UNIVERSITE CADI AYYAD

Ecole Nationale des Sciences

Appliquées Safi

Module : Mathématique pour l'ingénieur

Devoir libre

Filière : Génie Informatique et Intelligence Artificielle (G.I.I.A.)

Première Année G.I.I.A.

Réalisée par :

Houssam Meryam

Chebbab Aya

Année Universitaire : 2021-2022

Exercice 1 :

On considère l'équation :

$$e^{x} - (x + 5)$$

1. **Déterminer le nombre de la position approximative des solution positives de l'équation :**

Soit $f(x) = e^{x} - (x+5)$ (1)

L'équation (1) $\Leftrightarrow f(x) = 0$

1 – Cherchons les Solutions positifs de l'équation (1)

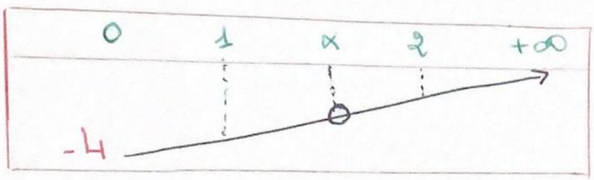
c à d les racines positives de f (Sur $[0;+\infty[$)

On a $f(0) = -4 \neq 0$, on étudie donc la fonction Sur $]0;+\infty[$.

c à d que f est strictement $\nearrow$ Sur $]0;+\infty[$

Donc sur $]0;+\infty[$, $x > 0 \Rightarrow f(x) > f(0) = -4$.

Donc Si f admet une racine Sur $]0;+\infty[$, elle sera unique



On remarque que $f(1) = e - 6 < 0$ et $f(2) = e^{2} - 7 > 0$,
et f est strictement monotone, donc la courbe de f coupe l'axe des abscisses en un seul point Sur $]0,+\infty[$, c'est la seule racine notée $\alpha \in]1,2[$

2.Algorithme de la bissection pour déterminer chacune des racines avec une erreur absolue inférieure à 10^{-7} :

```
Entrée [1]: import math
def bissection(a,b,f,n):
    for i in range(n):
        x_m = (a+b)/2
        if f(a)*f(x_m) < 0:
            b = x_m
        elif f(b)*f(x_m) < 0:
            a = x_m
        elif x_m == 0:
            print("solution exacte.")
            return x_m
        else:
            print("Erreur")
            return None
        print("itération %d : %.8f\t\t%.8f"  % (i, a, b))
    print()
    return (a+b)/2

def f(x):
    return math.exp(x) - (x+5)

print("La solution approché est :", bissection(1,2,f, 24))
```

Résultat :

```
itération 0 : 1.50000000		2.00000000
itération 1 : 1.75000000		2.00000000
itération 2 : 1.87500000		2.00000000
itération 3 : 1.87500000		1.93750000
itération 4 : 1.90625000		1.93750000
itération 5 : 1.92187500		1.93750000
itération 6 : 1.92968750		1.93750000
itération 7 : 1.93359375		1.93750000
itération 8 : 1.93554688		1.93750000
itération 9 : 1.93652344		1.93750000
itération 10 : 1.93652344		1.93701172
itération 11 : 1.93676758		1.93701172
itération 12 : 1.93676758		1.93688965
itération 13 : 1.93682861		1.93688965
itération 14 : 1.93682861		1.93685913
itération 15 : 1.93684387		1.93685913
itération 16 : 1.93684387		1.93685150
itération 17 : 1.93684387		1.93684769
itération 18 : 1.93684578		1.93684769
itération 19 : 1.93684673		1.93684769
itération 20 : 1.93684721		1.93684769
itération 21 : 1.93684721		1.93684745
itération 22 : 1.93684733		1.93684745
itération 23 : 1.93684739		1.93684745

La solution approché est : 1.9368474185466766
```

3 . Déterminons combien d'intégration de la méthode de la bissection pour calculer la racine la plus proche de 1 avec une précision de 10^{-8} :

De la même façon que la question 2,
le nombre d'itérations m dans ce cas vérifie

$$\frac{2-0}{2^m} < 10^{-8} \iff m > \frac{8\ln(10)}{\ln(2)} + 1$$

$$\iff m > 27,5754....$$

C'est le plus petit entier qui vérifie cette condition:
$m = 28$.

Exercice 4 :

On considère l'équation :

$$F(x) = 2x^2 - e^x$$

1.Calculons une approximation de x solution de la fonction par la méthode de Newton:

```
import math

def newton(f,d,x0,N):
    for i in range(N):
        x = x0 - f(x0)/d(x0)
        x0 = x

    return x


def f(x):
    return 2*x**2 - math.exp(x)

# dérivée de f
def d(x):
    return 4*x - math.exp(x)

solution = newton(f, d, 1 , 3)

print("la solution approchée est :", solution)
```

```
la solution approchée est : 1.4879618819282947
```

2.Calculons une approximation de x solution de la fonction par la méthode de Sécante:

```python
import math

def secante(f,a,b,n):
    for i in range(n):
        x = b - ( b - a ) * f(b) / ( f(b) - f(a) )
        a = b
        b = x

    return x

def f(x):
    return 2*x**2 - math.exp(x)

solution = secante(f,1,2,3)

print("la solution approchée avec la méthode de sécante est:", solution)
```

```
la solution approchée avec la méthode de sécante est: 1.4880958266049127
```

3 . Une méthode de point fixe et discuter sa converge :

3- On a $f(x)=0 \iff f(x)+x = x$ (*)

on pose $g(x) = f(x)+x$,

donc, on se ramène à déterminer un point fixe de g

$$g(x) = 2x^2 + x - e^x$$

On prend l'intervalle $[a,b]$, avec $a = \ln(4)$

on a $f(a).f(b) < 0$. et on considère la methode numeriqu

$$\begin{cases} x_{n+1} = g(x_n) \\ x_0 = a = \ln(4) \end{cases}$$

On a $g'(x) = 4x + 1 - e^x$

pour $\ln(4) < x < \ln(5)$, on a $4\ln(4) - 4 < g'(x) < 4\ln(5)-$

Donc $|g'(x)| = g'(x) > 4\ln(4) - 4 > 1$ sur $[a,b]$

Donc cette méthode est divergente.

Exercice 2 :

On considère l'équation :

$$x_0 = \frac{3}{2} \ ; \ x_{n+1} = x_n^2 - 2x_n + 2 = F(x_n)$$

1.Calculons x_2 :

```
def suite(X0,n):
    for i in range(n):
        x=X0**2 - 2*X0 + 2
        X0=x
    return x
print("x2=",suite(3/2,2))
```

```
x2= 1.0625
```

-

2 . La convergence de la limite :

Si (x_n) converge, on note $\ell = \lim\limits_{+\infty} x_n \in \mathbb{R}$

on a $x_{n+1} = F(x_n)$ et F continue, donc ℓ satisfait

$$\ell = F(\ell) \Leftrightarrow \ell^2 - 2\ell + 2 = \ell$$
$$\Leftrightarrow (\ell - 1)(\ell + 2) = 0$$
$$\Leftrightarrow \ell = 1 \text{ ou } \ell = 2$$

la limite est unique, on va montrer dans la question suivante que $\ell = 1$.

3.Montrons que F est contractante :

3- Si $f : [a,b] \to \mathbb{R}$

* f continue sur $[a,b]$ est d.V sur $]a,b[$
* $f([a,b]) \subset [a,b]$
* il existe $M \in]0,1[$, tq

$$|f'(t)| \leqslant M \quad \forall t \in [a,b]$$

Alors f est contractante

On a $F'(x) = 2x - 2$

* $F'(x) \geqslant 0$ pour $x \geqslant 1$
* $F'(x) \leqslant 0$ pour $x \leqslant 1$

	$3/4$		1		$5/4$
F'		$-$	0	$+$	
F	$17/16$	$\searrow$	1	$\nearrow$	$\frac{17}{16}$

Donc $f([3/4, 5/4]) \subset [3/4, 5/4]$

$F'(x) = 2x - 2 \qquad F''(x) = 2 > 0$

d'où F' est $\nearrow$ sur $[3/4, 5/4]$

donc $\frac{3}{4} < x < 5/4$

$$F'\left(\frac{3}{4}\right) < F'(x) < F'(5/4)$$

$$-1/2 < F'(x) < 1/2$$

4.Déduirons que x_n $converge$:

4- On a F est contractante.

et $x_{n+1} = F(x_n)$

donc $(x_n)_{n \geq 0}$ est une Suite de Cauchy

d'où (x_n) convergente.

5 .Calculons L'erreur $e_n = x_n - x^*$ et montrons que la convergence est quadratique :

5- On a

$$e_{n+1} = x_{n+1} - x^* = x_{n+1} - 1$$

$$= x_n^2 - 2x_n + 2 - 1 = x_n^2 - 2x_n + 1$$

$$= (x_n - 1)^2$$

$$|e_{n+1}| = |x_n - 1|^2$$

$$\leq |x_n - 1|^2 = |e_n|^2$$

donc la convergence est quadratique

Exercice 5 :

1 .Faire deux itérations par la méthode de Newton pour approcher la solution du système non linéaire suivant :

$$\begin{cases} x_1^2 - 10x_1 + x_2^2 + 8 \\ x_1x_2^2 + x_1 - 10x_2 + 8 \end{cases}$$

On prend

$$X^0 = (0,0)$$

```python
import numpy as np

Jf = lambda x,y: np.array([[2*x - 10, 2*y],[y**2 + 1, 2*x*y -10]])
f1 = lambda x,y: x**2 - 10*x + y**2 + 8
f2 = lambda x,y: x*(y**2) + x - 10*y + 8

def newton(n):
    x0 = np.zeros(2)
    for i in range(n):
        x = tuple(x0 - (np.linalg.inv(Jf(x0[0],x0[1]))).dot(np.array([f1(x0[0],x0[1]),f2(x0[0],x0[1])])))
        print(f"itération {i+1} = {x}")
        x0 = x
newton(2)
```

```
itération 1 = (0.8, 0.88)
itération 2 = (0.991787221105863, 0.9917117370961642)
```

2.Considérons le système :

$$\begin{cases} x_2 + x_1^2 - x_1 - {}^{1}\!/_{2} \\ x_2 + 5x_1x_2 - x_1^3 \end{cases}$$

- **Itération par la méthode de Newton en utilisant : $X^0 = (1,0)$**

```python
import numpy as np

Jf = lambda x1,x2: np.array([[2*x1 - 1, 1],[5*x2 - 3*(x1**2), 5*x1 +1]])
f1 = lambda x1,x2: x2 + (x1**2) - x1 -(1/2)
f2 = lambda x1,x2: x2 + 5*x1*x2 - (x1**3)

def newton(n,x0):
    for i in range(n):
        x = tuple(x0 - (np.linalg.inv(Jf(x0[0],x0[1]))).dot(np.array([f1(x0[0],x0[1]),f2(x0[0],x0[1])])))
        print(f"itération {i+1} = {x}")
        x0 = x
x0 = np.array([1,0])
newton(2,x0)
```

```
itération 1 = (1.2222222222222223, 0.2777777777777778)
```

- **Itération par la méthode de Newton en utilisant : $X^0(0,-0.2)$**

```python
x0=np.array([0, -0,2])
try :
    newton(1,x0)
except:
    print("\n\tNON ,On ne peut pas prendre   x0=(0, -0,2)")
```

```
	NON ,On ne peut pas prendre   x0=(0, -0,2)
```