Visualization of Unconstrained Optimization Algorithms

WEN Zihao, 56136475

May 2020

1 Introduction

A famous computer scientist at Stanford Donald Knuth once said, "An algorithm must be seen to be believed". However, optimization on functions in high dimensional spaces, like those encountered in deep learning, can be hard to visualize. But we can learn a lot from visualizing optimization paths on functions that has simply 2 dimensions. When the dimension of an optimization problem is less than three, one can plot the objective function and constraint set, visually check if it's convex and develop a better intuition to design an optimization algorithm. A visual representation also highlights the "geometry" of convex optimization and helps others to better understand the principle of the optimization technique. In this project, I visualize two unconstrained optimization algorithms, steepest descent and Newton method, respectively, plot them in 2-d and choose three test functions to test their performance.

2 Program Architecture

The demo program is designed as a web page based on html + CSS + javascript + d3.js. As shown in the figure 1, the demo consists of three parts: (from left to right) toolbar, visualization content and description column.

The toolbar can be shown or hidden by clicking the button on the upper left, or simply use the hotkey " ". The toolbar includes some useful tools, for example, you can select different test functions or use different cursor mode. There are two cursor type: one is "click", the other is "move". When the "click" mode is selected, clicking on the graph will start an optimization from the point that is clicked. When the "move" mode is selected, one can drag the graph to move it or use mouse wheel to zoom in and zoom out to see more clearly.

The description column shows some information about the current test function, e.g. its expression, what it looks like and its optimal(s). The math formula is rendered by MathJax.

The most important part of the demo web page is the visualization content. The visualization graph is a svg image drawn by d3.js. It consist of multiple rectangles, whose color indicates the function value at the upper-left corner. Note that the color is assigned according to the relative position of its function value at the function range of given x, y domain, so the color at a certain point may change once we change the domain

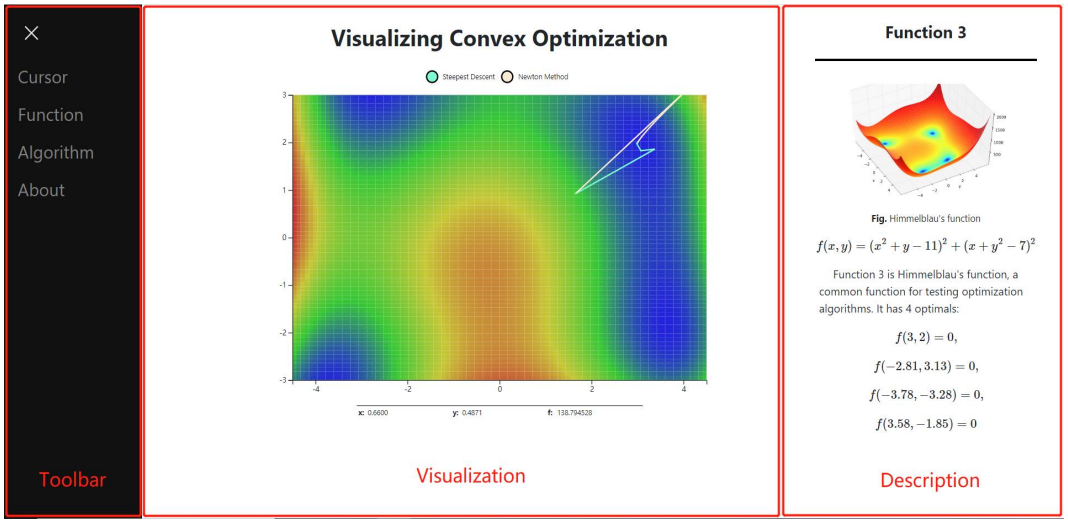


Figure 1: The demo web page

by dragging or zooming. The x, y coordinates and function value at the mouse position is show below the graph.

When "click" mode is activated, which is the default setting of cursor mode, clicking on the graph will start an optimization from that point. Then two optimization paths are drawn and I use transition to indicate the converging more clearly. The transition time is based on the number of iterations to converge. And I create 2 circles on the top of the graph so that one can hide or show the optimization path of a certain algorithm by toggling the circles.

When "move" mode is activated, user can drag or zoom to adjust the domain of the function to see more clearly. Note that the graph will only be rendered again when drag ends, because rendering the whole graph takes some time and it cannot be done in real-time. I also design a reset button at the lower-right corner in case we get lost and cannot go back to the origin. By clicking reset button, the graph will be re-rendered under default settings.

3 Methodology

3.1 Unconstrained Optimization Algorithms

In this project, I implement 2 unconstrained optimization algorithms: steepest descent and Newton method. Steepest descent, first proposed by Cauchy in 1847, is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. To find a local minimum of a function using gradient descent, we take steps proportional to the negative of the gradient (or approximate gradient) of the function at the current point. Algorithm 1 shows the steps of steepest descent. Here I use backtracking line search to search for the appropriate step size t, since exact line search is not suitable for all functions, especially complex ones.

Newton method is an iterative method for finding the roots of a differentiable function F, which are solutions to the equation $F(x) = 0$. In optimization, Newton method is applied to the derivative f' of a twice-differentiable function f to find the roots of the derivative (solutions to $f'(x) = 0$), also known as the stationary points of f. These

Algorithm 1 Steepest Descent

Input: GIVEN a starting point x

while stopping criterion not satisfied **do**
  $\Delta x := -\nabla f(x)$
  Backtracking Line Search: GIVEN $\alpha \in (0, 0.5)$, $\beta \in (0, 1)$, $t := 1$
  while $f(x) - f(x + t\Delta x) < \alpha|\nabla f(x)^T(t\Delta x)|$ **do**
    $t := \beta t$
  end while
  Update $x := x + t\Delta x$
end while

solutions may be minima, maxima, or saddle points. The steps of Newton method is as follows:

Algorithm 2 Newton Method

Input: GIVEN a starting point x and tolerance $\epsilon > 0$

while stopping criterion not satisfied **do**
  Compute Newton step and decrement $\Delta x_{nt} := -\nabla^2 f(x)^{-1}\nabla f(x)$ and $\lambda = (\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x))^{\frac{1}{2}}$
  Stopping criterion: QUIT if $\frac{\lambda^2}{2} \leq \epsilon$
  Line search: choose a step size $t > 0$
  Update $x := x + t\Delta x$
end while

3.2 Test Functions

I chose three functions for testing optimization algorithms. The first function, the simplest one, comes from tutorial 7. Its expression is

$$f(x, y) = \frac{1}{2}(x^2 + 2y^2) \tag{1}$$

It has a bowl-like geometry with only one global optimum:

$$f(0, 0) = 0 \tag{2}$$

The Hessian matrix of the function is positive definite so it is a convex function.

The second function is a function given by the following expression:

$$f(x, y) = x^2 + y^2 - 16e^{-\frac{(x-3)^2+y^2}{1.2}} - 20e^{-\frac{(x+3)^2+y^2}{2}} \tag{3}$$

It is basically a quadratic "bowl" with two gaussians creating minimas at around $(3, 0)$ and $(-3, 0)$. respectively. The size of these minima is controlled by the coefficient of the last two terms. It is not a convex function since it Hessian matrix is not positive definite.

The third function is Himmelblau's function, named after David Mautner Himmelblau (1924–2011), who introduced it. It is a multi-modal function, used to test the performance of optimization algorithms. The function is defined by:

$$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2 \tag{4}$$

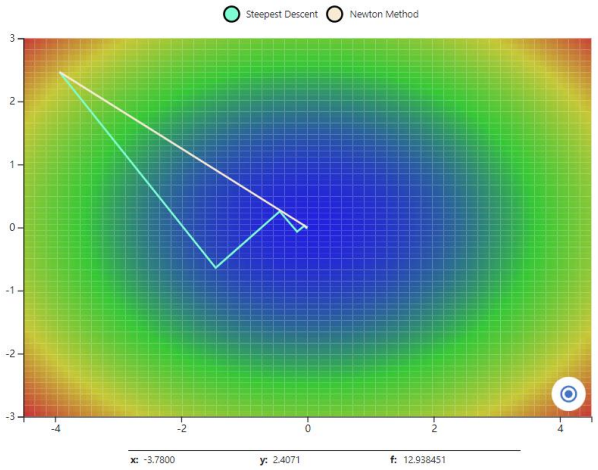


Figure 2: Result on function 1

It has one local maximum at $x = -0.270845$ and $y = -0.923039$ where $f(x, y) = 181.617$, and four identical local minimas:

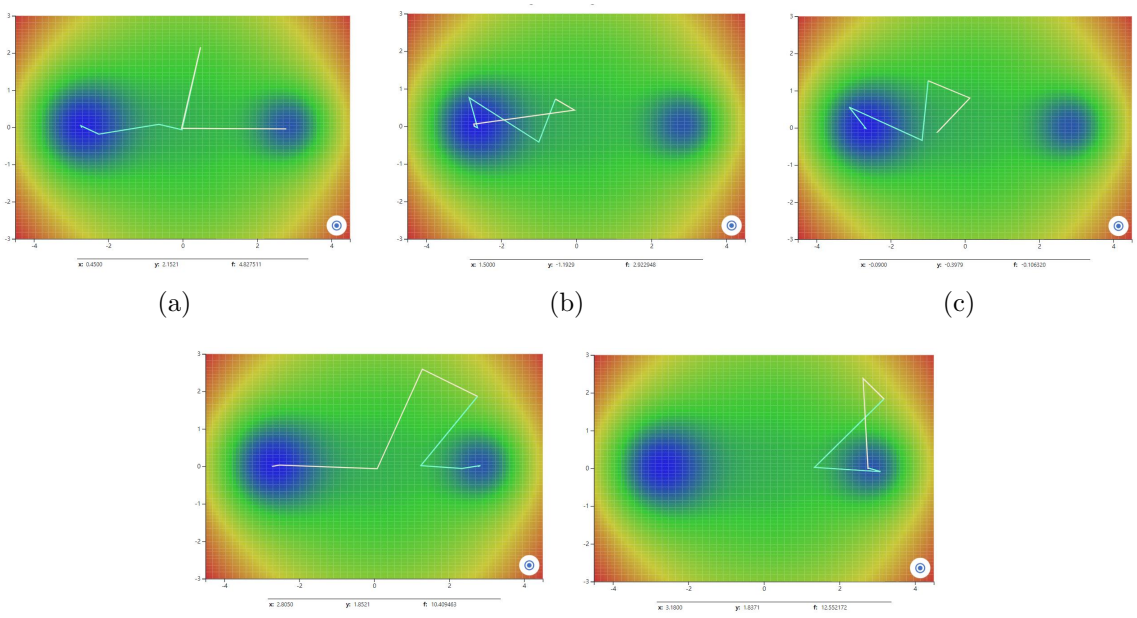
$$\begin{aligned} f(3.0, 2.0) &= 0.0 \\ f(-2.805118, 3.131312) &= 0.0 \\ f(-3.779310, -3.283186) &= 0.0 \\ f(3.584428, -1.848126) &= 0.0 \end{aligned} \tag{5}$$

4 Experiment

I did several test for both algorithms on all three functions and I get some interesting findings. The blue path corresponds to steepest descent and the antique white path corresponds to Newton method. I first test on function 1 which is the simplest one. As we can see in figure 2, both algorithms can converge to global minima, no matter where we start the optimization. Newton method converge faster than steepest descent and its optimization path is a straight line, while steepest descent has a wiggly optimization trajectory.

Things are different on function 2 since it has 2 local optimum and it is not a convex function. I find that in some initial points, the Newton method will stop after one small steps. This is because the Hessian on that point is negative definite. So I try to avoid those points and select the cases that Newton method can converge to a meaningful point. Obviously, steepest descent is more robust in non-convex functions. It can always converge to a local minima and has a high probability to converge to the global minima, the one on the left. Only when the initial point is close to the minima on the right, steepest descent will converge to it. Also, the converging path of steepest descent is wiggly. Relatively speaking, the optimization path of Newton method is less wiggly but sometimes it may converge to some saddle points (in figure 3(c)).

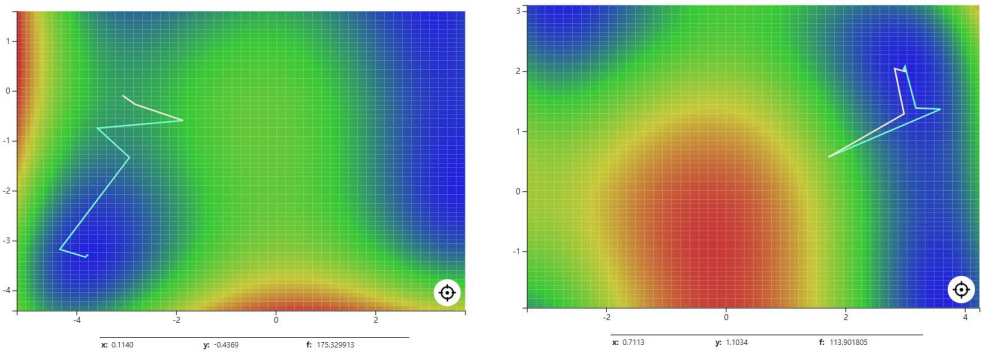
I got similar result on function 3: Steepest descent can always converge to local minima while Newton method may converge to local maxima or saddle points (in figure 4(a)).



(a) (b) (c)

(d) (e)

Figure 3: Result on function 2



(a) (b)

Figure 4: Result on function 3