Tutorial 9 : Gradient Descent

1 Unconstrained Optimization

Find the point x^* for which the function $f(x)$ is minimal

$$f(x^*) \leq f(x) \; \forall x \in \omega.$$

Usually $\Omega = \mathbb{R}^n$, i.e., $f(x) : \mathbb{R}^n \to \mathbb{R}$ with $x = (x_1, x_2, ..., x_n)^T$ where x is not constrained and can get any value.

Optimality Conditions

Goal

$$x^* = \arg\max_{x \in \mathbb{R}^n} f(x).$$

Assuming f is differentiable, a necessary condition for local minima of f is

$$\nabla f(x^*) = 0.$$

When f is twice differentiable, a necessary condition for local minima of f is

$$\nabla f(x^*) = 0 \text{ and } H(x^*) \succeq 0.$$

The last inequality means that the Hessian is a positive semi-definite matrix (meaning that all the eigenvalues of H are non-negative). When $H \succ 0$ the the condition becomes sufficient.

Gradient Descent

In most cases we cannot find x^* analytically or it is too computational expensive to compute. Therefore, we use an iterative solution where we start with an initial guess x_0 and create a sequence

$$x_{k+1} = x_k + d_k$$

which satisfies

$$f(x_{k+1)} \leq f(x_k).$$

The Algorithm

1. Initialization: Set the initial value x_0.
2. Update
$$x_{k+1} = x_k - \eta \nabla f(x_k).$$
3. Repeat step (2) until convergence:
   - $||x_{k+1} - x_k||_2 \leq \epsilon.$
   - $\Big(f(x_{k+1}) - f(x_k)\Big)^2 \leq \epsilon.$
   - $||\nabla f(x_k)|| \leq \epsilon.$

Explanation of the update rule:
The gradient algorithm is based on the first order Taylor expansion of $f(x)$

$$f(x_k + d) \approx f(x_k) + \nabla f(x_k)^T d.$$

Hence, for $d_k \triangleq -\eta \nabla f(x_k)$ where $\eta > 0$ we have

$$f(x_{k+1}) = f\Big(x_k - \eta \nabla f(x_k)\Big) = f(x_k) - \eta f(x_k)^T f(x_k) = f(x_k) - \eta ||f(x_k)||_2^2 \leq f(x_k)$$

The algorithm converges to a stationary point (extremum/saddle) where it hold that $\nabla f(x^*) = 0$.
When η is small the approximation is valid, but convergence rate might be small.
When η is large the approximation is invalid, convergence is not guaranteed.

Question 1

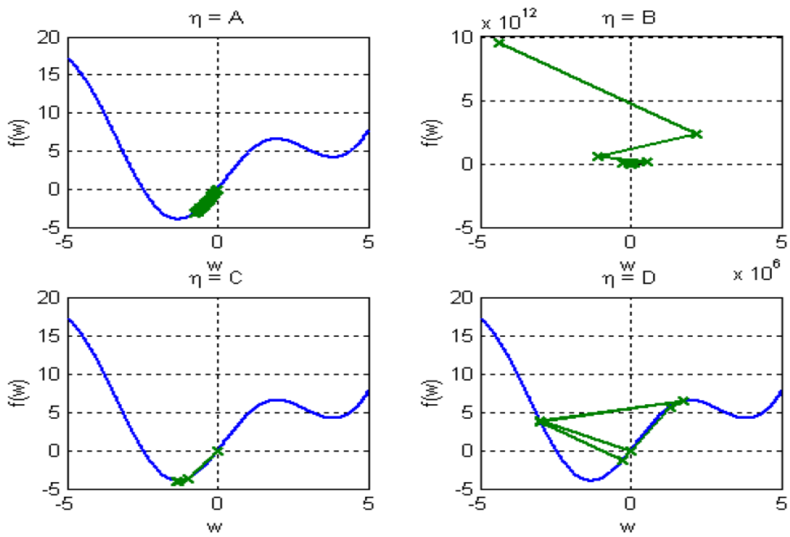
The following cost function is given: $f(w) = \frac{1}{2}w^2 + 5\sin(w)$.

(a) What is a necessary condition for a minimum point?

(b) Write down the update step of gradient descent for this problem.

(c) Calculate two iterations, for initial guess $w_0 = 0$ and step size $\eta = 0.2$.

(d) The following graphs show ten iterations of the gradient algorithm, for 4 different values of the step size, $\eta \in \{0.01, 0.2, 0.6, 3\}$. Match size to graph.



Solution

(a) The necessary condition for a minimum point is

$$\frac{df}{dw} = w + 5\cos(w) = 0.$$

(b) The update step is given by

$$w_{k+1} = w_k - \eta\Big(w_k + 5\cos(w_k)\Big) = (1-\eta)w_k - t\eta\cos(w_k).$$

(c) $w_0 = 0,\ \eta = 0.2.$

- First iteration:

$$w_1 = w_0 - \eta\Big(w_0 + 5\cos(w_0)\Big) = 0 - 0.2 \cdot 5 = -1.$$

- Second iteration

$$w_2 = w_1 - \eta\Big(w_1 + 5\cos(w_1)\Big) = -1 - 0.2 \cdot 1.7015 = -1.3403.$$

(d)
- Small step size - slow but sure convergence.
- Large step size - Large movements.
- Too big step size - divergence.

Therefore,

A $\eta = 0.01$.

B $\eta = 3$.

C $\eta = 0.2$.

D $\eta = 0.6$.

2 Linear Regression

Consider a linear classifier with input $x \in \mathbb{R}^d$ and output $y \in \mathbb{R}$ computed as follows

$$\varphi\left(\sum_{k=1}^{d} w_k x_k + b\right) = \varphi\left(w^T x + b\right),$$

where activation function $\varphi(\cdot)$ sets the type of the classifier. Typically x and w are extended to include the bias term such that $x_0 = 1$ and $w_0 = b$. Thus, we can write

$$\varphi\left(\sum_{k=1}^{d} w_k x_k + b\right) = \varphi\left(\sum_{k=0}^{d} w_k x_k\right) = \varphi\left(w^T x\right).$$

Common activation functions:

- Linear Perceptron - $\varphi(v) = sign(v)$.
- Logistic Activation - $\varphi(v) = \frac{1}{1+\exp(-v)}$.
- Hyperbolic Tangent Activation - $\varphi(v) = tanh(v) = \frac{e^{2v}-1}{e^{2v}+1}$.

Gradient Descent Based Learning Algorithm

The model for the classifier is $\hat{f}_w(x) = \varphi(w^T x)$. Given a labeled training set $D = \{x_i, y_i\}_{i=1}^n$, we define the loss function as

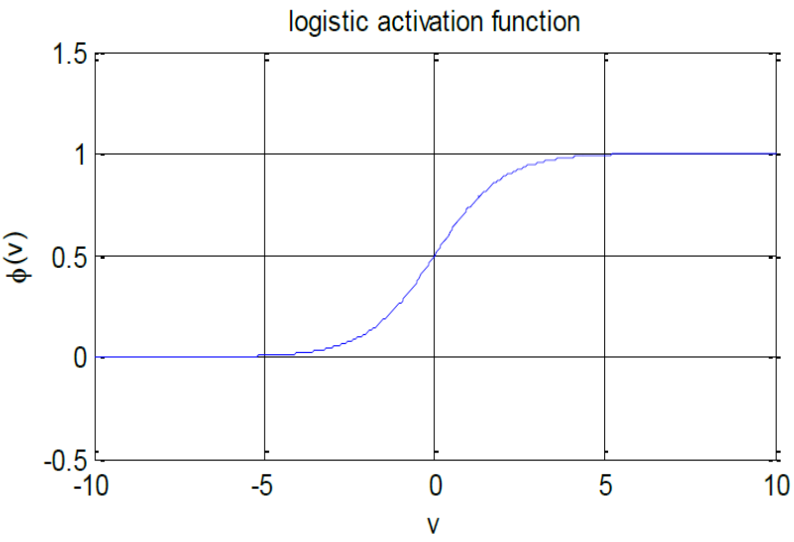
$$L(w) = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n}\Big(y_i - \hat{f}_w(x_i)\Big)^2.$$

The algorithm minimizes the loss function by performing gradient descent:

- Set initial value for the weights.
- Update the weight iteratively -
  (i) Online update - $w_{t+1} = w_t + \eta\Big(y_t - \hat{f}_w(x_t)\Big)\varphi'(w_t^T x_t)x_t$.
  (ii) Batch update - $w_{t+1} = w_t + \eta\sum_{i=1}^{n}\Big(y_i - \hat{f}_w(x_i)\Big)\varphi'(w_t^T x_i)x_i$.

Question 2

The logistic activation function $\varphi(v) = \frac{1}{1+\exp(-v)}$ is a smooth version of the Boolean activation function, and its graph is given in the following plot:



(a) Write down the online update rule for the logistic activation function.

(b) Consider a linear classifier with two input $x_1, x_2 \in [100, 200]$. The output should be binary $\hat{y} \in \{0, 1\}$. Set the initial weights to $w_1 = w_2 = 1$ and the step size to $\eta = 0.1$.

(a) At each iteration we update the weights as

$$w_{t+1} = w_t + \Delta w_t.$$

Find an upper bound to the size of the update $|\Delta w_t|$ in the first step of an online update algorithm.

(b) What is your conclusion regarding the learning rate?

(c) What is the cause of the problem? Suggest a solution.

Solution

(a) We consider the logistic activation function, hence,

$$\varphi(v) = \frac{1}{1+\exp(-v)}, \; \varphi'(v) = \frac{\exp(-v)}{\big(1+\exp(-v)\big)^2}.$$

Thus, the online update rule is

$$w_{t+1} = w_t + \eta\big(y_t - \varphi(v_t)\big)\varphi'(v_t)x_t = w_t\eta\left(y_t - \frac{1}{1+\exp(-v_t)}\right)\frac{\exp(-v_t)}{\big(1+\exp(-v_t)\big)^2}x_t,$$

where $v_t = w_t^T x_t$.

(b) Notice that

$$|\Delta w_t| \leq \eta\Big(\max_x y_t - \varphi(v_t)\Big)\Big(\max_x \varphi'(v_t)\Big)\Big(\max_x x_t\Big)$$

- $\max_x x_t = 200.$
- $0 \leq \varphi(v_t) \leq 1, \; y_t \in \{0, 1\} \;\Rightarrow\; \max_x y_t - \varphi(v_t) \;\leq 1.$
- $\max_x \varphi'(v_t) = \max_x \left(\frac{\exp(-w_t^T x_t)}{\big(1+\exp(-w_t^T x_t)\big)^2}\right) = \max_x \left(\frac{\exp(-x_1-x_2)}{\big(1+\exp(-x_1-x_2)\big)^2}\right) = \frac{exp(-200)}{\big(1+exp(-200)\big)^2} \approx 10^{-87}.$

(c) The update is extremely small which implies that practically there will be no update (the learning rate is zero).

(d) The cause for the problem is the fact that v_t is in the area where $\varphi(v_t)$ is saturated and the derivative there is approximately zero. To overcome this we can normalize the weight or the inputs such as $\tilde{x}_i = \frac{x_i - 150}{50}$. This ensure that in the first iterations the linear classifier will be in the linear phase of the activation function and it won't get "stuck".