Gradient Descent

Gradient Descent

Optimization Problem. One of the most common problems in numerical analysis is the problem of optimizing a multivariate fully-diffirentiable function:

> *Let $L(x_1, \ldots, x_n)$ be a multivariate differentiable function*
>
> $$L : \mathbb{R}^n \longrightarrow \mathbb{R}.$$
>
> *Find all (or some) points $\mathbf{x} \in \mathbb{R}$ that **minimize** the value of $L()$.*

General approach. To minimize a multivariate function we need to solve the system of equations:

$$\begin{cases} \frac{\partial L}{\partial x_1} = 0 \\ \frac{\partial L}{\partial x_2} = 0 \\ \ldots \\ \frac{\partial L}{\partial x_n} = 0 \end{cases}$$

Lack of analytical solutions. For many shapes of $L(x_1, \ldots, x_n)$ simple analytical solutions to the system of equations above do not exist.

Complexity of analytical solutions. For other situations, analytical solutions do exist, but they are expensive.

Iterative procedures. This problem is often approached from a numerical analysis perspective:

Given $L(x_1, \ldots, x_n)$ a multivariate differentiable function

$$L : \mathbb{R}^n \longrightarrow \mathbb{R},$$

approximate *the locations $\mathbf{x}$ where $L()$ is minimized.*

Gradient Descent. Gradient descent is an iterative procedure for optimizing $L()$. It works as follows.

1. Step 0. Pick a *learning rate* $\eta > 0$. Pick an arbitrary point $\mathbf{x}^(\mathbf{0}) \in \mathbb{R}^n$.
2. Step i+1. Let $\mathbf{x}^(\mathbf{i})$ be the approximation constructed on Step i of the process. Construct new approximation, $\mathbf{x}^{(\mathbf{i+1})}$ as follows:

$$\mathbf{x}^{(\mathbf{i+1})} = \mathbf{x}^{(\mathbf{i})} - \eta \nabla L(\mathbf{x}^{(\mathbf{i})})$$

3. Stopping. Different stoppage conditions can be used. For example, this process can terminate when

$$|L(\mathbf{x}^{(\mathbf{i})}) - L(\mathbf{x}^{(\mathbf{i+1})})| < \varepsilon,$$

for some value $\varepsilon > 0$ selected in advance.

Properties of Gradient Descent. Gradient Descent method has the following properties:

- If the learning rate parameter η is selected appropriately, gradient descent is guaranteed to converge to a **local minimum**.
- If the function $L()$ has multiple local minima, selection of the starting point $\mathbf{x}^{(\mathbf{0})}$ will determine the specific local minimum discovered.
- Each step of the gradient descent process is cheap. Given a training set $X = \{\mathbf{x_1}, \ldots, \mathbf{x_m}\}$ of data points and a function $L() = L(X)$, each step of the gradient descent process takes $O(mn)$ time.

Issues with Gradient Descent.

- Selection of the learning rate parameter η.
- Number of iterations.
- Arriving to a local optimum rather than a global one (i.e., different starting points lead to different answers).

Gradient Descent for Linear Regression. Here is a simple example of how gradient descent can be applied to a problem of optimizing the sum squared errors in the linear regression case.

Recall: given a dataset $X = \{\mathbf{x_1}, \ldots, \mathbf{x_m}\}$, and the values of a dependent variable $Y = \{y_1, \ldots, y_m\}$, where $\mathbf{x_i} \in \mathbb{R}^n$, we want to represent our prediction function $f()$ as a linear combination:

$$f(\mathbf{x}) = \beta_1 x_1, + \ldots + \beta_n x_n + \beta_0 = \mathbf{x}^T \beta,$$

where the vector $\beta = (\beta_0, \beta_1, \ldots, \beta_n)$ minimizes the function

$$L(\beta) = \sum_{i=1}^{m} (y_i - (\beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n))^2.$$

We know that

$$\frac{\partial L}{\partial \beta_j} = -2x_j(y_j - (\beta_0 + \beta_1 x_i + \ldots + \beta_n x_n))$$

From here:

$$\nabla L(\beta) = -2X^T(\mathbf{y} - X\beta)$$

This gives rise to the following straightforward gradient descent procedure:

1. $\beta^{(\mathbf{0})} = (0, 0, \ldots, 0)$
2. $\beta^{(\mathbf{i+1})} = \beta^{(\mathbf{i})} + 2\eta X^T(\mathbf{y} - X\beta^{(\mathbf{i})})$

Stochastic Gradient Descent

Gradient Descent on Training Sets. When gradient descent is used in machine learning scenarios to optimize the error function over the training set, often times, the objective function $L()$ can be represented as the sum of errors from individual points in the training set:

$$L(\beta) = \sum_{i=1}^{m} L_i(\beta),$$

where $L_i(\beta)$ is the error of prediction for data point $\mathbf{x_i}$.

In such situations, a simplified version of gradient descent is available called stochastic gradient descent.

Stochastic Gradient Descent. Consider the minimization problem for a function

$$L(\beta) = \sum_{i=1}^{m} L_i(\beta)$$

Applying gradient descent to such a function gives us the following iteration schema:

$$\beta^{(i+1)} = \beta^{(\mathbf{i})} - \eta \nabla L(\beta^{(\mathbf{i})}) = \beta^{(\mathbf{i})} - \eta \sum_{j=1}^{m} \nabla L_j(\beta^{(\mathbf{i})})$$

Stochastic Gradient Descent replaces the computation of the full gradient with the step-wise computation: the full gradient is approximated by each of its $L_j()$ components one at a time.

The schema looks as follows:

$$\beta^{(\mathbf{0})} = (0, 0, \ldots, 0)$$

$$\beta^{(\mathbf{i+1})} = \beta^{(\mathbf{i})} - \eta \nabla L_{(i+1) \mathbf{mod} m}(\beta^{(\mathbf{i})})$$