AMSC808N Final Exam - Problem 1

Guilherme de Sousa (gdesousa@umd.edu)

December 19, 2020

All codes for Problems 1 2 are available on ELMS together with this report submission. The chosen language was Python for simplicity. Additionally, one can find the codes for Final exam Problema 1 and 2 at Guilherme's Github

Question 1.

The goal of this question if to find a optimal solution (a^*, b^*) for the objective function

$$f(a,b) = \frac{1}{12}\sum_{j=0}^{5}\left[ReLU(ax_j - b) - g(x_j)\right]^2 \tag{1}$$

that describes a fitting of $g(x) = 1 - \cos x$ using a $ReLU$ function.

(a) First note that $ReLU(x) = \max(x, 0)$ thus for the regions when the argument $ax_j - b < 0$ we don't have any dependence of a and b for the objective function $f(a,b)$. This is precisely the flat region where $\nabla f = 0$.

To find a precise definition for this flat region set we must find $\{a, b\}$ s.t. $ax_j - b < 0 \quad \forall x_j \in [0, \pi/2]$. After some careful thought we can see that the region of stationary points consists of two regions:

$$\{\nabla f = 0\} = \{(a,b) | (a < 0, b/a < 0) \cup (a > 0, b < a\pi/2)\} \tag{2}$$

it's clear if we plot this (unbounded from below) region

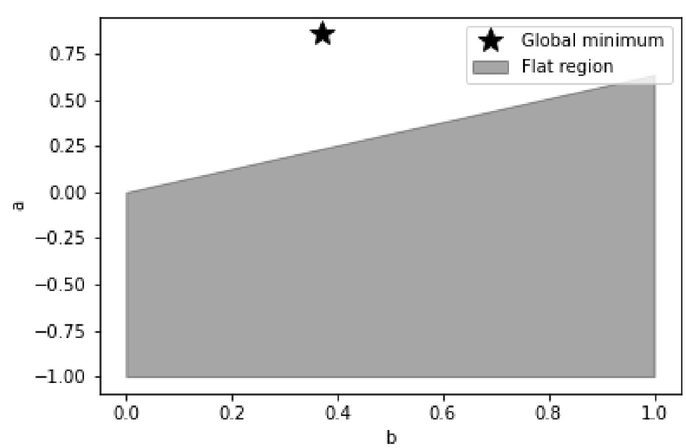

Figure 1: Flat region such that $\nabla f = 0$. Note that it's unbounded for negative values of a and positive values of b. Star denotes the global minimum.

Then, to find the optimal solution (a^*, b^*) we must solve for the minimizer of the objective function

Eq.1. To do this, first note that the solution for a linear least square problem is given by

$$\begin{aligned} f(x) &= ax - b \\ a &= \frac{\langle yx\rangle - \langle y\rangle\langle x\rangle}{\langle x^2\rangle - \langle x\rangle^2} \\ b &= a\langle x\rangle - \langle y\rangle \end{aligned} \tag{3}$$

where $\langle\cdot\rangle$ denotes average over all data and $y(x)$ is the set of points we want to fit using a linear regression $f(x)$.

Now we note that the $ReLU$ function simply cuts off the values of x_j that $ax_j - b < 0$ so effectively we are solving a linear least square problems (as in Eq.3) in those x values that satisfy the cutoff. Finally, we can find the optimal solutions by minimizing the following:

$$(a^*, b^*) = \min_{i=0,\cdots 5} f(a_i, b_i), \quad s.t. \begin{cases} a_i = \dfrac{\sum_{j=i}^5 (x_j - \langle x\rangle)(g(x_j) - \langle g\rangle)}{\sum_{j=i}^5 (x_j - \langle x\rangle)^2} \\ b_i = \dfrac{1}{6-i}\displaystyle\sum_{j=i}^5 (a_i x_j - g(x_j)) \end{cases} \tag{4}$$

Using Eq.(4) we find the optimal solution to be $(a^*, b^*) = (0.86, 0.37)$ and we plotted it together with the function $1 - \cos x$.

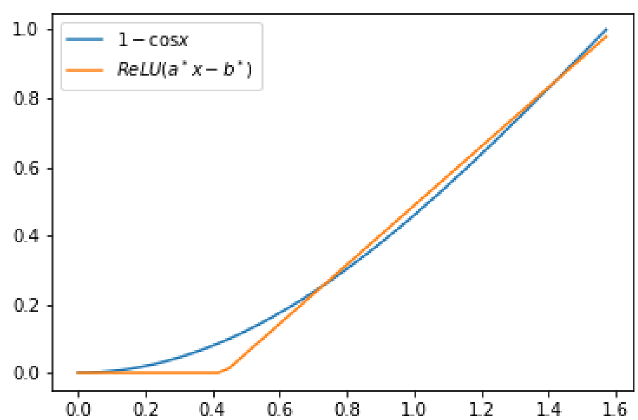

Figure 2: The optimal solution is zero when $g(x)$ is small and approximates kinda good near the endpoints.

(b) To find the smallest stepsize to reach the flat region we need to solve the iteration $(1,0) - \alpha^* \nabla f(1,0) =$ boundary of set $(\nabla f = 0)$. Solving this equation we find $\alpha^* = \frac{1}{\partial_a f(1,0) - \partial_b f(1,0) 2/\pi} = 1.51$. To numerically test this solution I implemented a gradient descent with varius different constant stepsizes $\alpha = [0.1, 1.31, 1.49, 1.51]$ using 1000 iterations.

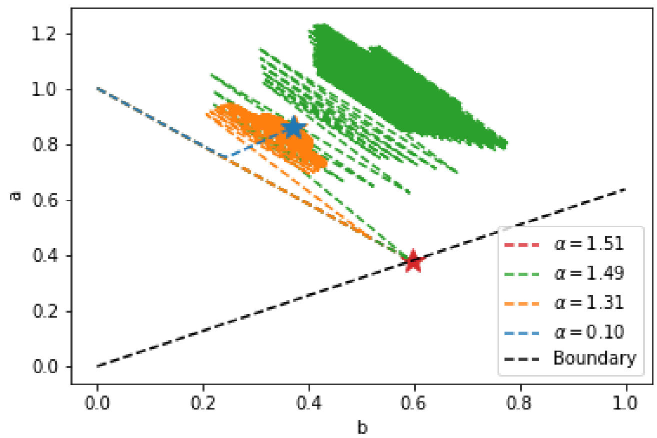

Figure 3: Gradient descent with different stepsizes, black dashed line is the boundary to the flat region. All initial conditions are (1,0) and the final position is denoted by a star. For $\alpha < 1.31 \pm 0.01$ (blue and orange) the method converges to the global minimum (0.86,0.37). Using $\alpha = \alpha^*$ (red) the method reaches the stationary region and stays there. Using $\alpha = 0.99\alpha^*$ (green) the method doesn't reach the flat region and doesn't converge to the global minimum.

From Fig.?? we found numerically that $\alpha < 1.31 \pm 0.01$ is the threshold for convergence to the global minimum. The region $1.31 < \alpha < 1.51$ the method doesn't converge as shown in green. For larger values $\alpha \geq 1.51$ the method reaches the flat region in the first step and stays there.

(c) Then we implemented the stochastic gradient descent with batch size of 1 point. For stepsize strategy I used $\alpha_k = 1.5/2^{k/50}$ to test the convergence to the global minimum. As shown in the next result we see clearly the convergence even if $\alpha = 1.5$ the Gradient Descent doesn't converge.

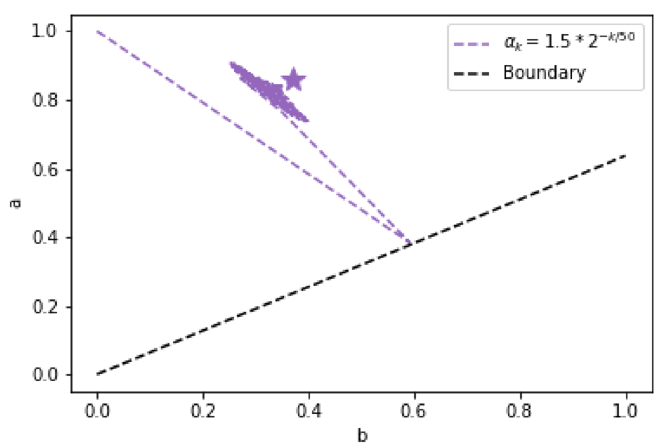

Figure 4: Using stochastic gradient descent the method converges using a exponential decrease $\alpha_k = 1.5/2^{k/50}$, every 50 steps the stepsize get smaller by a factor of 1/2.