© *J.-B. A. K. <jean-baptiste.apoung@math.u-psud.fr>*

Fiche de TP 4 : Méthodes de Lanczos pour calcul de valeurs/vecteurs propres.

On s'intéresse à la détermination des valeurs/vecteurs propres d'une matrice symétrique réelle A, creuse et de grande taille. Afin de mener à bien la résolution des exercices proposés, on fera usage des utilitaires :

- *GSL pour le stockage des matrices et vecteurs.*
- *Gnuplot pour la représentation graphique des matrices.*

Une présentation succincte de ces utilitaires est accessible sur Dokeos dans le dossier dédié au présent cours.

Note 1 (Principe de la méthode de lanczos)**.**

Le principe de la méthode de Lanczos consiste en deux points :

1. *Appliquer l'algorithme de Lanczos (qui est une simplification de l'algorithme d'Arnoldi lorsque la matrice est symétrique) pour construire une base de orthonormée de l'espace de l'espace de Krylov*

$$K_{k-1} = \{r_0, Ar_0, \ldots, A^{k-1}r_0\} \overset{def}{\equiv} \mathcal{K}_k(A, r_0)$$

Le résultat de cette procédure s'écrit sous la forme :

$$A\, V_k = V_k\, T_k + v_{k+1} e_k^*$$

où V_k est la matrice dont les colonnes forment la base orthonormée de K_{k-1} , T_k est une matrice tridiagonale symétrique e_k est le k-ième vecteur canonique de l'espace $\mathbb{R}^k$, e_k^* son adjoint (transposé dans notre cas réel) et v_{k+1} le vecteur qui, normalisé, serait le prochain élément de la base de K_k.

2. *Calculer les valeurs et vecteurs propres de la matrice tridiagonale T_k (soit analytiquement soit par une méthode de dichotomie dite de Givens). Et déduire les approximations des vecteurs et valeurs propres de A.*

On constate que pour cette méthode :

- la matrice A n'intervient que sous forme de produit matrice-vecteur ;
- V_k peut être stockée comme une matrice de taille $n \times k$ ou comme un tableaux de k vecteurs de taille n chacun.
- T_k étant tridiagonale et symétrique, elle peut être stockée en utilisant deux vecteurs ; un de taille k pour la diagonale principale et un autre de taille $k-1$ pour la première sur-diagonale.

Thème - 1 *Méthode de bissection (ou dichotomie) de Givens*

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

$$A = \begin{bmatrix} b_0 & c_0 & & 0 \\ c_0 & \ddots & \ddots & \\ & \ddots & \ddots & c_{n-2} \\ 0 & & c_{n-2} & b_{n-1} \end{bmatrix},$$

où $c_i \neq 0, \quad \forall 0 \leq i \leq n-1$.

Q-1 : On désigne par A_i la sous-matrice principale d'ordre i de A et on pose $p_i(\lambda) = \det(A_i - \lambda I_i)$, son polynôme caractéristique.

Q-1-1 : En développant le déterminant $\det(A_i - \lambda I_i)$ par rapport à la dernière colonne, montrer que la suite p_i vérifie la relation de récurrence

$$\left\| \begin{array}{l} p_0(\lambda) = 1, \quad p_1(\lambda) = b_0 - \lambda, \\ p_i(\lambda) = (b_{i-1} - \lambda)p_{i-1}(\lambda) - c_{i-2}^2 p_{i-2}(\lambda), \quad \forall i \geq 2. \end{array} \right.$$

Q-1-2 : Écrire une fonction **C** de prototype

```
double pc(const gsl_vector* b, const gsl_vector* c, int i, double lambda)
```

définissant la $i-$ème fonction de cette suite, où b et c sont respectivement des vecteurs de taille n et $n-1$ représentant la diagonale principale et la première sur-diagonale de la matrice tridiagonale symétrique A.

Q-1-3 : Écrire une fonction **C** de prototype

```
int Nracines(const gsl_vector* b, const gsl_vector* c, int i, double mu)
```

qui calcule pour une matrice tridiagonale symétrique A (stockée dans les vecteurs b et c), le nombre de racines de p_i qui sont strictement inférieures à **mu**, comme étant le nombre de changement de signe entre éléments consécutifs de l'ensemble $\{1, \text{sgn}p_1(\mu), \ldots \text{sgn}p_i(\mu)\}$, où on a posé

$$\text{sgn}p_i(\mu) = \left\{ \begin{array}{ll} signe\ de\ p_i(\mu) & si\ p_i(\mu) \neq 0, \\ signe\ de\ p_{i-1}(\mu) & si\ p_i(\mu) = 0. \end{array} \right.$$

Q-2 : On suppose que la matrice A a n valeurs propres distinctes $\lambda_0 < \ldots < \lambda_{n-1}$.

Q-2-1 : Soit $0 \leq i \leq n-1$ un entier. On suppose que la valeur propre λ_i se trouve dans l'intervalle $[r_0, s_0]$. Montrer que :
Nracines$(A, n, \frac{r_0+s_0}{2}) \geq i+1 \Longrightarrow \lambda_i \in [r_0, \frac{s_0+r_0}{2}[$
Nracines$(A, n, \frac{r_0+s_0}{2}) < i+1 \Longrightarrow \lambda_i \in [\frac{r_0+s_0}{2}, s_0]$.

En déduire qu'on peut déterminer la valeur propre λ_i par dichotomie avec une précision ε donnée.

Q-2-2 : Ecrire une fonction **C** de prototype **double bissectionGivens(const gsl_vector* b, const gsl_vector* c, int i, double r0, double s0, double tol)** qui détermine la $i-$ème valeur propre de la matrice tridiagonale **A** (stockée sous forme bande dans les vecteurs b et c), avec une précision **tol**, où **[a0,b0]** est un intervalle de départ contenant ladite valeur propre (on pourra prendre $r0 = -\|A\|_\infty$, $s0 = \|A\|_\infty$).

Thème - 2 *Méthode de Lanczos*

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

Q-3 : Programmer l'algorithme de Lanczos à travers une fonction **C**

Prototype de fonction pour l'algorithme de Lanczos

```
void lanczos(const MatriceCSR* A, gsl_vector* r0, int k, gsl_matrix* V, gsl_matrix* T, int* k0)
```

qui prend comme arguments

- la matrice symétrique A,
- un vecteur r_0 et un entier k définissant l'espace de Krylov $K_k = \{r_0, Ar_0, \ldots, A^k r_0\}$,

et fournit en retour

- un entier k_0 égal à k ou coïncidant avec la dimension critique de Krylov si k est supérieur à la dimension critique de Krylov.
- une matrice V de taille $n \times k_0$ dont les colonnes sont les k_0 premiers vecteurs de la base orthonormée de l'espace de Krylov K_{k_0},
- une matrice T tridiagonale de taille $k_0 \times k_0$ telle que $V^*AV = T$.

L'algorithme en pseudo code est de la forme

Algorithme de Lanczos

```
-------------------------------------------------------------------------------------------------
     ALGORITHME DE LANCZOS                      |                  COMMENTAIRES
-------------------------------------------------------------------------------------------------
Donnees: A, r0, k                               |
Resultat V, T, k0                               |
------------------------------------------------|------------------------------------------------
Initialization :                                |
                                                |
w = r0/|| r_0 ||                                 |
T mis a 0                                       |
V(:,0) = w                                      |  GSl dispose d'une fonction  gsl_matrix_set_col
for j = 0 à k-1                                 |
    w = A * V(:,j);                             |  GSl dispose d'une fonction  gsl_matrix_get_col
    for i = max(0,j-1) à   j                    |
        T(i,j) =  (w, V(:,i));                  |
        w = w - T(i,j) * V(:,i);                |
    end                                         |
    normj = ||w||; T(j+1,j) = normj             |
                                                |
    if(normj !=  0 )                            | Il vaut mieux faire: normj > epsilon
        V(:,j+1) = w/normj                      | GSl dispose d'un fonction  gsl_matrix_set_col
    else                                        |
        k0 = j STOP.                            |
        return                                  |
    end                                         |
 end                                            |
k0 = k;                                         |
-------------------------------------------------------------------------------------------------
Pour la programmation en C , T arrive avec une taille kmax x kmax, et V avec une taille n x kmax
En sortie de la fonction, comme on dispose de k0, on peut redimensionner V et T. ( kmax = k+1 )
```

Q-4 :

Q-4-1 : Générer aléatoirement une matrice symétrique définie positive A de taille $n \times n$, et déterminer sa plus grande valeur propre λ_A à l'aide de la fonction (des précédents Tps) **valeurs_propres_extremes**.

Q-4-2 : Faire varier $k = 2, \ldots n$, et comparer à chaque fois la plus grande valeur propre de T obtenue par la méthode de bissection de l'exercice précédent à λ_A.
(*On pourra prendre pour r_0 le premier vecteur colonne de la matrice A*).

Q-4-3 : A-t-on toujours $AV = VT$? Comment évolue la valeur absolue de la différence des plus grandes valeurs propres de A et de T en fonction de la norme de Frobenius de $AV - VT$? En fonction de la dernière composante du vecteur propre de T associé à sa plus grande valeur propre ?