TP 5 corrige

B. Landreau

Systemes lineaires : methodes iteratives

Exercice 1 : matrices d'iteration sur un exemple

On commence par creer la matrice. Vu la regularite de A, on a interet a creer une petite procedure qui donne la valeur d el'element (i,j).

```
> a:=proc(i,j) if i=j then 2 else 1 fi end;
```

$$a := \mathbf{proc}(i, j)\ \mathbf{if}\ i = j\ \mathbf{then}\ 2\ \mathbf{else}\ 1\ \mathbf{end\ if\ end\ proc}$$

```
> A:=matrix(5,5,(i,j)->a(i,j));
```

$$A := \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix}$$

On peut meme se dispenser de definir la procedure a .

```
> A:=matrix(5,5,(i,j)->if i=j then 2 else 1
  fi);
```

$$A := \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix}$$

```
> B:=([1,2,3,4,5]);
```

$$B := [1, 2, 3, 4, 5]$$

a) Solution exacte

```
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> X:=linsolve(A,B);
```

$$X := \left[\frac{-3}{2}, \frac{-1}{2}, \frac{1}{2}, \frac{3}{2}, \frac{5}{2}\right]$$

```
>
```

b) Matrices d'iteration

Jacobi

On commence par extraire la diagonale.

```
> M:=diag(seq(A[i,i],i=1..5));
```

$$M := \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

On obtient N par difference. Il faut utiliser **evalm** pour forcer le calcul de la difference.

```
> N:=evalm(M-A);
```

$$N := \begin{bmatrix} 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & -1 & 0 \end{bmatrix}$$

Et finalement P=M^(-1)N

```
> P_J:=evalm(inverse(M)&*N);
```

$$P\_J := \begin{bmatrix} 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 \end{bmatrix}$$

Remarque : pour Jacobi, on peut directement donner P.

```
> P_J:=matrix(5,5,(i,j)->if i<>j then
  -A[i,j]/A[i,i] else 0 fi);
```

$$P\_J := \begin{bmatrix} 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 \end{bmatrix}$$

Gauss-Seidel

On commence par extraire M et on deduit N puis P=M^(-1)N.

```
> M:=matrix(5,5,(i,j)->if i>=j then
  A[i,j] else 0 fi);
```

$$M := \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 1 & 2 & 0 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix}$$

```
> N:=evalm(M-A);
```

$$N := \begin{bmatrix} 0 & -1 & -1 & -1 & -1 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> P_GS:=evalm(inverse(M)&*N);
```

$$P\_GS := \begin{bmatrix} 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ 0 & \frac{1}{4} & \frac{-1}{4} & \frac{-1}{4} & \frac{-1}{4} \\ 0 & \frac{1}{8} & \frac{3}{8} & \frac{-1}{8} & \frac{-1}{8} \\ 0 & \frac{1}{16} & \frac{3}{16} & \frac{7}{16} & \frac{-1}{16} \\ 0 & \frac{1}{32} & \frac{3}{32} & \frac{7}{32} & \frac{15}{32} \end{bmatrix}$$

c) Polynome caracteristique et rayon spectral

On utilise la fonction toute prete **charpoly**.

```
> chi:=charpoly(P_J,X);
```

$$\chi := X^5 - \frac{5}{2}X^3 + \frac{5}{2}X^2 - \frac{15}{16}X + \frac{1}{8}$$

puis la fonction solve pour trouver les racines.

```
> solve(chi);
```

$$-2, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}$$

Il est clair que le rayon spectral est 2. La methode de Jacobi ne va donc pas converger.
Remarque : on aurait pu utiliser directement aussi une fonction qui donne toutes les valeurs propres.

```
> eigenvals(P_J);
```

$$-2, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}$$

Pour automatiser la recherche du rayon spectral, on pourrait faire :

```
> S:=fsolve(chi);L:=seq(abs(S[i]),i=1..5
  );rho:=max(L);
```

S := -2., .5000000000, .5000000000, .5000000000, .5000000000

L := 2., .5000000000, .5000000000, .5000000000, .5000000000

$$\rho := 2.$$

```
>
```

Et pour Gauss-Seidel ?

```
> phi:=charpoly(P_GS,X);solve(phi);
```

$$\phi := X\left(X^4 - \frac{49}{32}X^3 + \frac{31}{32}X^2 - \frac{9}{32}X + \frac{1}{32}\right)$$

$$0, \text{RootOf}(32\_Z^4 - 49\_Z^3 + 31\_Z^2 - 9\_Z + 1)$$

Pour avoir une evaluation des racines, on rajoute evalf.

```
> S:=evalf(solve(phi));max(seq(abs(S[i])
```

```
   ,i=1..5));
```

$$S := 0., .3142777810 + .08321563260\,I,$$
$$.4513472190 + .3032271640\,I,$$
$$.4513472190 - .3032271640\,I,$$
$$.3142777810 - .08321563260\,I$$
$$.5437472070$$

Cette fois le rayon spectral est plus petit que 1 donc la methode de Gauss-Seidel convergera.

```
>
```

Exercice 2 : Jacobi et Gaus-Seidel

a) Methode de Jacobi

On utilise 2 vecteurs X et Y, Y est l'itere de X par Jacobi puis X est remplace par Y.

```
> jacobi:=proc(A,B,X0,epsilon)
  local i,k,n,X,Y,test;
  # option trace;
  n:=rowdim(A);
  X:=copy(X0);
  Y:=vector(n);
  test:=false; # pour rentrer dans la
  boucle au depart
  for k to 100 while test=false do
    # calcul du nouveau vecteur Y a
  partir de l'ancien : X
    for i to n do
```

```
  Y[i]:=B[i]-sum(A[i,'j']*X['j'],'j'=1..
  i-1);

  Y[i]:=Y[i]-sum(A[i,'j']*X['j'],'j'=i+1
  ..n);
      Y[i]:=Y[i]/A[i,i];
    od;
    # calcul de la variation relative et
  du test
    # test est un booleen qui vaut true
  ou false

  test:=evalb(evalf(norm(Y-X))<evalf(eps
  ilon*norm(X)));
    # print(test);
    # Y remplace X
    X:=copy(Y);
    print(evalm(Y));
  od;
  printf("nombre d'iterations utiles
  :%d",k-1);
  RETURN(evalm(Y));
  end:
```

On notera l'emploi de la boucle
for k to 100 **while** test=false
qui signifie boucler sur k tant que le test
n'est pas satisfait. La borne 100 est un garde-fou pour le cas ou la methode ne converge pas.

Attention : pour les copies de variables de type matrice,vecteur... il faut absolument utiliser **copy.**

Essayons

```
> A:=matrix(4,4,(i,j)->if i=j then 4
  elif abs(i-j)=1 then -1 else 0 fi);
```

$$A := \begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix}$$

```
> X0:=vector(4,i->0.);B:=([1.,2.,3.,4.])
  ;
```

$$X0 := [0, 0, 0, 0]$$

$$B := [1., 2., 3., 4.]$$

```
> X:=jacobi(A,B,X0,0.001);
```

[.2500000000, .5000000000, .7500000000, 1.000000000]

[.3750000000, .7500000000, 1.125000000, 1.187500000]

[.4375000000, .8750000000, 1.234375000, 1.281250000]

[.4687500000, .9179687500, 1.289062500, 1.308593750]

[.4794921875, .9394531250, 1.306640625, 1.322265625]

[.4848632813, .9465332033, 1.315429688, 1.326660156]

[.4866333008, .9500732423, 1.318298340, 1.328857422]

[.4875183105, .9512329103, 1.319732666, 1.329574585]

[.4878082275, .9518127443, 1.320201874, 1.329933167]

```
nombre d'iterations utiles :9
```

$X :=$

[.4878082275, .9518127443, 1.320201874, 1.329933167]

Comparons a la solution exacte.

```
> Xexact:=linsolve(A,B);norm(X-Xexact);
```

Xexact :=

[.4880382775, .9521531101, 1.320574163, 1.330143541]

.000372289

Pas mal! Ressayons avec un epsilon plus petit.

```
> X:=jacobi(A,B,X0,10^(-6));norm(X-Xexac
  t);
```

[.2500000000, .5000000000, .7500000000, 1.000000000]

[.3750000000, .7500000000, 1.125000000, 1.187500000]

[.4375000000, .8750000000, 1.234375000, 1.281250000]

[.4687500000, .9179687500, 1.289062500, 1.308593750]

[.4794921875, .9394531250, 1.306640625, 1.322265625]

[.4848632813, .9465332033, 1.315429688, 1.326660156]

[.4866333008, .9500732423, 1.318298340, 1.328857422]

[.4875183105, .9512329103, 1.319732666, 1.329574585]

[.4878082275, .9518127443, 1.320201874, 1.329933167]

[.4879531860, .9520025255, 1.320436478, 1.330050469]

[.4880006315, .9520974160, 1.320513249, 1.330109120]

[.4880243540, .9521284703, 1.320551634, 1.330128312]

[.4880321175, .9521439970, 1.320564196, 1.330137909]

[.4880359993, .9521490785, 1.320570477, 1.330141049]

[.4880372698, .9521516190, 1.320572532, 1.330142619]

[.4880379048, .9521524505, 1.320573560, 1.330143133]

```
nombre d'iterations utiles :16
```

$X :=$

[.4880379048, .9521524505, 1.320573560, 1.330143133]

$.6596\ 10^{-6}$

```
>
```

b) Methode de Gauss-Seidel

Cette fois, un seul vecteur X suffit en theorie car on reutilise les coordonnees deja calculees.
En fait, il faut un second vecteur Y pour stocker X afin de pourvoir effectuer le test.

```
> gauss_seidel:=proc(A,B,X0,epsilon)
  local i,k,n,X,Y,test;
  # option trace;
  n:=rowdim(A);
  X:=copy(X0);
  test:=false; # pour rentrer dans la
  boucle au depart
  for k to 100 while test=false do
    # on enregistre X avant de le
  remplacer
    Y:=copy(X);
    # calcul du nouveau vecteur X
    for i to n do

  X[i]:=B[i]-sum(A[i,'j']*X['j'],'j'=1..
  i-1);

  X[i]:=X[i]-sum(A[i,'j']*X['j'],'j'=i+1
```

```
  ..n);
      X[i]:=X[i]/A[i,i];
    od;
    # calcul de la variation relative et
  du test
    # test est un booleen qui vaut true
  ou false

  test:=evalb(evalf(norm(X-Y))<evalf(eps
  ilon*norm(Y)));
    print(evalm(X));
  od;
  printf("nombre d'iterations utiles
  :%d",k-1);
  RETURN(evalm(X));
  end:
```

Essayons

```
> X:=gauss_seidel(A,B,X0,10^(-3));norm(X
  -Xexact);
```

[.2500000000, .5625000000, .8906250000, 1.222656250]

[.3906250000, .8203125000, 1.260742188, 1.315185547]

[.4550781250, .9289550783, 1.311035156, 1.327758789]

[.4822387695, .9483184815, 1.319019318, 1.329754830]

[.4870796205, .9515247348, 1.320319891, 1.330079973]

[.4878811838, .9520502688, 1.320532561, 1.330133140]

```
nombre d'iterations utiles :6
```

$X :=$

[.4878811838, .9520502688, 1.320532561, 1.330133140]

.0001570937

6 iterations au lieu de 9 pour Jacobi.

```
> X:=gauss_seidel(A,B,X0,10^(-6));norm(X
  -Xexact);
```

[.2500000000, .5625000000, .8906250000, 1.222656250]

[.3906250000, .8203125000, 1.260742188, 1.315185547]

[.4550781250, .9289550783, 1.311035156, 1.327758789]

[.4822387695, .9483184815, 1.319019318, 1.329754830]

[.4870796205, .9515247348, 1.320319891, 1.330079973]

[.4878811838, .9520502688, 1.320532561, 1.330133140]

[.4880125673, .9521362820, 1.320567356, 1.330141839]

[.4880340705, .9521503568, 1.320573049, 1.330143262]

[.4880375893, .9521526595, 1.320573981, 1.330143495]

[.4880381650, .9521530365, 1.320574133, 1.330143533]

```
nombre d'iterations utiles :10
```

$X :=$

[.4880381650, .9521530365, 1.320574133, 1.330143533]

$.1125\ 10^{-6}$

Cette fois, c'est 10 au lieu de 16. Gauss-Seidel est dans ce cas nettement plus rapide.

Exercice 3 : matrices d'iterations

On automatise ici les calculs de matrices d'iterations faits a l'exercice 1.
Pour Jacobi, on sait donner directement l'expression de P.

```
> iter:=proc(A)
```

```
  local PJ,MGS,NGS,PGS,rho,n,i,S,L,phi,chi;
  n:=rowdim(A);
  # Jacobi
  PJ:=matrix(n,n,(i,j)->if i<>j then
  -A[i,j]/A[i,i] else 0 fi);
  chi:=charpoly(PJ,x);
  S:=fsolve(chi);
  L:=seq(abs(S[i]),i=1..n);
  rho:=max(L);
  printf("matrice d'iteration de Jacobi
  :");
  print(evalm(PJ));
  printf("rayon spectral : %f",rho);
  #Gauss-Seidel
  MGS:=matrix(n,n,(i,j)->if i>=j then
  A[i,j] else 0 fi);
  NGS:=evalm(MGS-A);
  PGS:=evalm(inverse(MGS)&*NGS);
  phi:=charpoly(PGS,x);
  S:=evalf(solve(phi));
  L:=seq(abs(S[i]),i=1..n);
  rho:=max(L);
  printf("\nmatrice d'iteration de
  Gauss-Seidel :");
  print(evalm(PGS));
  printf("rayon spectral : %f",rho);
  end:
>
```

Essayons

```
> A:=matrix(5,5,(i,j)->if i=j then 2 else 1
  fi);
```

$$A := \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix}$$

```
> iter(A);
matrice d'iteration de Jacobi :
```

$$\begin{bmatrix} 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & \frac{-1}{2} \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 \end{bmatrix}$$

```
rayon spectral : 2.000000
matrice d'iteration de Gauss-Seidel :
```

$$\begin{bmatrix} 0 & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ 0 & \frac{1}{4} & \frac{-1}{4} & \frac{-1}{4} & \frac{-1}{4} \\ 0 & \frac{1}{8} & \frac{3}{8} & \frac{-1}{8} & \frac{-1}{8} \\ 0 & \frac{1}{16} & \frac{3}{16} & \frac{7}{16} & \frac{-1}{16} \\ 0 & \frac{1}{32} & \frac{3}{32} & \frac{7}{32} & \frac{15}{32} \end{bmatrix}$$

```
rayon spectral : .543747
```

Autre essai avec la matrice de l'exercice 2

```
> A:=matrix(4,4,(i,j)->if i=j then 4 elif
  abs(i-j)=1 then -1 else 0 fi);
```

$$A := \begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix}$$

```
> iter(A);
matrice d'iteration de Jacobi :
```

$$\begin{bmatrix} 0 & \frac{1}{4} & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

```
rayon spectral : .404508
matrice d'iteration de Gauss-Seidel :
```

$$\begin{bmatrix} 0 & \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{16} & \frac{1}{4} & 0 \\ 0 & \frac{1}{64} & \frac{1}{16} & \frac{1}{4} \\ 0 & \frac{1}{256} & \frac{1}{64} & \frac{1}{16} \end{bmatrix}$$

```
rayon spectral : .163627
```

Ca baigne!

Exercice 4 : deux exemples sur mesure!

Jacobi converge et Gauss-Seidel diverge

```
> A1:=matrix(3,3,[1,2,-2,1,1,1,2,2,1]);i
  ter(A1);
```

$$A1 := \begin{bmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{bmatrix}$$

```
matrice d'iteration de Jacobi :
```

$$\begin{bmatrix} 0 & -2 & 2 \\ -1 & 0 & -1 \\ -2 & -2 & 0 \end{bmatrix}$$

```
rayon spectral : 0.000000
matrice d'iteration de Gauss-Seidel :
```

$$\begin{bmatrix} 0 & -2 & 2 \\ 0 & 2 & -3 \\ 0 & 0 & 2 \end{bmatrix}$$

```
rayon spectral : 2.000000
```

Verifions la convergence de l'un et la divergence de l'autre.

```
>
> B:=([1.,1.,1.]);X0:=([0,0,0]);
```

$$B := [1., 1., 1.]$$

$$X0 := [0, 0, 0]$$

```
> jacobi(A1,B,X0,0.0001);
```

$$[1., 1., 1.]$$

$$[1., -1., -3.]$$

$$[-3., 3., 1.]$$

[-3., 3., 1.]

nombre d'iterations utiles :4

[-3., 3., 1.]

Ca converge et vers la bonne valeur!

```
> linsolve(A1,B);
```

[-3.000000001, 3.000000001, 1.000000000]

```
> gauss_seidel(A1,B,X0,0.0001);
```

[1., 0, -1.]

[-1., 3., -3.]

[-11., 15., -7.]

[-43., 51., -15.]

[-131., 147., -31.]

[-355., 387., -63.]

[-899., 963., -127.]

[-2179., 2307., -255.]

[-5123., 5379., -511.]

[-11779., 12291., -1023.]

[-26627., 27651., -2047.]

[-59395., 61443., -4095.]

[-131075., 135171., -8191.]

[-286723., 294915., -16383.]

[-622595., 638979., -32767.]

$[-.1343491\ 10^{7}, .1376259\ 10^{7}, -65535.]$

$$[-.2883587\ 10^{7}, .2949123\ 10^{7}, -131071.]$$

$$[-.6160387\ 10^{7}, .6291459\ 10^{7}, -262143.]$$

$$[-.13107203\ 10^{8}, .13369347\ 10^{8}, -524287.]$$

$$[-.27787267\ 10^{8}, .28311555\ 10^{8}, -.1048575\ 10^{7}]$$

$$[-.58720259\ 10^{8}, .59768835\ 10^{8}, -.2097151\ 10^{7}]$$

$$[-.123731971\ 10^{9}, .125829123\ 10^{9}, -.4194303\ 10^{7}]$$

$$[-.260046851\ 10^{9}, .264241155\ 10^{9}, -.8388607\ 10^{7}]$$

$$[-.545259523\ 10^{9}, .553648131\ 10^{9}, -.16777215\ 10^{8}]$$

$$[-.1140850691\ 10^{10}, .1157627907\ 10^{10}, -.33554431\ 10^{8}]$$

$$[-.2382364675\ 10^{10}, .2415919107\ 10^{10}, -.67108863\ 10^{8}]$$

$$[-.4966055939\ 10^{10}, .5033164803\ 10^{10}, -.134217731\ 10^{9}]$$

$$[-.1033476507\ 10^{11}, .1046898280\ 10^{11}, -.268435459\ 10^{9}]$$

$$[-.2147483652\ 10^{11}, .2174327198\ 10^{11}, -.536870919\ 10^{9}]$$

$$[-.4456028580\ 10^{11}, .4509715672\ 10^{11}, -.1073741839\ 10^{10}]$$

$$[-.9234179712\ 10^{11}, .9341553896\ 10^{11}, -.2147483699\ 10^{10}]$$

$$[-.1911260453\ 10^{12}, .1932735290\ 10^{12}, -.4294967399\ 10^{10}]$$

[
-.3951369928 10^{12}, .3994319602 10^{12}, -.8589934799 10^{10}
]
[
-.8160437900 10^{12}, .8246337248 10^{12}, -.1717987000 10^{11}
]
[
-.1683627190 10^{13}, .1700807060 10^{13}, -.3435974000 10^{11}
]
[
-.3470333600 10^{13}, .3504693340 10^{13}, -.6871948000 10^{11}
]
[
-.7146825640 10^{13}, .7215545120 10^{13}, -.1374389600 10^{12}
]
[
-.1470596816 10^{14}, .1484340712 10^{14}, -.2748779200 10^{12}
]
[
-.3023657008 10^{14}, .3051144800 10^{14}, -.5497558400 10^{12}
]
[-.6212240768 10^{14}, .6267216352 10^{14}, -.10995116 10^{13}]

$[-.1275433502\ 10^{15}, .1286428618\ 10^{15}, -.21990232\ 10^{13}]$

$[-.2616837700\ 10^{15}, .2638827932\ 10^{15}, -.43980464\ 10^{13}]$

$[-.5365616792\ 10^{15}, .5409597256\ 10^{15}, -.8796093\ 10^{13}]$

$[-.1099511637\ 10^{16}, .1108307730\ 10^{16}, -.17592186\ 10^{14}]$

$[-.2251799832\ 10^{16}, .2269392018\ 10^{16}, -.35184372\ 10^{14}]$

$[-.4609152780\ 10^{16}, .4644337152\ 10^{16}, -.70368744\ 10^{14}]$

$[-.9429411792\ 10^{16}, .9499780536\ 10^{16}, -.14073749\ 10^{15}]$

$[-.1928103605\ 10^{17}, .1942177354\ 10^{17}, -.28147498\ 10^{15}]$

$[-.3940649704\ 10^{17}, .3968797202\ 10^{17}, -.56294996\ 10^{15}]$

$[-.8050184396\ 10^{17}, .8106479392\ 10^{17}, -.11258999\ 10^{16}]$

$[-.1643813876\ 10^{18}, .1655072875\ 10^{18}, -.22517998\ 10^{16}]$

$[-.3355181746\ 10^{18}, .3377699744\ 10^{18}, -.45035996\ 10^{16}]$

$[-.6845471480\ 10^{18}, .6890507476\ 10^{18}, -.9007199\ 10^{16}]$

$[-.1396115893\ 10^{19}, .1405123092\ 10^{19}, -.18014398\ 10^{17}]$

$[-.2846274980\ 10^{19}, .2864289378\ 10^{19}, -.36028796\ 10^{17}]$

$[-.5800636348\ 10^{19}, .5836665144\ 10^{19}, -.7205759\ 10^{17}]$

$[-.1181744547\ 10^{20}, .1188950306\ 10^{20}, -.14411518\ 10^{18}]$

$[-.2406723648\ 10^{20}, .2421135166\ 10^{20}, -.28823036\ 10^{18}]$

$[-.4899916404\ 10^{20}, .4928739440\ 10^{20}, -.57646072\ 10^{18}]$

$[-.9972771024\ 10^{20}, .1003041710\ 10^{21}, -.11529215\ 10^{19}]$

$[-.2029141850\ 10^{21}, .2040671065\ 10^{21}, -.23058430\ 10^{19}]$

$[-.4127458990\ 10^{21}, .4150517420\ 10^{21}, -.46116860\ 10^{19}]$

$[-.8393268560\ 10^{21}, .8439385420\ 10^{21}, -.9223372\ 10^{19}]$

$[-.1706323828\ 10^{22}, .1715547200\ 10^{22}, -.18446744\ 10^{20}]$

$[-.3467987888\ 10^{22}, .3486434632\ 10^{22}, -.36893488\ 10^{20}]$

$[-.7046656240\ 10^{22}, .7083549728\ 10^{22}, -.7378698\ 10^{20}]$

$[-.1431467342\ 10^{23}, .1438846040\ 10^{23}, -.14757396\ 10^{21}]$

$[-.2907206872\ 10^{23}, .2921964268\ 10^{23}, -.29514792\ 10^{21}]$

$[-.5902958120\ 10^{23}, .5932472912\ 10^{23}, -.5902958\ 10^{21}]$

$[-.1198300498\ 10^{24}, .1204203456\ 10^{24}, -.11805916\ 10^{22}]$

$[-.2432018744\ 10^{24}, .2443824660\ 10^{24}, -.23611832\ 10^{22}]$

$[-.4934872984\ 10^{24}, .4958484816\ 10^{24}, -.47223664\ 10^{22}]$

$[-.1001141696\ 10^{25}, .1005864062\ 10^{25}, -.9444732\ 10^{22}]$

$[-.2030617588\ 10^{25}, .2040062320\ 10^{25}, -.18889464\ 10^{23}]$

$[-.4117903568\ 10^{25}, .4136793032\ 10^{25}, -.37778928\ 10^{23}]$

$[-.8349143920\ 10^{25}, .8386922848\ 10^{25}, -.7555786\ 10^{23}]$

$[-.1692496142\ 10^{26}, .1700051928\ 10^{26}, -.15111572\ 10^{24}]$

$[-.3430327000\ 10^{26}, .3445438572\ 10^{26}, -.30223144\ 10^{24}]$

$[-.6951323432\ 10^{26}, .6981546576\ 10^{26}, -.6044629\ 10^{24}]$

$[-.1408398573\ 10^{27}, .1414443202\ 10^{27}, -.12089258\ 10^{25}]$

$[-.2853064920\ 10^{27}, .2865154178\ 10^{27}, -.24178516\ 10^{25}]$

$[-.5778665388\ 10^{27}, .5802843904\ 10^{27}, -.4835703\ 10^{25}]$

$[-.1170240187\ 10^{28}, .1175075890\ 10^{28}, -.9671406\ 10^{25}]$

$[-.2369494592\ 10^{28}, .2379165998\ 10^{28}, -.19342812\ 10^{26}]$

$[-.4797017620\ 10^{28}, .4816360432\ 10^{28}, -.38685624\ 10^{26}]$

$[-.9710092112\ 10^{28}, .9748777736\ 10^{28}, -.7737125\ 10^{26}]$

$[-.1965229797\ 10^{29}, .1972966922\ 10^{29}, -.15474250\ 10^{27}]$

$[-.3976882344\ 10^{29}, .3992356594\ 10^{29}, -.30948500\ 10^{27}]$

$[-.8046610188\ 10^{29}, .8077558688\ 10^{29}, -.6189700\ 10^{27}]$

$[-.1627891138\ 10^{30}, .1634080838\ 10^{30}, -.12379400\ 10^{28}]$

$[-.3292920476\ 10^{30}, .3305299876\ 10^{30}, -.24758800\ 10^{28}]$

$[-.6660117352\ 10^{30}, .6684876152\ 10^{30}, -.4951760\ 10^{28}]$

$[-.1346878750\ 10^{31}, .1351830510\ 10^{31}, -.9903520\ 10^{28}]$

$[-.2723468060\ 10^{31}, .2733371580\ 10^{31}, -.19807040\ 10^{29}]$

$[-.5506357240\ 10^{31}, .5526164280\ 10^{31}, -.3961408\ 10^{29}]$

$[-.1113155672\ 10^{32}, .1117117080\ 10^{32}, -.7922816\ 10^{29}]$

$[-.2250079792\ 10^{32}, .2258002608\ 10^{32}, -.15845632\ 10^{30}]$

$[-.4547696480\ 10^{32}, .4563542112\ 10^{32}, -.31691264\ 10^{30}]$

$[-.9190466752\ 10^{32}, .9222158016\ 10^{32}, -.6338253\ 10^{30}]$

$[-.1857108109\ 10^{33}, .1863446362\ 10^{33}, -.12676506\ 10^{31}]$

```
nombre d’iterations utiles :100
```

$[-.1857108109\ 10^{33}, .1863446362\ 10^{33}, -.12676506\ 10^{31}]$

Ca diverge.

Gauss-Seidel converge et Jacobi diverge

```
> A2:=matrix(3,3,[2,-1,1,2,2,2,-1,-1,2])
  ;iter(A2);
```

$$A2 := \begin{bmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{bmatrix}$$

matrice d'iteration de Jacobi :

$$\begin{bmatrix} 0 & \frac{1}{2} & \frac{-1}{2} \\ -1 & 0 & -1 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

rayon spectral : 0.000000

matrice d'iteration de Gauss-Seidel :

$$\begin{bmatrix} 0 & \frac{1}{2} & \frac{-1}{2} \\ 0 & \frac{-1}{2} & \frac{-1}{2} \\ 0 & 0 & \frac{-1}{2} \end{bmatrix}$$

rayon spectral : .500000

>

```
> jacobi(A2,B,X0,0.0001);
```

[.5000000000, .5000000000, .5000000000]

[.5000000000, -.5000000000, 1.000000000]

[-.2500000000, -1.000000000, .5000000000]

[-.2500000000, .2500000000, -.1250000000]

[.6875000000, .8750000000, .5000000000]

[.6875000000, -.6875000000, 1.281250000]

[-.4843750000, -1.468750000, .5000000000]

[-.4843750000, .4843750000, -.4765625000]

[.9804687500, 1.460937500, .5000000000]

[.9804687500, -.9804687500, 1.720703125]

[-.8505859375, -2.201171875, .5000000000]

[-.8505859375, .8505859375, -1.025878907]

[1.438232423, 2.376464845, .5000000000]

[1.438232423, -1.438232423, 2.407348634]

[-1.422790529, -3.345581057, .5000000000]

[-1.422790529, 1.422790529, -1.884185793]

[2.153488161, 3.806976322, .5000000000]

[2.153488161, -2.153488161, 3.480232242]

[-2.316860202, -5.133720405, .5000000000]

[-2.316860203, 2.316860202, -3.225290304]

[3.271075253, 6.042150505, .4999999995]

[3.271075253, -3.271075253, 5.156612880]

[-3.713844067, -7.927688135, .5000000000]

[-3.713844068, 3.713844067, -5.320766100]

[5.017305085, 9.534610170, .4999999995]

[5.017305085, -5.017305085, 7.775957630]

[-5.896631360, -12.29326272, .5000000000]

[-5.896631360, 5.896631360, -8.594947040]

[7.745789200, 14.99157840, .5000000000]

[7.745789200, -7.745789200, 11.86868380]

[-9.307236500, -19.11447300, .5000000000]

[-9.307236500, 9.307236500, -13.71085475]

[12.00904563, 23.51809125, .5000000000]

[12.00904563, -12.00904563, 18.26356844]

[-14.63630704, -29.77261407, .5000000000]

[-14.63630704, 14.63630704, -21.70446056]

[18.67038380, 36.84076760, .5000000000]

[18.67038380, -18.67038380, 28.25557570]

[-22.96297975, -46.42595950, .5000000000]

[-22.96297975, 22.96297975, -34.19446963]

[29.07872469, 57.65744940, .5000000000]

[29.07872470, -29.07872469, 43.86808705]

[-35.97340587, -72.44681175, .5000000050]

[-35.97340588, 35.97340587, -53.71010880]

[45.34175734, 90.18351470, .4999999950]

[45.34175736, -45.34175734, 68.26263600]

[-56.30219665, -113.1043934, .5000000100]

[-56.30219670, 56.30219665, -84.20329505]

[70.75274585, 141.0054918, .4999999750]

[70.75274590, -70.75274585, 106.3791189]

[-88.06593240, -176.6318648, .5000000250]

[-88.06593240, 88.06593240, -131.8488986]

[110.4574155, 220.4148310, .5000000000]

[110.4574155, -110.4574155, 165.9361233]

[-137.6967694, -275.8935388, .5000000000]

[-137.6967694, 137.6967694, -206.2951541]

[172.4959618, 344.4919235, .5000000000]

[172.4959618, -172.4959618, 258.9939427]

[-215.2449523, -430.9899045, .5000000000]

[-215.2449523, 215.2449523, -322.6174284]

[269.4311904, 538.3623805, .5000000000]

[269.4311903, -269.4311904, 404.3967855]

[-336.4139880, -673.3279760, .4999999500]

[-336.4139880, 336.4139881, -504.3709820]

[420.8924851, 841.2849700, .5000000500]

[420.8924850, -420.8924852, 631.5887275]

[-525.7406065, -1051.981213, .4999999000]

[-525.7406065, 525.7406065, -788.3609100]

[657.5507585, 1314.601517, .5000000000]

[657.5507585, -657.5507585, 986.5761380]

[-821.5634485, -1643.626897, .5000000000]

[-821.5634485, 821.5634485, -1232.095173]

[1027.329311, 2054.158622, .5000000000]

[1027.329311, -1027.329311, 1541.243967]

[-1283.786639, -2568.073278, .5000000000]

[-1283.786639, 1283.786639, -1925.429959]

[1605.108299, 3209.716598, .5000000000]

[1605.108299, -1605.108299, 2407.912449]

[-2006.010374, -4012.520748, .5000000000]

[-2006.010374, 2006.010374, -3008.765561]

[2507.887968, 5015.275935, .5000000000]

[2507.887968, -2507.887968, 3762.081952]

[-3134.484960, -6269.469920, .5000000000]

[-3134.484960, 3134.484960, -4701.477440]

[3918.481200, 7836.462400, .5000000000]

[3918.481200, -3918.481200, 5877.971800]

[-4897.726500, -9795.953000, .5000000000]

[-4897.726500, 4897.726500, -7346.339750]

[6122.533125, 12244.56625, .5000000000]

[6122.533125, -6122.533125, 9184.049690]

[-7652.791410, -15306.08282, .5000000000]

[-7652.791410, 7652.791410, -11478.93712]

[9566.364265, 19132.22853, .5000000000]

[9566.364265, -9566.364265, 14349.79640]

[-11957.58034, -23915.66067, .5000000000]

[-11957.58034, 11957.58034, -17936.12051]

[14947.35043, 29894.20085, .5000000000]

[14947.35043, -14947.35043, 22421.27564]

[-18683.81304, -37368.12607, .5000000000]

[-18683.81304, 18683.81304, -28025.46956]

```
nombre d’iterations utiles :100
```

[-18683.81304, 18683.81304, -28025.46956]

Il est clair que ca diverge!

```
> gauss_seidel(A2,B,X0,0.0001);
```

[.5000000000, 0, .7500000000]

[.1250000000, -.3750000000, .3750000000]

[.1250000000, 0, .5625000000]

[.2187500000, -.2812500000, .4687500000]

[.1250000000, -.09375000000, .5156250000]

[.1953125000, -.2109375000, .4921875000]

[.1484375000, -.1406250000, .5039062500]

[.1777343750, -.1816406250, .4980468750]

[.1601562500, -.1582031250, .5009765625]

[.1704101563, -.1713867188, .4995117188]

[.1645507812, -.1640625000, .5002441405]

[.1678466798, -.1680908203, .4998779298]

[.1660156250, -.1658935548, .5000610350]

[.1670227051, -.1670837401, .4999694825]

[.1664733887, -.1664428712, .5000152590]

[.1667709349, -.1667861939, .4999923705]

[.1666107178, -.1666030883, .5000038150]

[.1666965484, -.1667003634, .4999980925]

[.1666507721, -.1666488646, .5000009540]

[.1666750907, -.1666760447, .4999995230]

```
nombre d’iterations utiles :20
```

[.1666750907, -.1666760447, .4999995230]

Ca converge, verifions la solution.

```
> linsolve(A2,B);
```

[.1666666667, -.1666666667, .5000000000]

Pas mal!

Exercice 5

On construit la matrice.

```
> A:=(n,a)->matrix(n,n,(i,j)->if i=j then a
  else 1 fi):
```

Verifions.

```
> A(3,2);
```

$$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

Youpi!

Jacobi

Construisons PJ

```
> n:=3;B:=A(n,a);
>
```

$$n := 3$$

$$B := \begin{bmatrix} a & 1 & 1 \\ 1 & a & 1 \\ 1 & 1 & a \end{bmatrix}$$

```
> PJ:=matrix(n,n,(i,j)->if i<>j then
  -B[i,j]/B[i,i] else 0 fi);
```

$$PJ := \begin{bmatrix} 0 & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & 0 & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & 0 \end{bmatrix}$$

```
> chi:=charpoly(PJ,X);solve(chi,X);
```

$$\chi := \frac{X^3 a^3 - 3 X a + 2}{a^3}$$

$$-2\frac{1}{a}, \frac{1}{a}, \frac{1}{a}$$

```
> for n from 2 to 5 do
    B:=A(n,a):
    PJ:=matrix(n,n,(i,j)->if i<>j then
  -B[i,j]/B[i,i] else 0 fi):
    chi:=charpoly(PJ,X):
    solve(chi,X);
  od;
```

$$B := \begin{bmatrix} a & 1 \\ 1 & a \end{bmatrix}$$

$$PJ := \begin{bmatrix} 0 & -\frac{1}{a} \\ -\frac{1}{a} & 0 \end{bmatrix}$$

$$\chi := \frac{X^2 a^2 - 1}{a^2}$$

$$\frac{1}{a}, -\frac{1}{a}$$

$$B := \begin{bmatrix} a & 1 & 1 \\ 1 & a & 1 \\ 1 & 1 & a \end{bmatrix}$$

$$PJ := \begin{bmatrix} 0 & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & 0 & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & 0 \end{bmatrix}$$

$$\chi := \frac{X^3 a^3 - 3 X a + 2}{a^3}$$

$$-2\,\frac{1}{a}, \frac{1}{a}, \frac{1}{a}$$

$$B := \begin{bmatrix} a & 1 & 1 & 1 \\ 1 & a & 1 & 1 \\ 1 & 1 & a & 1 \\ 1 & 1 & 1 & a \end{bmatrix}$$

$$PJ := \begin{bmatrix} 0 & -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & 0 & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & 0 & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} & 0 \end{bmatrix}$$

$$\chi := \frac{X^4 a^4 - 6 X^2 a^2 + 8 X a - 3}{a^4}$$

$$-3 \frac{1}{a}, \frac{1}{a}, \frac{1}{a}, \frac{1}{a}$$

$$B := \begin{bmatrix} a & 1 & 1 & 1 & 1 \\ 1 & a & 1 & 1 & 1 \\ 1 & 1 & a & 1 & 1 \\ 1 & 1 & 1 & a & 1 \\ 1 & 1 & 1 & 1 & a \end{bmatrix}$$

$$PJ := \begin{bmatrix} 0 & -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & 0 & -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & 0 & -\frac{1}{a} & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} & 0 & -\frac{1}{a} \\ -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} & -\frac{1}{a} & 0 \end{bmatrix}$$

$$\chi := \frac{X^5 a^5 - 10 X^3 a^3 + 20 X^2 a^2 - 15 X a + 4}{a^5}$$

$$-4\frac{1}{a}, \frac{1}{a}, \frac{1}{a}, \frac{1}{a}, \frac{1}{a}$$

On observe (et on peut le montrer facilement) que le spectre de PJ est {-(n-1)/a,1/a}.
le rayon spectral est donc (n-1)/a et on en deduit que la methode converge si et seulement si n-1<a.
verifions sur des exemples.

```
> n:=5;X0:=vector(n,i->0);B:=vector(n,i->1.);AA:=A(5,5.);
  jacobi(AA,B,X0,10^(-2));linsolve(AA,B);
```

$$n := 5$$

$$X0 := [0, 0, 0, 0, 0]$$

$$B := [1., 1., 1., 1., 1.]$$

$$AA := \begin{bmatrix} 5. & 1 & 1 & 1 & 1 \\ 1 & 5. & 1 & 1 & 1 \\ 1 & 1 & 5. & 1 & 1 \\ 1 & 1 & 1 & 5. & 1 \\ 1 & 1 & 1 & 1 & 5. \end{bmatrix}$$

[.2000000000, .2000000000, .2000000000, .2000000000, .2000000000]

[.04000000000, .04000000000, .04000000000, .04000000000, .04000000000]

[.1680000000, .1680000000, .1680000000, .1680000000, .1680000000]

[.06560000000, .06560000000, .06560000000, .06560000000, .06560000000]

[.1475200000, .1475200000, .1475200000, .1475200000, .1475200000]

[.08198400000, .08198400000, .08198400000, .08198400000, .08198400000]

[.1344128000, .1344128000, .1344128000, .1344128000, .1344128000]

[.09246976000, .09246976000, .09246976000,

.09246976000, .09246976000]
[.1260241920, .1260241920, .1260241920, .1260241920,
.1260241920]
[.09918064640, .09918064640, .09918064640,
.09918064640, .09918064640]
[.1206554829, .1206554829, .1206554829, .1206554829,
.1206554829]
[.1034756137, .1034756137, .1034756137, .1034756137,
.1034756137]
[.1172195090, .1172195090, .1172195090, .1172195090,
.1172195090]
[.1062243928, .1062243928, .1062243928, .1062243928,
.1062243928]
[.1150204858, .1150204858, .1150204858, .1150204858,
.1150204858]
[.1079836114, .1079836114, .1079836114, .1079836114,
.1079836114]
[.1136131109, .1136131109, .1136131109, .1136131109,
.1136131109]
[.1091095113, .1091095113, .1091095113, .1091095113,
.1091095113]
[.1127123910, .1127123910, .1127123910, .1127123910,
.1127123910]

[.1098300872, .1098300872, .1098300872, .1098300872, .1098300872]

[.1121359302, .1121359302, .1121359302, .1121359302, .1121359302]

[.1102912558, .1102912558, .1102912558, .1102912558, .1102912558]

[.1117669954, .1117669954, .1117669954, .1117669954, .1117669954]

[.1105864037, .1105864037, .1105864037, .1105864037, .1105864037]

[.1115308770, .1115308770, .1115308770, .1115308770, .1115308770]

```
nombre d'iterations utiles :25
```

[.1115308770, .1115308770, .1115308770, .1115308770, .1115308770]

[.1111111112, .1111111111, .1111111111, .1111111111, .1111111111]

Ca converge mais lentement.

```
> n:=5;X0:=vector(n,i->0);B:=vector(n,i->1.);AA:=A(5,4.);
  jacobi(AA,B,X0,10^(-2));linsolve(AA,B);
```

$$n := 5$$

$$X0 := [0, 0, 0, 0, 0]$$

$$B := [1., 1., 1., 1., 1.]$$

$$AA := \begin{bmatrix} 4. & 1 & 1 & 1 & 1 \\ 1 & 4. & 1 & 1 & 1 \\ 1 & 1 & 4. & 1 & 1 \\ 1 & 1 & 1 & 4. & 1 \\ 1 & 1 & 1 & 1 & 4. \end{bmatrix}$$

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,

.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]

[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,
.2500000000]
[0, 0, 0, 0, 0]
[.2500000000, .2500000000, .2500000000, .2500000000,

.2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

[.2500000000, .2500000000, .2500000000, .2500000000, .2500000000]

[0, 0, 0, 0, 0]

nombre d'iterations utiles :100

[0, 0, 0, 0, 0]

[.1250000001, .1250000000, .1250000000, .1250000000, .1250000000]

La, ca diverge!

```
> n:=5;X0:=vector(n,i->0);B:=vector(n,i->1.);AA:=A(5,10.);
  jacobi(AA,B,X0,10^(-2));linsolve(AA,B)
```

;

$$n := 5$$

$$X0 := [0, 0, 0, 0, 0]$$

$$B := [1., 1., 1., 1., 1.]$$

$$AA := \begin{bmatrix} 10. & 1 & 1 & 1 & 1 \\ 1 & 10. & 1 & 1 & 1 \\ 1 & 1 & 10. & 1 & 1 \\ 1 & 1 & 1 & 10. & 1 \\ 1 & 1 & 1 & 1 & 10. \end{bmatrix}$$

[.1000000000, .1000000000, .1000000000, .1000000000,
.1000000000]

[.06000000000, .06000000000, .06000000000,
.06000000000, .06000000000]

[.07600000000, .07600000000, .07600000000,
.07600000000, .07600000000]

[.06960000000, .06960000000, .06960000000,
.06960000000, .06960000000]

[.07216000000, .07216000000, .07216000000,
.07216000000, .07216000000]

[.07113600000, .07113600000, .07113600000,
.07113600000, .07113600000]

[.07154560000, .07154560000, .07154560000,
.07154560000, .07154560000]

```
nombre d'iterations utiles :7
[.07154560000, .07154560000, .07154560000,
    .07154560000, .07154560000]
[.07142857143, .07142857145, .07142857145,
    .07142857145, .07142857141]
```

La, c'est rapide.

Gauss-Seidel

Dans ce cas, le spectre de PGS n'est pas aussi simple.
Il faut faire plusieurs essais et se faire une opinion.
A vous de jouer!

```
>

>
>
>
```

FIN