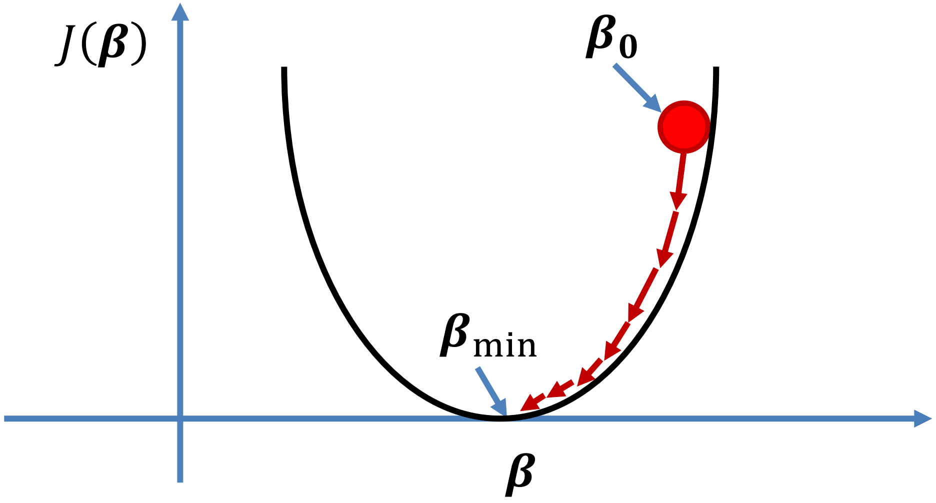
Modern Machine Learning
Gradient Descent

Kenneth E. Barner
Department of Electrical and Computer Engineering
University of Delaware

Gradient Descent

Objective: Optimize $h(\boldsymbol{\beta})$ with respect to parameters $\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_n]^T$

Methodology: Define a cost function $J(\boldsymbol{\beta})$ and adjust $\boldsymbol{\beta}$ until a minimum is reached



$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^{k} - \alpha \frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

$J(\boldsymbol{\beta}) \coloneqq$ Cost function

Assumptions: $J(\boldsymbol{\beta})$ is differentiable and convex

Define the following model

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}_{n\times 1} \quad \boldsymbol{X} = \begin{pmatrix} | & - & \mathbf{z}_1 & - \\ 1 & \vdots & \vdots & \vdots \\ | & - & \mathbf{z}_n & - \end{pmatrix}_{n\times(p+1)} \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}_{(p+1)\times 1}$$

The goal is to solve

$$\widehat{\boldsymbol{y}} = \mathbf{X}\boldsymbol{\beta}$$

Notice that we use the same linear regression model but here we work with the observations, $\mathbf{z} \in \mathbb{R}^p$, instead of variables (features), $\mathbf{x} \in \mathbb{R}^n$

To solve the linear regression problem we define the cost function $J(\boldsymbol{\beta})$ as

$$J(\boldsymbol{\beta}) = \frac{1}{2}\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 = \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 = (\mathbf{X}\boldsymbol{\beta} - \mathbf{y})^T(\boldsymbol{\beta}\mathbf{X} - \mathbf{y})$$

$$= \frac{1}{2}\left((\mathbf{X}\boldsymbol{\beta})^T - \mathbf{y}^T\right)(\mathbf{X}\boldsymbol{\beta} - \mathbf{y})$$

$$= \frac{1}{2}(\boldsymbol{\beta}^T\mathbf{X}^T - \mathbf{y}^T)(\mathbf{X}\boldsymbol{\beta} - \mathbf{y})$$

$$= \frac{1}{2}\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{y} + \frac{1}{2}\mathbf{y}^T\mathbf{y}$$

Define $\mathbf{R} = \mathbf{X}^T\mathbf{X}$. Note $\mathbf{R}$ is symmetric and positive semi-definite (positive eigenvalues)

Next, obtain a closed form expression of $\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ for the gradient descent derivation

$$\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \frac{\partial}{\partial \boldsymbol{\beta}}\left(\frac{1}{2}\boldsymbol{\beta}^T\mathbf{R}\boldsymbol{\beta} - \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{y} + \frac{1}{2}\mathbf{y}^T\mathbf{y}\right)$$
$$= \mathbf{R}\boldsymbol{\beta} - \mathbf{X}^T\mathbf{y} \qquad (*)$$

Result: the steepest descent update that solves the linear regression problem can be written as

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k - \alpha(\mathbf{R}\boldsymbol{\beta}^k - \mathbf{X}^T\mathbf{y})$$

Note: $(*)$ utilizes the results

$$\frac{\partial}{\partial \boldsymbol{\beta}}\boldsymbol{\beta}^T\mathbf{R}\boldsymbol{\beta} = \left(\mathbf{R}^{\boldsymbol{T}} + \boldsymbol{R}\right)\boldsymbol{\beta} = 2\mathbf{R}\boldsymbol{\beta}, \text{since } \mathbf{R} \text{ is symmetric}$$

$$\frac{\partial}{\partial \boldsymbol{\beta}}\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{y}$$

Gradient Descent

Objective: Determine a bound on the step size α that guarantees convergence

Recall $\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{R}^{-1}\mathbf{X}^T\mathbf{y}$ is the optimal coefficient vector

Now express the steepest descend update rules as

$$\begin{aligned}\boldsymbol{\beta}^{k+1} &= \boldsymbol{\beta}^k - \alpha\left(\mathbf{R}\boldsymbol{\beta}^k - \mathbf{X}^T\mathbf{y}\right)\\ &= \boldsymbol{\beta}^k - \alpha\mathbf{R}\left(\boldsymbol{\beta}^k - \mathbf{R}^{-1}\mathbf{X}^T\mathbf{y}\right)\\ &= \boldsymbol{\beta}^k - \alpha\mathbf{R}\left(\boldsymbol{\beta}^k - \boldsymbol{\beta}^*\right)\end{aligned}$$

Rearrange to evaluate the weight error at each iteration

$$\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^* = (\boldsymbol{I} - \alpha\mathbf{R})\left(\boldsymbol{\beta}^k - \boldsymbol{\beta}^*\right)$$

$$\boldsymbol{\epsilon}^{k+1} = (\boldsymbol{I} - \alpha\mathbf{R})\boldsymbol{\epsilon}^k,$$

Observation: for convergence it must hold that $\boldsymbol{\epsilon}^{k+1} \to 0$ in the limit (weight error goes to 0 as $k \to \infty$)

Gradient Descent

Using eigendecomposition: $\mathbf{R} = \mathbf{Q}\mathbf{\Omega}\mathbf{Q}^T$, where

$\mathbf{\Omega}$ is a diagonal matrix containing the eigenvalues of $\mathbf{R}$

$\mathbf{Q}$ is a unitary matrix ($\mathbf{Q}^T = \mathbf{Q}^{-1}$)

The weight error can now be expressed as:

$$\begin{aligned}
\boldsymbol{\epsilon}^{k+1} &= (\boldsymbol{I} - \alpha\mathbf{Q}\mathbf{\Omega}\mathbf{Q}^T)\boldsymbol{\epsilon}^k \\
\mathbf{Q}^T\boldsymbol{\epsilon}^{k+1} &= (\mathbf{Q}^T - \alpha\mathbf{Q}^T\mathbf{Q}\mathbf{\Omega}\mathbf{Q}^T)\boldsymbol{\epsilon}^k \\
&= (\boldsymbol{I} - \alpha\mathbf{\Omega})\mathbf{Q}^T\boldsymbol{\epsilon}^k \\
\mathbf{v}^{k+1} &= (\boldsymbol{I} - \alpha\mathbf{\Omega})\mathbf{v}^k,
\end{aligned}$$

where the transformed error is $\mathbf{v}_{k+1} = \mathbf{Q}^T\boldsymbol{\epsilon}_{k+1}$

Observation: $\boldsymbol{\epsilon}^{k+1} \rightarrow 0$ is equivalent to $\mathbf{v}^{k+1} \rightarrow 0$

Also $\mathbf{\Omega}$ is diagonal $\Rightarrow$ the individual components of $\mathbf{v}$ can be expressed as

$$v_j^{k+1} = (1 - \alpha\lambda_j)v_j^k,$$

where λ_j are the eigenvalues of matrix $\mathbf{R}$

Using recursion we have that

$$v_j^{k+1} = \left(1 - \alpha\lambda_j\right)^k v_j^0$$

Therefore for the error to decrease we need that

$$\left|1 - \alpha\lambda_j\right| < 1, \text{for all } j$$

Result: Since the eigenvalues are nonnegative, the step size α must satisfy

$$0 < \alpha < \frac{2}{\lambda_{\max}}$$

in order to guarantee convergence

Consider the component–wise update case:

The output, given a set of coefficients, is

$$h_{\boldsymbol{\beta}}(\boldsymbol{z}_i) = \beta_0 + z_{i1}\beta_1 + z_{i2}\beta_2 + \cdots + z_{ip}\beta_p$$

The cost function as

$$J(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{n}\left(h_{\boldsymbol{\beta}}(\boldsymbol{z}_i) - y_i\right)^2$$

Differentiating with respect to a single weight component, β_j

$$\frac{\partial J(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n}\left(h_{\boldsymbol{\beta}}(\mathbf{z}_{\mathrm{i}}) - y_i\right) z_{ij}$$

Gradient Descent

The gradient descent algorithm can be written as

$$\beta_j^{k+1} = \beta_j^k - \alpha \sum_{i=1}^{n} \left(h_{\boldsymbol{\beta}^k}(\mathbf{z}_{\mathrm{i}}) - y_i \right) z_{ij}$$

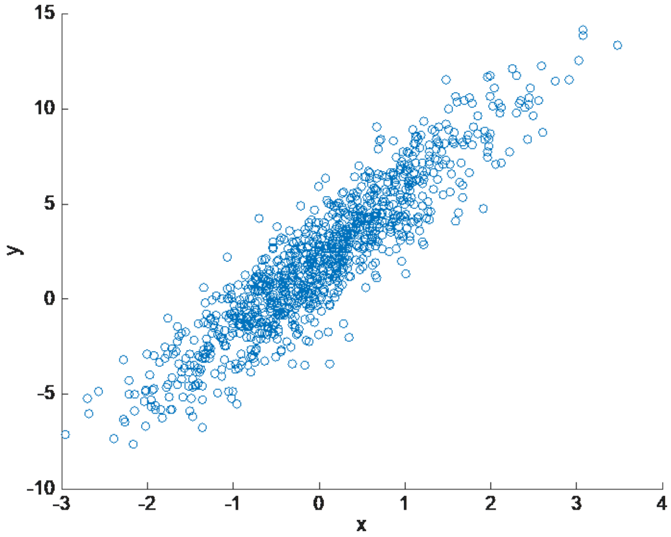
Stopping Criteria: We can assume convergence has been achieved when a certain number of iterations is reached or when

$$\left\| \boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k \right\|_2^2 < \delta,$$

where δ is a small positive constant.

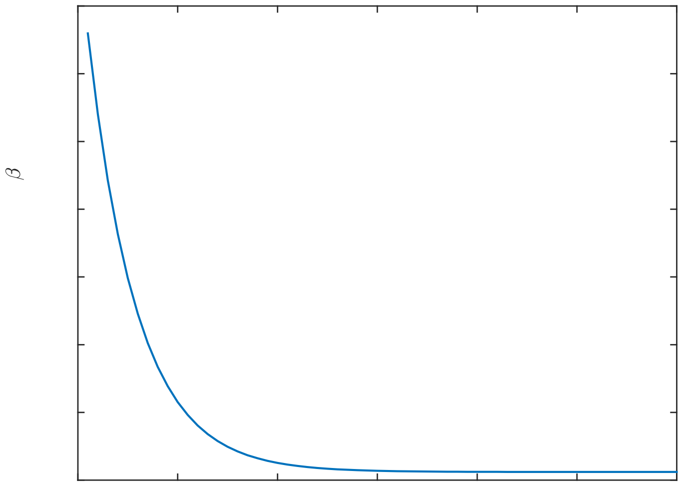
Example:

- Suppose we have a linear model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\eta}$, where $\boldsymbol{\eta}$ is $\mathcal{N} \sim (0, \sigma^2)$.
- We randomly generate the observations $\mathbf{X}$, and select the coefficients $\boldsymbol{\beta} = [2, 3.5]$, and the noise variance $\sigma^2 = 2.25$.
- For display purposes, consider the second dimension components: $x_2, \hat{\beta}_2 \in \mathbb{R}$

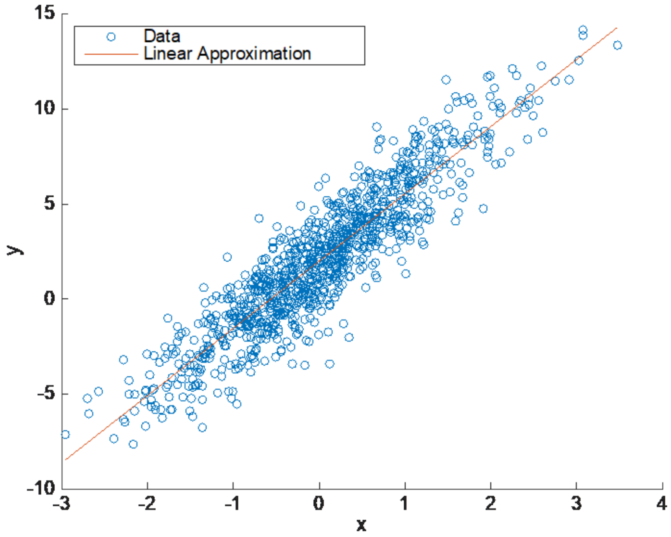


Data generated using the model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\eta}$

Results



Cost function $J(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{n}\left(h_{\boldsymbol{\beta}}(\boldsymbol{z}_i) - y_i\right)^2$ vs iteration number



Data and its linear approximation for the model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\eta}$

Gradient Descent

- Need to choose α
- May require many iterations
- Computationally simple (each iteration)

Closed Form Solution

- No need to choose α
- No need for iterations
- Matrix inversion may be computationally intensive