

Rapport TP2

GIGOUT Thomas

DAUNIQUE Wilfried

Explication du script de l'exercice 7 TP1 :

Le script inclus d'abord grâce au pathname le fichier fonctions.sci ou .sce dans lequel il y a les fonctions nécessaires à son fonctionnement

```
// Import et exécution du fichier fonctions.sce
pathname = get_absolute_file_path("Ex7_Script.sce")
exec(pathname+'fonctions.sce', -1);
```

On initialise ensuite différentes matrices colonnes, ainsi qu'une matrice de taille n,n.

Puis on remplit la matrice A,C et D de nombres aléatoires entre 1 et 10, et la B entre 20 et 30

```
// et de 1 matrice carrée (6x6)
// toutes à 0
A=zeros(n,1);
B=zeros(n,1);
C=zeros(n,1);
X=zeros(n,1);
D=zeros(n,1);
M=zeros(n,n);
Y=zeros(n,1);

//Création random des matrices 5x1
A=round(rand(n,1)*10+1); //Random entre 1 et 10
B=round(rand(n,1)*10+20); //Random entre 20 et 30
C=round(rand(n,1)*10+1); //Random entre 1 et 10
D=round(rand(n,1)*10+1); //Random entre 1 et 10

//Création des trois diagonales de M
for i=2:n-1
```

On alloue ensuite les 3 diagonales de M et on appelle les fonctions nécessaires à la résolution du triangle supérieur, ainsi que le résultat du produit MX, sachant que M n'est pas explicitement stockée. M est représentée par les matrices colonnes A,B,... etc.

```

    //Création des trois diagonales
for i=2:n-1
    M(i,i)=B(i);
    M(i,i+1)=C(i);
    M(i,i-1)=A(i);
end

//Appel de :
//RESOUTRI dans X qui résout le système
//PRODUIT dans Y qui calcule MX
X=RESOUTRI(A,B,C,D,n);
Y=PRODUIT(A,B,C,X,n);

```

```

disp(D,"D=",C,"C=",B,"B=",A,"A=");
disp(M,"Matrice M=");
disp(X,"X=");
disp(Y,"Y (Resultat de MATRICIELLE=");
disp(M*X,"MX = ","Preuve :");
disp(M*X-Y,"réel preuve:")

```

On les affiche ensuite, ainsi que la preuve.

| | | |
|-----|-----------------------------|--------------|
| A= | Matrice M= | |
| 3. | 0. 0. 0. 0. 0. 0. | |
| 9. | 9. 27. 3. 0. 0. 0. | |
| 1. | 0. 1. 29. 6. 0. 0. | |
| 4. | 0. 0. 4. 21. 3. 0. | |
| 8. | 0. 0. 0. 8. 26. 3. | |
| 7. | 0. 0. 0. 0. 0. 0. | |
| B= | X= | |
| 28. | 0.3023134 | |
| 27. | 0.1919032 | |
| 29. | 0.0325979 | |
| 21. | 0.4771261 | |
| 26. | -0.0500136 | |
| 27. | 0.1611146 | |
| C= | Y (Resultat de MATRICIELLE= | |
| 8. | 10. | |
| 3. | 8. | |
| 6. | 4. | |
| 3. | 10. | |
| 3. | 3. | |
| 3. | 4. | |
| 3. | Preuve : | |
| D= | MX = | réel preuve: |
| 10. | 0. | -10. |
| 8. | 8. | 8.882D-16 |
| 4. | 4. | 0. |
| 10. | 10. | 0. |
| 3. | 3. | 4.441D-16 |
| 4. | 0. | -4. |

Explication du script de l'exercice 0 du TP2 :

Le script inclus d'abord grâce au pathname le fichier fonctions.sci ou .sce dans lequel il y a les fonctions nécessaires à son fonctionnement

```
//Import et exécution du fichier fonctions.sci
pathname = get_absolute_file_path("Ex_Script.sce")
exec(pathname+'fonctions.sce', -1);
```

On crée ensuite une matrice 3x3 avec des nombres aléatoires positifs, entre 1 et 10, puis on l'affiche.

```
n=3
//Création d'une Matrice
A=round(rand(n,n)*10+1);
//affichage
disp(A,"Matrice A")
```

"Matrice A

| | | |
|----|----|----|
| 5. | 6. | 6. |
| 4. | 4. | 5. |
| 7. | 7. | 4. |

On résout ensuite la matrice grâce à la méthode magique de Gauss, on inverse ensuite avec la fonction inv(). Les étapes de la résolution seront affichées.

On viendra ensuite afficher la preuve en affichant la matrice inverse moins la matrice résolue par la méthode magique. Si les coefficients sont égaux ou très proches de 0, la preuve est vérifiée.

```
Z=MAGIQUE(A);

//Création de la matrice inverse de A
Y=inv(A)

//Preuve MAGIQUE crée la matrice inverse de A donc Y doit être égale à Z
disp(Y-Z,"preuve:");
disp("Les coefficients sont égaux à 0 ou très proches de 0. C'est alors vérifié")
```

ETAPE:

| | | | | | |
|----|------|------|------|----|----|
| 5. | 6. | 6. | 1. | 0. | 0. |
| 0. | -0.8 | 0.2 | -0.8 | 1. | 0. |
| 0. | -1.4 | -4.4 | -1.4 | 0. | 1. |

ETAPE:

| | | | | | |
|----|------|-------|------------|-------|----|
| 5. | 0. | 7.5 | -5. | 7.5 | 0. |
| 0. | -0.8 | 0.2 | -0.8 | 1. | 0. |
| 0. | 0. | -4.75 | -2.442D-15 | -1.75 | 1. |

ETAPE:

| | | | | | |
|----|------|-----------|------------|-----------|-----------|
| 5. | 0. | 8.882D-16 | -5. | 4.7368421 | 1.5789474 |
| 0. | -0.8 | 0. | -0.8 | 0.9263158 | 0.0421053 |
| 0. | 0. | -4.75 | -2.442D-15 | -1.75 | 1. |

Division:

| | | | | | |
|----|----|-----------|-----------|------------|------------|
| 1. | 0. | 8.882D-16 | -1. | 0.9473684 | 0.3157895 |
| 0. | 1. | 0. | 1. | -1.1578947 | -0.0526316 |
| 0. | 0. | 1. | 5.142D-16 | 0.3684211 | -0.2105263 |

Y =

| | | |
|-----|------------|------------|
| -1. | 0.9473684 | 0.3157895 |
| 1. | -1.1578947 | -0.0526316 |
| 0. | 0.3684211 | -0.2105263 |

preuve:

| | | |
|------------|------------|-----------|
| 2.220D-16 | -3.331D-16 | 0. |
| 2.220D-16 | -2.220D-16 | 0. |
| -5.960D-16 | 4.996D-16 | 1.665D-16 |

Les coefficients sont égaux à 0 ou très proches de 0. C est alors vérifié

Explication du script de l'exercice 1 du TP2

Le script inclus d'abord grâce au pathname le fichier fonctions.sci ou .sce dans lequel il y a les fonctions nécessaires à son fonctionnement

```
// Import et execution du fichier fonctions.sci
pathname = get_absolute_file_path("Ex1_Script.sce")
exec(pathname+'fonctions.sce', -1);
```

On initialise ensuite différentes matrices colonnes, ainsi qu'une matrice de taille n,n.

Puis on remplit la matrice A,C et D de nombres aléatoires entre 1 et 10, et la B entre 20 et 30

```
//et de 1 matrice carrée (6x6)
//toutes à 0
A=zeros(n,1);
B=zeros(n,1);
C=zeros(n,1);
X=zeros(n,1);
D=zeros(n,1);
M=zeros(n,n);
Y=zeros(n,1);

//Création random des matrices 5x1
A=round(rand(n,1)*10+1); //Random entre 1 et 10
B=round(rand(n,1)*10+20); //Random entre 20 et 30
C=round(rand(n,1)*10+1); //Random entre 1 et 10
D=round(rand(n,1)*10+1); //Random entre 1 et 10

//Création des trois diagonales de M
for i=2:n-1
```

On alloue ensuite les 3 diagonales de M et on appelle les fonctions nécessaires à la résolution du triangle supérieur, ainsi que le résultat du produit MX , sachant que M n'est pas explicitement stockée. M est représentée par les matrices colonnes A, B, \dots etc.

| | | | | | | | | | | | |
|-----|-------------------------------------|--------------|-----|-----|-----|-----|-----|-----|-----|-----|----|
| A= | Matrice M= | | | | | | | | | | |
| 7. | 28. | 5. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 9. | 9. | 21. | 5. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 1. | 0. | 1. | 23. | 7. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 8. | 0. | 0. | 8. | 29. | 10. | 0. | 0. | 0. | 0. | 0. | 0. |
| 3. | 0. | 0. | 0. | 3. | 28. | 2. | 0. | 0. | 0. | 0. | 0. |
| 5. | 0. | 0. | 0. | 0. | 5. | 25. | 9. | 0. | 0. | 0. | 0. |
| 9. | 0. | 0. | 0. | 0. | 0. | 9. | 30. | 10. | 0. | 0. | 0. |
| 7. | 0. | 0. | 0. | 0. | 0. | 0. | 7. | 26. | 7. | 0. | 0. |
| 6. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 6. | 30. | 7. | 0. |
| 3. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 3. | 21. | 0. |
| B= | X= | | | | | | | | | | |
| 28. | -0.0271913 | | | | | | | | | | |
| 21. | 0.3451696 | | | | | | | | | | |
| 23. | -0.0007679 | | | | | | | | | | |
| 29. | 0.2389276 | | | | | | | | | | |
| 28. | 0.1077244 | | | | | | | | | | |
| 25. | 0.1334666 | | | | | | | | | | |
| 30. | 0.2360793 | | | | | | | | | | |
| 26. | 0.2716422 | | | | | | | | | | |
| 30. | -0.1021788 | | | | | | | | | | |
| 21. | 0.3479303 | | | | | | | | | | |
| C= | Y (Resultat de PRODUIT MATRICIELLE= | | | | | | | | | | |
| 8. | 2. | | | | | | | | | | |
| 5. | 7. | | | | | | | | | | |
| 7. | 2. | | | | | | | | | | |
| 10. | 8. | | | | | | | | | | |
| 2. | 4. | | | | | | | | | | |
| 9. | 6. | | | | | | | | | | |
| 10. | 11. | | | | | | | | | | |
| 7. | 8. | | | | | | | | | | |
| 7. | 1. | | | | | | | | | | |
| 9. | 7. | | | | | | | | | | |
| D= | MX = | reel preuve: | | | | | | | | | |
| 2. | 0.9644912 | -1.0355088 | | | | | | | | | |
| 7. | 7. | 8.882D-16 | | | | | | | | | |
| 2. | 2. | -2.220D-16 | | | | | | | | | |
| 8. | 8. | 0. | | | | | | | | | |
| 4. | 4. | 0. | | | | | | | | | |
| 6. | 6. | 0. | | | | | | | | | |
| 11. | 11. | 0. | | | | | | | | | |
| 8. | 8. | 0. | | | | | | | | | |
| 1. | 1. | -2.220D-16 | | | | | | | | | |
| 7. | 7. | 8.882D-16 | | | | | | | | | |

Explication du script de l'exercice 2 du TP2

L'équation de Laplace

Le script inclus d'abord grâce au pathname le fichier fonctions.sci ou .sce dans lequel il y a les fonctions nécessaires à son fonctionnement

```
pathname = get_absolute_file_path("Ex2_Script.sce")
exec(pathname+'\fonctions.sce', -1);
```

On crée d'abord une matrice remplie de 0, de taille (n,n) (100x100).

On initialise ensuite toute sa diagonale à 2, et les parties inférieures et supérieures à -1.

On modifie les 4 valeurs qui ne sont pas affectées par la boucle for, à savoir :

- A(1,1) = 2

On va ensuite créer 3 matrices colonnes de 100x1 toujours à 0.

- A(1,2) = -1

Une matrice colonne accueillera les valeurs de la diagonale de A,
une autre accueillera les valeurs de la partie inférieure et la
dernière les valeurs de la partie supérieure.

- A(n,n) = 2

```
A=zeros(n,n); //matrice 100x100
for i=2:n-1
    A(i,i-1)=-1; //partie inférieure initialisée à -1
    A(i,i)=2; //Diagonale initialisée à 2
    A(i,i+1)=-1; //partie supérieure initialisée à -1
end
A(1,1)=2; //Modification des 4 valeurs non touchées par la boucle for
A(1,2)=-1;
A(n,n)=2;
A(n,n-1)=-1;
```

```
J=zeros(n,1); //initialisation de 3 matrices colonnes 100x1 à 0
```

```
K=zeros(n,1);
```

```
L=zeros(n,1);
```

```
for i=2:n-1 //On remplit les valeurs des 3 matrices colonnes selon les valeurs de A
    K(i)=A(i,i); //Toute la Diagonale est stockée dans K
    L(i)=A(i,i+1); //Toute la partie supérieure est stockée dans L
    J(i)=A(i,i-1); //Toute la partie inférieure est stockée dans J
end
```

```
J(n)=A(n,n-1);
```

```
K(1)=A(1,1);
```

```
K(n)=A(n,n);
```

```
L(1)=A(1,2);
```

A=

column 1 to 29

On calcule h. Dans une matrice x, on va ensuite calculer x(i) avec une boucle for ou chaque x(i) prendra la valeur i*h.

On applique ensuite les équations de Laplace dans F et G.

```

h=1/N+1;

    //Equation posée pour les noeuds
x=zeros(n,1);
for i=1:n
    x(i)=i*h;
end

//f(x)=6x-2;
F=zeros(n,1);

for i=1:n
    F(i)=6*x(i)-2;
end

G=zeros(n,1);

for i=1:n
    G(i)=h*h*F(i);
end

```

On utilise ensuite les fonctions RESOUTRI et PRODUIT et nous créerons ensuite la courbe avec linspace.

On affichera ensuite x,z ainsi que la courbe en fonction de x et z.

```

U=RESOUTRI(J,K,L,G,n); //Appel de RESOUTRI
U=PRODUIT(J,K,L,U,n); //Appel de P

//U(1)=0;
//U(0)=0
//U(n+1)=0
U=[0;U;0]

    //Construction de la courbe c
x=linspace(0,1,N+2);
x=x(2:N+1);
x(1)=0;
x(n)=1;
x=x';

z=zeros(n,1);
//U(x)=x*x*(1-x)
for i=1:n
    z(i)=x(i)*x(i)*(1-x(i));
end

//Affichage
disp(x)
disp(z)
plot(x,z);

```



