Analyse Numérique, Projet 2.8, Résolution de systèmes d'équations linéaires

Alexandre Vieira & Conrad Hillairet

6 décembre 2012

Table des matières

Présentation générale

Ce projet a pour but d'implémenter différentes méthodes de résolutions de systèmes d'équations linéaires. Ces méthodes sont nombreuses et ont des philosophies différentes. On en compte principalement deux familles :

1. les méthodes directes, qui cherchent les solutions exactes en un nombre exact d'itérations. Parmi elles, on compte les différentes décompositions QR (avec, par exemple, la méthode de Householder ou de Givens), ou la décomposition de Cholesky pour les matrices symétriques définies positives.
2. les méthodes itératives, qui à partir d'une suite de vecteur cherche à converger vers la solution exacte, et ce à partir d'un vecteur de départ quelconque. Il existe là encore différentes méthodes, comme la méthode de Jacobi ou de Gauss-Seidel.

Nous avions comme projet d'implémenter une méthode directe et une méthode de relaxation. Notre choix s'est porté sur deux méthodes :

1. la méthode de Gauss.
2. la méthode de relaxation basée sur la méthode de Gauss-Seidel.

La suite développera les différents codes et théorèmes que nous avons cherché à vérifier avec ces deux méthodes. Différents résultats numériques seront présents et commentés.

Codage des méthodes

Chaque méthode a été codée en simple et double précision. Cependant, pour les coder, il faut tout d'abord commencer par comprendre comment fonctionne chaque méthode.

Méthode de Gauss :

La méthode est simple : on transforme à la fois la matrice et le vecteur de solution pour arriver à une matrice triangulaire supérieure. On utilise enfin une méthode de remontée pour résoudre le système nouvellement créé. Un exemple valant mieux que de longs discours, en voici un : on cherche le système suivant :

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 6 \\ 6 \\ 6 \end{pmatrix}$$

On réécrit ce système de manière suivante :

$$\begin{pmatrix} 1 & 2 & 3 : & 6 \\ 3 & 2 & 1 : & 6 \\ 2 & 3 & 1 : & 6 \end{pmatrix}$$

On fait ensuite des modifications linéaires entre les lignes de ce tableau, en faisant en sorte de faire apparaître un zéro dans toutes les lignes de la première colonne, sauf la première ligne. Pour cela, on prend le premier élément de chaque ligne qu'on divise par le premier élément de la première ligne, qui sera appelé pivot. Cela nous donne :

$$\begin{pmatrix} 1 & 2 & 3 : & 6 \\ 0 & -4 & -8 : & -12 \\ 0 & -1 & -5 : & -6 \end{pmatrix}$$

On répète l'opération sur la deuxième colonne, et on essaye donc de mettre un zéro sur la troisième ligne. Cela nous donne :

$$\begin{pmatrix} 1 & 2 & 3 : & 6 \\ 0 & -4 & -8 : & -12 \\ 0 & 0 & -3 : & -3 \end{pmatrix}$$

On obtient donc un système avec une matrice supérieure. La résolution devient simple : on commence par avoir la dernière composante du vecteur, et on remonte à chaque fois pour obtenir les somposantes précédentes. On a donc comme vecteur de solution :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

On a donc l'idée de l'algorithme à appliquer. Nos différents codes sont accessibles dans la pièce jointe envoyée avec ce dossier.

Méthode de relaxation :

Il s'agit là d'une méthode itérative pour résoudre un système de la forme AX=b. Le principe repose sur la décomposition de la matrice A de la façon suivante :

$$A = M - N$$

En posant une certaine itération, on peut montrer qu'on arriver à une suite de la forme :

$$MX^{k+1} = NX^k + b$$

Dans la méthode de relaxation basé sur Gauss-Seidel, on pose les matrices M et N suivantes :

$$\begin{aligned} \omega M &= D + \omega L \\ \omega N &= (1-\omega)D - \omega U \end{aligned}$$

avec

$$A = D + L + U$$

L matrice triangulaire inférieure, U matrice triangulaire supérieure, toute deux à diagonale nulle, et D=diag(A)
Ainsi, sous forme matricielle, la méthode de relaxation se présente ainsi :

$$(D + \omega L)X^{k+1} = [(1-\omega)D - \omega U]X^k + \omega b$$

Pour calculer chaque composante de la k+1-ème itération, le calcul se fera donc ainsi :

$$X_i^{k+1} = (1-\omega)X_i^k + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}X_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}X_j^k\right]$$

Et on fera ce calcul pour i allant de 1 à n, n étant la taille de la matrice.
On remarque par ailleurs que si $\omega = 1$, on retrouve la méthode de Gauss-Seidel.

Il nous manque une condition d'arrêt des itérations. On utilisera donc la suivante : En posant $r^k = b - AX^k$ le vecteur de résidu, notre condition sera :

$$\frac{||r^k||}{||b||} < \varepsilon$$

ε étant un réel définissant une tolérance, qu'on prend en général assez petit.
La norme choisi pour notre projet était la norme 2. Chaque code est présent dans le fichier joint envoyé avec le dossier.

Résultats numériques

Méthode de Gauss :

On cherche à vérifier différents théorèmes grâce à quelques exemples numériques. Nous vérifierons :

- Que le programme marche normalement avec un exemple simple (dLU1)
- La condition nécessaire et suffisante pour que la méthode marche, i.e. chaque sous-matrice est inversible (dLU2)
- Etudier deux cas limites, i.e. une triangulaire supérieure et une triangulaire inférieure (dLU3 et dLU4)
- Tester le programmes avec des matrices mal conditionnées en rajoutant une petite variation sur le vecteur de solution (dLU5.* et dLU6.*)
- Tester la différence entre simple et double précision sur un exemple bien choisi (dLU7)

Voici un tableau récapitulant les résultats.

Fichier de données	Matrice	Vecteur résultat	Solution théorique	Solution du programme
dLU1	$\begin{pmatrix}1&1&1\\1&0&3\\1&2&1\end{pmatrix}$	$\begin{pmatrix}3\\4\\4\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$	$\begin{pmatrix}1.0000000\\1.0000000\\1.0000000\end{pmatrix}$
dLU2	$\begin{pmatrix}1&2&3\\0&2&4\\0&4&8\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$	Pas de solution	$\begin{pmatrix}NaN\\Infinity\\-Infinity\end{pmatrix}$
dLU3	$\begin{pmatrix}1&1&1\\0&1&2\\0&0&3\end{pmatrix}$	$\begin{pmatrix}3\\4\\3\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$	$\begin{pmatrix}1.0000000\\1.0000000\\1.0000000\end{pmatrix}$
dLU4	$\begin{pmatrix}3&0&0\\2&2&0\\1&1&1\end{pmatrix}$	$\begin{pmatrix}3\\4\\3\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$	$\begin{pmatrix}1.0000000\\1.0000000\\1.0000000\end{pmatrix}$
dLU5.1 $cond_1(A) = 27$	$\begin{pmatrix}6&3\\3&2\end{pmatrix}$	$\begin{pmatrix}9\\5\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}1.0000000\\1.0000000\end{pmatrix}$
dLU5.2 $cond_1(A) = 27$	$\begin{pmatrix}6&3\\3&2\end{pmatrix}$	$\begin{pmatrix}8.98\\5.03\end{pmatrix}$	Sert juste à tester le conditionnement	$\begin{pmatrix}0.56666660\\1.8000002\end{pmatrix}$
dLU6.1 $cond_1(A) = 2$	$\begin{pmatrix}1&1\\1&-1\end{pmatrix}$	$\begin{pmatrix}2\\0\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}1.0000000\\1.0000000\end{pmatrix}$
dLU6.2 $cond_1(A) = 2$	$\begin{pmatrix}1&1\\1&-1\end{pmatrix}$	$\begin{pmatrix}1.98\\0.03\end{pmatrix}$	Sert juste à tester le conditionnement	$\begin{pmatrix}1.0050000\\0.97500002\end{pmatrix}$
dLU7 simple précision	$\begin{pmatrix}1&1&0\\1&3&1\\6&3&3\end{pmatrix}$	$\begin{pmatrix}3\\7\\9\end{pmatrix}$	$\begin{pmatrix}\frac{2}{3}\\\frac{7}{3}\\-\frac{2}{3}\end{pmatrix}$	$\begin{pmatrix}0.66666675\\2.3333333\\-0.66666669\end{pmatrix}$
dLU7 double précision	$\begin{pmatrix}1&1&0\\1&3&1\\6&3&3\end{pmatrix}$	$\begin{pmatrix}3\\7\\9\end{pmatrix}$	$\begin{pmatrix}\frac{2}{3}\\\frac{7}{3}\\-\frac{2}{3}\end{pmatrix}$	$\begin{pmatrix}0.66666666666666652\\2.3333333333333335\\-0.66666666666666663\end{pmatrix}$

On remarque les résultats suivants :

– Le programme semble fonctionner, même dans des cas particuliers (et c'est quand même une bonne chose !)
– Le cas où le programme ne doit pas trouver de solution donne des résultats incohérents. On vérifie par la même occasion que si une des sous-matrices n'est pas inversible (ici, la matrice entière), on n'obtient pas de solution.
– Le conditionnement de la matrice garde une grande influance sur la résolution du système. On voit qu'une légère influance sur le système modifie beaucoup les solutions.

Méthode de relaxation :

Ici, les différentes propriétés que nous chercheront à vérifier sont :
- Si A symétrique définie positive et $0 < \omega < 2$, la méthode doit converger (drelax1 et drelax2)
- Si A symétrique définie positive et $\omega > 2$, la méthode diverge (drelax3)
- Si A symétrique définie positive et $0 < \omega < 2$, mais avec un vecteur initial différent, la méthode doit converger quand même (drelax4)
- On veut aussi montrer que pour une matrice autre que définie positive, on ne peut pas assurer la convergence (drelax5 et drelax6)

Voici les différents résultats :

Fichier de données	Matrice	Second Membre	Vecteur initial	ω	Vecteur Résultat
drelax1	$\begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$	$\begin{pmatrix} 9 \\ 5 \end{pmatrix}$	$\begin{pmatrix} -37 \\ 74 \end{pmatrix}$	0.7	$\begin{pmatrix} 0.99141359 \\ 1.0138178 \end{pmatrix}$
drelax2	$\begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$	$\begin{pmatrix} 9 \\ 5 \end{pmatrix}$	$\begin{pmatrix} -37 \\ 74 \end{pmatrix}$	1.5	$\begin{pmatrix} 0.99961936 \\ 1.0014930 \end{pmatrix}$
drelax3	$\begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$	$\begin{pmatrix} 9 \\ 5 \end{pmatrix}$	$\begin{pmatrix} -37 \\ 74 \end{pmatrix}$	2.1	$\begin{pmatrix} 537540.63 \\ -702509.31 \end{pmatrix}$
drelax4	$\begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$	$\begin{pmatrix} 9 \\ 5 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 707 \end{pmatrix}$	0.7	$\begin{pmatrix} 0.99180603 \\ 1.0131862 \end{pmatrix}$
drelax5	$\begin{pmatrix} 3 & 9 \\ 1 & 2 \end{pmatrix}$	$\begin{pmatrix} 12 \\ 3 \end{pmatrix}$	$\begin{pmatrix} -45 \\ 218 \end{pmatrix}$	0.3	$\begin{pmatrix} -887871.38 \\ 377082.41 \end{pmatrix}$
drelax6	$\begin{pmatrix} 6 & 3 \\ -3 & -2 \end{pmatrix}$	$\begin{pmatrix} 9 \\ -5 \end{pmatrix}$	$\begin{pmatrix} -45 \\ 28 \end{pmatrix}$	1.3	$\begin{pmatrix} 1.0024930 \\ 0.99700826 \end{pmatrix}$

On remarque les résultats suivants :
- Encore une fois, la méthode semble fonctionner !
- Si la méthode converge, le vecteur initial n'a d'influence que sur la vitesse de convergence (comme pour d'autres méthodes itératives)
- Notre ω doit bien se trouver entre 0 et 2
- Il existe quand même des cas où la méthode semble converger quand la matrice n'est pas symétrique définie positive. Cependant, dans ces cas là, rien ne nous assure le résultat de la méthode.

Comparaison entre les deux méthodes :

Ces deux méthodes ne s'appliquent évidement pas aux même systèmes. En voici un

exemple : On prend le système suivant :

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 3 & 1 \\ 6 & 3 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 9 \end{pmatrix}$$

Avec la méthode de Gauss, nous obtenons un résultat :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \approx \begin{pmatrix} 0.66666675 \\ 2.3333333 \\ -0.66666669 \end{pmatrix}$$

Avec la méthode de relaxation, la méthode ne converge pas.

Conclusion générale

Ce projet nous aura permis de consolider nos connaissances en Analyse Numérique et de voir de près comment peuvent se mettre en place les différentes méthodes que nous avions vu précédemment. Il nous a également permis de voir comment peuvent s'appliquer ces méthodes sur des cas concrets, et quand les théorèmes sur l'existence ou la convergence d'une méthode ou de l'autre s'appliquent vraiment.

On remarque surtout que les méthodes itératives peuvent effectuer moins de calculs pour avoir une solution approchée, mais que les méthodes directes ont justement l'avantage d'arriver à la solution exacte, mais avec des calculs beaucoup plus lourds, qui peuvent poser des problèmes d'arrondis sur machines.
Cependant, d'autres méthodes (parfois plus puissantes) restent encore accessibles, et peuvent donner des résultats plus intéressants.