} f[j] \sin\left(\frac{ij\pi}{N}\right).$$
Si on note $f_0 = \{0, f[0], \ldots, f[N-1], 0, \ldots, 0\} \in \mathbb{C}^{2N}$, alors, pour $i = 1, \ldots, N-1$, on a $\mathscr{S}(f)[i] = \Im\mathrm{m}(\widehat{f_0}[i])$.
5. On peut aussi considérer $\widetilde{f} = \{0, f[1], \ldots, f[N-1], 0, -f[N-1], \ldots, -f[1]\}$ le signal symétrisé par imparité. On a alors $\mathscr{S}(f)[k] = -2\mathrm{i}\widehat{f_0}[k]$. Comme V est orthogonale et symétrique, on a $\mathscr{S}^{-1} = \frac{2}{N}\mathscr{S}$. La procédure MATLAB 4.5 réalise la transformée en sinus 1D. De même, si $F \in \mathbb{C}^{(N-1)\times(N-1)}$, on note $\widetilde{F}$ la fonc-

Programme 4.5 Procédure `transformee_sinus`

```
function y = transformee_sinus(x)
n = length(x);
x = [0;x;0;-x(n:-1:1)]; x = fft(x);
y = real( x(2:n+1)/(-2i) );
```

tion impaire correspondante (voir le paragraphe 4.3, chap. IV). On a cette fois $VFV^{-1}[k,l] = -4\mathscr{F}(\widetilde{F})[k,l]$. Cette méthode est en fait identique à celle exposée au paragraphe 4.3, chap. IV. La procédure MATLAB 4.6 réalise la transformée en sinus 2D, mais utilise uniquement l'algorithme de transformée 1D (sur les lignes puis les colonnes).

Programme 4.6 Procédure `transformee_sinus_2d`

```
function y = transformee_sinus_2d(x)
y = zeros(size(x)); n = length(x);
for(i=1:n) y(i,:) = transformee_sinus(x(i,:)')'; end;
for(j=1:n) y(:,j) = transformee_sinus(y(:,j)); end;
```

Correction de l'exercice IV.6 : La procédure MATLAB 4.7 calcule un filtre gaussien 2D de taille (paramètre `n`) et de variance (paramètre `s`) données. Le programme 4.8 applique un filtre gaussien à une image chargée depuis un fichier, puis dessine l'image à l'écran.

Programme 4.7 Procédure `calcul_filtre`

```
function f = calcul_filtre(n,s)
x = -1:2/(n-1):1;
[X,Y] = meshgrid(x,x);
f = exp( -(X.^2+Y.^2)/(2*s) );
f = f / sum(sum(f));
```

Correction de l'exercice IV.7 :

1. $d(f,g)[u,v]$ mesure la similarité entre g et une portion de l'image f dont le coin inférieur gauche est situé en (u,v). On a
$$d(f,g)[u,v] = \mathrm{Corr}(f,g)[u,v] + P_{u,v}(f) + \|g\|_2^2.$$
Comme $\|g\|_2^2$ est constante, minimiser $d(f,g)$ revient à minimiser $\mathrm{Corr}(f,g)[u,v]$ si $P_{u,v}(f)$ varie peu.

Programme 4.8 Application d'un filtre gaussien

```
[im, cm] = imread('mettre ici le nom du fichier');
n = length(im); s = 0.01;
f = calcul_filtre(n,s);
y = filter2(f,im);
image(y); colormap(cm);
axis off; axis image;
```

2. $\mathrm{Corr}(f,g)$ est le produit de convolution acyclique de f avec $\widetilde{g}[x,y] = g[-x,-y]$.
3. On note $f_0[x,y] = f[x,y] - \widetilde{f}_{u,v}$ (que l'on suppose nul en dehors de $D(u,v)$), et $g_0[x,y] = g[x,y] - \widetilde{g}$. On a

$$\overline{\mathrm{Corr}}(f,g)[u,v] = \frac{\langle f_0, g_0\rangle}{\|f_0\|_2 \|g_0\|_2}.$$

Ainsi, $-1 \leqslant \overline{\mathrm{Corr}}(f,g) \leqslant 1$, et $\overline{\mathrm{Corr}}(f,g) = 1$ si et seulement si f_0 et g_0 sont égaux. Ceci donne bien une notion de ressemblance entre g et une portion de f, qui de plus est insensible aux modifications affines de l'intensité des deux images. Le problème est que cette quantité ne s'écrit pas comme une convolution.

4. Le numérateur se simplifie en $\sum_{(x,y)} f[x,y] g_0[x-u, y-v]$, puisque g_0 est de moyenne nulle. On obtient bien ainsi une convolution. La relation de récurrence se voit en faisant un dessin, les s_k étant des sommes sur des carrés qui se chevauchent. La procédure MATLAB 4.9 utilise cette récurrence pour remplir, par indices décroissants, s_k. Sa complexité est d'environ $6N^2$ opérations. On a $\|f_0\|_2^2 = s_2(u,v) - \frac{1}{P^2} s_1(u,v)^2$,

Programme 4.9 Procédure `somme_glissante`

```
function y = somme_glissante(x,P,k)
N = length(x); y =zeros(N,N);
for(u=N:-1:1) for(v=N:-1:1)
  if( u<N ) y(u,v)=y(u,v)+y(u+1,v); end;
  if( v<N ) y(u,v)=y(u,v)+y(u,v+1); end;
  if( u<N && v<N ) y(u,v)=y(u,v)-y(u+1,v+1); end;
  y(u,v)=y(u,v)+x(u,v)^k;
  if( u+P<=N ) y(u,v)=y(u,v)-x(u+P,v)^k; end;
  if( v+P<=N ) y(u,v)=y(u,v)-x(u,v+P)^k; end;
  if( u+P<=N && v+P<=N ) y(u,v)=y(u,v)+x(u+P,v+P)^k; end;
end; end;
```

ce qui peut se calculer avec des sommes glissantes, donc très rapidement. De plus, $\|g\|_2$ se calcule une fois pour toutes. La procédure 4.10 calcule $\overline{\mathrm{Corr}}(f,g)$ à l'aide de cet algorithme rapide.

Correction de l'exercice IV.8 :

1. On a

$$\mathscr{F}(T_v(f))[k,l] = \omega_N^{-(kv_1+lv_2)} \mathscr{F}(f)[k,l] \qquad \text{où} \quad \omega_n = e^{\frac{2i\pi}{N}}.$$

La procédure MATLAB 4.11 utilise l'algorithme FFT pour translater une image. La procédure 4.12 applique l'opérateur $S_\lambda^{(x)}$ en effectuant une translation sur chaque ligne de l'image. Il est à noter que les indices des entrées de l'image sont prises entre $-N/2$ et $N/2-1$, dans le but d'appliquer $S_\lambda^{(x)}$ pour tourner une image autour de son centre. Nous laissons le soin au lecteur d'écrire lui même la procédure `fft_transvec_y`.

Programme 4.10 Procédure `correlation_normalisee`

```
function y = correlation_normalisee(f,g)
N = length(f); P = length(g);
% renormalisation
g = g - mean(mean(g)); g = g/norm(g);
% calcul du numérateur
ff = zeros(N+P-1,N+P-1); ff(1:N,1:N) = f;
gg = zeros(N+P-1,N+P-1); gg(1:P,1:P) = g;
fg = real(ifft2( fft2(ff).*conj(fft2(gg)) ));
fg = fg(1:N,1:N);
% calcul du dénominateur
s1 = somme_glissante(f,P,1);
s2 = somme_glissante(f,P,2);
denom = sqrt( s2-1/P^2*(s1.^2) );
y = fg./denom;
```

Programme 4.11 Procédure `fft_translation`

```
function y = fft_translation(x, v)
n = length(x);
[s,t] = meshgrid( [0:(n/2-1),-n/2:-1] );
mult = exp( -2i*pi/n*( s*v(1) + t*v(2) ) );
y = fft2(x).*mult;
y = real( ifft2(y) );
```

Programme 4.12 Procédure `fft_transvec_x`

```
function y = fft_transvec_x(x, lambda)
n = length(x);
for k=1:n
  v = x(:,k); trans = lambda*(k-n/2-1);
  mult = exp( -2i*pi/n*( [0:(n/2-1),-n/2:-1]'*trans ) );
  v = fft(v).*mult; y(:,k) = real( ifft(v) );
end
```

2. On a

$$\begin{pmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \sin(\theta)\end{pmatrix} = \begin{pmatrix}1 & -\tan(\theta/2)\\ 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0\\ \sin(\theta) & 1\end{pmatrix}\begin{pmatrix}1 & -\tan(\theta/2)\\ 0 & 1\end{pmatrix}.$$

   Si on suppose qu'une image correspond à une fonction de $\mathbb{R}^2$ dans $\mathbb{R}$ discrétisée aux points $0,\ldots,N-1$, alors l'opérateur de rotation discrète s'écrit $R_\theta = S^{(x)}_{\lambda_1} S^{(y)}_{\lambda_2} S^{(x)}_{\lambda_3}$.
3. Pour faire tourner une image autour de son centre, il suffit d'utiliser, dans les algorithmes de calcul de $S^{(x)}_\lambda$ et $S^{(y)}_\lambda$, des points de discrétisation $-N/2,\ldots,N/2-1$ (c'est ce qu'on a fait pour `fft_transvec_x`). La procédure 4.13 permet de réaliser une rotation discrète. Cet algorithme est très rapide, puisqu'il nécessite

Programme 4.13 Procédure `fft_rotation`

```
function y = fft_rotation(x, theta)
y = fft_transvec_x( x,-tan(theta/2) );
y = fft_transvec_y( y,sin(theta) );
y = fft_transvec_x( y,-tan(theta/2) );
```

$O(N^2\log(N))$ opérations. De plus, il est bijectif, et vérifie $R_{\theta_1}R_{\theta_2} = R_{\theta_1+\theta_2}$. En quelque sorte, le passage par la TFD est la façon « naturelle » de discrétiser une

rotation continue. Par contre, puisque l'image est considérée comme une fonction continue, l'image se retrouve « découpée en morceaux » après la rotation. Pour empêcher ceci, il faut ajouter des zéros tout autour. C'est ce que réalise le programme 4.14, qui a permis de réaliser la figure 4.15, chap. IV. On peut noter que ce programme ajoute des 255 autour de l'image de départ. C'est pour obtenir un bord blanc plutôt que noir.

Programme 4.14 Rotation d'une image

```
[im, cm] = imread('mettre ici le nom du fichier');
n = length(im); p = ceil(n/2*(sqrt(2)-1));
x = ones(n+2*p,n+2*p)*255;
x((p+1):(n+p),(p+1):(n+p)) = im;
nbr = 6; rot = pi/(4*(nbr-1));
for r = 0:nbr-1
  y = fft_rotation(x,r*rot);
  subplot(1,nbr,r+1);
  image(y); colormap(cm);
  axis off; axis image;
end
```

Correction de l'exercice IV.9 :

1. La procédure MATLAB 4.15 réalise ceci.

Programme 4.15 Procédure `filtre_passe_bas`

```
function f = filtre_passe_bas(N)
f = (0:N-1)'; f = (f<=N/4)|(f>=3*N/4);
f = real(ifft(f));
```

2. Le programme MATLAB 4.16 dessine la transformée de Fourier d'un filtre, en ajoutant des zéros. On constate des oscillations (phénomène de Gibbs), car on essaie

Programme 4.16 Dessin de la transformée de Fourier continue par zero padding

```
N = 64; P = 1024;
f = filtre_passe_bas(N);
ff = [f(1:N/2); zeros(P-N,1); f((N/2+1):N)];
ff = real(fft(ff));
plot(ff); axis tight;
```

d'approcher une fonction discontinue (le filtre idéal est la fonction indicatrice de $[-\pi/2, \pi/2]$) par un polynôme trigonométrique.

3. Nous allons remplacer le passage brutal de 0 à 1 dans le filtre de la question 1 par une progression douce (sinusoïdale) de longueur $\varepsilon N/2$. La procédure 4.17 réalise ceci.

Correction de l'exercice IV.10 :

1. A la $k^{\text{ième}}$ itération du procédé, on note m_k le nombre minimal de bonbons qu'un enfant possède, n_k le nombre maximal, et s_k le nombre d'occurrences de m_k. Il est facile de voir que $m_{k+1} \geqslant m_k$ et $n_{k+1} \leqslant n_k$. De plus, on voit aussi que si $s_k > 1$, alors $s_{k+1} < s_k$, et si $s_k = 1$, alors $m_{k+1} > m_k$. Comme s_k est borné par n, toutes les n opérations, m_k augmente d'au moins 1, et donc $|m_k - n_k|$ vaut 0 après un nombre fini d'opérations.

Programme 4.17 Procédure `filtre_parametrable`

```
function f = filtre_parametrable(N,eps)
P1 = floor(eps*N/4); P = 2*P1+1; % P doit être impair
t = [1]; if(P~=1) t = (cos((0:P-1)'*pi/(P-1))+1)/2; end;
f = [ones(N/4-P1,1);t;zeros(N/2-P,1);t(P:-1:1);ones(N/4-P1-1,1)];
f = real(ifft(f));
```

2. On peut assimiler la distribution des bonbons à une distribution de probabilité (en la renormalisant pour que sa somme soit égale à 1), et on se trouve donc dans la situation de l'exercice I.10. Si on note $p^{(k)}$ la distribution des bonbons après k itérations, on a $p^{(k)} = v * \cdots * v * p^{(0)}$, avec $v = \{1/2, 1/2, 0, \ldots, 0\}$. En prenant la transformée de Fourier, on obtient $\widehat{p^{(k)}} = (\widehat{v})^k \cdot \widehat{p^{(0)}}$. On calcule $\widehat{v}[k] = \frac{1}{2}(1 + e^{-\frac{2ik\pi}{N}})$, et on constate que pour $k \neq 0$, $|\widehat{v}[k]| < 1$. Donc $p^{(k)} \to \{m, \ldots, m\}$, où $m = \widehat{p^{(0)}}[0]$ (le nombre moyen de bonbons).
3. Dans le cas où chaque enfant distribue la moitié à gauche et la moitié à droite, il n'y a pas de convergence, car on peut avoir $m_{k+1} = m_k$. La situation est analogue à celle rencontrée à l'exercice I.10 pour $v = \{0, 1/2, 0, \ldots, 0, 1/2\}$ lorsque n est pair.

Correction de l'exercice IV.11 :

1. Il faut prendre $R_0 = P_0 Q_0$, $R_1 = P_1 Q_0 + P_0 Q_1$ et $R_2 = P_1 Q_1$.
2. On peut effectuer le calcul astucieux suivant : $R_1 = (P_0 + P_1)(Q_0 + Q_1) - R_0 - R_2$.
3. En tout, il y a $\log_2(n)$ appels récursifs imbriqués. A chaque étape, il y a 3 appels récursifs, donc pour $k = 0, \ldots, \log_2(n)$, il y a 3^k appels au total. Le coût des additions à chaque étape est de $c \times 2^{\log_2(n)-k}$ (puisque les polynômes sont de degré $2^{\log_2(n)-k}$). Au total, le nombre d'opérations est de

$$\sum_{k=0}^{\log_2(n)} 3^k c 2^{\log_2(n)-k} = \mathrm{O}\left(n(3/2)^{\log_2(n)}\right) = \mathrm{O}\left(n^{\log_2(3)}\right).$$

La procédure MATLAB 4.18 calcule le produit de deux polynômes de même taille (représentés sous forme de vecteurs). Il utilise la procédure 4.19 qui additionne deux polynômes de degrés différents.

Programme 4.18 Procédure `karatsuba`

```
function r = karatsuba(p,q)
n = length(p)-1;
if(n==0) r=p*q; return; end;
k = floor((n+1)/2);
p0 = p(1:k); p1 = p((k+1):(n+1));
q0 = q(1:k); q1 = q((k+1):(n+1));
r0 = karatsuba(p0,q0); r2 = karatsuba(p1,q1);
r1 = karatsuba(add(p0,p1),add(q0,q1));
r1 = add(r1,-r0); r1 = add(r1,-r2);
r = add( r0, [zeros(k,1);r1] );
r = add( r, [zeros(2*k,1);r2] );
```

Programme 4.19 Procédure `add`

```
function r = add(p,q)
m = length(p); n = length(q);
if(m>=n) r = p + [q;zeros(m-n,1)];
else r = q + [p;zeros(n-m,1)]; end;
```

Correction de l'exercice IV.12 :

1. L'équation $v(k) = u_d[k]$ s'écrit $u_d = a * \psi_d$. En utilisant le théorème de convolution pour les séries de Fourier (voir [80]), on obtient $\widehat{u_d} = \widehat{a} \cdot \widehat{\psi_d}$. En remplaçant ψ par la fonction φ telle que

$$\varphi(x) = \sum_{k\in\mathbb{Z}} \Phi_d(k)\psi(x-k) \quad \text{avec} \quad \widehat{\Phi_d} = \frac{1}{\widehat{\psi_d}},$$

on se ramène à un problème d'interpolation directe. Même si ψ est à support compact, φ n'est en général pas à support compact.

2. β^n est à support dans $[(-n+1)/2, (n+1)/2]$. β^n permet de définir un schéma d'interpolation directe si et seulement si $\beta^n(k) = \delta_0(k)$, et on vérifie que ceci est le cas seulement pour $n = 0$ (interpolation constante par morceaux) ou $n = 1$ (interpolation linéaire).

3. Avec le théorème de convolution de la transformée de Fourier, il vient

$$\widehat{\beta^n}(\xi) = \left(\frac{\sin(\xi/2)}{\xi/2}\right)^{n+1}$$

On a $\beta_d^n = \beta^n \cdot \Pi_1$, d'où, avec le résultat de l'exercice II.9, il vient

$$\widehat{\beta_d^n}(\xi) = 2\pi\widehat{\beta^n} * \Pi_{2\pi}(\xi) = 2\pi \sum_{p\in\mathbb{Z}} (-1)^{p(n+1)} \left(\frac{\sin(\xi/2)}{\pi p + \xi/2}\right)^{n+1}.$$

Comme $\widehat{\beta_d^n}(\xi)$ est une fonction 2π-périodique, il suffit de l'étudier sur $[0, 2\pi[$. Si n est impair, tous les termes de la somme sont positifs, et $\widehat{\beta_d^n} > 0$. Si n est pair, c'est un peu plus compliqué. Le terme pour $p = 0$ de la somme est strictement positif. Pour le reste, il faut regrouper les termes correspondant à p et à $-p$ et utiliser le critère des séries alternées.

4. On a $\beta_{card}^n = \Phi_d^n * \beta^n$. En Fourier, on obtient

$$\widehat{\beta_{card}^n}(\xi) = \frac{\left(\frac{\sin(\xi/2)}{\xi/2}\right)^{n+1}}{2\pi\sum_{p\in\mathbb{Z}} \left(\frac{\sin(\xi/2)}{\pi p+\xi/2}\right)^{n+1}} = \frac{1/(2\pi)}{1+\Gamma_n(\xi)},$$

avec

$$\Gamma_n(\xi) = \sum_{p=1}^{+\infty} \left[\left(\frac{2\pi p}{\xi} + 1\right)^{-n-1} + \left(\frac{2\pi p}{\xi} - 1\right)^{n-1}\right].$$

Γ_n est de support infini. Comme la transformée de Fourier est une isométrie, on veut montrer la convergence dans $L^2(\mathbb{R})$:

$$\widehat{\beta_{card}^n} \xrightarrow[n\to\infty]{} \frac{1}{2\pi}\mathbf{1}_{[-\pi,\pi]}.$$

Par parité, nous allons évaluer seulement les intégrales

$$E_n = \int_0^{\pi} \left(\frac{1}{1+\Gamma_n(\xi)} - 1 \right)^2 \mathrm{d}\xi \qquad \text{et} \qquad F_n = \int_{\pi}^{+\infty} \left(\frac{1}{1+\Gamma_n(\xi)} \right)^2 \mathrm{d}\xi.$$

Pour la première, on a $E_n \leqslant \int_0^{\pi} \Gamma_n^2$, et on utilise le fait que si $\xi \in [0,\pi]$, alors on a $t = 2\pi/\xi > 2$, d'où la majoration

$$\Gamma_n(\xi) \leqslant 2\left(\frac{2\pi}{\xi} - 1 \right)^{-n-1} \sum_{p=1}^{+\infty} p^{-n-1} \leqslant 4(t-1)^{-n-1}.$$

Par changement de variable $\xi \to t$, on obtient

$$E_n \leqslant \int_2^{+\infty} \left[4(t-1)^{-n-1} \frac{2\pi}{t^2} \right]^2 \mathrm{d}t = \mathrm{O}\left(\frac{1}{n} \right).$$

Pour la deuxième intégrale, on utilise la minoration $\Gamma_n(\xi) \geqslant (\xi/\pi)^{n+1}$, d'où

$$F_n \leqslant \int_{\pi}^{+\infty} \left[\frac{1}{1+(\xi/\pi)^{n+1}} \right]^2 \mathrm{d}\xi = \mathrm{O}\left(\frac{1}{n} \right).$$

Au final, on voit que l'interpolation trigonométrique (c'est-à-dire par zero padding, aussi appelée interpolation de Shannon, comme le montre l'exercice II.10) peut être vue comme une interpolation spline de degré infini.

5. On a $c = \Phi_d * u_d$, ce qui s'écrit en Fourier $\widehat{c} = \widehat{u_d}/\widehat{\beta_d^n}$. Si l'on approche la transformée de Fourier continue par une transformée de Fourier discrète, et que l'on calcule $\widehat{c}$ par FFT, on va tronquer les fonctions considérées, ce qui est très mauvais (oscillations de Gibbs)

5 Correction des exercices du chapitre 5

Correction de l'exercice V.1 : On a $\mathscr{H}^2 = N\mathrm{Id}$, donc les valeurs propres de $\mathscr{H}$ sont incluses dans $\{\pm\sqrt{N}\}$. Pour $f \in \mathbb{R}^N$, on définit

$$\mathscr{U}_+(f) \overset{\text{déf.}}{=} \sqrt{N} f + \mathscr{H}(f) \quad \text{et} \quad \mathscr{U}_+(f) \overset{\text{déf.}}{=} \sqrt{N} f - \mathscr{H}(f).$$

On vérifie que ce sont bien des vecteurs propres de $\mathscr{H}$ associés aux valeurs propres $\sqrt{N}$ et $-\sqrt{N}$. La procédure MATLAB 5.1 calcule, à l'aide de ces vecteurs propres, une transformée de Hartley intermédiaire. On fera attention au fait qu'elle est à valeur dans $\mathbb{C}$.

Programme 5.1 Procédure `fht_interm`

```
function y = fht_interm(x,lambda)
N = length(x);
u1 = sqrt(N)*x+fht(x); u2 = sqrt(N)*x-fht(x);
y = ( sqrt(N)^lambda )*( u1 + (-1)^lambda*u2 );
```

Correction de l'exercice V.2 : On note $\alpha = 2\pi/N$, et $v_k^{\lambda} \in \mathbb{R}^N$ tel que $v_k^{\lambda}[l] = \cos(\alpha kl + \lambda)$. La transformée de Hartley généralisée s'écrit $\mathscr{H}_{\lambda}(f)[k] = \left\langle f, v_k^{\lambda} \right\rangle$. Pour définir une

transformée inverse, on cherche une famille de vecteurs biorthogonale aux v_k^λ sous la forme $\{v_k^{\lambda'}\}_{k=0}^{N-1}$. On veut donc que $\left\langle v_k^\lambda, v_{k'}^{\lambda'} \right\rangle = \gamma \delta_k^{k'}$, où $\gamma \in \mathbb{C}$ est une constante. En développant le produit scalaire, on obtient une expression similaire à celle de l'équation (1.3), chap. V. Pour que les deux premiers termes se simplifient, on impose $e^{i(\lambda+\lambda')} = -e^{-i(\lambda+\lambda')}$, par exemple $\lambda + \lambda' = \pi/2$. On obtient alors $\left\langle v_k^\lambda, v_{k'}^{\lambda'} \right\rangle = \frac{1}{2}\sin(2\lambda)\delta_k^{k'}$, et donc au final il vient $\mathscr{H}^\lambda \circ \mathscr{H}^{\lambda'} = \frac{1}{2}\sin(2\lambda)\mathrm{Id}$.

Correction de l'exercice V.3 : Soit $f \in \mathbb{R}^N$ avec $N = 2N_0 + 1$ et $g = \mathscr{H}(f)$. On note $\widetilde{g} = \{g[0], \ldots, g[N_0], 0, \ldots, 0, g[N_0+1], \ldots, g[N-1]\}$. Ceci permet de calculer, par FHT, les quantités $\mathscr{H}(\widetilde{g})[n] = F(nN/P)$, où l'on a noté $F(x) = \frac{1}{N}\sum_{k=0}^{N-1} g[k]\,\mathrm{cas}(2\pi kx/N)$. On peut donc évaluer la fonction F avec la précision que l'on veut. Avec la formule d'inversion, proposition 1.3, chap. V, F interpole f aux points $0, \ldots, N-1$. En exprimant les fonctions $x \mapsto \mathrm{cas}(2\pi kx)$ à l'aide des exponentielles complexes $x \mapsto e^{\frac{2ikx\pi}{N}}$, on voit que F est un polynômes trigonométrique de degré au plus $N-1$. L'interpolation trigonométrique de l'exercice III.6 utilise aussi un polynôme trigonométrique de degré au plus $N-1$. Comme il passe un seul polynôme trigonométrique de degré au plus N par N points distincts, on en déduit que ces deux interpolations sont les mêmes.

Correction de l'exercice V.4 : On peut écrire la transformée de Hartley 1D comme $\mathscr{H}(f)[k] = \left\langle f, \varphi_k^{(N_1)} \right\rangle$ avec $\varphi_k^{(N_1)}[n] = \mathrm{cas}(nk2\pi/N_1)$. La proposition 1.3, chap. V nous dit que $\left\langle \varphi_k^{(N_1)}, \varphi_{k'}^{(N_1)} \right\rangle = N_1 \delta_k^{k'}$. Les fonctions de Hartley 2D sont des produits tensoriels $\varphi_{(k_1,k_2)}^{(N_1,N_2)}[n_1,n_2] = \varphi_{k_1}^{(N_1)}(n_1)\varphi_{k_2}^{(N_2)}(n_2)$. Elles forment donc encore un système orthogonal, puisque

$$\left\langle \varphi_{(k_1,k_2)}^{(N_1,N_2)}, \varphi_{(k_1',k_2')}^{(N_1,N_2)} \right\rangle = \left\langle \varphi_{k_1}^{(N_1)}, \varphi_{k_1'}^{(N_1)} \right\rangle \left\langle \varphi_{k_2}^{(N_2)}, \varphi_{k_2'}^{(N_2)} \right\rangle = N_1 N_2 \delta_{k_1}^{k_1'} \delta_{k_2}^{k_2'},$$

d'où la formule d'inversion proposée. L'algorithme de calcul consiste à appliquer la procédure FHT sur chaque ligne de la matrice $f \in \mathbb{R}^{N_1 \times N_2}$, puis sur chaque colonne. C'est ce que réalise la procédure 5.2.

Programme 5.2 Procédure `fht2d`

```
function y = fht2d(x)
n = length(x); y = zeros(n,n);
for(i=1:n) y(i,:) = fht(x(i,:)')'; end;
for(j=1:n) y(:,j) = fht(y(:,j)); end;
```

Correction de l'exercice V.5 : Soit p un nombre premier tel que $4|p-1$. On note ζ un générateur de $\mathbb{F}_p^*$, et $\gamma = \zeta^{\frac{p-1}{4}}$. Dans l'équation (1.1), chap. V, on remplace $\mathrm{cas}(2kl\pi/N)$ par $\frac{1}{2}(\zeta^{kl} + \zeta^{-kl}) + \frac{1}{2\gamma}(\zeta^{kl} - \zeta^{-kl})$. La démonstration de la proposition 1.3, chap. V est encore valide si on remplace ω par ζ et i par γ. Le programme MAPLE 5.3 réalise une transformée de Hartley sur un corps fini en utilisant une extension de corps, comme expliqué au paragraphe 1.4, chap. VI pour la TFD.

Correction de l'exercice V.6 :

1. Il faut prendre $G_{i,j} = g[i-j] = g[j-i]$. La multiplication par G correspond bien à la convolution acyclique qui définit y.

Programme 5.3 Transformée de Hartley sur un corps cyclotomique

```
> with (numtheory): n := 16: p := 5:
> liste_div := op(Factor( cyclotomic(n,X) ) mod p ):
> P := liste_div[1];
> alias ( zeta = RootOf(P) ): # racine nième primitive
> g := 2: # 2 est une racine carrée de -1 modulo 5
```

$$P := X^4 + 3$$

```
> cas := proc(x)
> 1/2*( zeta^(x)*(1-g) + zeta^(-x)*(1+g) );
> end proc:
```

Transformée de Hartley, version $O(n^2)$:

```
> Hartley := proc(f)
>    local res, formule;
>    formule := 'f[1+1]*cas((k-1)*1)';
>    res := [ seq( sum( formule, 'l'=0..n-1 ) mod p , k=1..n)  ];
>    return (Normal(res) mod p);
> end proc:
```

Test simple :

```
> hasard := rand(0..(p-1)):
> x := [seq( hasard(), i=1..n )];
> y := simplify(Hartley(x));  # Hartley(x) n'est plus dans F_p.
> evalb( x = Hartley(y)/n mod p ); # doit être égal.
```

$$x := [0,1,1,1,2,2,1,4,0,1,2,0,1,0,2,0]$$

$$y := [3,3\zeta^2+4+4\zeta+3\zeta^3,2+4\zeta^2,1+4\zeta^3+3\zeta^2,3,2\zeta^2+2\zeta+\zeta^3+4,2,$$
$$2\zeta^3+2\zeta^2+1,0,\zeta+2\zeta^3+3\zeta^2+4,2+\zeta^2,\zeta^3+3\zeta^2+1,1,3\zeta+4\zeta^3+2\zeta^2+4,$$
$$2,2\zeta^2+3\zeta^3+1]$$

true

2. La matrice T est le bloc constitué des m premières lignes et des n premières colonnes. Pour calculer Tx, on calcule $\widetilde{y} = C\widetilde{x}$ où $\widetilde{x} = (x,0,\ldots,0)^{\mathrm{T}} \in \mathbb{C}^M$, et on extrait de $\widetilde{y}$ les m premières composantes pour trouver y. On a $\widetilde{y} = c * \widetilde{x}$, ce qui se calcule rapidement par FFT.
3. Il faut donc considérer $c = \{g[0],\ldots,g[N-1],0,\ldots,0,g[N-1],\ldots,g[1]\} \in \mathbb{C}^M$. La procédure MATLAB 5.4 réalise le calcul de transformée en Z vectorielle par l'algorithme CZT.

Programme 5.4 Procédure `czt`

```
function y = czt(x,z)
n = length(x);
g = z.^(1/2*(0:n-1)'.^2); h = x./g;
k = ceil(log2(2*n-1)); M = 2^k;
g = [g; zeros(M-2*n+1,1); g(n:-1:2)];
h = [h; zeros(M-n,1)];
y = ifft( fft(g).*fft(h) );
y = y(1:n)./g(1:n);
```

Correction de l'exercice V.7 : On note $x_n = f(n)$ le signal discrétisé, et $y_n^{(i)} \approx \int_0^n f(t)\mathrm{d}t$ le résultat obtenu avec la méthode (M_i). Soit $X = \mathscr{Z}(x_n)$ et $Y^{(i)} = \mathscr{Z}(y_n^{(i)})$ les transformées en Z. On note $H^{(i)} \overset{\text{déf.}}{=} Y^{(i)}/X$ les fonctions de transfert. On a

$$H^{(1)}(z) = \frac{1}{z-1}, \quad H^{(2)}(z) = \frac{1}{2}\frac{z+1}{z-1}, \quad H^{(3)}(z) = \frac{1}{2}\frac{1+4z+z^2}{z^2-1}.$$

La figure 8.1 montre les réponses fréquentielles de ces trois filtres. On constate qu'ils amplifient beaucoup les basses fréquences (ils ne sont pas stables), ce qui est normal, puisque l'on a réalisé des intégrateurs, qui lissent le signal d'entrée.

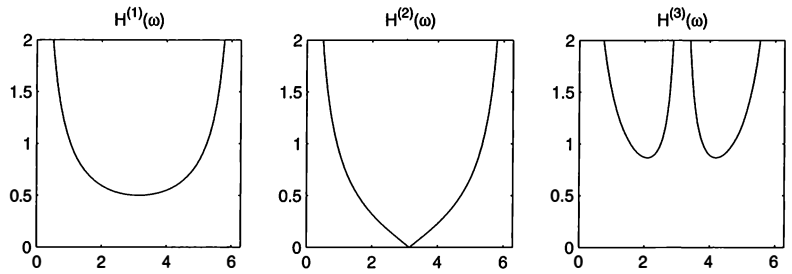

FIG. 8.1 – *Réponses fréquentielles des trois filtres*

Correction de l'exercice V.8 :

1. On a
$$\beta^3(x) = \begin{cases} 2/3 - |x|^2 + |x|^3/2 & \text{si } 0 \leqslant |x| \leqslant 1, \\ (2-|x|)^3/6 & \text{si } 1 \leqslant |x| \leqslant 2, \\ 0 & \text{sinon,} \end{cases}$$
ce qui donne $\beta_d^3 = \{\ldots, 0, 1/6, 2/3, 1/6, 0, \ldots\}$.
2. On a la décomposition
$$\mathscr{Z}(\Phi_d^3)(z) = \frac{-6\alpha}{1-\alpha^2}\left(\frac{1}{1-\alpha z^{-1}} + \frac{1}{1+\alpha z} - 1\right) \quad \text{avec} \quad \alpha = \sqrt{3}-2.$$
La fraction en z^{-1} (respectivement en z) correspond à un filtre récursif causal (respectivement anti-causal) qui est stable. Pour calculer c, il faut filtrer u_d par les deux filtres, (l'un selon les indices croissants et l'autre selon les indices décroissants), ajouter les résultats, soustraire u_d, et multiplier le tout par $-6\alpha/(1-\alpha^2)$.
3. Avec la question précédente, il vient $b_0 = -6\alpha/(1-\alpha^2)$ et $b_1 = \alpha$. On peut imposer $c^+[0] = c^-[K-1] = 0$. Pour des conditions plus complexes, on regardera [74]. La procédure MATLAB 5.5 calcul les coefficients de l'interpolation par cette méthode.

Programme 5.5 Procédure `coef_spline_1`

```
function c = coef_spline_1(ud)
K = length(ud); alpha = sqrt(3)-2;
b1 = alpha; b0=-6*b1/(1-b1^2);
c1 = zeros(K,1); c2 = zeros(K,1);
for i=2:K
  c1(i) = ud(i)+b1*c1(i-1);
end
for i=(K-1):-1:1
  c2(i) = ud(i)+b1*c2(i+1);
```

4. On a la décomposition

$$\mathscr{Z}(\Phi_d^3)(z) = \frac{-6\alpha}{(1-\alpha z^{-1})(1-\alpha z)}.$$

Le vecteur c s'obtient donc par la composition de deux filtres (l'un causal, l'autre anti-causal). Le programme MATLAB 5.6 utilise cette deuxième décomposition.

Programme 5.6 Procédure `coef_spline_2`

```
function c = coef_spline_2(ud)
K = length(ud); alpha = sqrt(3)-2;
c = zeros(K,1); d = zeros(K,1);
for i=2:K
  d(i) = 6*ud(i)-alpha*d(i-1);
end
c(K) = -6*alpha/(1-alpha^2)*(2*d(K)-6*ud(K));
for i=(K-1):-1:1
  c(i) = alpha*( c(i+1)-d(i+1) );
end
```

Correction de l'exercice V.9 : En utilisant le fait que $a \mapsto g^a$ est une bijection de $\{0,\dots,p-2\}$ dans $\mathbb{F}_p^*$, on obtient

$$\widehat{f}(g^{-b}) = f(0) + \sum_{x\in\mathbb{F}_p^*} f(x)\omega_p^{-xg^{-b}} = f(0) + \sum_{a=0}^{p-2} f(g^a)\omega_p^{-g^{a-b}}.$$

On note $\widetilde{f}$ et $\widetilde{h}$ les vecteurs de $\mathbb{C}^{p-1}$ définis par $\widetilde{f}[k] = f(g^k)$ et $\widetilde{h}[k] = \omega_p^{-g^{-k}}$. On vérifie que la définition de $\widetilde{h}$ est indépendante d'une translation de k par $p-1$, donc $\widetilde{h}$ peut être vu comme une fonction $(p-1)$-périodique. En conséquence, l'expression de $\widehat{f}(g^{-b})$ correspond bien à la convolution circulaire $\widetilde{f} * \widetilde{h}[b]$. On peut donc calculer une TFD de longueur p grâce à une convolution de longueur $p-1$, donc à 3 TFD de longueur $p-1$. Ceci est avantageux car p n'admet pas de factorisation, alors que $p-1$ en admet une (il est au moins divisible par 2), ce qui permet d'utiliser par exemple la méthode de Cooley-Tukey, paragraphe 2.4, chap. III.
La procédure MATLAB 5.7 utilise cette méthode. Il faut lui fournir un générateur de $\mathbb{F}_p^*$ dans le paramètre `g`. Elle utilise une fonction auxiliaire `invmod`, programme 5.8 qui calcule un inverse modulo p.

Correction de l'exercice V.10 :

1. On a

$$\widehat{f}(y_k) \approx \frac{a}{N}\sum_{s=0}^{N-1} f(x_s)e^{-\mathrm{i}x_s y_k} = \frac{a}{N}e^{\mathrm{i}\delta}\sum_{s=0}^{N-1}\widetilde{f}[s]e^{-\frac{2\mathrm{i}\pi}{N}sk\gamma} = \frac{a}{N}e^{\mathrm{i}\delta}G(\widetilde{f},\gamma)[k],$$

où $\delta = \frac{N}{2}\left(\zeta - \frac{2\pi}{a}\gamma\frac{N}{2}\right)$ et $\widetilde{f}[s] = f[s]e^{-\mathrm{i}\zeta s}$. Si $\zeta = \frac{p}{q}$, on a

$$G(\widetilde{f},\gamma) = \sum_{k=0}^{N-1}\widetilde{f}[s]e^{-\frac{2\mathrm{i}\pi}{qN}s(pk)}.$$

Ceci peut se calculer en ajoutant $N(q-1)$ zéros à la fin de f, puis en calculant une TFD de taille Nq. L'utilisation de la transformée de Fourier fractionnaire est très avantageuse si q est grand.

Programme 5.7 Procédure `fft_chirp`

```
function y = fft_chirp(x,g)
p = length(x); f = zeros(p-1,1); h = zeros(p-1,1);
for k=0:p-2
  j = mod(g^k,p); jj = invmod(j,p);
  f(k+1) = x(j+1); h(k+1) = exp(-2i*pi/p*jj);
end
h = ifft(fft(f).*fft(h));
y = zeros(p,1); y(1) = sum(x);
for k=0:p-2
  j = mod(g^k,p); jj = invmod(j,p);
  y(jj+1) = x(1) + h(k+1);
end
```

Programme 5.8 Procédure `invmod`

```
function y = invmod(x,p)
[u,y,d] = gcd(x,p); y = mod(y,p);
```

2. Nous allons définir une transformée de Fourier fractionnaire G_{sim} qui utilise la méthode de Simpson à la place de la méthode des rectangles pour l'intégrale de Fourier. On suppose que $N = 2N_0 + 1$, et en faisant attention à bien découper les sommes, on obtient

$$G_{\text{sim}}(f,\gamma)[k] = \frac{1}{3}\sum_{s=0}^{N_0-1} f[2s]\omega_N^{-2sk\gamma} + \frac{4}{3}\sum_{s=0}^{N_0-1} f[2s+1]\omega_N^{-(2s+1)k\gamma} + \frac{1}{3}\sum_{s=1}^{N_0} f[2s]\omega_N^{-2sk\gamma} = G(g,\gamma),$$

où g est défini par $g[0] = \frac{1}{3}f[0]$, $g[N-1] = \frac{1}{3}f[N-1]$ et

$$\begin{cases} g[2k] = \frac{2}{3}f[2k] & \text{pour } k = 1,\ldots,N_0-1, \\ g[2k+1] = \frac{4}{3}f[2k+1] & \text{pour } k = 0,\ldots,N_0-1. \end{cases}$$

La procédure MATLAB 5.9 utilise cette méthode. Pour calculer $G_{\text{sim}}(f,\gamma)$, il faut l'appeler avec le paramètre `alpha` égal à $(\omega_N)^\gamma$.

Programme 5.9 Procédure `czt_simpson`

```
function y = czt_simpson(x,alpha)
N = length(x); N0 = (N-1)/2; y = zeros(N,1);
y(1) = x(1)/3; y(N) = x(N)/3;
y( 2*(1:(N0-1))+1 ) = 2/3*x( 2*(1:(N0-1))+1 );
y( 2*(0:(N0-1))+2 ) = 4/3*x( 2*(0:(N0-1))+2 );
y = czt(y,alpha);
```

Correction de l'exercice V.11 : La procédure MATLAB 5.10 réalise la transformée de Fourier fractionnaire $G(f,\alpha)$, tout simplement en utilisant l'algorithme `czt` (procédure 5.4). On peut ensuite l'appliquer sur les lignes puis les colones d'une matrice pour calculer une transformée 2D, comme le fait la procédure 5.11.

Programme 5.10 Procédure `frft`

```
function y = frft(x, alpha)
N = length(x);
w = exp( -2i*pi*alpha/N );
y = czt(x,w);
```

Programme 5.11 Procédure `frft2d`

```
function y = frft2d(x,alpha)
n = length(x);
for i=1:n
  y(i,:) = frft( x(i,:), alpha );
end
for j=1:n
  y(:,j) = frft( y(:,j), alpha );
end
```

6 Correction des exercices du chapitre 6

Correction de l'exercice VI.1 : La procédure MAPLE 6.1 calcule le premier entier tel que Φ_n ait un coefficient égal à $\pm k$. Attention, elle est très lente.

Programme 6.1 Procédure `CycloCoef`

```
CycloCoef := proc(k)
  local i,j,P,s:
  for i from 0 to 10000 do
    P := cyclotomic(i,X): s := degree(P):
    for j from 0 to s do
      if abs(coeff(P,X,j))=k then return(i): end if:
    end do:
  end do:
end proc:
```

Correction de l'exercice VI.2 :

1. Une racine principale ζ sur $\mathbb{F}_p$ est une racine primitive. Le groupe engendré par ζ est de cardinal n, c'est un sous-groupe de $\mathbb{F}_p^*$, donc $n|p-1$.
2. On a $\text{pgcd}(\zeta, p) = 1$, donc $\text{pgcd}(\zeta, p^r) = 1$, donc ζ est inversible dans $\mathbb{Z}/p^r\mathbb{Z}$. Comme $\Phi(p^r) = p^{r-1}(p-1)$, on a, avec le théorème d'Euler, $\zeta^{\Phi(p^r)} = \zeta_0^{p-1} = 1$.
3. Dans $\mathbb{F}_p$, on a $\zeta^p = \zeta$, donc $\zeta^s = (\zeta^p)^s = \cdots = (\zeta^{p^{r-1}})^s$. Donc, comme $s < n \leqslant p$, $\zeta_0^s - 1$ est inversible dans $\mathbb{F}_p$. Ceci signifie que $\text{pgcd}(\zeta_0^s - 1, p) = 1$, donc on a $\text{pgcd}(\zeta_0^s - 1, p^r) = 1$, et $\zeta_0^s - 1$ est aussi inversible dans $\mathbb{Z}/p^r\mathbb{Z}$.
4. Avec le théorème chinois, on a $\mathbb{Z}/m\mathbb{Z} \simeq \prod \mathbb{Z}/p_i^{k_i}\mathbb{Z}$. Dans chaque $\mathbb{Z}/p_i^{k_i}\mathbb{Z}$, on choisit une racine $n^{\text{ième}}$ principale ζ_i. On vérifie alors que $(\zeta_1, \ldots, \zeta_r) \in \prod \mathbb{Z}/p_i^{k_i}\mathbb{Z}$ est une racine $n^{\text{ième}}$ principale.

Correction de l'exercice VI.3 :

1. Si ζ est une racine $mn^{\text{ième}}$ de l'unité, alors $\alpha = \zeta^m$ est une racine $n^{\text{ième}}$ de l'unité. De plus, $\alpha^i - 1 = \zeta^{mi} - 1$ n'est pas diviseur de zéro, puisque $0 < mi \leqslant mn$. Idem pour $\beta = \zeta^n$.

Réciproquement, si ζ^m est une racine principale $n^{\text{ième}}$, alors $\zeta^{mn} = (\zeta^m)^n = 1$ donc ζ est une racine $mn^{\text{ième}}$ de l'unité. Soit $0 < i < mn$, et a tel que $a\zeta^i = a$. Montrons que $a = 0$. Par division euclidienne, on écrit $i = nq + r$, avec $0 \leqslant r < n$, et $q < m$. Si $r = 0$, alors, comme $\zeta^{nq} - 1$ n'est pas diviseur de zéro (puisque ζ^m est racine principale $n^{\text{ième}}$), on a terminé. Si $r > 0$, alors on a, en itérant $a\zeta^i = a$, la relation $a(\zeta^i)^k = a$. En prenant $k = m$, on obtient $a\zeta^{mnq+mr} = \zeta^{mr} = a$, ce qui implique $a = 0$ car ζ^m est racine principale $n^{\text{ième}}$.

2. Les racines carrées de l'unité sont racines du polynôme $(X-1)(X+1)$ donc sont -1 ou $+1$. Pour obtenir une racine principale, il faut donc que $\zeta = -1$ et que $\zeta - 1 = -2$ ne soit pas diviseur de zéro. Donc si 2 n'est pas diviseur de zéro, -1 est la seule racine carrée principale.
3. La propriété a été démontrée à la question précédente pour $k = 1$. On suppose la propriété démontrée jusqu'au rang $k - 1 > 0$. Avec la question 1, ζ est une racine $(2^k)^{\text{ième}}$ principale si et seulement si $(2^{k-1})^{\text{ième}}$ est une racine carrée principale (donc égale à -1) et si ζ^2 est une racine $(2^{k-1})^{\text{ième}}$. En appliquant l'hypothèse de récurrence à ζ^2, on a $(\zeta^2)^{2^{k-2}} = \zeta^{2^{k-1}} = -1$.

Correction de l'exercice VI.4 :

1. On note $v_k^0 = 1 + v_k$ et $v_k^1 = v_k$. On a la décomposition

$$\widetilde{f}(v_0, \ldots, v_{n-1}) = \sum_{(i_0,\ldots,i_{n-1}) \in \{0,1\}^n} \widetilde{f}(i_0, \ldots, i_{n-1}) \prod_{k=0}^{n-1} v_k^{i_k}.$$

En développant les produits, on trouve bien un polynôme de la forme demandée. Comme il y a 2^n tels polynômes, et 2^n fonctions booléennes, la décomposition trouvée est unique.

2. On a

$$\mathscr{W}(f)(k) = \sum_{u \in (\mathbb{F}_2)^n} (-1)^{\langle u,k\rangle + \widetilde{f}(u)},$$

donc $\mathscr{W}(f)(k)$ est égal au nombre de 0 moins le nombre de 1 dans le vecteur $\{\widetilde{f}(u) + \langle u,k\rangle\}_{u \in (\mathbb{F}_2)^n}$. Ceci signifie que

$$\mathscr{W}(f)(k) = 2^n - 2d(f, f_{k,0}).$$

Comme on a aussi

$$d(f, 1 + f_{k,0}) = d(f, f_{k,1}) = 2^n - d(f, f_{k,0}),$$

il vient

$$\min\left(d(f, f_{k,0}), d(f, f_{k,1})\right) = \frac{1}{2}\left(2^n - |\mathscr{W}(f)(k)|\right).$$

D'où le résultat, en passant au min sur l'ensemble des $f_{k,b}$.

3. Si f vérifie $|\mathscr{W}(f)(k)| = 2^{n/2}$, alors $N(f) = 2^{n-1} - 2^{n/2-1}$. Soit g une fonction telle que $\exists k$, $|\mathscr{W}(g)(k)| \neq 2^{n/2}$. Avec la formule de Plancherel, proposition 4.7, chap. I, on a

$$\sum_{s=0}^{2^n-1} |\mathscr{W}(g)(s)|^2 = 2^{2n}.$$

Ainsi, comme il y a 2^n termes dans la somme, $\exists k$ tel que $|\mathscr{W}(g)(k)| > 2^{n/2}$. On a donc $N(g) < 2^{n-1} - 2^{n/2-1}$.

4. On a

$$\mathscr{W}(h)(w) = \sum_{t\in(\mathbb{F}_2)^{n+m}} (-1)^{\langle w,t\rangle+\widetilde{h}(t)}$$
$$= \sum_{r\in(\mathbb{F}_2)^n}\sum_{s\in(\mathbb{F}_2)^m} (-1)^{\langle u,r\rangle+\widetilde{f}(r)}(-1)^{\langle v,s\rangle+\widetilde{g}(s)} = \mathscr{W}(f)(u)\mathscr{W}(g)(v).$$

Si $\widetilde{f}$ et $\widetilde{g}$ sont bents, $|\mathscr{W}(f)(u)| = 2^{n/2}$ et $|\mathscr{W}(g)(v)| = 2^{m/2}$ et donc on a bien $|\mathscr{W}(h)(w)| = 2^{(m+n)/2}$. Réciproquement, si par exemple $\widetilde{f}$ n'est pas bent, nous avons déjà vu à la question 3 que $\exists u_0$, $|\mathscr{W}(f)(u_0)| > 2^{n/2}$. Si on suppose que $\widetilde{h}$ est bent, alors pour $w = (u_0, v)$,

$$2^{(m+n)/2} = |\mathscr{W}(h)(w)| = |\mathscr{W}(f)(u_0)||\mathscr{W}(g)(v)| \Longrightarrow \forall v, |\mathscr{W}(g)(v)| < 2^{m/2},$$

ce qui est impossible car $N(g) \leqslant 2^{m-1} - 2^{m/2-1}$.
On vérifie que $\mathscr{W}(f_0) = \{2,2,2,-2\}$, donc f_0 est bent. Ainsi la fonction

$$f(u_0,\ldots,u_{n-1}) = u_0u_1 + \cdots + u_{n-2}u_{n-1}$$

est bent.

5. La dimension de $R(1,n)$ est $n+1$, et sa distance minimale est 2^{n-1}. Ceci vient du fait que les fonctions $\widetilde{f_{a,b}}$, pour $a \neq 0$, prennent 2^{n-1} fois la valeur 0, et 2^{n-1} fois la valeur 1.
On a $f_{a,b}(u) = (-1)^b\mathscr{W}(\delta_a)(u)$, ce qui se calcule rapidement à l'aide de l'algorithme FWT. La procédure MATLAB 6.2 réalise ce codage. Elle prend comme paramètre a sous la forme d'un entier $0 \leqslant a \leqslant 2^n - 1$.

Programme 6.2 Procédure `encode_rm`

```
function y = encode_rm(a,b,n)
y = zeros(2^n,1); y(a+1) = 1;
y = fwt(y); y = (1-y)/2;
if(b==1) y = 1-y; end;
```

Avec la question 2, on voit que $d(F_{a,b}, F)$ est minimale lorsque $|\mathscr{W}(f)(a)|$ est maximum. Ensuite, on a $b = 0$ si $\mathscr{W}(f)(a) > 0$, et $b = 1$ sinon. La procédure MATLAB 6.3 réalise ce décodage, et prend en entrée un vecteur x de taille 2^n représentant F.

Programme 6.3 Procédure `decode_rm`

```
function y = decode_rm(x)
N = length(x);
f = fwt((-1).^x);
[v,a] = max(abs(f));
y = encode_rm( a-1,f(a)<0,log2(N) );
```

Correction de l'exercice VI.5 :

1. On a, en utilisant la formule de Plancherel, proposition 4.7, chap. I,

$$E((f-h)^2) = \sum_{\alpha\neq\beta} a_\alpha^2 = 1 - a_\beta^2.$$

2. On note $I(x)$ la fonction qui vaut 1 si $f(x) \neq h_0(x)$, et 0 sinon. On a

$$\mathbb{P}(f(x) \neq h_0(x)) = \frac{1}{2^n} \sum_x I(x).$$

Il faut donc montrer que $I(x) \leqslant (f(x) - h(x))^2$. Si $f(x) \neq h_0(x)$, alors $I(x) = 0$ et l'inégalité est vraie. Sinon, alors nécessairement $|f(x) - h(x)| \geqslant 1$, et on a bien $I(x) = 1 \leqslant (f(x) - h(x))^2$.

3. En appliquant la borne de Chernoff-Hoeffding aux $X_i = f(x_i)$, qui sont i.i.d. et tels que $E(X_i) = E(f), X_i \in [-1, 1]$, on obtient

$$\mathbb{P}(|\widetilde{c_\beta} - c_\beta| \geqslant \lambda) \leqslant 2e^{-\lambda^2 m/2} \leqslant \delta.$$

Avec une probabilité inférieure à δ, on a donc

$$\mathbb{P}(f(x) \neq \varphi_0(x)) \leqslant E((f - \widetilde{a_\beta}\chi_\beta)^2) \leqslant \sum_{\alpha \neq \beta} a_\alpha^2 + (a_\beta - \widetilde{a_\beta})^2 \leqslant 1 - a_\beta^2 + \lambda^2.$$

4. On note $S_d \overset{\text{déf.}}{=} \{s \setminus w(s) < d\}$. En utilisant la borne de Chernoff-Hoeffding, on a $\mathbb{P}(|a_s - \widetilde{a_s}| \geqslant \lambda) \leqslant 2e^{-\lambda^2 m/2}$. De plus, sous la condition $|a_s - \widetilde{a_s}| \leqslant \lambda$, on a

$$E((f - \varphi)^2) \leqslant \alpha + \sum_{s \in S_d} (a_s - \widetilde{a_s})^2 \leqslant \alpha + n^d \lambda^2,$$

puisque $\text{Card}(S_d) \leqslant n^d$. Pour avoir $\mathbb{P}(f(x) \neq \varphi(x)) \leqslant \alpha + \varepsilon$, il faut donc imposer $\lambda \leqslant \sqrt{\varepsilon / n^d}$, et pour que ceci ait lieu avec une probabilité $1 - \delta$, il faut que l'on ait $2e^{-\lambda^2 m/2} n^d \leqslant \delta$, puisque

$$\mathbb{P}(\forall s \in S_d, |a_s - \widetilde{a_s}| \geqslant \lambda) \leqslant \sum_{s \in S_d} \mathbb{P}(|a_s - \widetilde{a_s}| \geqslant \lambda) \leqslant 2e^{-\lambda^2 m/2} n^d.$$

Correction de l'exercice VI.6 :

1. On note G et H des matrices génératrices et de contrôle. On a

$$y \in \mathscr{C}^\perp \Leftrightarrow \forall x \in (\mathbb{F}_p)^n, \langle Gx, y \rangle = 0 \Leftrightarrow \forall x \in (\mathbb{F}_p)^n, \langle x, G^{\mathrm{T}} y \rangle = 0 \Leftrightarrow G^{\mathrm{T}} y = 0.$$

Donc une matrice de contrôle de $\mathscr{C}^\perp$ est G^{T}, et une matrice génératrice est H^{T}.

2. Une telle forme simplifie le codage (ainsi que le décodage, comme nous allons le voir sur la forme de H). On peut choisir $H = (A | \mathrm{Id}_{n-m})$, et on vérifie bien que $HG = 0$, avec $\text{rang}(G) = m - n$.

3. Deux codes sont équivalents si on peut passer d'une matrice G_1 du premier code à une matrice G_2 du deuxième par

– opérations élémentaires sur les colonnes (qui ne modifient pas l'espace engendré par les colonnes, donc le code). Ceci correspond aux opérations $C_i \leftarrow \lambda C_i$ ainsi que $C_i \leftarrow C_i + \lambda C_j$, pour $C_i \neq C_j$ des colonnes et $\lambda \neq 0$.

– permutation des lignes (ce qui correspond à une permutation des symboles).

Par pivotage de Gauss (voir [16]), on peut se ramener, par ces opérations, à une forme systématique.

Correction de l'exercice VI.7 :

1. La distance minimale d'un code est le nombre minimal de colonnes linéairement dépendantes. Comme deux colonnes sont ici distinctes, leur somme modulo 2 n'est jamais nulle, et elles sont donc indépendantes. Par contre, cette somme est nécessairement égale à une troisième colonne, donc on peut trouver trois colonnes liées. La distance minimale est donc de 3.
   Comme H possède k lignes, la dimension de $\mathscr{C}$ est $2^k - 1 - k$. Si $v' = v + e$ est le mot reçu, avec $w(e) = 1$ et $v \in \mathscr{C}$, alors $s = Hv' = He$ est le syndrome. Ainsi, s est une colonne de H. Par exemple, si on a pris comme $i^{\text{ième}}$ colonne de H la décomposition de i en binaire, alors la position de l'erreur dans v' est simplement l'entier dont s est la décomposition binaire. La fonction MATLAB 6.4 réalise le décodage. Elle utilise la procédure `ecriture_binaire` 6.5, qui décompose un entier en écriture binaire.

Programme 6.4 Procédure `decode_hamming`

```
function y = decode_hamming(x)
n = length(x); k = log2(n+1);
H = zeros(k,n); y = x;
for(j=1:n) H(:,j) = ecriture_binaire(j,k); end;
s = mod(H*x,2);
e = dot(s,2.^(0:(k-1)));
if(e~=0) y(e)=1-y(e); end;
```

Programme 6.5 Procédure `ecriture_binaire`

```
function y = ecriture_binaire(x,k)
y = zeros(k,1);
for(i=1:k) q = floor(x/2); y(i) = x-2*q; x = q; end;
```

2. Dans la base $\{\alpha, \ldots, \alpha^k\}$ de $\mathbb{F}_{2^k}$ comme espace vectoriel sur $\mathbb{F}_2$, α s'écrit, sous forme vectorielle, $(1,0,\ldots,0)$, α^2 s'écrit $(0,1,,0,\ldots,0)$, etc. Les α^i, pour i entre 1 et n, décrivent toutes les représentations binaires des nombres entre 1 et n. Un mot de $\mathscr{C}_\alpha$, le code BCH généré par α (de distance assignée n) est représenté par un polynôme P et $P \in \mathbb{C}_\alpha$ si et seulement si $P(\alpha) = \cdots = P(\alpha^n) = 0$. En écrivant P sous la forme d'un vecteur binaire $\widetilde{P}$ de taille n, les égalités précédentes s'écrivent $H\widetilde{P} = 0$. Ce code est donc bien le code de Hamming de taille n.
3. Il y a $n+1 = 2^k$ mots dans une boule de rayon 1. Comme la distance minimale du code est 3, les boules centrées en des mots du code sont disjointes. Cet ensemble de boules contient donc $2^m \times 2^k = 2^{2^k-1} = 2^n$ mots, c'est-à-dire tout l'espace $(\mathbb{F}_2)^n$. Dans le cas $n = 7$, on peut prendre

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix},$$

   et on vérifie que $HG = 0$, où G est la matrice de l'exemple 3.12, chap. VI.
4. La matrice génératrice de $\mathscr{C}^\perp$ est $G_0 = H^{\mathrm{T}}$. Pour H, on a choisi pour $i^{\text{ième}}$ colonne $(1 \leqslant i \leqslant 2^k - 1)$ la décomposition de i en écriture binaire. On fait précéder G_0 d'une ligne de 0 (ce qui ne change pas le poids des mots). On voit alors que la première colonne est une alternance de 0 et de 1, que la deuxième est une alternance de

de 00 et de 11, et ainsi de suite. Toutes les colonnes de G_0 ont ainsi pour poids 2^{k-1}. On voit facilement qu'une opération élémentaire sur les lignes du type $L_i \leftarrow L_i + \sum_{j\in J} L_j$ se contente d'effectuer une permutation sur les colonnes de la matrice, donc le poids des colonnes reste 2^{k-1}. Par de telles opérations, on obtient tous les mots non nuls de $\mathscr{C}^\perp$, qui ont donc comme poids 2^{k-1}. Comme la distance entre deux mots est le poids du mot différence, ce code est bien simplexe.

5. Sur $\mathbb{F}_q$, on choisit pour les colonnes de H des représentants des droites vectorielles, c'est-à-dire de l'espace projectif $\mathbb{P}(\mathbb{F}_q^n)$. Comme Card $(\mathbb{P}(\mathbb{F}_q^n)) = \frac{q^k-1}{q-1}$ (chaque droite a $q-1$ éléments non nuls), on obtient bien la taille et la dimension souhaitées. La preuve de la distance minimale est inchangée. Le code est encore parfait, un boule de rayon 1 contenant $n(q-1)+1$ mots.

Correction de l'exercice VI.8 :

1. La représentation binaire des entiers permet d'établir une bijection entre l'ensemble $\{1,\ldots,2^k-1\}$ et $\mathbb{F}_2^k\backslash\{0\}$. Pour $k=3$, on a

$$W_{\mathscr{C}_7}(X,Y) = X^7 + 7X^3Y^4 + 7X^4Y^3 + X^7$$

2. Voir exercice VI.7, question 4.
3. On a $W_{\mathscr{C}^\perp}(X,Y) = Y^n + 2^{k-1}X^{n-2^{k-1}}Y^{2^{k-1}}$, d'où le résultat avec le théorème 4.5, chap. VI.
4. On note $P(Y) = W_{\mathscr{C}}(1,Y)$. On a $A_s = \frac{1}{s!}\frac{\mathrm{d}^s P}{\mathrm{d}Y^s}(0)$. Le sript MAPLE 6.6 effectue le calcul, et on trouve bien $A_1 = A_2 = 0$, ce qui est logique, puisque la distance minimale est 3. Voici quelques valeurs de A_k

k	1	2	3	4	5	6
A_k	0	0	$\frac{n(n-1)}{6}$	$\frac{n(n-1)(n-3)}{24}$	$\frac{n(n-1)(n-3)(n-7)}{120}$	$\frac{n(n-1)(n-3)(n-5)(n-7)}{720}$

Programme 6.6 Calcul de A_k pour un code de Hamming

```
P := Y -> 1/(n+1)*( (1+Y)^n+n*(1-Y)^((n+1)/2)*(1+Y)^((n-1)/2) );
A := s -> factor( 1/(s!)*eval( diff(P(Y),Y$s), Y=0 ) );
```

5. Pour montrer la symétrie, il faut montrer que $P(Y) = Y^n P(1/Y)$, ce qui se vérifie facilement.

Correction de l'exercice VI.9 : On vérifie que $W_{\mathscr{C}}(X,Y) = X^8 + 14X^4Y^4 + Y^8$, et que ce polynôme est invariant par le changement de variable $(X,Y) \mapsto \frac{1}{\sqrt{2}}(X+Y, X-Y)$ (on le vérifie aisément à l'aide de MAPLE).

Correction de l'exercice VI.10 :

1. $\mathscr{C}$ est un code de dimension 1 et de distance minimale n. Son polynôme énumérateur est $Y^n + (q-1)X^n$.
2. On a $x \in \mathscr{C}^\perp \Leftrightarrow \sum x_i = 0$ dans $\mathbb{F}_q$. Le code dual correspond au code du bit de parité (exemple 3.1, chap. VI que l'on étend à $\mathbb{F}_q$). Il est de dimension $n-1$ et de distance minimale 1. On voit facilement que $\mathscr{C} = \mathscr{C}^\perp$ si et seulement si $n=1$ (code trivial) ou $n=2$.

Correction de l'exercice VI.11 :

1. Dans le cas de la construction par matrice de Walsh ($n = 2^k$), les mots de $\mathscr{A}_n$ sont les mots de $R(1,k)$, le code de Reed-Muller (voir exercice VI.4, question 5), auxquels on a enlevé le premier symbole. De façon équivalente, $R(1,k)$ correspond au code $\mathscr{A}_n$ avec un bit de parité en plus (qui vaut toujours 1). Pour cette construction, $\mathscr{A}_n$, tout comme $\mathscr{B}_n$, est donc linéaire. Pour $n = 12$, avec la construction de Paley, on obtient le code $\mathscr{B}_{12}$ suivant, qui est non-linéaire.

$\mathscr{A}_{12}$	$\mathscr{B}_{12} \setminus \mathscr{A}_{12}$
(00000000000) (00010110111)	(11111111111) (11101001000)
(11011100010) (10001011011)	(00100011101) (01110100100)
(01101110001) (11000101101)	(10010001110) (00111010010)
(10110111000) (11100010110)	(01001000111) (00011101001)
(01011011100) (01110001011)	(10100100011) (10001110100)
(00101101110) (10111000101)	(11010010001) (01000111010)

2. L'orthogonalité des lignes de H_n signifie exactement que deux lignes ont autant d'entrées égales que d'entrées qui diffèrent. Deux mots de $\mathscr{A}_n$ sont donc distants de $n/2$, qui est la distance minimale. Si on note $u, v \in \mathscr{A}_n$ et $\overline{u}, \overline{v}$ leurs compléments, on a $d(u,v) = d(\overline{u},\overline{v}) = n/2$, et $d(u,\overline{v}) = n/2 - 1$. Donc la distance minimale de $\mathscr{B}_n$ est $n/2 - 1$. Le code $\mathscr{A}_n$ est de taille $n-1$ avec n éléments, et $\mathscr{B}_n$ est de taille $n-1$ avec $2n$ éléments.

Correction de l'exercice VI.12 :

1. En développant l'égalité de MacWilliams, on obtient

$$|\mathscr{C}| W_{\mathscr{C}^\perp}(1,Y) = \sum_{k=0}^{n} A_k \left(\sum_{j=0}^{n-k} C_{n-k}^j Y^j \right) \left(\sum_{j=0}^{k} (-1)^j C_k^j Y^j \right).$$

   En égalant les coefficients de ces deux polynômes, on trouve les égalités demandées.

2. On dérive k fois l'égalité $2^{n-m} \mathscr{W}_{\mathscr{C}}(X,1) = \mathscr{W}_{\mathscr{C}^\perp}(X+1, X-1)$. En utilisant la règle de Leibniz pour dériver un produit, on obtient

$$2^{n-m} \sum_{i=0}^{n-k} A_i k! C_{n-i}^k X^i = \sum_{i=0}^{k} A_i' \sum_{s=0}^{k} C_k^s \frac{\mathrm{d}^{k-s}}{\mathrm{d}X^{k-z}} (X+1)^{n-i} \frac{\mathrm{d}^s}{\mathrm{d}X^s} (X-1)^i.$$

   En faisant $X = 1$ dans cette égalité, seul le terme correspondant à $s = i$ reste dans la deuxième somme de droite. On obtient les égalités souhaitées.

3. Il n'y a aucun mot de poids $i \in \{1, \ldots, d-1 = n-m\}$ dans $\mathscr{C}$. Pour $\mathscr{C}^\perp$, il faut montrer que sa distance minimale est $m+1$. En effet, d est égal au plus petit nombre de colonnes linéairement dépendantes dans H (une matrice de contrôle de $\mathscr{C}$). Donc le plus grand nombre de colonnes linéairement indépendantes dans H est égal à $d-1 = n-m$. Or H^{T} est la matrice génératrice de $\mathscr{C}^\perp$, donc dès qu'un mot de $\mathscr{C}^\perp$ a moins de $n-k$ coordonnées non nulles, ce mot est nul. En conséquence, la distance minimale de $\mathscr{C}^\perp$ est au moins de $n-(n-k)+1 = k+1$. Il y a en fait égalité en appliquant la borne de Singleton à $\mathscr{C}^\perp$. En restreignant les sommes aux indices i tels que $A_i \neq 0$ et $A_i' \neq 0$, en utilisant $C_n^{n-k} = C_n^k$, et $A_0' = 1$, on obtient les m équations demandées.

4. On a un système de m équations à m inconnues (les A_i, pour $i = n-m+1,\dots,n$). Ce système est en fait triangulaire, et il se résout facilement par remontée. Ainsi, les A_i sont uniquement déterminés. Par exemple, pour $i = m-1$, on a $A_d = C_n^{k-1}$.

Correction de l'exercice VI.13 :

1. Avec la proposition 4.15, chap. VI, on sait que $B_i' \geqslant 0$. Avec l'expression de A_i' trouvée à l'exercice VI.12, question 1, qui s'étend à la distribution B_i' (grâce au théorème 4.11, chap. VI), on obtient les inégalités souhaitées.
2. Si $\mathscr{C}$ a pour distance minimale d, alors $B_1 = \cdots = B_d = 0$, et le cardinal de $\mathscr{C}$ est $\sum_{i=0}^{n} B_i$. La distribution de distance de $\mathscr{C}$ appartient donc à E_n^d, et on trouve la borne annoncée.

7 Correction des exercices du chapitre 7

Correction de l'exercice VII.1 : La droite $\mathrm{Vect}(e_1)$ est stable, donc la représentation est réductible. Par contre, si la représentation était décomposable, on aurait la somme de sous-représentations $K^2 = \mathrm{Vect}(e_1) \oplus \mathrm{Vect}(e_2 + \lambda e_1)$, où $\lambda \in K$. Il est facile de voir que $\mathrm{Vect}(e_2 + \lambda e_1)$ ne peut être stable.

Correction de l'exercice VII.2 : On note $\varphi = A\delta_e$, où e est l'élément neutre de G. On a

$$Af = \sum_{g\in G} f(g) A\delta_g = \sum_{g\in G} f(g)\tau_g(A\delta_e) = \sum_{g\in G} f(g)(\tau_g f) = f * \varphi.$$

Correction de l'exercice VII.3 : On peut voir $\chi_W = \chi_U \chi_V$ comme le caractère de la représentation des morphismes sur $W = \mathscr{L}(U, V^*)$, ou bien comme le produit tensoriel $W = U \otimes V$. Comme U est non triviale, χ_W est distinct de χ_U et de χ_V, donc on a bien construit une nouvelle représentation. Comme $|\chi_U| = 1$, on a

$$\sum_{g\in G} |\chi_W(g)|^2 = \sum_{g\in G} |\chi_V(g)|^2 = |G|,$$

donc W est bien irréductible.

Correction de l'exercice VII.4 : On note $\rho_k : G \to GL(V_k)$, $k = 1,\dots,p$, les représentations irréductibles de G, et $\sigma_l : H \mapsto GL(W_l)$, $l = 1,\dots,q$, celles de H. On note $m_k = \dim(U_k)$ et $n_l = \dim(V_l)$. Pour $(k,l) \in \{1,\dots,p\} \times \{1,\dots,q\}$, on définit

$$\tau_{k,l}(g,h) = \rho_k(g) \otimes \sigma_l(h) \in GL(V_k \otimes W_l),$$

qui est une représentation de $G \times H$ sur $V_k \otimes W_l$. On a $\chi_{\tau_{k,l}} = \chi_{\rho_k}\chi_{\sigma_l}$, en particulier, $\|\chi_{\tau_{k,l}}\|_2 = \|\chi_{\rho_k}\|_2 \|\chi_{\sigma_l}\|_2 = 1$, donc $\tau_{k,l}$ est irréductible. De plus, on a

$$\sum_{k,l} \dim(V_k \otimes W_l)^2 = \sum_{k,l} m_k^2 n_l^2 = |G| \times |H| = |G \times H|,$$

donc on a bien trouvé toutes les représentations irréductibles.

Correction de l'exercice VII.5 :

1. Pour $\sigma \in \mathfrak{S}_n$, on considère la matrice $M_\sigma : e_i \mapsto e_{\sigma(i)}$. Le groupe $G = \mathfrak{S}_n$ est isomorphe au groupe matriciel $H = \{M_\sigma\}_{\sigma\in G}$, et l'action de $\mathfrak{S}_n$ par permutation des indéterminées correspond à l'action linéaire de H sur $K[X_1,\dots,X_n]$.

Un résultat classique affirme que l'anneau des invariants est généré par les polynômes

$$\sigma_k(X_1,\dots,X_n) \overset{\text{déf.}}{=} \sum_{i_1<\cdots<i_k} X_{i_1}X_{i_2}\cdots X_{i_k}.$$

2. On a $K[X,Y]^{V_4} = K[X^2,Y^2]$. L'écriture d'un élément de $K[X,Y]^{V_4}$ en fonction de X^2 et Y^2 est unique.
   On a $K[X,Y]^{C_2} = K[X^2,Y^2,XY]$. La décomposition n'est pas unique, puisque l'on a la relation $(XY)^2 = X^2Y^2$.
3. Si $f = \sum c_\alpha X^\alpha \in K[X_1,\dots,X_n]^G$, alors $f = \sum c_\alpha R_G(X^\alpha)$. Ainsi, si on prouve que $R_G(X^\alpha)$ s'écrit comme un polynôme en $R_G(X^\beta)$ pour $|\beta| \leqslant |G|$, on pourra écrire f en fonction de ces $R_G(X^\beta)$.
4. On a $(U_A)^k = \sum_{|\alpha|=k} a_\alpha (A\cdot X)^\alpha u^\alpha$. En sommant ces égalités pour $A \in G$, on trouve l'expression des $S_k = S_k(U_A\ ;\ A \in G)$ voulue. Tout polynôme symétrique en les U_A s'écrit en fonction de $|G|$ sommes de Newton $S_1,\dots,S_{|G|}$. Comme S_k est un polynôme symétrique, il existe $F \in K[Y_1,\dots,Y_{|G|}]$ tel que $S_k = F(S_1,\dots,S_{|G|})$, ce qui donne l'égalité polynomiale demandée. En égalant les coefficients de u^α des deux membres de l'égalité, on voit que $|G|a_\alpha R_G(X^\alpha)$ est égal à un polynôme en $R_G(X^\beta)$, pour $|\beta| \leqslant |G|$. Comme K est de caractéristique 0, on peut diviser par $|G|a_\alpha$.
5. On peut calculer les $R_{C_4}(X^\beta)$, pour $|\beta| \leqslant 4$:

X^iY^i	$R_{C_4}(X^iY^i)$	X^iY^i	$R_{C_4}(X^iY^i)$
X	0	XY^2	0
Y	0	Y^3	0
X^2	$(X^2+Y^2)/2$	X^4	$(X^4+Y^4)/2$
XY	0	X^3Y	$(X^3Y-XY^3)/2$
Y^2	$(X^2+Y^2)/2$	X^2Y^2	X^2Y^2
X^3	0	XY^3	$-(X^3Y-XY^3)/2$
X^2Y	0	Y^4	$(X^4+Y^4)/2$

L'anneau des invariants est donc généré par

$$P_1 = X^2+Y^2,\quad P_2 = X^4+Y^4,\quad P_3 = X^3Y-XY^3,\quad P_4 = X^2Y^2.$$

Cependant, on note que $P_2 = P_1^2 - 2P_4$, donc on peut supprimer P_2 de cette liste.

Correction de l'exercice VII.6 :

1. L'action de G étant linéaire, elle conserve le degré des composantes homogènes. Comme deux polynômes sont égaux si et seulement si leurs composantes homogènes sont égales, on en déduit le résultat souhaité.
2. Le théorème 2.11, chap. VII, nous dit que $\dim_{\mathbb{C}}(V_s^G) = \operatorname{tr}(R_G)$, où R_G est l'opérateur de Reynolds pour la représentation ρ_s. Comme $\operatorname{tr}(R_G) = \frac{1}{|G|}\sum_{g\in G}\operatorname{tr}(A^{[s]})$, on obtient la formule demandé.
3. Soit $A \in G$, de valeurs propres $\omega_1,\dots,\omega_n$. A un changement de base près, on peut écrire

$$A = A^{[1]} = \operatorname{diag}(\omega_1,\dots,\omega_n),\quad A^{[s]} = \operatorname{diag}(\omega_1^s,\dots,\omega_n^s,\omega_1^{s-1}\omega_2,\dots).$$

On a donc

$$\mathrm{tr}\left(A^{[s]}\right) = \sum_{i_1 \leqslant \cdots \leqslant i_s} \omega_{i_1} \cdots \omega_{i_s},$$

ce qui correspond bien à la puissance de λ^s dans l'expansion de

$$\det(\mathrm{Id} - \lambda A)^{-1} = \prod_{i=1}^{n} (1 - \lambda \omega_i)^{-1}.$$

4. Le groupe G_1 a pour série de Molien $[(1-\lambda)(1-\lambda^2)]^{-1}$. Le groupe G_2 a pour série de Molien $(1-\lambda^2)^{-2}$. Le nombre de polynômes linéairement indépendants borne le nombre de polynômes algébriquement indépendants, ce qui permet de limiter la recherche.

Correction de l'exercice VII.7 :

1. On a
$$\langle \chi_1, \chi_\pi \rangle = \frac{1}{|G|} \sum_{g \in G} |\chi_\pi(g)| = \frac{1}{|G|} \sum_{g \in G} |X_g|.$$

2. On note $G_x = \{g \in G \setminus g \cdot x = x\}$ le stabilisateur de X, et $Gx = \{g \cdot x \setminus g \in G\}$ l'orbite de x. On a $|G| = |G_x||Gx|$ (voir par exemple [58]).
On note $x_1, \ldots, x_t$ des représentants des t orbites distinctes de l'action de G sur X. On note $T = \{(g,x) \in G \times X \setminus g \cdot x = x\}$. En comptant « dans les deux directions » les éléments de T, on a

$$\sum_{g \in G} |X_g| = |T| = \sum_{x \in X} |G_x| = \sum_{i=1}^{t} \sum_{x \in Gx_i} |G_x| = \sum_{i=1}^{t} |G_{x_i}||Gx_i| = t|G|.$$

3. Le cardinal de X (les colliers « virtuels ») est de $2^6 = 64$. Le nombre de colliers « réels » différents est t, le nombre d'orbites de l'action de D_6 sur X, en appliquant une isométrie $\sigma \in D_6$ à l'hexagone régulier dont les affixes sont les $e^{ik\pi/3}$ (que l'on assimile à un collier !). On note $\{c_0, \ldots, c_5\} \in X$ un collier virtuel. On calcule $|X_\sigma|$ en faisant les distinctions suivantes.
   - Si $\sigma = \mathrm{Id}$, alors $|X_{\mathrm{Id}}| = 64$.
   - Si σ est la rotation d'angle $\pm\pi/6$, alors si c est stable sous σ, il doit vérifier $c_{i+1} = c_i$. Ainsi, le collier est unicolore, d'où $|X_\sigma| = 2$.
   - Si σ est la rotation d'angle $\pm\pi/3$, alors si c est stable sous σ, il doit vérifier $c_0 = c_2 = c_4$ et $c_1 = c_3 = c_5$. On a 2 choix de couleurs à faire, d'où $|X_\sigma| = 4$.
   - Soit σ une symétrie dont l'axe fait un angle de $\pi/3$ avec les abscisses (même raisonnement avec 0 et $2\pi/3$). Alors si c est stable sous σ, il doit vérifier $c_0 = c_2$ et $c_3 = c_5$. Il y a 4 choix de couleurs à faire, donc $|X_\sigma| = 16$.
   - Soit σ une symétrie dont l'axe fait un angle de $\pi/6$ avec les abscisses (même raisonnement avec $\pi/2$ et $5\pi/6$). Alors si c est stable sous σ, il doit vérifier $c_0 = c_3$, $c_1 = c_2$ et $c_4 = c_5$. Il y a 3 choix de couleurs à faire, donc $|X_\sigma| = 8$.

   Au final, on a donc

$$t = \frac{1}{12}(64 + 2 \times 2 + 2 \times 4 + 3 \times 16 + 3 \times 8) = 13 \text{ colliers différents.}$$

4. On écrit f sous forme matricielle $f = \{\varphi(s,t)\}_{(s,t)\in X^2}$. On voit facilement que le fait que f soit un opérateur d'entrelacement est équivalent à $\varphi(g\cdot s, g\cdot t) = \varphi(s,t)$, donc φ est constante sur chacune des orbites de $X\times X$ sous l'action de G définie par $g\cdot(s,t) = (g\cdot s, g\cdot t)$. Or la double transitivité de G est équivalente au fait que $X\times X$ ait exactement 2 orbites

$$O_1 = \{(s,s) \setminus s\in X\} \quad \text{et} \quad O_2 = \{(s,t) \setminus s\neq t\in X\}.$$

Si on identifie f à φ, l'espace $\mathrm{Hom}_G(V)$ admet pour base $\{\mathbf{1}_{O_1}, \mathbf{1}_{O_2}\}$, les fonctions indicatrices de O_1 et O_2.
Nous avons déjà vu, lors de la démonstration du théorème 3.7, chap. VII, le fait que $\dim(\mathrm{Hom}_G(V)) = \dim(\mathscr{L}(V,V)^G)$, où on a considéré sur $\mathscr{L}(V,V)$ la représentation des morphismes associée à G. On a aussi vu que $\dim(\mathscr{L}(V,V)^G)$ est égal à $\mathrm{tr}(R_G) = \langle\chi_\pi,\chi_\pi\rangle$, où R_G est l'opérateur de Reynolds associé à la représentation des morphismes. Au final, on a donc $\langle\chi_\pi,\chi_\pi\rangle = 2$.
Il est évident que l'espace $U = \mathbf{1}\mathbb{C}$ engendré par le vecteur constant égal à 1 est invariant sous G. Il admet donc un supplémentaire stable W. On a $\chi_\pi = \chi_1 + \chi_W$, où χ_1 est le caractère de la représentation triviale. Avec la question 2, on a $\langle\chi_\pi,\chi_1\rangle = 1$, puisque G agit transitivement sur X. On a donc

$$\langle\chi_W,\chi_W\rangle = \langle\chi_\pi,\chi_\pi\rangle - 2\langle\chi_\pi,\chi_1\rangle + \langle\chi_1,\chi_1\rangle = 2 - 2\times 1 + 1 = 1.$$

Donc W est bien irréductible.
Le groupe $\mathfrak{S}_n$ agit doublement transitivement sur $X = \{1,\dots,n\}$. Dans la construction précédente, W correspond à la représentation standard, qui est ainsi irréductible.

Correction de l'exercice VII.8 :

1. Il faut montrer que f est un opérateur d'entrelacement :

$$\rho(h)\circ f\circ\rho(h^{-1}) = \sum_{g\in K}\rho(hgh^{-1}) = f.$$

On a $\mathrm{tr}(f) = d_\rho r(\rho,K) = \sum_{g\in K}\mathrm{tr}(\rho(g)) = \mathrm{Card}\,(K)\chi_\rho(K)$.

2. On a

$$\chi(K)\chi(K^{-1}) = \frac{d_\rho}{\mathrm{Card}\,(K)} r(\rho,K)\chi(K^{-1}).$$

Comme ρ est irréductible, on a $\sum_{g\in G}\chi(g)\chi(g^{-1}) = |G|$, d'où

$$|G| = \sum_K\sum_{g\in K}\chi(g)\chi(g^{-1}) = d_\rho\sum_K r(\rho,K)\chi(K^{-1}).$$

3. Si $K''\not\subseteq K\cdot K'$, alors $a(K,K',K'') = 0$. Soit $K''\subseteq K\cdot K'$. Si $hh' = h_1h_1'\in K''$, alors $uxu^{-1} = uhu^{-1}\cdot uh'u^{-1} = uh_1u^{-1}\cdot uh_1'u^{-1}$, donc $a(K,K',hh') = a(K,K',h_1h_1')$.

4. On a

$$\begin{aligned} r(\rho,K)r(\rho,K')\mathrm{Id}_V &= \left(\sum_{k\in K}\rho(g)\right)\left(\sum_{k\in K'}\rho(g)\right) \\ &= \sum_{(g,h)\in K\times K'}\rho(gh) = \sum_{K''}a(K,K',K'')\sum_{u\in K''}\rho(u), \end{aligned}$$

d'où la formule demandée.

5. La formule précédente montre que le produit de deux générateurs de A est encore un élément de A. Donc A est bien un anneau. Il admet un nombre fini de générateurs, donc il est de type fini sur $\mathbb{Z}$. Ceci est l'une des définitions équivalentes d'entier algébrique (voir [63]).
On a $\frac{|G|}{d_\rho} = \sum_K r(\rho,K)\chi(K^{-1})$. Les $r(\rho,K)$ sont des entiers algébriques. De plus, les $\chi(K^{-1})$ sont des racines de l'unité, donc des entiers algébriques. Ainsi, $|G|/d_\rho$ est à la fois un nombre rationnel et un entier algébrique, donc c'est un entier.

Correction de l'exercice VII.9 :

1. On note A la matrice $\rho(X)$. On a
$$A_{g,h} = \langle A\delta_h, \delta_g\rangle = \sum_{a\in G} X_a \langle \delta_{ah}, \delta_g\rangle = X_{gh^{-1}}.$$
2. La matrice de $\rho_{U\oplus V}$ s'écrit comme diagonale par blocs de $\rho_U(X)$ et $\rho_V(X)$, d'où la formule en prenant le déterminant.
3. Comme les V_i sont irréductibles, les morphismes d'algèbres ρ_i sont surjectifs (sinon, il y aurait une sous-représentation non triviale). Supposons que l'on ait une relation du type $\sum c_{jk}\lambda_{jk}(X) = 0$. Par la surjectivité de ρ_i, on peut trouver une valeur x_0 de X dans $\mathbb{C}[G]$ telle que $\rho_i(x_0) = E_{j_0k_0}$ (la matrice avec un 1 en (i_0, j_0), et des 0 partout ailleurs). On obtient $\sum c_{jk}\lambda_{jk}(x_0) = c_{j_0k_0} = 0$, d'où l'indépendance.
4. On note $D_n(Y_1,\dots,Y_{n^2})$ le déterminant générique en n^2 variables. En développant selon la première ligne, on obtient la relation
$$D_n(Y_1,\dots,Y_{n^2}) = D_{n-1}(Y_2,\dots)Y_1 + B(Y_2,\dots,Y_{n^2}).$$
Ainsi, D_n s'écrit comme un polynôme de degré 1 dans $A[Y_1]$, avec $A = \mathbb{C}[Y_2,\dots,Y_{n^2}]$ qui est factoriel. Par récurrence, si on a supposé D_{n-1} irréductible, D_n est encore irréductible.
5. D'après la question 3, on peut compléter les n_i^2 formes linéaires λ_{jk} en une base des formes linéaires, notée $\{Y_1,\dots,Y_{|G|}\}$. Dans cette nouvelle base, on a l'égalité $\Theta_\rho(G)(X) = D_{n_i}(Y_1,\dots,Y_{n_i^2})$, qui est irréductible en tant que polynôme en Y_i. Par changement inverse de coordonnées, on voit que $\Theta_\rho(G)(X)$ est encore irréductible en tant que polynôme en X_g.
6. Comme $\rho_i(1) = \mathrm{Id}_{n_i}$, X_1 n'apparaît que sur la diagonale de $\rho_i(X)$. En écrivant l'expansion du déterminant, on obtient, en écrivant seulement les termes de degré n_i et $n_i - 1$ en X_1,
$$\Theta_{\rho_i}(X) = \prod_{j=1}^{n_i}\lambda_{jj}(X) + \cdots = \prod_{h\in G}\sum_{g\in G}\langle \rho_i(g)\delta_h, \delta_h\rangle X_g + \cdots$$
$$= X_1^{n_i} + \sum_{g\neq 1} X_1^{n_i-1}\left(\sum_{h\in G}\langle \rho_i(g)\delta_h, \delta_h\rangle\right)X_g,$$
d'où l'expression demandée. Les coefficients des termes en $X_gX_1^{n_i-1}$ déterminent donc χ_i, et donc ρ_i. Si Θ_{ρ_i} et Θ_{ρ_j} sont proportionnels, ils sont égaux (le terme dominant en X_1 est égal à 1), donc $\rho_i = \rho_j$.
7. La décomposition de la représentation régulière, proposition 4.5, chap. VII, donne, avec la question 2, la factorisation demandée. Comme les $\Theta_{\rho_i}(G)$ sont deux à deux non proportionnels et irréductibles, c'est bien la factorisation de $\Theta(G)$ en facteurs irréductibles.

Correction de l'exercice VII.10 :

1. Le produit sur G_p est défini par $(b,a)\cdot(b',a') = (b+ab',aa')$. L'inverse est donné par $(b,a)^{-1} = (-a^{-1}b,a^{-1})$. L'élément neutre est $(0,1)$. On peut voir G_p comme $(\mathbb{F}_p,+)\rtimes_\varphi(\mathbb{F}_p^*,\cdot)$, où $\varphi : \mathbb{F}_p^* \to \mathrm{Aut}(\mathbb{F}_p)$ est défini par $\varphi(a) : b \mapsto ab$ (voir [58] pour la définition du produit semi-direct).
2. On a une action de groupe de G_p sur $\mathbb{F}_p$ via $(b,a)\cdot x = a(x+b)$. π est l'action par translation induite sur $\mathbb{C}[\mathbb{F}_p]$. Le fait qu'elle soit unitaire est immédiat, puisque $x \mapsto (b,a)\cdot x$ est une permutation de $\mathbb{F}_p$.
3. $f \in E$ équivaut à $\langle f,\mathbf{1}\rangle = 0$, où l'on a noté $\mathbf{1}$ la fonction constante égale à 1. Comme π est unitaire, on a donc
$$\left\langle f_{(b,a)},\mathbf{1}\right\rangle = \left\langle f_{(b,a)},\mathbf{1}_{(b,a)}\right\rangle = \langle f,\mathbf{1}\rangle = 0,$$
donc $f_{(b,a)} \in E$. On note $\omega_p = e^{2i\pi/p}$ et $e_k : x \mapsto \omega_p^{xk}$. Les e_k sont les caractères additifs de $\mathbb{F}_p$, donc ils forment une base orthogonale de $\mathbb{C}[\mathbb{F}_p]$. Comme on a la décomposition $\mathbb{C}[\mathbb{F}_p] = E \oplus \mathrm{Vect}(e_0)$ (somme orthogonale), les $\{e_k\}_{k=1}^{p-1}$ forment une base orthogonale de E. On a $\pi(b,a)(e_k) = \omega_p^{a^{-1}b}e_{ka^{-1}}$. On note χ le caractère associé à la restriction de π à E. D'après le calcul précédent, si $a \neq 1$, $e_{ka^{-1}} \neq e_k$ et donc $\chi(b,a) = 0$. Si $a = 1$, on a
$$\chi(b,a) = \sum_{k=0}^{p-1}\omega_p^{bk} - 1 = \begin{cases} -1 & \text{si } b \neq 0, \\ p-1 & \text{si } b = 0. \end{cases}$$
On note $\langle\cdot,\cdot\rangle$ le produit scalaire normalisé sur $\mathbb{C}[G_p]$. On a donc
$$|G_p|\,\langle\chi,\chi\rangle = \sum_{b\in\mathbb{F}_p}\chi(b,1)^2 = (p-1)^2 + p - 1 = |G_p|,$$
ainsi, d'après le théorème 4.4, chap. VII, π restreint à E est irréductible.

Correction de l'exercice VII.11 :

1. On a, en utilisant la formule de Plancherel,
$$\mathscr{W}(f)(b,a) = \frac{1}{p}\sum_{n=0}^{p-1}\widehat{f}(n)\omega_p^{bn}\overline{\widehat{\psi}(an)}.$$
2. On note $\Phi(x)$ le membre de droite de l'égalité. On rappelle que, comme f et ψ sont dans E, on a $\widehat{f}(0) = \widehat{\psi}(0) = 0$. On a
$$\begin{aligned}\widehat{\Phi}(k) &= \sum_{(b,a)}\frac{1}{p}\sum_{n=0}^{p-1}\widehat{f}(n)\omega_p^{bn}\overline{\widehat{\psi}(an)}\omega_p^{-kb}\widehat{\psi}(ak) \\ &= \frac{1}{p}\sum_{n=0}^{p-1}\widehat{f}(n)\left(\sum_{b=0}^{p-1}\omega_p^{b(n-k)}\right)\left(\sum_{a=1}^{p-1}\overline{\widehat{\psi}(an)}\widehat{\psi}(ak)\right).\end{aligned}$$
On utilise ensuite le fait que $\sum_{b=0}^{p-1}\omega_p^{b(n-k)} = p\delta_n^k$ ainsi que
$$\sum_{a=1}^{p-1}|\widehat{\psi}(ak)|^2 = \sum_{a=0}^{p-1}|\widehat{\psi}(ak)|^2 = \begin{cases} 0 & \text{si } k = 0, \\ p^2\langle\psi,\psi\rangle & \text{sinon.}\end{cases}$$
On obtient au final
$$\widehat{\Phi}(k) = p^2\langle\psi,\psi\rangle\,\widehat{f}(k).$$

3. En reprenant la démonstration précédente, on a cette fois

$$\sum_{a=1}^{p-1} |\widehat{\psi}(ak)|^2 = \begin{cases} (p-1)|\widehat{\psi}(0)|^2 & \text{si } k=0, \\ \sum_{a=1}^{p-1} |\widehat{\psi}(a)|^2 & \text{sinon.} \end{cases}$$

On a donc $\widehat{\Phi} = d_\psi \widehat{f}$, ce qui donne la formule d'inversion.
On note $\mathscr{W}^* : \mathbb{C}[G_p] \to \mathbb{C}[\mathbb{F}_p]$ l'application définie par

$$\mathscr{W}^* \varphi = \sum_{(b,a)\in G_p} \varphi(b,a)\psi_{b,a}.$$

Il est facile de voir que $\mathscr{W}^*$ est l'adjoint de $\mathscr{W}$ pour les produits scalaires usuels sur $\mathbb{C}[\mathbb{F}_p]$ et $\mathbb{C}[G_p]$, c'est-à-dire que

$$\langle \mathscr{W} f, \varphi \rangle_{\mathbb{C}[G_p]} = \langle f, \mathscr{W}^* \varphi \rangle_{\mathbb{C}[\mathbb{F}_p]}, \quad \text{pour } f \in \mathbb{C}[\mathbb{F}_p] \text{ et } \varphi \in \mathbb{C}[G_p].$$

La formule d'inversion s'écrit ainsi $\mathscr{W}^* \circ \mathscr{W} = d_\psi \mathrm{Id}$, donc $\frac{1}{\sqrt{d_\psi}}\mathscr{W}$ est une isométrie (bien sûr non bijective, la transformée en ondelette étant très « redondante »).
Il est amusant de remarquer que la démonstration de la formule d'inversion sur un corps fini est en tout point semblable à celle de la transformée en ondelettes continue (voir [51]).

4. La procédure MATLAB 7.1 calcule la transformée en ondelette. C'est une procédure lente ($O(p^2)$ opérations), il n'y a pas d'algorithme dichotomique, comme il en existe pour les transformées en ondelettes « classiques » (voir la transformée de Haar, exercice II.4). La procédure 7.2 calcule la transformée inverse (en supposant la condition d'admissibilité $f \in E$ et $\psi \in E$). Ces deux procédures utilisent la fonction `invmod`, programme 5.8.

Programme 7.1 Procédure `transfo_ondelettes`

```
function y = transfo_ondelettes(x,psi)
p = length(psi); y = zeros(p-1,p);
for(a=1:p-1) for(b=0:p-1)
  ordre = mod(invmod(a,p)*((0:p-1)-b), p)+1;
  y(a,b+1) = dot(x,psi(ordre));
end; end;
```

Programme 7.2 Procédure `reconstruct_ondelettes`

```
function y = reconstruct_ondelettes(x,psi)
p = length(psi); c = p*dot(psi,psi); y = zeros(p,1);
for(a=1:p-1) for(b=0:p-1)
  ordre = mod(invmod(a,p)*((0:p-1)-b), p)+1;
  y = y + x(a,b+1)*psi(ordre);
end; end;
y = y/c;
```

8 Correction des exercices du chapitre 8

Correction de l'exercice VIII.1 : On a

$$(\Phi K \Phi^*)_{i,j} = \sum_{s=1}^{p} k_s \chi_i(C_s)\chi_j(C_s) = \sum_{g\in G} \chi_i(g)\chi_j(g) = |G|\delta_i^j,$$

en utilisant la propriété d'orthogonalité des caractères, théorème 3.7, chap. VII. Comme les colonnes d'une matrice unitaire sont orthogonales, on obtient la propriété d'orthogonalité souhaitée.

Correction de l'exercice VIII.2 : La droite (Oz) est stable, donc la représentation n'est pas irréductible. On calcule le caractère χ qui vérifie $\chi(r^k) = 1 + 2\cos(2k\pi/n)$ ainsi que $\chi(sr^k) = -1$. On se place par exemple dans le cas où n est pair. On obtient les coefficients de décomposition

$$\langle \chi, \chi_{\psi_1} \rangle = \langle \chi, \chi_{\psi_3} \rangle = \langle \chi, \chi_{\psi_4} \rangle = 1, \quad \langle \chi, \chi_{\psi_2} \rangle = 1,$$

ainsi que $\langle \chi, \chi_1 \rangle = 1$ et $\langle \chi, \chi_k \rangle = 0$ pour $k \neq 1$.

Correction de l'exercice VIII.3 : Les matrices de transformation s'écrivent

$$\rho((12)) = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \rho((123)) = \frac{1}{2}\begin{pmatrix} -1 & -\sqrt{3} \\ \sqrt{3} & -1 \end{pmatrix}.$$

En ajoutant le caractère trivial et le caractère alterné, on obtient la table suivante :

	1	3	2
	Id	(12)	(123)
χ_1	1	1	1
χ_ε	1	−1	1
χ_ρ	2	0	−1

Correction de l'exercice VIII.4 : On note χ_{so} le caractère de la représentation par permutation des sommets, et χ_{ar} celui de la permutation des arêtes. On a

	1	6	8	6	3
	Id	(12)	(123)	(1234)	(12)(34)
χ_{so}	8	0	2	0	0
χ_{ar}	12	2	0	0	0

On obtient donc les multiplicités suivantes :

	χ_1	χ_ε	χ_s	χ_W	$\chi_{W'}$
χ_{so}	1	1	1	1	0
χ_{ar}	1	0	2	1	1

Correction de l'exercice VIII.5 :

1. $\rho_{W'}((12)(34))$ est une involution (donc est diagonalisable) de trace 2. C'est nécessairement l'identité.
2. Si ρ est triviale sur H, alors $H \subset \ker(\rho)$, donc ρ passe au quotient par H qui est distingué. Réciproquement, si ρ passe au quotient, alors $\rho = \widetilde{\rho} \circ \pi$ qui est trivial sur H puisque π l'est.
3. On a $H = \{\text{Id}, (12)(34), (13)(24), (14)(23)\}$, qui est distingué. L'action de $\mathfrak{S}_4$ sur le cube donne naissance à une permutation des paires de faces opposées, c'est-à-dire à une application $\varphi : \mathfrak{S}_4 \to \mathfrak{S}_3$ (après une numérotation convenable des faces). Il est facile de voir que les permutations qui laissent stables les paires de faces opposées sont les éléments de H. On a donc $\ker(\varphi) = H$ (ce qui montre que H est distingué), et par passage au quotient, un isomorphisme entre $\mathfrak{S}_4/H$ et $\mathfrak{S}_3$.

4. D'après la question 1, $\rho_{W'}$ est triviale sur H, le groupe engendré par $(12)(34)$. Donc avec la question 2, $\rho_{W'}$ s'identifie à un élément de $\widehat{\mathfrak{S}_4/H}$, c'est-à-dire de $\widehat{\mathfrak{S}_3}$. Comme cette représentation est irréductible, par considération de dimension, c'est nécessairement la représentation standard.

Correction de l'exercice VIII.6 :

1. On conserve les notations de la correction de l'exercice VII.7. On note $Gx_1, \dots, Gx_t$ les orbites, avec $x_1 = e$ l'élément neutre de G. Les orbites formant une partition de X, on a l'équation aux classes $1 + \sum_{i=2}^{t} |Gx_i| = |X| = |G|^p = 0 \mod p$, puisque $p||G|$. Mais comme les $|Gx_i|$ divisent p, on a $|Gx_i| \in \{1, p\}$, et l'égalité précédente impose qu'au moins un $x_i = (g, \dots, g)$ vérifie $|Gx_i| = 1$. Ceci signifie exactement que g est d'ordre p.
2. Avec le résultat de VII.8, on a que 2 divise $|G|$, donc avec la question précédente, G possède un élément t d'ordre 2.
3. $\varphi \overset{\text{déf.}}{=} \det \circ \rho : G \mapsto \mathbb{C}^*$ est une représentation de degré 1 dont le noyau est un groupe simple non réduit à l'élément neutre de G (car si $\ker(\varphi) = \{1\}$, alors φ est injectif et G est commutatif). Comme G est simple, on a $\ker(\varphi) = G$, et $\det(\rho(g)) = 1$, donc ρ est bien à valeur dans $SL_2(\mathbb{C})$. Comme $X^2 - 1$ est le polynôme minimal de $\rho(t)$, ce dernier est diagonalisable. Comme $\det(\rho(t)) = 1$, et que $\rho(t) \neq \mathrm{Id}$ (car ρ est injectif puisque $\ker(\rho)$ est un sous-groupe distingué de G), ses deux valeurs propres sont égales à -1. On peut donc trouver $P \in GL_2(\mathbb{C})$ telle que $P\rho(t)P^{-1} = -\mathrm{Id}_2$, et donc $\rho(t) = -\mathrm{Id}_2$. De plus, pour tout $g \in G$, on a $\rho(gtg^{-1}) = \rho(g)(-\mathrm{Id}_2)\rho(g)^{-1} = \rho(t)$. Mais ρ est injectif, donc $gtg^{-1} = t$, et $t \in Z(G)$, le centre de G. Comme $Z(G)$ est distingué, on a $Z(G) = \{1\}$, ce qui est une contradiction, car $t \neq 1$.

Correction de l'exercice VIII.7 :

1. Il faut montrer que φ est un morphisme d'algèbre. La linéarité est évidente. Il reste à vérifier les relations sur les générateurs, par exemple $\varphi(jk) = \varphi(i)$.
2. Il suffit de constater que l'application $\varphi(q) \mapsto \psi(q)$ est un isomorphisme d'algèbre. Ceci est évident, puisque $a + ib \mapsto \left(\begin{smallmatrix} a & -b \\ b & a \end{smallmatrix}\right)$ est un isomorphisme d'algèbre de $\mathbb{C}$ dans $\mathrm{Sim}(\mathbb{R}^2)$ (les similitudes de $\mathbb{R}^2$). Ceci montre aussi que la représentation obtenue est unitaire. De plus, on vérifie que $\|\chi_\psi\|_2 = 1$, donc cette représentation est irréductible.
3. On a la représentation triviale ρ_1 ainsi que les représentations suivantes :

$$\begin{aligned} &\rho_2(\pm 1) = \rho_2(\pm i) = 1, \quad \rho_2(\pm j) = \rho_2(\pm k) = -1, \\ &\rho_3(\pm 1) = \rho_3(\pm j) = 1, \quad \rho_3(\pm i) = \rho_3(\pm k) = -1, \\ &\rho_4(\pm 1) = \rho_4(\pm k) = 1, \quad \rho_4(\pm i) = \rho_4(\pm j) = -1. \end{aligned}$$

On vérifie que ces représentations sont bien irréductibles, et si on note ρ_5 la représentation de la question précédente, et n_i les dimensions des représentations, on a $\sum n_i^2 = 4 \times 1^2 + 2^2 = |H_8|$, donc on a bien toutes les représentations irréductibles. On fixe un ordre parmi les éléments de H_8, et on note les entrées des matrices H_8 sous forme de vecteurs de taille 8, ce qui donne

$$\begin{aligned} v_0 &= (1,1,1,1,1,1,1,1), & v_4 &= \sqrt{2}(1,-1,0,0,0,0,\mathrm{i},-\mathrm{i}), \\ v_1 &= (1,1,1,1,-1,-1,-1,-1), & v_5 &= \sqrt{2}(0,0,-1,1,-\mathrm{i},\mathrm{i},0,0), \\ v_2 &= (1,1,-1,-1,1,1,-1,-1), & v_6 &= \sqrt{2}(0,0,1,-1,-\mathrm{i},\mathrm{i},0,0), \\ v_3 &= (1,1,-1,-1,-1,-1,1,1), & v_7 &= \sqrt{2}(1,-1,0,0,0,0,-\mathrm{i},\mathrm{i}), \end{aligned}$$

et forme une base orthogonale de $\mathbb{C}^8$.

4. L'exercice II.7 explique comment on peut construire, par produit tensoriel, une base orthogonale de $\mathbb{C}^{8n}$ à partir d'une base de $\mathbb{C}^8$. La procédure MATLAB 2.6 permet de calculer, pour $f \in \mathbb{C}^{8n}$, $(A^{\otimes n})f$, les coefficients de f dans la base orthonormée construite. Les lignes de A sont les vecteurs v_i.

Correction de l'exercice VIII.8 :

1. Les éléments de G sont des rotations, elles conservent la distance à l'origine, donc le polynôme $X^2+Y^2+Z^2$.
2. On note $V(XYZ) = \{(x,y,z) \in \mathbb{R}^3 \setminus xyz = 0\}$. Comme les éléments de G conservent l'ensemble des faces du cubes, ils conservent l'union des trois plans de coordonnées, c'est-à-dire $V(XYZ)$.
   On note $I(V(XYZ)) = \{P \in K[X,Y,Z] \setminus \forall (x,y,z) \in V(XYZ), P(x,y,z) = 0\}$. Soit alors $P \in I(V(XYZ))$. En faisant la division euclidienne de P par X en tant que polynôme en X (ce qui est possible car le coefficient dominant de X est inversible dans $K[Y,Z]$), on écrit $P(X,Y,Z) = XQ(X,Y,Z) + R(Y,Z)$. Comme $P(0,Y,Z) = 0$, on a $R = 0$. En continuant avec les variables Y et Z, on trouve $P = \lambda XYZ$ avec $\lambda \in \mathbb{R}$.
   Soit $A \in G$. Comme $V(XYZ)$ est stable par A, on a, pour $(x,y,z) \in V(XYZ)$, l'égalité $f(A \cdot (x,y,z) = 0$, c'est-à-dire $f(A \cdot (X,Y,Z)) \in I(V(XYZ))$.
   On obtient donc $f(A \cdot (X,Y,Z)) = \lambda f$. Comme $A^n = \mathrm{Id}$ pour un certain n, on a nécessairement $\lambda = \pm 1$.
3. Cette fois-ci, $V(f)$ est l'union des 4 plans orthogonaux aux trois grandes diagonales. Comme ces diagonales sont stables par G, on en déduit que $V(f)$ est stable par G. On peut faire le même raisonnement qu'à la question précédente, en commençant cette fois une division euclienne par $X+Y+Z$.
   De même, $V(g)$ est l'union des 6 plans orthogonaux aux 6 paires de diagonales opposées inscrites dans les faces du cubes. Une fois de plus, $V(g)$ est stable par G.

Correction de l'exercice VIII.9 :

1. n doit être pair et on a $k = n/2$.
2. Comme le polynôme $W_{\mathscr{C}}$ est homogène de degré n, l'identité de MacWilliams se réécrit $\mathscr{W}_{\mathscr{C}}(A \cdot (X,Y)) = \mathscr{W}_{\mathscr{C}}(X,Y)$.
3. On a $A^2 = \mathrm{Id}_2$ donc $G_1 = \{A, \mathrm{Id}_2\}$. En appliquant l'opérateur de Reynolds, on trouve

$$R_G(X) = \frac{\sqrt{2}+1}{2\sqrt{2}}(X + (\sqrt{2}-1)Y), \quad R_G(X^2) = \frac{1}{2}(X^2 + (X+Y)^2/2).$$

   On peut enlever la constante multiplicative devant $R_G(X)$ et soustraire $3/4 R_G(X)^2$ à $R_G(X^2)$ pour obtenir les deux invariants de $K[X,Y]^{G_1}$ annoncés. Pour montrer que ce sont les seuls, on peut utiliser la série de Molien calculée à l'exercice VII.6, question 5, ou bien calculer $R_G(XY)$, $R_G(Y)$ et $R_G(Y^2)$ pour voir qu'ils s'écrivent en fonction des invariants déjà trouvés.
4. Si v est un mot du code, on a $\langle v, v\rangle = \langle v, 1\rangle = 0$, donc $\sum v_i = 0 \mod 2$. Ainsi, $\mathscr{C}$ ne contient que des mots de poids pair, et $\mathscr{W}_{\mathscr{C}}(-X,-Y) = \mathscr{W}_{\mathscr{C}}(X,Y)$. Donc $\mathscr{W}_{\mathscr{C}}$ est invariant sous l'action de A et de $-\mathrm{Id}$.

5. Le programme 8.1 permet de calculer des générateurs de l'anneau des invariants en essayant tous les $R_G(X^k)$, pour $|k| \leqslant |G|$. Les temps de calcul deviennent très importants pour un groupe conséquent. L'explosion combinatoire est double, à la fois au niveau du cardinal du groupe et de l'ensemble des X^k pour $k \leqslant |G|$.

Programme 8.1 Fichier `polynomes-invariants.msw`

Applique une matrice a un polynome :

```
> action_matrice := proc(A,p)
>    local g,v,m;
>    v := array([[x],[y]]);
>    m := evalm(A&*v);
>    g := subs ({x=m[1,1],y=m[2,1]},p);
>    return(g)
> end proc:
```

Un exemple invariant :

```
> A:=1/sqrt(2)*matrix(2,2,[[1,1],[1,-1]]);
> expand( action_matrice(A,x^2+y^2) );
```

$$A := \frac{1}{2}\sqrt{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$x^2+y^2$$

Calcule l'opérateur de Reynolds :

```
> operateur_reynolds := proc(G,p)
>    local i,r;
>    r := (1/nops(G))*sum('action_matrice(G[k],p)','k'=1..nops(G));
>    return(r)
> end proc:
```

Calcule des générateurs de l'anneau des invariants :

```
> polynomes_invariants := proc(G)
>    local i,j,r;
>    r := [];
>    for i from 1 to nops(G) do
>    for j from 0 to i do
>       r := [op(r),expand(operateur_reynolds(G,x^j*y^(i-j)))];
>    end do: end do:
> end proc:
```

Un exemple en rapport aux codes auto-duaux, pour le cas où 2 divise les poids des mots du code :

```
> B:=matrix(2,2,[[-1,0],[0,-1]]):
> G:=[A,-A,B,-B]:
> polynomes_invariants( G, 4 );
```

$$[0,0,\frac{1}{4}x^2-\frac{1}{2}xy+\frac{3}{4}y^2,\frac{1}{4}x^2-\frac{1}{4}y^2+\frac{1}{2}xy,\frac{3}{4}x^2+\frac{1}{2}xy+\frac{1}{4}y^2,0,0,0,0,$$
$$\frac{1}{8}x^4-\frac{1}{2}x^3y+\frac{3}{4}x^2y^2-\frac{1}{2}xy^3+\frac{5}{8}y^4,\frac{1}{8}x^4-\frac{1}{4}x^3y+\frac{3}{4}xy^3-\frac{1}{8}y^4,$$
$$\frac{1}{8}x^4+\frac{1}{4}x^2y^2+\frac{1}{8}y^4,\frac{1}{8}x^4+\frac{3}{4}x^3y-\frac{1}{4}xy^3-\frac{1}{8}y^4,$$
$$\frac{5}{8}x^4+\frac{1}{2}x^3y+\frac{3}{4}x^2y^2+\frac{1}{2}xy^3+\frac{1}{8}y^4]$$

Annexe A
Programmes MATLAB

Voici l'ensemble des programmes MATLAB évoqués dans les chapitres précédents. Chaque programme constitue un fichier à part entière. La plupart sont des *procédures*, cela signifie que ces programmes doivent être recopiés dans un fichier portant le même nom. Par exemple, la procédure `fht` est écrite dans le fichier `fht.m`.

1 Algorithme FWT

Le programme `fwt` est une implémentation MATLAB de l'algorithme de *transformée de Walsh rapide* présenté au paragraphe 2.2, chap. II. Il est récursif, mais n'utilise pas de mémoire supplémentaire, donc est relativement efficace. Il est à noter qu'à un facteur $1/N$ près, la transformée de Walsh est sa propre inverse, donc la routine n'inclut pas de paramètre pour calculer la transformée inverse (il suffit de diviser le résultat par N).

Programme 1.1 Procédure `fwt`

```
function y = fwt(x)
N = length(x); % N doit être une puissance de 2
if(N==1) y = x; return; end;
P = N/2;
x = [fwt(x(1:P)) ; fwt(x((P+1):N))];
y = zeros(N,1);
y(1:P) = x(1:P) + x((P+1):N);
y((P+1):N) = x(1:P) - x((P+1):N);
```

2 Algorithme FHT

Le paragraphe 1.2, chap. V, expose le fonctionnement de l'algorithme de *transformée de Hartley rapide*. Voici une implémentation MATLAB de cet algorithme.

- Procédure `fht` (programme 2.1) : l'algorithme FHT proprement dit. Pour calculer la transformée de Hartley inverse, il suffit d'utiliser la routine `fht` et de diviser le résultat par N, la longueur de l'échantillon.
- Procédure `operateur_chi` (programme 2.2) : permet de calculer l'opérateur χ_N^x.
- Procédure `fht_convol` (programme 2.3) : permet de calculer la convolution de deux signaux réels à l'aide de l'algorithme FHT.

Programme 2.1 Procédure `fht`

```
function y = fht(f)
% N doit être une puissance de 2
N = length(f); N1 = N/2;
if( N==1 ) y = f; return; end;
y = zeros(size(f));
% construction des deux sous-vecteurs
f_p = f(1:2:N); f_i = f(2:2:N);
% appels récursifs
f_p = fht(f_p); f_i = fht(f_i);
% application de l'opérateur chi
f_i = operateur_chi(f_i,0.5);
% mixage des deux résultats
y(1:N1) = f_p + f_i; y((N1+1):N) = f_p - f_i;
```

Programme 2.2 Procédure `operateur_chi`

```
function y = operateur_chi(a,x)
N = length(a); a_inv = [a(1);a(N:-1:2)];
y = a.*cos( 2*pi*x*(0:N-1)'/N ) + a_inv.*sin( 2*pi*x*(0:N-1)'/N );
```

Programme 2.3 Procédure `fht_convol`

```
function y = fht_convol(x,y)
% N doit être une puissance de 2
N = length(x); y = zeros(size(x));
a = fht(x); b = fht(y);
a_inv = [a(1);a(N:-1:2)];
b_inv = [b(1);b(N:-1:2)];
y = 0.5*( a.*b - a_inv.*b_inv + a.*b_inv + a_inv.*b );
y = fht(y)/N;
```

3 Algorithme FFT

Voici les différentes implémentations de l'algorithme FFT présentées en détail aux paragraphes 2.1, chap. III, 2.5, chap. III et 2.6, chap. III.

- Procédure `fft_rec` (programme 3.1) : version naïve et récursive de l'algorithme (décimation temporelle).
- Procédure `fft_dit` (programme 3.2) : implémentation efficace (à la fois en utilisation mémoire et en rapidité) de l'algorithme (non récursive et décimation temporelle).
- Procédure `fft_dif` (programme 3.3) : version décimation fréquentielle de l'algorithme.
- Procédure `operateur_s` (programme 3.4) : implémentation de l'opérateur $\mathscr{S}$.
- Procédure `rev_bits` (programme 3.5) : classe le vecteur selon le sens inverse des bits des indices.
- Procédure `rev_index` (programme 3.6) : calcule l'entier obtenu par inversion des bits d'un autre entier.

Il faut garder à l'esprit que ces programmes MATLAB ont avant tout un but pédagogique. L'implémentation est loin d'être aussi efficace que celles que l'on peut réaliser dans un langage rapide (par exemple en C). Il existe de nombreux logiciels disponibles sur internet qui sont très performants, par exemple *FFTW* [33].

Programme 3.1 Procédure `fft_rec`

```
function y = fft_rec(f,dir)
% N doit être une puissance de 2
N = length(f); N1 = N/2;
if( N==1 ) y = f; return end;
y = zeros(size(f));
% construction des deux sous-vecteurs
f_p = f(1:2:N); f_i = f(2:2:N);
% appels récursifs
f_p = fft_rec(f_p,dir);
f_i = fft_rec(f_i,dir);
% application de l'opérateur S
f_i = operateur_s(f_i,dir*0.5);
% mixage des deux résultats
y(1:N1) = f_p + f_i; y((N1+1):N) = f_p - f_i;
```

Programme 3.2 Procédure `fft_dit`

```
function y = fft_dit(f,dir)
% N doit être une puissance de 2
N = length(f); ldn = floor(log2(N));
f = rev_bits(f);
for ldm=1:ldn
  m = 2^ldm; m1 = m/2;
  for j=0:m1-1
    e = exp(-dir*2.0i*pi*j/m);
    for r=0:m:N-m
      u = f(r+j+1); v = f(r+j+m1+1)*e;
      f(r+j+1) = u + v; f(r+j+m1+1) = u - v;
    end
  end
end
y = f;
```

4 Multiplication de grands entiers par FFT

Les programmes MATLAB qui suivent permettent de multiplier des grands entiers représentés par leur décomposition dans une base b donnée.

- Procédure `mult_entiers` (programme 4.1) : permet de multiplier deux entiers représentés sous forme de vecteurs. On doit bien sûr fournir la base utilisée.
- Procédure `number2vector` (programme 4.2) : fonction pratique qui permet de passer de la représentation sous forme de nombre entier à la représentation sous forme de vecteur (d'un intérêt limité cependant, car MATLAB ne manipule pas des entiers de taille arbitraire).
- Procédure `vector2number` (programme 4.3) : fonction inverse de la précédente.
- Fichier `test_mult_entiers.m` (programme 4.4) : petit programme de test.

5 Résolution de l'équation de Poisson

Voici les différents fichiers pour implémenter la résolution de l'équation de Poisson décrite au paragraphe 4.3, chap. IV.

- Fichier `poisson.m` (programme 5.1) : fichier principal, construit les différentes ma-

Programme 3.3 Procédure `fft_dif`

```
function y = fft_dif(f,dir)
% N doit être une puissance de 2
N = length(f); ldn = floor(log2(N));
for ldm=ldn:-1:1
  m = 2^ldm; m1 = m/2;
  for j=0:m1-1
    e = exp(-dir*2.0i*pi*j/m);
    for r=0:m:N-1
      u = f(r+j+1); v = f(r+j+m1+1);
      f(r+j+1) = u + v; f(r+j+m1+1) = (u-v)*e;
    end
  end
end
% remet le vecteur transformé dans le bon ordre
y = rev_bits(f);
```

Programme 3.4 Procédure `operateur_s`

```
function y = operateur_s(a,x)
N = length(a);
y = a.*exp( -2.0i*x*(0:N-1)'*pi/N );
```

Programme 3.5 Procédure `rev_bits`

```
function y = rev_bits(x)
n = length(x); t = floor(log2(n)); y = zeros(n,1);
for i=0:n-1
  j = rev_index(t,i); y(j+1) = x(i+1);
end
```

trices, calcule les FFT en 2D et résout l'équation de convolution.

- Procédure `f` (programme 5.2) : le membre de droite de l'équation. Il s'agit de la fonction $f(x,y) = (x^2+y^2)e^{xy}$. Elle est calculée pour que la solution soit connue à l'avance.

- Procédure `sol` (programme 5.3) : la solution exacte de l'équation (on triche un peu, on utilise une équation dont on connaît déjà la solution !). On a pris $s(x,y) = e^{xy}$. Son laplacien est donc la fonction f du fichier `f.m`.

- Procédure `u_0y`, `u_1y.m`, `u_x0.m` et `u_x1.m` (programme type : 5.4) : valeur de la solution u sur chacun des bords $x = 0$, $x = 1$, $y = 0$ et $y = 1$. On n'a reporté que le programme de la fonction `u_0y.m`, les autres s'écrivant de la même manière.

Le choix de la fonction solution est parfaitement arbitraire, on pourra faire des essais avec d'autres fonctions (en prenant soin de mettre à jour la valeur du laplacien dans le fichier `f.m`). On pourra faire des essais avec des conditions sur le bord arbitraires (mais on n'aura plus de solution exacte de référence ...). De même, il est facile de changer la précision de la résolution pour observer la convergence de l'erreur commise par la méthode des différences finies.

Programme 3.6 Procédure `rev_index`

```
function y = rev_index(t,index)
y = 0; tmp = index;
for i=0:t-1
  bit = mod(tmp,2);
  tmp = floor(tmp/2);
  y = y*2 + bit;
end
```

Programme 4.1 Procédure `mult_entiers`

```
function r = mult_entier(x,y,b)
N = length(x);
% ajout de zéros pour convolution acyclique
x = [x;zeros(N,1)]; y = [y;zeros(N,1)];
% calcule la convolution
r = round( real( ifft(fft(x).*fft(y)) ) );
for i=1:2*N-1 % enlève les retenues
  q = floor(r(i)/b);
  r(i) = r(i)-q*b; r(i+1) = r(i+1)+q;
end
```

Programme 4.2 Procédure `number2vector`

```
function y = number2vector(x,b)
N = floor( log(x)/log(b) )+1; y = zeros(N,1);
for i=1:N
  q = floor(x/b); y(i) = x - q*b; x = q;
end
```

6 Résolution de l'équation de la chaleur

Les programmes qui suivent permettent de résoudre l'équation de la chaleur par la méthode décrite au paragraphe 4.2, chap. IV.

- Fichier `chaleur.m` (programme 6.1) : calcule la solution de l'équation pour différentes valeurs de temps en appelant le programme `solve_eq.m`, puis dessine l'évolution de la solution.
- Procédure `solve_eq` (programme 6.2) : résout l'équation de la chaleur pour un temps donné en calculant les coefficients de Fourier par FFT.
- Procédure `f` (programme 6.3) : la répartition initiale de la chaleur au temps $t = 0$. On a pris ici une fonction échelon (donc discontinue).

Bien sûr, il est facile de modifier ces programmes, notamment pour faire des essais avec différentes conditions initiales, ainsi qu'avec d'autres valeurs du paramètre temps.

Programme 4.3 Procédure `vector2number`

```
function y = vector2number(v,b)
N = length(v); y = sum( v.*( b.^(0:N-1)' ) );
```

Programme 4.4 Fichier `test_mult_entiers.m`

```
x = 262122154512154212; y = 314134464653513212;
b = 20; % la base
xx = number2vector(x,b); yy = number2vector(y,b);
zz = mult_entiers(xx,yy,b); z = vector2number(zz,b);
z - x*y % le résultat doit valoir zéro
```

Programme 5.1 Fichier `poisson.m`

```
% quelques constantes
N = 30; h = 1/N; nb_iter = 30;
M = zeros(N+1, N+1); f_val = zeros(N-1, N-1);
% on commence avec x=h (seulement les points du centre)
for i=1:N-1 % calcul du membre de droite
  for j=1:N-1
    x = i*h; y = j*h;
    f_val(i,j) = f(x,y);
  end
end
for i=1:N-1 % ajout des termes de bord
  x = i*h;
  f_val(i,1) = f_val(i,1) - 1/h^2 * f_0y(x);
  f_val(i,N-1) = f_val(i,N-1) - 1/h^2 * f_1y(x);
  f_val(1,i) = f_val(1,i) - 1/h^2 * f_x0(x);
  f_val(N-1,i) = f_val(N-1,i) - 1/h^2 * f_x1(x);
end
% on rend la matrice impaire
ff = [zeros(N-1,1),f_val,zeros(N-1,1),-f_val(:,N-1:-1:1)];
ff = [zeros(1,2*N); ff; zeros(1,2*N); -ff(N-1:-1:1,:)];
ff = fft2(ff); % on calcule la FFT
d = -4/h^2 * sin( (0:2*N-1) * pi/(2*N)).^2;
for i=1:2*N % résolution du système d*u+u*d = ff
  for j=1:2*N
    s = d(i) + d(j);
    if( s==0 ) s=1; end; % éviter la division par 0
    ff(i,j) = ff(i,j) / s;
  end
end
ff = real( ifft2( ff ) ); % on calcule la transformée inverse
% on extrait la solution
u = zeros(N+1,N+1); u(2:N, 2:N) = ff(2:N,2:N);
for i=1:N+1 % on remet les termes du bord
  x = (i-1)*h;
  u(i,1) = f_0y(x); u(i,N+1) = f_1y(x);
  u(1,i) = f_x0(x); u(N+1,i) = f_x1(x);
end
surf(u); title('Résolution par FFT');
```

Programme 5.2 Procédure `f`

```
function r = f(x,y)
r = (x^2+y^2)*exp(x*y);
```

Programme 5.3 Procédure `sol`

```
function r = sol(x,y)
r = exp(x*y);
```

Programme 5.4 Procédure `u_0y`

```
function r = f_0y(y)
r = sol(0,y);
```

Programme 6.1 Fichier `chaleur.m`

```
% nombre de points d'interpolation pour le calcul de l'intégrale
M = 2^8; h = 1/M;
% valeur de f aux points d'interpolation
f_val = zeros(M,1);
for( i=1:M ) f_val(i) = f((i-1)*h); end;
% calcul de la fft
dft_val = fft(f_val);
% calcul des coefficients de fourier
fcoef = zeros(M,1);
for n=1:M
  i = 1 + mod(-(n-1),M); % il faut renverser les indices
  fcoef(n) = h*dft_val(i);
end
% dessine une évolution de la solution
for t = [0.01,0.02,0.03,0.04,0.05,0.06]
  xx = [xx, real( solve_eq(0, fcoef) )];
end
plot(xx);
```

Programme 6.2 Procédure `solve_eq`

```
function u = solve_eq(t, fcoef_val)
prec = 300; h = 1/prec; % précision du tracé
u= zeros(prec+1, 1);
M = length(fcoef_val); % taille de la solution
v = [0:M/2,-M/2+1:-1]'; % fréquences des coefficients
% calcule la solution
for i=0:prec
  x = i*h;
  w = exp(-2.0*pi*pi*t*v.*v + 2.0i*pi*x*v) .* fcoef_val;
  u(i+1) = sum(w);
end
```

Programme 6.3 Procédure `f`

```
function y = f(x)
if( x<0.3 ) y = 0;
elseif( x<0.7) y = 1;
else y = 0;
end
```

Annexe B
Programmes MAPLE

Ce chapitre rassemble l'ensemble des programmes MAPLE du livre. Chaque programme constitue un fichier `.dsw` à part entière. Ils ont souvent été coupés en plusieurs morceaux par souci de clareté.

1 Transformée sur un corps fini

Le fichier `fft-corps-fini.msw` réalise successivemente

1. une recherche des facteurs irréductibles de $X^n - 1$ sur un corps fini $\mathbb{F}_p$ (on a pris $p = 2$). Par la commande `alias`, on nomme α une racine primitive de l'unité.
2. une implémentation naïve de la transformée de Fourier sur le corps cyclotomique $\mathbb{F}_{p^r}$. Dans le cas où n est de la forme 2^s, il est possible d'implémenter une version récursive de l'algorithme. Ceci est fait pour la transformée de Fourier sur un anneau, appendice 2.
3. un test sur un vecteur $f \in \mathbb{F}_p^n$ tiré au hasard. On peut constater que $\widehat{f} \notin \mathbb{F}_p^n$, puisqu'on est obligé de faire les calculs dans une extension cyclotomique de $\mathbb{F}_p$.

2 Transformée sur un anneau

Le programme MAPLE `fft-anneau.msw` calcule une transformée de taille n à valeur dans un anneau $\mathbb{Z}/m\mathbb{Z}$ pour un entier m judicieusement choisi (conformément aux explications données au paragraphe 2, chap. VI). On a choisi n de la forme 2^s, ce qui permet d'implémenter un algorithme récursif de type FFT. On utilise une fonction intermédiaire, `FFT_rec`, qui permet de mettre à jour à chaque appel la racine principale de l'unité.

3 Multiplication de grands entiers

Le programme MAPLE `mult-grands-entiers.mws` permet de calculer le produit de deux entiers représentés par leur décomposition dans une base b donnée. Ce programme utilise les constantes n et m ainsi que la fonction `xFFT` qui se trouve dans le fichier `fft-anneau.msw`, 2.1. Voici les différentes choses que l'on peut trouver dans ce programme.

1. On calcule d'abord une valeur de b optimale, de façon à satisfaire à $n(b-1)^2 < m$.

Programme 1.1 Fichier `fft-corps-fini.msw`

Les paramètres pour faire un TFD de taille n fixé sur $\mathbb{F}_p$:

```
> with(numtheory): n := 16: p := 3:
```

Liste des facteurs de $X^n - 1$. Choix d'un facteur irréductible de degré r et de la racine primive associée : on constate que r est bien l'ordre de p dans $(\mathbb{Z}/n\mathbb{Z})^*$.

```
> liste_div := op(Factor( cyclotomic(n,X) ) mod p );
> P := liste_div[1];
> alias( alpha = RootOf(P) ):
```

$$liste\_div := X^4+2X^2+2, X^4+X^2+2$$

$$P := X^4+2X^2+2$$

Transformée de Fourier, version $O(n^2)$:

```
> TFD := proc(f, signe)
>    local res, formule;
>    # pour plus de lisibilité, on écrit à part la formule de TFD :
>    formule := 'f[l+1]*alpha^(-signe*(k-1)*l)';
>    res := [ seq( sum( formule, 'l'=0..n-1 ) mod p , k=1..n)  ];
>    if signe=-1 then res := 1/n*res mod p end if;
>    return(Normal(res) mod p);
> end proc:
```

Test simple :

```
> hasard := rand(0..(p-1)):
> x := [seq( hasard(), i=1..n )];
> y := TFD(x,1);  # TFD(x) n'est plus à coefficients dans F_2.
> evalb( x = TFD(y,-1) ); # Mais on retombe bien sur nos pattes.
```

$$x := [0,2,0,2,1,2,2,2,1,1,1,0,0,2,2,1]$$

$$y := [1,2\alpha^3+2\alpha+\alpha^2+2,1,2\alpha^3+\alpha+2\alpha^2,\alpha^2+1,\alpha^3+\alpha,1,2\alpha^3+\alpha,1,$$
$$\alpha^3+\alpha+\alpha^2+2,2,\alpha^3+2\alpha+2\alpha^2,2\alpha^2+2,2\alpha^3+2\alpha,2,\alpha^3+2\alpha]$$

true

2. Ensuite plusieurs fonctions très utiles sont définies (pour passer de la représentation sous forme de nombre à celle sous forme de vecteur).
3. La fonction `prod_entiers` calcule le produit de convolution des deux vecteurs, puis propage les retenues.
4. Enfin, un test est effectué. Bien sûr, l'utilité de ces fonctions est de multiplier des nombres entiers que MAPLE ne sait pas manipuler (car trop grands), ce qui n'est pas le cas dans ce test (car on fait vérifier à MAPLE que le produit est juste).

4 Décodage des codes BCH

Ce programme MAPLE utilise la fonction `FFT` définie dans le programme 1. Il faudra donc recopier cette procédure au début du programme. Le programme a été découpé en trois parties :

– Partie 1 (programme 4.1) : recherche des facteurs irréductibles de $X^n - 1$ sur $\mathbb{F}_2$, et construction du polynôme générateur du code BCH.
– Partie 2 (programme 4.2) : définition de routines pour manipuler des mots du code à la fois sous forme de vecteurs et de polynômes, pour générer des mots au hasard.
– Partie 3 (programme 4.3) : la première partie de l'algorithme de décodage, on calcule les valeurs de $\sigma_1, \ldots, \sigma_t$.

Programme 2.1 Fichier `fft-anneau.msw`

Définition des paramètres de la transformée

```
> s := 4: n := 2^s: m := 2^(2^(s-1)) + 1:
```

Sous-procédure récursive :

```
> FFT_rec := proc(f, signe, zeta)
>    local nn, n1, s, t, r;
>    nn := nops(f); n1 := nn/2; # taille du vecteur
>    if nn=1 then return(f) end if; # fin de l'algorithme
>    # construction des deux sous-vecteurs de taille n1
>    s := [ seq(f[2*k+1], k=0..n1-1) ];
>    t := [ seq(f[2*k], k=1..n1) ];
>    # calcul des deux sous-FFT :
>    s := FFT_rec(s, signe, zeta^2 mod m);
>    t := FFT_rec(t, signe, zeta^2 mod m);
>    # mixage des deux résultats
>    a := seq( s[k]+zeta^(-signe*(k-1))*t[k] mod m, k=1..n1 );
>    b := seq( s[k]-zeta^(-signe*(k-1))*t[k] mod m, k=1..n1 );
>    r := [a,b];
>    return(r);
> end proc:
```

Procédure principale (attention, le nom FFT est protégé en MAPLE ...) :

```
> xFFT := proc(f, signe)
>    local r;
>    r := FFT_rec(f,signe,2);
>    if signe=-1 then r := 1/n*r mod m;
>    else r; end if
> end proc:
```

Un test :

```
> hasard := rand(0..m-1):
> x := [seq( hasard(), i=1..n )];
> y := xFFT(x,+1);
> evalb( x = xFFT(y,-1) ); # On retombe bien sur nos pattes.
```

$$x := [179, 220, 230, 218, 49, 253, 197, 218, 67, 177, 136, 127, 190, 106, 210, 255]$$

$$y := [5, 250, 28, 179, 190, 157, 195, 216, 198, 11, 13, 43, 5, 59, 49, 238]$$

true

– Partie 4 (programme 4.4) : la deuxième partie de l'algorithme de décodage, on calcule les valeurs de $\widehat{\varepsilon_0}, \widehat{\varepsilon_{2t+1}}, \ldots, \widehat{\varepsilon_{n-1}}$.

Programme 3.1 Fichier `mult-grands-entiers.msw`

b désigne la base de calcul. Il faut que $n(b-1)^2 < m$.

```
> b := floor( evalf(sqrt(m/n))+1 ):
```

Calcule le produit point à point :

```
> cw_mult := proc(a,b)
>    [seq( a[i]*b[i], i=1..n )]:
> end proc:
```

Transforme un entier en vecteur :

```
> number2vector := proc(x)
>    local N, res, i, r, q, xx:
>    N := floor( log(x)/log(b) )+1;
>    res := []: xx := x:
>    for i from 1 to N do
>       xx := iquo(xx,b,'r'):
>       res := [op(res), r]:
>    end:
>    return(res):
> end proc:
```

Transforme un vecteur en entier :

```
> vector2number := proc(v)
>    add(v[k]*b^(k-1), k=1..nops(v));
> end proc:
```

Calcule le produit de convolution :

```
> convol := proc(f,g)
>    xFFT( cw_mult(xFFT(f,1),xFFT(g,1)), -1):
> end proc:
```

Calcule le produit de deux entiers représentés sous forme de vecteurs de taille n. Attention, les $n/2$ dernières entrées des vecteurs doivent être nulles.

```
> prod_entiers := proc(x,y)
>    local res, i:
>    res := convol(x,y):
>    for i from 1 to n-1 do
>       res[i] := irem(res[i],b,'q'):
>       res[i+1] := res[i+1]+q;
>    end:
>    return(res):
> end proc:
```

Un test :

```
> hasard := rand(0..b-1):
> xx := [seq( hasard(), i=1..n/2 ), seq(0, i=1..n/2)];
> yy := [seq( hasard(), i=1..n/2 ), seq(0, i=1..n/2)];
> x := vector2number(xx): y := vector2number(yy);
> zz := prod_entiers(xx,yy);
> evalb( vector2number(zz) = x*y ); # il doit y avoir égalité ...
```

$$xx := [4,0,0,3,3,1,0,4,0,0,0,0,0,0,0,0]$$

$$yy := [3,0,4,1,4,2,3,0,0,0,0,0,0,0,0,0]$$

$$y := 55853$$

$$zz := [2,2,1,1,3,3,2,2,1,0,3,3,2,4,2,0]$$

true

Programme 4.1 Fichier `decodage-bch.msw` partie 1

r : degrés des facteurs irréductibles de $X^n - 1$ sur $\mathbb{F}_2$; t : capacité de correction.

```
> with(numtheory): with(linalg):
> n := 15: t := 3: delta:=2*t+1:
```

Liste des facteurs de $X^n - 1$choix d'un facteur irréductible de degré r et de la racine primive associée : on constate que r est bien l'ordre de p dans $(\mathbb{Z}/n\mathbb{Z})^*$.

```
> liste_div := op(Factor( X^n-1 ) mod 2 ):
> P := liste_div[2];
> alias( alpha = RootOf(P) ):
```

$$P := X^4 + X^3 + 1$$

Calcule le polynôme générateur du code de distance prescrite $2t+1$, le PPCM des polynômes minimaux des α^i, pour $i = 1, \ldots, 2t$

```
> calc_gen := proc()
>    local result, Q, i, liste_pol_rest:
>    result := P: # on sait déjà que P est dans le PPCM
>    liste_pol_rest := {liste_div} minus {P}:
>    # alpha^2 est racine de P, donc on peut le sauter
>    for i from 3 to 2*t do
>    for Q in liste_pol_rest do
>       if Eval(Q, X=alpha^i) mod 2 = 0 then
>       result := result*Q:
>       liste_pol_rest:=liste_pol_rest minus {Q}: break:
>    end if: end do: end do:
>    result := Expand(result) mod 2
> end proc:
```

Polynôme générateur et dimension du code :

```
> G := sort( calc_gen() ); d := n - degree(G);
```

$$G := X^{10} + X^9 + X^8 + X^6 + X^5 + X^2 + 1$$

$$d := 5$$

Programme 4.2 Fichier `decodage-bch.msw` partie 2

[...] Suite du script précédent

Calcule le mot de taille n (liste de 0/1) correspondant à un polynôme de degré $n-1$

```
> Mot := proc(Q)
>    [seq( coeff(Q, X,it), it=0..n-1 )]
> end proc:
```

Calcule le polynôme de degré $n-1$ correspondant à un mot de taille n

```
> Pol := proc(mot)
>    sum(mot[it]*X^(it-1), it=1..n);
> end proc:
```

Calcule le syndrôme d'indice i, i.e. $P(\alpha^i)$:

```
> Syndi := proc(pol, i)
>    Eval(pol, X = alpha^i) mod 2;
> end proc:
```

Calcule un vecteur aléatoire avec `nb_erreurs` erreurs

```
> Aleat := proc(nb_erreurs)
>    local hasard:
>    hasard := rand(1..(n-1)):
>    Mot( add(X^hasard(), i=1..nb_erreurs) mod 2 );
> end proc:
```

Calcule un mot du code au hasard

```
> MotCode := proc()
>    local Q;
>    Q := Randpoly(d-1, X) mod 2;
>    Q := Expand( Q*G ) mod 2;
>    Mot(Q);
> end proc:
```

On simule une transmission avec erreur :

```
> mot_code := MotCode();
> mot_transmis := mot_code + Aleat(3) mod 2;
> p_recu := Pol(mot_transmis);
```

$$mot\_code := [0,1,1,1,1,0,0,0,1,0,0,1,1,0,1]$$

$$mot\_transmis := [0,1,1,1,1,0,0,0,1,0,0,0,0,0,0]$$

$$p\_recu := X + X^2 + X^3 + X^4 + X^8$$

Programme 4.3 Fichier `decodage-bch.msw` partie 3

[...] Suite du script précédent

1ère partie : Résolution des équations pour $i = n-t \ldots n-1$ pour trouver $\sigma[1] \ldots \sigma[t]$

Calcule de l'équation polynomiale à résoudre (attention, on note la ε *transformée de Fourier* de l'erreur) :

```
> eqn := (1+add(sigma[i]*Z^i,i=1..t))*
>          (add(epsilon[n-i]*Z^i,i=1..n)):
> eqn := rem(eqn,Z^n-1,Z,'q');  # l'équation est modulo Z^n-1
```

$$\begin{aligned} eqn := {} & (\sigma_1\varepsilon_2+\sigma_3\varepsilon_4+\varepsilon_1+\sigma_2\varepsilon_3)Z^{14}+(\sigma_3\varepsilon_5+\varepsilon_2+\sigma_1\varepsilon_3+\sigma_2\varepsilon_4)Z^{13} \\ & +(\varepsilon_3+\sigma_2\varepsilon_5+\sigma_1\varepsilon_4+\sigma_3\varepsilon_6)Z^{12}+(\sigma_3\varepsilon_7+\sigma_2\varepsilon_6+\sigma_1\varepsilon_5+\varepsilon_4)Z^{11} \\ & +(\sigma_1\varepsilon_6+\varepsilon_5+\sigma_3\varepsilon_8+\sigma_2\varepsilon_7)Z^{10}+(\sigma_1\varepsilon_7+\sigma_3\varepsilon_9+\varepsilon_6+\sigma_2\varepsilon_8)Z^9 \\ & +(\sigma_3\varepsilon_{10}+\sigma_2\varepsilon_9+\sigma_1\varepsilon_8+\varepsilon_7)Z^8+(\sigma_1\varepsilon_9+\varepsilon_8+\sigma_2\varepsilon_{10}+\sigma_3\varepsilon_{11})Z^7 \\ & +(\varepsilon_9+\sigma_1\varepsilon_{10}+\sigma_2\varepsilon_{11}+\sigma_3\varepsilon_{12})Z^6+(\sigma_2\varepsilon_{12}+\varepsilon_{10}+\sigma_1\varepsilon_{11}+\sigma_3\varepsilon_{13})Z^5 \\ & +(\sigma_1\varepsilon_{12}+\varepsilon_{11}+\sigma_3\varepsilon_{14}+\sigma_2\varepsilon_{13})Z^4+(\sigma_1\varepsilon_{13}+\sigma_2\varepsilon_{14}+\varepsilon_{12}+\sigma_3\varepsilon_0)Z^3 \\ & +(\sigma_1\varepsilon_{14}+\varepsilon_{13}+\sigma_3\varepsilon_1+\sigma_2\varepsilon_0)Z^2+(\varepsilon_{14}+\sigma_1\varepsilon_0+\sigma_3\varepsilon_2+\sigma_2\varepsilon_1)Z+\sigma_3\varepsilon_3+\varepsilon_0 \\ & +\sigma_2\varepsilon_2+\sigma_1\varepsilon_1 \end{aligned}$$

Calcule les équations à résoudre, liste les valeurs de ε connues, pour $i = 1 \ldots 2t$, puis évalue les équations :

```
> list_eqn1 := {seq( coeff(eqn,Z,i), i=n-t..n-1 )}:
> epsilon_connu := {seq( epsilon[i] = Syndi(p_recu,i), i=1..2*t )};
> eqn_eval1 := eval(list_eqn1, epsilon_connu);
```

$$\begin{aligned} & epsilon\_connu := \\ & \{\varepsilon_2=\alpha^3+\alpha^2, \varepsilon_1=\alpha^3, \varepsilon_6=\alpha^3+1, \varepsilon_5=1, \varepsilon_4=\alpha^3+\alpha^2+\alpha+1, \varepsilon_3=\alpha^3+\alpha+1\} \end{aligned}$$

$$\begin{aligned} eqn\_eval1 := {} & \{\sigma_1(\alpha^3+\alpha^2+\alpha+1)+\alpha^3+\alpha+1+\sigma_3(\alpha^3+1)+\sigma_2, \\ & \sigma_2(\alpha^3+\alpha^2+\alpha+1)+\alpha^3+\alpha^2+\sigma_3+\sigma_1(\alpha^3+\alpha+1), \\ & \alpha^3+\sigma_3(\alpha^3+\alpha^2+\alpha+1)+\sigma_2(\alpha^3+\alpha+1)+\sigma_1(\alpha^3+\alpha^2)\} \end{aligned}$$

Met sous forme matricielle les équations :

```
> m1 := matrix(t,t):
> b1 := vector(t):
> i := 1:
> for eq in eqn_eval1 do
>    for j from 1 to t do
>       m1[i,j] := coeff(eq,sigma[j],1);
>    end do:
>    b1[i] := eval( eq, [seq(sigma[k]=0,k=1..t)] );
>    i := i+1:
> end do:
```

Calcule les valeurs de σ en résolvant le système :

```
> sigma_val := Linsolve(m1,b1) mod 2:
> sigma_connu := { seq(sigma[i]=sigma_val[i], i = 1..t) };
```

$$sigma\_connu := \{\sigma_1=\alpha^3+1, \sigma_2=\alpha^3+\alpha^2+\alpha, \sigma_3=\alpha^2+1\}$$

Programme 4.4 Fichier `decodage-bch.msw` partie 4

2e partie : Résolution des équations pour $i = 0 \ldots n-2t-1$ pour trouver $\varepsilon[0]$, $\varepsilon[2t+1] \ldots \varepsilon[n-1]$

Calcule les équations pour $i = 0 \ldots n-2t-1$, puis les évalue :

```
> list_eqn2 := {seq( coeff(eqn,Z,i), i=0..n-2*t-1 )}:
> eqn_eval2 := eval(list_eqn2, epsilon_connu):
> eqn_eval2 := eval(eqn_eval2, sigma_connu);
```

$$
\begin{aligned}
&\mathit{eqn\_eval2} := \{\%1\,\varepsilon_{14} + \varepsilon_{12} + (\alpha^3+1)\,\varepsilon_{13} + (\alpha^2+1)\,\varepsilon_0,\\
&\varepsilon_{14} + (\alpha^3+1)\,\varepsilon_0 + (\alpha^2+1)(\alpha^3+\alpha^2) + \%1\,\alpha^3,\\
&\varepsilon_{13} + (\alpha^3+1)\,\varepsilon_{14} + \%1\,\varepsilon_0 + (\alpha^2+1)\,\alpha^3,\\
&\varepsilon_0 + \%1\,(\alpha^3+\alpha^2) + (\alpha^3+\alpha+1)(\alpha^2+1) + (\alpha^3+1)\,\alpha^3,\\
&\varepsilon_{11} + (\alpha^2+1)\,\varepsilon_{14} + \%1\,\varepsilon_{13} + (\alpha^3+1)\,\varepsilon_{12}, \varepsilon_{10} + (\alpha^3+1)\,\varepsilon_{11} + (\alpha^2+1)\,\varepsilon_{13} + \%1\,\varepsilon_{12},\\
&\varepsilon_8 + (\alpha^3+1)\,\varepsilon_9 + \%1\,\varepsilon_{10} + (\alpha^2+1)\,\varepsilon_{11}, (\alpha^2+1)\,\varepsilon_{12} + \%1\,\varepsilon_{11} + (\alpha^3+1)\,\varepsilon_{10} + \varepsilon_9,\\
&(\alpha^2+1)\,\varepsilon_{10} + \varepsilon_7 + \%1\,\varepsilon_9 + (\alpha^3+1)\,\varepsilon_8\}\\
&\%1 := \alpha^3 + \alpha^2 + \alpha
\end{aligned}
$$

Met sous forme matricielle les équations :

```
> # les indices de epsilon a calculer
> epsilon_indices := [0,seq(i, i=2*t+1..n-1)]:
> m2 := matrix(n-2*t,n-2*t):
> b2 := vector(n-2*t):
> i := 1:
> for eq in eqn_eval2 do
>    j:= 1:
>    for index in epsilon_indices do
>       m2[i,j] := coeff(eq,epsilon[index],1):
>       j := j+1;
>    end do:
>    b2[i]:=eval(eq,[epsilon[0]=0,seq(epsilon[k]=0,k=2*t+1..n-1)]);
>    i := i+1:
> end do:
```

Calcule les valeurs de $\varepsilon[0]$, $\varepsilon[2t+1] \ldots \varepsilon[n-1]$, puis regroupe toutes les valeurs :

```
> epsilon_val := Linsolve(m2,b2) mod 2:
> epsilon_val := [epsilon_val[1], seq(Syndi(p_recu,it),it=1..2*t),
> seq(epsilon_val[it],it=2..n-2*t)];
```

$$
\begin{aligned}
&\mathit{epsilon\_val} := [1, \alpha^3, \alpha^3+\alpha^2, \alpha^3+\alpha+1, \alpha^3+\alpha^2+\alpha+1, 1, \alpha^3+1, \alpha^3+\alpha+1, \alpha^3+\alpha,\\
&\alpha^3+\alpha^2+\alpha, 1, \alpha^3+\alpha^2+\alpha, \alpha^3+\alpha^2+1, \alpha^3+\alpha^2+1, \alpha^3+1]
\end{aligned}
$$

On peut maintenant déterminer l'erreur par transformée de Fourier inverse :

```
> erreurs := Normal( TFD(epsilon_val,-1) ) mod 2;
> mot_corrige := mot_transmis - erreurs mod 2:
> evalb( mot_corrige = mot_code );
```

$$\mathit{erreurs} := [0,0,0,0,0,0,0,0,0,0,0,1,1,0,1]$$

$$\mathit{true}$$

Bibliographie

[1] J.L. Alperin et Rowen B. Bell. *Groups and Representations*. Springer Verlag, New York, 1995.

[2] Jörg Arndt. Algorithms for Programmers. 2002.

[3] Michael Artin. *Algebra*. Prentice Hall, New York, 1991.

[4] Laszlo Babai. The Fourier Transform and Equations over Finite Abelian Groups. *Technical Report, TR-2001-01*, 1989.

[5] David H. Bailey. The Computation of π to 29.360.000 Decimal Digits Using Borweins' Quartically Convergent Algorithm. *Mathematics of Computation*, 50(181):283–296, 1987.

[6] David H. Bailey et Paul N. Swarztrauber. The Fractional Fourier Transform and Applications. *RNR Technical Report*, 1995.

[7] Riccardo Bernardini et Jelena Kovacevic. Designing Local Orthogonal Bases on Finite Groups I: Abelian Case. *Journal of Fourier Analysis and Applications*, 6(1):1–23, 2000.

[8] Dan Boney. Learning Using Group Representations. *Proc. COLT 1995*, pages 418–426, 1995.

[9] John P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover Publications, 2000.

[10] Ronald Bracewell. *The Hartley Transform*. Oxford University Press, New York, 1986.

[11] E. Oran Brigham. *Fast Fourier Transform and its Applications*. Prentice Hall, Englewood Cliffs, 1988.

[12] C. Sidney Burrus. Notes on the FFT. 1997.

[13] Cagatay Candan, M. Alper Kutay, et Haldun M. Ozaktas. The Discrete Fractional Fourier Transform. *IEEE Trans. on Signal Processing*, 48:1329–1337, 2000.

[14] Gianfranco Cariolaro, Tomaso Erseghe, Peter Kraniauskas, et Nicola Laurenti. Multiplicity of Fractional Fourier Transforms and Their Relationships. *IEEE Transactions on Signal Processing*, 48(1):227–241, 2000.

[15] Henri Cartan. *Théorie élémentaire des fonctions d'une variable complexe*. Hermann, Paris, 1961.

[16] Philippe G. Ciarlet. *Introduction à l'analyse numérique et à l'optimisation*. Dunod, Paris, 1990.

[17] Jon Claerbout. *Fundamentals of Geophysical Data Processing*. McGraw Hill, 1976.

[18] Henri Cohen. *A Course in Computational Agebraic Number Theory*. Springer Verlag, Berlin, Heidelberg, New York, 1991.

[19] Michael J. Collins. *Representations and Characters of Finite Groups*. Cambridge University Press, Cambridge, 1990.

[20] James W. Cooley et John W. Tukey. An Algorithm for the Machine Calculation of Complex Fourier Series. *Math. Computat.*, 19:297–301, 1965.

[21] Thomas Cormen, Charles Leiserson, et Ronald Rivest. *Introduction à l'algorithmique*. Dunod, Paris, 1992.

[22] David A. Cox, John B. Little, et Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Algebraic Geometry and Commutative Algebra, 2nd ed.* Springer Verlag, New York, Berlin, 1996.

[23] Jean Pierre Demailly. *Analyse numérique et équations différentielles*. EDP, Grenoble, 1996.

[24] Michel Demazure. *Cours d'algèbre. Primalité, divisibilité, codes*. Cassini, Paris, 1997.

[25] Gilbert Demengel. *Transformées de Fourier généralisées*. Ellipses, Paris, 1999.

[26] Persi Diaconis. *Group Representations in Probability and Statistics*. IMS Lecture Series 11, Institute of Mathematical Statistics, Hayward, California, 1988.

[27] Bradley W. Dickinson et Kenneth Steiglitz. Eigenvectors and Functions of the Discrete Fourier Transform. *IEEE Transaction on Acoustics, Speech, and Signal Processing*, 30(1):25–31, 1982.

[28] David L. Donoho et Philip B. Stark. Uncertainty Principles and Signal Recovery. *SIAM Journal of Applied Mathematics*, 49:906–931, 1989.

[29] Harry Dym et H.P. Mc Keam. *Fourier Series and Integrals*. Academic press, New York, San Francisco, London, 1972.

[30] Noam D. Elkies. Lattices, Linear Codes, and Invariants, part I. *Notices of the AMS*, 47(10):1238–1245, 2002.

[31] W.H. Press et Al. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1988.

[32] Kristin Flornes, Axel Grossman, Matthias Holschneider, et Bruno Torrésani. Wavelets on Discrete Fields. *Applied and Computational Harmonic Analysis*, 1:137–147, 1994.

[33] Matteo Frigo et Steven G. Johnson. FFTW: An Adaptive Software Architecture for the FFT. *ICASSP conference proceedings*, 3:1381–1384, 1998.

[34] Christine Froidevaux, Marie-Claude Gaudel, et Michèle Soria. *Types de données et algorithmes*. Ediscience international, 1990.

[35] William Fulton et Joe Harris. *Representation Theory : a First Course*. Springer Verlag, Berlin, Heidelberg, New York, 1991.

[36] Roe Goodman et Nolan R. Wallach. *Representations and Invariants of the Classical Groups*. Cambridge University Press, Cambridge, 1999.

[37] Ronald L. Graham, Donald E. Knuth, et Oren Patashnik. *Concrete Mathematics*. Addison-Wesley, Reading, 1994.

[38] Jean-Pierre Kahane et Pierre Gilles Lemarié-Rieusset. *Séries de Fourier et ondelettes*. Cassini, Paris, 1998.

[39] Donald E. Knuth. *The Art of Computer Programming, Volume 2, Seminumerical Algorithms*. Addison-Wesley, Reading, 1997.

[40] Eyal Kushilevitz et Yishay Mansour. Learning Decision Trees Using the Fourier Spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1991.

[41] T.Y. Lam. Representations of Finite Groups: a Hundred Years, Part I. *Notices of the AMS*, 45(3):368–372, 1998.

[42] T.Y. Lam. Representations of Finite Groups: a Hundred Years, Part II. *Notices of the AMS*, 45(4):465–474, 1998.

[43] Serge Lang. *Algebra*. Addison-Wesley, Reading, 1965.

[44] Philippe Langevin. *Les sommes de caractères et la formule de Poisson dans la théorie des codes, des séquences et des fonctions booléennes*. Université de Toulon, 1999.

[45] Reinhard C. Laubenbacher. Eisenstein Misunderstood Geometric Proof of the Quadratique Reciprocity Theorem. *College Mathematics Journal*, 25:29–34, 1994.

[46] Franz Lemmermeyer. *Reciprocity Laws: From Euler to Eisenstein*. Springer Verlag, Berlin, Heidelberg, New York, 2000.

[47] J.P. Lewis. Fast Normalized Cross-Correlation. *Vison Interface*, pages 120–123, 1995.

[48] Rudolf Lidl et Harald Niederreiter. *Finite Fields*. Cambridge University Press, Cambridge, 1983.

[49] Larry S. Liebovitch, Yi Tao, Angelo T. Todorov, et Leo Levine. Is There an Error Correcting Code in the Base Sequence in DNA ? *Biopphysical Journal*, 71:1539-1544, 1996.

[50] F.J. MacWilliams et N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977.

[51] Stéphane Mallat. *Une exploration des signaux en ondelettes*. Editions de l'école Polytechnique, Palaiseau, 2000.

[52] David K. Malsen et Daniel N. Rockmore. Generalized FFTs – a Survey of Some Recent Results. *Proc. 1995 DIMACS Workshop in Groups and Computation*, 28:183–238, 1995.

[53] Ewa Matusiak, Murad Özaydin, et Tomasz Przebinda. The Donoho-Stark Uncertainty Principle for a Finite Abelian Group. 2000.

[54] Yves Meyer. *Ondelettes et opérateurs 1, Ondelettes*. Hermann, Paris, 1980.

[55] Jean-Yves Ouvrard. *Probabilité 2*. Cassini, Paris, 2000.

[56] Odile Papini et Jacques Wolfman. *Algèbre discrète et codes correcteurs*. Springer Verlag, Berlin, 1995.

[57] Enes Passalic et Thomas Johansson. Further Results on the Relation Between Nonlinearity and Resiliency for Boolean Functions. *Conference on Cryptography and Coding*, 1994.

[58] Daniel Perrin. *Cours d'algèbre*. Ellipses, Paris, 1996.

[59] Edmon Ramis, Claude Deschamps, et Jacques Odoux. *Tome 1 : algèbre*. Masson, Paris, 1979.

[60] Daniel N. Rockemore. The FFT – An Algorithm the Whole Family Can Use. *Computing in Science and Engineering*, 2(1):60–64, 2000.

[61] O.S. Rothaus. On Bent Functions. *Journal of Combinatorial Theory*, 20A:300–305, 1976.

[62] Walter Rudin. *Analyse réelle et complexe*. Dunod, Paris, 1987.

[63] Pierre Samuel. *Théorie des nombres*. Hermann, Paris, 1967.

[64] Jean-Pierre Serre. *Représentations linéaires des groupes finis*. Hermann, Paris, 1966.

[65] Jean-Pierre Serre. *Cours d'arithmétique*. PUF, Paris, 1970.

[66] Claude E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27:379–423 and 623–656, 1948.

[67] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997.

[68] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, 1986.

[69] Gilbert Strang. The Discrete Cosine Transform. *SIAM Review*, 41(1):135–147, 1999.

[70] Paul N. Swarztrauber et Roland A. Sweet. The Fourier and Cyclic Reduction for Solving Poisson's Equation. *Handbook of Fluid Fynamics and Fluid Machinery*, 1996.

[71] P.N. Swarztrauber, R.A. Sweet, W.L. Briggs, V.E. Henson, et J. Otto. Bluestein's FFT for Arbitrary N on the Hypercube. *Parallel Computing*, Vol.17:607–617, 1991.

[72] Audrey Terras. *Fourier Analysis on Finite Groups and Applications*. Cambridge University Press, Cambridge, 1999.

[73] Ronald F. Ullmann. An Algorithm for the Fast Hartley Transform. *Stanford Exploration Project*, SEP-38, 1984.

[74] M. Unser, A. Aldroubi, et M. Eden. Fast B-spline Transforms for Continous Image Representation and Interpolation. *IEE Transaction on Pattern Analysis and Machine Intelligence*, 13(3):277–285, 1991.

[75] Martin Vetterli et Pierre Duhamel. Split-radix Algorithms for Length-p^m DFT's. *IEEE Transactions on Accoustic, Speech, Signal Processing*, 37(1):57–64, 1989.

[76] André Warusfel. *Structures algébriques finies*. Hachette, Paris, 1971.

[77] Hermann Weyl. *Symétrie et mathématique moderne*. Flammarion, Paris, 1964.

[78] Robert Wich. *Z Transform, Theory and Applications*. D.Reidel Publishing Compaghy, 1987.

[79] Herbert S. Wilf. *Generatingfunctionology*. Academic Press, Boston, San Diego, New York, 1990.

[80] Claude Zuily et Hervé Queffelec. *Eléments d'analyse pour l'agrégation*. Masson, Paris, 1995.

Index

Achevé d'imprimer en janvier 2004 par Normandie Roto Impression s.a.s. (France) - N° d'impression : 033242 - Dépôt légal : janvier 2004

La collection *Mathématiques à l'Université* se propose de mettre à la disposition des étudiants de troisième, quatrième et cinquième années d'études supérieures en mathématiques des ouvrages couvrant l'essentiel des programmes actuels des universités françaises. Certains de ces ouvrages pourront être utiles aussi aux étudiants qui préparent le CAPES ou l'agrégation, ainsi qu'aux élèves des grandes écoles.
Nous avons voulu rendre ces livres accessibles à tous : les sujets traités sont présentés de manière simple et progressive, tout en respectant scrupuleusement la rigueur mathématique. Chaque volume comporte un exposé du cours avec des démonstrations détaillées de tous les résultats essentiels et de nombreux exercices. Les auteurs de ces ouvrages ont tous une grande expérience de l'enseignement des mathématiques au niveau supérieur.

Ce livre rassemble tout ce qu'il faut savoir sur la transformée de Fourier discrète. Il s'adresse à un public d'algébristes qui désirent étendre leurs connaissances vers diverses applications (maîtrise, master, DEA, DEES, ...). Les agrégatifs pourront trouver une grande quantité de développements autour du programme officiel. Il sera aussi très utile aux élèves d'écoles d'ingénieurs qui découvriront des sujets classiques sous un jour nouveau.

L'auteur fait alterner la présentation des fondements algébriques de la théorie de Fourier avec l'exposé des applications auxquelles celle-ci donne lieu. De nombreuses extensions de la théorie conduisent en outre à aborder des domaines d'études connexes tels que le traitement du signal, les codes correcteurs ou les représentations linéaires.

Enfin, ce livre contient de nombreux outils. Plus de quatre-vingts programmes MATLAB et MAPLE permettent au lecteur de mettre en œuvre ce qu'il vient d'apprendre. Une grande quantité d'exercices corrigés fournit autant d'occasions d'asseoir ses connaissances en travaillant sur des sujets qui sortent de l'ordinaire.

9 782729 818678

ISBN 2-7298-1867-7