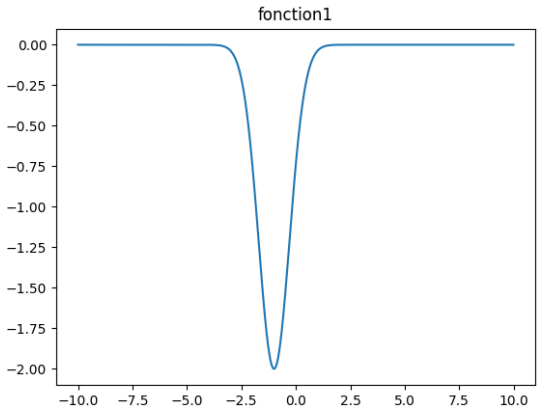
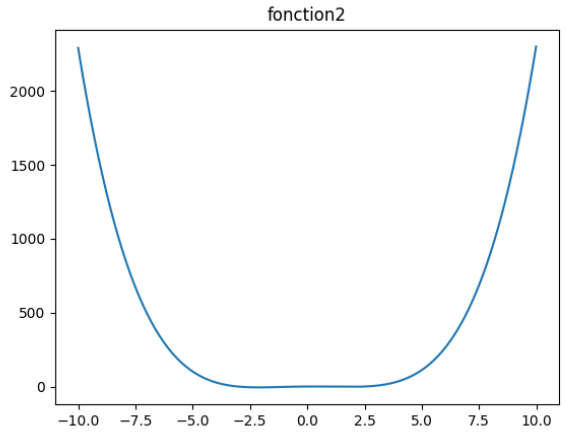
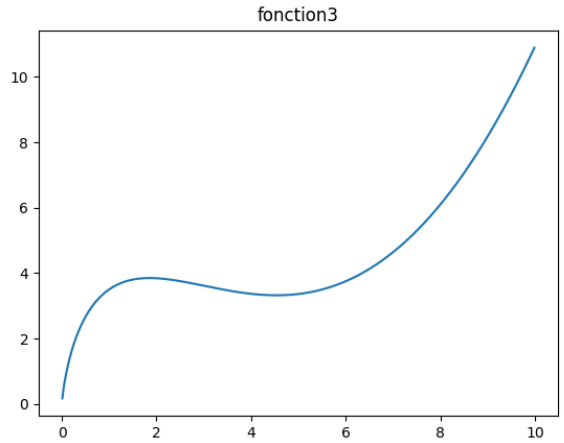
Exercice1 :

1)


fonction1
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


fonction2
2000
1500
1000
500
0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


fonction3
10
8
6
4
2
0
0
2
4
6
8
10

Q :2)

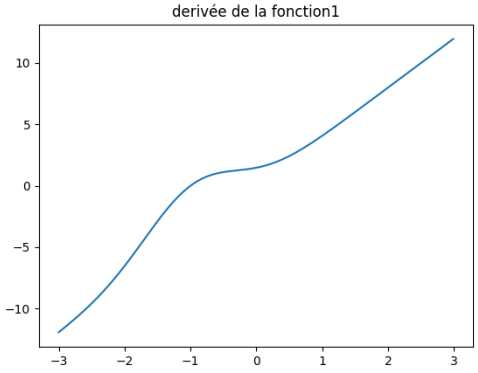
Les fonctions ont pour extrema locaux a vue d'œil :

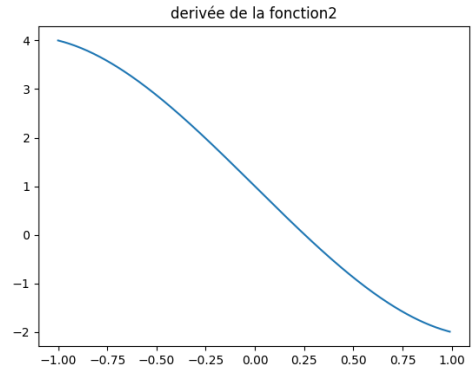
f1->(-1,-2)

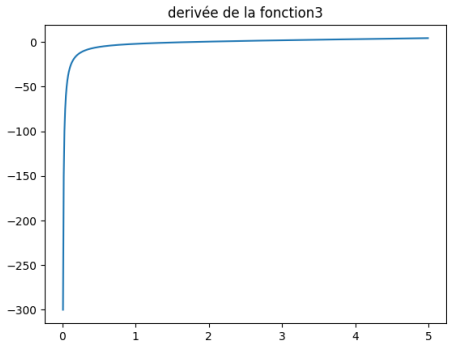
f2->(0,0)

f3->(2,3)

Affichage des dérivées :







On peut voir que notre approximation est plus au moins juste.

Exercice 2 :

Q. 1)

```
#fonction Newton à une variable
def Newton(f,fP,fPP,x,EPS):
    cpt=0
    xk=x
    while(np.abs(fP(xk))>=EPS):
        if (fPP(xk)!=0):
            xkplus1=xk-fP(xk)/fPP(xk)
            xk=xkplus1
            cpt+=1
        else :
            print("la methode de newton echoue")
            exit()
    return (cpt,xk,fP(xk))
```

Q .2)

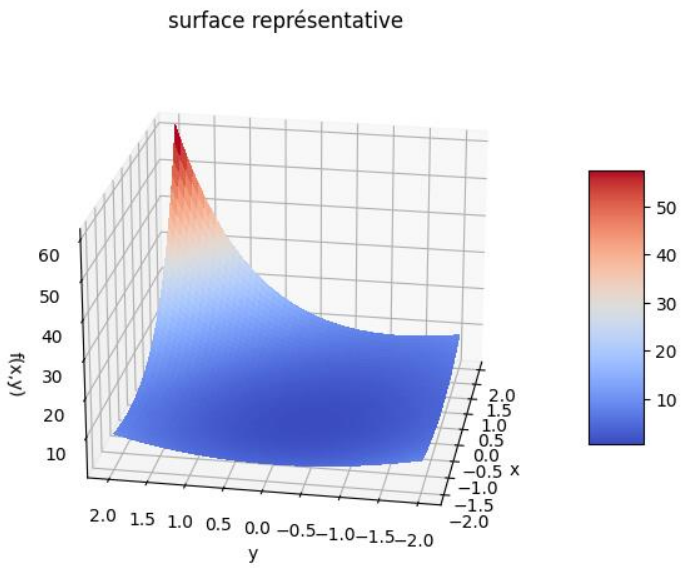
```
(a,b,c)=Newton(f1,fprime1,fseconde1,-1,0.001)
print(a,b,c)
(d,e,f)=Newton(f2,fprime2,fseconde2,0,0.001)
print(d,e,f)
(i,j,k)=Newton(f3,fprime3,fseconde3,1.5,0.001)
print(i,j,k)
```

```
PS C:\Users\lazra\OneDrive\Desktop\MonoS3\LU2IN021\s10-11> python3 ./fonction.py
0 -1 0.0
2 0.2540983606557377 1.2666258409232967e-05
2 1.731958762886598 -0.00018409425625898201
```

Ainsi : Si on éloigne trop la valeur de départ x0 la méthode de newton échoue.

Exercice3 :

Q :1)



On peut apercevoir un optimum global en plus au moins (0,0)

Q :2)

```
#gradient de la fonction f4 en un point (x,y)
def Gradientf4(x,y):
    return np.array([np.exp(x+y)+2*x,np.exp(x+y)+2*y])
```

Q :3)

```
#Hessiennne de la fonction f4 en un point (x,y)
def Hessiennef4(x,y):
    M= [[2+np.exp(x+y),np.exp(x+y)],[np.exp(x+y),2+np.exp(x+y)]]
    return np.array(M)
```

Exercice 4 :

Q :1)

```
def Newton2Var(f,Gradient,Hessienne,x,y,EPS):
    cpt=0
    Xk=[x,y]
    Xkplus1=np.linalg.solve(Hessienne(Xk[0],Xk[1]),-Gradient(Xk[0],Xk[1]))+Xk

    #critere d'arret la distance entre ||Xk+1-Xk|| est inferieur a epislone
    while(Distance(Xkplus1[0],Xkplus1[1],Xk[0],Xk[1])>EPS):
        Xk=Xkplus1
        Xkplus1=np.linalg.solve(Hessienne(Xk[0],Xk[1]),-
Gradient(Xk[0],Xk[1]))+Xk
        cpt+=1
    return (cpt,Xkplus1,f(Xkplus1[0],Xkplus1[1]))
```

Q :2)

```
#On prend le meme valeur pour episole que la question 2.2
(a,b,c)=Newton2Var(f4,Gradientf4,Hessiennef4,4,5,0.001)
print(a,b,c)
```

Retourne :

```
PS C:\Users\lazra\OneDrive\Desktop\MonoS3\LU2IN021\s10-11> python3 ./fonction.py
12 [-0.28357164 -0.28357164] 0.727969046338202
```

12 itérations

Xk*=(-0.28357164 ,-0.28357164)

f4(Xk*)= 0.727969046338202

Exercice 5 :

Q :1)

```
def Gradient2Var(f,Gradient,x,y,EPS,alpha):
    X=[x,y]
    cpt=0
    D=-Gradient(X[0],X[1])
    while(Distance(D[0],D[1],0,0)>EPS):
        X+=alpha*D
        cpt+=1
        D=-Gradient(X[0],X[1])
    return (cpt,X,f(X[0],X[1]))
```

Q :2)

```
(a,b,c)=Gradient2Var(f4,Gradientf4,2,1,0.001,0.4)
print(a,b,c)
```

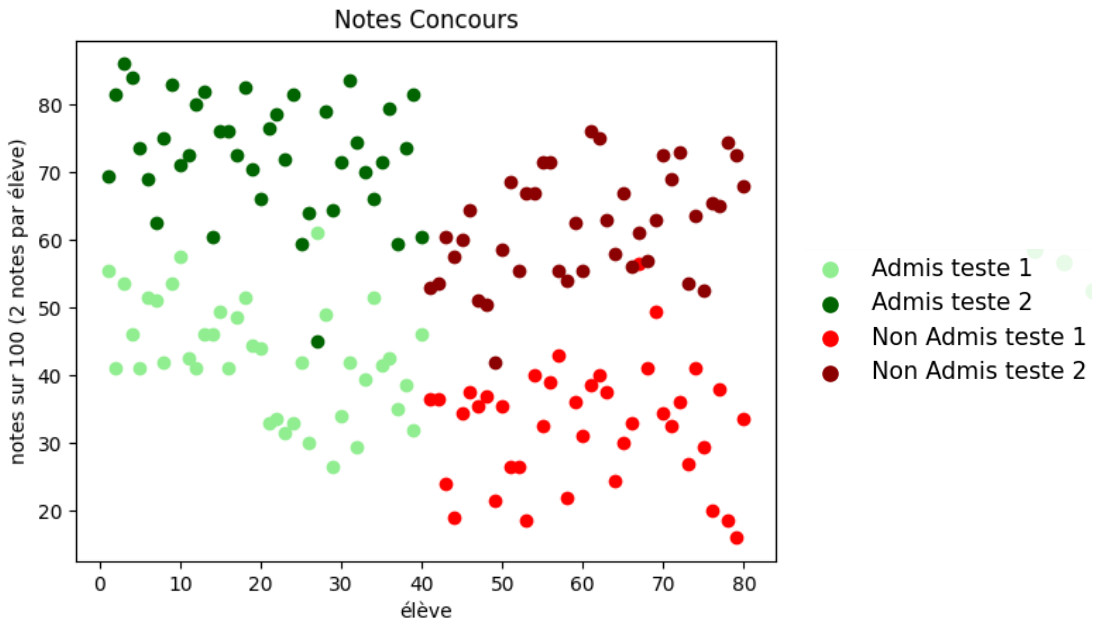
Retourne :

```
PS C:\Users\lazra\OneDrive\Desktop\MonoS3\LU2IN021\s10-11> python3 ./fonction.py
7 [-0.2837359 -0.2837487] 0.7279691376922798
```

Si : alpha >0.5 l'algo du gradient Diverge valeur optimale de alpha entre [0.1 ,0 .5]

Exercice 6 :

Q:1)



Q :2)

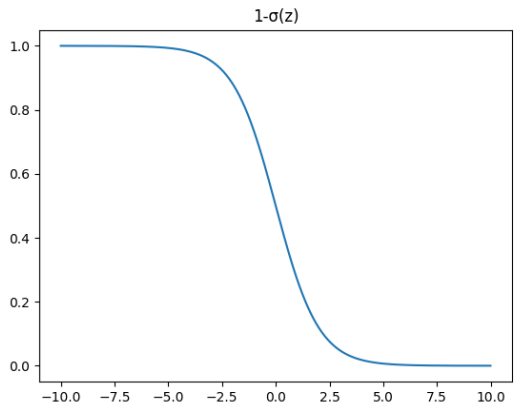
On peut voir à travers ce graphique que les notes du 2nd teste sont bien meilleures que celles du premier. En effet, on peut voir que la meilleure note du premier teste est de 61% tandis que la meilleure note du second teste est de 86%. De plus, on constate également une différence en ce qui concerne les notes minimales : les notes les plus basses du premier et du second teste sont respectivement de 16% et 42%. Globalement le 2nd teste a été mieux réussi.

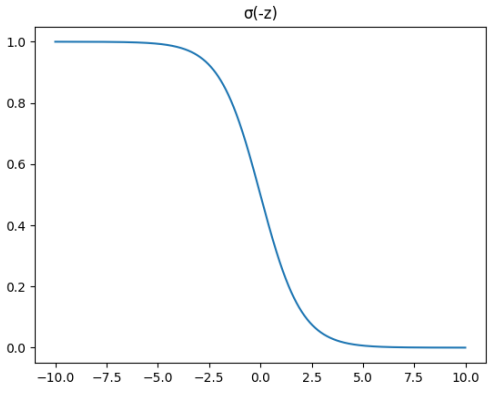
Dans un premier temps on remarque que toute note en dessous de 25% est éliminatoire. De plus, on peut dire qu'il est plus évident de prédire l'admission ou la non-admission au concours lorsque les points représentant les 2 notes sont rapprochés comme par exemple pour l'élève 9 de la liste des non admis soit le 49 sur le graphique. Au contraire, il est plus difficile de se prononcer quant à ce

sujet lorsque la différence entre les 2 notes est plus élevée, par exemple pour le 32ème élève où il est difficile de juger à l'œil nu.

Exercice 7 :

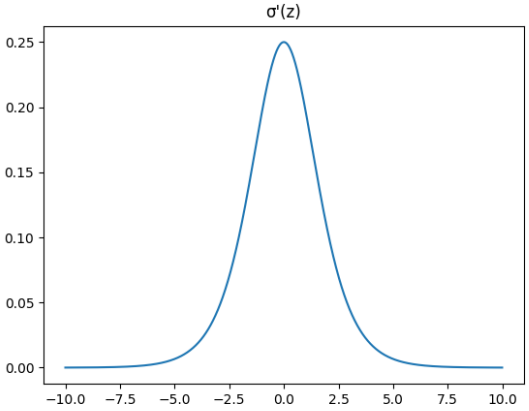
On plot le graph de 1-σ (z) et σ (-z) :

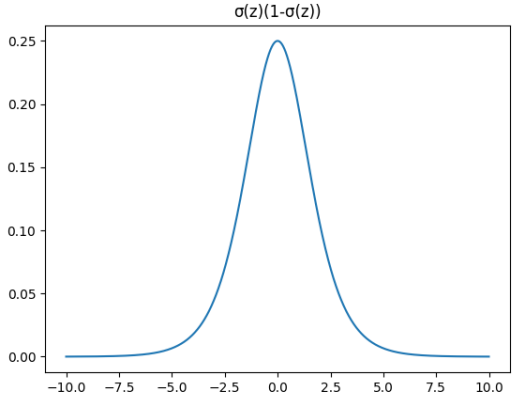




Les graphiques sont identiques, cela nous montre l'équivalence dans la relation ainsi 1-σ (z)=σ (-z).

On plot le graph de σ'(z) et σ(z)(1-σ(z)) :





Les graphiques sont identiques, cela nous montre l'équivalence dans la relation ainsi σ'(z)=σ(z)(1-σ(z)).

Exercice 8 :

8.2 $f(\theta) = \sum_{(x_1,x_2)\in X_a} \ln(1+e^{-f(\theta)}) + \sum_{(x_1,x_2)\in \bar{X}_a} \ln(1+e^{f(\theta)})$

8.4 si on derive par rapport a θ_0

- $\frac{\partial f}{\partial(\theta_0)}(\theta) = \sum_{(x_1,x_2)\in X_a} \frac{-e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2)}}{1+e^{-f(x)}} + \sum_{(x_1,x_2)\in \bar{X}_a} \frac{e^{f(\theta)}}{1+e^{f(\theta)}}$

$\frac{\partial f}{\partial(\theta_i)}(\theta) = \sum_{(x_1,x_2)\in X_a} -\frac{1}{1+e^{-f(\theta)}} \cdot e^{-[\theta_0+\theta_1 x_1+\theta_2 x_2]} + \sum \frac{1}{1+e^{f(\theta)}}$ folle $f(\theta)$

$$\boxed{\frac{\partial f}{\partial \theta_i}(\theta) = \sum_{(x_1,x_2)\in X_a} -\sigma(-f(\theta)) \cdot \frac{\partial f}{\partial \theta_0} + \sum_{(x_1,x_2)\in \bar{X}_a} \sigma(-f(\theta)) \cdot \frac{\partial f}{\partial \theta_i}(\theta)}$$

Ainsi on remarque cette forme est generique due a la derivé de ~~that~~ exponentiel,

8.3 Soit nous avons derivée theta; nous savons que ~~theta~~ $\frac{\partial f}{\partial i \partial j} = \frac{\partial F}{\partial j \partial i}$. Ainsi la forme ~~de~~ trouver precedement est une bonne partie.

- $\frac{\partial f}{\partial(\theta_i)}(\theta) = \underbrace{\sum_{(x_1,x_2)} -\frac{1}{1+e^{-f(\theta)}} \cdot f'(\theta) \cdot e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2)}}_{1} + \underbrace{\sum \frac{1}{1+e^{f(\theta)}} f'(\theta) \cdot e^{f(\theta)}}_{2}$

Procedons en 2 partie,

~~Ainsi~~ ~~derive de uv : u'v + uv'~~

$\frac{\partial F}{\partial \theta_i \partial \theta_j}(\theta) = \sum_{(x_1,x_2)} \frac{1}{1+e^{-f(\theta)}} \cdot \frac{\partial f}{\partial \theta_i} \cdot \frac{\partial f}{\partial j} \cdot \frac{1}{(1+e^{-f(\theta)})^2}$

$(-\cdot-=+)$ const

partie 1 : suite

$$\frac{\partial f}{\partial i}(\theta) = \sum_{(x_i, x_j) \in x_a} -\frac{1}{1+e^{f(\theta)}} \cdot \underbrace{\frac{\partial f}{\partial \theta_i}}_{\text{constant}}$$

$$\frac{\partial f}{\partial \theta_i \partial \theta_j} = 0 \quad \text{done}$$

Ainsi

$$\frac{\partial F}{\partial i\, \partial j}(\theta) = -\frac{-(1+e^{-f(\theta)})'}{(1+e^{-f(\theta)})^2} \cdot \frac{\partial f}{\partial \theta_i}$$

$$= (1+e^{-f(\theta)})' \cdot \frac{1}{(1+e^{-f(\theta)})^2} \cdot \frac{\partial f}{\partial \theta_i}$$

$$= (1+\exp(f(\theta)))' \cdot \sigma(f(\theta))^2 \cdot \frac{\partial f}{\partial \theta_i}$$

$$= \left(\frac{-\partial f'(\theta) \cdot e^{-f(\theta)}}{\partial(\theta_j)}\right) \cdot \sigma(f(\theta))^2 \cdot \frac{\partial f}{\partial \theta_i}$$

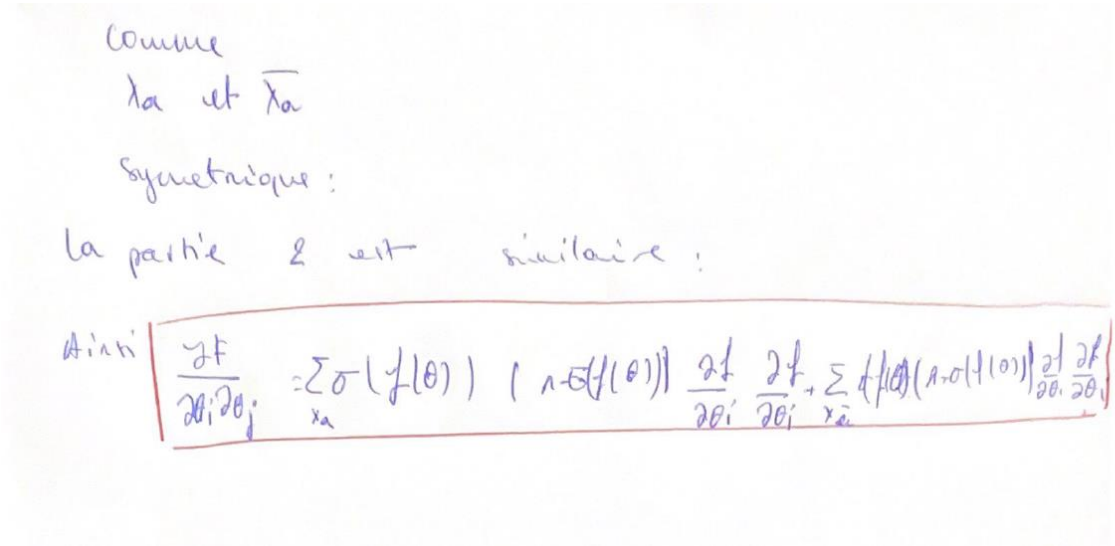
$$= \left(1 - \frac{\partial f(\theta)}{\partial \theta_i} \cdot \frac{1}{e^{f(\theta)}}\right) \sigma(f(\theta))^2 \cdot \frac{\partial f}{\partial \theta_i}$$

$$= \sigma(f(\theta)) \frac{\partial f}{\partial \theta_i} \frac{\partial f}{\partial \theta_j} - \sigma(f(\theta))^2 \frac{\partial f}{\partial \theta_i} \frac{\partial f}{\partial \theta_j}$$

$$= (\sigma(f(\theta)) - \sigma(f(\theta))^2) \frac{\partial f}{\partial \theta_i} \cdot \frac{\partial f}{\partial \theta_j}$$

$$\boxed{\frac{\partial F}{\partial \theta_i \partial \theta_j} = \sigma(f(\theta))(1 - \sigma(f(\theta)) \frac{\partial f}{\partial \theta_i} \cdot \frac{\partial f}{\partial \theta_j}}$$

partie (2)



Exercice 9 :

Le gradient correspond a la dérivée première des trois Thêta .

Quant a la hessienne elle correspond à :

$$H(f) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial x \partial z} & \frac{\partial^2 f}{\partial y \partial z} & \frac{\partial^2 f}{\partial z^2} \end{pmatrix}$$

```python
#Exercice 9:

def f(X,THETA):
    (x1,x2)=X
    (a,b,c)=THETA
    return a+b*x1+c*x2

def fprime(X,i):
    X=(x,y)
    if (i<0 and i >2):
        print("Usage : fprime(Tuple[float,float], int i) avec 0<=i<=2")
        exit()
    elif (i==0):
        return 1
    elif(i==1):
        return x
    else:
        return y

def F(THETA):
    p1=0
```

```
    p2=0
    for i in range(0,len(x)):
        X=(x[i],y[i])
        X1=(x1[i],y1[1])
        p1+=np.log(1+np.exp(-f(X,THETA)))
        p2+=np.log(1+np.exp(-f(X1,THETA)))
    return p1+p2


def Fprime(THETA,i):
    if i>2 or i<0:
        print("Usage : (Fprime(Tuple[float,float,float], int i) avec 0<=i<=2")
        exit()
    else:
        cpt1=0
        cpt2=0
        if (i==0):
            for i in range(0,len(x)):
                X=(float(x[i]),float(y[i]))#notes d'un eleve admis
                X1=(float(x1[i]),float(y1[i]))#notes d'un eleve admis
                cpt1+=-sigmoide(-f(X,THETA))*fprime(X,0)
                cpt2+=sigmoide(f(X1,THETA))*fprime(X1,0)
            return cpt1+cpt2
        elif(i==1):
            for i in range(0,len(x)):
                X=(float(x[i]),float(y[i]))#notes d'un eleve admis
                X1=(float(x1[i]),float(y1[i]))#notes d'un eleve admis
                cpt1+=-sigmoide(-f(X,THETA))*fprime(X,1)
                cpt2+=sigmoide(f(X1,THETA))*fprime(X1,1)
            return cpt1+cpt2
        else :
            for i in range(0,len(x)):
                X=(float(x[i]),float(y[i]))#notes d'un eleve admis
                X1=(float(x1[i]),float(y1[i]))#notes d'un eleve admis
                cpt1+=-sigmoide(-f(X,THETA))*fprime(X,2)
                cpt2+=sigmoide(f(X1,THETA))*fprime(X1,2)
            return cpt1+cpt2

def Fseconde(THETA,i,j):
    """derivée partielles seconde de F avec i et j les element a derivérée"""
    cpt1=0
    cpt2=0
    for p in range(0,len(x)):
        X=(float(x[p]),float(y[p]))#notes d'un eleve admis
        X1=(float(x1[p]),float(y1[p]))#notes d'un eleve admis
        cpt2+=sigmoide(f(X1,THETA))*(1-
sigmoide(f(X1,THETA)))*fprime(THETA,i)*fprime(THETA,j)
        cpt1+=sigmoide(f(X,THETA))*(1-
sigmoide(f(X,THETA)))*fprime(THETA,i)*fprime(THETA,j)
```

```
    return cpt1+cpt2


def GradientF(THETA):
    return np.array([Fprime(THETA,0),Fprime(THETA,1),Fprime(THETA,2)])


def HessienneF(THETA):

Hess=[[Fseconde(THETA,0,0),Fseconde(THETA,0,1),Fseconde(THETA,0,2)],[Fseconde(
THETA,0,1),Fseconde(THETA,1,1),Fseconde(THETA,1,2)],[Fseconde(THETA,0,2),Fseco
nde(THETA,1,2),Fseconde(THETA,2,2)]]
    return np.array(Hess)

def Distance3Var(X):
    X=(a,b,c)
    return np.sqrt(a**2+b**2+c**2)

def Newton3Var(f,Gradient,Hessienne,X,EPS):
    cpt=1
    Xk=X
    Xkplus1=np.linalg.solve(HessienneF(Xk),-G)+Xk

    #critere d'arret la distance entre ||Xk+1-Xk|| est inferieur a epislone
    while(Distance(XKplus1)>EPS):
        Xk=Xkplus1
        Xkplus1=np.linalg.solve(Hessienne(Xk),-Gradient(Xk))+Xk
        cpt+=1
    return (cpt,Xkplus1,F(Xkplus1))
```