DEEP LEARNING FOR PHYSICAL SYSTEMS ME504
ASSIGNMENT 1

Name: Shubha Tanaya
Entry no: 2021MEB1325

Q1. You are required to code the program which can solve the unconstrained optimization using the gradient descent method. Some guidelines regarding the program is:

- **The program should have three different functions.**
- **The learning rate should be allowed to change by the user.**
- **The program should give the optimal point and optimal value of function as output.**
- **The program should also generate the plot of function values with iterations.**
- **For two variable case, plot the variables history in the contour plot.**
- **You can test the program for the following functions and one function of your choice.**

$$f(x) = (x-2)^2$$

$$g(x,y) = 5x^2 - 30x + y^2 - 8y + 4xy + 34$$

Answer:

➔ 1st function: gdm(learning_rate, iterations, initial_x, initial_y)
2nd function: function(x, y)
3rd function:def partial_derivative_x(x, y, epsilon=1e-6)

➔ The program takes learning rate, number of iterations and initial position ie(xo,yo) as input from the user.

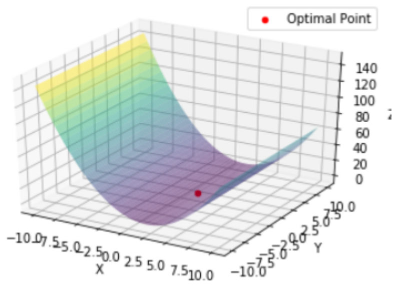
➔ The program plots the function value f(x,y) vs number of iterations. It also plots how the value of x and y changes from the initial values to the optimal values as the number of iterations progresses.

➔ Testing the program for the following functions:

◆ f(x) = (x-2)**2

From Analytical solution:

Optimal point: x=2



```
Optimal Point: [1.99999998 0.        ]
Optimal Value: 3.239791986321692e-16
```

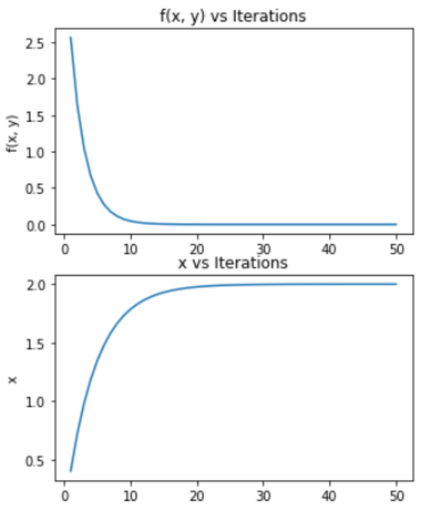
Optimal value is: 0

From graph plotting:

From the result of the program:

Enter the learning rate: 0.1
Enter the number of iterations: 50
Enter the initial x point: 0

Optimal point: x = 1.999970955053256
Optimal value is: 8.436089313582581e-10

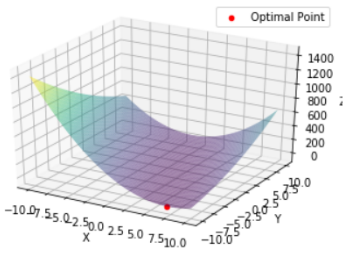


◆ **g(x,y)=5x**2 -30x +y**2 -8y +4xy +34**

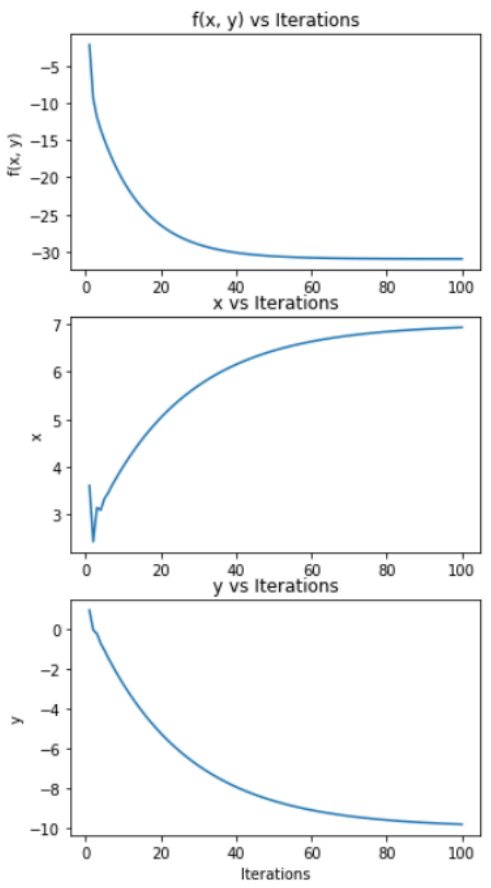
From Analytical solution:

Solving partial differential equations-
fx=10x+4y-30; fy=4x+2y-8
Optimal point is (7,-10)
Local minimum of f(7,-10)=−31

From graph plotting:



```
Optimal Point: [ 6.99999943 -9.99999955]
Optimal Value: -30.999999999999204
```



From the result of the Program:

Enter the learning rate: 0.12
Enter the number of iterations: 100
Enter the initial x point: 0

Enter the initial y point: 0

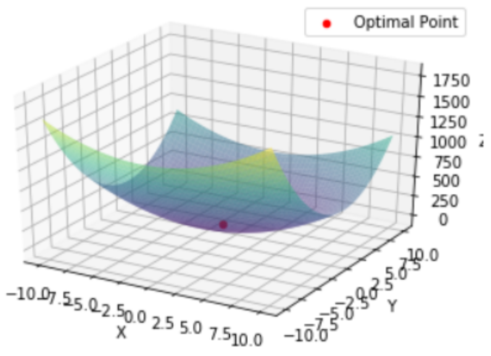
Optimal point: x = 6.931945131185557, y = -9.835702149914027
Optimal value is: -30.994573965146742

- **g(x,y)=4x**2 + 9y**2 +8x - 36y + 24**

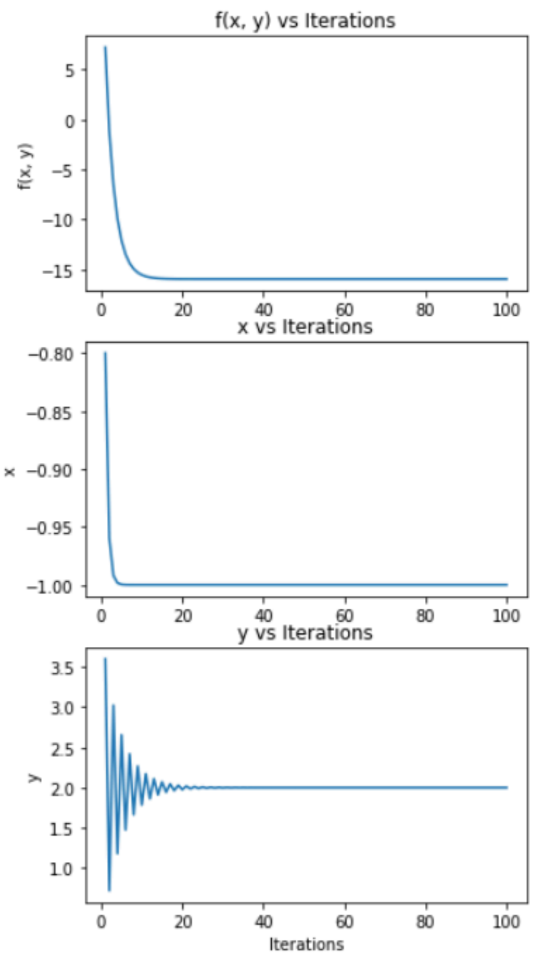
From Analytical solution:

Optimal point is (-1,2)
Local minimum of f(-1,2)=−16

From Graph Plotting:



```
Optimal Point: [-1.00000013  1.99999998]
Optimal Value: -15.999999999999936
```



From the result of the Program:

Enter the learning rate: 0.1
Enter the number of iterations: 100
Enter the initial x point: 0
Enter the initial y point: 0

Optimal point: x = -1.0000005001842283, y = 1.9999994982811131
Optimal value is: -15.999999999996732

➔ In Q1., you can take any function and run the case with three different learning rates and discuss the plots generated in the report.

For the function : 5*x**2 + y**2 -30*x - 8*y + 34 +4*x*y Optimal point is (7,-10)
Local minimum of $f(7,-10)=-3$. Let us vary the learning rate and plot the corresponding graphs, keeping the number of iterations 50 and initial point (0,0).

```
 Optimal point : x = 6.4428914286907, y = -8.655022088461806
Optimal value is: -30.636379507301342
```

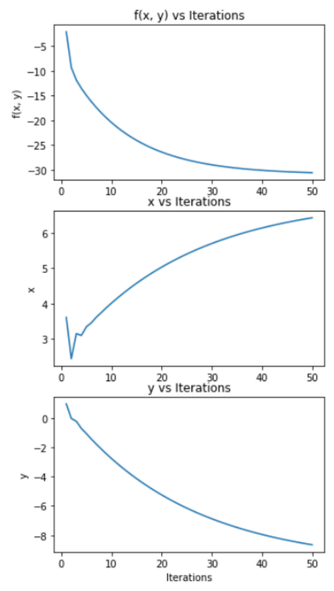


FIG 1

```
 Optimal point : x = 3.1545463867233003, y = -0.7302978666623439
Optimal value is: -13.719893150449273
```

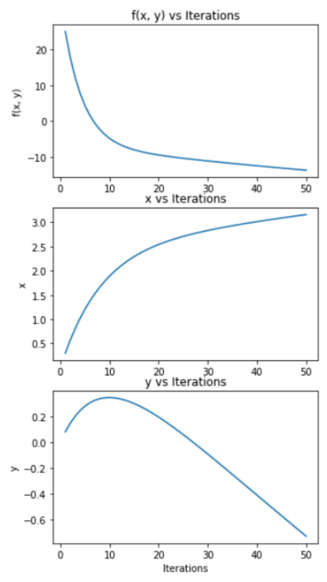


FIG 2

```
 Optimal point : x = -8391293875.925899, y = 27361417816.87654
Optimal value is: 1.823254584896705e+20
```

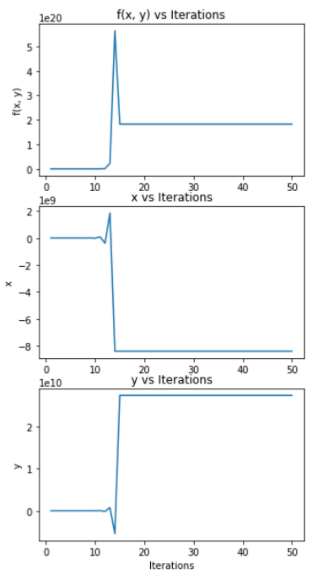


FIG 3

Learning rate: in Fig 1, 2 and 3 are 0.12, 0.005 and 0.5 respectively.

OBSERVATIONS:

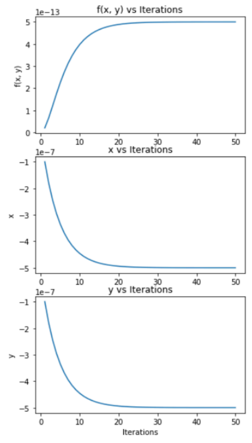
If the learning rate is considerably lower, all the local variations undulations will be captured. It will give us information about the local minima and will fail to capture the overall nature of the graph itself. That is convergence might be very slow, and the algorithm might get stuck in local minima.This will lead to misleading interpretations. Also, the model might underfit the data, as it might take too many small steps and fail to explore the entire function value.

Also if the learning rate is considerably high then we are taking larger steps towards the minima of the cost function; this implies we might miss or overshoot the optimal point and the function may not converge. Also, the optimization process might oscillate around the minimum or diverge entirely, which leads to the instability of the model.

Q2. Use the above developed program to understand the gradient descent method better. In order to do so consider a function, f(x, y) = x**2 + a*y**2 with the value of the a changing from one to 1000 . Play with the learning rate and value of the a to observe the effect of these on the convergence on the optimization iterations.

a=1

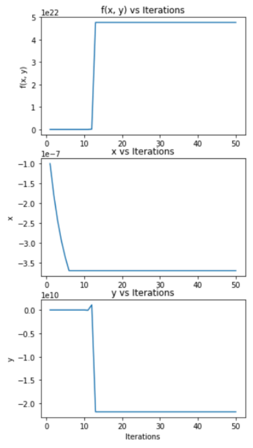
```
 Optimal point : x = -4.999928637615364e-07, y = -4.999928637615364e-07
Optimal value is: 4.999857276249246e-13
```



Enter the learning rate: 0.1
Enter the number of iterations: 50
Enter the initial x point: 0
Enter the initial y point: 0

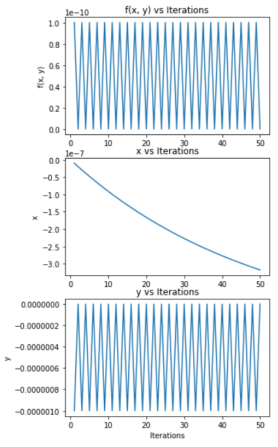
a=100

```
 Optimal point : x = -3.702677462792161e-07, y = -21831021988.317497
Optimal value is: 4.76593521054402e+22
```



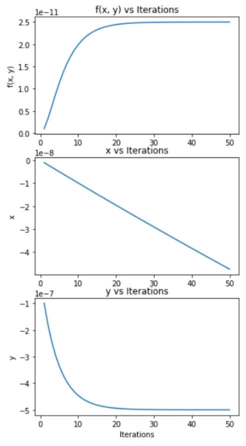
Enter the learning rate: 0.1
Enter the number of iterations: 50
Enter the initial x point: 0
Enter the initial y point: 0

```
 Optimal point : x = -3.1791515995644133e-07, y = 2.117582368135751e-22
Optimal value is: 1.0107004893012967e-13
```



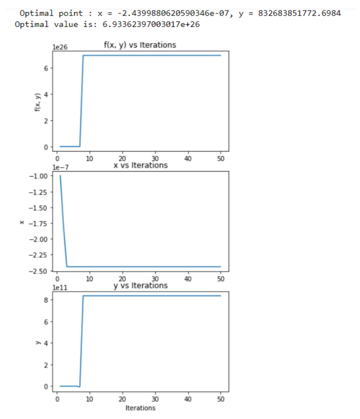
Enter the learning rate: 0.01
Enter the number of iterations: 50
Enter the initial x point: 0
Enter the initial y point: 0

```
 Optimal point : x = -4.7626590997982115e-08, y = -4.999928637615364e-07
Optimal value is: 2.5001554673416322e-11
```

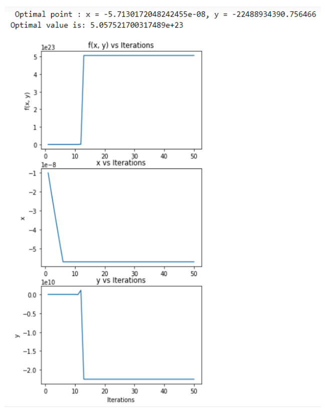


Enter the learning rate: 0.001
Enter the number of iterations: 50
Enter the initial x point: 0
Enter the initial y point: 0

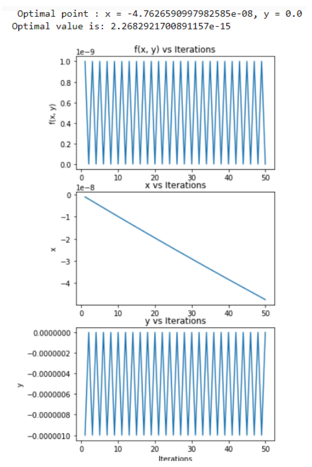
For a=1000
Enter the number of iterations: 50
Enter the initial x point: 0
Enter the initial y point: 0



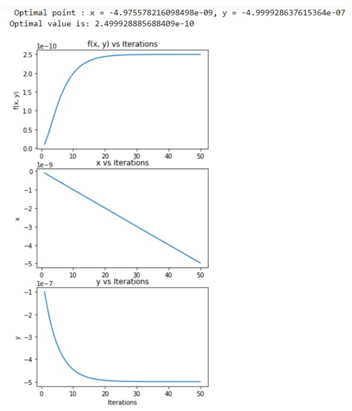
```
 Optimal point : x = -2.4399880620590346e-07, y = 832683851772.6984
Optimal value is: 6.93362397003017e+26
```



```
 Optimal point : x = -5.7130172048242455e-08, y = -22488934390.756466
Optimal value is: 5.057521700317489e+23
```



```
 Optimal point : x = -4.7626590997982585e-08, y = 0.0
Optimal value is: 2.2682921700891157e-15
```



```
 Optimal point : x = -4.975578216098498e-09, y = -4.999928637615364e-07
Optimal value is: 2.499928885688409e-10
```

Learning rate =0.1, 0.01, 0.001, 0.0001 from left to right

It is given that “ a” changes from 1 to 1000; as “a ” keeps increasing the curvature along the y-axis gets much steeper compared to the x-axis. This implies that the gradient descent method will converge much slower along the y-axis compared to the x-axis.
For smaller values of “a” the step size requirements taken by the gradient descent method would be almost equal because the curvature is the same along both axes.
But for higher values of “a” gradient descent should take much smaller step sizes along the y-axis compared to the x-axis to avoid overshooting the minimum.