UNIVERSITE DE GENEVE Section de mathématiques

Faculté des sciences

Méthodes itératives – Corrigé série 4

Corrigé exercice 1 a) Une rotation de Givens est une matrice de la forme :

$$\mathbf{G}^{(i,j,\theta)} = \begin{bmatrix} 1 & & & 0 & & & & 0 & & & \\ & \ddots & & \vdots & & & & \vdots & & & \\ & & 1 & 0 & & & & 0 & & & \\ 0 & \cdots & 0 & \cos\theta & 0 & \cdots & 0 & \sin\theta & 0 & \cdots & 0 \\ & & & 0 & & & & 0 & & & \\ & & & \vdots & & & & \vdots & & & \\ & & & 0 & & & & 0 & & & \\ 0 & \cdots & 0 & -\sin\theta & 0 & \cdots & 0 & \cos\theta & 0 & \cdots & 0 \\ & & & 0 & & & & 0 & 1 & & \\ & & & \vdots & & & & \vdots & & \ddots & \\ & & & 0 & & & & 0 & & & 1 \end{bmatrix}$$

où les $\cos\theta$ et $\sin\theta$ apparaissent aux lignes et colonnes i, j, $i < j$. Commençons par considérer les matrices 2×2

$$\hat{\mathbf{G}} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \qquad \mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

$\mathbf{G}$ est obtenue en extrayant les lignes et colonnes i et j de la matrice précédente. Nous cherchons à choisir θ telle que l'élément $(\hat{\mathbf{G}}\mathbf{A})_{21} = 0$. L'équation à résoudre est $-a\sin\theta + c\cos\theta = 0$. Si $a \neq 0$, cela équivaut à $\tan\theta = \frac{c}{a}$, soit $\theta = \arctan\frac{c}{a}$. Si $a = 0$et $c \neq 0$, cela équivaut à $\theta = \pi/2$.

On peut donc procéder à la décomposition **QR** d'une matrice de Hessenberg $\mathbf{H}$ de taille n en choisissant une suite de $n-1$ rotation de Givens $(\mathbf{G})_l$ pour lesquelles (i,j) prend succesivement les valeurs $(1,2), (2,3), (3,4), \ldots, (k-1,k)$ où chaque rotation de Givens $\mathbf{G}_l$ annule l'élément sous la diagonale situé sur la l–ième colonne. En pseudo code l'algorithme s'écrit :

```
R=H
Pour i allant de 1 à k-1
    si (R(i,i) ~=0) alors
         theta=arctan(R(i+1,i)/R(i,i))
    sinon
       theta=pi/2
    finsi
    R(i,:)=cos(theta)*R(i,:)+sin(theta)*R(i+1,:)
    R(i+1,:)=cos(theta)*R(i+1,:)-sin(theta)*R(i,:)
    thetatab(i)=theta % Pour l'exercice suivant
finpour
```

Il existe un mode de calcul de $\cos\theta$ et $\sin\theta$ permettant d'éviter de calculer theta. Il est bien plus stable numériquement.

b) Il faut avoir gardé en mémoire l'ensemble des rotations de Givens ayant été appliqué sur les $k-1$ premières colonnes. On applique alors toutes ces rotations dans l'ordre sur la k–ième colonnes. Il ne reste plus qu'à calculer la dernière rotation de Givens.

Corrigé exercice 2 Si on connait une bonne approximation initiale x_0, il suffit d'appliquer GMRES au problème $\mathbf{A}\boldsymbol{x}' = \boldsymbol{b} - \mathbf{A}\boldsymbol{x}_0$, puis de définir $\boldsymbol{x}_k = \boldsymbol{x}'_k + \boldsymbol{x}_0$.

Corrigé exercice 3 On vérifie immédiatement :

$$\mathbf{A}^2 = \begin{bmatrix} \mathbf{I} & 2\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix},$$

donc $\mathbf{A}^2 - 2\mathbf{A} + \mathbf{I} = \mathbf{0}$. En particulier, $\mathbf{A}^{-1}\boldsymbol{b} = 2\boldsymbol{b} - \mathbf{A}\boldsymbol{b}$. Ceci pour tout $\boldsymbol{b}$, donc $\mathbf{A}^{-1}\boldsymbol{b}$ appartient à l'espace de Krylov $\mathcal{K}_2(\mathbf{A}, \boldsymbol{b})$.

Corrigé exercice 4 a) On emploie les formules trigonométriques usuelles

$$\begin{aligned} \cos(a+b) &= \cos(a)\cos(b) - \sin(a)\sin(b), \\ \cos(a-b) &= \cos(a)\cos(b) + \sin(a)\sin(b), \end{aligned}$$

donc pour $t \in [0,1]$:

$$\begin{aligned} C_{k+1}(t) &= C_k(t)\cos(\arccos(t)) - \sin(k\arccos(t))\sin(\arccos(t)), \\ &= tC_k(t) - \sin(k\arccos(t))\sin(\arccos(t)), \\ C_{k-1}(t) &= tC_k(t) + \sin(k\arccos(t))\sin(\arccos(t)). \end{aligned}$$

Sommons les deux inégalités obtenues et on obtient le résultat demandé :

$$C_{k+1}(t) + C_{k-1}(t) = 2C_k(t).$$

b) Soit $q(t)$ un polynôme tel que $q(0) = 1$ et dont la norme $L^\infty([\alpha, \beta])$ est inférieure ou égale à celle du polynôme p défini dans l'énoncé. Le but est de démontrer que $p - q$ est nulle. Pour ce faire, on va compter le nombre de 0 de $p - q$. Tout d'abord, rappelons les propriété du polynôme p :
- $p(0) = 0$.
- p atteint $k+1$ fois son extremum sur $[\alpha, \beta]$. Il atteint son extremum noté ξ aux points $x_i = \alpha + \frac{\arccos(\frac{i}{k}\pi)}{|\beta-\alpha|}$. On a $p(x_i) = (-1)^k\xi$.

Notons $r = p - q$, nous avons $r(x_i) \leq 0$ quand i est impair et $r(x_i) \geq 0$ quand i est pair. De plus, pour i entre 1 et $k-1$, $r(x_i) = 0$ implque que q atteint un extremum en x_i et donc $r'(x_i) = 0$. Ceci nous permet d'avoir l'alternative suivante pour tout intervalle $(x_i, x_i + 1)$:
- $r(x_i)r(x_{i+1}) < 0$
- x_i ou x_{i+1} sont racines doubles du polynôme r.
- $i = 0$ et α est racine du polynôme r ou $i = k - 1$ et β est racine du polynôme r.

Cela nous permet de comptabiliser une racine de r par intervalle, soit k racines. Il y en a aussi une en 0 car $p(0) = q(0) = 1$. Par conséquent r a $k + 1$ racines et est nul. Donc $p = q$.

Corrigé exercice 5 a) Code Pour Arnoldi

```
function [Q,H]=ARNOLDI(A,b,k)
% ARNOLDI: Calcule une base orthogonale d'un espace de Krylov
%  [Q,H]=ARNOLDI(A,b,k) Calcul d'une base orthogonale de l'espace
%  de Krylov engendré par (b,Ab,...,A^(k-1)b) par la méthode de
%  Arnoldi. Renvoie cette base stockée en vecteurs colonnes dans
%  Q et la matrice Hessenberg des coefficients d'orthogonalisation dans H
if nargin < 2
  error('pas assez d arguments')
end
if nargin < 3
  k=lentgh(b);
end

k=min(length(b)+1,k);

Q(:,1)=b/norm(b);
i=1;
```

```
while i<k && norm(Q(:,i))~=0 % ou abs(norm(...))< petit nombre
  Q(:,i+1)=A*Q(:,i);
  for j=1:i
    H(j,i)=Q(:,i+1)'*Q(:,j);
    Q(:,i+1)=Q(:,i+1)-H(i,j)*Q(:,j);
  end
  H(i+1,i)=norm(Q(:,i+1));
  if H(i+1,i)~=0
    Q(:,i+1)=Q(:,i+1)/H(i+1,i);
  end
  i=i+1;
end
```

Code Pour Lanczos

```
function [Q,T]=LANCZOS(A,b,k)
% LANCZOS: Calcule une base orthogonale d'un espace de Krylov
%   [Q,T]=LANCZOS(A,b,k)  Calcul d'une base orthogonale de
%   l'espace de Krylov engendré par (b,Ab,...,A^(k-1)b) par
%   la m?thode de Lanczos. A doit etre une matrice symétrique.
%   Renvoie cette base stockée en vecteurs colonnes dans Q
%    et la matrice tridiagonale des coefficients
%    d'orthogonalisation dans T.
if nargin < 2
  error('pas assez d arguments')
end
if nargin < 3
  k=lentgh(b);
end

k=min(length(b),k);

Q(:,1)=b/norm(b);
beta=0
i=1
while i<=k && norm(Q(:,i))~=0 % ou abs(norm(...))< petit nombre
  Q(:,i+1)=A*Q(:,i);
  Q(:,i+1)=Q(:,i+1)-beta*Q(:,max(1,i-1));
  T(max(i-1,1),i)=beta;

  T(i,i)=Q(:,i+1)'*Q(:,i);
```

```
  Q(:,i+1)=Q(:,i+1)-T(i,i)*Q(:,i);
  beta=norm(Q(:,i+1));
  if T(i+1,i)~=0
    Q(:,i+1)=Q(:,i+1)/beta;
  end
  T(i+1,i)=beta;
  i=i+1;
end
```

b) Code Pour GMRES

```
function [x,Res]=GMRES(A,b,tol)
% GMRES: Résolution d'un système linéaire par projection sur Krylov
%    [x,R]=GMRES(A,b,tol)
%   Calcule x dans l'espace engendré par (b,...,A^(k-1)b) réalisant
%   le minimum de norm(b-A*x,2) ou k est le plus petit entier tel
%   que cette norme est inferieur à tol. Au pire, GMRES s'arrête
%   apres size(A,1) itérations. R contient l'historique de la norme
%    des résidus.
%
%    [x,Res]=GMRES(A,b) équivaut à GMRES(A,b,1e-6)

if nargin < 2
  error('pas assez d arguments')
end
if nargin < 3
  tol=1e-6;
end

Nres(1)=norm(b);
SecM(1)=norm(b);
Q(:,1)=b/norm(b);
i=1;
while abs(NRes(i))>tol  %Arnoldi
  Q(:,i+1)=A*Q(:,i);
  for j=1:i
    H(j,i)=Q(:,i+1)'*Q(:,j);
    Q(:,i+1)=Q(:,i+1)-H(i,j)*Q(:,j);
  end
  H(i+1,i)=norm(Q(:,i+1));
 Q(:,i+1)=Q(:,i+1)/H(i+1,i); %Applications de toutes les
```

```
   %rotations de Givens
%précédentes à la dernière
%colonne de H
  R(i+1,i)=0; %Changement de taille
  R(:,i)=H(:,i);
  for j=1:i-1
    tmp=c(j)*R(j,i)+s(j)*R(j+1,i);
    R(j+1,i)=-s(j)*R(j,i)+c(j)*R(j+1,i);
    R(j,i)=tmp;
  end  %Calcul des nouveaux
   %paramètres pour la dernière
%rotation de Givens.
  c(i)=R(i,i)/sqrt(R(i,i)^2+R(i+1,i)^2);
  s(i)=R(i+1,i)/sqrt(R(i,i)^2+R(i+1,i)^2); %Application de la derniere
   %rotation à la derniere colonne
  R(i,i)=c(i)*R(i,i)+s(i)*R(i+1,i);
  R(i+1,i)=0; %Application de la dernière rotation au second membre
  SecM(i+1)=-s(i)*SecM(i);
  SecM(i)=c(i)*SecM(i); %Calcul de la norme du résidu
  NRes(i+1)=SecM(i+1);
  i=i+1;
end
%Calcul de y
y=R\SecM;
%Calcul de x
x=Q*y;
```

c) Script de calcul

```
% Serie 4 script

G=numgrid('L',14);
A=delsq(G);
b=ones(size(A,1),1);

[Q,T]=Lanczos(A,b,15);
Ttilde=T(1:size(T,2),:);
[V,D]=eig(Ttilde);
D=diag(D);
[Value,Argmin]=min(D);
v=Q(:,1:size(Q,2)-1)*V(:,Argmin);
```

```
u=G;
u(G>0)=v(G(G>0));
figure;
mesh(u);
title('Premier mode propre');%Rajouter valeur propre
                             %Rajouter nombre pour Lanczos.


%Tester GMRES
A=delsq(numgrid('S',16));%Construction du second membre
n=14;
b=ones(size(A,1),1);%Calcul des param?tres optimaux:
rhoJ=(cos(pi/(n+1))+cos(pi/(n+1)))/2;
wSOR=2/(1+sqrt(1-rhoJ^2));

[x,resGMR]=GMRES(A,b);
[x,resS]=SOR(A,b,wSOR);
[x,resJ]=Jacobi(A,b);
[x,resG]=GaussSeidel(A,b);
figure
  semilogy(1:length(resS),resS,'b',1:length(resGMR),resGMR,'g', ...
    1:length(resJ),resJ,'r',1:length(resG),resG,'k')
   Legend('SOR','GMRES','Jacobi','Gauss-Seidel');
```