Imperial College London

COURSEWORK #2

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Mathematics for Machine Learning

Author:
Alexandre ALLANI (CID: 01797836)

Date: October 27, 2019

1 Differentiation

1.1 Question a

We have :

$$f_1(x) = x^\top \boldsymbol{B} x - x^\top x + \boldsymbol{a}^\top x - \boldsymbol{b}^\top x \tag{1}$$

Which can also be written (I_2 is the identity matrix for in $\mathbb{R}^{2x2}$:

$$f_1(x) = x^\top (\boldsymbol{B} - I_2)x + (\boldsymbol{a}^\top - \boldsymbol{b}^\top)x \tag{2}$$

We want (2) to be equal to:

$$(x - \boldsymbol{c})^\top \boldsymbol{C}(x - \boldsymbol{c}) + c_0 = x^\top \boldsymbol{C} x - x^\top \boldsymbol{C}\boldsymbol{c} - \boldsymbol{c}^\top \boldsymbol{C} x + \boldsymbol{c}^\top \boldsymbol{C}\boldsymbol{c} + c_0 \tag{3}$$

By identifying same orders element of (3) to (2) we already know that :

$$\boldsymbol{C} = (\boldsymbol{B} - I_2) = \begin{bmatrix} 3 & -2 \\ -2 & 3 \end{bmatrix} \tag{4}$$

We can also identify first order elements, hence :

$$-x^\top \boldsymbol{C}\boldsymbol{c} - x^\top \boldsymbol{C}\boldsymbol{c} = (\boldsymbol{a}^\top - \boldsymbol{b}^\top)x \tag{5}$$

Let $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and $x = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$ where $x_1 \in \mathbb{R}$, $x_2 \in \mathbb{R}$, $c_1 \in \mathbb{R}$ and $c_2 \in \mathbb{R}$. Then :

$$x^\top \boldsymbol{C}\boldsymbol{c} = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 3 & -2 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = (3c_1 - 2c_2)x_1 + (-2c_1 + 3c_2)x_2 \tag{6}$$

$$x^\top \boldsymbol{C}\boldsymbol{c} = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} 3 & -2 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = (3x_1 - 2x_2)c_1 + (-2x_1 + 3x_2)c_2 \tag{7}$$

$$(\boldsymbol{a}^\top - \boldsymbol{b}^\top)x = -2x_1 + 0x_2 \tag{8}$$

Hence:

$$x^\top \boldsymbol{C}\boldsymbol{c} + x^\top \boldsymbol{C}\boldsymbol{c} = (6c_1 - 4c_2)x_1 + (-4c_1 + 6c_2)x_2 = -2x_1 + 0x_2 \tag{9}$$

This leads to the following system of equation:

$$\begin{cases} 6c_1 - 4c_2 & = -2 \\ -4c_1 + 6c_2 & = 0 \end{cases} \Leftrightarrow \begin{cases} c_1 & = -3/5 \\ c_2 & = -2/5 \end{cases} \tag{10}$$

Finally we have:

$$c_0 = -\boldsymbol{c}^\top \boldsymbol{C}\boldsymbol{c} \Leftrightarrow c_0 = -0.6 \tag{11}$$

The solutions are:

$$\boldsymbol{C} = \begin{bmatrix} 3 & -2 \\ -2 & 3 \end{bmatrix}, \boldsymbol{c} = \begin{bmatrix} -3/5 \\ -2/5 \end{bmatrix}, c_0 = -0.6$$

1.2 Question b

To tell that f_1 has a minimum, we can compute its gradient and then its hessian. If there is a point in which the gradient is equal to $\mathbf{0}$ then f_1 has a local extremum. Then if the hessian matrix is positive definite in this point, we know that this extremum is a minimum. Let's compute the gradient of f_1. We know that it will be a 1×2 matrix:

$$\frac{\mathrm{d}f_1}{\mathrm{d}x}(x) = (x - \boldsymbol{c})^\top (\boldsymbol{C}^\top + \boldsymbol{C}) \tag{12}$$

Then :

$$\frac{\mathrm{d}f_1}{\mathrm{d}x}(x) = \mathbf{0}^\top \Leftrightarrow (x - \boldsymbol{c})^\top = \mathbf{0}^\top \Leftrightarrow x = \boldsymbol{c} \tag{13}$$

We can deduce that f_1 attains a local extremum in $\boldsymbol{c}$. The Hessian matrix is:

$$\frac{\mathrm{d}^2 f_1}{\mathrm{d}x^2}(x) = (\boldsymbol{C}^\top + \boldsymbol{C}) = \begin{bmatrix} 6 & -4 \\ -4 & 6 \end{bmatrix} \tag{14}$$

The Hessian matrix is constant, it is then the same matrix when evaluated at $\boldsymbol{c}$. Let's determine the eigenvalues of this matrix:

$$det(\begin{bmatrix} 6-\lambda & -4 \\ -4 & 6-\lambda \end{bmatrix}) = (\lambda - 2)(\lambda - 10) \tag{15}$$

The eigenvalues for the Hessian matrix of f_1 evaluated at $\boldsymbol{c}$ are $\lambda = 2$ and $\lambda = 10$.

The Hessian matrix is positive definite, hence $f_1$attains a minimum in $\boldsymbol{c}$

1.3 Question c

The gradient for each function will be a 1×2 matrix.
The gradient for f_1 has already been computed:

$$\frac{\mathrm{d}f_1}{\mathrm{d}x}(x) = (x - \boldsymbol{c})^\top (\boldsymbol{C}^\top + \boldsymbol{C})$$

Using the chain rule, we obtain the gradient for f_2 :

$$\frac{\mathrm{d}f_2}{\mathrm{d}x}(x) = -2(x - \boldsymbol{b})^\top sin((x - \boldsymbol{b})^\top (x - \boldsymbol{b})) + (x - \boldsymbol{a})^\top (\boldsymbol{B}^\top + \boldsymbol{B})$$

For f_3, we need to compute the following gradient fist:

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{1}{10} log \left|\tfrac{1}{100}\boldsymbol{I} + xx^\top\right|\right) \tag{16}$$

To do that, first we compute the value of determinant knowing $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ where $x_1 \in \mathbb{R}$, $x_2 \in \mathbb{R}$:

$$g(x) = \left|\tfrac{1}{100}\boldsymbol{I} + xx^\top\right| = \begin{vmatrix} 1/100 + x_1^2 & x_1x_2 \\ x_1x_2 & 1/100 + x_2^2 \end{vmatrix} = (1/100+x_1^2)(1/100+x_2^2) - x_1^2x_2^2 \quad (17)$$

We know that $g\colon \mathbb{R}^2 \to \mathbb{R}$, then the gradient will be of dimension 2×1. The gradient is then:

$$\frac{\mathrm{d}g}{\mathrm{d}x}(x) = \begin{bmatrix} \frac{\mathrm{d}g}{\mathrm{d}x_1}(x) \\ \frac{\mathrm{d}g}{\mathrm{d}x_2}(x) \end{bmatrix} = \frac{2}{100}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{2}{100}x \quad (18)$$

Form (18) we can deduce the gradient of f_3:

$$\begin{aligned} \frac{\mathrm{d}f_3}{\mathrm{d}x}(x) = \; & 2\times exp(-(x-\boldsymbol{a})^\top(x-\boldsymbol{a}))(x-\boldsymbol{a})^\top \\ & + exp(-(x-\boldsymbol{b})^\top\boldsymbol{B}(x-\boldsymbol{b}))(x-\boldsymbol{b})^\top(\boldsymbol{B}+\boldsymbol{B}^\top) \\ & + \frac{1}{10}\left[\frac{2}{100}\frac{1}{\left|\frac{1}{100}\boldsymbol{I}+xx^\top\right|}x\right] \end{aligned}$$

The gradients are implemented in the python file joined to this report.

1.4 Question d

The gradient descent algorithm is implement in the python file joined to this report. The function is called *grad_desc* and take as argument

- *grad* : Gradient function that will be used for gradient descent
- *step* : Number of iterations of the gradient descent algorithm
- *start* : Starting point of the algorithm
- *e* : Learning rate / step size

For the function f_2, **Figure 1** shows the gradient descent with f_2's contour plot. From the algorithm, the minimum is obtained around the point $x_{minimum} = \begin{bmatrix} -0.15649 \\ 0.92624 \end{bmatrix}$
For the function f_3, **Figure 2** shows the gradient descent with f_3's contour plot. From the algorithm, the minimum is obtained around the point $x_{minimum} = \begin{bmatrix} 0 \\ 0.04686 \end{bmatrix}$

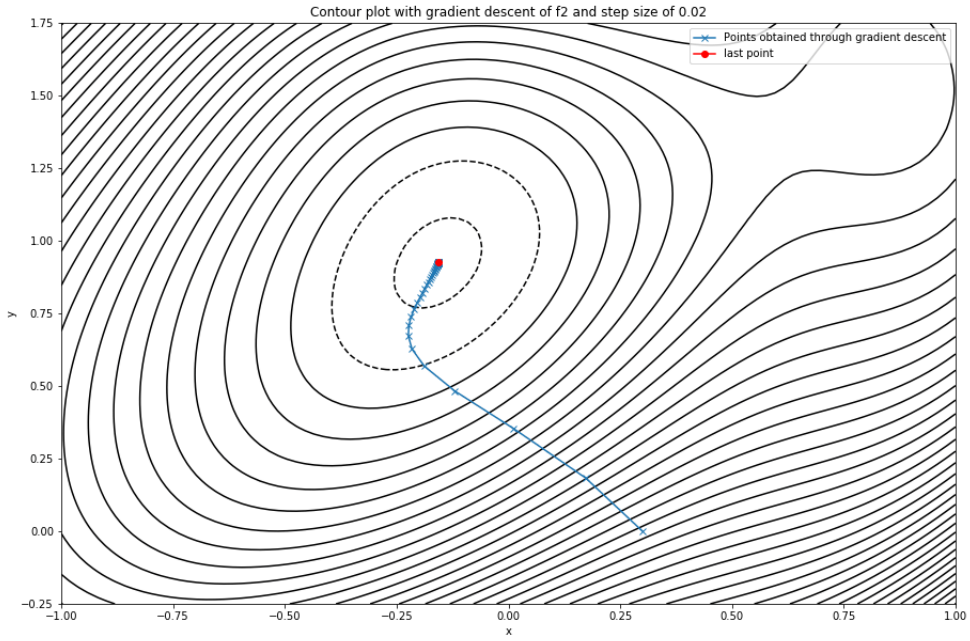

Figure 1: Gradient descent of f_2

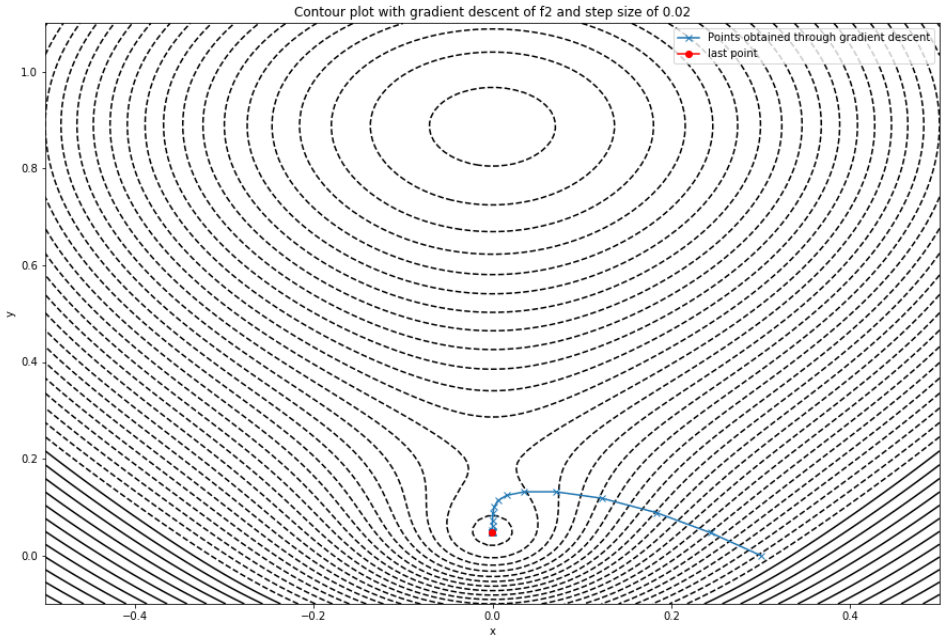

Figure 2: Gradient descent of f_3

1.5 Question e

The plots used are made with the gradient of f_2. We can see in general that if the step size (learning rate) is too small, then it takes more step to obtain the minimum. **Figure 1** shows it. We can even have the case where we are not even close to the minimum within the 50 iterations (**Figure 3**).
However the more we increase the learning rate, the faster we arrive to the minimum, (**Figure 4**),so less iterations is needed.
Nevertheless, if we push too far, and increase too much the learning rate, we are never going to find the minimum because the gradient descent will oscillate around it (**Figure 5**). These observations are quite logical, by subtracting the gradient to the point, we are heading toward a minimum (As the trace of each descent shows it, the gradient is orthogonal to the isolines). When the learning rate is too small, it might take infinite steps to reach the minimum. When it's too high, each substraction will pass via the minimum but will pass it.

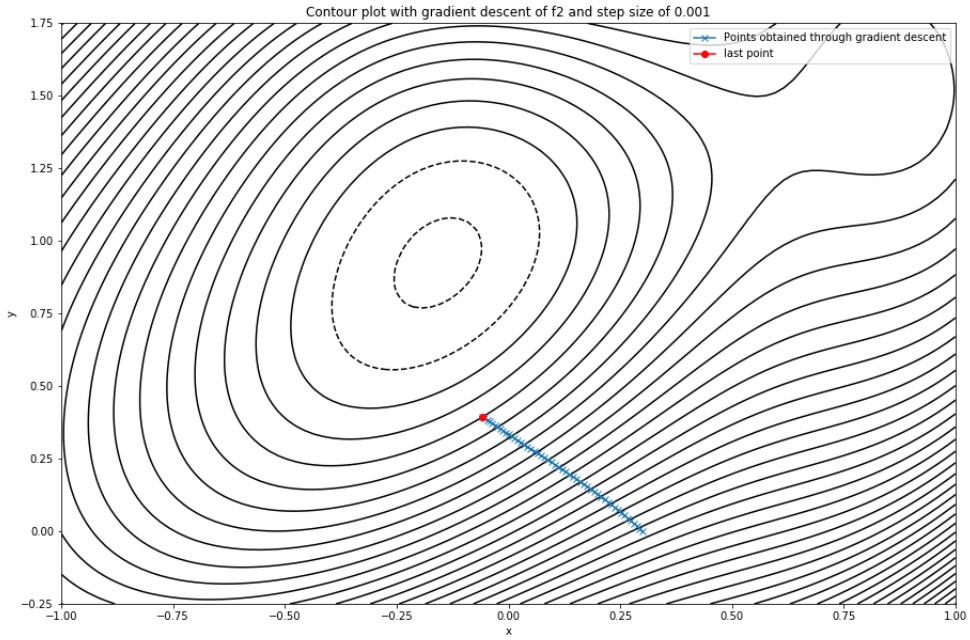

Figure 3: Gradient descent of f_2

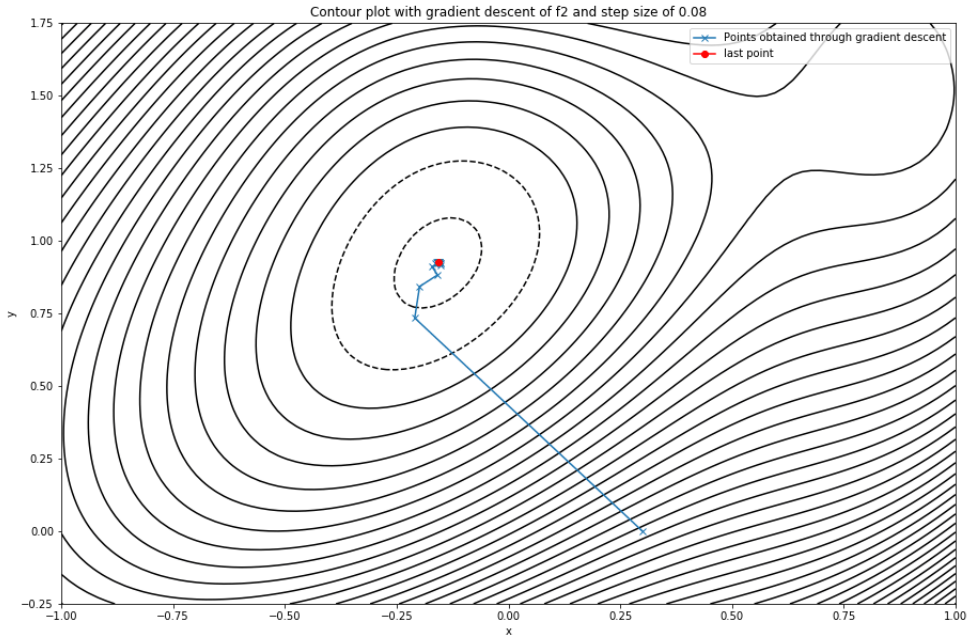

Figure 4: Gradient descent of f_2

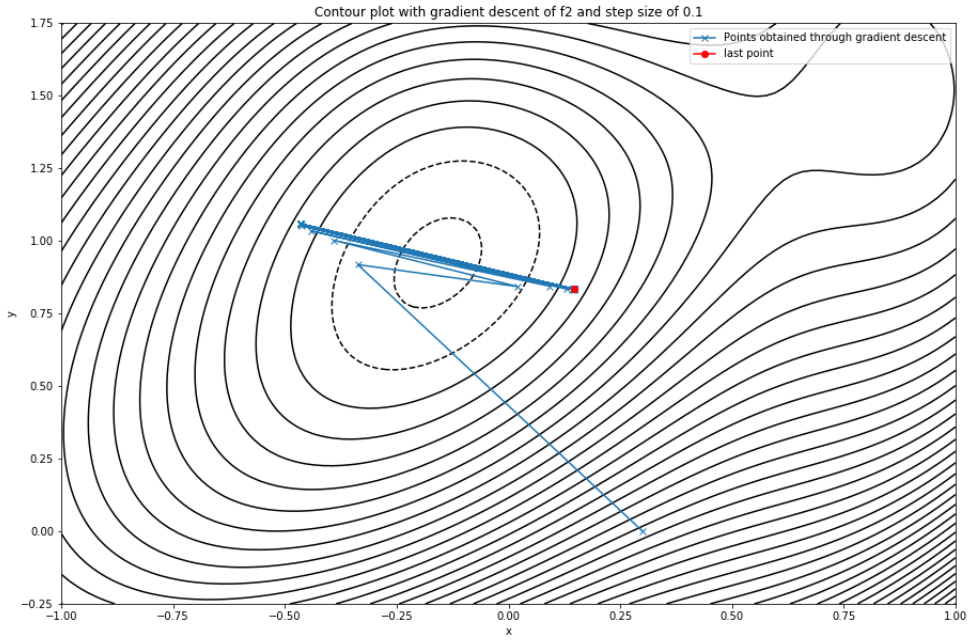

Figure 5: Gradient descent of f_2

When the learning rate is grossly mis-specified, the oscillation is greater. For the function f_2, this leads to a huge error as we can see on **Figure 6**.
For function f_3, the error the point also oscillates, but around another local minimum and not the one from before (**Figure 7**).
If we choose another starting point (with an adapted learning rate) $x_{start} = \begin{bmatrix} 0.3 \\ 0.8 \end{bmatrix}$ the gradient descent will give the coordinates of another minimum : $x_{minimum} = \begin{bmatrix} 0 \\ 0.88710 \end{bmatrix}$ (**Figure 8**)

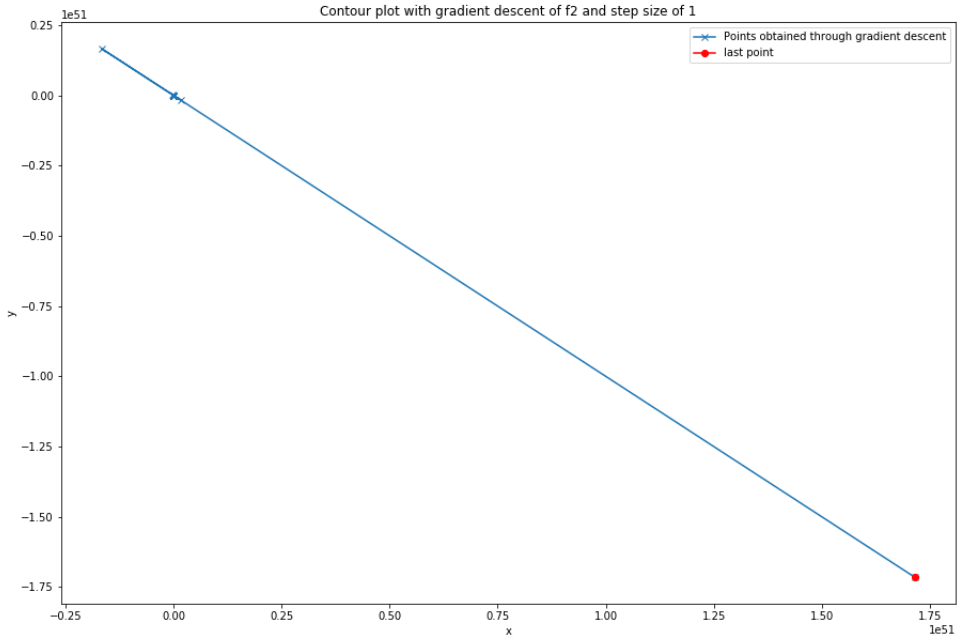

Figure 6: Gradient descent of f_2

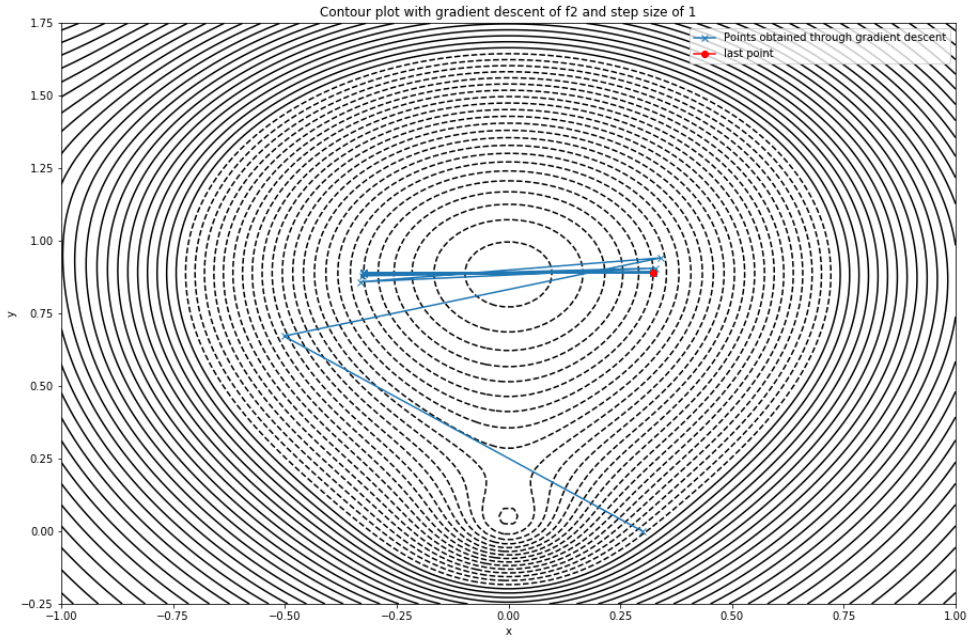

Figure 7: Gradient descent of f_3

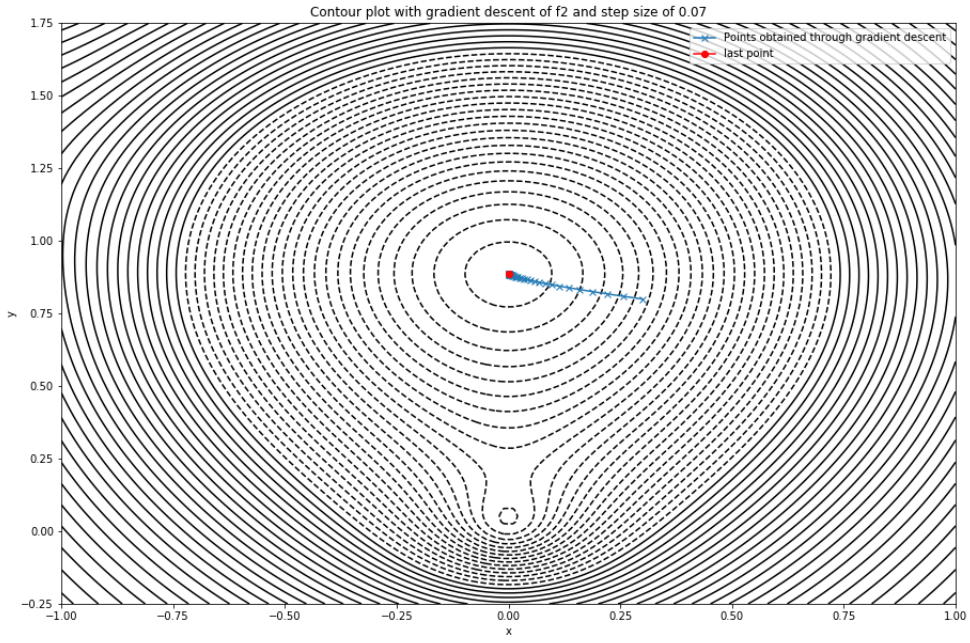

Figure 8: Gradient descent of f_3