In [3]:

```
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt
```

using gradient descent

In [2]:

```
neta = 0.03
```

In [4]:

```
x = sp.Symbol('x')
y = sp.Symbol('y')
```

In [6]:

```
z = (x*sp.sin(20*y) + y*sp.sin(20*x)   )**2 * sp.cosh(sp.sin(10*x)*x) + (x*sp.cos(1
```

In [7]:

```
zprimex = z.diff(x)
zprimey = z.diff(y)
```

In [8]:

```
xi, yi = 0.5, 0.5
```

In [9]:

```
for i in range(1000):
    xi = xi - neta*zprimex.evalf(subs = {x: xi, y:yi})
    yi = yi - neta*zprimey.evalf(subs = {x: xi, y:yi})

    if i%1 == 0:
        print(xi, yi)
```

```
0.222295626983961 0.363266754799910
0.250724756745240 0.400736155416072
0.340511264908313 0.439490071965773
0.128068697103075 0.462226474478785
0.211310105304693 0.376137376213196
0.224209627859331 0.401349903956004
0.266794269429828 0.426491680215172
0.379794005720417 0.567171652550629
0.142457120722110 0.547564206711736
0.150440282282577 0.551481396076900
0.103787232368807 0.525527072160283
0.182544128890834 0.496283179764164
0.0413956909106244 0.497933813562409
-0.141176848163950 0.531370620401154
-0.156694834772013 0.540414012750304
-0.0657058387655617 0.521334889380726
0.0484862368723937 0.517112503482424
-0.126448072666406 0.547451645064764
-0.214643459127333 0.513099175607089
-0.0966415674649478 0.524002376449504
```

using simulated annealing

In [75]:

```
thetals = [ [0.5, 0.5]]
T = 1
```

In [11]:

```
def h(a):
    x = a[0]
    y = a[1]
    flag =    (x*np.sin(20*y) + y*np.sin(20*x)    )**2 * np.cosh(np.sin(10*x)*x) + (
    return 1.0/flag
```

In [12]:

```
def box_muller():
    a = np.zeros(2)
    u1 = np.random.uniform(size=1)
    u2 = np.random.uniform(size=1)
    a[0] = np.sqrt(-2*np.log(u1))* np.cos(2*np.pi*u2)
    a[1] = np.sqrt(-2*np.log(u1))* np.sin(2*np.pi*u2)
    return a
```

In [76]:

```
for theta in thetals:
    for i in range(2000):
        old = theta
        G = theta + box_muller()
        u = np.random.uniform(size = 1)
        delh = h(G) - h(theta)
        if u>0 and u<min(1, np.exp(delh/T)):
            theta = G

    print(theta)
```

```
[0.0004282  0.01021434]
```