TD Maple n°7 : Matrices et algèbre linéaire

Lycée Louis le Grand

PCSI 1

Lundis 16 et 23 mars 2009

1 Tableaux

Avant d'aborder les structures de l'algèbre linéaire, intéressons nous un peu aux tableaux. Un tableau est une structure ordonnée à n lignes et p colonnes dans laquelle on peut stocker autant de variables que ce tableau a de cases (np cases). Avant d'utiliser un tableau, il faut le définir par l'instruction :

```
[> tab :=array(1..n,1..p) ;
```

Attention, les entiers n et p doivent être connus à l'avance.

`tab[i,j]` désigne la variable située à la i-ème ligne et à la j-ème colonne du tableau `tab`. `tab[i,j]` est une variable comme n'importe quelle autre variable Maple, on peut lui affecter une valeur, l'utiliser dans des calculs, etc... Tant que `tab[i,j]` n'a pas de valeur, Maple la considère comme n'importe quelle variable non-affectée et la note $tab_{i,j}$.

Pour faire afficher un tableau, on utilise l'instruction `print` :

```
[> print(tab) ;
```

Remarque : En fait, un tableau n'est pas forcément un tableau à deux entrées. Ça peut être un tableau à une entrée auquel cas il s'identifie à une liste ou à un vecteur :

```
[> tab :=array(1..10) ;
```

Pour accéder au i-ème élément, on tape `tab[i]`.

Mais ça peut aussi être un tableau à 5 entrées ou à 10 ou à 666 :

```
[> tab :=array(1..2,1..6,1..4,1..98,1..3) ;
```

Noter enfin qu'un tableau peut être indicé à partir de 0 mais dans ce cas (comme dans le cas d'un tableau à plus de deux entrées), l'instruction `print` permettant d'afficher le tableau renvoie une notation symbolique pas vraiment satisfaisante :

```
[> tab :=array(0..5) ;
[> print(tab) ;
```

$$ARRAY([0..5], [(0) = tab[0], (1) = tab[1], (2) = tab[2], (3) = tab[3], (4) = tab[4], (5) = tab[5]])$$

Très petit exercice

1. Créer un tableau A à deux lignes et trois colonnes,
2. Faire afficher A,
3. Affecter la valeur 7 à la case située en ligne 2, colonne 2,
4. Faire afficher A.

2 Structures de l'algèbre linéaire

Les instructions relatives à l'algèbre linéaire ne sont pas directement accessibles. Il faut d'abord charger la librairie *linalg* avec l'instruction `with(linalg)`. Après validation de cette instruction, Maple affiche toutes les fonctions relatives à l'algèbre linéaire. Elles sont nombreuses, nous ne verrons ici que les principales mais n'hésitez pas à consulter l'aide si vous voulez en utiliser d'autres.

2.1 Les matrices

Il y a deux façons pour saisir une matrice à n lignes et p colonnes. La première est d'utliser la syntaxe suivante :
`M :=matrix(n,p,[`$M_{11}, M_{12}, ..., M_{1p}, M_{21}, M_{22}, ..., M_{2p}, ..., M_{n1}, ..., M_{np}$`,])`.

On entre donc les coefficients un à un, en procédant ligne par ligne et de gauche à droite. Comme pour les tableaux, les entiers n et p doivent être connus à l'avance. Voici deux exmples :

```
[>M :=matrix(3,3,[0,1,-1,-3,4,-3,-1,1,0]) ;
[>M :=matrix(2,3,[x,y,x,u,v,z]) ;
```

L'autre manière de définir une matrice, c'est d'utiliser une fonction $f : [\![1, n]\!] \times [\![1, p]\!] \to \mathbb{K}$ où $\mathbb{K}$ est le corps auquel appartiennent les coefficients de la matrice :

```
[>f :=(i,j)->i*exp(5*I*Pi)+j) ;
[>M :=matrix(3,3,f) ;
```

ou encore :

```
[>M :=matrix(3,3,(i,j)->i*j) ;
```

Enfin, on peut créer une matrice aléatoire de n lignes et p colonnes par l'instruction :

```
[>M :=randmatrix(n,p) ;
```

Pour afficher le contenu d'une matrice, on utilise l'instruction `print(M)`. Pour accéder au terme $M_{i,j}$ d'une matrice, on utilise la syntaxe `M[i,j]`.

Pour connaître le nombre de lignes ou de colonnes d'une matrice, on utilise :

```
[>rowdim(M) ;
[>coldim(M) ;
```

La matrice nulle est simplement notée 0 et la matrice identité est notée `&*()`. Mais il peut être intéressant de créer un alias sur la matrice identité en utilisant l'instruction :

```
[>alias(Id=&*()) ;
```

2.2 Les vecteurs

Pour créer un vecteur, on utilise l'instruction `vector` :

```
[>v :=vector(4,[1,2,4,1]) ;
[>v[3] ;
```

Cette instruction crée un vecteur *colonne* bien qu'il soit écrit en ligne. Cependant, si vous avez besoin de faire la distinction entre vecteurs lignes et vecteurs colonnes (ce qui risque d'arriver assez souvent si vous faites de l'algèbre linéaire), il vaut mieux définir les vecteurs à partir de l'instruction `matrix` :

```
[>v :=matrix(1,4,[1,2,4,1]) ;
[>v :=matrix(4,1,[1,2,4,1]) ;
```

2.3 Opérations algébriques sur les matrices

2.3.1 Addition, élévation à une puissance entière, inversion

Ces trois opérations sur les matrices s'écrivent naturellement comme si l'on manipulait des réels. La seule différence, c'est que Maple n'effectue pas directement les calculs car les calculs sur les matrices sont assez coûteux. Maple ne fait les calculs qu'après demande express de votre part. Cette demande se fait par l'instruction

`evalm`. Tapez la suite d'instructions suivante et regardez ce qui se passe.

```
[>A :=matrix(2,2,[a,b,b,c]) ;B :=matrix(2,2,[a,d,c,b]) ;
[>C :=A+B ;
[>print(C) ;
[>evalm(C) ;
[>Acube :=A^3 ;
[>evalm(Acube) ;
[>Ainv :=A^(-1) ;
[>evalm(Ainv) ;
```

Evidemment, l'addition ne fonctionne qu'avec des matrices de mêmes dimensions et l'élévation à une puissance entière ainsi que l'inversion ne fonctionnent qu'avec des matrices carrées.

2.3.2 Produit de deux matrices

Le produit de deux matrices A et B (de dimensions compatibles) se fait par l'opérateur `&*` :

```
[>A :=matrix(2,2,[a,b,b,c]) ;B :=matrix(2,2,[a,d,c,b]) ;
[>C :=A&*B ;
[>evalm(C) ;
```

ou bien par l'intruction `multiply` :

```
[>C :=multiply(A,B) ;
[>evalm(C) ;
```

2.4 Opérateurs définis sur l'espace des matrices

```
[>M :=matrix(3,3,[1,-3,3,3,-5,3,6,-6,4]) ;
```

Rang d'une matrice :

```
[>rank(M) ;
```

Transposée d'une matrice :

```
[>tanspose(M) ;
```

Base du noyau d'une matrice :

```
[>kernel(M) ;
```

Base de l'image d'une matrice :

```
[>colspace(M) ;
```

Pour les matrices carrées, il y a d'autres opérateurs utiles :
Trace d'une matrice carrée :

```
[>trace(M) ;
```

Déterminant d'une matrice carrée :

```
[>det(M) ;
```

Valeurs propres[1] d'une matrice carrée (calculées numériquement ou formellement) :

```
[>eigenvals(M) ;
```

$$4, -2, -2$$

[1] On rapelle que si $M \in \mathcal{M}_n(\mathbb{K})$, une valeur propre λ est un scalaire appartenant à $\mathbb{K}$ définit par $\exists u \in \mathbb{K}^n - \{0\}, \quad Mu = \lambda u$

Chaque valeur propre est écrite autant de fois que son ordre de multiplicité. Ainsi, 4 est valeur propre simple et -2 est valeur propre double.

Vecteurs propres d'une matrice carrée :
`[>eigenvects(M) ;`

$$[4, 1, \{[1, 1, 2]\}], [-2, 2, \{[1, 1, 0], [-1, 0, 1]\}]$$

La réponse de Maple se lit comme suit : 4 est valeur propre simple et une base du sous-espace propre associé[2] est donnée par $[1, 1, 2]$, -2 est valeur propre double et une base du sous-espace propre associé est donnée par $[1, 1, 0], [-1, 0, 1]$[3].

2.5 La fonction map

`map` est une fonction indispensable pour manipuler des matrices. Elle permet d'appliquer une fonction à tous les éléments d'une matrice. Par exemple, `map(x->x^2,M)` met tous les éléments de la matrice M au carré.

3 Exercices

3.1 Exercice 1

Etant données deux matrices A et B appartenant respectivement à $\mathcal{M}_{n,p}(\mathbb{R})$ et $\mathcal{M}_{p,q}(\mathbb{R})$, la matrice produit $C = AB$ est définie par :

$$\forall i \in [\![1, n]\!], \quad \forall j \in [\![1, q]\!], \quad C_{ij} = \sum_{k=1}^{p} A_{ik} B_{kj}$$

1. Créer une procédure `multmat(A,B)` qui renvoie la matrice C sans utiliser le produit matriciel &* ou l'instruction `multiply`.
2. Tester votre procédure.

3.2 Exercice 2 : Noyau, image d'une application linéaire

Soit $A = X^4 - 1$, $B = X^4 - X$ et u l'application qui à un polynôme P de $\mathbb{R}_3[X]$, associe le reste de la division euclidienne de AP par B.

1. Montrer avec Maple que u est linéaire. On pourra utiliser la fonction `rem`.
2. Déterminer la matrice de u dans la base canonique de $\mathbb{R}_3[X]$. On pourra utiliser la fonction suivante qui renvoie les coefficients d'un polynôme p de degré d :
   `polyvect :=(p,d)->[seq(coeff(p,indets(p)[1],i),i=0..d)] ;`
3. Déterminer alors le noyau de u, son image, ses valeurs propres et ses vecteurs propres.

3.3 Exercice 3

1. Ecrire le plus rapidement possible la matrice $A = (a_{ij})_{1 \leq i,j \leq 5}$ telle que :

$$\begin{cases} a_{ii} = 1 - (\frac{1}{2})^{5-i} & \text{si} \quad i < 5 \\ a_{ii} = 1 & \text{si} \quad i = 5 \\ a_{ij} = (\frac{1}{2})^{5-j} & \text{si} \quad i = j + 1 \\ a_{ij} = 0 & \text{sinon} \end{cases}$$

2. On note v le vecteur de $\mathbb{R}^5$ tel que $v = (1, 0, 0, 0, 0)$ et f l'endomorphisme de $\mathbb{R}^5$ dont A est la matrice dans la base canonique. Pour tout $n \in \mathbb{N}$, on note u_n la 5-ème coordonnée de l'image de v par f^n. Imprimer la suite des couples (n, u_n) pour n variant de 1 à 50. On utilisera `multiply`.
3. Tracer le graphe $n \mapsto u_n$.
4. Trouver la plus petite valeur de n pour que u_n soit supérieur à 0,99.

[2] i.e. le sous-epace vectoriel $ker(M - \lambda Id)$ avec $\lambda = 4$.
[3] Cette matrice est donc diagonalisable dans $\mathcal{M}_n(\mathbb{R})$.