319 : Exercices faisant intervenir des algorithmes de calcul matriciel.

Prérequis :

* Matrices, déterminant, polynôme caractéristique, valeurs propres, vecteurs propres.

* Probabilités, vecteurs gaussiens.

Exercice 1 : Formule de Cramer

Soit $A \in GL_n(\mathbb{R})$ et $b \in \mathbb{R}^n$.

a) On note A_i le $i^{ème}$ vecteur colonne de A. On cherche à résoudre le système (S) $AX = b$, montrer que $x_i = \dfrac{det(A_1, \dots, A_{i-1}, b, A_{i+1}, \dots, A_n)}{det(A)}$ est l'unique solution de (S)

b) Déterminer un algorithme utilisant la fonction det pour résoudre (S) ; le programmer en scilab.

Très facile

Pour des étudiants qui viennent de découvrir : matrices déterminants et systèmes de Cramer

---

Exercice 2 : Algorithme de la puissance

Soit $A \in \mathcal{M}_n(\mathbb{R})$ diagonalisable, de valeurs propres réelles $(\lambda_1, \dots, \lambda_n)$, associées à des vecteurs propres $(e_1, \dots, e_n)$ et telles que $|\lambda_1| \leqslant \dots \leqslant |\lambda_{n-1}| < \lambda_n$. On pose $x_0 = \sum_{i=1}^{n} \beta_i e_i$ avec $\beta_n \neq 0$ et on définit par récurrence sur $k \geqslant 1$, $y_k = Ax_{k-1}$ et $x_k = y_k / \|y_k\|$.

1) Montrer que $\|y_k\| \underset{k\to\infty}{\longrightarrow} \lambda_n$ et $x_k \underset{k\to\infty}{\longrightarrow} \pm e_n/\|e_n\|$

2) Déterminer et écrire en Scilab un algorithme permettant d'obtenir une valeur approchée de λ_n et un vecteur propre associé approché.

Méthode numérique d'approximation

Application à certaines matrices symétriques

GASK p. 216

Exercice 3 : Décomposition LU

1) Soit $A \in \mathcal{M}_n(\mathbb{R})$ dont tous les mineurs principaux dominants sont non nuls. Montrer qu'il existe un unique couple (L, U), avec U triangulaire supérieure et L triangulaire inférieure à diagonale unité tel que $A = LU$.

2) Écrire un algorithme pour obtenir une telle décomposition et le tester.

Cas favorable de l'algorithme du pivot de Gauss

GASK p. 114 ou

XENS Alg. T2 p. 57

Exercice 4 : Cholesky

Soit $A \in \mathcal{M}_n(\mathbb{R})$, symétrique et définie positive.

1) Montrer qu'il existe $L \in \mathcal{M}_n(\mathbb{R})$ triangulaire inférieure à diagonale positive telle que $LL^t = A$

2) Donner un algorithme qui permet d'obtenir cette décomposition.

3) Déduire de ce qui précède une méthode pour simuler un vecteur aléatoire gaussien de moyenne m et de matrice de covariance A définie positive à partir d'un vecteur aléatoire gaussien centré réduit X.

GASK p. 121 pour la preuve de l'algorithme

Simulation de vecteurs aléatoires

Exercice 5 : Algorithme de Souriau-Faddeev

1) Soit $A \in \mathcal{M}_n(\mathbb{R})$.
On pose $\chi_A(X) = \det(XI_n - A)$. Montrer que $\chi'_A(X) = tr(com(XI_n - A))$.

2) On pose $B_0 = I_n$ et $B_k = AB_{k-1} - \frac{tr(AB_{k-1})}{k} I_n$ pour $1 \leqslant k \leqslant n-1$.
Montrer que $\chi_A(X) = X^n - tr(AB_0)X^{n-1} - \frac{tr(AB_1)}{2} X^{n-2} - \dots - \frac{tr(AB_{n-1})}{n}$.

3) Programmer un algorithme pour calculer, au signe près, les coefficients du polynôme caractéristique d'une matrice $A \in \mathcal{M}_n(\mathbb{R})$.

Utilisation de la trace, de la comatrice.
L'algorithme donne le poca et aussi A^{-1}

Gourdon alg. p. 215

Programmes en Scilab

```
function x=cramer(A,b)
    n=size(A,1);
    d=det(A);
    B=A;
    for i=1:n
        B(:,i)=b;
        x(i)=det(B)/d;
        B(:,i)=A(:,i);
    end;
endfunction
```

```
function [l,u]=puissance(A)
    n=size(A,1); x0=ones(n,1)/sqrt(n);
    converge=0; eps=1.e-6;
    iter=0; iterMax=100;

    while (iter<iterMax)&(~converge)
        u=A*x0;
        x=u/norm(u);
        converge=norm(x-x0)<eps;
        x0=x; iter=iter+1;
    end
    l=norm(u);
```

```
function [L,U]=FactoLU(A)
    [m,n]=size(A);
    if m~=n then, error('la matrice n''est pas carree'), end;
    zero=1.e-16;
    for k=1:n-1
        if abs(A(k,k))<zero then, error('erreur :-pivot nul'), end;
        for i=k+1;n
            A(i,k)=A(i,k)/A(k,k);
            for j=k+1:n
                A(i,j)=A(i,j)-A(i,k)*A(k,j);
            end;
        end;
    end;
    U=triu(A); L=A-U+eye(n,n);
endfunction
```

```
function A=cholesky(A)
    [m,n]=size(A);
    if m~=n then, error('la matrice n''est pas carree'),end;
    zero=1.e-16;
    if norm(A-A','inf')>zero then
        error('matrice non symetrique')
        end;
    for j=1:n
        for k=1:j-1
            A(j,j)=A(j,j)-A(j,k)*A(j,k);
        end;
        if A(j,j)<zero then, error('matrice non positive'), end;
        if abs(A(j,j))<zero then, error('matrice non definie'), end;
        A(j,j)=sqrt(A(j,j));
        for i=j+1:n
            for k=1:j-1
                A(i,j)=A(i,j)-A(j,k)*A(i,k);
            end
            A(i,j)=A(i,j)/A(j,j);
        end;
    end;
    A=tril(A);
endfunction
```

```
function [B,p]=faddeev(A)
    n=size(A,1);
    B=eye(n,n);
    p=ones(1,n+1);

    for i=2:n+1
        p(i)=-trace(A*B)/(i-1);
        if(i~=n+1) then B=A*B+p(i)*eye(n,n);end;
    end
    if(p(n+1)~=0) then B=-B/p(n+1);
        else B=("A n''est pas inversible"); end;
endfunction
```

[GASK] : "Algèbre linéaire numérique" de Grégoire Allaire et Sidi Mahmoud Kaber

Pour les programmes en Scilab (sauf faddeev) voir :
"Introduction à Scilab, Exercices pratiques, corrigés d'algébre linéaire" des mêmes auteurs.

José Gregorio : http://agregorio.net