AGENDA

- Introduction
- Moving along the gradient
- Gradient descent and learning rate
- Application to linear regression

INTRODUCTION

What is gradient descent?

Gradient descent is an optimization algorithm used to find the values of parameters (coefficients) of a function *(f)* that minimizes a cost function (cost).

When do we use it?

Gradient descent is best used when the parameters cannot be calculated analytically (e.g. using linear algebra) and must be searched for by an optimization algorithm.

Fundamental calculus behind gradient descent

Gradient descent is based on the observation that if the multi-variable function *F(x)* is defined and differentiable in the neighbourhood of a point a, then *F(x)* decreases fastest if one goes from a in the direction of the negative gradient of *F(x)* at *a* :

$$-\nabla F(a)$$

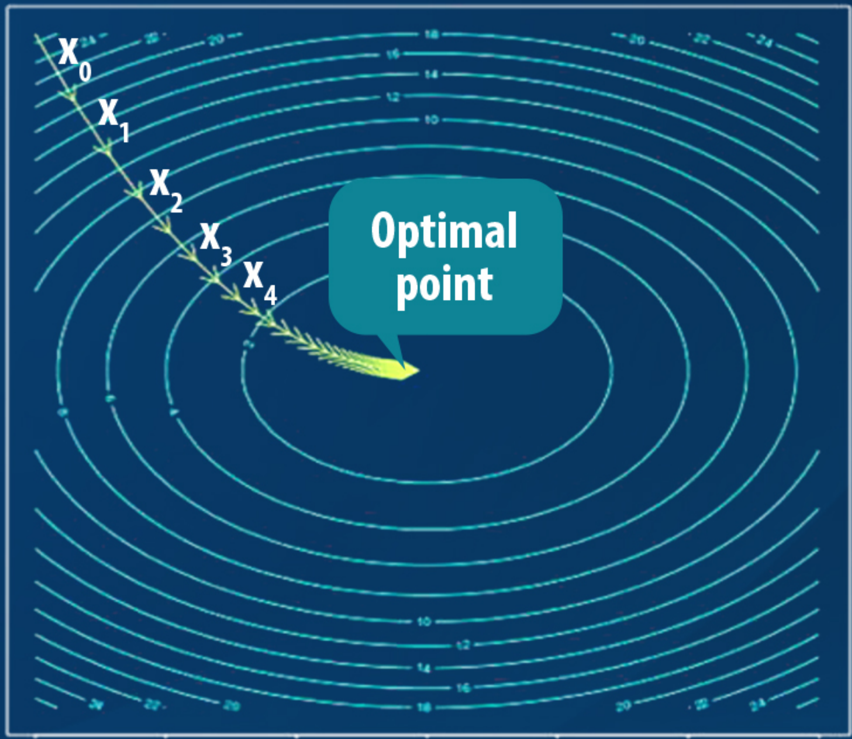


Illustration of gradient descent on a series of level sets.

MOVING ALONG THE GRADIENT

If we were to move from a_n to a_{n+1}

$$a_{n+1} = a_n - \Upsilon \nabla F(a_n)$$

Learning rate

Thus,

$$F(a_n) \geq F(a_n+1).$$

With this observation in mind :

One starts with a guess x_0 for local minimum of *F(x)* and considers the decreasing sequence x_0, x_1, x_2 such that

$$x_{n+1} = x_n - \Upsilon_n \nabla F(x_n),\ n \geq 0.$$

We have a monotonic sequence

$$F(x_0) \geq F(x_1) \geq F(x_2) \geq \ldots,$$

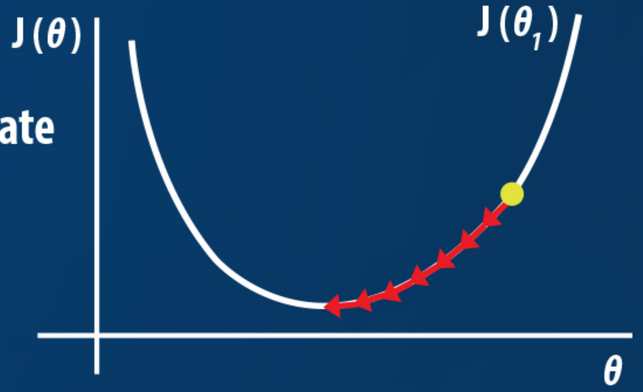
GRADIENT DESCENT AND LEARNING RATE

$$\boxed{\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)}$$

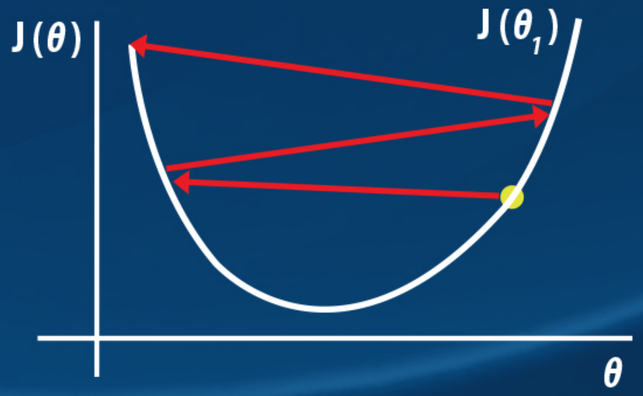
Where α is learning rate

If α is too small, gradient can be slow.

Too Low

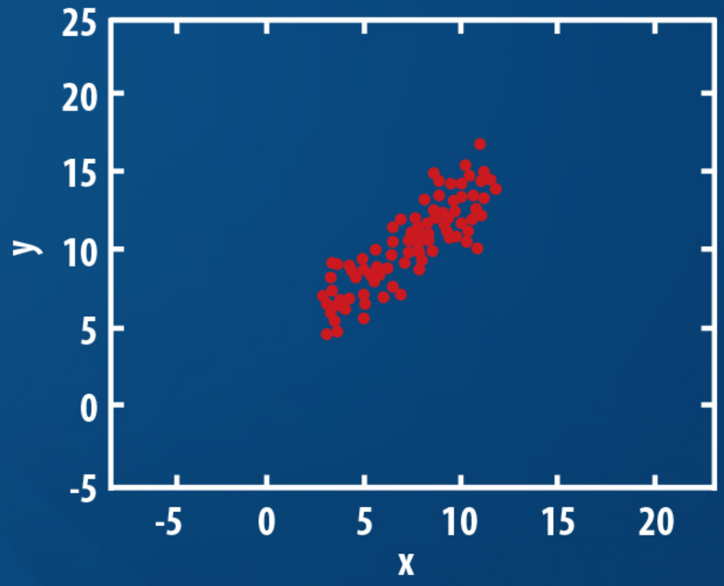


Too High



If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.

APPLICATION TO LINEAR REGRESSION



Let's suppose we want to model the above set of points with a line. To do this we'll use the standard $y = mx + b$ line equation

Where m is the line's slope and b is the line's y-intercept.

How do we compute the slope and intercept for best fit line ?

1. Define cost function
2. Set initial guess for slope and intercept
3. Compute cost based on initial value and store
4. Compute gradients at current point
5. Update slope and intercept
6. Iterate steps 3 to 5 till optimal error is reached

APPLICATION TO LINEAR REGRESSION: STEPS

Line equation

$$y = mx + b$$

1. Initialization

$$m = m_0 \quad b = b_0$$

Cost compute

2. $$f(m, b) = \frac{1}{N} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$$

3. Gradients

$$\frac{\partial}{\partial m} = \frac{2}{N} \sum_{i=1}^{N} -x_i (y_i - (mx_i + b))$$

$$\frac{\partial}{\partial b} = \frac{2}{N} \sum_{i=1}^{N} -(y_i - (mx_i + b))$$

4. Update rule

$$m_{j+1} = m_j - \alpha_m \frac{\partial f}{\partial m}$$

$$b_{j+1} = b_j - \alpha_b \frac{\partial f}{\partial b}$$

5. Iterate 2 - 4 till error optimized or converged