Optimization & Optimal Research
Practical Session

Abstract

The aim of this practical session is to write the algorithm we have studied during the class. I will ask you to study the convexity of the function, find the global minimum or local minima. We will also so the influence of the learning rate for the gradient descent with a fix step and see a condition of convergence of this algorithm. You can use the language you want, but I will use R for the correction and I will not be able to help you if you are using an other one (maybe if you are using Matlab).

Introduction (20 minuts)

We will use the following functions :

$$f_1(x,y) = x^2 + \frac{y^2}{20},$$
$$f_2(x,y) = \frac{x^2}{2} + \frac{y^2}{2},$$
$$f_3(x,y) = (1-x)^2 + 10(y-x^2)^2,$$
$$f_4(x,y) = \frac{x^2}{2} + x\cos(y).$$

For the function f_3 we will take $(x_0, y_0) = (-1, 1)$ as the initialization of our algorithms.

1. Compute the gradient of each functions.

   The gradient of each function is given by :

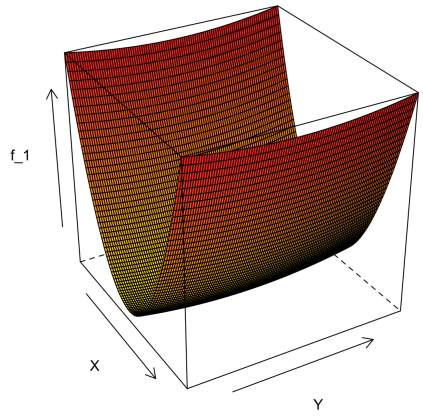
$$\nabla f_1(x,y) = \begin{pmatrix} 2x, & \frac{y}{10} \end{pmatrix}$$
$$\nabla f_2(x,y) = \begin{pmatrix} x, & y \end{pmatrix}$$
$$\nabla f_3(x,y) = \begin{pmatrix} 40x^3 - 40xy + 2x - 2, & 20y - 20x^2 \end{pmatrix}$$
$$\nabla f_4(x,y) = \begin{pmatrix} x + \cos(y), & -x\sin(y) \end{pmatrix}$$

2. Which of the functions ar convex ? Why ? You can use what we have studied during the lessons or in exercises.

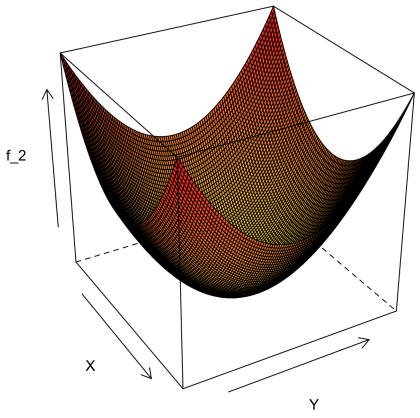
   The functions f_1 and f_2 are convex as the sum of two convex functions. The function f_3 is not convex as we have seen in class.
   Neither is f_4. If we compute the Hessian matrix we have :

$$\nabla^2 f_4(x,y) = \begin{pmatrix} 1 & -\sin(y) \\ -\sin(y) & -x\cos(y) \end{pmatrix}.$$

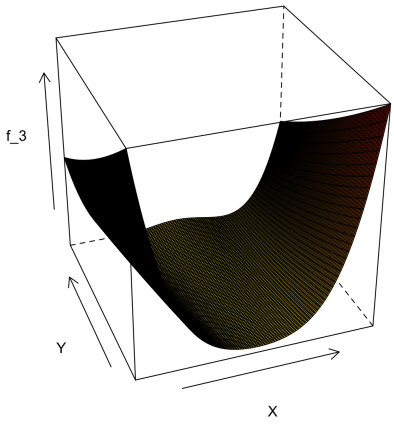
   If we take $(x, y) = (2, 0)$, we see that the trace is equal to $-1 < 0$.



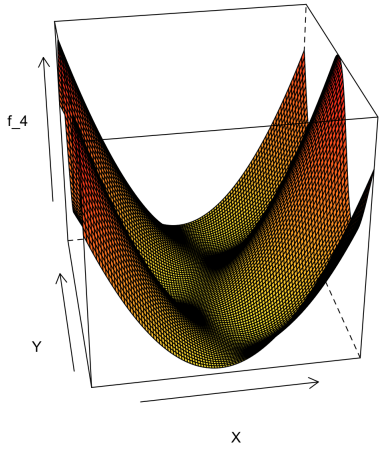
A)



B)



C)



D)

Figure 1: Graph of the four studied functions. A) Function f_1, B) Function f_2, C) Function f_3, D) Function f_4.

3. Plot these functions.

See Figure 1

4. What is the global minimum of each functions ?

The global minimum of the function f_1 and f_2 is reached at the point $(0, 0)$ and is equal to 0.
Note that the function f_3 is non-negative. Furthermore, we have $f(1, 1) = 0$ and for all $(x, y) \neq (1, 1),\ f(x, y) > 0$. So the minimum of f_3 is reached at the point $(1, 1)$ and is equal to 0.
For the function f_4 we need to solve *Euler's Equation* to find the *critical points* :

$$\nabla f_4(x, y) = \begin{pmatrix} x + \cos(y), & -x\sin(y) \end{pmatrix} = \begin{pmatrix} 0, & 0 \end{pmatrix}.$$

The second argument $-x\sin(y)$ is equal to zero when $x = 0$ or $y = k\pi$ for $k \in \mathbb{Z}$.
• If $x = 0$ then the first argument is equal to 0 when $\cos(y) = 0$ so $y = \dfrac{\pi}{2} + k\pi$.
• If $y = k\pi$ then the first argument is equal to 0 when $x = -cos(k\pi) = -(-1)^k = (-1)^{k+1}$.

The global minimum of f_4 is reached at the points $(-1, 2k\pi)$ and is equal to $-\dfrac{1}{2}$.

We now want to solve the following optimization problem

$$\min_{(x,y)\in\mathbb{R}^2} f(x, y),$$

for each function f using the different algorithm studied in class.

1 Gradient descent with a constant learning rate (40 minuts)

We recall that the gradient descent algorithm with constant learning rate $\eta > 0$ updates the weights at each iteration as follows :

$$u_{k+1} \leftarrow u_k - \eta\nabla f(u_k).$$

1. Define a function called *gradientdescent* using this gradient descent algorithm.

See the code

2. Use your function for the different values of η with the function (f_1, f_2, f_3, f_4). What do you notice ? Represent the convergence of (x, y)

See the code

- For the function f_1 :

As we will see in the text question, the algorithm converge if the learning rate : η is less than 1. For a learning rate $1 < \eta < 20$ the algorithm diverge but the variable y converge toward 0 unlike x. If $20 < \eta$ both variables diverge. The graphics are reprensented in Figure 2.

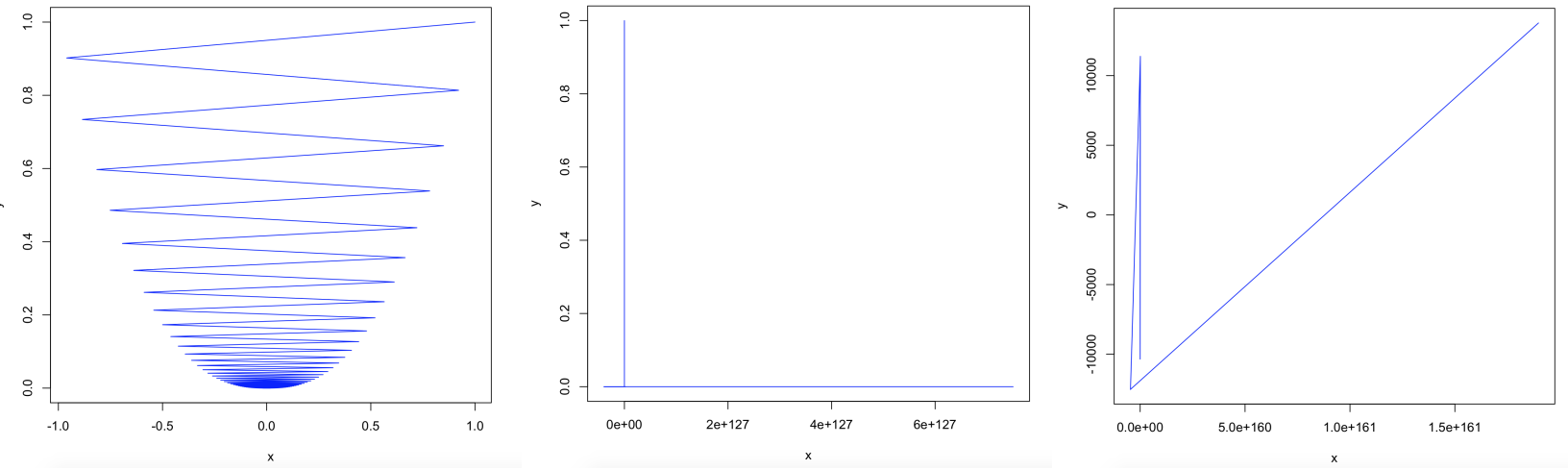


Figure 2: Convergence of the iterates for the function f_1 for the different values of η. From left to right we have $eta = 0.98, 10, 21$ respectively. We have done 100 iterations and we choose $u_0 = (1, 1)$

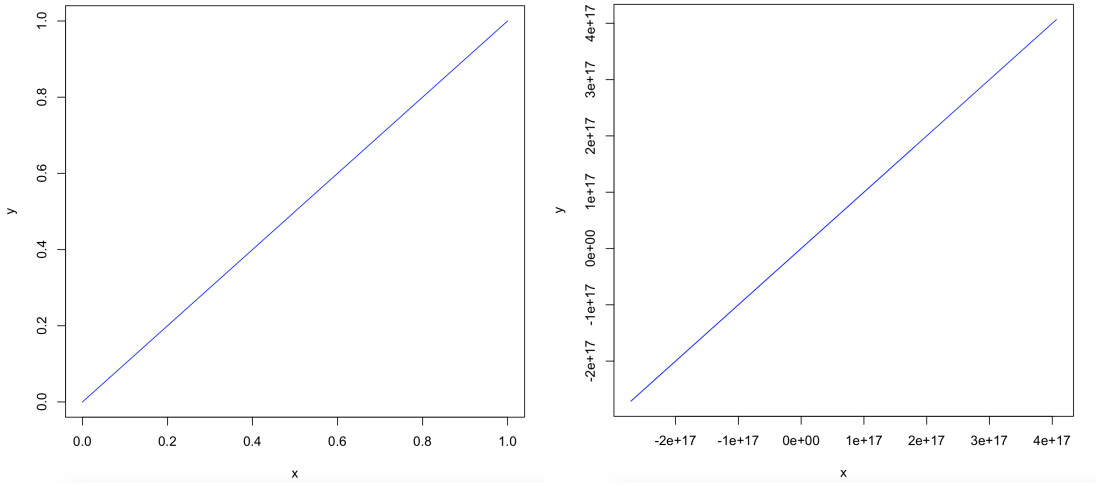


Figure 3: Convergence of the iterates for the function f_2 for the different values of η. From left to right we have $\eta = 0.5$ and $\eta = 2.5$ respectively. We have done 100 iterations and we choose $u_0 = (1, 1)$

- For the function f_2 :

  Same as before, we notice that the algorithm converge for $0 < \eta < 2$ and diverge otherwise. See Figure 3.

  For the functions f_1 and f_2 you can choose any starting point you want, you will always reach the point $(0, 0)$ if you respect the condition on the learning rate.
- For the function f_3 :

  We have see that $u = (1, 1)$ is the point where the function reaches its global minimum so we have to choose a different starting point, let say $u_0 = (-1, 1)$.

  Because the function is not convex, it's also interested to test different values of the starting point to see which point(s) we reach with this algorithm. But if you"re looking for the critical points (the

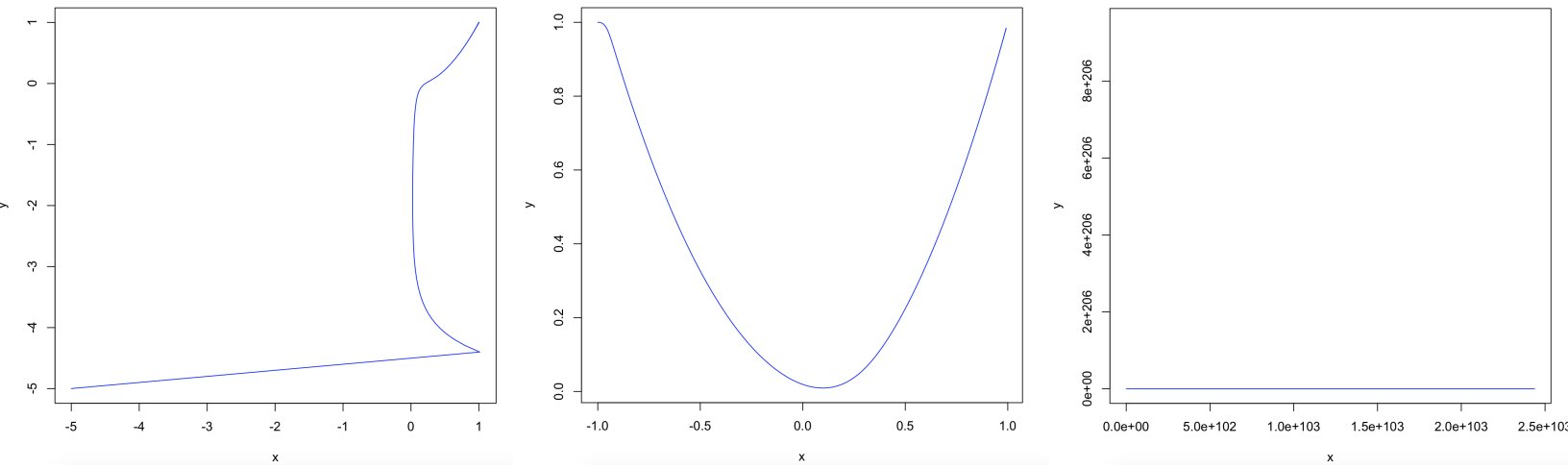


Figure 4: Convergence of the iterates for the function f_3 for the different values of η and starting points. From left to right we have $\eta = 0.001, 0.001$ and $\eta = 0.08$ respectively. The starting points are respectively $u_0 = (-5, -5)$, $u_0 = (-1, 1)$ and $u_0 = (-1, 1)$.

points u for which we have $\nabla f_3(u) = 0.$, the only solution is given by $u = (1, 1)$.
In practice, we will use a small learning rate.

The Figure 4 illsutrate the behaviour of the solution (the iterates) with respect to different values of the learning rate and values of initialization. We can notice that if we take an enough small learning rate and if we start at the point $u_0 = c(-1, 1)$ we reach the optimal point but we need a huge number of iterations to reach it, compared to the previous functions.
Try to implement the method using a learning rate $\eta = 0.5$ and the starting point $u_0 = (-1, 1)$. What do you notice ?

You should remark that the algorithm converge in 1 step !

- For the function f_4 :

For this function, we have seen that we have different global minima. So we will just test different values of u_0 and keep the same learning rate η for all experiments, results are presented on Figure 5.
In the instances presetend, we reach two different local minimum where the function is equal to 0. Even if you start around the the global minimum $u = (0, 2k\pi)$ it's really hard to reach it.

3. Now we consider the function f_2, if you have not done it, test the algorithm for $\eta = 1.9$ and $\eta = 2.1$. Give a condition on η so that the algorithm converges.

   We suppose that the function f is $\alpha-$elliptical and the gradient function ∇f is $L-$lipschitzien. We can show that this algorithm converges if we take η such that : $0 < \eta < \frac{2\alpha}{L^2}$.

   (a) Find the value of L such that (Definition of ∇f is L-lipschitzien) :

$$\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| \leq L\|(x_1, y_1) - (x_2, y_2)\|$$

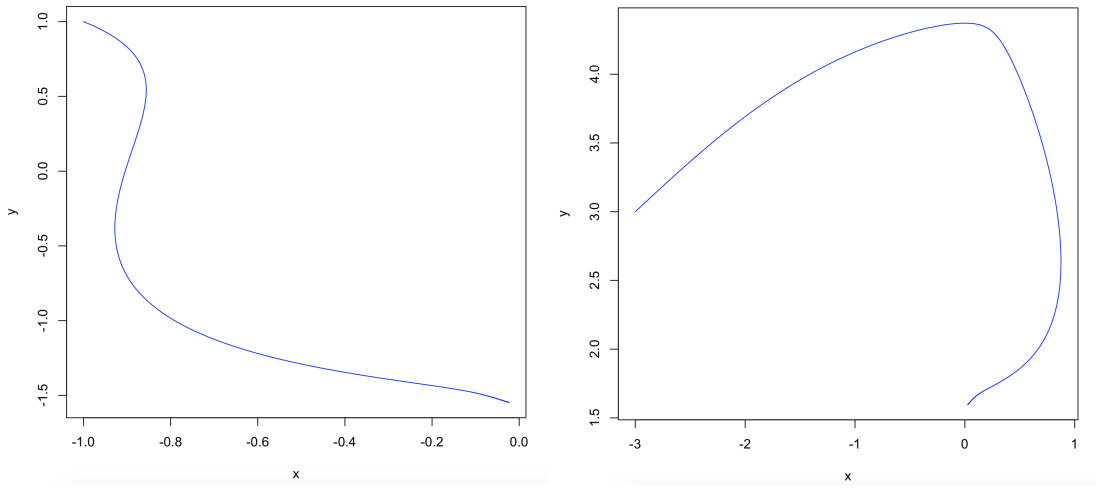


Figure 5: Convergence of the iterates for the function f_4 for the different starting points. From left to right we have $\eta = 0.001$ and the starting points are respectively $u_0 = (-1, 1)$, $u_0 = (-3, 3)$.

We have : $\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| = \|(x_1, y_1) - (x_2, y_2)\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.
And $\|(x_1, y_1) - (x_2, y_2)\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.
So we have :

$$\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| \leq \|(x_1, y_1) - (x_2, y_2)\|,$$

in other words, $L = 1$

(b) Compute λ_{min} the smallest eigenvalue of f. We admit that $\alpha = \lambda_{min}$ and conclude.

The Hessian matrix of f_2 is given by :

$$\nabla^2 f_2(x, y) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

So the eigenvalues of the matrix are equal to 1 and $\lambda_{min} = 1 = \alpha$

According to the theorem in red, the algorithm converges when $0 < \eta < \dfrac{2 \times 1}{1} = 2$. Its effectively what we have noticed before.

(c) Try to do the same for the function f_1.

We begin by calculating the eigenvalues of the Hessian matrix of f_1 given by :

$$\nabla^2 f_2(x, y) = \begin{pmatrix} 2 & 0 \\ 0 & \dfrac{1}{10} \end{pmatrix}.$$

So the eigenvalues are equal to 2 and $\dfrac{1}{10}$ and $\alpha = \lambda_{min} = \dfrac{1}{10}$.

Now we have to find a value of L such that :

$$\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| \leq L\|(x_1, y_1) - (x_2, y_2)\|$$

We have : $\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| = \sqrt{4(x_1 - x_2)^2 + \frac{1}{100}(y_1 - y_2)^2}$.
And $\|(x_1, y_1) - (x_2, y_2)\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.
So we have :

$$\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| \leq 2\|(x_1, y_1) - (x_2, y_2)\|,$$

in other words, $L = 4$. So the algorithm converges if $0 < \eta < \frac{2 \times \frac{1}{10}}{2^2} = \frac{1}{20}$.

Supplementary

The result is different is different from what we have seen when we have implemented the gradient descent for the function f_1. In fact, the theorem gives us a condition on η such that the algorithm converge, but it does not say that it gives the best range for η.

We have previously noticed that the algorithm converge for $\eta < 1$. We will try to explain why.
First you have to noticed that :

$$\nabla f_1(x, y) = \begin{pmatrix} g(x), & h(y) \end{pmatrix}$$

and the Hessian matrix is diagonal. So we can study the convergence of variable x and y independently.

• If we study the function with respect to x (so we consider that y is constant), we can show that g is 2-Lipschitzien. Moreover, we have $\nabla^2 f_1(x) \geq 2$ for all x.
According to the theorem, the gradient with optimal step converge if :

$$0 < \eta < \frac{2 \times 2}{2^2} = 1.$$

• If we study the function with respect to y (so we consider that x is constant), we can show that h is $\frac{1}{10}$-Lipschitzien. Moreover, we have $\nabla^2 f_1(y) \geq \frac{1}{10}$ for all x.
According to the theorem, the gradient with optimal step converge if :

$$0 < \eta < \frac{2 \times \frac{1}{10}}{\left(\frac{1}{10}\right)^2} = 20.$$

So the real range of values of η is $0 < \eta < min(1, 20) = 1$. For these values, the algorithm converge.

2 Gradient descent with optimal step (30 minuts)

Now the learning rate is no more constant, it is determined by solving the problem :

$$\eta^{(k)} = \underset{eta>0}{Argmin}\ f(u_k - \eta \nabla f(u_k)).$$

1. Give an explicit expression of η for the first and/or second function(s).

According to what we have seen in class, the optimal learning rate ρ for quadratic function is given by :

$$\rho = \frac{\|Au - b\|^2}{\|Au - b\|_A^2}.$$

• For the function f_1, we have $A = \begin{pmatrix} 2 & 0 \\ 0 & \frac{1}{10} \end{pmatrix}$ and $b = 0$. So $Au - b = \begin{pmatrix} 2x \\ \frac{y}{10} \end{pmatrix}$ and $A(Au - b) = \begin{pmatrix} 4x \\ \frac{y}{100} \end{pmatrix}$. So $\|Au - b\|^2 = 4x^2 + \frac{y^2}{100}$ and $\|Au - b\|_A^2 = 8x^2 + \frac{y^2}{1000}$. So

$$\rho = \frac{4x^2 + \frac{y^2}{100}}{8x^2 + \frac{y^2}{1000}}.$$

• For the function f_2 we have $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $b = 0$. So $Au - b = \begin{pmatrix} x \\ y \end{pmatrix}$ and $A(Au - b) = \begin{pmatrix} x \\ y \end{pmatrix}$. So $\|Au - b\|^2 = x^2 + y^2$ and $\|Au - b\|_A^2 = x^2 + y^2$. So

$$\rho = \frac{x^2 + y^2}{x^2 + y^2} = 1.$$

2. Implement the algorithm.

   See the code : The algorithm is implemented for quadratic function f_1 and f_2.
   We will see how to do it for f_3 later.

3. Solve the problem of minimization of function f_1 and compare to the previous algorithm.

   The graphs presented on Figure 6 show that the gradient descent with optimal step converges faster than the gradient descent with fix step.
   THis is normal because, at each step, we choose the optimal learning along the given direction so that the value of f deacreases the most.

4. Do the same for the function f_3.

   I will do the calculous later, but you have to find the value of the learning rate η that minmize a polynom of degree 3 (remember that the learning rate is positive).

3 Newton's Method (30 minuts)

The Newton's Method is solving Euler's Equation

$$\nabla f(u) = 0.$$

An iteration of the Newton's algorithm is given by :

$$u_{k+1} \leftarrow u_k - (H_f(u))^{-1} \nabla f(u),$$

where H_j refers to the Hessian matrix of the function f.

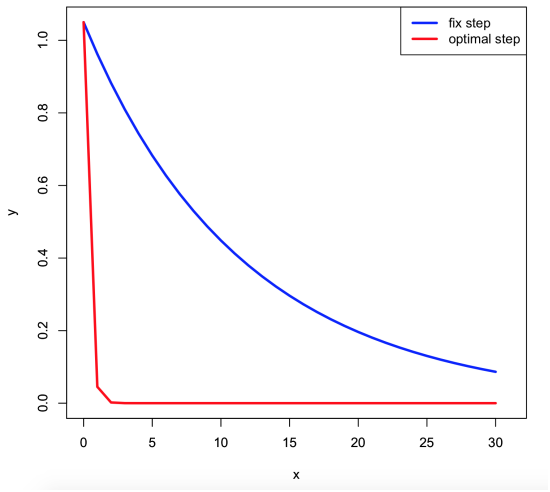


Figure 6: We compare the convergence in terms of number of iterates of the gradient descent with a fix step (the blue curve) in VS gradient descent with optimal step (the red curve). We started the algorithm at the point $u_0 = (-1, 1)$ and we took a learning rate equal to 0.98 for the version with a fix step.

Remark : It is possible to improve this method combining it with a line search algorithm, setting :

$$u_{k+1} \leftarrow u_k - \eta(H_f(u))^{-1}\nabla f(u),$$

where η is constant or satisfies the Wolfe's condition.

1. Compute the Hessian matrix for the functions f_1, f_3 and f_4.

   The Hessian matrices of the functions are given by :

$$\nabla^2 f_1(x,y) = \begin{pmatrix} 2 & 0 \\ 0 & \frac{1}{10} \end{pmatrix} \qquad \nabla^2 f_2(x,y) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\nabla^2 f_3(x,y) = \begin{pmatrix} 120x^2 - 40y + 2 & -40x \\ -40x & 20 \end{pmatrix} \quad \nabla^2 f_4(x,y) = \begin{pmatrix} 1 & -\sin(y) \\ -\sin(y) & -x\cos(y) \end{pmatrix}$$

2. Implement the Newton's method for the function f_2 and f_3 to find the global minimum.

   See the code

3. For the function f_2 and/or f_3, compare the convergence of the three algorithms.

   See the code

   • On Figure 7, we compare the three algorithms with the function f_1. We can see that both Newton's method and gradient descent with optimal step are better than the classical gradient descent and the two reach the optimal value at the same speed. Here we also see that the Newton's method is better than the Gradient Descent with optimal step but it's not obvious.

   We need an other example (i.e an other function) to compare the speed of convergence (in term of iterations) to see which algorithm is faster between Gradient Descent with optimal step and Newton's method.

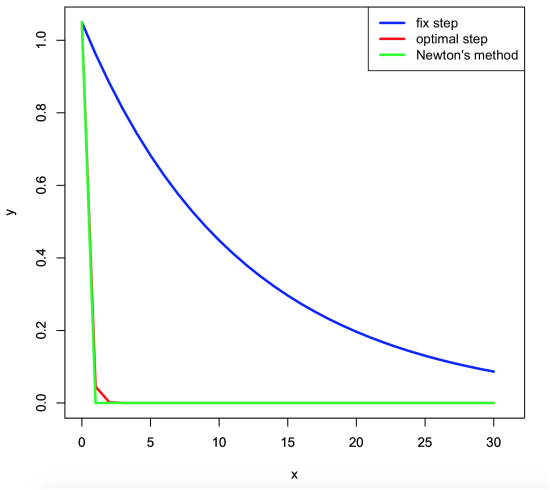


Figure 7: The three algorithms are used to minimize the function f_1 we a starting point equal to $u_0 = (-1, 1)$. The learning rate η is equal to 0.98 for the fix step method.

4. What are the avantage(s) and drawback(s) of this method ?

   This algorithm is faster than the two others so it requires less iterations to reach the global minimum of a convex function or local minimum of the other functions.
   It works well when you work in low dimensionnal space and with few data (it depends on the context).

   This algorithm is weak when it comes to use it for problem with a huge number or variables. You' ll have to invert a matrix of size $p \times p$ which cost $O(p^3)$. Even it requires less iteration than the others you will spend more time to invert the Hessian matrix.
   Moreover, in a machine learning context, you can spend lots of time to calculate the Hessian matrix if you have lots of data (it depends on the expression of the Hessian matrix).