**ennCM CS**

Copyright © 2000-2017 MATHICSE-EPFL Lausanne **1**

ÉCOLE POLYTECHNIQUE
  
FÉDÉRALE DE LAUSANNE

**Exercices Matlab - Analyse Numérique - 2017
  
Section MA
  
Prof. A. Quarteroni
  
Séance 1 - Introduction à Matlab**

**Exercice 1**

On considère les matrices

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 5 | 3 | 0 |  | 4 | 3 | 2 |
| A= | 1 | 1 | —4 | , *B=* | *0* | *1* | 0 |
|  | 3 | 0 | 0 |  | 5 | 0 | **1/2** |

Créer un répertoire de travail et écrire un fichier ".m" dans lequel placer les instructions pour calculer (sans utiliser de boucles), la matrice *C* = *AB* (produit matriciel) et la matrice *D* qui a comme éléments Dii = AiiBii (produit composante par composante).

**Solution 1**

Dans le script "exl.m", nous avons :

|  |
| --- |
| cic  clear all  close all  A= [5 3 0; 1 1 -4; 3 0 0];  B= [4 3 2; 0 1 0; 5 0 1/2];  C = A\*B  D = A.\*B |

On obtient :

» exl

C =

|  |  |  |
| --- | --- | --- |
| 20 | 18 | 10 |
| -16 | 4 | 0 |
| 12 | 9 | 6 |
| D= |  |  |
| 20 | 9 | 0 |
| 0 | 1 | 0 |
| 15 | 0 | 0 |

**Exercice 2**

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 2

Définir (sans utiliser de boucles) la matrice diagonale de taille *ri* = 5 dont la diagonale est un vecteur de points équirépartis entre 3 et 6 (i.e. [3, 3.75, 4.5, 5.25, 6]).

**Solution 2**

**Exercice 3**

» M = diag(linspace(3,6,5))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3.0000 | 0 | 0 | 0 | 0 |
| 0 | 3.7500 | 0 | 0 | 0 |
| 0 | 0 | 4.5000 | 0 | 0 |
| 0 | 0 | 0 | 5.2500 | 0 |
| 0 | 0 | 0 | 0 | 6.0000 |

Écrire une fonction pour calculer :

1. le produit, composante par composante, entre deux vecteurs *x* et y ;
2. le produit scalaire entre les mêmes vecteurs *x* et y ;
3. un vecteur dont les éléments sont définis par :

*V1* = X1 Yn, V2 = X2 Yn-1, • • • , Vn-1 = Xn-1 Y2, Vn = *Xn* yi

Utiliser et compléter la définition suivante :

|  |
| --- |
| **function** [ElByElProd, ScalProd, v] = operations(x,y)   * *(ELBYELPROD] = OPERATIONS(X,Y) is the element by element product of vectors X and Y. NOTE: X and Y can be row or column vectors.* * *[ELBYELPROD, SCALPROD] = OPERATIONS(X,Y) returns also the scalar product of*   *vectors X and Y.*   * *[ELBYELPROD, SCALPROD, V] = OPERATIONS(X,Y) returns also the vector V*   *which is defined as: V(1) = X(1)Y(n)*  *V(2) = X(2)Y(n-1) V(N) = X(N)Y(1)*  **return** |

Tester la fonction avec Matlab. **Solution 3**

**function** [ElByElProd, ScalProd, v] = operations(x,y)

|  |
| --- |
| * *(ELBYELPROD] = OPERATIONS(X,Y) is the element by element product of vectors X and Y. NOTE: X and Y can be row or column vectors.* * *[ELBYELPROD, SCALPROD] = OPERATIONS(X,Y) returns also the scalar product of vectors X and Y.* * *[ELBYELPROD, SCALPROD, V] = OPERATIONS(X,Y) returns also the vector V*   *which is defined as: V(1) = X(1)Y(n)*  *V(2) = X(2)Y(n-1)*  *V (N) = X (N) Y (1)*  size\_x = size(x);  size\_y = size(y);  **if** (size\_x(1) -= size\_y(1) II size\_x(2) -= size\_y(2)) disp('!!! ERROR: vectors size is different !!!') **return**  **end**  **if** (min(size\_x) > 1)  disp('!!! ERROR: X and Y are matrices !!!') **return**  **end**  ElByElProd = x.\*y;  **if** (size\_x(2) >= size\_x(1))  ScalProd = x\*y';  **else**  ScalProd = x'\*y;  **end**  n = length(x);  v = x.\*y(end:-1:1);  *%% On peut aussi utiliser une boucle for:*   * *v = [ ];*   *-f-5 for i = 1:n*   * *v = [v x(i)\*y(n-i+1)];* * *end*   **return** |

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| » | x= | [1 | 4 | 7 | 2 | 1 | 2]; |
| » | y= | [0 | 9 | 1 | 4 | 3 | 0]; |

» [ElByElProd, ScalProd, v] = operations(x,y) ElByElProd =

0 36 7 8 3 0

ScalProd =

54

=

0 12 28 2 9 0

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 4

Exercice 4

En utilisant la commande diag, définir en Matlab la matrice A E Rn" avec n = 10

- 2 —1

—1 2 —1

—1 2 —1

—1 2

Ensuite, calculer les quantités suivantes :

1. le déterminant de A;
2. les normes 11A 11 1, 11A112, 11/41 Io° (tapez **help** norm pour voir les options) ;
3. le rayon spectral de A, *p(A).* On rappelle que *p(A) =* IXi(A)1, où Ài(A) sont
     
   les valeurs propres de A. Vérifier que, puisque A est symétrique et définie positive, on a

*P(A) = IIAII2;*

Visualiser les vecteurs propres vi, *j* E {1, ... ,10} en utilisant les commandes [v, **lambda] =eig (A)** et **plot (v) .**

En utilisant Matlab, vérifier que la matrice V (dont les colonnes sont égales aux vecteurs propres de A) permet de diagonaliser la matrice A. En particulier, vérifier que

*17-1AV = D = diag(À1, • • •* An).

Visualiser finalement la structure des matrices **A, V, D** (avec la commande spy).

**Solution** 4

On peut définir la matrice A avec la commande suivante :

**n=10;**

**A=2\*diag(ones(1,n))-diag(ones(1,n-1),1)-diag(ones(1,n-1),-1); A**

Après, on calcule son déterminant par **det (A)**

**ans =**

**11**

et les normes Mill, I1A112, 11,411. par

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 5

nrm1=norm(A,1) nrm2=norm(A,2) nrminf=norm(A,inf)

et on obtient

nrml =

4

nrm2 =

3.9190

nrminf =

4

Pour évaluer le rayon spectral de A, il faut calculer ses valeurs propres et en prendre le maximum, en valeur absolue :

rho=max(abs(eig(A)))

rho =

3.9190

On peut directement vérifier que, A étant symétrique et définie positive, on a *p(A) = I* 142. Pour calculer et visualiser les vecteurs propres vj, *j* E 11, ... ,10}, il faut utiliser les commandes

[V, D]=eig(A); figure

plot (V)

ou, si on veut visualiser chaque vecteur propre individuellement, la boucle suivante :

figure

for i=1:10

plot(V(:,i)) pause

end

Avec la première option, on a le graphe de la Figure 1.

Pour vérifier que la matrice V (dont les colonnes sont égales aux vecteurs propres de A) permet de diagonaliser la matrice A, c'est-à-dire V-1AV = *D =* diag(Ai, , An), il faut utiliser les commandes suivantes :

[V,D]=eig(A)

inv(V)\*A\*V % ou mieux (V\A)\*V

D

dif = norm(D-inv(V)\*A\*V); % ou mieux dif = norm(D-(V\A)\*V);

on obtient

dif =

3.5003e-15

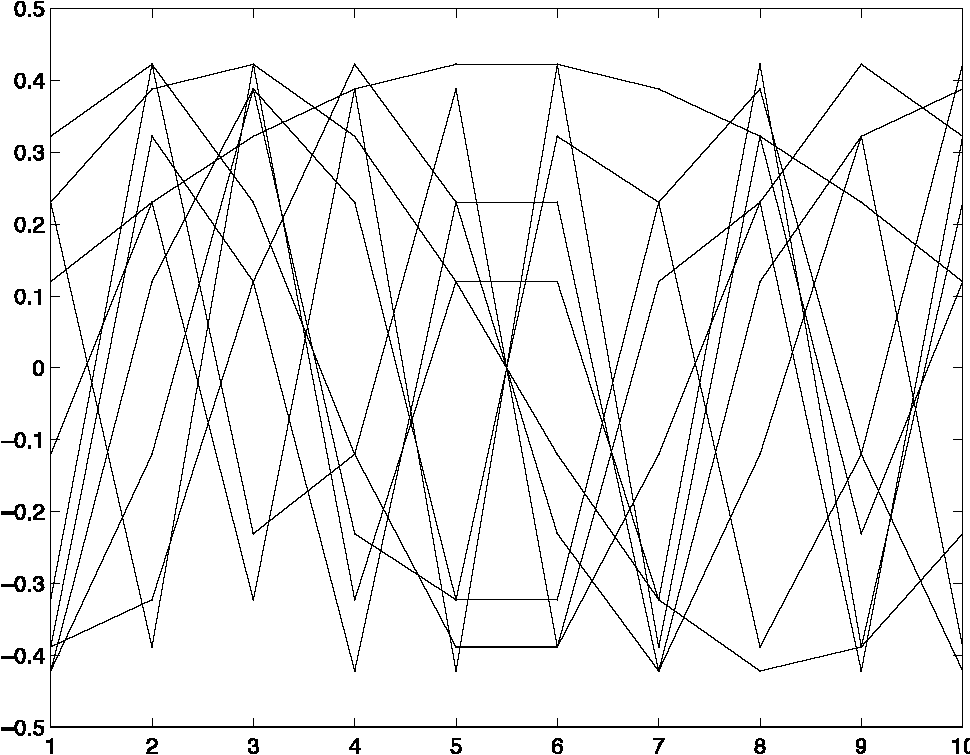


FIGURE 1 - Composantes des vecteurs propres de la matrice A.

Enfin, pour visualiser la structure des matrices A, V, D, il faut taper

spy (A)
  
spy (D)
  
spy (V)

On peut observer que la matrice A est tri-diagonale, la matrice *D* est diagonale alors que la matrice V des vecteurs propres est pleine : tous ses éléments sont différentes de zéro.

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 |  |  |

10

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 6

FIGURE 2 — Structure des matrices A, *D* et V.

Exercice 5 a) Soit

T2

*f (x) =* sin(x), *x* E [1,20]

une fonction qu'on veut représenter graphiquement en choisissant 10 points, 20 points et 100 points dans l'intervalle de définition. Écrire un fichier ".m" pour réaliser les trois graphiques sur la même figure et avec trois couleurs différentes. Quelle est la meilleure représentation ?

b) Faire la même chose pour les fonctions :

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 7

*g(x) =* — 1 2 cos(sin(x)) exp(—x) *+ (i ±j- x) , x* E [1, 20]

*h(x) = x(1— x)+*

sin(x) cos(x) , *x* E [1, 20]

*x3*

**Solution 5**

a) Le script "ex4.m" est :

FIGURE 3 — Graphe de la fonction f.

f=@ (x) x.^2/2.\*sin(x); xl = linspace(1,20,10);

x2 = linspace(1,20,20);

x3 = linspace(1,20,100);

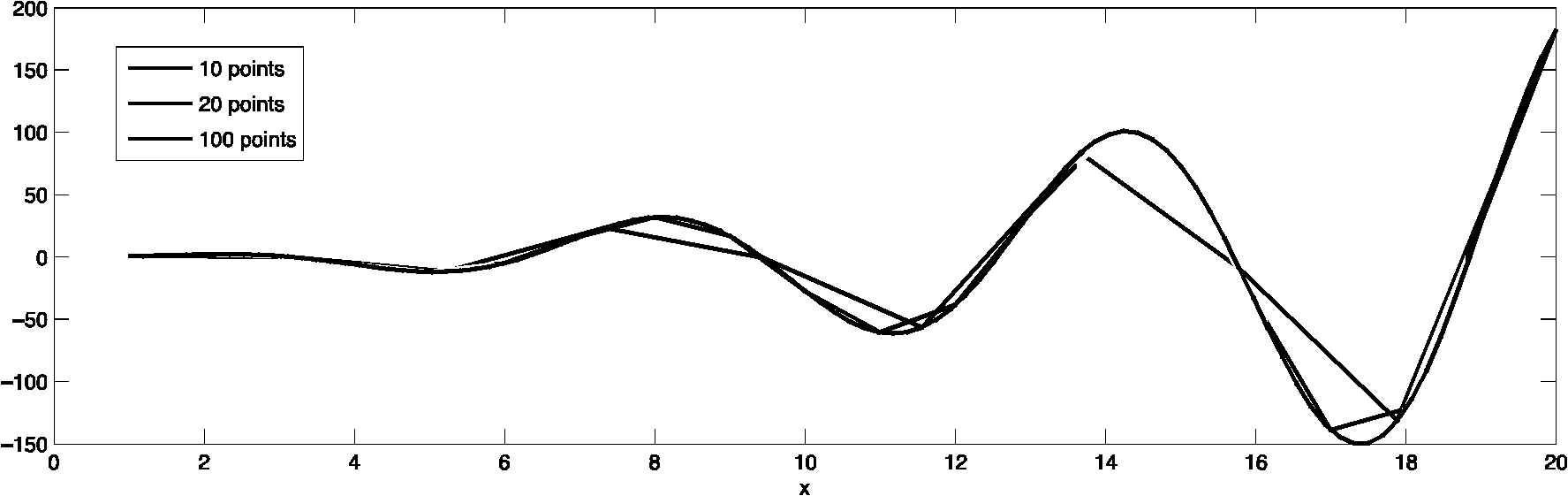
figure

plot(xl,f(x1),'r') hold on

plot(x2,f(x2),'g') plot(x3,f(x3),'m') legend('10 points','20 points','100 points')

xlabel('x')

ylabel('f(x)')



2

8

**4**

**6**

**18**

**20**

* 10 points
* **20** points
* 100 points

b) On replace la fonction *f* par les fonctions *g* et *h :*

|  |
| --- |
| g=@(x) x.^3/6.\*cos(sin(x)).\*exp(-x)+(1./(1+x)).'2; h=@ (x) x.\*(1-x)+sin(x).\*cos(x)./(x."3); |

**ennCM CS**

ÉCOLE POLYTECHNIQUE
  
FÉDÉRALE DE LAUSANNE

**Exercices Théoriques - Analyse Numérique - 2017
  
Section MA
  
Prof. A. Quarteroni
  
Corrigé Séance 2 Normes matricielles et conditionnement des systèmes linéaires**

|  |  |
| --- | --- |
| **Exercice 1**  Soit il • Il une norme vectorielle. Prouver que la fonction  liAxii  MALI = sup  xo 11x11  est une norme matricielle, en remarquant que la relation (1) est équivalente 1 à  IIAII = sup11.11=1 liAxii• | (1) |

**Solution 1**

En utilisant l'astuce, on montre directement que cette définition forme une norme matricielle.

1. Si liAxii > 0, alors Mil = supI xIH1 liAxii > O. De plus

IIAII liAxii

= sup = 0 <=> liAxii = 0 Vx **0,**

**3co** 11x11

Ax = 0 Vx **0** si et seulement si A = 0. Donc Mil = 0 si et seulement si A = 0.

1. Soit un scalaire a,

liaAll = sup liaAxii = lai sup liAxii = lai MAIL 11x11-1 11x11-1

1. Vérifions enfin l'inégalité triangulaire. Par définition du supremum, si x **0** alors

liAxii

< 11A11, IlAx11 < 11All

11x11 —

ainsi, en prenant x de norme 1, on obtient

II(A+ B)xli liAxii + liBxli l'Ail +

d'où on déduit MA + Bil = sup11x11=„1 ii(A + B)xll < l'Ail +

1. En effet, on peut définir pour tout x 0 un vecteur unitaire u x/IIxII de sorte que (1) s'écrive

IlAll = sup IIAuII = IlAwil avec >II = 1. 11.11=1

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 1

**Exercice 2**

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 2

Soit III • III une norme matricielle subordonnée à une norme vectorielle II • II. Prouver que

1. IlAxII < IIIAIII >II, i.e• III • III est une norme compatible (ou consistante) avec II • II ;
2. III/III = 1;
3. 111AB111 111A111 111B111, i•e• 111'111 est sous-multiplicative.

**Solution 2**

1. Par définition du supremum, si **x 0** alors

**11111111 = sup MAYM > 11/43(11**

**y0 bril — Ilx11 11AXII 111;1111 113(11.**

1. Par la définition de la norme matricielle subordonnée à une norme vectorielle

III/III = sup 111-x11 =1 ic#) 11x11

1. Par la compatibilité de la norme, on déduit

IIIABIII = sup IIABx1I < sup IIIAIII IIBx1I = IIIAIII sup IIBx1I = IIIAIII IIIBIII• 11x11=1 114=1 11x11=1

**Exercice 3**

Montrer que, si II • II est une norme matricielle consistante avec une norme vectorielle II • II, alors *p(A)* IIAII, VA E C"n.

**Solution 3**

Si À est une valeur propre de A, alors il existe **y 0,** vecteur propre de A, tel que Av = Àv. Ainsi, puisque 11 • Il est consistante,

IÀI 11v11 = IlAvIl = IlAvIl IIAII 11v11

et donc IÀI < IIAII. Cette inégalité étant vraie pour toute valeur propre de A, elle l'est en particulier quand AlI est égale au rayon spectral.

**Exercice 4**

Étant donnée la matrice A E R2'2, a11 = a22 = **1,** a12 = 'y, a21 = 0, vérifier que pour *7 >* **0,** *K 00( A) = K1(A) = (1 +* -y)2. Soit Ax = **b** le système linéaire où **b** est tel que x = **(1 -** -y, 1)T soit la solution. Trouver une majoration du type

|  |  |
| --- | --- |
|  | 1161311„, 11b11. ' |

avec *C* > **0** une constante qui ne dépend que de 11A-1110„, 111311,, et No°. Le vecteur (x + Sx) est la solution du système perturbé A(x + Sx) = **(b + 6b)** avec **Sb** une perturbation du vecteur **b.** Le problème est-il bien conditionné par rapport à -y -+ oo ?

**Solution 4** On a

|  |  |  |  |
| --- | --- | --- | --- |
| Ainsi, puisque -y > 0, | A = [1 -y1  0 1 | **A-1** | = [1 —'y  [o 1 j. |

|  |  |  |
| --- | --- | --- |
|  | 2  = max  2  = max E I aij I  i=1,2  j=1 | = max{ 1, 1 + -y} = 1 +  = max{ 1 + -y, 1} = 1 + |

1124-1111 = max{ 1, 1 +-y} = 1 +7,
  
MA-111,„ = max11 + 7, 11 =1 + y.

Copyright © 2000-2017 MATHICSE-EPFL Lausanne **3**

Par conséquent,

Ki(A) = 11Alli IIA-1111 = Ko()(A) = 11,411.11A-111. = **(1 +1')2.**

On a

b = Ax = [11 ,

en particulier 11131100 = 1. En perturbant le second membre du système Ax = b (on ne perturbe pas la matrice), on obtient un système perturbé de la forme

|  |  |
| --- | --- |
| donc  On en tire que  En divisant par 11x11„,, on trouve | A(x + Sx) = **b + Sb,  Sx = A-16b.  116x1100 MA-11100116b1100.**  118x11.. < 11A 111-116b110.. 11,(110° —IIxHoc |

En plus, puisque IlblIcx, = 1, on peut écrire

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 4

116x1100 < IIA-11100 116b1100

11x1100 - 11x1100 111:1100 '

*C*

avec *C* = 11/4-111,)/11x11„‹, la constante cherchée. On a donc que

1 + -y

*C*=

max{1, Il - -yll.

On voit bien que *C* -+ 1 quand -y -+ oo. Ainsi, pour le cas particulier de **b = (1, 1)T,** le problème est bien conditionné. Remarquons que, dans le cas général **(b** arbitraire), le problème est mal conditionné pour -y grand. En effet, *Koo(A) -+* oo quand -y -+ oo. Cet exercice met en évidence que le fait d'avoir une matrice avec un grand conditionnement n'empêche pas nécessairement le système global d'être bien conditionné pour des choix particuliers du second membre **b.**

**Exercice 5**

Supposons que PAH < -ylIAII, 11614 < -ylIbIl avec -y E 111+ et 6A E Rn <n, **613** E Rn. On veut montrer que, si -yK(A) < 1, on a les inégalités suivantes :

(2)

Ilx + Sx1I < 1 + -yK(A) lIxII - 1 - -yK(A)

1163(11 < 2-y

K(A). (3)

lIxII 1 - -yK(A)

1. Si *C* est une matrice carrée telle que *p(C) <* 1, on sait (voir le Théorème 1.5 du livre) que *I - C* est inversible. Montrer que dans ce cas on a

1*C 11(I C)-111 < 1\_1C*

*1+ 11C II - - 1 - 11C II«*

où II • II est une norme matricielle subordonnée à une norme vectorielle telle que IICII < 1.

1. Montrer l'inégalité (2) en utilisant le résultat du point 1 et le fait que (A+8/4)(x+6x) = **b+613.**
2. Montrer l'inégalité (3) (Sugg. : utiliser l'inégalité triangulaire Ilx + 6xII < IlxII + II6xII). **Solution 5**

a) Puisque2 II/II = 1, on a (Exercice 2, Série 2)

1= II/II < g-Cille- C)-1II < (1+ IICIDII(/ - C)-11I,

ce qui donne la première inégalité de (4). Pour la seconde, en remarquant que *I = I - C + C* et en multipliant à droite les deux membres par *(I -* C)-1, on a *(I - C)-I- = I + C (I - C)-1.* En prenant les normes, on obtient

II(/- C)-1II < 1+ IICIIII(/ - C)-1II,

d'où on déduit la seconde inégalité, puisque IICII < 1.

2. On suppose (toujours) que Il • II est une norme matricielle subordonnée à une norme vectorielle.

1. Soit (A + 5A)(x + 6x) = b + Sb.

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 5

Alors, on a

(I + A-1 8A)(x + 6x) = x + A-1 Sb.

De plus, puisque 7K(A) < 1 et M6AM < -yMAil on a

MA-16AM < MA-1MMSAM 7MA-1MMAM =7K(A) < 1.

Alors, p(A-16A) < 1 et I + A-1SA est inversible. En prenant l'inverse de cette matrice et en passant aux normes, on obtient

Mx + êxil < A-16A)-111 + 7MA-111 MbM).

Alors l'inégalité du point 1 donne

1

x 8)(11 1 \_A-16A (11x1I + 711 A-111 111311)

ce qui implique

IIx+ SxM < 1 + 7K(A)

Mxil — 1 — *7K(A)'*

puisque MA-16AM <7K(A) et MbM < MAil MxM•

1. Montrons que

|  |  |
| --- | --- |
| 1164  xi 1 — KA*K* (A)  ill 7 (  )  En retranchant Ax = b de (2), on a | (5) |

Abc = —SA(x + 6x) +

En prenant l'inverse de A et en passant aux normes, on obtient l'inégalité suivante M6xil < MA-1SAM IIx + SxII + MA-4 MSbil

< 7K(A)Mx+SxM +7MA-1M Mbil.

En divisant les deux membres par ilxM et en utilisant l'inégalité triangulaire Mx + 6xM 116x11 + 11x11, on obtient finalement (5).

Exercice 6

a) Soit A E R71 matrice symétrique définie positive et soient Ai et yi, i = 1, , n, les

valeurs propres et les vecteurs propres de A. Montrer que si x est la solution du système linéaire Ax = b, alors

X = E(CiPti)Vi, i=1

où *ci* est la i-ème composante de b dans la base des vecteurs propres de A.

b) On se donne maintenant le système linéaire Ax = b suivant

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 6

1001 1000 xi [bil

1000 1001 [x2 j [b2 '

où A est mal-conditionnée avec les valeurs propres À1 = 1, À2 = 2001.

En décomposant le second membre sur la base des vecteurs propres de la matrice A, expliquer pourquoi, lorsque b = (2001, 2001)T, une petite perturbation Sb = (1, 0)T produit de grandes variations dans la solution, et réciproquement, si b = (1, —1)T, une petite variation Sx = (0.001, 0)T dans la solution induit de grandes variations dans b.

**Solution 6**

1. Puisque A est une matrice symétrique, il existe une matrice V orthogonale et une matrice *D* diagonale dont tous les coefficients sont réels, telles que

*17-1 AV = D = diag(k, .. • , 4)*

ou, de façon équivalente, Avi = Àivi pour i = 1, , n, de sorte que les vecteurs colonnes de V

soient les vecteurs propres de A. De plus, les vecteurs propres sont deux à deux orthogonaux (et peuvent être normalisés : donc, on a que vTvi = Sit, où Shi est le symbole de Kronecker) et on déduit que les vecteurs propres d'une matrice symétrique sont orthogonaux et engendrent l'espace M n tout entier.

Donc, soient b = Ein CiVi le membre de droite du système linéaire Ax = b et x sa solution ; en écrivant aussi x dans la base des vecteurs propres de A, on a :

Ax = A Exivi=Ecivi.

i=i i=i

et, puisque Avi = Àivi, on trouve :

ExiAivi—Ecivi.

i=i i=i

Donc on trouve :

E(iXi — Ci)vi = O.

i=1

c'est-à-dire xiAi = ci et

X =

i=1

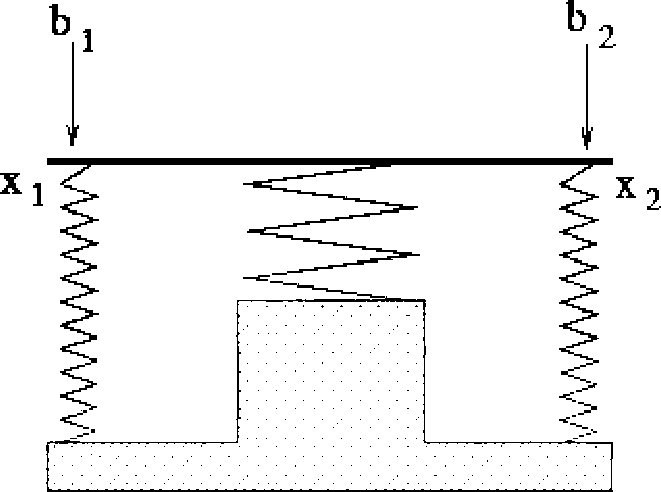
1. Les vecteurs propres de la matrice A sont vi = (1, —1)T (qui correspond à À1 = 1) et **v2 =** (1, 1)T (qui correspond à la valeur propre À2 = 2001). Soit b = (2001, 2001)T et Sb = (1, 0)T. Alors,

[20011 [11 1 , , 1 4003

b + Sb = 2001 + 0

2001v2 + —2(vi v2) = 2v1 + 2 v2.

[ [



Si on écrit la solution x comme combinaison linéaire des vecteurs propres on trouve

Copyright © 2000-2017 MATHICSE-EPFL Lausanne 7

1 4003 1 4003 1.5

+

X = 1V1 -r 2001 v2 = 2 vi + 4002 v2 '--' [0.5] •

Ainsi on voit que l'erreur Sx par rapport à la solution exacte x = (1, 1) est Sx r.\_.- (0.5, —0.5)T. Réciproquement, soit b = (1, —1)T. La solution exacte du système est x = (1, —1)T. On exprime la solution perturbée par rapport aux vecteurs propres :

[1.0011 2.001 0.001

-1

x+Sx= 2 vi + 2 v2,

j

d'où *ci* = 2.001/2 et e2 = 2.001/2. Donc

b + 613 = [2.0011

[ 0

et Sb = (1.001,1)T.

*Remarque* Le système linéaire de cet exercice pourrait être obtenu de l'analyse d'une barre rigide attachée dans sa partie central à un ressort de raideur 4000 et connectée aux extremités à deux ressorts de raideur 1 (voir la figure ci-dessous). On applique des forces b1 et b2 aux extremités de la barre et on observe ses déplacements verticaux xi et x2.

Si les forces bi et b2 sont équilibrées (par exemple bi = 2001, b2 = 2001), alors de petits changements Sb engendrent des mouvements significatifs de la barre (grand Sx).

A l'inverse, si les forces ne sont pas équilibrées (par exemple bi = 1, b2 = —1), alors on peut obtenir de petits déplacements Sx même si on impose de forts changements Sb sur les forces exercées.

**enn CM CS**

ECOLE POLYTECHNIQUE
  
FÉDÉRALE DE LAUSANNE

**Exercices Théoriques & Matlab - Analyse Numérique - 2017
  
Section MA
  
Prof. A. Quarteroni
  
Corrigé Séance 3 - Systèmes linéaires : méthodes directes**

**Exercice 1 (théorique)**

Considérons le cas particulier d'un système linéaire dont la matrice est tri-diagonale et inversible :

/ai ci

0

cri—i an j

*A=* b2 a2

0 *br,*

a) Montrer qu'il existe deux matrices bi-diagonales *L* et *U* de la forme

|  |  |  |  |
| --- | --- | --- | --- |
| *1* /32  *L (  0* | 0) 1  en 1 | *U*= | /ai y1 0  ct2  • • l'n-1  ° an |

telles que A = *LU* et donner les expressions des coefficients ai, /3 et en fonctions des

Copyright © 2000-2017 CMCS-EPFL Lausanne 1

coefficients de A. Ces formules sont connues sous l'appellation *d'Algorithme de Thomas.*

b) Obtenir les formules résultantes de l'extension de l'algorithme de Thomas à la résolution du système Ax = **f,** avec **f = (fe1** E Rn, donnée par

1. trouver **y** tel que **Ly = f,**
2. trouver x tel que Ux = **y.**

c) Combien d'opérations virgule flottante requiert l'algorithme précédent ?

**Solution 1**

a) On appelle Li la i-ième ligne de *L* et Ui la j-ième colonne de *U,* il faut vérifier que

Li Uj = [A]ii.

Pour la matrice A, on a des éléments pas nulles seulement pour i = *j* ou i = *j* 1. On obtient les équations suivantes :

* si i = *j* on a*,* + ai = ai ;
* si i = *j* — 1 on a = ;

si i = *j* + 1 on a eiai = bi.

Copyright © 2000-2017 CMCS-EPFL Lausanne 2

Donc, ai, /3i et s'obtiennent facilement avec les relations suivantes :

al = al, bi = = ai — *ci.* (1)

1. La résolution du système Ax = **f** revient à résoudre deux systèmes bidiagonaux, *Ly =* **f** et Ux = *y,* pour lesquels on a les formules suivantes :

*(Ly =* **f) : Yi =** fi, Yi = *fi - OiYi-i, i -* 2, ... , n,

Yn Yi — "Yixi-Ei (2)

(Ux = *X y) :*

\_\_ n — , Xi = i = n — 1, . . . , 1.

an ai

1. L'algorithme requiert 8n — 7 flops : 3(n — 1) flops pour la factorisation (??) et 5n — 4 flops pour la substitution (??).

**Exercice 2 (Matlab)**

On considère un câble élastique qui occupe au repos le segment [0, 1], fixé aux extrémités, sur lequel on applique une force d'intensité *f (x).* Son déplacement au point *x, u(x),* est la solution de l'équation différentielle suivante :

*{*

*—u" (x) = f (x), x* E (0,1), u(0) = 0, u(1) = 0. Soit *N EN, h=* 1/N et xi = *ih* pour i = 0, *N ;* pour approcher la solution *u(x),* on considère la

discrétisation de l'intervalle (0, 1) en *N* sous-intervalles (xi, xi+i), et on construit une approximation ui de u(xi) par la méthode des différences finies. Cette méthode requiert de résoudre numériquement le système linéaire tridiagonal **Au = b** qui suit :

(3)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *1* | 2 —1  **—1** 2 —1 |  | Ul  U2 |  | *f(xi)-f (x2)* |  |
| *h2* |  |  |  |  |  | (4) |
|  | **1** 2 —1 |  | *UN-2* |  | *f (x N-2)* |  |
|  | **—1** 2 |  | *UN-1 \_* |  | *f (x N-1) -* |  |

où **u =** [ui, u2, uNAT et **b = *[f*** *(xi), f (x2), f (xN\_1)1T .* Plus *N* est grand, plus l'approximation sera précise et plus la taille du système linéaire à résoudre sera élevée.

a) On suppose que la force appliquée soit *f (x) = x(1 — x)* et on prend *N =* 20 intervalles.
  
Construire la matrice A et le vecteur **b** correspondants, à l'aide des commandes suivantes :

f = @(x) x.\*(1-x);

N=20; h = 1/N;

x=linspace(h, 1-h, N-1)'; on transpose pour avoir un vecteur colonne

b= f(x);

Calculer la factorisation *LU* de A avec la commande Matlab lu. Vérifier que la matrice de permutation *P* est l'identité (on sait de la théorie qu'aucune permutation de lignes n'est effectuée dès que la matrice est symétrique définie positive, ce qui est le cas de A).

Copyright © 2000-2017 CMCS-EPFL Lausanne 3

Calculer également la factorisation de Cholesky de A (commande chol) et remarquer qu'elle diffère de la précédente.

b) Mettre en oeuvre l'algorithme de *substitution directe* pour la résolution d'un système trian­gulaire inférieure et l'algorithme de *substitution rétrograde* pour la résolution d'un système triangulaire supérieure. Utiliser et compléter les fonctions suivantes :

|  |
| --- |
| **function** [x] = subs\_directe(L,b)  96 *1XJ* = *SUBS DIRECrE(L,B) resout le systeme triangulaire inferieure L\*X = B -2,-*  **return** |

**function** [x] = subs\_retrograde(U,b)

*96 SUBS RETROGRADE(U,b, \_,sout le systeme triangulaire superieure U\*X = B*

**return**

Calculer la solution du système linéaire Au = b à partir de la factorisation A = *LU,* en utilisant les fonctions subs\_direct et subs\_retrograde pour résoudre les deux systèmes triangulaires.

1. A l'aide de la commande plot, représenter le déplacement **u** du câble aux noeuds xi définis au point 1.
2. Etudier le comportement du conditionnement de la matrice *A, K(A),* lorsque *N* augmente, en traçant le graphe des valeurs de *K(A)* pour *N =* 10,20, ... , 120. Tracer le graphe biloga­rithmique des mêmes valeurs avec la commande loglog. Quel type de courbe obtient-on ? Si on suppose une relation linéaire 1og10 *K(A)* = m 1og10 *N c* pour le graphe bilogarithmique, alors on a *K(A)* = *C Nrri* (avec *C* = 10c) : calculer les constantes m et *C.* De combien *K(A)* croît-il lorsque on double le nombre *N* des sous-intervalles ?

**Solution 2**

a) On définit la matrice A et le vecteur **b** avec les commandes suggérées dans l'énoncé :

f= @(x) x.\*(1-x) ;

N=20; h = 1/N;

x=linspace (h, 1-h,N-1) ' ;

b=f (x) ;

A=(N-2)\* (diag(2\*ones (N-1, 1) , -diag(ones (N-2 , 1) , -diag (ones (N-2,1) , -1)) ;

Ensuite, on calcule la factorisation *LU* de A. En général, Matlab peut décider d'effectuer des permutations de lignes pendant le processus de factorisation, ce qui mène à une factorisation *PA = LU.* Ainsi, la syntaxe de la commande lu à utiliser est la suivante :

[L,U,P] = lu(A);

De cette façon, on a stocké dans la variable **P** la matrice de permutation *P.* Dans notre cas, on peut vérifier que **P** est la matrice identité : par exemple, on peut calculer l'écart maximal entre les éléments de *P* et de *I,* et on obtient :

Copyright © 2000-2017 CMCS-EPFL Lausanne 4

max (max **(abs(P-eye(N-1))))**

ans =

0

Donc, Matlab calcule la factorisation *LU* sans permutation. Les facteurs ont été stockés dans les variables **L** et U. Ces facteurs diffèrent du facteur *H* de la factorisation de Cholesky A = *HHT,* que l'on calcule avec la commande

**H = chol(A)' ;** car

**max(max(abs(L - H)))** ans =

**27.2843**

|  |
| --- |
| b),  **function** [x] = subs\_directe(L,b)  *SUBS DIRECTE(L,B) resout le systeme triangulaire inferieure L\*X = B*  x = zeros(size(b,1), 1); [m,n]=size(L);  **if** m n  disp('Error: the matrix is not square!'); b = []; **return**  **end**  **if** m length(b)  disp(['Error: the dimension of the matrix and of the vector' ' are not consistent!']);  x = [];  **return**  **end**  1 = min(diag(abs(L)));  **if** 1 == 0  disp('Error: the matrix is singular'); x = [];  **return**  **end**  **for** j=1:n  x(j)-(b(i)—L(j,1:j-1)\*x(1:j-1))/L(j,j);  **end**  **return** |

**function** [x] = subs\_retrograde(U,b)

D'après le cours, on exploite la factorisation LU de la façon suivante :

*\* [X] = SUBS RETROGRADE (U,B) resout le systeme triangulaire superieure U\*X = B* **x =** zeros (size **(b, 1) , 1) ;**

**[m,** n] =size **(U) ;**

**if m n**

disp ( 'Error : the matrix is not square! ' ) ; = **;**

**return**

**end**

**if** m length(b)

disp(['Error: the dimension of the matrix and of the vector' ' are not consistent!']);

= []**;**

**return**

**end**

*1 =* min (diag (abs (U) ) ) ;

|  |  |
| --- | --- |
| **if 1 == 0** di sp  = **;**  **return end** | che matrix is singular'); |

**for j = n: —1:1**

**x(j)=(b(j)—U(j,** j+1:n)\*x (j+1:n) ) /U(j, **j); end;**

**return**

Copyright © 2000-2017 CMCS-EPFL Lausanne 5

y = subs\_directe(L,b);

u = subs\_retrograde(U,y);

c) La figure ?? montre les déplacements ui aux noeuds xi (cercles rouges). Si *f (x)* est un polynôme, il est facile de trouver la solution exacte du problème différentiel : dans notre cas 1, on a *u(x)* = x4/12 — x3/6 + x/12. On a ajouté le graphe de *u(x)* afin de montrer qu'avec *N =* 20 sous-intervalles, on obtient déjà une solution approchée assez précise. La figure ?? a été obtenue par les commandes :

u\_exact = 'x.-4 / 12 - x.-3 / 6 + x / 12'; fplot(u\_exact, [0,1]);

hold on;

plot(x, u, 'or');

Il faut remarquer que la taille du vecteur u est *N — 1;* en effet seules les valeurs des déplace­ments aux noeuds xi pour i = 1, , *N —* 1 sont inconnues, car uo = *uN* = 0 (conditions aux bords).

1. Comme *u''(x)* = *—x +* x2, on intègre deux fois et on trouve *u(x)* = x4/12 — x3/6 + *ax + b,* où a et *b* sont deux constantes que l'on trouve en imposant les conditions aux bords :

*u(0) = b =* 0, *u(1) = 0 > a=* 1/12.

Copyright © 2000-2017 CMCS-EPFL Lausanne 6

|  |  |
| --- | --- |
| 0.03 0.025 0.02 0.015 0.01 0.005 |  |
|  |

FIGURE 1 - Solution exacte *u(x)* et déplacements approchés ui

d) Pour chaque valeur de *N,* on a une matrice A différente. Donc, il faut coder une boucle qui, pour *N = 10,20,30, ... ,120,* construit cette matrice et calcule son conditionnement. Cette valeur sera ensuite mémorisée dans un vecteur k. La boucle peut se coder comme suit :

for N = 10:10:120;

h = 1/N; A=(N-2)\*(diag(2\*ones(N-1,1),0)-diag(ones(N-2,1),1)-diag(ones(N-2,1),-1)); k(N/10) = cond(A);

disp(sprintf('N = %i: K(A) = %e',N,k(N/10))); % ceci pour afficher

% les valeurs calculees

end

et on obtient :

|  |  |  |  |
| --- | --- | --- | --- |
| N = 10: | K(A) | = | 3.986346e+01 |
| N = 20: | K(A) | = | 1.614476e+02 |

|  |  |  |  |
| --- | --- | --- | --- |
| N = 110: | K(A) | = | 4.903279e+03 |
| N = 120: | K(A) | = | 5.835434e+03 |

Le graphe de *K(A)* en fonction de *N* (commande plot( [10:10:120] ,k)) et le graphe bi-lo­garithmique (commande loglog([10:10:120],k); grid on) sont affichés en figure ??. On voit que le graphe bi-logarithmique est bien celui d'une droite, donc du type

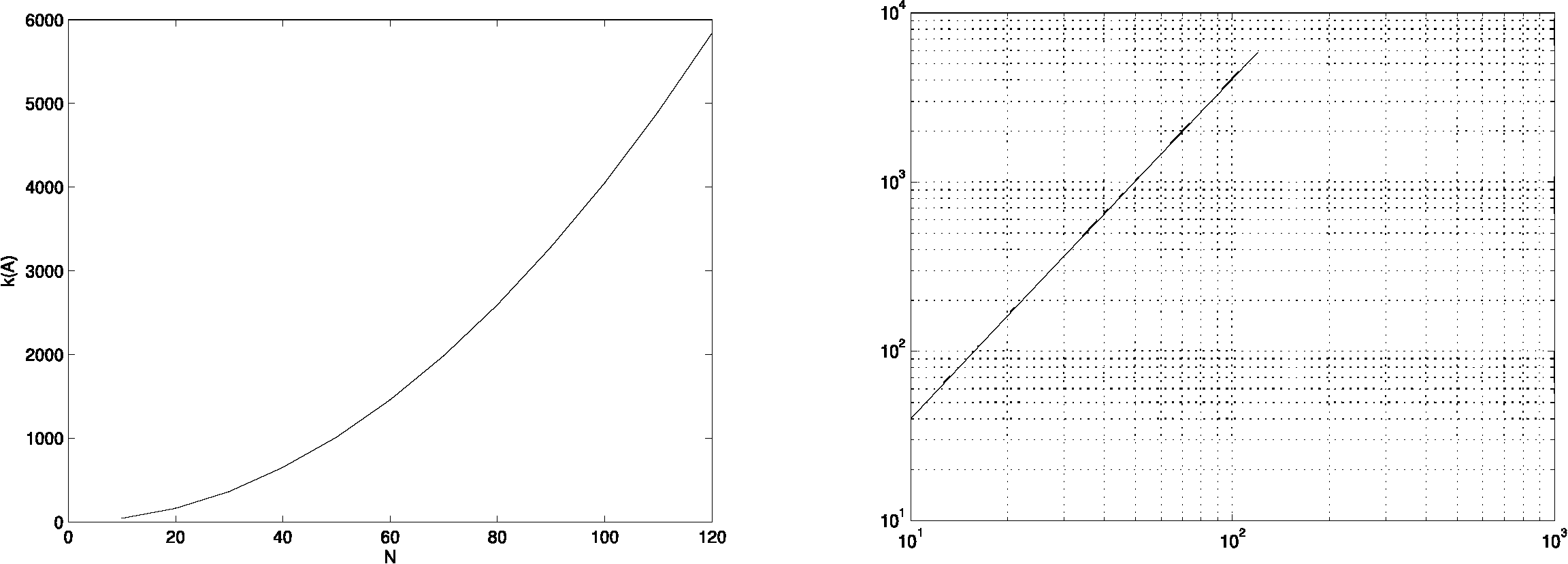
logio *K =* m logio *N + c.*

On calcule m et c directement sur le graphe, en mesurant la pente entre les abscisses 1 *(N =* 10) et 2 *(N =* 100), ou bien en utilisant Matlab :

m = ( log10(k(10)) - log10(k(1)) ) / (log10(120) - log10(10)) m =

2.0066

FIGURE 2 — Nombre de conditionnement *K(A)* en fonction de *N,* graphes linéaire (à gauche) et bi-logarithmique (à droite)



le

le

102

101

101

le

102

Copyright © 2000-2017 CMCS-EPFL Lausanne 7

c = log10(k(1)) - m\*log10(10)

c =

-0.4060

C = 10-c

C=

0.3926

La pente est presque égale à 2, donc la formule à proposer sera bien

*K(A)* = *CN2,*

avec *C* ^-' 0.3926. Donc, *K(A)* croît quadratiquement avec *N,* ce qui signifie que la solution du système linéaire par la méthode de factorisation *LU* devient de plus en plus sensible aux perturbations sur les données et aux erreurs d'arrondi.

**Exercice 3 (théorique)**

Etudier l'existence et l'unicité de la factorisation *LU* des matrices suivantes :

1 2 0 1 () (0 1)

A= ( *C* =

1 2) ' B= 1 0) ' V) 2 *j* '

**Solution 3**

On rappelle que si *M E rixn,*

la factorisation *LU* de *M* avec lii = 1 pour i = **1, ... ,** n existe et est unique si et seulement si les sous-matrices principales Mi de *M* d'ordre i = 1, ... , n — 1 sont inversibles (voir Théorème 3.4 à la page 77 du livre). Dans ce cas :

*= ,*

*(1 0) (un\_* U12) ( Ull U12

*l*21 1 0 U22 121U11 121U12 + U22

*M*

La matrice singulière A, dont la sous-matrice principale Al = 1 est inversible, admet une unique factorisation *LU.* La matrice inversible *B* dont la sous-matrice B1 est singulière n'admet pas de factorisation, tandis que la matrice (singulière) *C,* dont la sous-matrice Ci est singulière, admet une infinité de factorisations de la forme *C* = *LeUe,* avec /el = 1, l8 *= e, l2 = 1, =* 0, uÇ:2 = **1** et u22 = — 2 — 0, VO e R.

Copyright © 2000-2017 CMCS-EPFL Lausanne 8

**Exercice 4 (théorique)**

Soit A la matrice donnée par

(1 2 3 4

= 2 5 **1 10** A

3 1 35 5

4 10 5 45

avec *det(A) >* 0. Effectuer la factorisation de Cholesky de la matrice A, après avoir remarqué qu'une telle factorisation existe.

**Solution 4**

La matrice est symétrique et définie positive. En effet, la matrice est symétrique et tous les mineurs principaux dominants de A sont positifs (critère de Sylvester). On a vu au cours que les coefficients hii de *HT* (triangulaire inférieure), avec A = *HT H,* peuvent être calculés comme suit : h11 = all et, pour i = 2, ... ,n,

|  |  |
| --- | --- |
| , 1 .. *-* j-1  a, E *hikhik*  nii = , *j* = 1, *...* , *i* 1,  nii  k=1  i-1 )1/2  *hii = aii —* E *qk*  (  k=1 | (5) |

On obtient ainsi

*(1* 0 0 0 *HT—* 2 1 0 0

3 —5 1 0 •

4 2 3 4

**enn CM CS**

ECOLE POLYTECHNIQUE
  
FÉDÉRALE DE LAUSANNE

**Exercices Théoriques & Matlab - Analyse Numérique - 2017
  
Section MA
  
Prof. A. Quarteroni
  
Corrigé Séance 4 - Systèmes linéaires : méthodes directes et itératives**

**Exercice 1 (théorique)**

Etudier l'existence et l'unicité de la factorisation *LU* de Gauss des matrices suivantes :

= (1 2 (0 1 (0 1

A )

*B = C=* 2) 1 0) 0 2.

Répéter l'exercice dans le cas où il est possible d'utiliser une matrice de pivots.

**Solution 1**

On rappelle que si *M* E Rn", la factorisation *LU* de *M* avec /ii = 1 pour i = **1, ,** n existe et est unique si et seulement si les sous-matrices principales Mi de *M* d'ordre i = 1, , n — 1 sont inversibles (voir Théorème 3.4 à la page 77 du livre). Dans ce cas :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *M =* (**1**  *121* | **0)** (Uil  *1 0* | U12) U22 | ( Ull  /21U11 | U12  /21U12 + U22 |

La matrice singulière A, dont la sous-matrice principale Al = 1 est inversible, admet une unique factorisation *LU.* La matrice inversible *B* dont la sous-matrice B1 est singulière n'admet pas de factorisation, tandis que la matrice (singulière) *C,* dont la sous-matrice Cl est singulière, admet une infinité de factorisations de la forme *C* = *L04* avec *= 1, 11:1 = 0, 111 = 1, u1131 =* 0, u2 = **1** et

2 — *e,vo* E R.

22— —

On se donne la possibilité d'utiliser une matrice de permutation pour les matrices *B* et *C* (A n'en a pas besoin). Pour *B,* si on utilise une matrice de permutation

**(0** 1

1 **0) '**

on a que *PB = I.* Par conséquent, la factorisation de *PB* est triviale. Pour *C,* il n'existe pas de matrice *P* telle que le mineur principal d'ordre 1 de *PC* est non nul. En revanche, il existe une matrice

*Q* = (o

0)

qui permute les colonnes telle que

*Q (21 00)*

Le mineur principal d'ordre 1 de *CQ* est non nul, donc il existe une unique factorisation *LU* de *CQ.*

Copyright © 2000-2017 CMCS-EPFL Lausanne 1

**Exercice 2 (théorique)**

Copyright © 2000-2017 CMCS-EPFL Lausanne 2

Soit A la matrice donnée par

1 2 3 4

= 2 5 **1 10** A

3 1 35 5

4 10 5 45

avec *det(A) >* **O.** Effectuer la factorisation de Cholesky de la matrice A, après avoir remarqué qu'une telle factorisation existe.

**Solution 2**

La matrice est symétrique et définie positive. En effet, la matrice est symétrique et tous les mineurs principaux dominants de A sont positifs (critère de Sylvester). On a vu au cours que les coefficients hii de *HT* (triangulaire inférieure), avec A = *HT H,* peuvent être calculés comme suit : h11 = all et, pour i = 2, ... ,n,

1 i—i

|  |
| --- |
| *1,* — **1,**  *(1)* |

hii = h*77 ii* (aii — Ehikhik i

.),

*k=1*

*i-1 )112*

*!Lü =* aii — Eqk

(

*k=1*

On obtient ainsi

***HT***

*(***1 0 0 0** *2* 1 0 0 3 —5 1 0 4 2 3 4

**Exercice 3 (Matlab)**

1. En prenant n = 1000, écrire en Matlab la matrice n x n suivante :

( 1 1T

A =

**1** *—I*

où **1** est un vecteur colonne unitaire de longueur n — 1 et *I* est la matrice identité (n — 1) x (n — 1). Ensuite calculer la factorisation LU de la matrice A avec Matlab (en utilisant la commande **lu** avec *L, U* et *P* comme sorties ; écrire **help lu** pour apprendre comme utiliser la commande).

1. Visualiser la structure des facteurs *L* et *U* et de la matrice de permutation *P* (en utilisant la commande **spy).**
2. Soient *P* et C2 deux matrices de permutation. On considère maintenant la matrice À = *PAC2* obtenue en effectuant une permutation des lignes avec *P* et une permutation des colonnes avec Q et on note par A = *LU* la factorisation LU de la matrice À. Trouver une permutation des lignes P et des colonnes *Q* telles que les facteurs *L* et U soient creux.
3. Calculer une factorisation À = *LU* de la matrice À à l'aide de la commande lu et visualiser la structure des facteurs *L* et Ù.
4. Transformer au format creux (en utilisant la commande sparse) les matrices *L, U* du point (a), et les matrices *L,* U du point (d). En utilisant la commande whos comparer la taille en mémoire de ces matrices. Que peut-on remarquer ?

**Solution 3**

1. Pour construire la matrice

(1 1T

A =

**1** *—I*

on utilise en Matlab les commandes suivantes :

n=1000; a11=1; a12=ones(1, n-1);

a21=ones(n-1, 1);

a22=-eye(n-1,n-1);

A=[a11,a12;a21,a22];

Ensuite, pour calculer la factorisation LU, on utilise la commande :

[L,U,P]=1u(A);

1. Pour visualiser la structure des facteurs *L* et *U,* on écrit

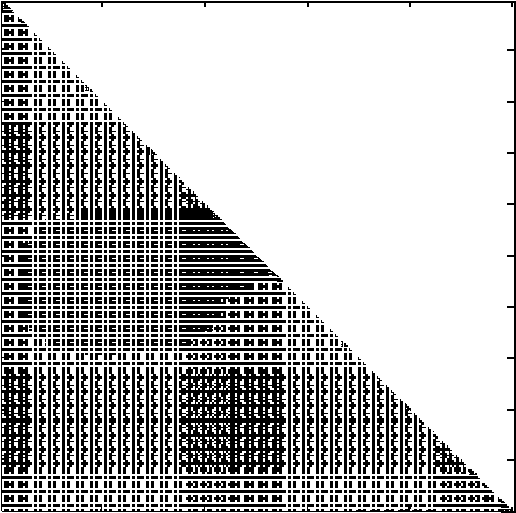
**f** igure (1)

spy (L)

f igure (2)

spy (U)

On voit que les facteurs sont pleins :



•

1111k, \_

1000

800

200

1000

0

1000

0

111111111111111111■111111111111111111111■11

•

|  |  |  |
| --- | --- | --- |
| 1111 |  |  |
|  | |

1000

800

200

400 600

ru = WWW

400 600

ru = 500500

FIGURE 1 - Structures des matrices L et U telles que A=LU.

Copyright © 2000-2017 CMCS-EPFL Lausanne 3

c) On peut permuter la première ligne de A avec la dernière, ainsi que la première colonne de A avec la dernière, en utilisant la matrice de permutation *P* donnée par

/ 0 0 • • • 0 1 \
  
0 1 ... 0 0

P=

0 0 - - - 1 0
  
\ 1 0 ••• 0 0 f

et en exécutant le produit À = *PAP :*

P=zeros (n,n) ; P (1 , n)=1;

P (n, 1)=1;

P(2:n-1, 2:n-1)=eye(n-2,n-2);

tildeA=P\*A\*P;

d) La factorisation LU de À, telle que À = *if I ,* ainsi que les structures des matrices *L* et Û, sont données par :

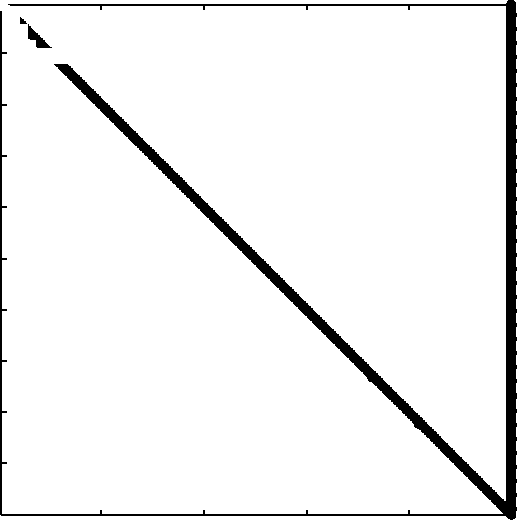
[tildeL , tildeU] =lu (tildeA) ;

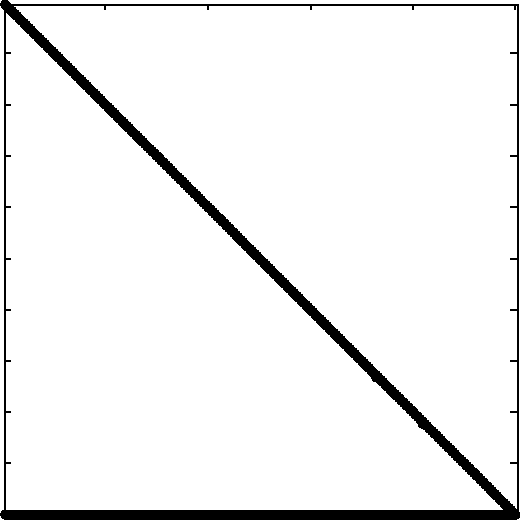
f igure (3)

spy (tildeL) figure(4)

spy(tildeU)

On trouve les structures pésentées dans la Figure 2.





1000

1000

200

200

800

800

1000

0

400 600

nz = 1999

1000

0

400 600

nz = 1999

FIGURE 2 — Structures des matrices *L* et Û telles que À = *Lù.*

Copyright © 2000-2017 CMCS-EPFL Lausanne 4

e) On peut noter que le nombre d'éléments non nuls passe de 500500 à 1999, donc il est 250 fois plus petit ! Cette différence devient importante surtout quand on utilise un format de représentation des matrices creuses qui ne garde que les entrées non-nulles. Cela est fait en Matlab à l'aide de la commande sparse. Normalement, si on travaille avec des matrices creuses et qu'on les converties au format *creux,* on gagne en memoire et en performance. Pour cet exemple, on peut voir la différence en tapant

sparseU=sparse (U) ;

sparseL=sparse (L) ;

sparse\_tildeU=sparse(tildeU); sparse\_tildeL=sparse(tildeL); whos

Copyright © 2000-2017 CMCS-EPFL Lausanne 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Size | Bytes | Class | Attributes |
| (...) |  |  |  |  |
| L | 1000x1000 | 8000000 | double |  |
| U | 1000x1000 | 8000000 | double |  |
| sparseL | 1000x1000 | 6010004 | double | sparse |
| sparseU | 1000x1000 | 6010004 | double | sparse |
| (...) |  |  |  |  |
| tildeL | 1000x1000 | 8000000 | double |  |
| tildeU | 1000x1000 | 8000000 | double |  |
| sparse\_tildeL | 1000x1000 | 27992 | double | sparse |
| sparse\_tildeU | 1000x1000 | 27992 | double | sparse |

On voit que l'utilisation de la mémoire pour des matrices pleines est de 8 MB (en fait, comme les entrées sont des double, donc 8 Bytes, 8\* 1000 \* 1000 = 8000000 Bytes = 8 MB). Pour la première factorisation *(LU),* l'utilisation est même plus grande avec le format sparse, tandis que si on utilise la deuxième factorisation *(LU),* chaque facteur occupe moins de 40 KB, c'est-à-dire plus de 200 fois moins que dans le premier cas.

Exercice 4 (Matlab)

En considérant la matrice de Hilbert A E Rn"

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1 1 1 1  1 î i  2 3 4  1  3 |  |  |
|  |  |  |
| A= | Aii = |  |
|  |  |  |  |
|  |  |  |  |

1. Calculer avec Matlab les éléments de la matrice A avec n = 10 (en utilisant la commande hilb) et le nombre de conditionnement *K2(A)* avec la commande cond. Enregistrer les nombres de conditionnement calculés pour n = 1, ... , 10 dans un vecteur avec 10 composantes.
2. Visualiser sur un graphe en échelle semilogarithmique (linéaire sur les abscisses et logarith­mique sur les ordonnées) la valeur de *K2* (A) en fonction de n, pour 1 < n < 10 en utilisant la commande semilogy. En déduire que K2(A) se comporte comme d'''.
3. Construire un vecteur colonne xex aléatoire avec n composantes en utilisant la commande rand et calculer le vecteur b = Axex. Pour chaque n = 1, 2, ... , 10 résoudre le système linéaire Ax = b avec la commande \ qui met en oeuvre une méthode directe et calculer l'erreur relative

*e*

*r = Ilx — )(exil*

*>exil*

Visualiser l'erreur sur une graphe en échelle semilogarithmique et en déduire que cette erreur
  
se conduit comme K2(A). Visualiser ensemble l'erreur relative *Er* et son estimation èr =

K2(

A)11'11 en échelle semilogarithmique, où r est le résidu. 11b11

**Solution 4**

Copyright © 2000-2017 CMCS-EPFL Lausanne 6

1. On peut calculer les quantités nécessaires avec les commandes suivantes, dans une boucle for pour n= 1,2, . . . , 10 :

help hilb

HILB Hilbert matrix.

HILB(N) is the N by N matrix with elements 1/(i+j-1), which is a f amous example of a badly conditioned matrix.

for n=1:10

A=hilb(n);

KA(n)=cond(A);

end

1. Le conditionnement de la matrice A est enregistré dans le vecteur KA

KA = 1.0e+13 \*

Columns 1 through 4

0.00000000000010 0.00000000000193 0.00000000005241 0.00000000155137

Columns 5 through 8

0.00000004766073 0.00000149510586 0.00004753673566 0.00152575754584

Columns 9 through 10

0.04931544439891 1.60252863765249

et son comportement en fonction de n peut être visualisé avec les commandes suivantes :

figure(1);

semilogy([1:1:10],KA,'r'); grid;

xlabel('n');

ylabel('K(A)');

On peut observer que le conditionnement a une comportement linéaire sur le graphique, c'est-à-dire K2(A) croit (presque) exponentiellement en fonction de n.

1. On veut construire un vecteur colonne xex aléatoire avec n composantes, calculer le vecteur **b =** Axex, résoudre (pour chaque n = 1,2, ... , 10) le système linéaire Ax = **b** et calculer l'erreur relative

Er 11X - Xed

IIXed

on peut utiliser le code suivant :

for n=1:10

A=hilb(n);

KA(n)=cond(A); xex=rand(n,1); b=A\*xex;

x=A\b;

err(n)=norm(x-xex)/norm(xex);

end

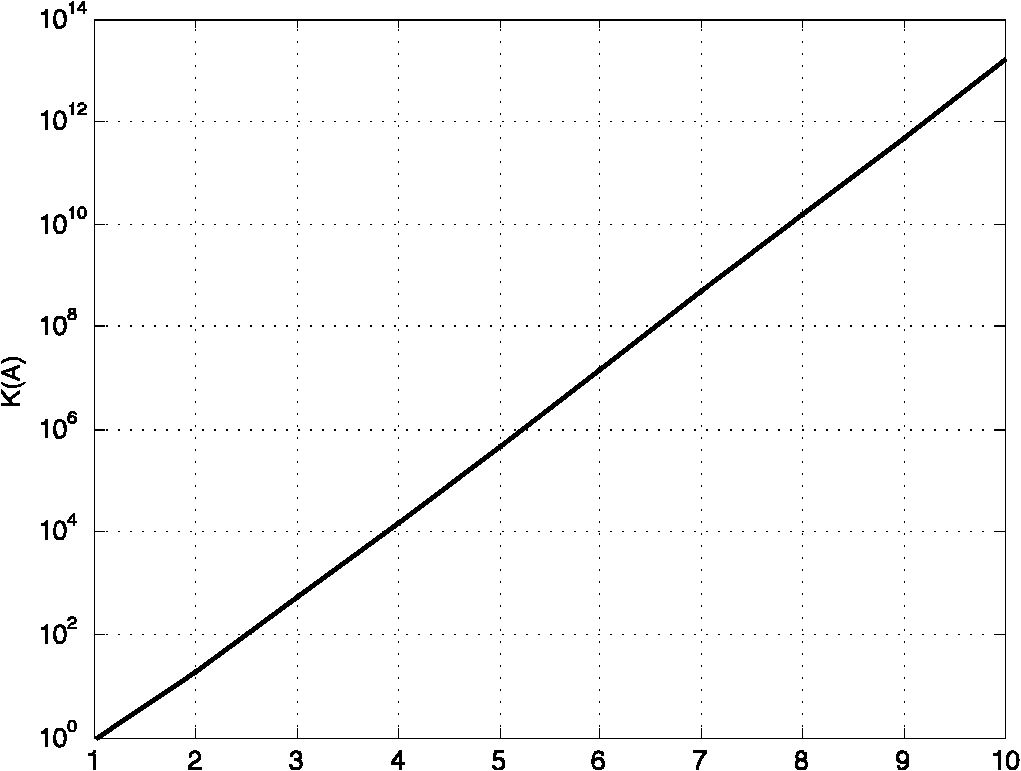


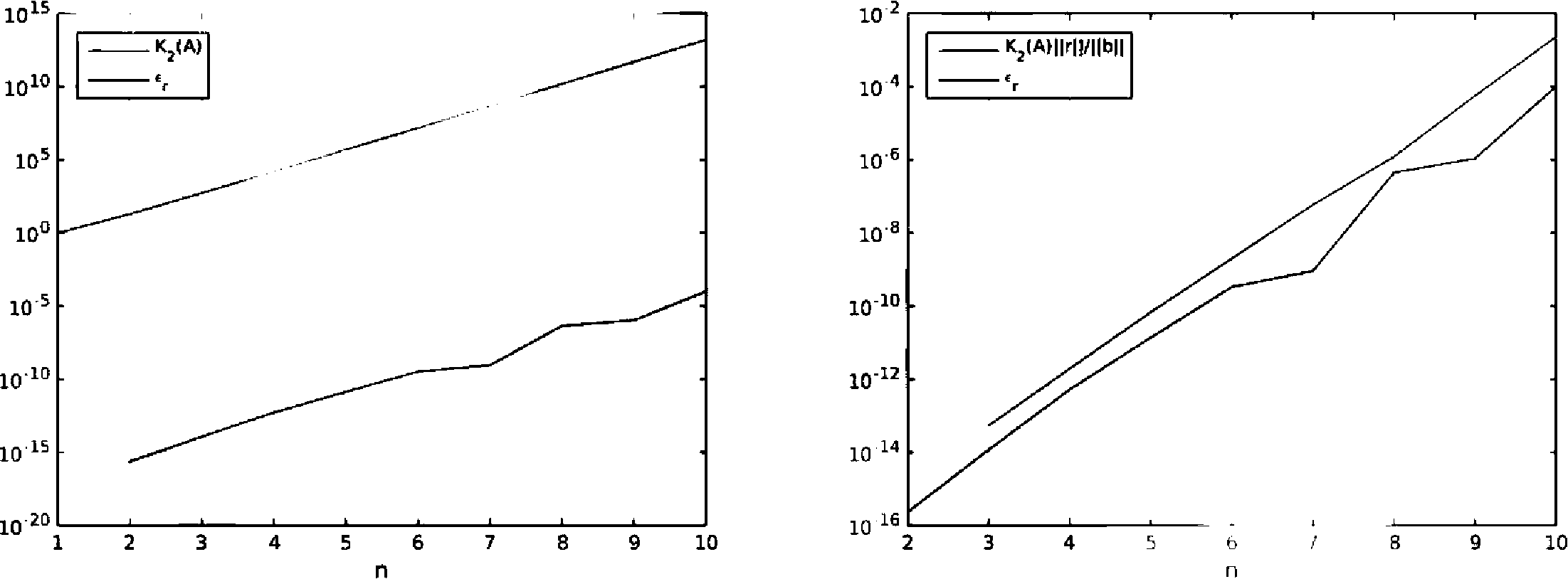
FIGURE 3 — Graphique du conditionnement de la matrice de Hilbert en fonction de n ; l'échelle est linéaire sur les abscisses et logarithmique sur les ordonnées.

Ensuite, pour visualiser le comportement du conditionnement et de l'erreur relative *Er* sur le même graphique, on peut taper :

figure(2);

semilogy( [1:1:10] ,KA, 'r' , [1:1:10] ,err, 'h') ;grid; xlabel('n');

ylabel('KA(n) & err(n)');



100

5

10

10 10

10'15

7 8

9

10

10"

2 3 4 5 6

n

1010

K7iA)

(a) (b)

Copyright © 2000-2017 CMCS-EPFL Lausanne 7

FIGURE 4 — Graphique du conditionnement de la matrice de Hilbert et de l'erreur relative *Er =*IIen

11x-11 II

fonction de n à la gauche 4(a), graphique du *Er* = **K2 (À)** Will et de l'erreur relative *er* en function de n à la droite 4(b); l'échelle est linéaire sur les abscisses et logarithmique sur les ordonnées.

Le graphique montre le conditionnement (ligne rouge) et l'erreur (ligne bleu) en fonction de n dans une échelle semi-logarithmique. On peut observer que l'erreur relative se comporte de la même façon que le conditionnement, même si elle est plus petite. Cela confirme le fait que, plus le conditionnement de la matrice est grand, plus les erreurs d'arrondi sont amplifiées dans la résolution du système linéaire, conduisant à un erreur relative toujours plus grande,

en accord avec l'inégalité suivante :

Copyright © 2000-2017 CMCS-EPFL Lausanne 8

11(5x11 <  *K (A) ( 1151)111152411*

*1130 — 1— K(A)116A11111All 111311 ± 11All ) •*