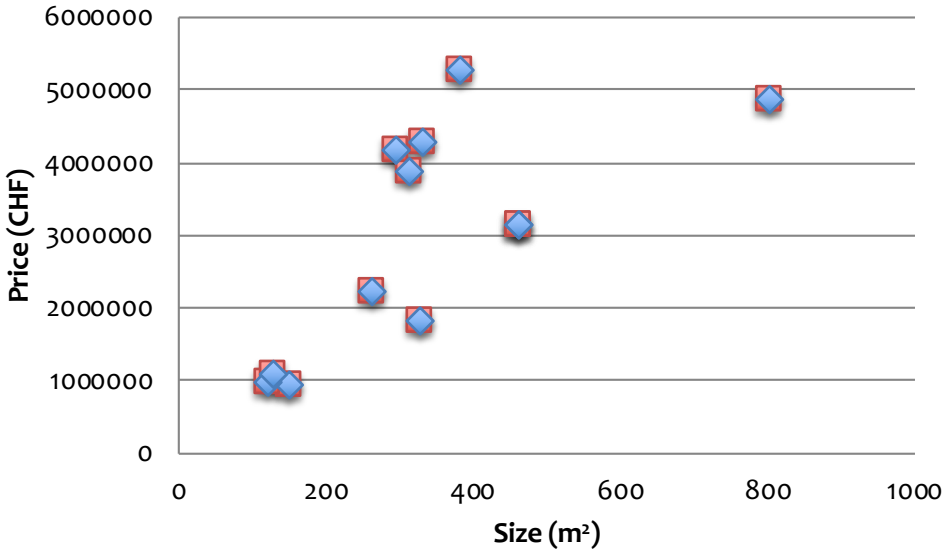
Gradient Descent

Stephan Robert

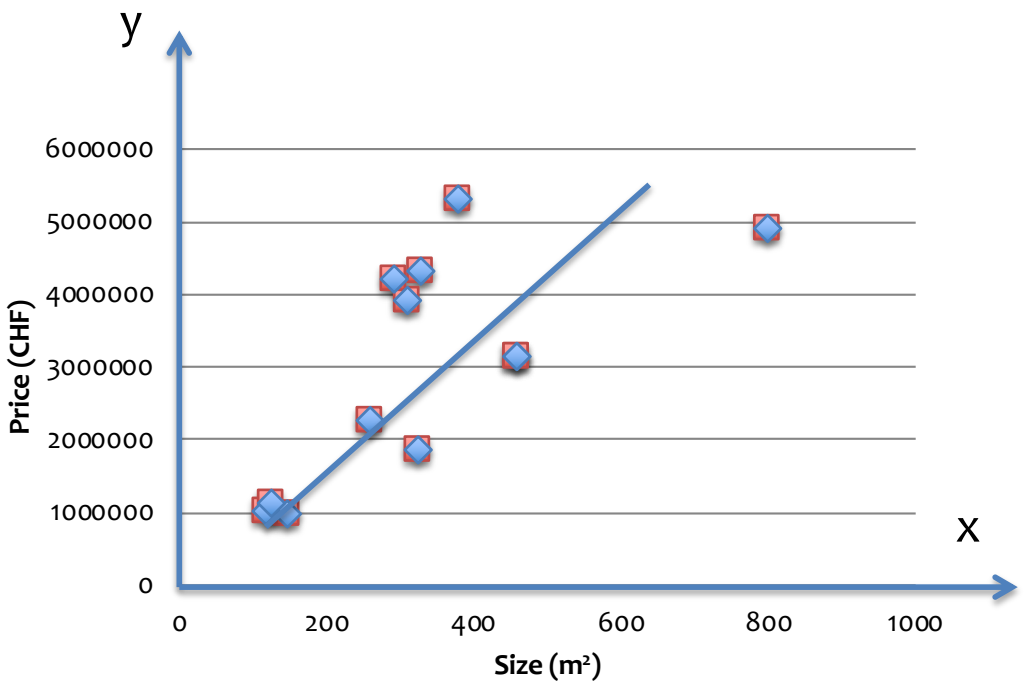
Model

- Other inputs
  - # of bathrooms
  - Lot size
  - Year built
  - Location
  - Lake view



Model

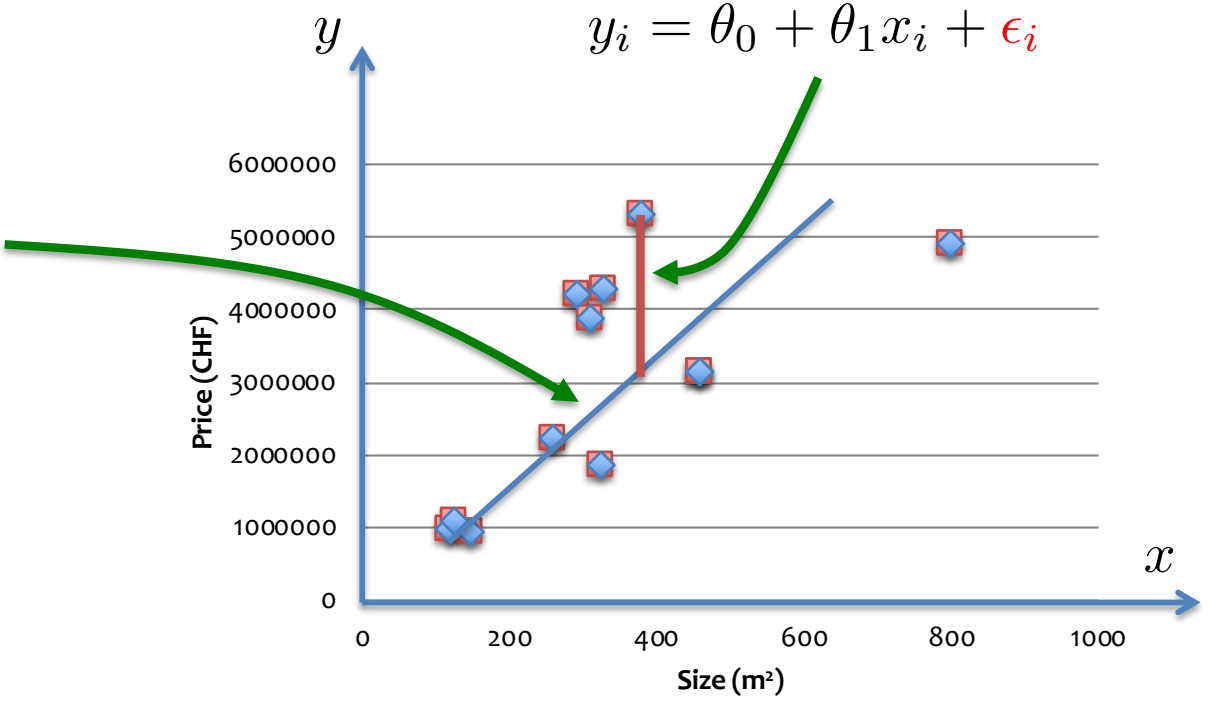
- How it works…
  - Linear regression with one variable



Model

- How it works…
  - Linear regression with one variable

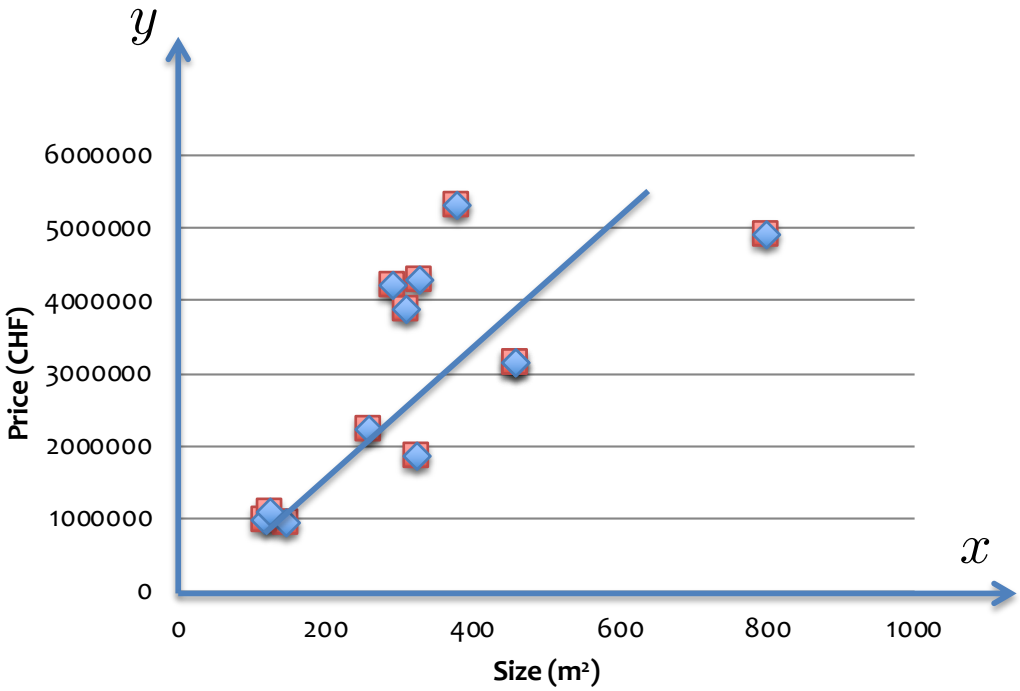
$$f_\theta(x) = f(x) = \theta_0 + \theta_1 x$$



Model

- Idea:
  - Choose θ_0, θ_1 so that $f(x)$ is close to y for our training example

$$f(x) = \theta_0 + \theta_1 x$$



Model

Cost function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

Aim:

$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1) = \min_{\theta_0, \theta_1} \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

Other functions

- Quadratic

$$f(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

- Higher order polynomial

$$f(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \dots$$

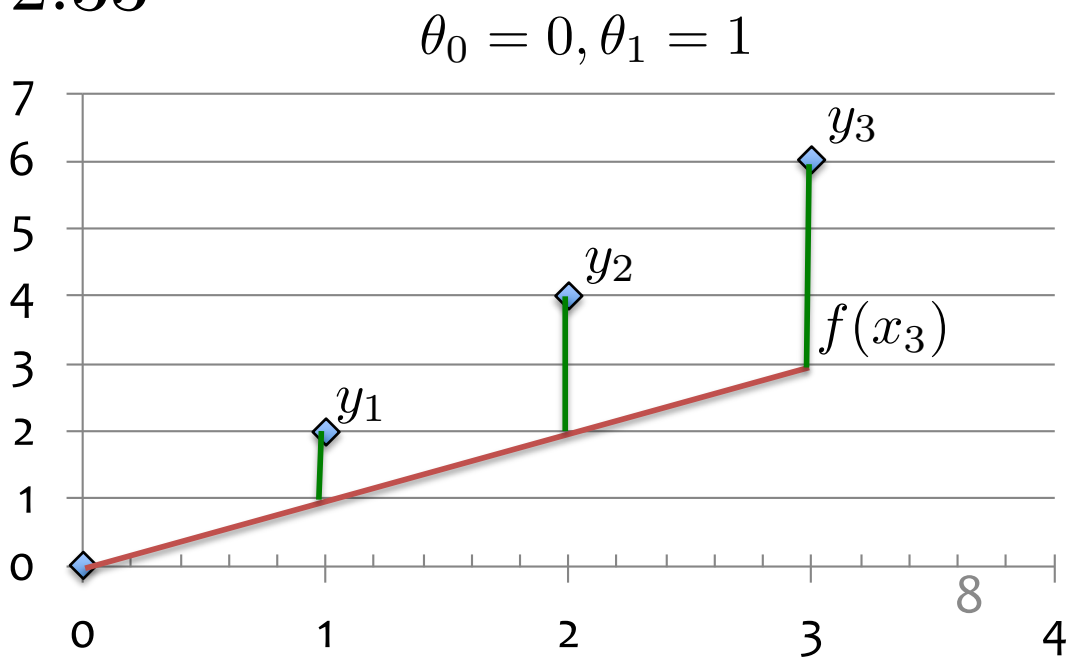
A very simple example

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2 =$$

$$\frac{1}{2m}((1-2)^2 + (2-4)^2 + (3-6)^2 =$$

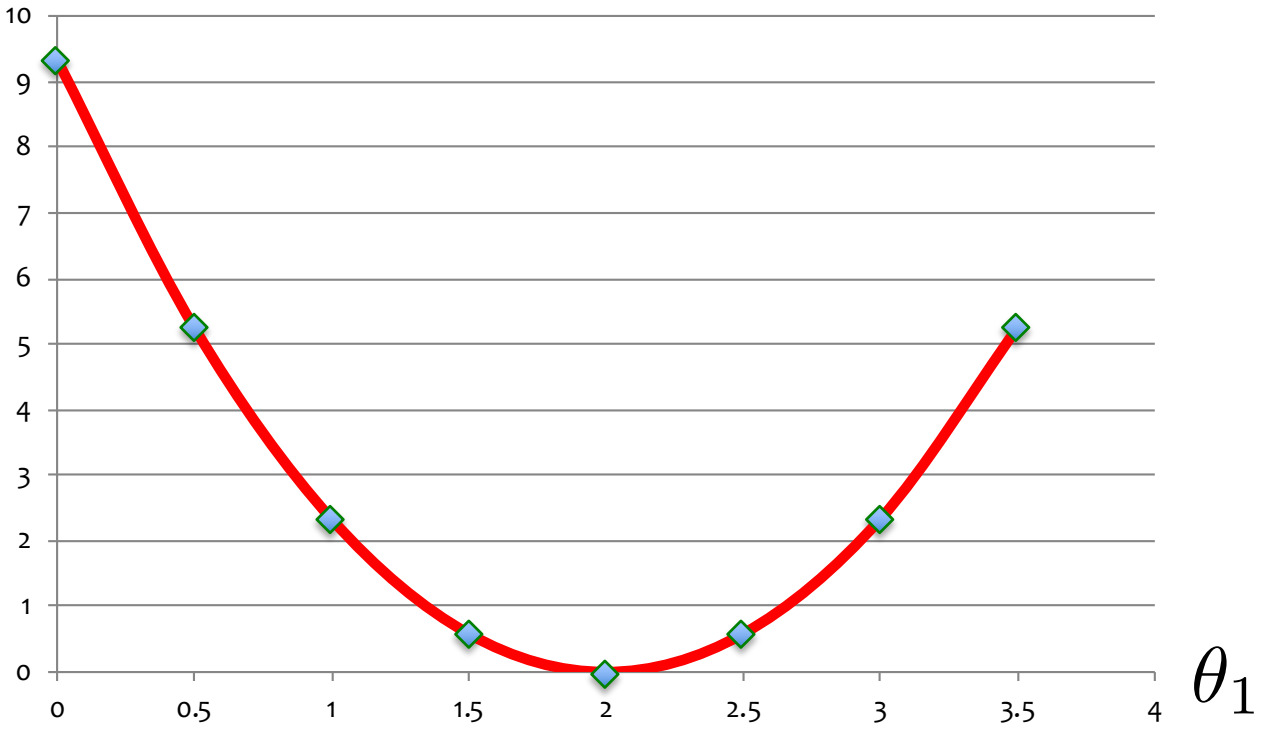
$$\frac{1}{6}(1+4+9) = \frac{14}{6} = 2.33$$

$$f(x_i) = \theta_0 + \theta_1 x_i$$



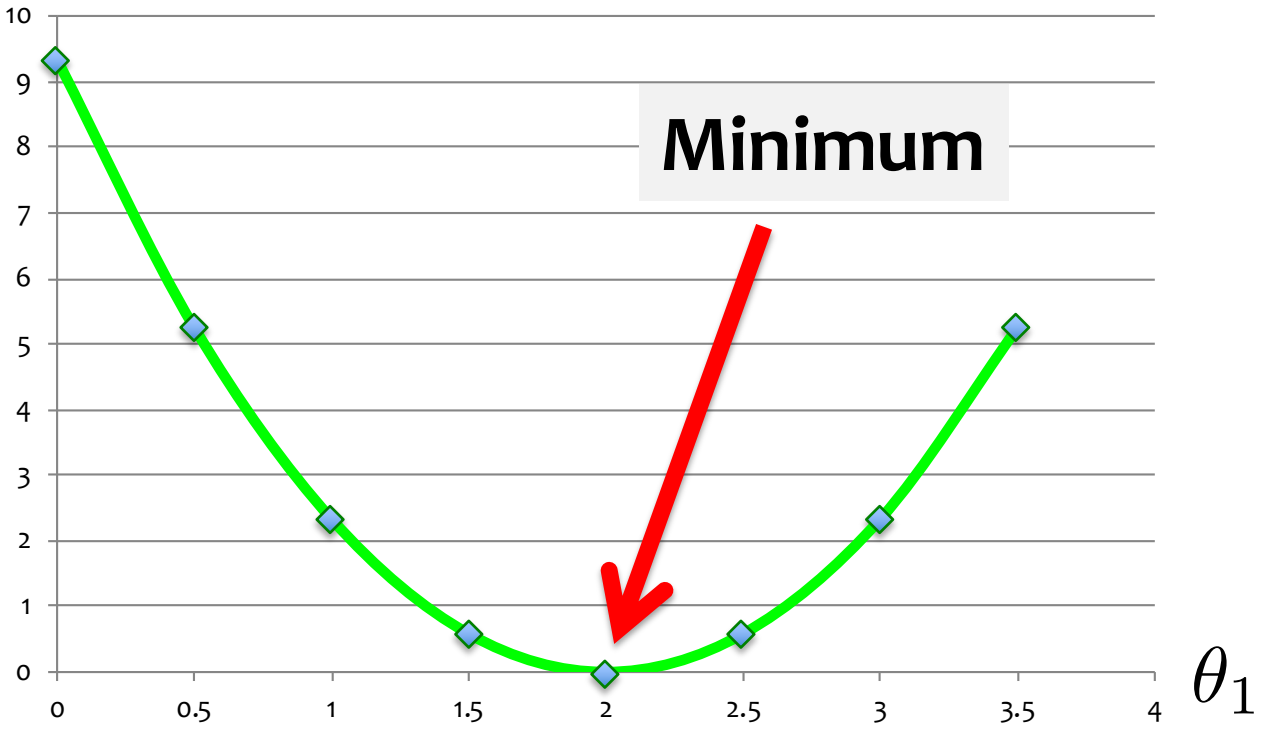
A very simple example

$$J(0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$



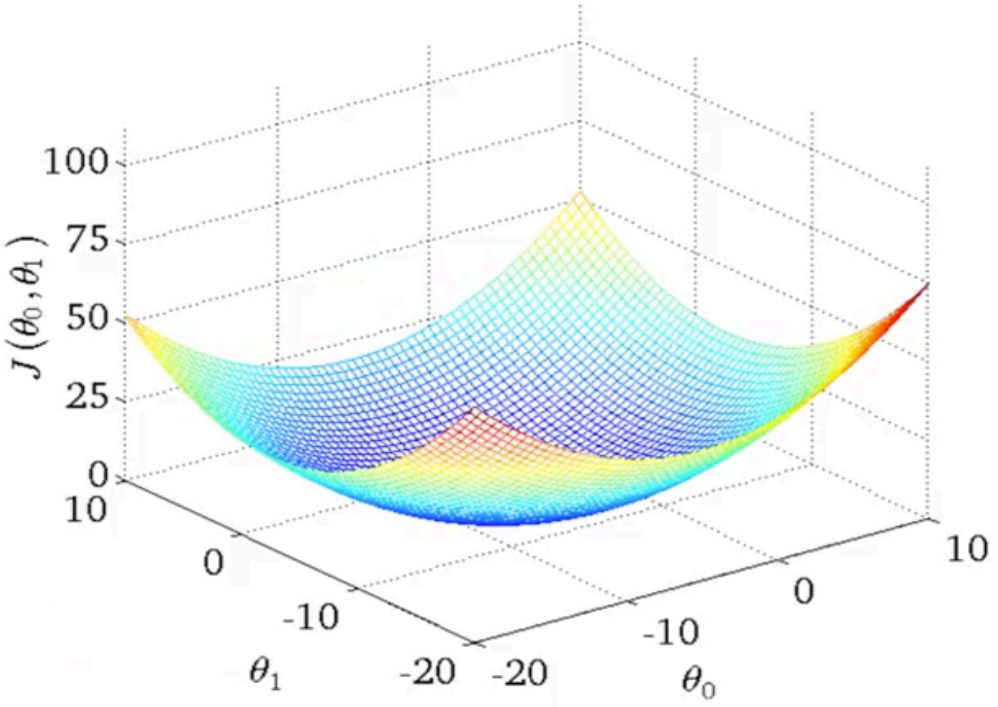
A very simple example

$$J(0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$



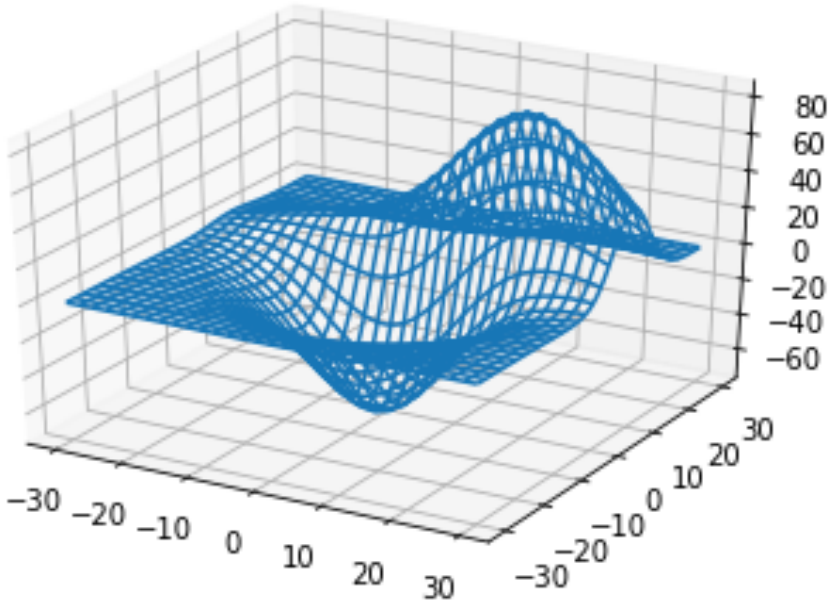
A very simple example

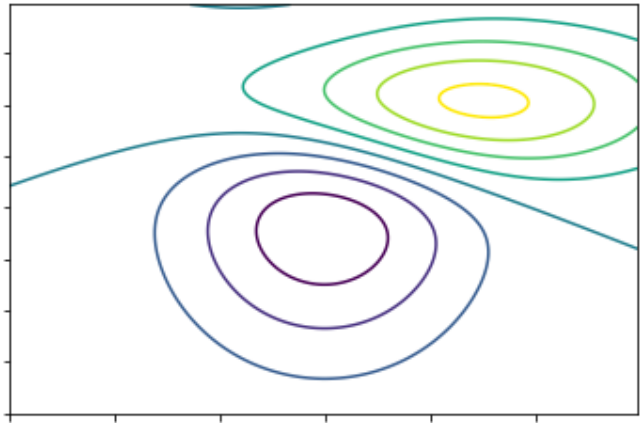
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$



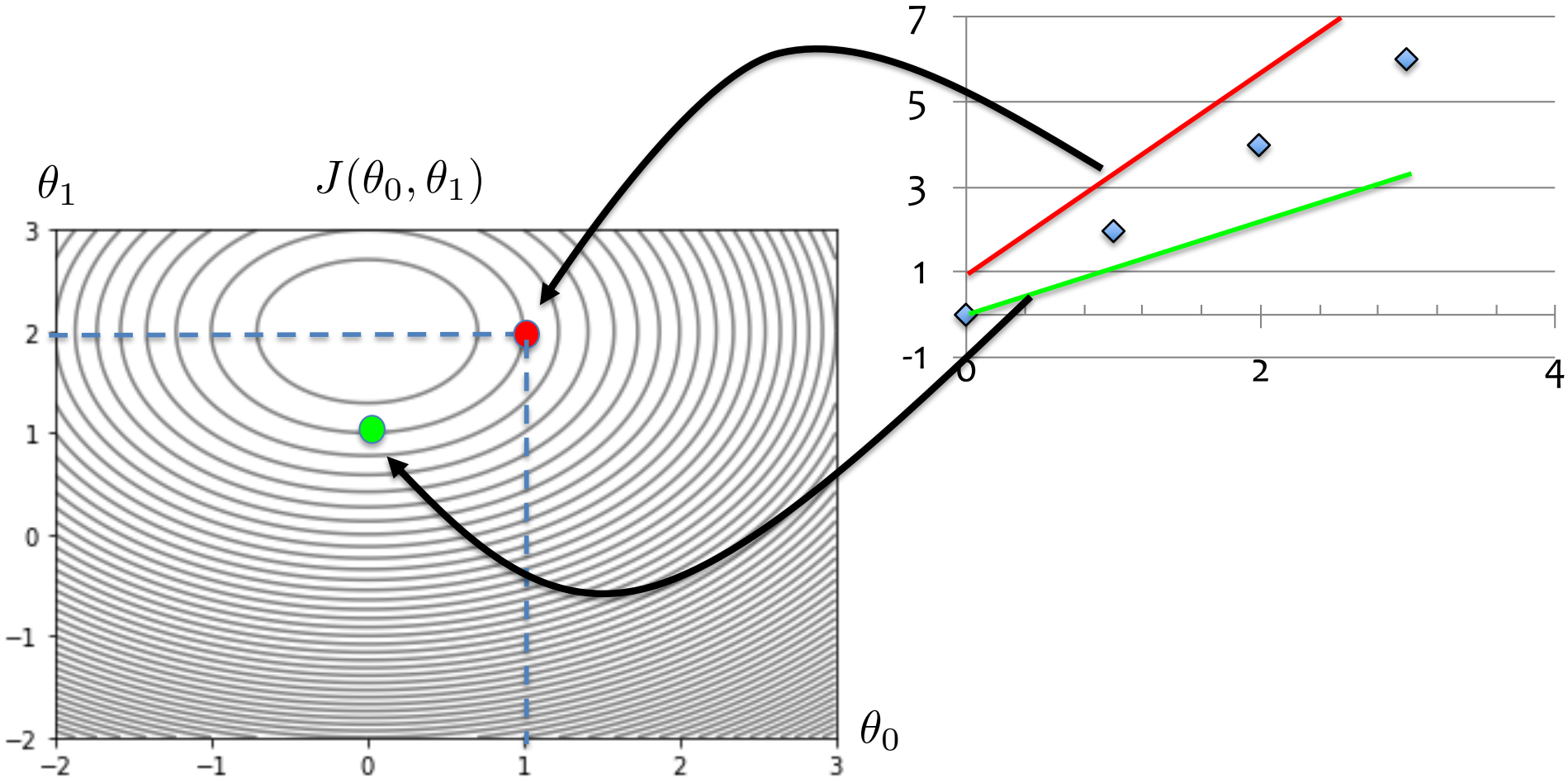
From Andrew Ng, Stanford

3D-plots





Cost function



Gradient descent (intuition)

Have some function $J(\theta_0, \theta_1)$

Goal: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Solution: we start with some (θ_0, θ_1) and we keep changing these parameters to reduce $J(\theta_0, \theta_1)$ until we hopefully end up to a **minimum**

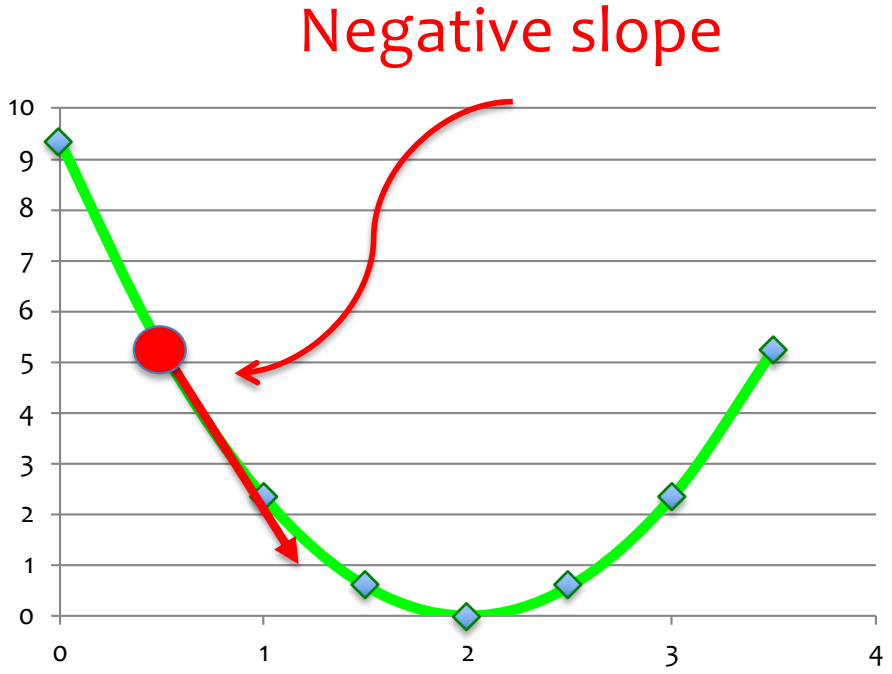
Gradient descent (intuition)

Have some function $J(\theta_0, \theta_1)$

Goal: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Proposition:

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$



Gradient descent

Formally

$J(\boldsymbol{\Theta}) : \mathbb{R}^n \rightarrow \mathbb{R}$ is **convex** and **differentiable**

$$\boldsymbol{\Theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

We want to solve $\min_{\boldsymbol{\Theta} \in \mathbb{R}^n} J(\boldsymbol{\Theta})$

i.e. find $\boldsymbol{\Theta}^*$ such that $J(\boldsymbol{\Theta}^*) = \min_{\boldsymbol{\Theta} \in \mathbb{R}^n} J(\boldsymbol{\Theta})$

Gradient descent

Algorithm

- Choose an initial $\boldsymbol{\Theta}[0] \in \mathbb{R}^{n+1}$
- Repeat $\boldsymbol{\Theta}[k] = \boldsymbol{\Theta}[k-1] - \alpha[k-1]\nabla J(\boldsymbol{\Theta}[k-1])$

  $k = 1, 2, 3, \ldots$
- Stop at some point

Example with 2 parameters

- Choose an initial $\boldsymbol{\Theta}[0] = (\theta_0[0], \theta_1[0])$
- Repeat

$$\theta_j[k] = \theta_j[k-1] - \alpha[k-1]\frac{\partial}{\partial\theta_j[k-1]}J(\theta_0[k-1], \theta_1[k-1])$$

$$j = 0, 1$$

- Stop at some point

Example with 2 parameters

Implementation detail

Repeat (simultaneous update)

$$\text{temp}_0 = \theta_0[k-1] - \alpha[k-1]\frac{\partial}{\partial\theta_0[k-1]}J(\theta_0[k-1],\theta_1[k-1])$$

$$\text{temp}_1 = \theta_1[k-1] - \alpha[k-1]\frac{\partial}{\partial\theta_1[k-1]}J(\theta_0[k-1],\theta_1[k-1])$$

$$\theta_0[k] = \text{temp}_0 \qquad k = 1, 2, 3, \ldots$$

$$\theta_1[k] = \text{temp}_1$$

Learning rate

- Search of **one parameter only** (one is supposed to be set already)

$$\theta_1[k] = \theta_1[k-1] - \alpha[k-1]\frac{\partial}{\partial\theta_1[k-1]}J(\theta_0[k-1], \theta_1[k-1])$$

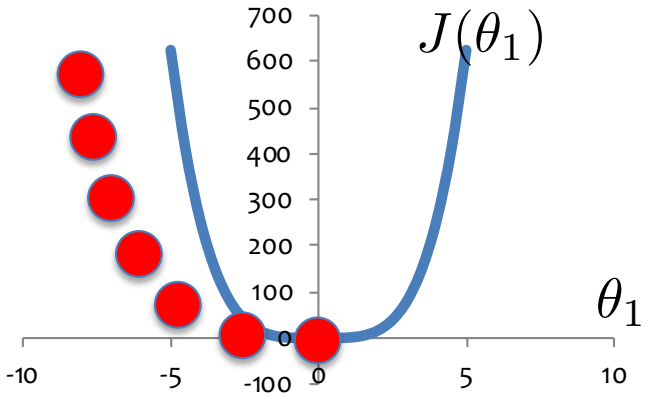
$$\alpha_{k-1} = \alpha_k = \alpha$$

$$\theta_1[k] = \theta_1[k-1] - \alpha\frac{\partial}{\partial\theta_1[k-1]}J(\theta_0[k-1], \theta_1[k-1])$$

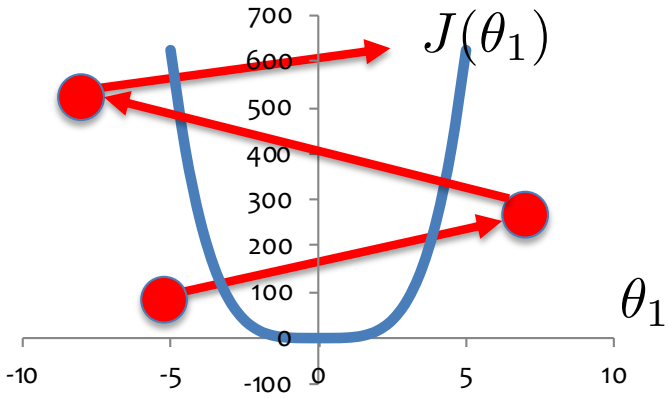
$$\theta_1[k] = \theta_1[k-1] - \alpha\frac{\partial}{\partial\theta_1[k-1]}J(\theta_0, \theta_1[k-1])$$

Learning rate

$$\theta_1[k] = \theta_1[k-1] - \alpha \frac{\partial}{\partial \theta_1[k-1]} J(\theta_0, \theta_1[k-1])$$



α too small



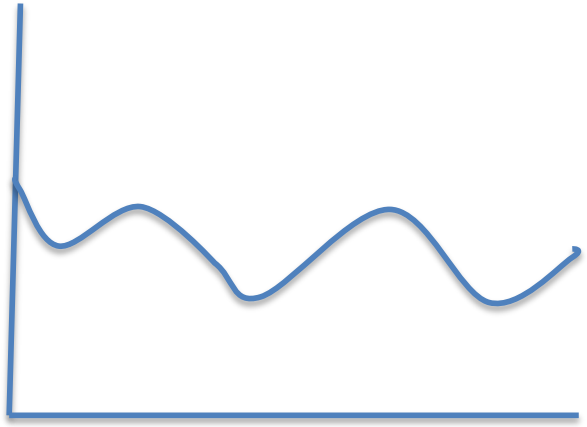
α too large

Gradient Descent

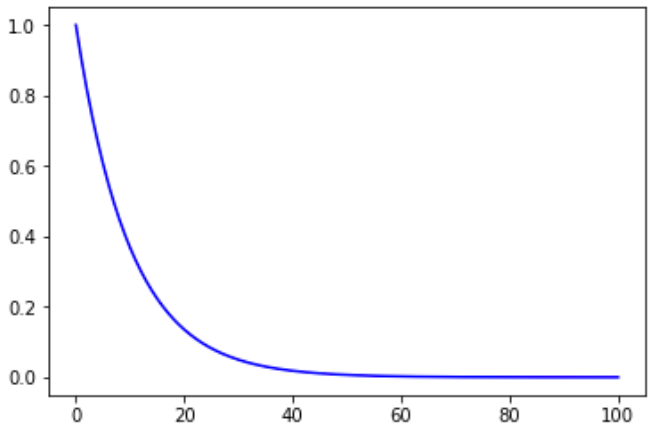
Learning rate (intuitive)

- Practical trick:
  - Debug: J should decrease after each iteration
  - Fix a stop-threshold (0.001?)
  - Observe the cost function

$$J(\Theta^*) = \min_{\Theta \in \mathbb{R}^n} J(\Theta)$$



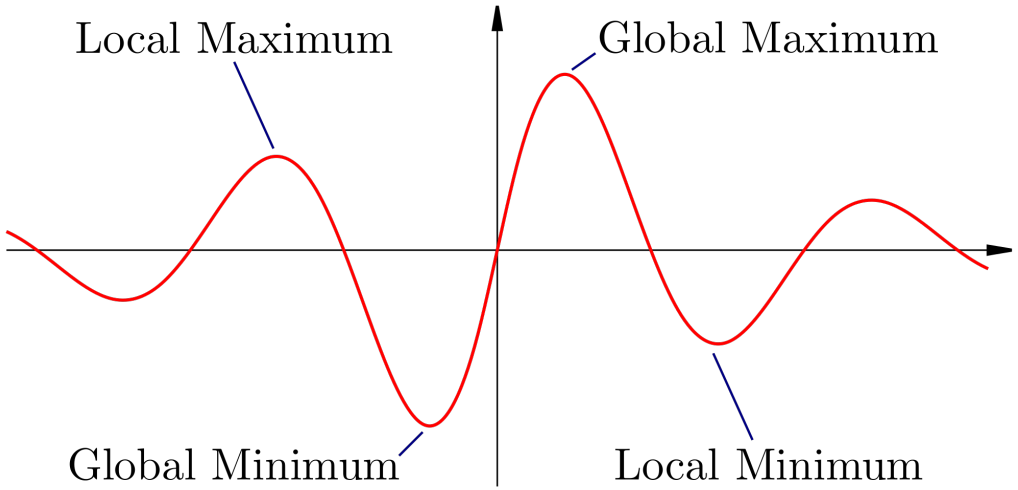
$J(\Theta)$



Number of iterations

Gradient descent

Important: We assume there are **no local minimums** for now!



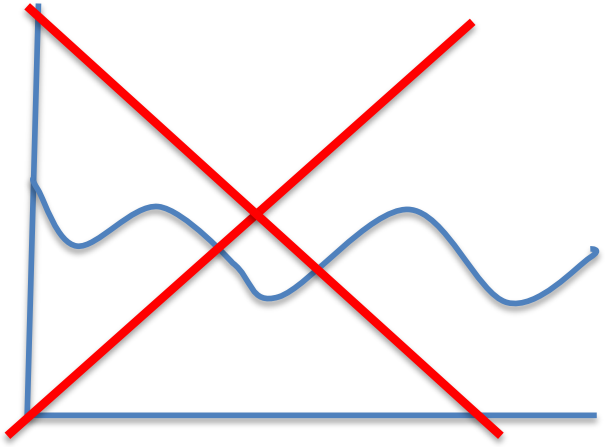
Wikimedia

Gradient Descent

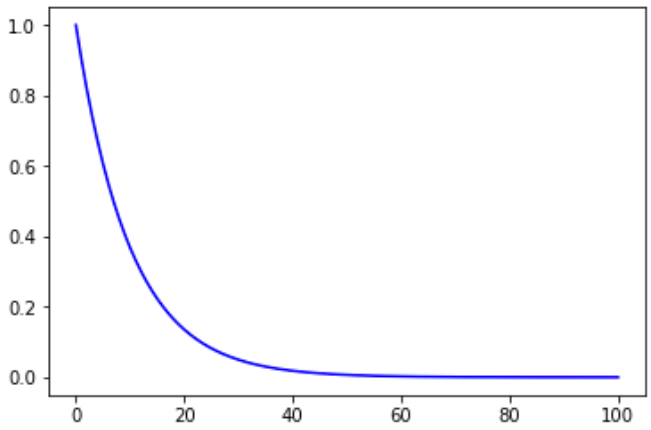
Learning rate (intuitive)

- Practical trick:
  - Debug: J should decrease after each iteration
  - Fix a stop-threshold (0.001?)
  - Observe the cost function

$$J(\Theta^*) = \min_{\Theta \in \mathbb{R}^n} J(\Theta)$$



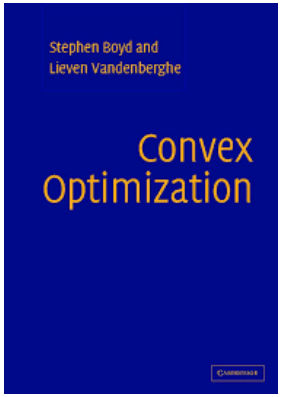
$J(\Theta)$



Number of iterations

Gradient Descent

Backtracking line search (Boyd)



- The porridge is

- Convergence analysis might help
  - Take an arbitrary parameter and fix it: $0.1 \leq \beta \leq 0.8$
  - At each iteration start with $\alpha = 1$ and while

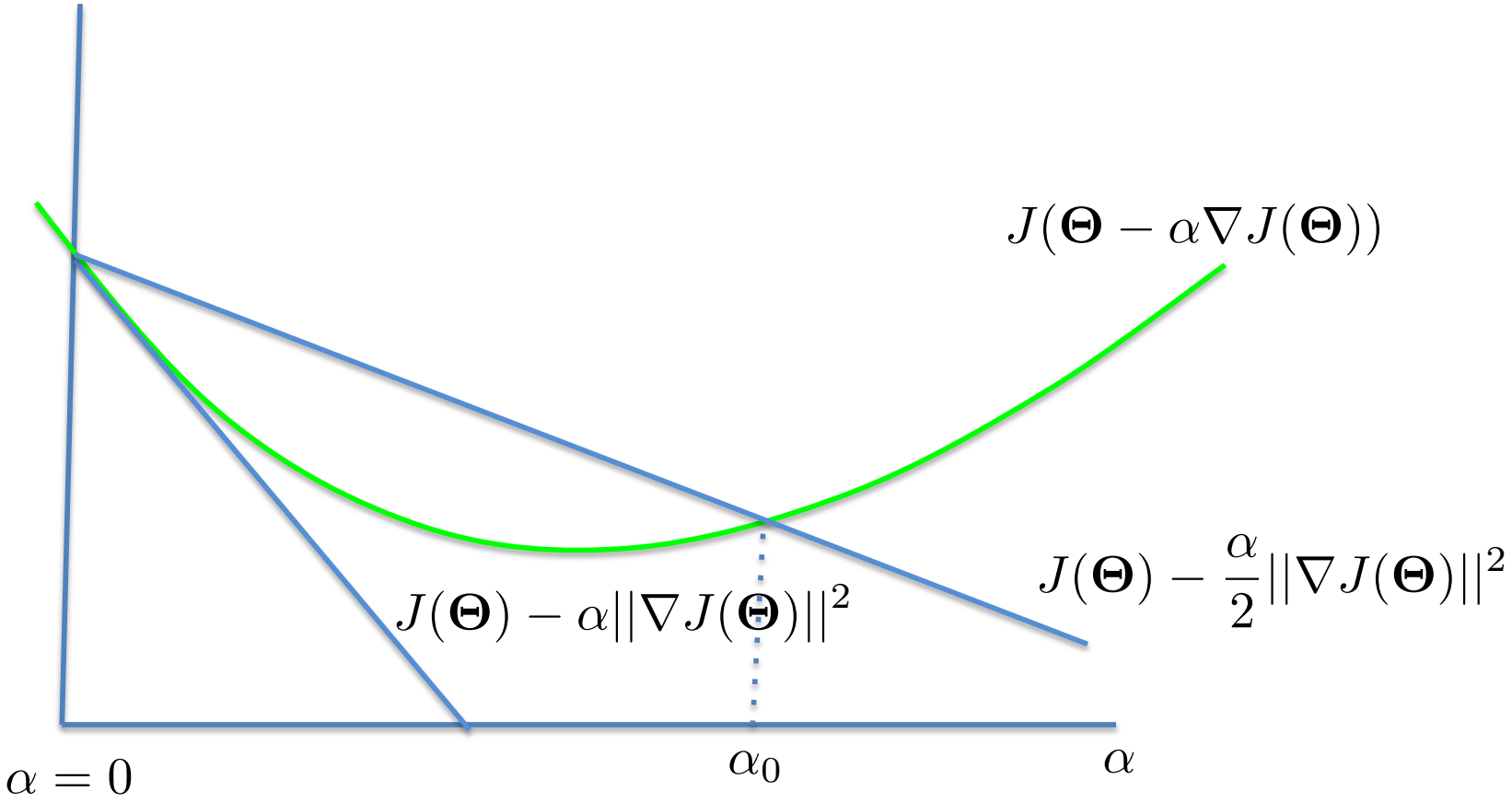
$$J(\mathbf{\Theta} - \alpha \nabla J(\mathbf{\Theta})) > J(\mathbf{\Theta}) - \frac{\alpha}{2} ||\nabla J(\mathbf{\Theta})||^2$$

update $\alpha = \beta\alpha$

Simple but works quite well in practice (check!)

Gradient Descent
Backtracking line search (Boyd)

- Interpretation

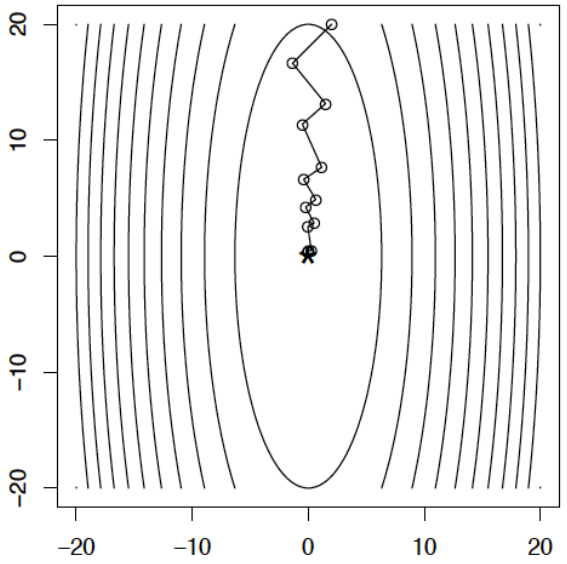


Gradient Descent

Backtracking line search (Boyd)

- Example (13 steps)
  - (Geoff Gordon & Ryan Tibshirani)
  - Convergence analysis by GG&RT

$$J = \frac{10(\theta_0^2) + \theta_1^2}{2}$$



$$\beta = 0.8$$

Gradient Descent
Exact line search (Boyd)

- At each iteration do the best along the gradient direction

$$\alpha = \arg\max_{s} J(\mathbf{\Theta} - s\nabla J(\mathbf{\Theta}))$$

  - Often not much more efficient than backtracking, not really worth it…

Gradient descent

linear regression

Repeat

$$\theta_j[k] = \theta_j[k-1] - \alpha[k-1]\frac{\partial}{\partial\theta_j[k-1]}J(\theta_0[k-1], \theta_1[k-1])$$

Stop at some point $j = 0, 1$

$$f(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$

$$= \frac{1}{2m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x_i - y_i)^2$$

Gradient descent

linear regression

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i)^2$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_0} \left(\frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i)^2 \right)$$

$$= \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i)$$

Gradient descent

linear regression

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i)^2$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_1} \left(\frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i)^2 \right)$$

$$= \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i) x_i$$

Gradient descent
linear regression

- Choose an initial $\boldsymbol{\Theta}[0] = (\theta_0[0], \theta_1[0])$

$$\alpha_{k-1} = \alpha_k = \alpha$$

- Repeat

$$\theta_0[k] = \theta_0[k-1] - \alpha \frac{\partial}{\partial \theta_0[k-1]} J(\theta_0[k-1], \theta_1[k-1])$$

$$\theta_1[k] = \theta_1[k-1] - \alpha \frac{\partial}{\partial \theta_1[k-1]} J(\theta_0[k-1], \theta_1[k-1])$$

- Stop at some point

Gradient descent
linear regression

- Repeat

$$\theta_0[k] = \theta_0[k-1] - \frac{\alpha}{m}\sum_{i=1}^{m}(\theta_0[k-1] + \theta_1[k-1]x_i - y_i)$$

$$\theta_1[k] = \theta_1[k-1] - \frac{\alpha}{m}\sum_{i=1}^{m}(\theta_0[k-1] + \theta_1[k-1]x_i - y_i)x_i$$

- Stop at some point

WARNING: Update θ_0 and θ_1 simultaneously

Generalization

- Higher order polynomial

$$f(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \dots$$

- Multi-features

$$f(x_1, x_2, \dots) = f(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots$$

WARNING: Notation

$f(x) : (x)_1 = x_1$ is a **scalar**

$f(x_1, x_2, \dots) : x_1$ is a **variable**

$(x_1)_i$ is a **scalar**

Gradient Descent, n=1

Linear Regression

- Choose an initial $\Theta[0] = (\theta_0[0], \theta_1[0])$

$$\alpha_{k-1} = \alpha_k = \alpha$$

- Repeat

$$\theta_0[k] = \theta_0[k-1] - \frac{\alpha}{m} \sum_{i=1}^{m} (\theta_0[k-1] + \theta_1[k-1]x_i - y_i)$$

$$\theta_1[k] = \theta_1[k-1] - \frac{\alpha}{m} \sum_{i=1}^{m} (\theta_0[k-1] + \theta_1[k-1]x_i - y_i)x_i$$

- Stop at some point

Gradient Descent, n>1

Linear Regression

- Choose an initial $\boldsymbol{\Theta}[0] = (\theta_0[0], \theta_1[0], \theta_2[0], \ldots, \theta_n[0])$
- Repeat

$$\theta_0[k] = \theta_0[k-1] - \frac{\alpha}{m}\sum_{i=1}^{m}(f(\mathbf{x}_i) - y_i)$$

$$\theta_1[k] = \theta_1[k-1] - \frac{\alpha}{m}\sum_{i=1}^{m}(f(\mathbf{x}_i) - y_i)\,(x_1)_i$$

$$\theta_2[k] = \theta_2[k-1] - \frac{\alpha}{m}\sum_{i=1}^{m}(f(\mathbf{x}_i) - y_i)\,(x_2)_i$$

$$\vdots$$

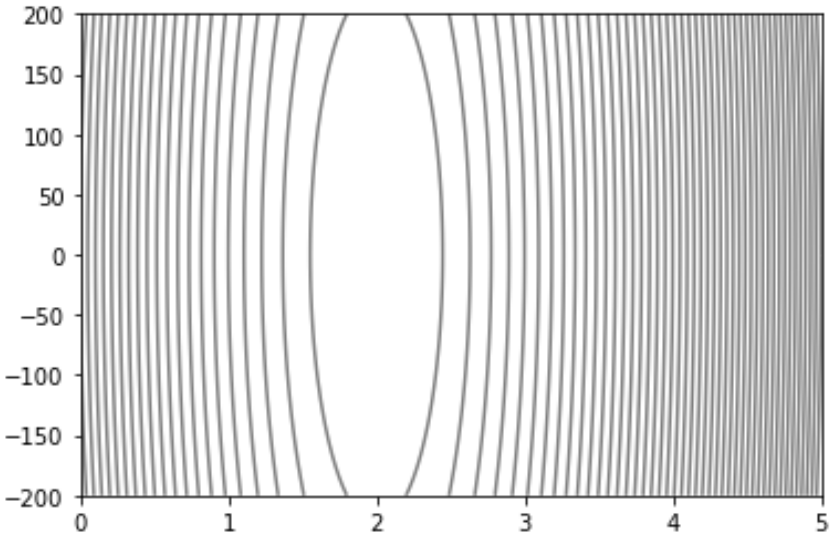
- Stop at some point

Gradient Descent
Scaling

- Practical trick:
  - all features on the similar scale

    e.g. size of the house = 300 m², number of floors: 3
  - Mean normalization

$$x_i = \frac{\text{Size}_i - \text{MeanSize}}{\text{MaxSize-MinSize}}$$



Vector Notation

$$\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^{n+1} \qquad \mathbf{\Theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

$$f(x_0, x_1, x_2, \ldots, x_n) = f(\mathbf{x}) = \mathbf{\Theta}^t \mathbf{x}$$

Often: $(x_0)_i = 1, i \in \{0, 1, \ldots, n\}$

$$= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

Vector Notation

$$(\mathbf{x})_i = \begin{pmatrix} (x_0)_i \\ (x_1)_i \\ \vdots \\ (x_n)_i \end{pmatrix} \qquad \mathbf{\Theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

$$f((x_0)_i, (x_1)_i, (x_2)_i, \ldots, (x_n)_i) = f((\mathbf{x})_i) = \mathbf{\Theta}^t \mathbf{x}_i$$
$$= \theta_0 + \theta_1 (x_1)_i + \theta_2 (x_2)_i + \ldots + \theta_n (x_n)_i$$

Vector Notation

$$\mathbf{X} = \begin{pmatrix} (\mathbf{x})_1^t \\ (\mathbf{x})_2^t \\ \vdots \\ (\mathbf{x})_m^t \end{pmatrix} = \begin{pmatrix} (x_0)_1 & (x_1)_1 & \cdots & (x_n)_1 \\ (x_0)_2 & (x_1)_2 & \cdots & (x_n)_2 \\ \vdots & & & \\ (x_0)_m & (x_1)_m & \cdots & (x_n)_m \end{pmatrix} \in \mathbb{R}^{m \times (n+1)}$$

m

(n+1)

Vector Notation

$$\mathbf{y} = \begin{pmatrix} (y)_1 \\ (y)_2 \\ \vdots \\ (y)_m \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{R}^m \qquad \mathbf{\Theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

Cost function

$$J(\mathbf{\Theta}) = \frac{1}{2m} (\mathbf{X\Theta} - \mathbf{y})^t (\mathbf{X\Theta} - \mathbf{y})$$

Example

m=4=#nb of samples

x_0	x_1	x_2	x_3	x_4	y
	Size	#bedrooms	#floors	Age	Price (mios)
1	125	3	2	20	0.8
1	220	5	2	15	1.2
1	400	7	3	5	4
1	250	4	2	10	2

$$\mathbf{X} = \begin{pmatrix} 1 & 125 & 3 & 2 & 20 \\ 1 & 220 & 5 & 2 & 15 \\ 1 & 400 & 7 & 3 & 5 \\ 1 & 250 & 4 & 2 & 10 \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} 0.8 \\ 1.2 \\ 4 \\ 2 \end{pmatrix} \in \mathbb{R}^m$$

Example

$$\mathbf{X} = \begin{pmatrix} 1 & 125 & 3 & 2 & 20 \\ 1 & 220 & 5 & 2 & 15 \\ 1 & 400 & 7 & 3 & 5 \\ 1 & 250 & 4 & 2 & 10 \end{pmatrix} \in \mathbb{R}^{m \times (n+1)}$$

$$\mathbf{X}^t = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 125 & 220 & 400 & 250 \\ 3 & 5 & 7 & 4 \\ 2 & 2 & 3 & 2 \\ 20 & 15 & 5 & 10 \end{pmatrix} \in \mathbb{R}^{(n+1) \times m}$$

Normal equation

- Method to resolve Θ analytically (Least squares)
  - Derivates = 0

$$\boldsymbol{\Theta}^* = (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}$$

Comparison

- **Gradient Descent**
  - Choose a learning parameter
  - Many iterations
  - Works even for large n (in the order of 10^6)
- **Normal Equation**
  - No need to choose a learning parameter
  - No iteration
  - Need to compute $(\mathbf{X}^t\mathbf{X})^{-1}$
  - Slow when n is large (-> 10'000)