Neurofuzzy Control and Applications
Numerical Optimization

Kontopoulos Stefanos

Introduction

This exercise introduces some computational issues in numerical optimization using the gradient descent method.

Question 1

Our function is as follows:

$$f(x_1, x_2) = x_1^2 + (x_2 - 1)^2 + (x_1 - x_2)^4 \tag{1}$$

Gradient Descent

Running the algorithm for the gradient descent method with step 0.1, we get the result shown in the figure 1.

The figure 2 shows for step 0.3. This value was chosen as an extreme example, as we observe there is no convergence and the values of x become very high.

We also notice that the smaller the step the more points it finds up to convergence 3.

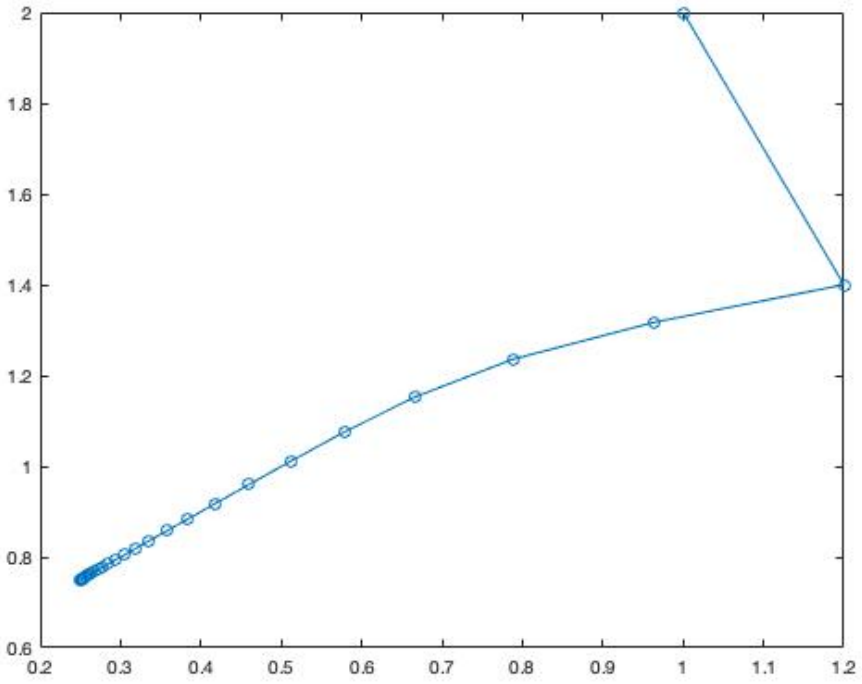

Figure 1: Gradient Descent 0.1.

Newton method

Applying the Newton method we have the result shown in the figure 4.

We see that it results in convergence with fewer steps and therefore it is a faster method, compared to the gradient descent.

Question 2

Our function is as follows:

$$f(x_1, x_2) = A \cdot x_1^2 + \frac{1}{A} \cdot x_2^2 \tag{2}$$

, with $A > 0$.

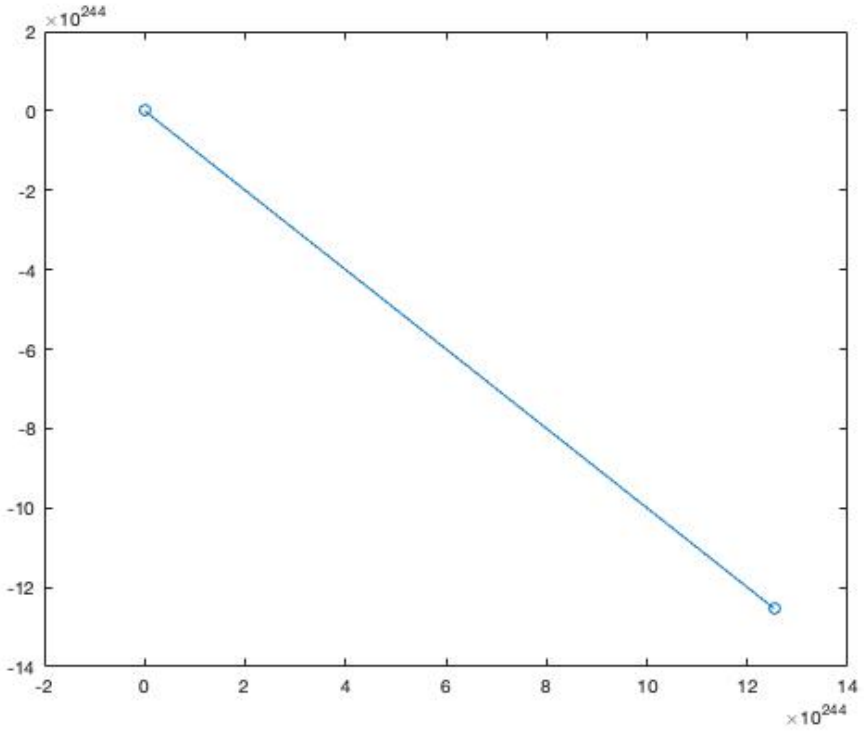

Figure 2: Gradient Descent 0.3.

Condition Number

The condition number was calculated using the Hessian function table.

$$H = \begin{bmatrix} \frac{\partial^2 f(x_1,x_2)}{\partial x_1^2} & \frac{\partial^2 f(x_1,x_2)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x_1,x_2)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x_1,x_2)}{\partial x_2^2} \end{bmatrix} \tag{3}$$

Specifically it emerged from the formula:

$$K(H) = \frac{\lambda_{max}(H)}{\lambda_{min}(H)} \tag{4}$$

, where λ_{max}, λ_{min} is the maximum and minimum eigenvalue of the Hessian table respectively.

$$H = \begin{bmatrix} 2 \cdot A & 0 \\ 0 & 2/A \end{bmatrix} \tag{5}$$

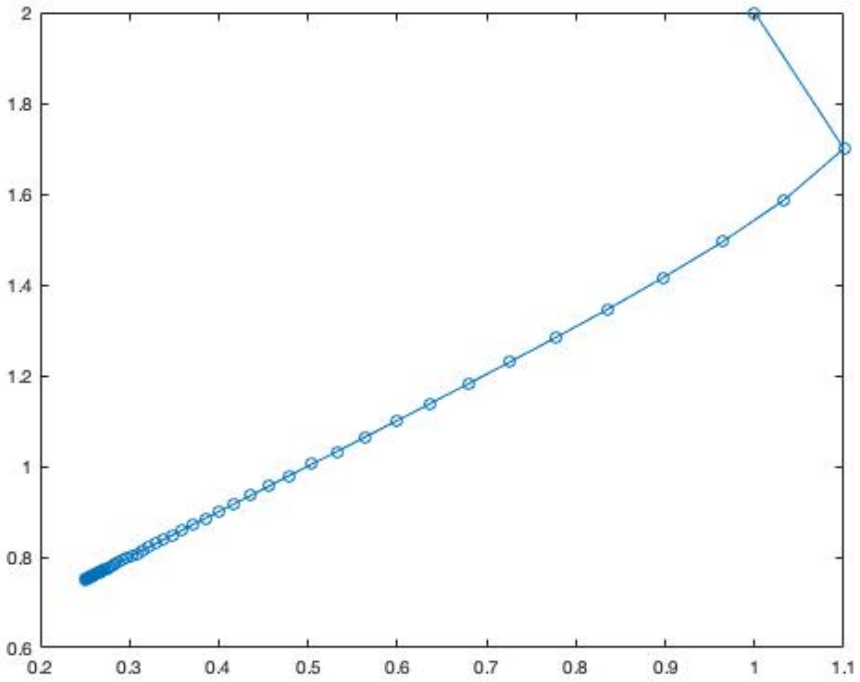

Figure 3: Gradient Descent 0.05.

We observe that the condition number increases for each value of A (Table 1). Specifically, this is the square value of A each time, ie $K \propto A^2$, with $A = 1, 2, 3, ..., 10$.

1	36
4	49
9	64
16	81
25	100

Table 1: Condition numbers.

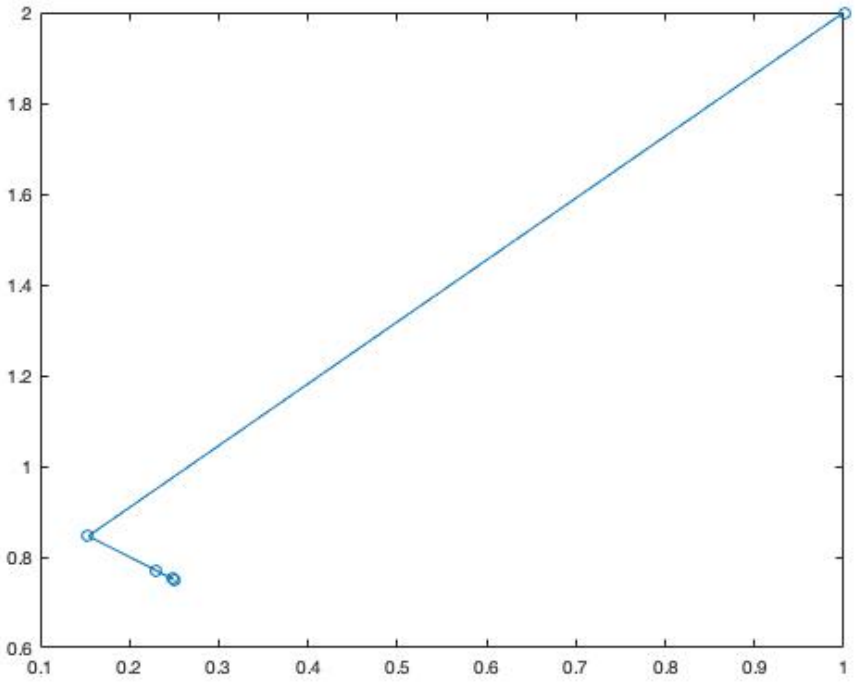

Figure 4: Newton method

Gradient Descent With Momentum

Here the same algorithm was applied as in Question 1, with step 0.2 and as shown in the figure 5, results in convergence.

Question 3

In this case we assume that Q is a symmetric positive semi-definite array produced as follows:

$$Q = A \cdot A^{\intercal} \tag{6}$$

, where A is a random table.

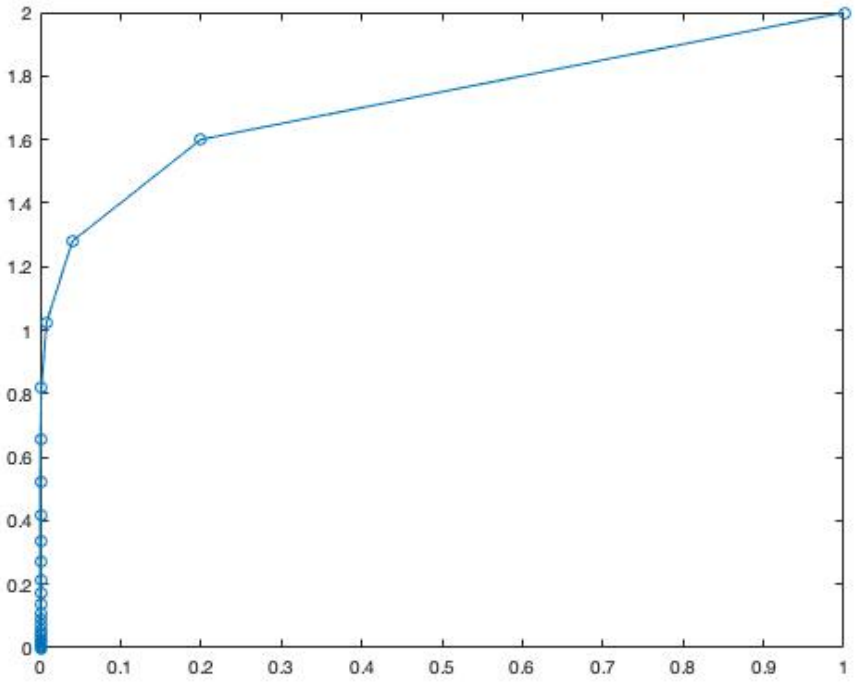

Figure 5: Gradient Descent With Momentum.

Condition Number

Now the condition number was calculated without the use of a hessian table.

$$K(Q) = \frac{\lambda_{max}(Q)}{\lambda_{min}(Q)} \tag{7}$$

In this case there is no analogy with the dimension of table A, as it is a random table each time larger. 10 of the 20 values of the condition number are shown in the Table2.

1	-5.2	-16.6	-4.2	-1.3	-5.5	-1	-1.6	...	-1.4

Table 2: Condition numbers.

Question 4

We have the function:

$$f(x_1, x_2) = \max(x_1 + x_2, 0.9 \cdot x_1 - 1.1 \cdot x_2 + 1, -0.8 \cdot x_1 + 1.2 \cdot x_2 - 1, 2 - 1.1 \cdot x_1 - 0.9 \cdot x_2) \tag{8}$$

Applying the gradient descent algorithm, we observe that for short step lengths it never converges. Figure 6 uses step 0.1.

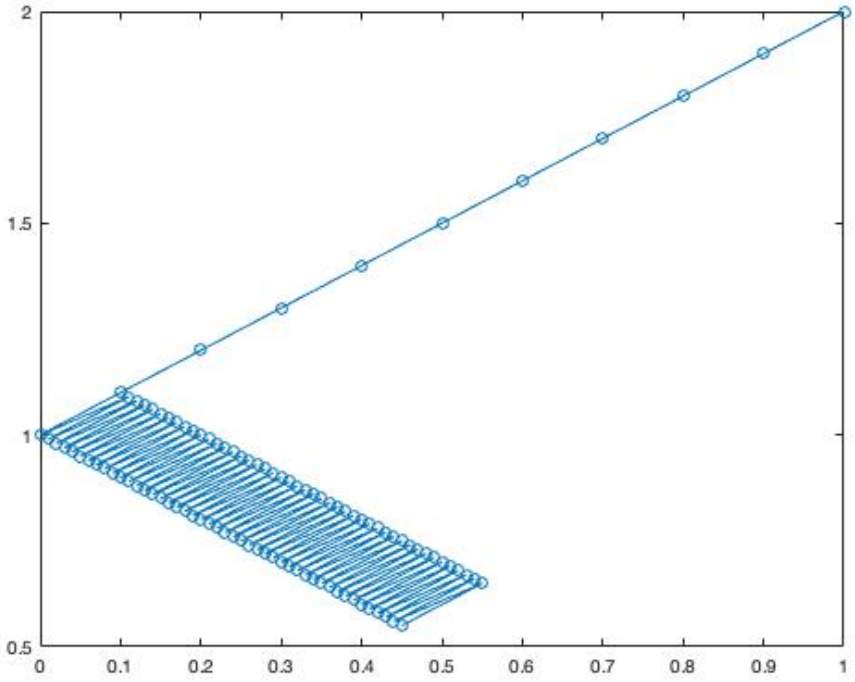

Figure 6: Not producible

This is because the function is discontinuous or not producible at this point. If we use a step of $1/k$ we expect some slow convergence:

$$\frac{1}{k} \cdot \sum_{n=0}^{+\infty} \alpha_n^2 < +\infty \tag{9}$$

, but in this example, for step $1/k = 10$ it's not obvious.

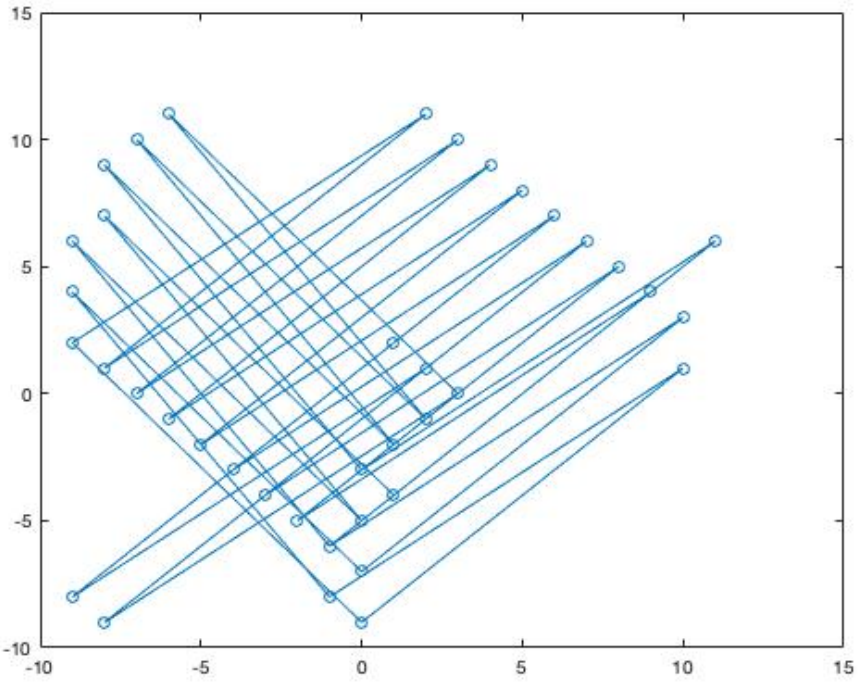

Figure 7: Does not converge