FARAOUN Hichem

TP01

Optimisation numérique

I. Page 1

1. Déclarer la fonction $f(x)$

```
syms f(x) x ;
f(x) = (1/2) * (x^2) - sin(x);
```

2. Comparaison des résultats

```
f(1)
vpa(f(1), 2)
vpa(f(1), 3)
vpa(f(1), 5)
```

Ici on calcule f (1) et le paramètre (2,3, 5 …) signifie le nombre de chiffres après la virgule

3. Les commandes :

gradient(f(x))	Calcul de Gradient de f
diff(f(x))	Aussi calcul de gradient de f
taylor(f(x))	Développement de Taylor d'ordre 5
taylor(f(x),x,pi/2,'Order',3)	Développement de Taylor d'ordre 3

II. Page 2

1.

```
syms x1 x2 x3 f(x1, x2, x3) alpha
f(x1, x2, x3)=(x1-4)^4+ (x2-3)^2+ 4*(x3+5)^4

x=[4; 2; -1];
g=gradient(f(x1, x2, x3));
grad=vpa(subs(g, [x1, x2, x3], x.'));

    a=0.004200;
    phi=subs(f(x1, x2, x3), [x1;x2;x3],x-alpha*grad)
    Q_1=vpa(subs(diff(phi),alpha,a));
    Q_2=vpa(subs(diff(diff(phi)),alpha,a));
    a=a-Q_1/Q_2
x=x-a*grad
```

- Ce script fait la méthode du Steepest Descent une seule itération
- Déclaration des variables et fonction
- Calcul du gradient
- Calcul du gradient avec x= [4 ;2 ; -1]
- Recherche du pas alpha
- Mis à jour des nouveaux x

III. Page 3

1. Résolution par méthode de Newton

```
>> vpa(xk)

ans =

0.73908513321516064176525915477955
```

2. Résolution par méthode de Sécante

```
>> vpa(xkn)

ans =

0.0039951196262432762406082215641578
```

3. Résolution par méthode de Steepest Descent

```
x =

                                  4
  2.0079368402748558893109003451027
 -5.0636622207262153271809766925763
```

```
a =

0.0039684201374279446554501725513441
```

4. Résolution par méthode du gradient conjugué

```
x =

 0.83333333333333333333333333333333
                                  0
 0.27777777777777777777777777777778
```