Discussion 7 – Gradient Descent

Data 100, Spring 2020

Suraj Rampure

Friday, March 6th, 2020

Minimizing Loss

Recall, in order to find the optimal value of our parameter $\hat{\beta}$, we need to find the value of β that minimizes our average loss, $L(\beta)$. $\rightarrow \frac{1}{n}\sum_{i=1}^{n}(\cdots)$

- Different choices of loss functions will lead to different values of $\hat{\beta}$.

$L_2: (y-\hat{y})^2$

$L_1: |y-\hat{y}|$

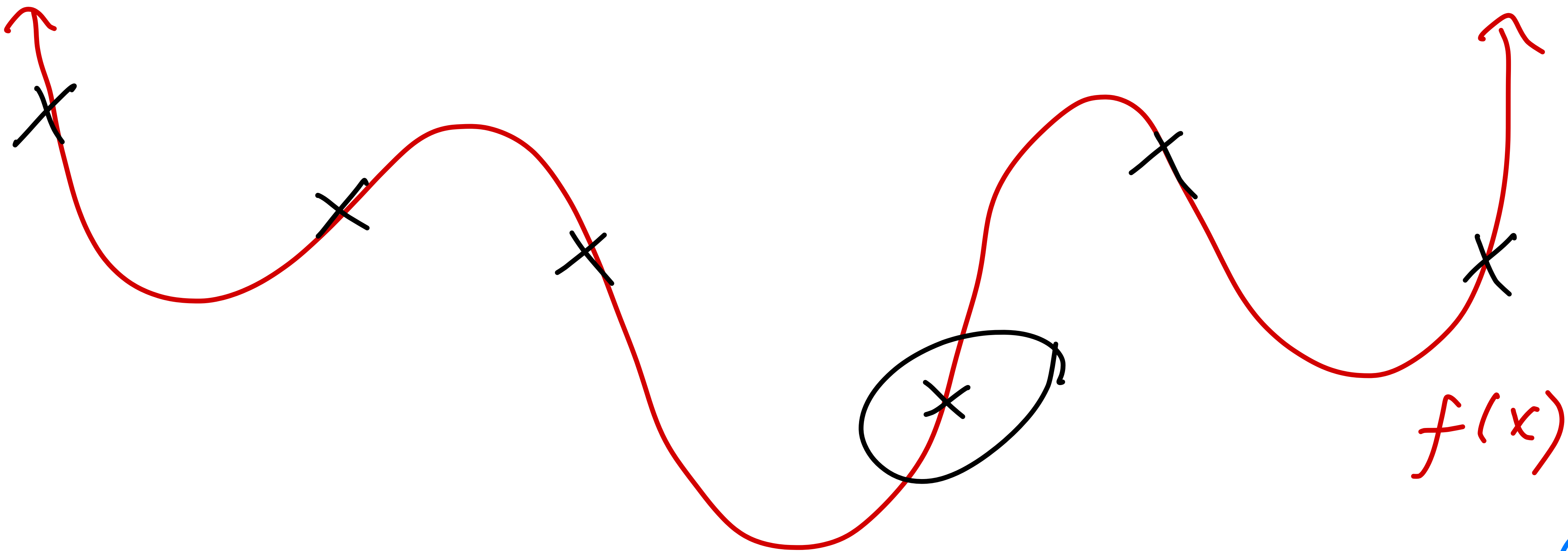
Sometimes, we're able to find an **analytical** solution for the minimizing value of $\hat{\beta}$.

- For instance, for simple linear regression where our model is $\hat{y}_i = \beta_0 + \beta_1 x_i$, from Data 8 we know that $\hat{\beta}_1 = r\frac{SD(y)}{SD(x)}$, and $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$.
- As we look at more and more complex loss functions, though, this becomes less common, and so we need to look at **numerical techniques** (like gradient descent).

Minimizing Loss – A Naive Approach

Suppose we want to find the value x^* that minimizes $f(x)$ (where x is either a scalar or vector).

Let's suppose we don't know what gradient descent is, and that we also can't compute the derivative/gradient of f, so we can't set it equal to 0 and solve. **How else can we estimate x^*?**



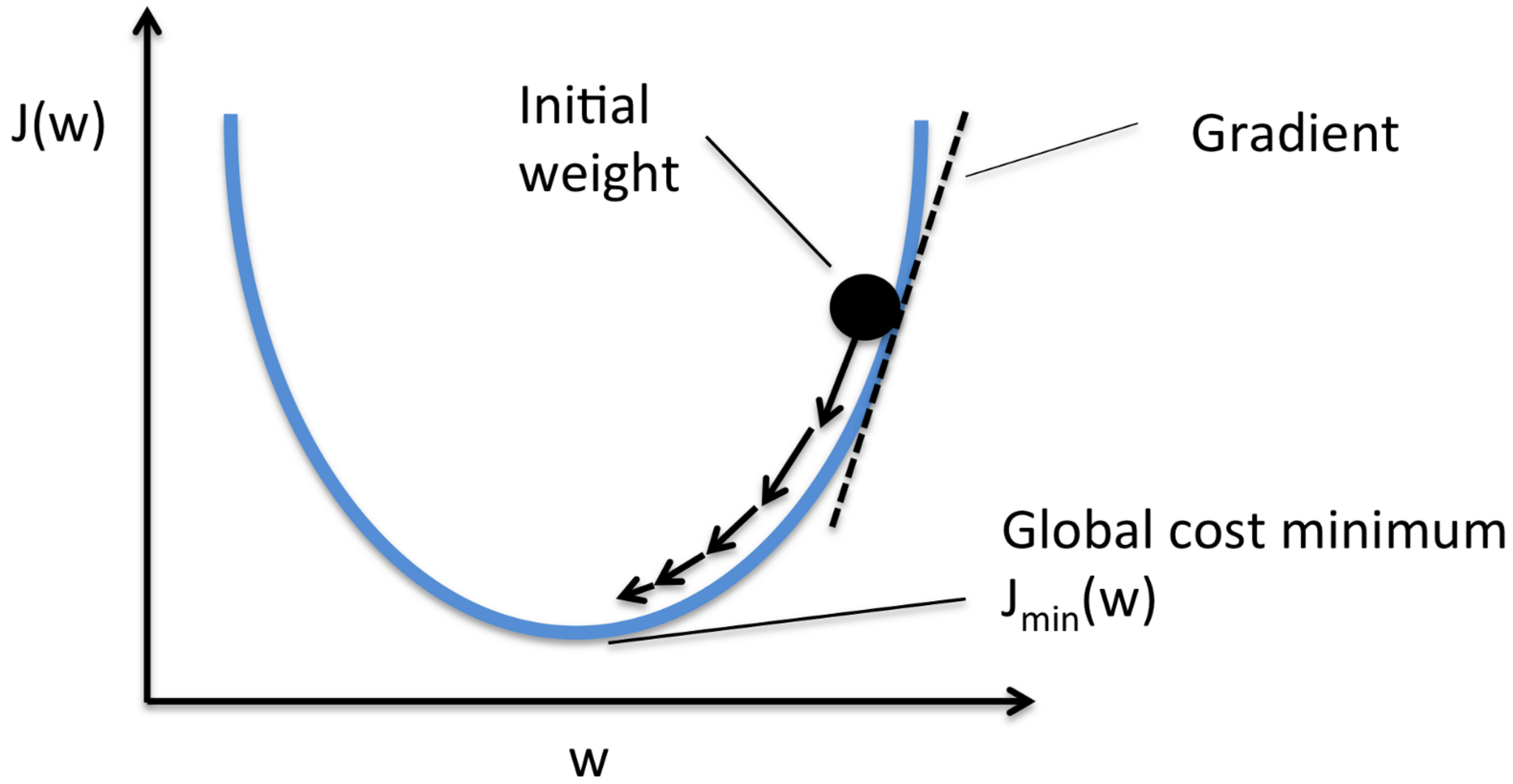
$\Rightarrow$ Sample several x-values, pick x that gives minimum $f(x)$

$\rightarrow$ no guarantee that you select true minimum

Gradient Descent

Goal: Identify the global minimum of a function.

- We know that any minimum of a function occurs where the gradient is 0.
- We also know that **gradients point in the direction in which a function is increasing**.
- Hence, gradient descent tries to find the point at which the gradient is 0, by **moving in the opposite direction of the gradient**, iteratively.



Gradient descent update equation

$$\beta^{(t+1)} = \beta^{(t)} - \alpha \nabla L(\beta^{(t)})$$

learning rate

"how big of a step to take"

decaying LR: gets smaller as t increases

(smaller steps when near to minimum)

simple linear regression

$$\hat{y}_i = \underline{\beta_0} + \underline{\beta_1} x_i$$

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$$

$$\nabla L(\vec{\beta}) = \begin{bmatrix} \frac{\partial L}{\partial \beta_0} \\ \frac{\partial L}{\partial \beta_1} \end{bmatrix} \qquad \vec{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

Links to Demos

- https://www.benfrederickson.com/numerical-optimization/
- https://alykhantejani.github.io/images/gradient_descent_line_graph.gif