

Solveurs polynomiaux à haute intensité arithmétique pour l'équation de la chaleur et l'équation de Poisson

Thierry Dumont
en collaboration avec A. Darte (LIP/ENSL)

2 Décembre 2017

Idée générale : Pour obtenir des méthodes numériques efficaces, il ne suffit plus de considérer les critères habituels (ordre, stabilité, coût algorithmique), mais **il faut réviser l'usage des méthodes en tenant compte de l'architecture des machines actuelles.**

Idée générale : Pour obtenir des méthodes numériques efficaces, il ne suffit plus de considérer les critères habituels (ordre, stabilité, coût algorithmique), mais **il faut réviser l'usage des méthodes en tenant compte de l'architecture des machines actuelles.**

Plan :

- ▶ Exemple introductif.
- ▶ Le modèle du *roofline* et ses conséquences.
- ▶ Algorithmes, discrétisations et implantation efficaces.

Un exemple introductif. Modèle fluide en physique des plasmas (simulation d'un *streamer*).

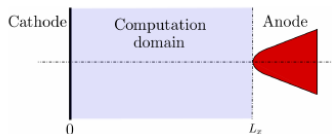


Figure 1: Computational domain for the studied point-to-plane geometry.

Décharges périodiques :

Un fort voltage est appliqué pendant $\simeq 10^{-8}$ seconde, suivi par une période relaxation de $\simeq 10^{-4}$ seconde.

Distance entre la cathode et l'anode : 1 cm.

Simulation d'un streamer

Modèle de Drift-Diffusion :

- ▶ Système de Réaction–Diffusion–Convection :

$$\begin{aligned}\partial_t n_e - \partial_x \cdot n_e \vec{v}_e - \partial_x \cdot (D_e \partial_x n_e) &= n_e \alpha |\vec{v}_e| - n_e \eta |\vec{v}_e| + n_e n_p \beta_{ep} + n_n \gamma \\ \partial_t n_p + \partial_x \cdot n_p \vec{v}_p - \partial_x \cdot (D_p \partial_x n_p) &= n_e \alpha |\vec{v}_e| - n_e n_p \beta_{ep} + n_n n_p \beta_{np} \\ \partial_t n_n - \partial_x \cdot n_n \vec{v}_n - \partial_x \cdot (D_n \partial_x n_n) &= n_e \eta |\vec{v}_e| - n_n n_p \beta_{np} - n_n \gamma\end{aligned}$$

- ▶ Couplé à une équation de Poisson :

$$\varepsilon_0 \partial_x^2 V = -q_e (n_p - n_n - n_e), \quad \vec{E} = -\partial_x V, \quad \vec{v}_i = \mu_i \vec{E}.$$

Simulation d'un streamer

Modèle de Drift-Diffusion :

- ▶ Système de Réaction–Diffusion–Convection :

$$\begin{aligned}\partial_t n_e - \partial_x \cdot n_e \vec{v}_e - \partial_x \cdot (D_e \partial_x n_e) &= n_e \alpha |\vec{v}_e| - n_e \eta |\vec{v}_e| + n_e n_p \beta_{ep} + n_n \gamma \\ \partial_t n_p + \partial_x \cdot n_p \vec{v}_p - \partial_x \cdot (D_p \partial_x n_p) &= n_e \alpha |\vec{v}_e| - n_e n_p \beta_{ep} + n_n n_p \beta_{np} \\ \partial_t n_n - \partial_x \cdot n_n \vec{v}_n - \partial_x \cdot (D_n \partial_x n_n) &= n_e \eta |\vec{v}_e| - n_n n_p \beta_{np} - n_n \gamma\end{aligned}$$

- ▶ Couplé à une équation de Poisson :

$$\varepsilon_0 \partial_x^2 V = -q_e (n_p - n_n - n_e), \quad \vec{E} = -\partial_x V, \quad \vec{v}_i = \mu_i \vec{E}.$$

Échelles de temps les plus rapides de la réaction $\simeq 10^{-14}$ seconde :

Ce problème est vraiment multi-échelles !

Plasma, méthode numérique

Splitting de Strang :

$$\mathcal{S}_{\Delta t}(u_0) = \mathcal{R}_{\Delta t/2} \circ \mathcal{D}_{\Delta t/2} \circ \mathcal{C}_{\Delta t} \circ \mathcal{D}_{\Delta t/2} \circ \mathcal{R}_{\Delta t/2}(u_0).$$

Comment choisir \vec{v}_i in $\mathcal{C}_{\Delta t}$?

1. Calculer \vec{v}_i à partir de u_0 (résoudre une équation de Poisson)
=> méthode d'ordre 1.

Plasma, méthode numérique

Splitting de Strang :

$$\mathcal{S}_{\Delta t}(u_0) = \mathcal{R}_{\Delta t/2} \circ \mathcal{D}_{\Delta t/2} \circ \mathcal{C}_{\Delta t} \circ \mathcal{D}_{\Delta t/2} \circ \mathcal{R}_{\Delta t/2}(u_0).$$


Comment choisir \vec{v}_i in $\mathcal{C}_{\Delta t}$?

1. Calculer \vec{v}_i à partir de u_0 (résoudre une équation de Poisson) \Rightarrow méthode d'ordre 1.
2. Calculer $u_{1/2} = \mathcal{C}_{\Delta t/2} \circ \mathcal{D}_{\Delta t/2} \circ \mathcal{R}_{\Delta t/2}(u_0)$, puis calculer \vec{v}_i à partir de $u_{1/2}$ (une équation de Poisson) \Rightarrow méthode d'ordre 2.

Plasma, méthode numérique

- ▶ On a des approximations d'ordre 1 et 2 => pas de temps adaptatif.


1. Max DUARTE et al. « A new numerical strategy with space-time adaptivity and error control for multi-scale streamer discharge simulations ». In : *J. Comput. Phys.* 231.3 (2012), p. 1002–1019.

2. Stéphane DESCOMBES et al. « Task-based adaptive multiresolution for time-space multi-scale reaction-diffusion systems on multi-core architectures ». In : *SMAI Journal of Computational Mathematics* 3 (avr. 2017), p. 29–51. 

Plasma, méthode numérique

- ▶ On a des approximations d'ordre 1 et 2 => pas de temps adaptatif.
- ▶ La solution développe de forts gradients qui se propagent : il faut adapter le maillage. Nous utilisons une méthode multiresolution / volumes finis.^{1 2}

1. Max DUARTE et al. « A new numerical strategy with space-time adaptivity and error control for multi-scale streamer discharge simulations ». In : *J. Comput. Phys.* 231.3 (2012), p. 1002–1019.

2. Stéphane DESCOMBES et al. « Task-based adaptive multiresolution for time-space multi-scale reaction-diffusion systems on multi-core architectures ». In : *SMAI Journal of Computational Mathematics* 3 (avr. 2017), p. 29–51. 

Plasma, méthode numérique

- ▶ Réaction : Radau5.
- ▶ Équation de la chaleur et équation de Poisson :
 - ▶ Factorisation incomplète : coûteuse.
 - ▶ Solution : coûteuse.
 - ▶ **La plus grande partie de temps calcul est employée à les résoudre.**

Plasma, méthode numérique

- ▶ Réaction : Radau5.
- ▶ Équation de la chaleur et équation de Poisson :
 - ▶ Factorisation incomplète : coûteuse.
 - ▶ Solution : coûteuse.
 - ▶ **La plus grande partie de temps calcul est employée à les résoudre.**

Pour comprendre pourquoi, il faut comprendre (un peu) le fonctionnement des ordinateurs *contemporains*.

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \text{ (8 flops)}.$$

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \quad (8 \text{ flops}).$$

Supposons que nous ayons une machine qui possède 2×12 cœurs, avec une fréquence d'horloge de $2,2 \cdot 10^9$ cycles par seconde et qui peut faire 2 instructions AVX simultanément (architecture Intel Broadwell).

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \quad (8 \text{ flops}).$$

Supposons que nous ayons une machine qui possède 2×12 cœurs, avec une fréquence d'horloge de $2,2 \cdot 10^9$ cycles par seconde et qui peut faire 2 instructions AVX simultanément (architecture Intel Broadwell).

La performance « pic » est :

AVX	fréq.	cœurs
2×8		

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \quad (8 \text{ flops}).$$

Supposons que nous ayons une machine qui possède 2×12 cœurs, avec une fréquence d'horloge de $2,2 \cdot 10^9$ cycles par seconde et qui peut faire 2 instructions AVX simultanément (architecture Intel Broadwell).

La performance « pic » est :

AVX		fréq.		cœurs
2×8	\times	$2,2 \cdot 10^9$		

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \quad (8 \text{ flops}).$$

Supposons que nous ayons une machine qui possède 2×12 cœurs, avec une fréquence d'horloge de $2,2 \cdot 10^9$ cycles par seconde et qui peut faire 2 instructions AVX simultanément (architecture Intel Broadwell).

La performance « pic » est :

$$2 \times 8 \quad \times \quad 2,2 \cdot 10^9 \quad \times \quad 2 \times 12$$

Processeurs contemporains

SIMD (instructions AVX) :

En un tour d'horloge, faire :

$$y_i = a_i x_i + b_i, \quad i = 1, 4 \text{ (8 flops).}$$

Supposons que nous ayons une machine qui possède 2×12 cœurs, avec une fréquence d'horloge de $2,2 \cdot 10^9$ cycles par seconde et qui peut faire 2 instructions AVX simultanément (architecture Intel Broadwell).

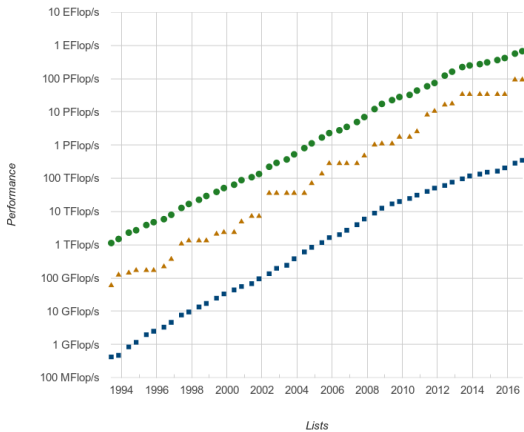
La performance « pic » est :

$$\begin{array}{ccccccc} \text{AVX} & & \text{fréq.} & & \text{cœurs} & & \\ 2 \times 8 & \times & 2,2 \cdot 10^9 & \times & 2 \times 12 & = & 844,8 \text{ Gflops/second.} \end{array}$$

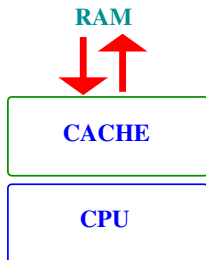
Performances presque impossibles à atteindre !

Pourquoi ?

Performance Development



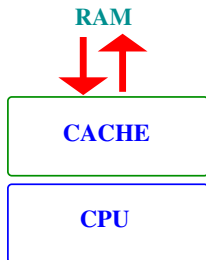
L'intensité arithmétique et le modèle en Toit (Roofline Model)



En pratique, dans la plupart des cas, les performances sont limitées par la bande passante de la machine.

Performances atteignables (en GFlops/sec) ?

L'intensité arithmétique et le modèle en Toit (Roofline Model)



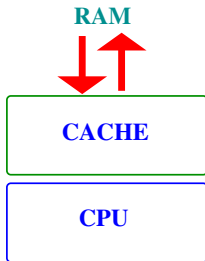
En pratique, dans la plupart des cas, les performances sont limitées par la bande passante de la machine.

Performances atteignables (en GFlops/sec) ?

Intensité arithmétique d'un programme

$I_a = \text{nombre d'opérations} / \text{nombre de flottants doubles échangés.}$

L'intensité arithmétique et le modèle en Toit (Roofline Model)



En pratique, dans la plupart des cas, les performances sont limitées par la bande passante de la machine.

Performances atteignables (en GFlops/sec) ?

Intensité arithmétique d'un programme

$I_a = \text{nombre d'opérations} / \text{nombre de flottants doubles échangés.}$

Performances atteignables (GFlops/sec) =

$\min(\text{Performance « pic », Bande passante de la machine} \times I_a).$

L'intensité arithmétique et le modèle en Toit (Roofline Model)

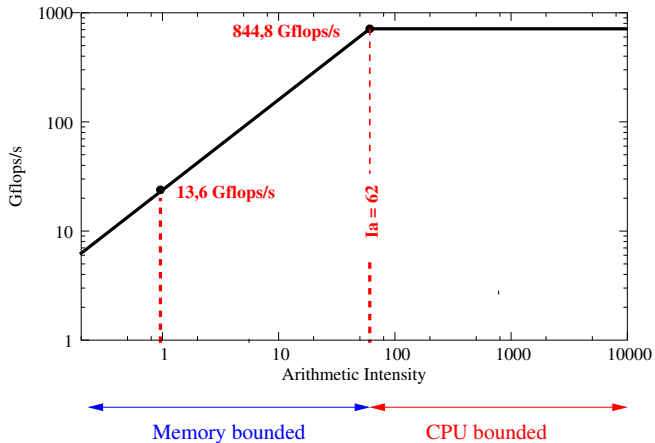
Broadwell, 2×12 cœurs : bande passante $\simeq 13,6$ Giga-doubles/s.

L'intensité arithmétique et le modèle en Toit (Roofline Model)

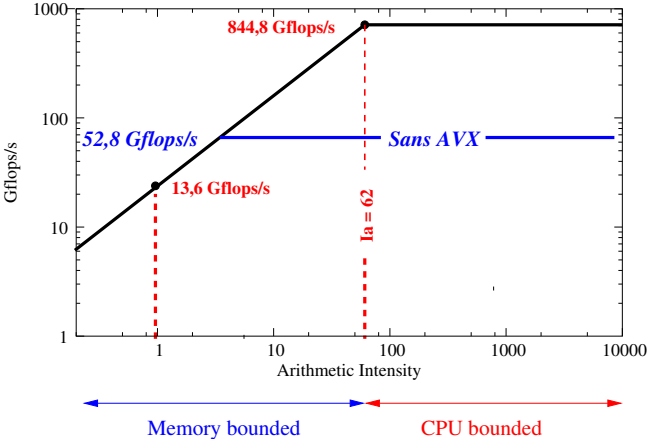
Broadwell, 2×12 cœurs : bande passante $\simeq 13,6$ Giga-doubles/s.

Pour atteindre la performance *pic* vous avez besoin de : $I_a \simeq 62!$

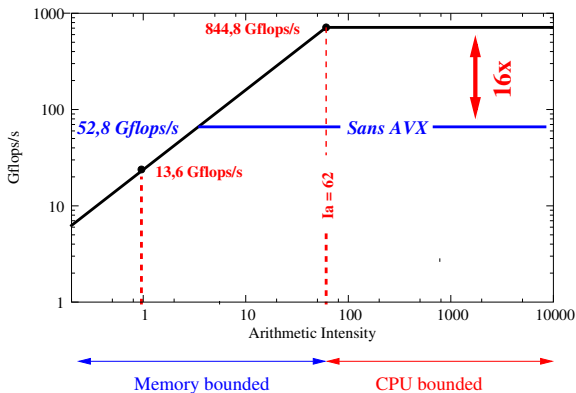
Modèle en Toit



Modèle en Toit



Modèle en Toit³



Si l'intensité arithmétique dépasse 4, il faut vectoriser.

3. Samuel WILLIAMS, Andrew WATERMAN et David PATTERSON. « Roofline: An Insightful Visual Performance Model for Multicore Architectures ». In : *Commun. ACM* 52.4 (avr. 2009), p. 65–76.

L'intensité arithmétique et le modèle en Toit (Roofline Model) : quelques exemples.

Unité utilisée : double.

1. Produit scalaire $s = \sum_{i=1}^n x_i \cdot y_i$: $I_a = 1$.

L'intensité arithmétique et le modèle en Toit (Roofline Model) : quelques exemples.

Unité utilisée : double.

1. Produit scalaire $s = \sum_{i=1}^n x_i \cdot y_i$: $I_a = 1$.
2. Appliquer le stencil à 7 points du Laplacien (3d) :
 $I_a = 8/8 = 1$.

L'intensité arithmétique et le modèle en Toit (Roofline Model) : quelques exemples.

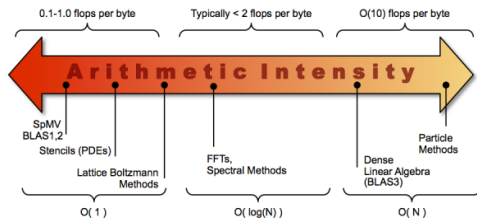
Unité utilisée : double.

1. Produit scalaire $s = \sum_{i=1}^n x_i \cdot y_i$: $I_a = 1$.
2. Appliquer le stencil à 7 points du Laplacien (3d) :
 $I_a = 8/8 = 1$.
3. Produit de 2 Matrices $C = A \cdot B$: $I_a = 2 n^3 / 4 n^2 = \mathcal{O}(n)$.

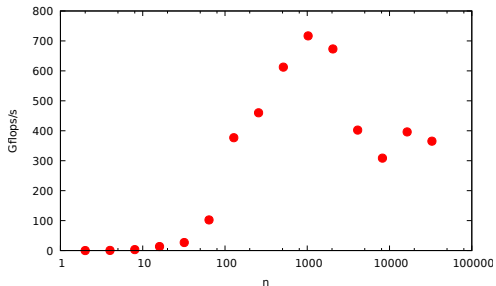
L'intensité arithmétique et le modèle en Toit (Roofline Model) : quelques exemples.

Unité utilisée : double.

1. Produit scalaire $s = \sum_{i=1}^n x_i \cdot y_i$: $I_a = 1$.
2. Appliquer le stencil à 7 points du Laplacien (3d) :
 $I_a = 8/8 = 1$.
3. Produit de 2 Matrices $C = A \cdot B$: $I_a = 2 n^3 / 4 n^2 = \mathcal{O}(n)$.

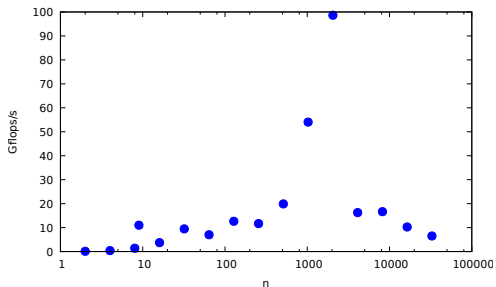


Intensité arithmétique : expériences



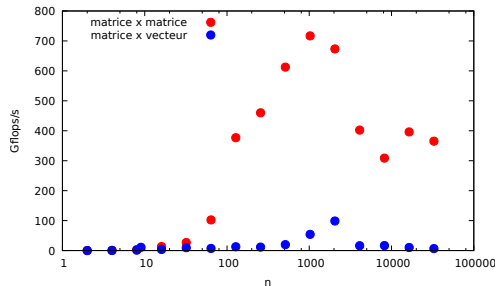
Produit de deux matrices (DGEMM, Intel mkl version parallèle).

Intensité arithmétique : expériences



Produit Matrice \times Vecteur (DGEMV, Intel mkl, version parallèle).

Intensité arithmétique : expériences



Matrice x Matrice et Matrice x Vecteur.

Intensité arithmétique : expériences

Appliquer le stencil à 7 points du Laplacien 3-d, stocké dans une matrice CSR/CSL (cad.. stocker les coefficients et les indices des lignes) :

Intensité arithmétique : expériences

Appliquer le stencil à 7 points du Laplacien 3-d, stocké dans une matrice CSR/CSL (cad.. stocker les coefficients et les indices des lignes) :

	0	1	2	3	4		0	1	2	3	4	5						
0	2.0		3.5		6.7	rowptr	0	3	5	7	10	12						
1		8.2		9.2			0	1	2	3	4	5	6	7	8	9	10	11
2		1.1	2.8			colind	0	2	4	1	3	1	2	0	2	3	1	3
3	3.0		1.5	4.5			0	1	2	3	4	5	6	7	8	9	10	11
4		2.5		8.9		values	2.0	3.5	6.7	8.2	9.2	1.1	2.8	3.0	1.5	4.5	2.5	8.9

Intensité arithmétique : expériences

Appliquer le stencil à 7 points du Laplacien 3-d, stocké dans une matrice CSR/CSL (cad.. stocker les coefficients et les indices des lignes) :

	0	1	2	3	4		0	1	2	3	4	5										
0	2.0		3.5		6.7	rowptr	0	3	5	7	10	12										
1		8.2		9.2			0	1	2	3	4	5	6	7	8	9	10	11				
2		1.1	2.8			colind	0	2	4	1	3	1	2	0	2	3	1	3				
3	3.0		1.5	4.5			0	1	2	3	4	5	6	7	8	9	10	11				
4		2.5		8.9		values	2.0	3.5	6.7	8.2	9.2	1.1	2.8	3.0	1.5	4.5	2.5	8.9				

- ▶ Bande passante : 37/2 doubles ; Flops : 13 $\Rightarrow I_a \simeq 0,7$.

Intensité arithmétique : expériences

Appliquer le stencil à 7 points du Laplacien 3-d, stocké dans une matrice CSR/CSL (cad.. stocker les coefficients et les indices des lignes) :

	0	1	2	3	4		0	1	2	3	4	5										
0	2.0		3.5		6.7	rowptr	0	3	5	7	10	12										
1		8.2		9.2			0	1	2	3	4	5	6	7	8	9	10	11				
2		1.1	2.8			colind	0	2	4	1	3	1	2	0	2	3	1	3				
3	3.0		1.5	4.5			0	1	2	3	4	5	6	7	8	9	10	11				
4		2.5		8.9		values	2.0	3.5	6.7	8.2	9.2	1.1	2.8	3.0	1.5	4.5	2.5	8.9				

- ▶ Bande passante : 37/2 doubles ; Flops : 13 $\Rightarrow I_a \simeq 0,7$.
- ▶ Bdwth machine : 13,6 Giga doubles/s
 \Rightarrow Atteignable = $0,7 \times 13,6 = 9,52$ Gflops.

Intensité arithmétique : expériences

Appliquer le stencil à 7 points du Laplacien 3-d, stocké dans une matrice CSR/CSL (cad.. stocker les coefficients et les indices des lignes) :

	0	1	2	3	4		0	1	2	3	4	5		0	1	2	3	4	5	6	7	8	9	10	11
0	2.0		3.5		6.7	rowptr	0	3	5	7	10	12													
1		8.2		9.2																					
2		1.1	2.8			colind	0	2	4	1	3	1	2	0	2	3	1	3							
3	3.0		1.5	4.5																					
4		2.5		8.9		values	2.0	3.5	6.7	8.2	9.2	1.1	2.8	3.0	1.5	4.5	2.5	8.9							

- ▶ Bande passante : 37/2 doubles ; Flops : 13 $\Rightarrow I_a \simeq 0,7$.
- ▶ Bdwth machine : 13,6 Giga doubles/s
 \Rightarrow Atteignable = $0,7 \times 13,6 = 9,52$ Gflops.

Mesuré : 8,99 Gflops.

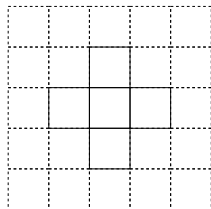
Pas d'espoir ? Le stencil à 7 points ($I_a = 1$) est-il limité à 13,6 Gflop/s ?

Pas d'espoir ? Le stencil à 7 points ($I_a = 1$) est-il limité à 13,6 Gflop/s ?

Non ! On peut réutiliser les données en cache.

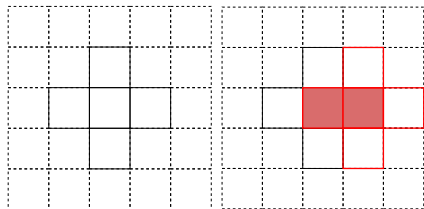
Pas d'espoir ? Le stencil à 7 points ($I_a = 1$) est-il limité à 13,6 Gflop/s ?

Non ! On peut réutiliser les données en cache.



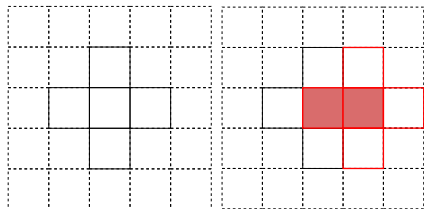
Pas d'espoir ? Le stencil à 7 points ($I_a = 1$) est-il limité à 13,6 Gflop/s ?

Non ! On peut réutiliser les données en cache.



Pas d'espoir ? Le stencil à 7 points ($I_a = 1$) est-il limité à 13,6 Gflop/s ?

Non ! On peut réutiliser les données en cache.



k	$Max.I_a$	Gflops/s
0	4,0	54,4
1	3,3	44,8
2	3,0	40,8
3	2,8	38,1

$$Y = S_7(U) + \sum_{i=0}^k \alpha_i V_i.$$

Éviter :

- ▶ Les produits scalaires.
- ▶ Les matrices CSR/CSL (LU incomplet).
- ▶ Les combinaisons linéaires de grands vecteurs..
- ▶ Les méthodes peu parallélisables.
- ▶ Les méthodes à faible intensité arithmétique.

Éviter :

- ▶ Les produits scalaires.
- ▶ Les matrices CSR/CSL (LU incomplet).
- ▶ Les combinaisons linéaires de grands vecteurs..
- ▶ Les méthodes peu parallélisables.
- ▶ Les méthodes à faible intensité arithmétique.

Préférer :

- ▶ ... les méthodes à grande intensité arithmétique.
- ▶ Les méthodes « embarrassingly parallel ».

Équation de la chaleur

Comment éviter :

- ▶ Les matrices CSR/CSL (LU incomplet).
- ▶ Les combinaisons linéaires de grands vecteurs..
- ▶ Les méthodes peu parallélisables.
- ▶ Les produits scalaires.

Méthodes de Runge–Kutta et problèmes linéaires

$$\frac{du}{dt} = F(t, u), \quad u(t_0) = u_0,$$

$$F : [t_0, +\infty[\times \mathbb{R}^n \rightarrow \mathbb{R}^n.$$

Méthodes de Runge–Kutta et problèmes linéaires

$$\frac{du}{dt} = F(t, u), \quad u(t_0) = u_0,$$

$$F : [t_0, +\infty[\times \mathbb{R}^n \rightarrow \mathbb{R}^n.$$

Méthode de Runge–Kutta, avec s étages :

$$k_j = F\left(x_0 + c_j \delta t, y_0 + \delta t \left(\sum_{i=1}^s a_{ij} k_i\right)\right) \quad j = 1, \dots, s,$$

$$u_1 = u_0 + \delta t \sum_{j=1}^s b_j k_j.$$

c_1	a_{11}	a_{12}	\cdots	a_{1s-1}	a_{1s}
c_2	a_{21}	a_{22}	\cdots	a_{2s-1}	a_{2s}
\vdots	\vdots		\ddots		\vdots
c_s	a_{s1}	a_{s2}	\cdots	a_{ss-1}	a_{ss}
	b_1	b_2	\cdots	b_{s-1}	b_s

Tableau de Butcher

c_1	a_{11}	a_{12}	\cdots	a_{1s-1}	a_{1s}
c_2	a_{21}	a_{22}	\cdots	a_{2s-1}	a_{2s}
\vdots	\vdots		\ddots		\vdots
c_s	a_{s1}	a_{s2}	\cdots	a_{ss-1}	a_{ss}
	b_1	b_2	\cdots	b_{s-1}	b_s

Tableau de Butcher

Classification :

- ▶ **Méthode implicite** : résoudre un système linéaire de taille $s \times n$. Exemple : Radau5.

c_1	a_{11}	a_{12}	\cdots	a_{1s-1}	a_{1s}
c_2	a_{21}	a_{22}	\cdots	a_{2s-1}	a_{2s}
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
c_s	a_{s1}	a_{s2}	\cdots	a_{ss-1}	a_{ss}
	b_1	b_2	\cdots	b_{s-1}	b_s

Tableau de Butcher

Classification :

- ▶ **Méthode implicite** : résoudre un système linéaire de taille $s \times n$. Exemple : Radau5.
- ▶ **Diagonalement implicite** : $\forall i, \forall j > i, a_{ij} = 0$. Résoudre s systèmes de taille n ; (en général : $\forall i a_{ii} = \gamma$).

c_1	a_{11}	a_{12}	\cdots	a_{1s-1}	a_{1s}
c_2	a_{21}	a_{22}	\cdots	a_{2s-1}	a_{2s}
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
c_s	a_{s1}	a_{s2}	\cdots	a_{ss-1}	a_{ss}
	b_1	b_2	\cdots	b_{s-1}	b_s

Tableau de Butcher

Classification :

- ▶ **Méthode implicite** : résoudre un système linéaire de taille $s \times n$. Exemple : Radau5.
- ▶ **Diagonalement implicite** : $\forall i, \forall j > i, a_{ij} = 0$. Résoudre s systèmes de taille n ; (en général : $\forall i a_{ii} = \gamma$).
- ▶ **R-K. Explicite** : $\forall i, \forall j \geq i, a_{ij} = 0$. pas de système à résoudre !

Méthodes de Runge–Kutta et problèmes linéaires

On regarde :

$$\frac{du}{dt} = \lambda u, \quad \lambda \in \mathbb{C}$$

Méthodes de Runge–Kutta et problèmes linéaires

On regarde :

$$\frac{du}{dt} = \lambda u, \quad \lambda \in \mathbb{C}$$

Posons : $z = \lambda \delta t$. Alors :

- ▶ si la méthode est **(diagonalement) implicite** : $u_1 = Q(z)u_0$ où Q est une **fraction rationnelle**.
- ▶ si la méthode est **explicite** : $u_1 = Q(z)u_0$, mais Q est un **polynôme**.

Runge–Kutta méthode : stabilité

$$\mathcal{D} = \{z \in \mathbb{C}, |Q(z)| < 1\}.$$

Runge–Kutta méthode : stabilité

$$\mathcal{D} = \{z \in \mathbb{C}, |Q(z)| < 1\}.$$

- ▶ **Méthode explicite** : \mathcal{D} est **borné** dans toutes les directions. (la taille de \mathcal{D} est de l'ordre de 1).
- ▶ **Méthode implicite** :
 - ▶ A-stabilité : $\mathbb{C}^- \subset \mathcal{D}$.
 - ▶ L-stabilité : A-stabilité et $Q(z) \rightarrow 0$ quand $\operatorname{Re}(z) \rightarrow -\infty$.

Runge–Kutta méthodes : stabilité

Pour :

$$\frac{du}{dt} = \varepsilon \Delta u,$$

on a $z \simeq \varepsilon \delta t / h^2$. Alors :

- ▶ méthode explicite *Classique* : $dt < Ch^2/\varepsilon$,

Runge–Kutta méthodes : stabilité

Pour :

$$\frac{du}{dt} = \varepsilon \Delta u,$$

on a $z \simeq \varepsilon \delta t / h^2$. Alors :

- ▶ méthode explicite *Classique* : $dt < Ch^2/\varepsilon$,
- ▶ Méthode (diagonalement) implicite et A-stable : pas de borne pour dt (excepté la précision !)

Runge–Kutta méthodes : stabilité

Pour :

$$\frac{du}{dt} = \varepsilon \Delta u,$$

on a $z \simeq \varepsilon \delta t / h^2$. Alors :

- ▶ méthode explicite *Classique* : $dt < Ch^2/\varepsilon$,
- ▶ Méthode (diagonalement) implicite et A-stable : pas de borne pour dt (excepté la précision !)

Y a-t-il *de la place* entre ces méthodes ?

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Une méthode RK. explicite d'ordre p a au moins p étages. Par exemple la méthode RK4 classique :

$$Q(z) = 1 + z + z^2/2 + z^3/6 + z^4/24.$$

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Une méthode RK. explicite d'ordre p a au moins p étages. Par exemple la méthode RK4 classique :

$$Q(z) = 1 + z + z^2/2 + z^3/6 + z^4/24.$$

Idée pour l'ordre p , utiliser $s > p$ étages :

$$Q(z) = \sum_{i=0}^p \frac{z^i}{i!} + \sum_{i=p+1}^s a_i z^i,$$

et choisir $a_i, i = p + 1, s$ de sorte que \mathcal{D} ait la plus grande extension possible le long de l'axe réel négatif.

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

- ▶ 1^{re} génération. Dumka (Lebedev). Polynôme optimaux de la forme

$$\prod_{i=0}^s (1 - \alpha_i z).$$

Pour $du/dt = Au$:

$$u_1 = \prod_{i=0}^s (I - \alpha_i \delta t A) u_0.$$

Produit de méthodes d'Euler explicites \Rightarrow problème de stabilité, résolu par un ordonnancement astucieux des coefficients a_j .

Un peu trop délicat.

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

- ▶ 2^e génération. Méthodes Rock (A. Abdulle). Polynôme quasi optimal.
 - ▶ Pour un s donné on a un ensemble de $s - p$ polynômes orthogonaux par un poids w .
 - ▶ le poids w est un polynôme.

4. A. ABDULLE et A. MEDOVIKOV. « Second order Chebyshev methods based on orthogonal polynomials ». In : *Numer. Math.* 90.1 (2001), p. 1–18.

5. A. ABDULLE. « Fourth order Chebyshev methods with recurrence relation ». In : *SIAM J. Sci. Comput.* 23.6 (2002), 2041–2054 (électronique).

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

- ▶ 2^e génération. Méthodes Rock (A. Abdulle). Polynôme quasi optimal.
 - ▶ Pour un s donné on a un ensemble de $s - p$ polynômes orthogonaux par un poids w .
 - ▶ le poids w est un polynôme.

Difficultés :

- ▶ reconstruire une méthode de Runge–Kutta.
(apparaît comme la composition de deux méthodes RK).
- ▶ trouver son ordre.

(B-series, groupe de Butcher).

4. A. ABDULLE et A. MEDOVIKOV. « Second order Chebyshev methods based on orthogonal polynomials ». In : *Numer. Math.* 90.1 (2001), p. 1–18.

5. A. ABDULLE. « Fourth order Chebyshev methods with recurrence relation ». In : *SIAM J. Sci. Comput.* 23.6 (2002), 2041–2054 (électronique).

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

- ▶ 2^e génération. Méthodes Rock (A. Abdulle). Polynôme **quasi optimal**.
 - ▶ Pour un s donné on a un ensemble de $s - p$ polynômes orthogonaux par un poids w .
 - ▶ le poids w est un polynôme.

Difficultés :

- ▶ reconstruire une méthode de Runge–Kutta.
(apparaît comme la composition de deux méthodes RK).
- ▶ trouver son ordre.

(B-series, groupe de Butcher).

Existe pour l'ordre 2 et l'ordre 4 (Rock2, Rock4)^{4 5}.

La taille du domaine de stabilité le long de \mathbb{R}^- croît comme p^2 .

4. A. ABDULLE et A. MEDOVIKOV. « Second order Chebyshev methods based on orthogonal polynomials ». In : *Numer. Math.* 90.1 (2001), p. 1–18.

5. A. ABDULLE. « Fourth order Chebyshev methods with recurrence relation ». In : *SIAM J. Sci. Comput.* 23.6 (2002), 2041–2054 (electronique).

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Cas linéaire :

1. 1^{re} méthode : récurrence à 3 termes.
2. 2^e méthode : RK explicite classique

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Cas linéaire :

1. 1^{re} méthode : récurrence à 3 termes.
2. 2^e méthode : RK explicite classique \rightarrow polynôme

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Cas linéaire :

1. 1^{re} méthode : récurrence à 3 termes.
2. 2^e méthode : RK explicite classique \rightarrow polynôme \rightarrow schéma de Horner pour l'évaluer.

Méthodes de Runge–Kutta explicites à grand domaine de stabilité

Cas linéaire :

1. 1^{re} méthode : récurrence à 3 termes.
2. 2^e méthode : RK explicite classique \rightarrow polynôme \rightarrow schéma de Horner pour l'évaluer.

Pour $du/dt = Au$, il suffit de savoir calculer :

- ▶ $v \leftarrow \alpha Ax + \beta y$,
- ▶ $v \leftarrow \alpha Ax + \beta y + \gamma z$.

Pour le « Laplacien à sept points » on peut atteindre 30 Gflops/s.

Comparaison entre Crank-Nikolson et Rock2.

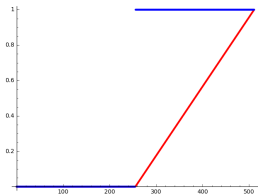
$$R = \frac{1 + z/2}{1 - z/2}$$

(A-stable mais pas L-stable!).

Comparaison entre Crank-Nikolson et Rock2.

$$R = \frac{1 + z/2}{1 - z/2}$$

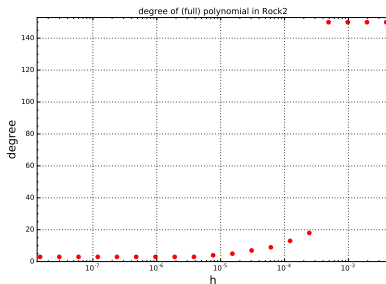
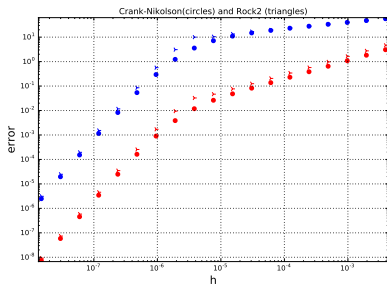
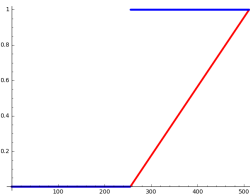
(A-stable mais pas L-stable!).



Comparaison entre Crank-Nikolson et Rock2.

$$R = \frac{1 + z/2}{1 - z/2}$$

(A-stable mais pas L-stable!).



Précision/ h comparables. Pouvez-vous résoudre un grand système linéaire pour un coût inférieur à 5 produits matrice \times vecteur ?

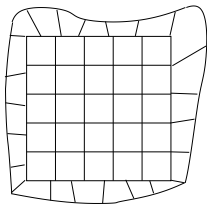
30 Gigaflops/s... comment faire mieux ?

Stencils a haute intensité arithmétique avec les méthodes DG

- ▶ Maillage cartésien 3d : on obtient un *stencil* à 7 matrices.
- ▶ Haute intensité I_a .
- ▶ Libre choix des bases de polynômes.

Stencils a haute intensité arithmétique avec les méthodes DG

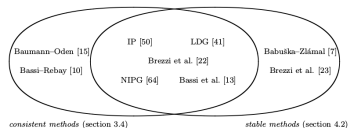
- ▶ Maillage cartésien 3d : on obtient un *stencil* à 7 matrices.
- ▶ Haute intensité I_a .
- ▶ Libre choix des bases de polynômes.



(assez facile DG!)

Stencils a haute intensité arithmétique avec les méthodes DG

Quelle méthode DG ?



Le meilleur choix semble être la méthode de pénalité intérieure
Interior Penalty (IP)⁶.

6. Douglas N. ARNOLD et al. « Unified analysis of discontinuous Galerkin methods for elliptic problems ». In : *SIAM J. Numer. Anal.* 39.5 (2001/02), p. 1749–1779.

Méthode DG / Ip

Les inconnues sont σ_h et u_h .

$$\int_K \sigma_h \cdot \tau dx = - \int_K u_h \nabla \cdot \tau dx + \int_{\partial K} \hat{u}_K \eta_K \cdot \tau ds \quad \forall \tau \in \Sigma(K),$$
$$\int_K \sigma_h \cdot \nabla v dx = \int_K f v dx + \int_{\partial K} \hat{\sigma}_K \cdot \eta_K v ds \quad \forall v \in P(K).$$

Où $\hat{\sigma}_K$ sont \hat{u}_K des flux numériques.

Méthode DG / Ip

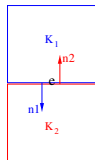
Les inconnues sont σ_h et u_h .

$$\int_K \sigma_h \cdot \tau dx = - \int_K u_h \nabla \cdot \tau dx + \int_{\partial K} \hat{u}_K \eta_K \cdot \tau ds \quad \forall \tau \in \Sigma(K),$$

$$\int_K \sigma_h \cdot \nabla v dx = \int_K f v dx + \int_{\partial K} \hat{\sigma}_K \cdot \eta_K v ds \quad \forall v \in P(K).$$

Où $\hat{\sigma}_K$ sont \hat{u}_K des flux numériques.

Soient K_1 et K_2 deux éléments voisins avec une face commune e .



Sur e :

$$\phi(x) \in \mathbb{R}^d \quad : \quad \phi = \frac{1}{2}(\phi_1 + \phi_2) \quad [\phi] = \phi_1 \cdot \eta_1 + \phi_2 \cdot \eta_2,$$

$$\phi(x) \in \mathbb{R} \quad : \quad \phi = \frac{1}{2}(\phi_1 + \phi_2) \quad [\phi] = \phi_1 \eta_1 + \phi_2 \eta_2.$$

Flux

$$\hat{u} = u_h, \quad \hat{\sigma} = \nabla_h u_h - \eta_e h_e^{-1} [u_h].$$

Mais on peut éliminer σ_h :

Forme primale

$$B_h(u_h, v) = \int_{\Omega} \nabla_h u_h \cdot \nabla_h v dx - \int_{\Gamma} ([u_h] \cdot \nabla_h v + \nabla_h u_h \cdot [v]) ds + \int_{\Gamma} \alpha [u_h] \cdot [v] ds.$$

avec $\alpha = \eta_e h_e^{-1}$ sur chaque $e \in \mathcal{E}$.

On implante la méthode en utilisant :

- ▶ La base de Legendre :

$$Q_{i,j,k} = P_{i,j,k}(x, y, z) = p_i(x) p_j(y) p_k(z),$$

avec :

$$p_l(s) = L_l((2s - h)/h), \quad l = 0, \text{ degré.}$$

(normalisée pour avoir une matrice de masse identité).

- ▶ pour des polynômes de degrés 2 à 5 (thanks to SageMath software).

Meilleures performances pour le degré 3 :

- ▶ I_a croit avec le degré.
- ▶ Les ordinateurs actuels vectorisent les opérations sur les vecteurs de taille multiple de 4.

DG : stencil pour des polynômes de degré 3

- ▶ $A_{i,i}$ est une matrice 64×64 avec 4 termes non nuls par ligne.
- ▶ If $i \neq j$, $A_{i,j}$ a 256 termes non nuls, avec une structure régulière (boucle sur une matrice 4×4 pour effectuer le produit).

DG : stencil pour des polynômes de degré 3

- ▶ $A_{i,i}$ est une matrice 64×64 avec 4 termes non nuls par ligne.
- ▶ If $i \neq j$, $A_{i,j}$ a 256 termes non nuls, avec une structure régulière (boucle sur une matrice 4×4 pour effectuer le produit).

I_a ?

- ▶ Flops :

$$A_{i,j}, i \neq j : 6 \times 512 = 3072$$

$$A_{i,i} : 1 \times 512 = 512$$

$$\text{Total} : 3584 \text{ flops.}$$

DG : stencil pour des polynômes de degré 3

- ▶ $A_{i,i}$ est une matrice 64×64 avec 4 termes non nuls par ligne.
- ▶ If $i \neq j$, $A_{i,j}$ a 256 termes non nuls, avec une structure régulière (boucle sur une matrice 4×4 pour effectuer le produit).

I_a ?

- ▶ Flops :

$$A_{i,j}, i \neq j \quad : \quad 6 \times 512 = 3072$$

$$A_{i,i} \quad : \quad 1 \times 512 = 512$$

$$\text{Total} \quad : \quad 3584 \text{ flops.}$$

- ▶ Bande passante de la méthode : $8 \times 64 = 512$ (doubles).

Donc, $I_a = 7$ sans réutilisation des données.

DG : stencil pour des polynômes de degré 3

$$l_a = 7.$$

- ▶ Performance « pic » **théorique** :
 $7 \times 13,6 = 95,2$ Gigaflops/second.

L'équation de Poisson

Gradient conjugué préconditionne.

Préconditionnement Tchébichef :

Trouver $s \in \mathbb{P}_k$ qui minimise :

$$\max_{\lambda \in [a,b]} |1 - \lambda s(\lambda)|.$$

La solution est un polynôme de Tchébichef « shifté ».

Pour résoudre $Ax = B$, utiliser $M^{-1} = s(A)$ comme préconditionneur :

L'équation de Poisson

Gradient conjugué préconditionne.

Préconditionnement Tchébichef :

Trouver $s \in \mathbb{P}_k$ qui minimise :

$$\max_{\lambda \in [a,b]} |1 - \lambda s(\lambda)|.$$

La solution est un polynôme de Tchébichef « shifté ».

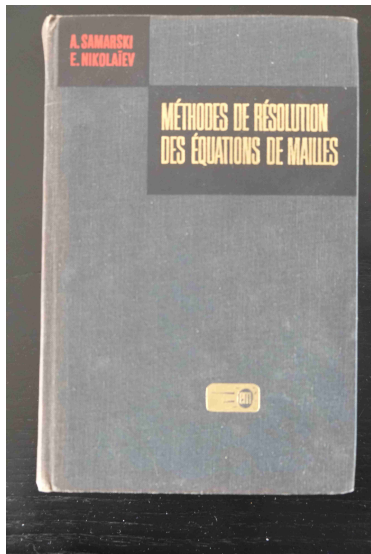
Pour résoudre $Ax = B$, utiliser $M^{-1} = s(A)$ comme préconditionneur :

Évaluation avec la formule de récurrence à 3 termes.

$r_0 := b - Ax_0; u_0 = M^{-1}r_0; p_0 = u_0;$
for $i = 0, \dots$ do :
 $s := Ap_i$
 $\alpha := \langle r_i, u_i \rangle / \langle s, p_i \rangle$
 $x_{i+1} := x_i + \alpha p_i$
 $r_{i+1} := r_i - \alpha s$
 $u_{i+1} := M^{-1}r_{i+1}$
 $\beta := \langle r_{i+1}, u_{i+1} \rangle / \langle r_i, u_i \rangle$
 $p_{i+1} := u_{i+1} + \beta p_i$

```
 $r_0 := b - Ax_0; u_0 = M^{-1}r_0; p_0 = u_0;$   
for  $i = 0, \dots$  do :  
   $s := Ap_i$   
   $\alpha := \langle r_i, u_i \rangle / \langle s, p_i \rangle$   
   $r_{i+1} := r_i - \alpha s$   
   $u_{i+1} := M^{-1}r_{i+1}$   
   $\beta := \langle r_{i+1}, u_{i+1} \rangle / \langle r_i, u_i \rangle$   
   $x_{i+1} := x_i + \alpha p_i$   
   $p_{i+1} := u_{i+1} + \beta p_i$ 
```

Retour vers le passé



На французском языке

© Главная редакция физико-математической литературы
издательства «Наука», 1978

© Traduction française Editions Mir 1981

$\prod_{m=1}^{-1} S_{m,}$
 $\tau_{k+1}^{-1} r_k$ et en
 (14)
 $(E - T_p)^{-1}$
 e le schéma
 Si la norme
 e » de l'opé-
 naturel de

du problème sur le choix
 des paramètres d'itération

1. Famille de base des schémas itératifs. Au chapitre V on a
 montré que les problèmes de différences aux limites pour équations
 elliptiques constituent des systèmes spéciaux d'équations algébri-
 ques qui peuvent être assimilés à des équations opératorielles de
 première espèce

$$Au = f \tag{1}$$

dans l'espace hilbertien réel H . Dans quelques cas particuliers ces
 systèmes peuvent être résolus de façon efficace par des méthodes
 directes étudiées dans les chapitres I-IV. Dans le cas général l'une
 des méthodes approchées de résolution des équations de mailles
 elliptiques est la méthode itérative. On commencera l'étude des
 méthodes itératives par les méthodes à deux couches les plus simples,
 à savoir par la *méthode de Tchébychev* et la *méthode itérative simple*.

Pour la résolution approchée de l'équation (1) à opérateur li-
 néaire non dégénéré A donné dans H , examinons le *schéma itératif*
implicite à deux couches

$$B \frac{y_{k+1} - y_k}{\tau_{k+1}} + Ay_k = f, \quad k=0, 1, \dots, \tag{2}$$

avec approximation initiale quelconque $y_0 \in H$. $\{\tau_k\}$ est ici la suite
 des paramètres d'itération, quant à B , c'est un opérateur linéaire

us compliqués
plis. De (5)
st autoadjoint
nt dit
A) $D^{-1/2}$. (6)

ent, les hypo-
sont équiva-

nal des para-
le l'opérateur
e l'opérateur
du polynôme

d'opérateurs
doivent être
me $P_n(t) =$
mètres, soit

l'aspect $P_n(t) = \prod_{h=1}^n (1 - \tau_h t)$ dont le maximum du module sur le segment $[\gamma_1, \gamma_2]$ est minimal.

Réolvons ce problème. Vu que la forme du polynôme est déterminée par la condition de normalisation $P_n(0) = 1$, le problème posé se formule de la façon suivante : parmi tous les polynômes de degré n prenant au point $t = 0$ la valeur 1 trouver le polynôme s'écartant le moins de zéro sur le segment $[\gamma_1, \gamma_2]$ ne comprenant pas le point 0.

La solution de ce problème a été obtenue par le mathématicien russe V. A. Markov en 1892 et est donnée dans l'annexe. Le polynôme cherché $P_n(t)$ a la forme

$$P_n(t) \equiv q_n T_n \left(\frac{1 - \tau_0 t}{\rho_0} \right), \quad q_n = \frac{1}{T_n \left(\frac{1}{\rho_0} \right)}, \quad (1)$$

où $T_n(x)$ est le polynôme de Tchébychev de première espèce de degré n ,

$$T_n(x) = \begin{cases} \cos(n \arccos x), & |x| \leq 1, \\ \operatorname{ch}(n \operatorname{Arch} x), & |x| \geq 1, \end{cases}$$

$$q_n = \frac{2\rho_1^n}{1 + \rho_1^{2n}}, \quad \tau_0 = \frac{2}{\gamma_1 + \gamma_2}, \quad \rho_0 = \frac{1 - \frac{\gamma_1}{\sigma_1}}{1 + \frac{\gamma_1}{\sigma_1}}, \quad \rho_1 = \frac{1 - \sqrt{\frac{\gamma_1}{\sigma_1}}}{1 + \sqrt{\frac{\gamma_1}{\sigma_1}}}, \quad \sigma_1 = \frac{\gamma_1}{\gamma_2}. \quad (2)$$

nie par le rapport des constantes μ_1 et μ_2 des valeurs propres de D . Les cas de $D = AB^{-1}A$ et de $D = A^*A$ méritent une attention particulière. Avec ce choix de l'opérateur D la norme de l'erreur dans H_D peut être calculée au cours des itérations. En effet, avec $D = AB^{-1}A$, il vient

$$\|z_n\|_D^2 = (Dz_n, z_n) = (B^{-1}Az_n, Az_n) = (B^{-1}r_n, r_n) = (w_n, r_n),$$

et avec $D = A^*A$:

$$\|z_n\|_D^2 = (Az_n, Az_n) = (r_n, r_n),$$

où $r_n = Az_n = Ay_n - Au = Ay_n - f$ est le résidu de la n -ième itération et $w_n = B^{-1}r_n$ la correction. Ces grandeurs peuvent être obtenues au cours des itérations.

4. Stabilité de la méthode sous le rapport des calculs. En étudiant la convergence de la méthode de Tchébychev on a admis que le processus de calcul était parfait, c'est-à-dire que les calculs s'effectuaient avec un nombre infini de chiffres. Dans un calcul réel toutes les opérations de calcul se réalisent avec un nombre fini de chiffres et à chaque étape du calcul apparaissent des erreurs d'arrondi. Les erreurs d'arrondi associées aux opérations arithmétiques engendrent l'erreur de la méthode.

Dans les méthodes itératives, l'erreur de calcul de la méthode est constituée des erreurs impliquées par chaque itération. Si le nombre d'itérations est suffisamment grand et la méthode itérative est susceptible d'accumuler les erreurs d'arrondi de chaque itération,

Tableau 5

n	q_n	$\varepsilon_{\text{réel}}$			
		$\mathfrak{M}_n^{(1)}$	$\mathfrak{M}_n^{(2)}$	$\mathfrak{M}_n^{(3)}$	\mathfrak{M}_n^*
16	$8,79 \cdot 10^{-1}$	$8,14 \cdot 10^{-1}$	$8,14 \cdot 10^{-1}$	$8,14 \cdot 10^{-1}$	$8,14 \cdot 10^{-1}$
24	$7,58 \cdot 10^{-1}$	$9,62 \cdot 10^{-1}$	$7,11 \cdot 10^{-1}$	$7,11 \cdot 10^{-1}$	$7,11 \cdot 10^{-1}$
32	$6,30 \cdot 10^{-1}$	$3,38 \cdot 10^3$	$3,55 \cdot 10^2$	$5,63 \cdot 10^{-1}$	$5,63 \cdot 10^{-1}$
40	$5,09 \cdot 10^{-1}$	$3,07 \cdot 10^7$	$2,44 \cdot 10^6$	$5,03 \cdot 10^{-1}$	$4,85 \cdot 10^{-1}$
48	$4,04 \cdot 10^{-1}$	arrêt	$3,46 \cdot 10^{10}$	$2,47 \cdot 10^0$	$3,04 \cdot 10^{-1}$
56	$3,17 \cdot 10^{-1}$	—	$1,02 \cdot 10^{15}$	$2,29 \cdot 10^2$	$3,10 \cdot 10^{-1}$
64	$2,47 \cdot 10^{-1}$	—	arrêt	$1,87 \cdot 10^4$	$2,23 \cdot 10^{-1}$
72	$1,92 \cdot 10^{-1}$	—	—	$1,73 \cdot 10^6$	$1,72 \cdot 10^{-1}$
80	$1,49 \cdot 10^{-1}$	—	—	arrêt	$1,44 \cdot 10^{-1}$
...
256	$4,97 \cdot 10^{-4}$	—	—	—	$4,80 \cdot 10^{-4}$
...
512	$1,23 \cdot 10^{-7}$	—	—	—	$1,15 \cdot 10^{-7}$

Les résultats des calculs pour $N = 10$ sont donnés au tableau 5. Dans ce tableau, entre les suites mentionnées de paramètres $\mathfrak{M}_n^{(1)}$

ffit de fixer
i en qualité
coïncident,
aximale de

ormules

r_1). (15)

éfinies par

2, ..., n;

$k+1$.

... (16)

ve réelle.

$x_n = T_{n,0} x_0 +$

tire l'estima-

φ_j ||. (17)

indépendante

te de paramè-

tration pour

nsemble \mathfrak{M}_n .

é de manière

le.

On a donc montré que, pour tout $x \in H$, on a l'estimation

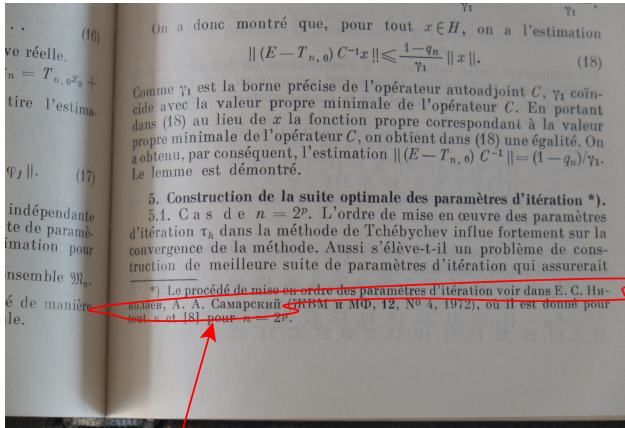
$$\|(E - T_{n,0}) C^{-1} x\| \leq \frac{1 - q_n}{\gamma_1} \|x\|. \quad (18)$$

Comme γ_1 est la borne précise de l'opérateur autoadjoint C , γ_1 coïncide avec la valeur propre minimale de l'opérateur C . En portant dans (18) au lieu de x la fonction propre correspondant à la valeur propre minimale de l'opérateur C , on obtient dans (18) une égalité. On a obtenu, par conséquent, l'estimation $\|(E - T_{n,0}) C^{-1}\| = (1 - q_n)/\gamma_1$. Le lemme est démontré.

5. Construction de la suite optimale des paramètres d'itération *).

5.1. Cas de $n = 2^v$. L'ordre de mise en œuvre des paramètres d'itération τ_k dans la méthode de Tchébychev influe fortement sur la convergence de la méthode. Aussi s'élève-t-il un problème de construction de meilleure suite de paramètres d'itération qui assurerait

* Le procédé de mise en ordre des paramètres d'itération voir dans E. C. Николаев, А. А. Самарский (ЖВМ и МФ, 12, N° 4, 1972), où il est donné pour tout n et [8] pour $n = 2^v$.



Samarskii & Nikolaev

Noël approche !

amazon.fr Premium

Livres anglais et étrangers ▾

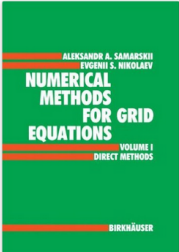
Parcourir les boutiques ▾

Chez Dumont Ventes Flash Chèques-cadeaux Vendre Aide

Livres anglais et étrangers Recherche détaillée Nos rubriques Nouveautés Meilleures ventes Bonnes affaires Livres audio Tous les livres Vendre

Livres anglais et étrangers > Science > Mathematics

Feuilleter ↴



Numerical Methods for Grid Equations (Anglais) Relié – 1 décembre 1988
de **A. A. Samarskii** (Auteur), **E. S. Nikolaev** (Auteur), **S. G. Nash** (Traduction)

[Soyez la première personne à écrire un commentaire sur cet article](#)

▸ Voir les formats et éditions

Relié
EUR 189,66 *Premium*

2 d'occasion à partir de EUR 58,92
6 neufs à partir de EUR 60,44

Toutes les idées cadeaux Livres [Cliquez ici](#)

Implantation et Résultats

Grille 128^3 éléments (512^3 inconnues).

Indexation des éléments par la fonction de Peano (= Z-curve) =>
réutilisation des données en cache.

Implantation et Résultats

Grille 128^3 éléments (512^3 inconnues).

Indexation des éléments par la fonction de Peano (= Z-curve) => réutilisation des données en cache.

► $Y = AX$.

197 Gflops/s, $3 \cdot 10^9$ Pts/s.

Sans réutilisation des caches : 95,2 Gflops/s.

Implantation et Résultats

Grille 128^3 éléments (512^3 inconnues).

Indexation des éléments par la fonction de Peano (= Z-curve) => réutilisation des données en cache.

▶ $Y = AX$.

197 Gflops/s, $3 \cdot 10^9$ Pts/s.

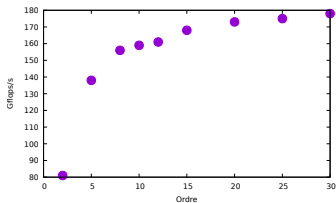
Sans réutilisation des caches : 95,2 Gflops/s.

▶ $Y' = AY$, Rock4.

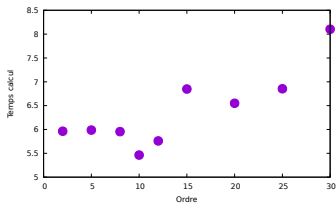
185 Gflops.

Résultats

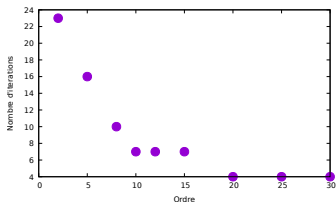
► $AX = F$, GCP.



Gflops/s.



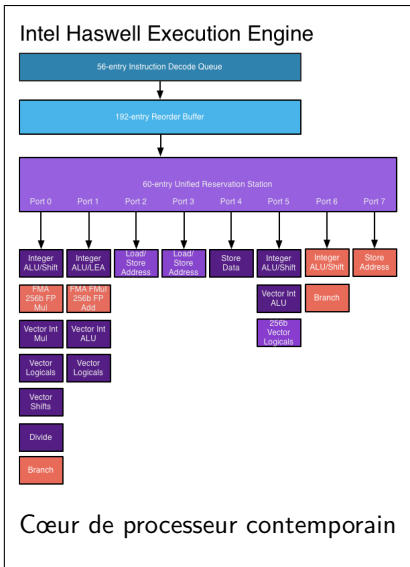
Temps calcul (s).



Nb. itérations.

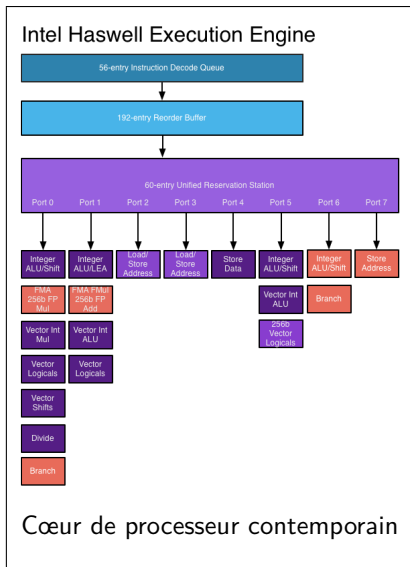
Peut-on faire mieux ?

- Mieux utiliser les cœurs superscalaires et limiter les accès au cache.



Peut-on faire mieux ?

- Mieux utiliser les cœurs superscalaires et limiter les accès au cache.

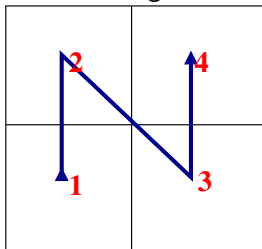


Outil Intel IACA.

Réorganiser le code à l'échelle « microscopique ».

Peut-on faire mieux ?

- ▶ Meilleur « tiling ».

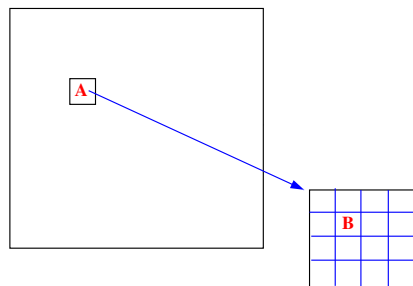


Ordre « Z » (Peano)

$$x = 0, x_1 x_2 \dots x_n.$$

$$\text{Peano}(x, y, z) = x_1 y_1 z_1 x_2 y_2 z_2 \dots x_n y_n z_n.$$

Peut-on faire mieux ?



Double tiling.

Boucles imbriquées sur les **A** et les **B**.

Tailles choisies pour que **A** tienne dans le cache L2 et **B** dans le cache L1.

Merci !