Rapport Tp1

GIGOUT Thomas

DAUNIQUE Wilfried

Explication du Script Ex5_Script :

Le script d'abord inclus (grace au pathname) le Fichier Fonctions.sci dans lequel il y a les fonctions RESOUSUP et RESOUINF.

```
////////////////////////////////////////////////////////////////////////////
pathname = get_absolute_file_path("EX6_Script.sce")
exec(pathname+'\Fonctions.sci', -1);
////////////////////////////////////////////////////////////////////////////
```

Ensuite on initialise aléatoirement la Matrice A et b pour pouvoir utiliser les fonctions et faire des tests puis on affiche ces matrices :

```
    //INITIALISATION

//initialisation de la taille n
n=5;


//initialisation aléatoire de la Matrice A de taille 5,5
A=round(10*rand(n,n));

//initialisation aléatoire de la Matrice b de taille 5,1
b=round(10*rand(5,1));


  //affichage
disp("affichage de A :")
disp(A);
disp("affichage de b :")
disp(b);
disp("////////////////////////////////////////////////////////////////////////////")
```

On appelle donc d'abord RESOUINF en lui passant en paramètre A,b et n et renvoie X la solution de l'algo RESOUINF qui se trouve être une matrice 5,1 :

```
//Appelle de la fonstion RESOUINF qui renvoit une matrice
//X de taille 5,1 et qui a pour parametres
//la Matrice A et b et la taille n
X=RESOUINF(A,b,n);
```

On test alors avec la Matrice triangulaire inversé de A : D Si la relation AX=b est vérifier on affiche ensuite C qui doit si tout se passe bien afficher 0 :

```
//initalisation d'une Matrice D correspondant A la matrice
//triangulaire inferieur inversible de A
D=tril(A);

//Affichage de la Matrice X solution
disp("Affichage de la Matrice X solution de RESOUINF sur A et b:")
disp(X);

//Test Comparaison
    //si toutes les valeurs de C sont =0 ou très proche de zéros
    //alors RESOUINF fonctionne
C=D*X-b;
//Affichage de la matrice 5,1 C
disp(C,"Solution: AX-b");
```

<u>Exemple d'un Fonctionnement du Script :</u>

RESOUINF

Initialisation :

```
 affichage de A :

   6.    6.    6.    1.    7.
   10.   9.    6.    7.    0.
   1.    1.    8.    3.    6.
   7.    8.    1.    5.    3.
   4.    9.    6.    10.   3.

 affichage de b :

   6.
   1.
   6.
   7.
   3.
```

Appel de la Fonction puis Affichage de X la matrice retourné par la fonction :

```
FONCTION RESOUINF :

Affichage de la Matrice X solution de RESOUINF sur A et b:

   1.
  -1.
   0.75
   1.45
  -3.6666667
```

Test de comparaison entre AX et b en calculant AX-b :

```
Solution: AX-b

   0.
   0.
   0.
   0.
   4.441D-16

AX-b=0 donc AX = b (ou tres proche de 0)
```

*Les valeurs sont égales ou très proche de 0 (4.441*10^+16)*

Donc le test prouve que le programme Fonctionne

RESOUSUP

On appelle ensuite RESOUSUP avec les mêmes paramètres

```
Affichage de la Matrice Y solution de RESOUSUP sur A et b:

   0.3111111
  -0.3111111
  -0.3
   0.8
   1.
```

On test ensuite comme pour RESOUINF mais cette fois ci en utilisant la matrice inversé triangulaire supérieur

```
Solition : AY-b

  0.
  4.441D-16
  0.
  0.
  0.

AY-b=0 donc AY = b (ou tres proche de 0)

Cela prouve que le programme fonctionne
```

Même constat que pour RESOUINF

Explication du Script Ex6_Script :

INITIALISATION

On initialise aléatoirement la Matrice A et b pour pouvoir utiliser les fonctions et faire des tests puis on affiche ces matrices :

```
//INITIALISATION

//initialisation de la taille n
n=5;


//initialisation aléatoire de la Matrice A de taille 5,5
A=round(10*rand(n,n));

//initialisation aléatoire de la Matrice b de taille 5,1
b=round(10*rand(5,1));


//affichage
disp("affichage de A :")
disp(A);
disp("affichage de b :")
disp(b);
disp("////////////////////////////////////////////////////////////////////////")
```

REDUC

On appelle ensuite REDUC en lui passant A,b,n ,la fonction retourne la matrice triangulaire supérieur et le vecteur on affiche ensuite U et c respèctivement la matrice triangulaire supérieur et le vecteur:

```
//Appelle de la fonction REDUC
//Prend en parametre A Matrice 5,5
//.....b Matrice 5,1
//.....n la taille de la Matrice carré A(5x5/n=5)
//Renvoie une Matrice U 5,5 et une matrice c 5,1
[U,c]=REDUC(A,b,n);
//affichage de la Matrice triangulaire Supérieur et du vecteur c
disp("Matrice triangulaire inversible de A :")
disp(U,"U=");
disp("Vecteur:")
disp(c,"c=");
//Appelle de RESOUSUP sur la solution
//sur la Matrice triangulaire superieur inversible U
//.....et le vecteur b et la taille n
```

GAUSS

On appelle ensuite GAUSS qui prend en paramètre A,b,n

Et renvoie L un vecteur 5,1

```
//Appelle de GAUSS
//prend en parametre A(Matrice 5,5)
//.....,b(vecteur 1,5)
//.....,n(taille 5 de A(n,n ou b(1,n
//retourne L Solution
L=GAUSS(A,b,n);
```

On multiplie alors pour le test A et L puis on les compare à b le vecteur de base

Si les vecteur sont égo la Matrice à est bien La Matrice triangulaire inversible et L le vecteur permettant de l'inverser

```
H=A*L;
disp("COMPARAISON");
disp("b=");
disp(b);
disp("H=");
disp(H);
disp("SI H==b alors GAUSS fonctionne");
```

<u>Exemple d'un Fonctionnement du Script :</u>

Initialisation

```
affichage de A :

  4.   4.   2.   0.   2.
  2.   0.   0.   2.   6.
  9.   5.   8.   5.   3.
  1.   8.   1.   7.   4.
  5.   6.   9.   9.   9.

affichage de b :

  8.
  8.
  3.
  4.
  4.
```

Appelle de REDUC puis affichage de U et c les valeur de retour de cette fonction

```
Matrice triangulaire inversible de A :

U=

  4.    4.    2.     0.           2.
  0.   -2.   -1.     2.           5.
  0.    0.    5.5    1.         -11.5
  0.    0.    0.     14.545455   14.727273
  0.    0.    0.     0.          12.525

Vecteur:

c=

  8.
  4.
 -23.
  3.4545455
  18.975
```

Appelle de GAUSS renvoie L

*Comparaison de b et H(A*L)*

```
COMPARAISON

b=

  8.
  8.
  3.
  4.
  4.

H=

  8.
  8.
  3.
  4.
  4.

SI H==b alors GAUSS fonctionne
```

H==b donc GAUSS fonctionne