ANALYSE NUMÉRIQUE

T.P. N° 1

Vincent BELLUOT
Denis POPA

N° Groupe : A14

Observations :

Notes :	ANALYSE	:	**Total :**
	RÉSULTATS	:	
	PROGRAMMATION	:	
	RAPPORT	:	

EISTI, 17 avril 2013

Résolution de systèmes linéaires

17 avril 2013

Table des matières

1 Introduction

En mathématiques, un système d'équations linéaires est un ensemble d'équation linéaires qui portent sur les mêmes inconnues.
La résolution des SEL appartient aux problèmes les plus anciens dans les méthématiques et ceux-ci apparaissent dans beaucoup de domaines, comme en traitement du signal, en optimisation linéaire, ou dans l'approximation de problèmes en analyse numérique.
Dans le cadre de notre TP, on s'intéresse à la résolution d'un SEL

$$Ax = b;\ A \in \mathbb{R}^{n\ \times n},\ x,\ b \in \mathbb{R}^{n}$$

du point de vue numérique.
Le but de ce TP est d' étudier la qualité de la solution sous trois aspects :
1. Influence de la valeur de conditionnement de la matrice.
2. Influence de la dimension de la matrice.
3. Influence de la méthode de résolution utlilisée (directe ou gradient conjuguée).

Nous avons donc réalisé un programme Scilab capable de créer des matrices carrées régulières, avec la dimension et les valeurs de conditionnement paramétrables. Nous avons implémenté quatres méthodes de résolutions de SEL pour comparer l'erreur par rapport à la solution exacte et discuter de la meilleure méthode.

2 Méthodes et programme

2.1 Mise en place des expériences

Pour faire l'étude de ces différentes méthodes de la résolution d'un SEL et déterminer l'influence de la valeur de conditionnement de la matrice initiale ainsi que sa dimension, nous avons dans un premier temps créé une matrice diagonale

$$\Delta = diag(1, k^{-\frac{1}{n-1}}, k^{-\frac{1}{n-2}}, \cdots, k^{-1})$$

et la matrice tridiagonale

$$H = \begin{bmatrix} 2 & -1 & 0 & \cdots & \cdots & \cdots \\ -1 & 2 & -1 & 0 & \cdots & \cdots \\ 0 & -1 & \ddots & \ddots & 0 & \cdots \\ \cdots & 0 & \ddots & \ddots & -1 & 0 \\ \cdots & \cdots & 0 & -1 & 2 & -1 \\ \cdots & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix} \text{ de dimension } (n \times n)$$

Ensuite nous nous servons de la fonction Scilab *svd* pour faire une décomposition en valeurs singulières de cette matrice H, en faisant l'appel $[U, S, V] = svd(H)$. Les matrices U et V sont orthogonales et grâce à elles, nous pouvons construire la matrice

$$A = U^{t}\Delta V$$

Nous calculons b de telle sorte que $b = A\xi$ avec $\xi = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$

Nous appliquons les 4 méthodes de résolutions

- méthode de résolution directe, faisant partie de la bibliothèque Scilab $[x] = linsolve(A, -b)$,
- méthode du gradient conjugué sans pré-conditionnement grâce à la fonction Scilab $[x, fail, err, iter, res] = pcg(A, b, rr, nbMaxIter)$,
- méthode du gradient conjugué avec pré-conditionnement $[x, fail, err, iter, res] = pcg(A, b, rr, nbMaxIter, P)$ avec $P = diag(A)$,
- méthode du gradient conjugué avec pré-conditionnement $[x, fail, err, iter, res] = pcg(A, b, rr, nbMaxIter, P)$ avec $P = (D - L)D^{-1}(D - L^T)$ oú $D = diag(A)$ et L est la partie triangulaire inférieure de la matrice A.

Les paramètres $fail$, $iter$, res ne sont pas utiles dans notre étude. rr est le seuil pour l'erreur résiduelle que nous avons fixé à 10^{-12} et $nbMaxIter$ fixé à 10. Enfin, nous calculons la norme de l'erreur relative pour la première méthode car pour les autres, elle est déjà donnée par le paramètre err de la fonction pcg.

2.2 Programmes

Nous avons partagé notre travail en deux fichiers Scilab. Le premier fichier **etude.sce** est le programme qui concentre toutes les mesures demandées dans l'énoncé, c'est-à-dire une matrice de dimension $n = 100, 400, 800, 1200$ et pour chacune d'entre elles, les valeurs de conditionnements $k = 10^3, 10^6, 10^{10}, 10^{16}$, ainsi que la résolution par les quatre méthodes. Il y a donc en tout 64 mesures. Le deuxième fichier **test.sce** est le programme qui permet à l'utilisateur de rentrer lui même la dimension de la matrice et la valeur de conditionnement qu'il souhaite en tapant les valeurs dans la console. De cette façon nous pouvons mesurer l'erreur avec les quatres méthodes pour n'importe quelles valeurs.
Dans les deux fichiers on retrouve les même fonctions qui :

- crée la matrice diagonale Δ,
- calcule le conditionnement de la matrice entrée en paramètre. Cette fonction sert à vérifier que le conditionnement de A est égal à la valeur de k,
- construit la matrice tridiagonale H,
- crée la matrice A par une décomposition grâce à la fonction Scilab svd,
- calcule b dans $b = Ax$ avec $x = [1, 1, \cdots, 1]^T$,
- résoud le système avec la fonction Scilab $linsolve$,
- calcule l'erreur relative par la première méthode de résolution,
- calcule l'erreur relative par la deuxième méthode avec le gradient conjugué sans pré-conditionnement,
- calcule l'erreur relative par la troisième et quatrième méthode de résolution, le gradient conjugué avec pré-conditionnement.
  Il faut prendre une matrice $P = diag(A)$ pour la troisième méthode et $P = (D - L)D^{-1}(D - L^T)$ oú $D = diag(A)$ et L est la partie triangulaire inférieure de la matrice A pour la quatrième méthode.

Enfin nous avons une fonction *main* dans laquelle on crée la matrice A en utilisant les fonctions que nous avons implémentées. La fonction *main* a pour paramètres d'entrée deux entiers K et N qui sont respectivement le conditionnement de la matrice A, et sa dimension. Elle calcule ensuite la norme de l'erreur relative pour chaque méthode, en prenant soin de changer la matrice P pour la méthode trois et quatre. Cette fonction renvoie un vecteur $[E1, E2, E3, E4]$ qui sont respectivement la norme des erreurs des méthodes de résolutions, E1 pour la méthode de résolution directe, E2 méthode du gradient conjugué sans pré-conditionnement, E3 méthode du gradient conjugué avec pré-conditionnement $P = diag(A)$, et E4 la dernière méthode.

2.3 Limites

Une des limites de notre programme est que le fichier réalisant les 64 mesures prend beaucoup de temps et nécéssite la présence de l'utilisateur pour que le programme se déroule convenablement. Il faut que l'utilisateur appuie sur une touche pour continuer à afficher, ce qui fait qu'il est impossible de laisser tourner le programme pour revenir plus tard avec les résultats. L'autre limitation concerne la mémoire utilisée. Si on prend une valeur de matrice trop importante (nous avons testé avec une dimension initiale de 10000×10000) on obtient une erreur *stacksize* malgré le fait que nous l'ayons paramétré au maximum.

3 Résultats

==========

Les erreurs relatives à chaque méthode de résolution sont listées de la manière suivante :
E1, erreur de la méthode de résolution directe avec linsolve ;
E2, erreur de la méthode du gradient conjugué sans pré-conditionnement ;
E3, erreur de la méthode du gradient conjugué avec pré-conditionnement, avec P=diag(A) ;
E4, erreur de la méthode du gradient conjugué avec pré-conditionnement, avec P=(D-L)*inv(D)*(D-L^t);
==========
==========
$Matrice 100 * 100$
==========
$Conditionnement 10^3$
$E1 = 0.000000000005348882051$
$E2 = 0.0189352569212012343536$
$E3 = 0.0189029890012946434619$
$E4 = 0.0071217578538137022923$
==========
$Conditionnement 10^6$
$E1 = 8.9426938649431342298612$
$E2 = 0.0191220739994641120152$
$E3 = 0.0193042825679333243660$
$E4 = 0.0217121180863589745280$
==========

$Conditionnement 10^{10}$
$E1 = 8.9437431119301535886734$
$E2 = 0.0085892440282820739506$
$E3 = 0.0088109753010673953766$
$E4 = 0.016655787907217840 8042$
==========
$Conditionnement 10^{16}$
$E1 = 9.3451467908636569603686$
$E2 = 0.0055998548668947569634$
$E3 = 0.0057719420982575769460$
$E4 = 0.0133023800482627643782$
==========
$Matrice 400 * 400$
==========
$Conditionnement 10^{3}$
$E1 = 0.000000000022524240595$
$E2 = 0.0033445649619674362209$
$E3 = 0.0033687735337411183964$
$E4 = 0.0041449143599976709731$
==========
$Conditionnement 10^{6}$
$E1 = 15.650454462219604678808$
$E2 = 0.000842232 4836211646498$
$E3 = 0.000840109 9567675521370$
$E4 = 0.007350160 2467070763225$
==========
$Conditionnement 10^{10}$
$E1 = 15.66538329621800684777$
$E2 = 0.003406996602950005 7050$
$E3 = 0.003408985036205052 2041$
$E4 = 0.009554559150473193 5961$
==========
$Conditionnement 10^{16}$
$E1 = 15.68237883713406510822$
$E2 = 0.002659365559524523 8414$
$E3 = 0.002662044471582622 4735$
$E4 = 0.005672319996028579 3780$
==========
$Matrice 800 * 800$
==========
$Conditionnement 10^{3}$
$E1 = 24.430094467062005492153$
$E2 = 0.009560264208730712 1432$
$E3 = 0.009558966273292078 5415$
$E4 = 0.003928981529770982 0840$
==========
$Conditionnement 10^{6}$
$E1 = 24.438969704181729980519$
$E2 = 0.007883945801888727 1368$
$E3 = 0.007907581522001657 5091$

$E4 = 0.0087991170209964324200$
==========
$Conditionnement 10^{10}$
$E1 = 25.055510481364574815188$
$E2 = 0.0026938101097932227318$
$E3 = 0.0027156173843559693916$
$E4 = 0.0068286255454667174267$
==========
$Conditionnement 10^{16}$
$E1 = 25.089645239297219347918$
$E2 = 0.0036808136782039858198$
$E3 = 0.0037064728038327947497$
$E4 = 0.0095568152475177473720$
==========
$Matrice 1200 * 1200$
==========
$Conditionnement 10^{3}$
$E1 = 29.777162926961789679581$
$E2 = 0.0086971059022343047418$
$E3 = 0.0086849191380181690880$
$E4 = 0.0039823826294114140698$
==========
$Conditionnement 10^{6}$
$E1 = 29.792091435223500894836$
$E2 = 0.0068980322355066663736$
$E3 = 0.0069304393872696408788$
$E4 = 0.0077975827361760861878$
==========
$Conditionnement 10^{10}$
$E1 = 30.513865554577211725018$
$E2 = 0.0026288820060955909660$
$E3 = 0.0026328241057440439181$
$E4 = 0.0063702520560646735959$
==========
$Conditionnement 10^{16}$
$E1 = 30.562553109409702045696$
$E2 = 0.0036551483519059973885$
$E3 = 0.0036653247181908069248$
$E4 = 0.0078944751852141701459$

4 Discussion

Nous avons testé pour toutes les méthodes de résolution, différents paramètres pour voir leur influence sur la véracité des résultats comme la dimension de la matrice ainsi que la valeur de conditionnement.

Pour la résolution linéaire on remarque que lorsque la dimension de la matrice et la valeur de conditionnement sont relativement petites nous avons une erreur extrêmement faible. Cependant lorsque la dimension de la matrice atteint $798 \times$

798 l'erreur devient très grande. (Nous nous sommes servis de notre second programme pour trouver cette valeur.) Pour toute dimension de matrice si on augmente le conditionnement nous avons une erreur trop grande.

Pour la méthode de gradient conjugué sans pré-conditionnement on remarque qu'en moyenne la valeur de l'erreur est acceptable quelque soit la dimension de la matrice et la valeur du conditionnement mais on remarque aussi que pour chaque dimension de la matrice, il y a une valeur de conditionnement où l'erreur est plus faible. Par exemple pour une matrice 100×100 la meilleure valeur de conditionnement testée est 10^{16} alors que pour une matrice 1200×1200 la meilleure valeur de conditionnement testée est 10^{10}.

Pour la méthode de gradient conjugué avec pré-conditionnement avec $P = diag(A)$, on remarque que les valeurs d'erreur sont très proches des valeurs de la méthode de gradient conjugué sans pré-conditionnement, elles varient de ± 0.0001 à chaque calcul.

Pour la méthode gradient conjugué avec pré-conditionnement avec $P = (D - L)D^{-1}(D - L^T)$, on remarque que la dimension de la matrice n'influe pas sur l'erreur, alors que plus la valeur de conditionnement est faible plus l'erreur est faible.

5 Conclusion

Pour conclure ce TP, après différents calculs pour différencier les méthodes et voir quelle méthode est la meilleure, nous observons que lorsque nous sommes face à une matrice de petite taille avec une petite valeur de conditionnement la résolution linéaire est incontestablement la meilleure alors que si nous voulons une résolution qui donne un résultat acceptable à tous les coups nous choisissons la méthode de gradient conjugué sans pré-conditionnement ou la méthode de gradient conjugué avec pré-conditionnement où $P = diag(A)$ car ce sont celles qui en moyenne ont une erreur acceptable. La méthode gradient conjugué avec pré-conditionnement où $P = (D - L)D^{-1}(D - L^T)$ est intéressante lorsque la dimension de la matrice est très importante et que la valeur du conditionnement est faible.