Optim-Mauritius-2023-CS3

January 11, 2023

In [1]:
```
# pattern search method for Rosenbroxk
import math
import numpy as np
```

In [2]:
```
def Pattern_search(f,X0,Niter):
    n=len(X0)
    alpha=1
    rho=0.5
    tau=1.5
    D=np.concatenate([np.eye(n),-np.eye(n)])  # set of directions
    n=2*n
    Xk=X0
    for k in range(Niter):
        i=0
        while(i<=n-1 and f(Xk+alpha*D[i,:])>=f(Xk)):
            i=i+1
        if (i==n):#echec
            alpha=rho*alpha
        else:
            Xk=Xk+alpha*D[i,:]
            alpha=alpha*tau
        list_Xk[k] = Xk
    return Xk
```

In [7]:
```
d=2
def J(v):
    x = v[0]
    y = v[1]
    return 100*(y-x**2)**2+(x-1)**2
```

In [8]:
```
N=3000
X0=np.array([-1.4,0.4])
list_Xk = [0]*N
Pattern_search(J,X0,N)
```

Out[8]:
```
array([ 0.98882362,  0.97774581])
```

In [9]:
```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```python
def Rosenbrock(x,y):
    return 100*(y-x**2)**2+(x-1)**2

x, y = np.meshgrid(np.linspace(-0.5,1.5, 200), np.linspace(-0.5,1.5, 200))
z = Rosenbrock(x,y)
graphe = plt.contour(x,y,z,[1, 4,20,100])

xk_1 = [list_Xk[k][0] for k in range(N)]
xk_2 = [list_Xk[k][1] for k in range(N)]
plt.plot(xk_1, xk_2, "b:o") #x_k pour methode pattern search

plt.clabel(graphe,inline=1,fontsize=10,fmt='%3.2f')
plt.title("Courbe de niveaux de Rosenbrock avec les points de la méthode pattern search
plt.show()
```