Université Abou Bekr Belkaid Tlemcen **2019-2020**
Département de Mathématiques **Examen final**
Module: Analyse numérique 2 **Durée: 1h30**

Exercice 1 : On considère le système $Ax = b$ où

$$A = \begin{pmatrix} 9 & -3 & 3 \\ -3 & 10 & -4 \\ 3 & -4 & 18 \end{pmatrix} \quad \text{et } b = \begin{pmatrix} 3 \\ -4 \\ 18 \end{pmatrix}$$

1) Vérifier que la matrice A est à diagonale strictement dominante. 1
2) Vérifier que la matrice A est définie positive. 1
3) Pour la résolution approchée du système $Ax = b$, on considère la méthode itérative :

$$\begin{cases} x^0 \text{ donné} \\ x^{(k+1)} = Bx^{(k)} + c \end{cases}$$

1

a) Ecrire la méthode itérative de Jacobi. Calculer le rayon spectral de B. 1
En déduire que la méthode de Jacobi converge. 0.5
b) Effectuer 3 itérations de la méthode de Jacobi à partir de $x^{(0)} = (1, 1, 0)^t$. 1.5
c) Ecrire la méthode itérative de Gauss-Seidel. 1.5
d) Montrer en utilisant deux résultats différents que la méthode de Gauss-Seidel converge. 1
e) Effectuer 3 itérations de la méthode de Gauss-Seidel à partir de $x^{(0)} = (1, 1, 0)^t$. 1.5

Exercice 2 : Soit le problème de Cauchy suivant :

$$\begin{cases} y'(t) = -y^2(t) \quad t \in [1,\ 1.6] \\ y(1) = 1 \end{cases}$$

1) Approximer la solution de l'équation différentielle par la méthode d'Euler avec un pas 0.2. 2
2) Approximer la solution de l'équation différentielle par la méthode de Runge Kutta d'ordre2 avec un pas 0.2. 2
3) Comparer la valeur $y(1.6)$ des deux methodes avec la solution exacte. 1

Correction

Exercice 1 : On considère le système $Ax = b$ où

$$A = \begin{pmatrix} 9 & -3 & 3 \\ -3 & 10 & -4 \\ 3 & -4 & 18 \end{pmatrix} \quad \text{et } b = \begin{pmatrix} 3 \\ -4 \\ 18 \end{pmatrix}$$

1) La matrice A est à diagonale strictement dominante si

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}| \qquad \forall i = 1, ..., n \qquad 0.5$$

On a

$$9 > |-3| + |3|, \quad 10 > |-3| + |-4| \quad \text{et} \quad 18 > |3| + |-4| \qquad 0.5$$

alors A est à diagonale strictement dominante.
2) Une matrice A est dite définie positive si $\forall (x_1, x_2, x_3)^t \neq 0 : (x_1, x_2, x_3)^t A (x_1, x_2, x_3) > 0$.
En effet $\forall (x_1, x_2, x_3)^t \neq 0$

0.5

$$\begin{aligned} (x_1, x_2, x_3)^t A (x_1, x_2, x_3) &= 9x_1^2 + 10x_2^2 + 18x_3^2 - 6x_1x_2 + 6x_1x_3 - 8x_2x_3; \\ &= 3(x_1 - x_2)^2 + 3(x_1 + x_3)^2 + 4(x_2 - x_3)^2 + 3x_1^2 + 3x_2^2 + 11x_3^2 > 0. \end{aligned}$$

donc A est définie positive. 0.5
3) a) La méthode de Jacobi s'écrit donc

$$(J) \begin{cases} x^{(0)} \text{ donné} \\ x^{(k+1)} = Bx^{(k)} + c \end{cases}$$

où

$$B = \begin{pmatrix} 0 & 1/3 & -1/3 \\ 3/10 & 0 & 2/5 \\ -1/6 & 2/9 & 0 \end{pmatrix} \quad \text{et} \quad c = \begin{pmatrix} 1/3 \\ -2/5 \\ 1 \end{pmatrix} \qquad 1$$

$$det(B - \lambda I) = \begin{vmatrix} -\lambda & 1/3 & -1/3 \\ 3/10 & -\lambda & 2/5 \\ -1/6 & 2/9 & -\lambda \end{vmatrix} = -\lambda^3 + \frac{11}{45}\lambda - \frac{2}{45}.$$

On a

$$\begin{aligned} det(B - \lambda I) = 0 &\Longrightarrow 45\lambda^3 - 11\lambda + 2 = 0 & 0.5 \\ &\Longrightarrow 45\lambda^3 + 15\lambda^2 + 6\lambda - 15\lambda^2 - 5\lambda + 2 = 0 \\ &\Longrightarrow 3\lambda(15\lambda^2 + 5\lambda - 2) - (15\lambda^2 + 5\lambda - 2) = 0 \\ &\Longrightarrow (3\lambda - 1)(15\lambda^2 + 5\lambda - 2) = 0 \\ &\Longrightarrow \lambda = \frac{1}{3} \text{ ou } \lambda = \frac{-5 - \sqrt{145}}{30} \text{ ou } \lambda = \frac{-5 + \sqrt{145}}{30} & 0.5 \end{aligned}$$

le rayon spectral de B est $\rho(B) = Max\left\{ \left|\frac{1}{3}\right|, \left|\frac{-5 - \sqrt{145}}{30}\right|, \left|\frac{-5 + \sqrt{145}}{30}\right| \right\} = \frac{5 + \sqrt{145}}{30}$
Puisque $\rho(B) < 1$ donc la méthode de Jacobi converge.

0.5

b) La méthode de Jacobi s'écrit : pour $x^{(0)} = (1,1,0)^t$

$$\begin{cases} x_1^{(k+1)} &= \left[-\left(\qquad\qquad -3 \;\; x_2^{(k)} \;\; +3 \;\; x_3^{(k)}\right) \;\; +3\right]/9, \\ x_2^{(k+1)} &= \left[-\left(-3 \;\; x_1^{(k)} \qquad\qquad -4 \;\; x_3^{(k)}\right) \;\; -4\right]/10, \\ x_3^{(k+1)} &= \left[-\left(3 \;\; x_1^{(k)} \;\; -4 \;\; x_2^{(k)} \qquad\qquad \right) \;\; +18\right]/18, \end{cases}$$

0.75

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$
1	0.6667	-0.1	1.0556
2	-0.0519	0.2222	0.8667
3	0.1185	-0.0689	1.0580

0.75

c) La méthode de Gauss-Seidel s'écrit donc

$$(J)\begin{cases} x^{(0)} \text{ donné} \\ x^{(k+1)} = Bx^{(k)} + c \end{cases}$$

où

$$B = \begin{pmatrix} 9 & 0 & 0 \\ -3 & 10 & 0 \\ 3 & -4 & 18 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 3 & -3 \\ 0 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1/3 & -1/3 \\ 0 & 1/10 & 3/10 \\ 0 & -1/30 & 11/90 \end{pmatrix}$$

et

$$c = \begin{pmatrix} 9 & 0 & 0 \\ -3 & 10 & 0 \\ 3 & -4 & 18 \end{pmatrix}^{-1} \begin{pmatrix} 3 \\ -4 \\ 18 \end{pmatrix} = \begin{pmatrix} 1/3 \\ -3/10 \\ 79/90 \end{pmatrix}$$

1.5

d)**Théorème1** : Si A est à diagonale strictement dominante alors les méthodes de Gauss-Seidel et Jacobi sont convergentes. 0.5

Théorème2 : On suppose A symétrique définie positive. Alors la méthode de relaxation converge pour tout $\omega \in]0,2[$ (en particulier la méthode de Gauss-Seidel converge). 0.5

D'après Théorème1 , Théorème2 , 1) et 2) la méthode de Gauss-Seidel converge.

e) La méthode de Gauss-Seidel s'écrit : pour $x^{(0)} = (1,1,0)^t$

$$\begin{cases} x_1^{(k+1)} &= \left[-\left(\qquad\qquad -3 \;\; x_2^{(k)} \;\; +3 \;\; x_3^{(k)}\right) \;\; +3\right]/9, \\ x_2^{(k+1)} &= \left[-\left(-3 \;\; x_1^{(k+1)} \qquad\qquad -4 \;\; x_3^{(k)}\right) \;\; -4\right]/10, \\ x_3^{(k+1)} &= \left[-\left(3 \;\; x_1^{(k+1)} \;\; -4 \;\; x_2^{(k+1)} \qquad\qquad \right) \;\; +18\right]/18, \end{cases}$$

0.75

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$
1	$+0.6667$	-0.2	0.8444
2	-0.0148	-0.0667	0.9876
3	-0.0181	-0.0104	1.0007

0.75

Exercice 2 : Soit le problème de Cauchy

$$\begin{cases} y'(t) = -y^2(t) & t \in [1,\ 1.6] \\ y(1) = 1 \end{cases}$$

1) La méthode d'Euler avec un pas $h = 0.2$.

$$\begin{cases} t_0 = 1 \\ t_{n+1} = t_n + h \\ y_0 = 1 \\ y_{n+1} = y_n - hy_n^2 \end{cases} \qquad 1$$

t_n	y_n
1	1
1.2	0.8
1.4	0.672
1.6	0.5817

1

2) La méthode de Runge Kutta d'ordre2 avec un pas $h = 0.2$.

$$\begin{cases} t_0 = 1 \\ t_{n+1} = t_n + h \\ y_0 = 1 \\ k_1 = -y_n^2 \\ k_2 = -(y_n + hk_1)^2 \\ y_{n+1} = y_n + \dfrac{h}{2}[k_1 + k_2] \end{cases} \Longrightarrow \begin{cases} t_0 = 1 \\ t_{n+1} = t_n + h \\ y_0 = 1 \\ y_{n+1} = -\dfrac{h^3}{2}y_n^4 + h^2y_n^3 - hy_n^2 + y_n \end{cases} \qquad 1$$

t_n	y_n
1	1
1.2	0.836
1.4	0.7176
1.6	0.6283

1

3) la solution exacte : $y(t) = \dfrac{1}{t}$, ce qui donne $y(1.6) = \dfrac{1}{1.6} = 0.625$ 0.5

La méthode d'Euler :
la solution est $y_3 = 0.5817$ avec une erreur $|y(1.6) - y_3| = 0.0433$
La méthode de Runge Kutta d'ordre2
la solution est $y_3 = 0.6283$ avec une erreur $|y(1.6) - y_3| = 0.0033$. 0.5
la meilleur approximation est $y_3 = 0.6283$.

Université Abou Bekr Belkaid Tlemcen — 2019-2020

Département de Mathématiques — Examen final

Module: Analyse numérique 2 — Durée: 1h30

```
import numpy as np
import numpy.linalg as alg
def Jacobi(____________________):  0.5
    n=len(b)
    for k in range(1,Iter+1):
        x_1=np.zeros(n)
        for i in range(n):
            S=0
            for j in range(n):
                if j != i :
                    S=S+A[i,j]*x_0[j]
            x_1[i]= (b[i]-S)/A[i,i]
        x_0= x_1
    return x_1
def Gauss_Seidel(____________________):  0.5
    n=len(b)
    for k in range(1,Iter+1):
        x_1=np.zeros(n)
        for i in range(1,n):
            s1 = np.dot(A[i,   :i], x_1[   :i])
            s2 = np.dot(A[i,i+1: ], x_0[i+1: ])
            x_1[i] = (b[i] - s1 - s2) / A[i, i]
        x_0= x_1
    return x_1
# 1) Vérifiant que la matrice A est à diagonale strictement dominante.
def dominante(A):




                                                    2.5




A = np.array([[9 ,-3,3],[-3,10,-4],[3 ,-4,18]],dtype='float')
b = np.array([3,-4,18],dtype='float')
x_0= np.array([1,1,0],dtype='float')
if __________________________________________________  0.5
#   En déduire que les deux méthodes convergent.
                                                          0.5

#2) la solution la plus proche de la solution exacte.
    sol_j=Jacobi(____________________________)
    sol_g=Gauss_Seidel(___________________________)  0.5
    if ______________________________________________________
        print("la solution la plus proche de la solution exacte est\n", sol_j)
    else:
        print("la solution la plus proche de la solution exacte est\n", sol_g)
else: print("la matrice A n'est pas à diagonale strictement dominante")
```

```python
import numpy as np
import numpy.linalg as alg
def Jacobi(A, b, x_0, Iter):    0.5
    n=len(b)
    for k in range(1,Iter+1):
        x_1=np.zeros(n)
        for i in range(n):
            S=0
            for j in range(n):
                if j != i :
                    S=S+A[i,j]*x_0[j]
            x_1[i]= (b[i]-S)/A[i,i]
        x_0= x_1
        print(x_1)
    return x_1
def Gauss_Seidel(A, b, x_0, Iter):  0.5
    n=len(b)
    for k in range(1,Iter+1):
        x_1=np.zeros(n)
        for i in range(1,n):
            s1 = np.dot(A[i,   :i], x_1[   :i])
            s2 = np.dot(A[i,i+1: ], x_0[i+1: ])
            x_1[i] = (b[i] - s1 - s2) / A[i, i]
        x_0= x_1
    return x_1
# 1) Vérifiant que la matrice A est à diagonale strictement dominante.
def dominante(A):
    for i in range (len(A)):
        S=0
        for j in range(len(A)):
            if j!=i:                          2.5
                S+=abs(A[i,j])
        if abs(A[i,i])<=S:
            M=False
        else: M=True
    return M
A = np.array([[9 ,-3,3],[-3,10,-4],[3 ,-4,18]],dtype='float')
b = np.array([3,-4,18],dtype='float')
x_0= np.array([1,1,0],dtype='float')
if dominante(A)==True:   0.5
#   En déduire que les deux méthodes convergent.
    print("la matrice A est à diagonale strictement dominante")  0.5
    print("alors les méthodes de Gauss-Seidel et Jacobi convergent.")
#2) la solution la plus proche de la solution exacte.
    sol_j=Jacobi(A, b, x_0, Iter=3)             0.5
    sol_g=Gauss_Seidel(A, b, x_0, Iter=3)
    if alg.norm(np.dot(A,sol_j)-b)<=alg.norm(np.dot(A,sol_g)-b):
        print("la solution la plus proche de la solution exacte est\n", sol_j)
    else:
        print("la solution la plus proche de la solution exacte est\n", sol_g)
else: print("la matrice A n'est pas à diagonale strictement dominante")
```