# COSC 522 HW 4

Yangsong Gu

10/21/2021

**Problem 1: (35) On Gradient Descent (GD). Find the global minimum of f(t) = 50 * sin(t) + t2 over -10<=t<=10. This problem intends to give you a hands-on experience on how gradient descent works and how it can get trapped at the local minima**

a) **(5) Plot this function. Visualize the multiple local minima and the global minimum.**

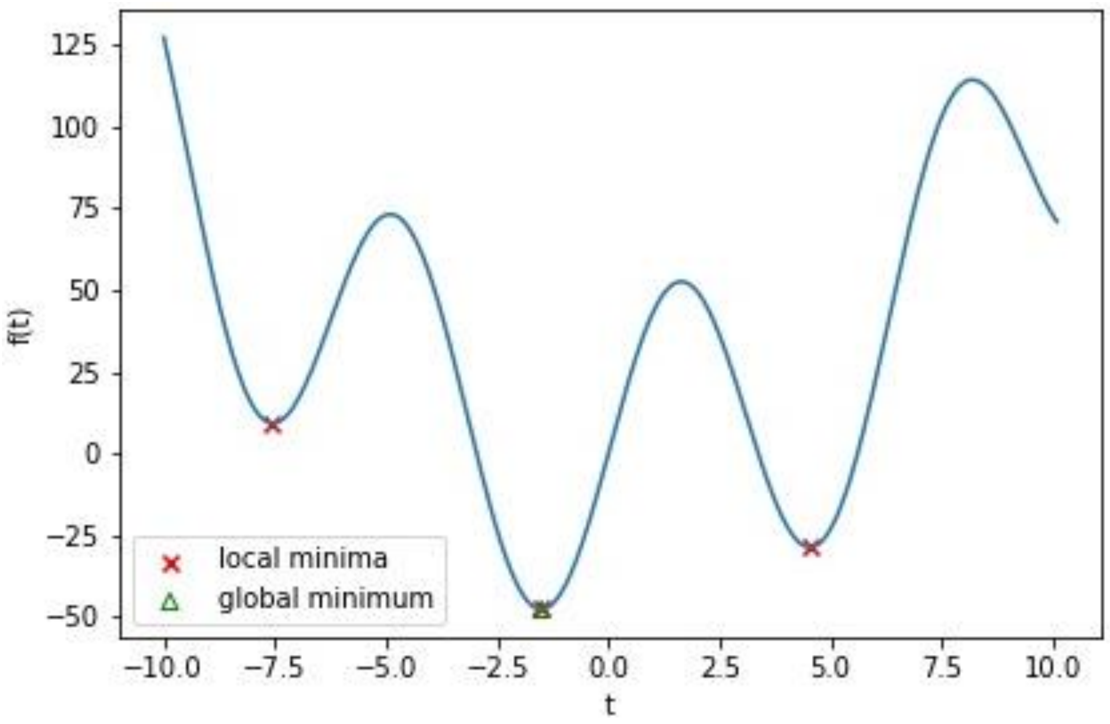

Figure 1. local minima and global minimum.

Figure 1 demonstrates two local minima and 1 global minimum when t varies between -10 and 10. By visual inspection, the local minima are obtained when t = -7.55 and t = 4.53, the global minimum is obtained when t = -1.51.

b) **(30) Implement gradient descent in Python to find the local minimum.**
   i) **(10) Pick a starting point at t=7. What's the minimum? Show the convergence path. Experimenting with different learning rate**

Figure 2-5 display the convergence path of initial t = 7 at different learning rate (e.g., 0.001, 0.01, 0.02, 0.03). The accuracy $\epsilon$ is set as 0.0001. The minimum is obtained when t = 4.53. Comparing between figures, we found that the number of iterations reduced as the learning rate increases, at the same time, the "learner" is prone to miss the local minimum (see figure 5).

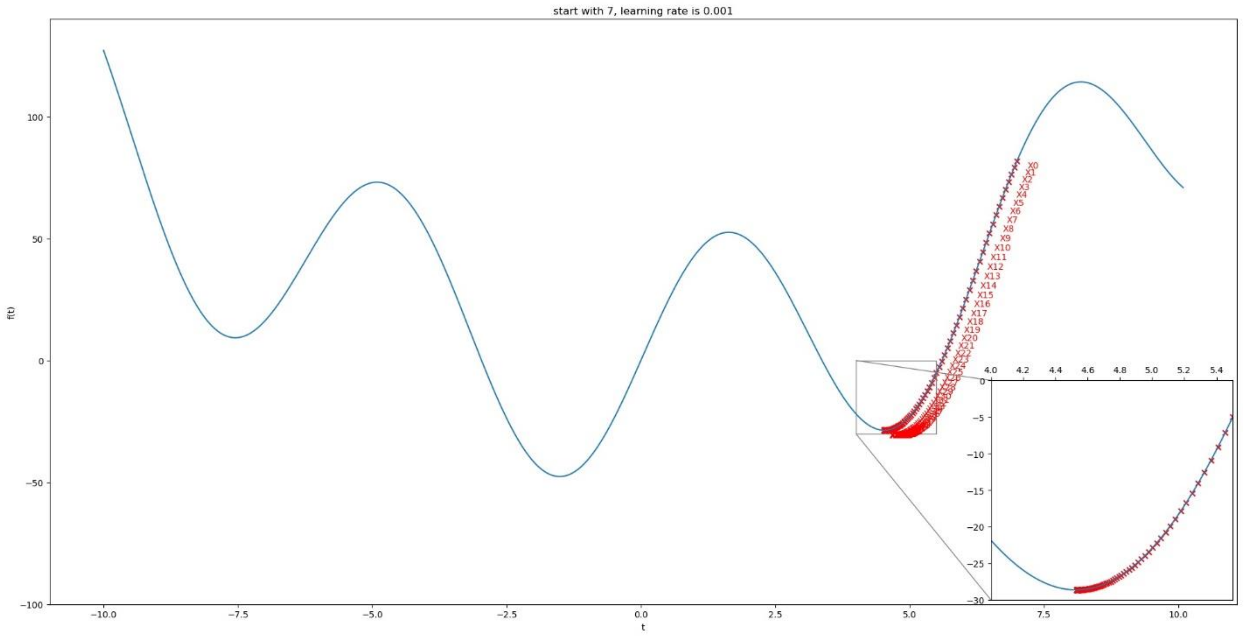

Figure 2 initial t = 7, learning rate c = 0.001

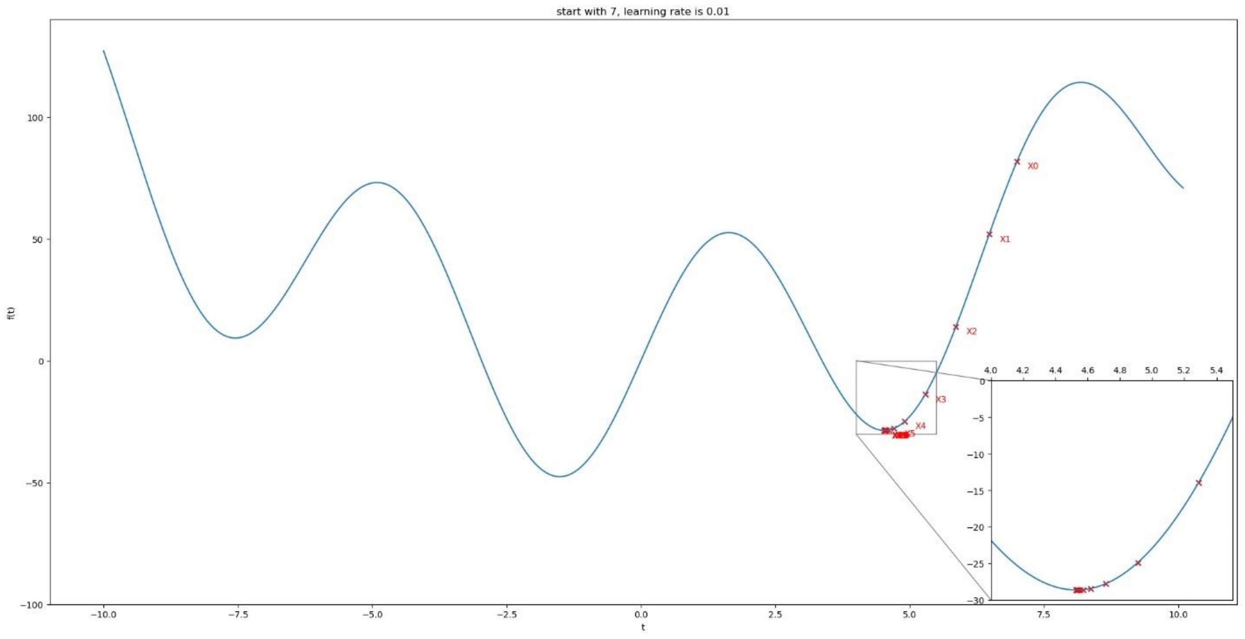

Figure 3 initial t = 7, learning rate c = 0.01

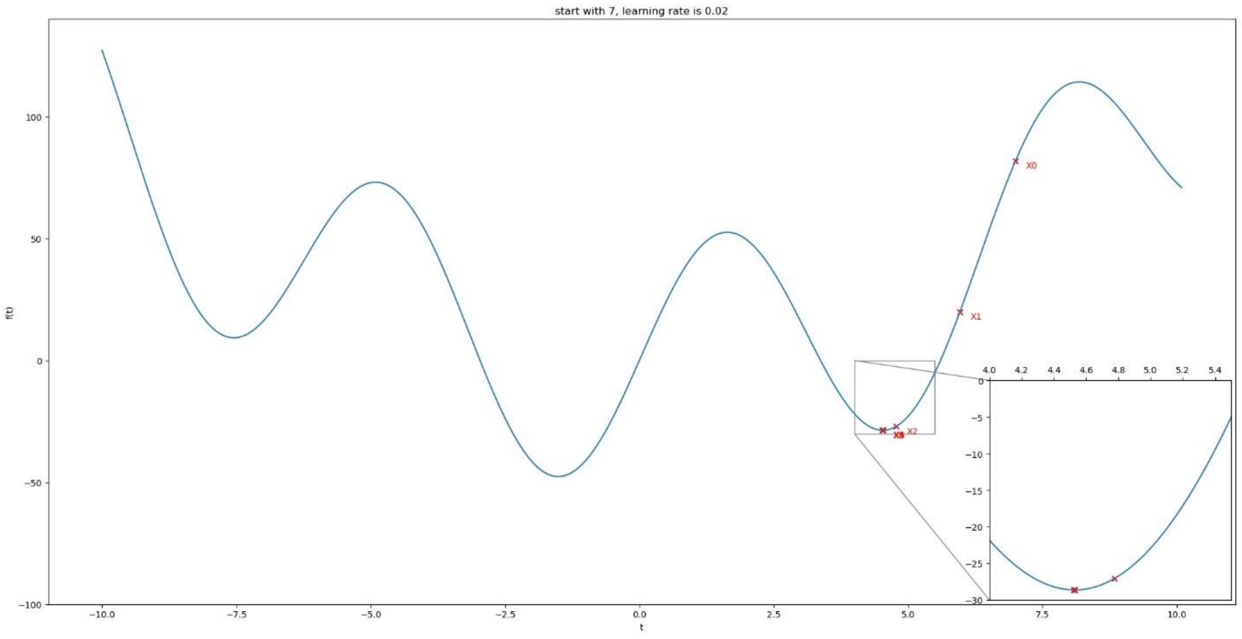

Figure 4 initial t = 7, learning rate c = 0.02

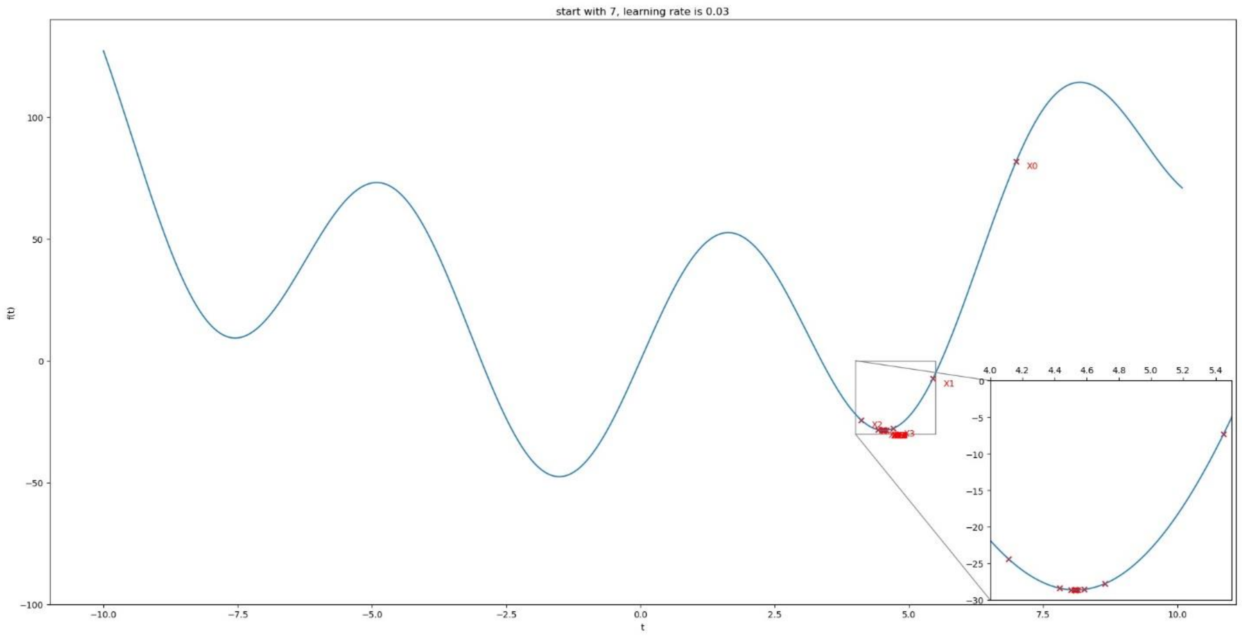

Figure 5 initial t = 7, learning rate c = 0.03

**ii) (10) Pick a starting point at t=1. What's the minimum? Show the convergence path. Experimenting with different learning rate.**

Figure 6-9 show the convergence path of initial t = 1 under different learning rate scenarios (i.e., 0.001, 0.01, 0.02, 0.03). The local minimum is obtained when t = -1.51.

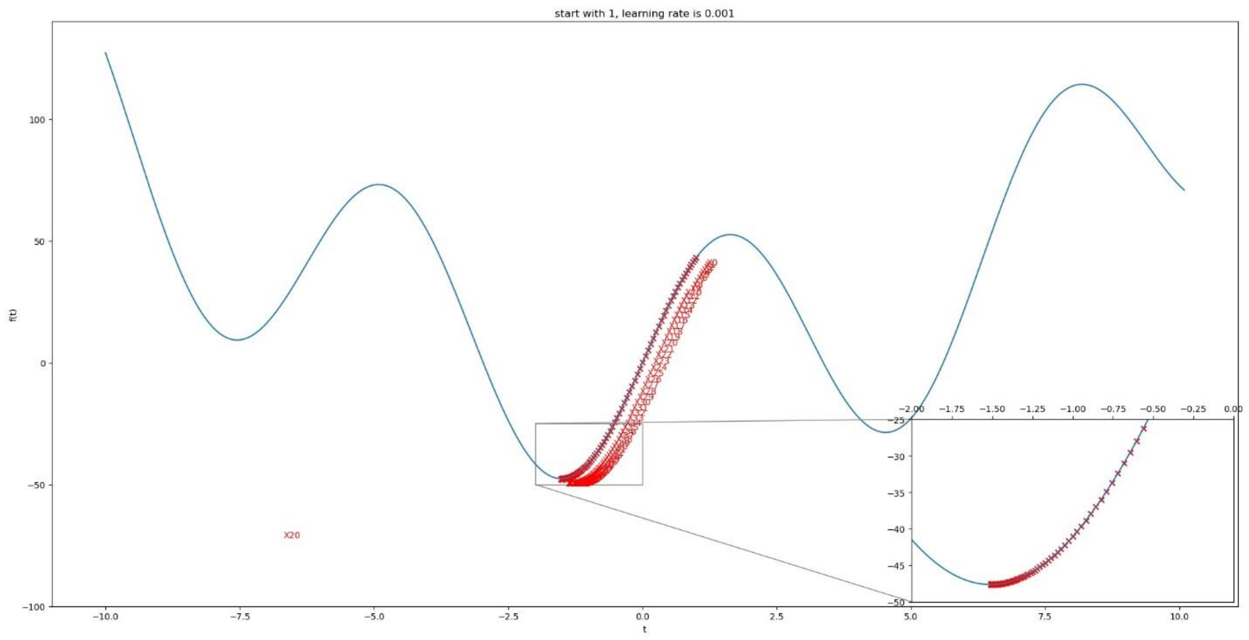

Figure 6 initial t = 1, learning rate c = 0.001

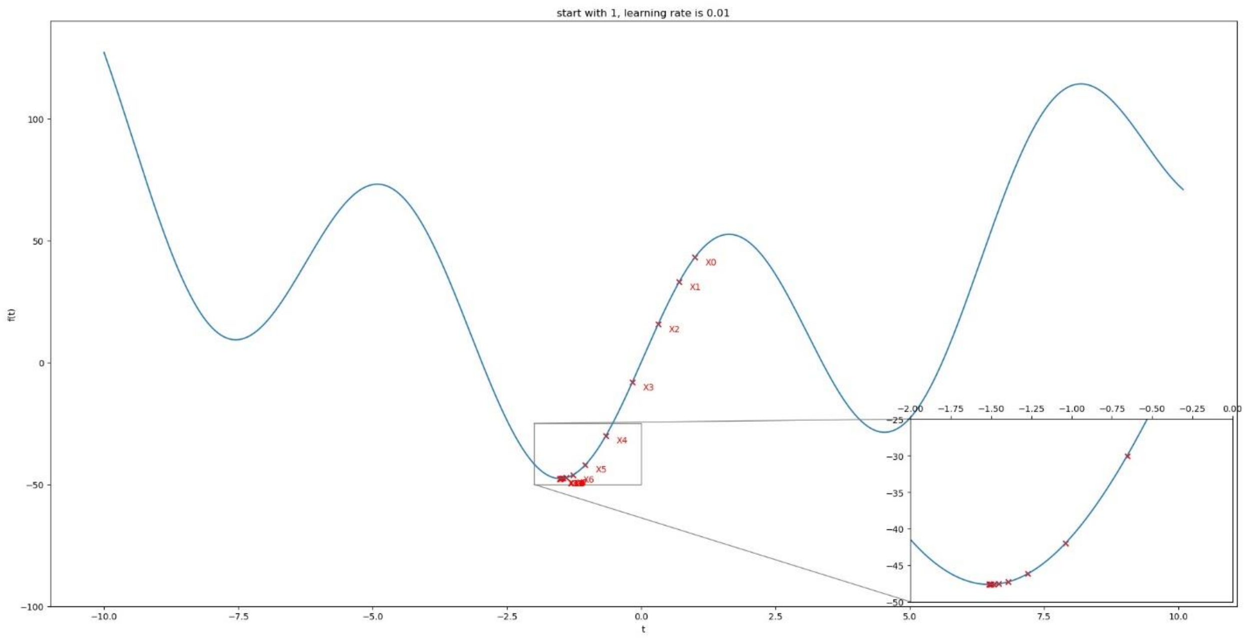

Figure 7 initial t = 1, learning rate c = 0.01

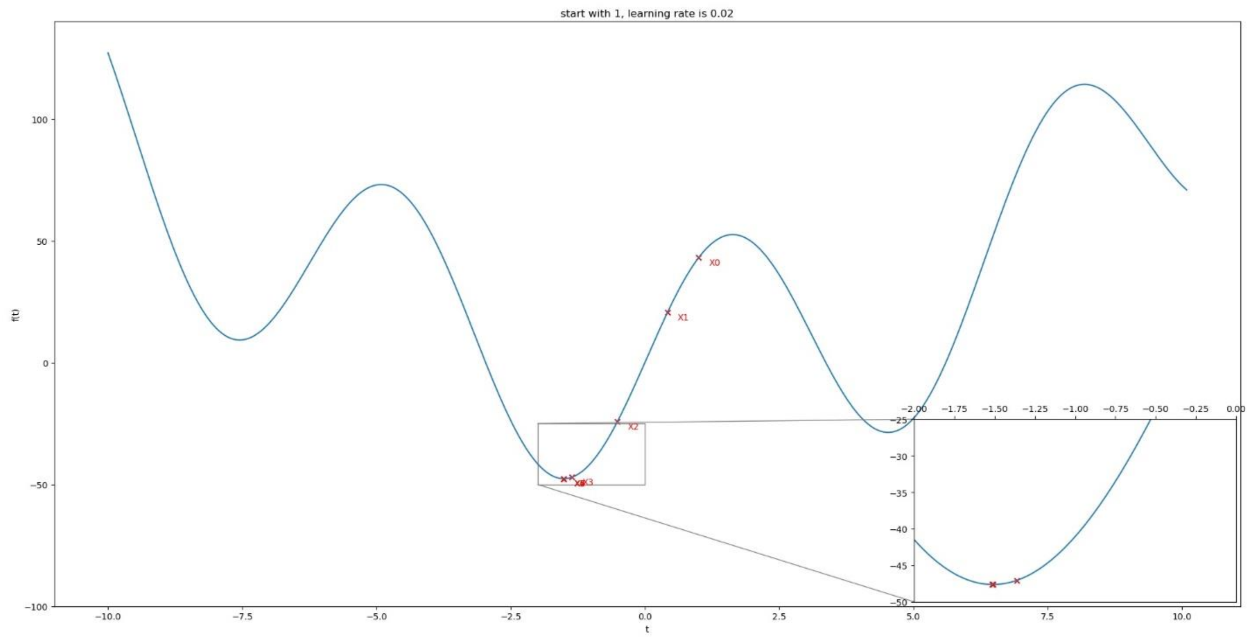

Figure 8 initial t = 1, learning rate c= 0.02

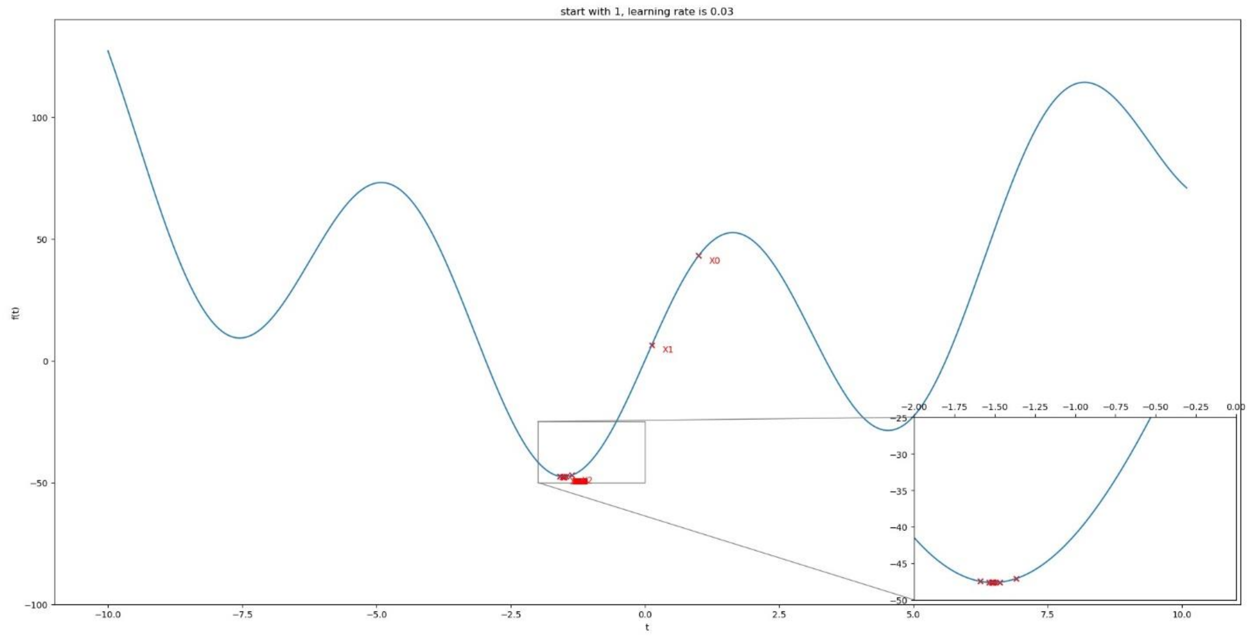

Figure 9 initial t = 1, learning rate c= 0.03

**iii) Comment on the results from the above experiments.**

Comparing above experiments with different initial value and learning rate, we found that different starting point might achieve the different local minimum. In this case, starting t at 7 gets the local minimum while starting t at 1 gets the global minimum. In addition, a large learning rate could speed the convergence, but it can also lead to missing the local minimum. As we can see that when c = 0.0001, almost all solutions are in the side of local minimum, while some solutions distributed in two sides of local minimum when c is increased to 0.03. If c is continuously increased, the convergence path zigzags and probably fails to converge in the end. Therefore, the learning rate should be carefully chosen before the search.

**Problem 2 One perceptron**

**a) (15) Use Perceptron to implement the OR logic. Show output from each iteration (that is, the two inputs, the targeted output, and the Perceptron output) with the maximum number of iterations being 10.**

**Table 1 OR gate**

| iteration | X | w | z | T |
|---|---|---|---|---|
| 1 | [[0,0],[1,0],[0,1],[1,1]] | [1.20333005, 0.00562272, -0.10429384] | [0. 0. 1. 1.] | [0,1,1,1] |
| 2 | [[0,0],[1,0],[0,1],[1,1]] | [ 1.20333005, 1.00562272, -0.10429384] | [1. 1. 0. 1.] | [0,1,1,1] |
| 3 | [[0,0],[1,0],[0,1],[1,1]] | [1.20333005, 1.00562272, 0.89570616] | [1. 1. 1. 1.] | [0,1,1,1] |
| 4 | [[0,0],[1,0],[0,1],[1,1]] | [1.20333005, 1.00562272, 0.89570616] | [0. 1. 1. 1.] | [0,1,1,1] |

b) (15) Use Perceptron to implement the XOR logic. Show output from each iteration (that is, the two inputs, the targeted output, and the Perceptron output) with the maximum number of iterations being 10.

**Table 2 XOR gate**

| iteration | X | w | z | T |
|---|---|---|---|---|
| 1 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 ,1.41395373, -0.57670865] | [0. 1. 0. 0.] | [1,0,0,1] |
| 2 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 ,1.41395373, 0.42329135] | [1. 1. 1. 0.] | [1,0,0,1] |
| 3 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 ,1.41395373, 0.42329135] | [0. 1. 1. 0.] | [1,0,0,1] |
| 4 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 ,1.41395373, 0.42329135] | [0. 1. 1. 0.] | [1,0,0,1] |
| 5 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 ,1.41395373, 0.42329135] | [0. 1. 1. 0.] | [1,0,0,1] |
| … | … | … | … | … |
| 10 | [[0,0],[1,0],[0,1],[1,1] | [0.55105346 1.41395373 0.42329135] | [0. 1. 1. 0.] | [1,0,0,1] |

Table 2 shows the classification results of XOR gate using one perceptron. It was found that one perceptron cannot converge in this case.

**Problem 3: (35) Comparison between FLD, PCA, and Perceptron.**

For AND gate, we have

$$X = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

The corresponding label is

$$y = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

First of all, FLD and PCA algorithm conducted on project 3 were applied to reduce the X to 1 dimension. We denote the processed data as fX and pX, respectively. The projection vectors are:

$$w_{fld} = [-2, -2]^T$$
$$w_{pca} = [0,1]^T$$

Secondly, the minimum distance classifier (known as case 1) was applied on fX and pX. The decision boundaries on projected data are:

For FLD,

$$x = -\frac{8}{3}$$

The corresponding decision boundary on 2D is

$$[-2, -2] * [x1, x2] = -\frac{8}{3}$$

From that we get:

$$\boldsymbol{x2 = -x1 + \frac{4}{3}}$$

Likewise, for PCA, the linear boundary on projection domain is

$$x = \frac{2}{3}$$

The corresponding decision boundary on 2D is

$$[0\ 1] * [x1, x2] = \frac{2}{3}$$

From that we get:

$$\boldsymbol{x2 = \frac{2}{3}}$$

Third, single layer perceptron is applied on the X and y. The converged weights is taken as the coefficient of decision boundary. That is

**X2= 1.76 - 1.25X1**

The above decision boundaries are plotted on figure 10. The yellow point denote label 1 while blue points are label 0. Two grey arrows reveal the projection vector or PCA (0,1) and FLD (-0.707, -0.707). It can be found that PCA+MD did not correctly classify all samples while FLD+MD and one perceptron did.

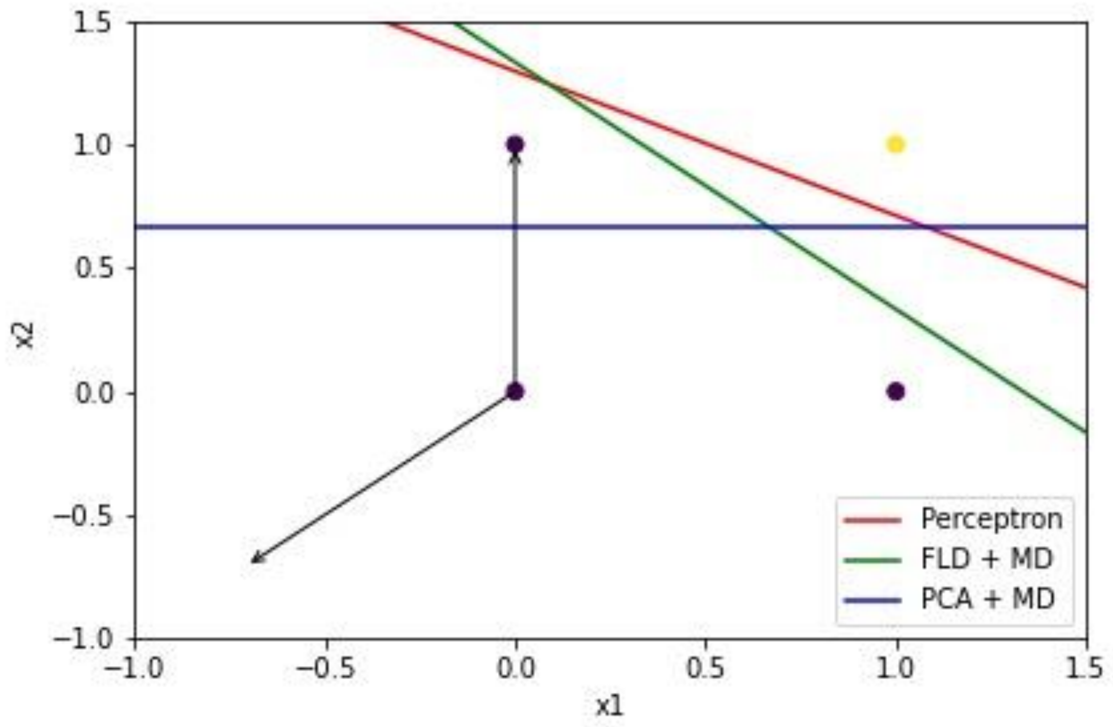

Figure 10 Decision boundaries

**APPENDIX**

Please see attachment.