MACS 3 Calcul Haute Performance. Module 2
TP A rendre le 10 octobre 2015

Analyse des méthodes itératives

1 Le laplacien dans un carré

1) Créer la fonction `A2d(n)` qui forme la matrice des différences finies 2D pour le laplacien avec conditions de Dirichlet dans un carré pour un maillage de même taille dans les deux directions. n est le nombre de points intérieurs dans chaque direction. On utilisera les fonctions `delsq` et `numgrid` de matlab. Comparer avec l'utilisation du programme suivant

```
h=1/(n+1);
e=4/h^2*ones(n^2,1);                % diagonal of the discrete Laplacian
e1=-1/h^2*repmat([ones(n-1,1);0],n,1);
e2=-1/h^2*ones(n^2,1);
B=zeros(n^2,3);
B(1:length(e2),1)=e2;                % prepare format for spdiags
B(1:length(e1),2)=e1;
B(1:length(e),3)=e;
B(1:length(e1),4)=flipud(e1);        % flipud because of spdiags conventions
B(1:length(e2),5)=flipud(e2);

A=spdiags(B,[-n  -1 0 1 n ],n^2,n^2);
```

2) Calculer de façon analytique et numérique les valeurs propres et le conditionnement de la matrice. On pourra comparer avec les résultats donnés par les fonctions `eig` et `cond` de matlab.

3) Tracer la courbe du conditionnement en fonction de n.

2 Les méthodes itératives stationnaires

1) Créer les fonctions `GaussSeidel(A,b,x0,tol,maxiter)`, `Jacobi(A,b,x0,tol,maxiter)` et `SOR(A,b,relax,x0,tol,maxiter)` . Calculer le paramètre optimal pour SOR et les facteurs de convergence théoriques `[rhoJ rhoGS rhoSOR ]`.

2) On se donne

```
b=rand(N^2,1);
x0=zeros(size(b));
```

Résoudre le système linéaire $Ax = b$ pour n=10. Tracer les courbes de convergence en échelle semilog (expliquer pourquoi)

```
semilogy(0:length(resJ)-1,resJ,'-r',0:length(resGS)-1,resGS,'-b',...
    0:length(resSOR)-1,resSOR,'-m')
xlabel('iteration');
```

```
ylabel('residual');
legend('Jacobi','Gauss Seidel','SOR');
title(['finite differences 2D, n=',int2str(n)])
```

Où intervient le facteur de convergence ? Quel est le taux de convergence numérique ? Le comparer avec le théorique.

3) Calculer les valeurs du taux de convergence numérique pour $n, 2n, 4n$ pour les 3 méthodes. Les placer sur une courbe, conclure.

4) Trouver les fonctions de `matlab` correspondantes. Comparer.

3 Les méthodes de Krylov

1) Gradient conjugué : Utiliser la fonction `CG(A,b,x0,tol,maxiter)` suivante

```
function [x,res]=CG(A,b,x0,tol,maxiter);
% CG conjugate gradient method
% [X,R]=CG(A,b,x0,n) computes an approximation for the solution
% of the linear system Ax=b performing n steps of the conjugate
% gradient method starting with x0, and returns in res the norm of the residual
x=x0; r=b-A*x;
res(1)=norm(r,inf);
p=zeros(size(b));
rho=1;
i=1;
while(i<=maxiter & res(i)>tol)
rhoold=rho; rho=r'*r;
beta=rho/rhoold;
p=r+beta*p;
z=A*p;
alpha=rho/(p'*z);
x=x+alpha*p;
r=r-alpha*z;
res(i+1)= norm(r,inf)
%pause
i=i+1;
end;
```

Comparer la convergence théorique et numérique avec celle des méthodes précédentes.

2) Gradient conjugué préconditionné. Modifier le code précédent en `CGprecd(C,A,b,x0,tol,maxiter)` où C est le préconditionneur.

3) Utiliser le code précédent pour le préconditionneur SOR, avec paramètre égal à 1 ou au paramètre optimal.

4) On considère le code suivant

```
nn=size(A,2);
Ch=tril(A);
for k=1:nn
    Ch(k,k)=sqrt(Ch(k,k));
    Ch(k+1:nn,k)=Ch(k+1:nn,k)/Ch(k,k);
    for j=k+1:nn
        Ch(j:nn,j)=Ch(j:nn,j)-Ch(j:nn,k)*Ch(j,k);
    end
end
```

Que fabrique t-il ? Le modifier pour fabriquer un Cholevski incomplet, que l'on utilisera avec `CGprecd(C,A,b,x0,tol,maxiter)` .

Tracer les courbes de convergence pour ces 4 méthodes comme dans la partie précédente, pour 3 valeurs de n.

4 Matrice non symétrique

Explorer le GMRES de Matlab, avec l'exemple dans `http://www.mathworks.com/help/matlab/ref/gmres.html`. Ecrire éventuellement son propre code.