Maîtrise de Mathématiques Fondamentales et Applications
Grands Systèmes Linéaires
feuille de TP 8

26 novembre 2009

Exercice - 1 *Méthode de la puissance*

Soit $A \in \mathcal{M}_n(\mathbb{R})$ une matrice inversible.

1- Méthode des puissances itérées
Écrire une fonction MATLAB de prototype `[lambda,u,nr]=puissance(A, tol, itermax)`
qui calcule la pus grande (en valeur absolue) des valeurs propres d'une matrice inversible A. Elle prend en entrée

- une matrice A,
- *itermax* un nombre maximum d'itérations,
- *tol* un paramètre ε de test d'arrêt pour la convergence.

Et en sortie elle fournit

- la valeur propre *lambda*,
- le vecteur propre associé u,
- le nombre réel d'itérations effectués.

2- Méthode des puissances itérées inversées
Écrire une fonction MATLAB de prototype `[lmda,u,nr]=puissanceI(A, tol, itermax)`
qui calcule la pus petite (en valeur absolue) valeur propre d'une matrice inversible A. Les arguments sont comme ci-dessus.

3- Méthode de déflation
a- Écrire une fonction MATLAB de prototype `function [lambda,V,N]=Deflation(A,tol,itermax)`
qui fait appel à la méthode des puissances itérées, pour déterminer toutes les valeurs propres (qu'on suppose simples) de la matrice A. Les arguments d'entrées sont comme ci-dessus. Les arguments en sortie sont :

- un vecteur *lambda* contenant les valeurs propres de la matrice A,
- une matrice V dont les colonnes sont les vecteurs propres de A associés aux valeurs propres *lambda*,
- un vecteur N correspondant au nombre d'itérations réellement effectuées pour le calcul de chaque valeur propre.

b- Évaluer cette fonction sur la matrice du laplacien obtenue par différences finies `A = laplace1D(20)`, en prenant `tol = 1e-11, itermax = 600`. Et comparer les résultats obtenus avec ceux fournis par la commande `eig(A)`.

Exercice - 2 *Méthode de Jacobi de calcul de valeurs et vecteurs propres*

Soit $A \in \mathcal{M}_n(\mathbb{R})$ une matrice symétrique définie positive. On se propose de déterminer les valeurs et vecteurs propres de la matrice A.

1- Matrice de rotation de Givens

a- Expliquer la construction d'une telle matrice de Givens pour les indices p, q de la matrice A.
b- Écrire une fonction MATLAB de prototype `function [B,p,q]=Givens(A)`
qui effectue les opérations suivantes :

- localise les indices $p < q$ telle que $|A_{p,q}| = \max_{i<j} |A_{i,j}|$
- retourne la matrice B de Givens associée à ces indices.

c- Générer aléatoirement une matrice symétrique A.

- Déterminer la matrice `[B,p,q]=Givens(A)`, et calculer `A1 = B' * A * B`.
- Comparer la norme de Frobenius de A et celle de $A1$.
- Comparer la norme euclidienne des vecteurs formés des éléments diagonaux de A et de $A1$.
- Comment se comporte le carré des termes extra-diagonaux de $A1$ par rapport celui de A ?
- Vérifier que $A1$ ne diffère de A que par les lignes et colonnes d'indices p, q.

2- Méthode de Jacobi.
Écrire une fonction MATLAB de prototype `function[A,V,nr]=MonJacobi(A,tol,itermax)`
qui prend en entrée :

- une matrice symétrique A,
- une valeur *tol* désignant un petit paramètre pour tester la convergence, qui a lieu lorsque la somme des carrées des coefficients extra-diagonaux de A est inférieure à *tol*,
- un entier *itermax* représentant le nombre maximal d'itérations,

et qui retourne

- une matrice A (*la matrice d'entrée A est écrasée*) diagonale dont les termes diagonaux sont les valeurs propres de A,
- une matrice V dont les colonnes sont les vecteurs propres de A associés aux valeurs propres de A dans l'ordre de leur apparition,
- un entier nr correspondant au nombre d'itérations réellement effectuées.

3- Essai d'évaluation de la méthode de Jacobi
a- Modifier la fonction `MonJacobi` en `function[A,V,nr,N]=MonJacobiM(A,tol,itermax)`, où N est un vecteur tel que $N(k)$ représente le nombre de coefficients de A de module supérieur à *tol*, à l'itération k. [*On pourra utiliser la commande* `N(k)= length(find(abs(A)>tol))`].

b- Générer la matrice du laplacien `A=laplca1D(30)`. Et pour `tol = 1e-11, itermax=6000`,

- exécuter `[lmda,V,nr,N]=MonJacobiM(A,tol,itermax)`,
- représenter sur un graphique `figure(1)`, l'évolution du nombre d'éléments de A de module supérieur à *tol*. Qu'observe-t-on ?
- exécuter la commande
  `figure(2);l= sort(diag(lmda));plot(1 :length(l),l,'r*-',1 :length(l),eig(A),'b-');`
  Commenter les observations.

c- Présenter quelques inconvénients de la méthode de Jacobi.