Newton method principles

Lluís Garrido – lluis.garrido@ub.edu

October 2016

Abstract

This laboratory is focused on unconstrained optimization of a function $f(x)$ and, in particular, on understanding the Newton method. Why does this method work and is usually much better than the gradient descent?

1 A simple quadratic function

We begin by focusing on a simple two-dimensional quadratic function. Concretely,

$$f(x) = 100\, x_1^2 + x_2^2$$

where , $x \in R^2$, $x = (x_1, x_2)^T$ (vectors are expressed column-wise). Observe that the function is convex and, thus, it has one unique stationary point which corresponds to the minimum. You are proposed to follow the next steps:

1. We are first going to minimize the previous function using the gradient descent algorithm. That is,

   $$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

   Given an initial guess x^0, count the number of steps that are needed in order to find the minimum so that $\alpha^k < 10^{-5}$. The value of α^k will be computed using the backtracking algorithm you implemented in the previous lab.

   For the backtracking algorithm, it is recommended that at each iteration k an initial value of $\alpha = 1$ is tested. Do not store the value of α for $k-1$ as initial value for iteration k. This behavior is recommended for the Newton algorithm since the Newton algorithm automatically "scales" the descent direction.

2. In a general case, an unconstrained minimization algorithm uses the next algorithm

   $$x^{k+1} = x^k + \alpha^k d^k$$

   where d^k is a descent direction. For the Newton method the descent direction d^k is the solution of

   $$\nabla^2 f(x^k)\, d^k = -\nabla f(x^k)$$ 

You are now requested to minimize the previous function using the Newton method. As has been done with the gradient descent, count the number of steps that are needed in order to find the minimum so that $\alpha^k < 10^{-5}$. In order to be fair, be sure to use the same backtracking algorithm and the same initial points as has been done for the gradient descent.

3. Compare the number of iterations that are needed to get to the minimum. It may be interesting to plot the path that each of the method follows.

2 The exercise of lab 1

We are now going to focus on the function you studied in lab 1

$$f(x_1, x_2) = x_1^2\,(4 - 2.1\,x_1^2 + \frac{1}{3}x_1^4) + x_1 x_2 + x_2^2\,(-4 + 4x_2^2)$$

Observe that this function is not convex. You are requested to perform the next experiments

1. Recover the experiments you performed in the previous lab. Indeed, take an initial point x^0, quite far away from a minimum, and compute the number of iterations that are needed to get to the minimum, i.e. $\alpha^k < 10^{-5}$.

2. We are now going to focus on the Newton method. As stated before, the Newton method uses a descent direction which is the solution of

$$\nabla^2 f(x^k)\,d^k = -\nabla f(x^k) \tag{1}$$

The vector d^k is a descent direction only if the Hessian is positive definite. If the Hessian is not positive definite the obtained vector d^k is not necessarily a descent direction. Thus, another approach has to be devised if the Hessian is not positive definite. There are several methods that try to tackle the previous problem. For instance, a simple method is to use the gradient descent in case the Hessian is not positive definite. Another method is to perform the descent only for those components $(x_1, x_2)^T$ for which the corresponding eigenvalue is positive (recall that the eigenvalue gives us information about the shape of the function: convex or concave).

In this lab we propose to use the gradient descent in case the Hessian at iteration x^k is not positive definite. Thus, the algorithm to implement can be summarized as follows:

   (a) Starting from an initial point x^0, compute the Hessian at each iteration k.

   (b) If the Hessian is positive definite, use the Newton method to perform the descent. Otherwise, use the gradient descent to perform the descent. In both cases use the backtracking algorithm to compute a good value for α^k. Stop the algorithm as soon as $\alpha^k < 10^{-5}$.

It is interesting to analyze which of both methods (Newton or gradient descent) is used at each iteration k. A simple way to proceed is to plot the path the minimization algorithm follows: use red dots if the algorithm uses a gradient descent, and use green dots if the Newton method is used. You should observe that "near" the minimum the Newton method is mostly used. The function is convex near the minimum and that allows to approach it in a fast way.

3. Compare the Newton method (item 2) with the pure gradient descent (item 1). You may test how many iterations are required to arrive to the minimum and compare the path that each of the method follows to arrive to the minimum. .

Just some words to finish. You should observe that the Newton method requires, in general, less iterations than the gradient descent to arrive to the minimum. But the disadvantage of the Newton method is that it requires solving the linear system of equations (1) and thus it requires higher computational effort. In addition, one needs to know if the Hessian matrix if positive definite in order to be sure that the vector d^k is a descent direction. There are methods that tackle the previous computational issues, namely the conjugate gradient descent algorithm. In any case, take into account that the Newton method has great advantages over the gradient descent method. But this does not necessarily mean that the Newton method is always better than the gradient descent. In some cases the gradient descent may be good enough for the problem to be solved.

3 The Rosenbrock function

Let us consider the Rosenbrock function, see figure 1. The function is defined as

$$f(x_1, x_2) = (a - x_1)^2 + b\,(x_2 - x_1^2)^2$$

The function has a global minimum at $(x_1, x_2) = (a, a^2)$, where $f(x_1, x_2) = 0$. The global minimum is inside a long, narrow, parabolic shaped valley.

The convergence to the global minimum is difficult. In the previous lab you should have seen that you require a lot of iterations of the gradient descent in order to arrive to the minimum. This is due to the fact that during iterations the gradient descent continuously jumps from one side to the other side of the valley without taking into account the shape of the valley. How does Newton perform here?

Assume you take $a = 1$ and $b = 100$ for the Rosenbrock function. You are asked to perform the next experiments:

1. Take an initial point x^0 and test how many iterations are required in order to arrive to the minimum using the gradient descent. For that issue use the backtracking algorithm and stop the iterations as soon as $\alpha^k < 10^{-5}$.

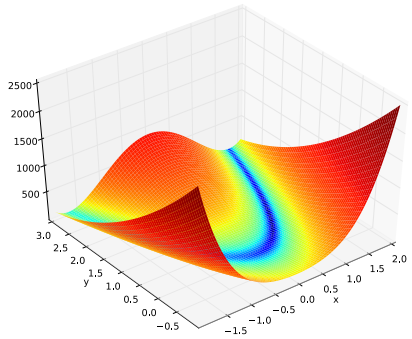

Figure 1: The Rosenbrock function of two variables, $a = 1$ and $b = 100$. Plot obtained from wikipedia, see `https://en.wikipedia.org/wiki/Rosenbrock_function`.

2. Try now to use the Newton algorithm and see how many iterations are required to arrive to the minimum. It is interesting to analyze which of both methods (Newton or gradient descent) is used at each iteration. Use two different colors to plot which of both methods is used at each iteration.

Report

You are asked to deliver an *individual* report of section 2 and 3. Just explain each of the steps you have followed. If you want to include some parts of code, please include it within the report. Do not include it as separate files. You may just deliver the Python notebook if you want.

The deadline to deliver this report is October 25th at 3 p.m. (15h).