COMPUTATIONAL INTELLIGENCE

(INTRODUCTION TO MACHINE LEARNING) SS17

Lecture 2:

- Linear Regression
- Gradient Descent
- Non-linear basis functions

LINEAR REGRESSION MOTIVATION

Why Linear Regression?

- Simplest machine learning algorithm for regression
  - Widely used **in biological, behavioral and social sciences** to describe and to extract relationships between variables from data
  - Prediction **of real-valued outputs**
  - Easy to implement, fast to execute
  - **Benchmark** algorithm for comparison with more complex algorithms

- Introduction to **notation and concepts** that we will need again later in the course
  - Data format, vector & matrix notation
  - Learning from data by minimizing a **cost function**
  - Gradient descent
  - Non-linear features and basis functions
  - Preparation for neural networks

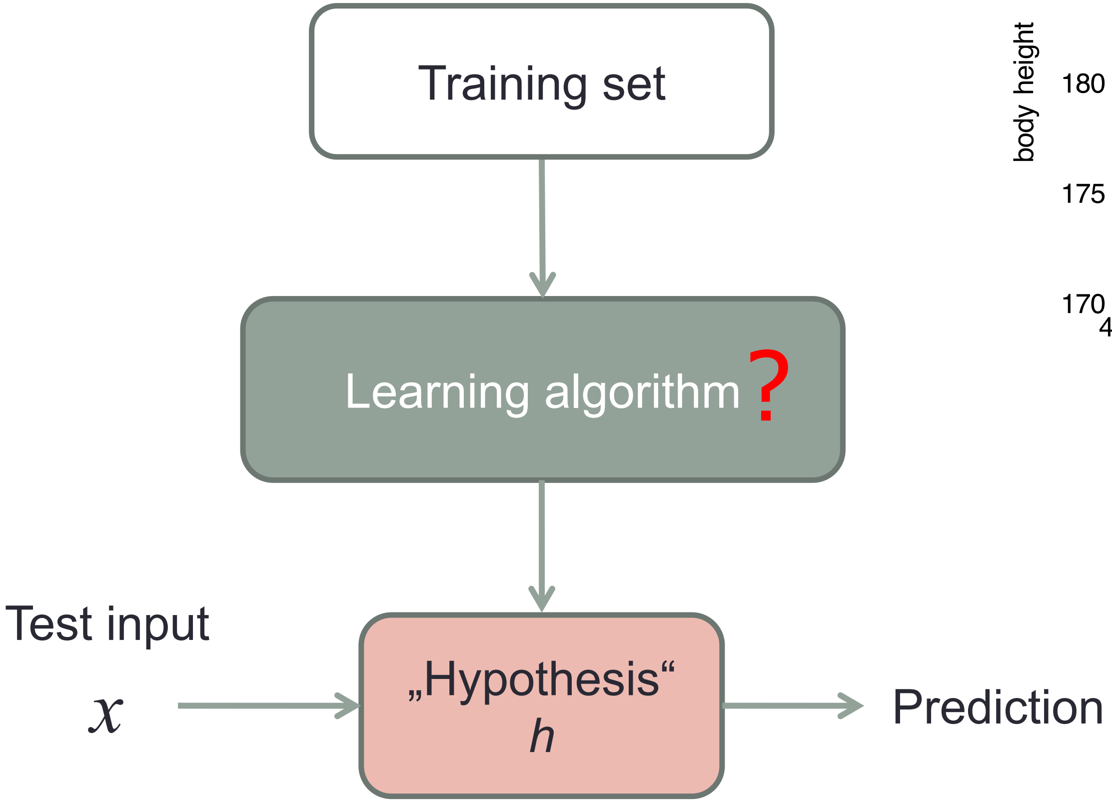
Applications of (linear) regression

- Brain computer interfaces
  - https://www.youtube.com/watch?v=Ae6En8-eaww
- Neuroprosthetic control
  - https://www.youtube.com/watch?v=X_AI4MiY6L4

LINEAR REGRESSION WITH ONE INPUT

Linear regression with one input

$$\langle x^{(1)}, y^{(1)}\rangle \;\ldots\; \langle x^{(m)}, y^{(m)}\rangle$$



Training set

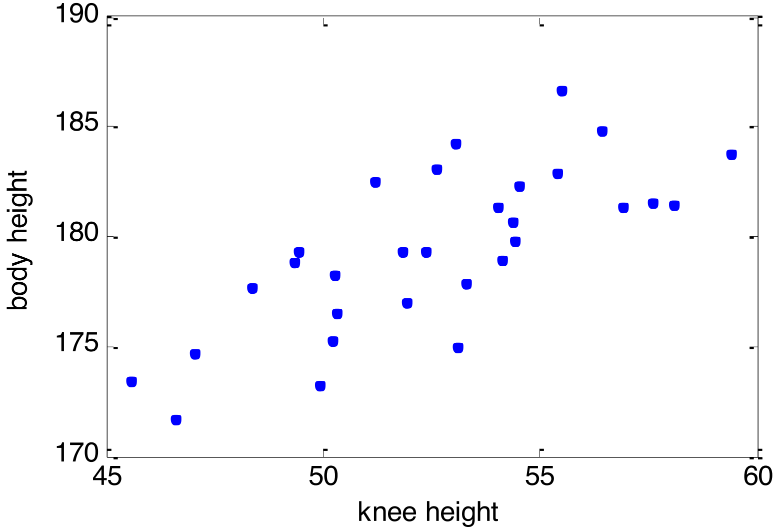
Learning algorithm ?

Test input

x

„Hypothesis“
h

Prediction



Hypothesis

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot x$$

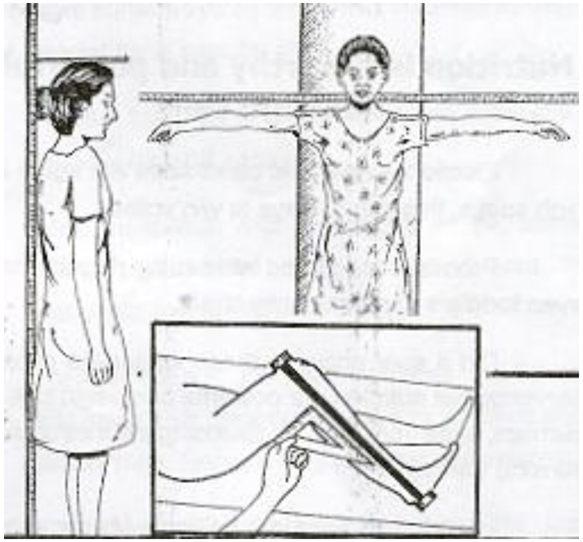
Parameters ?

$$\boldsymbol{\theta} = (\theta_0, \theta_1)$$

A regression problem

- We want to learn to predict a **person's height** based on his/her **knee height** and/or **arm span**

- This is useful for patients who are **bed bound** or in a wheelchair and cannot stand to take an accurate measurement of their height

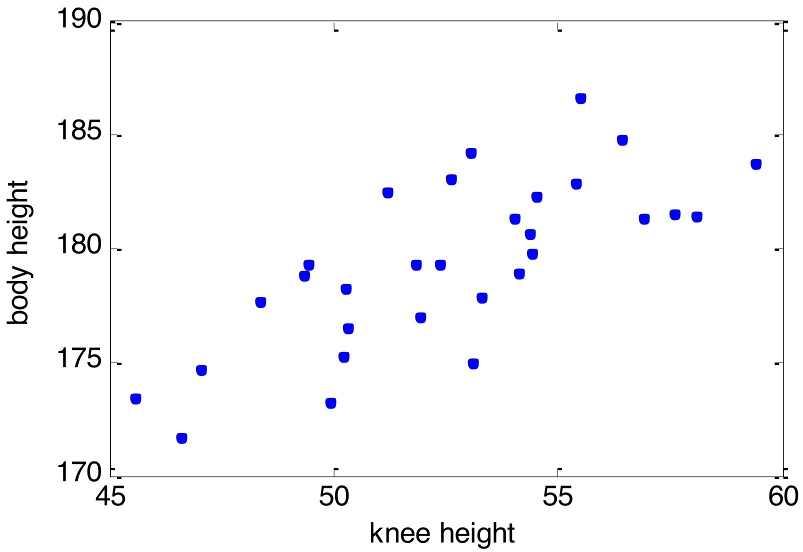


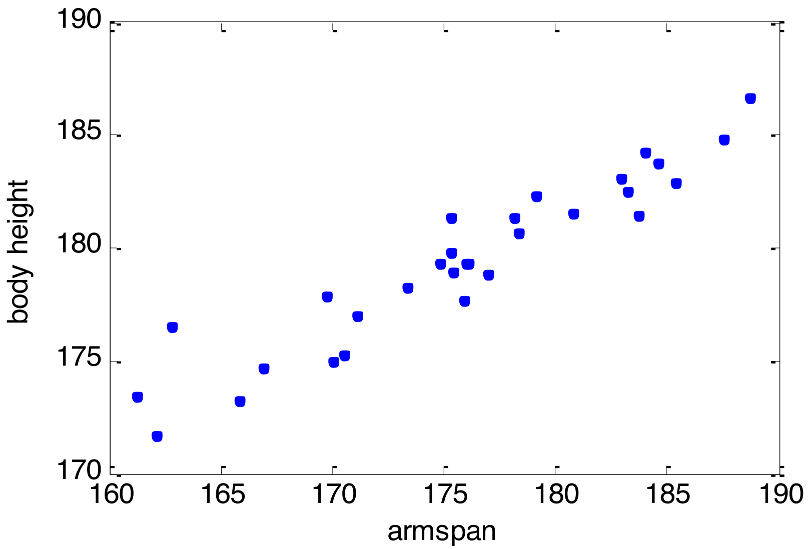
Knee Height [cm]	Arm span [cm]	Height [cm]
50	166	171
56	172	175
52	174	168
…	…	…

Example Data

Knee height [cm]	Arm span [cm]	Height [cm]
50	166	171
56	172	175
52	174	168
…	…	…

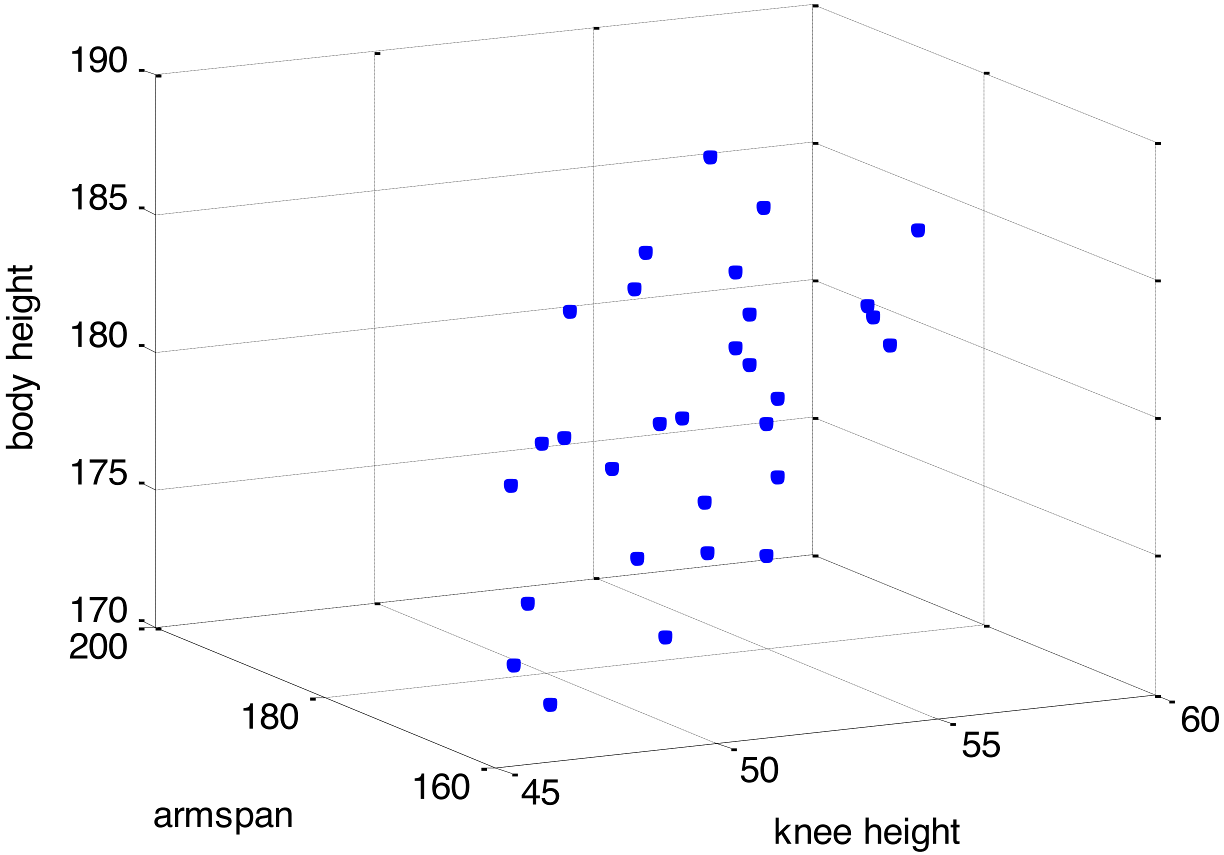
m=30 data points





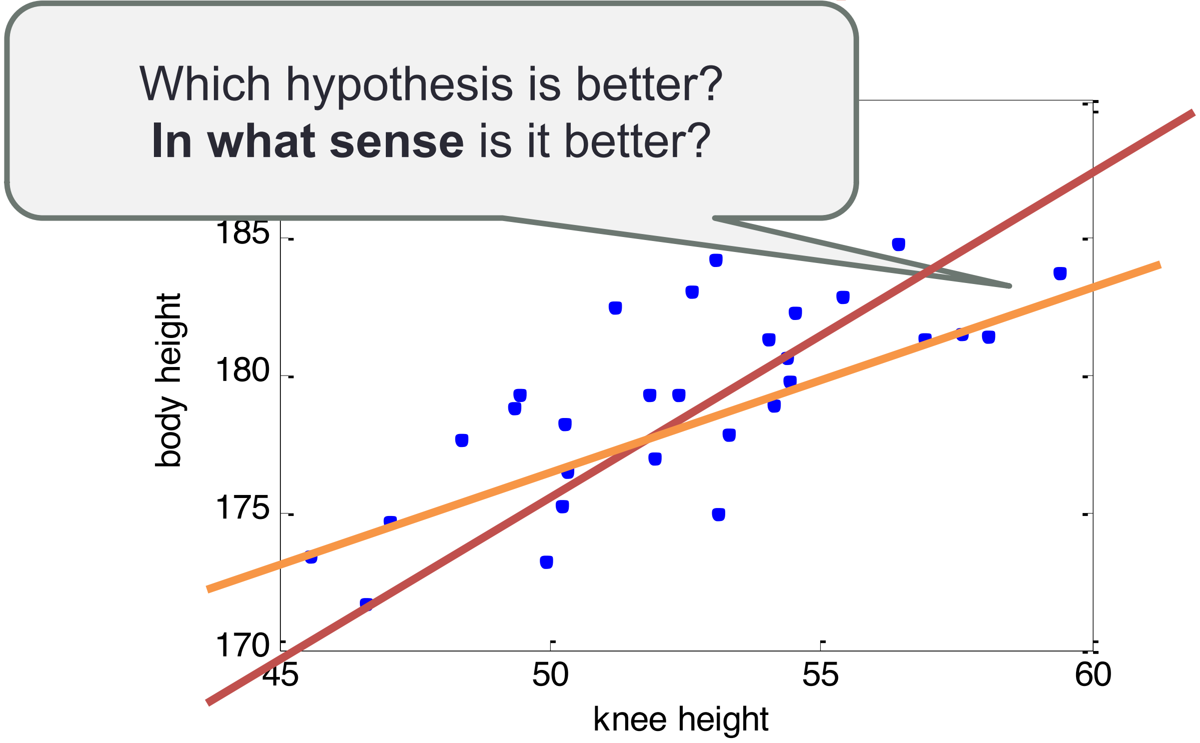
Example Data

Knee Height [cm]	Arm span [cm]	Height [cm]
50	166	171
56	172	175
52	174	168
…	…	…



Linear regression with one input

Knee Height [cm]	Height [cm]
50	171
56	175
52	168
…	…



Which hypothesis is better?
In what sense is it better?

Hypothesis

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot x$$

Parameters ?

$$\boldsymbol{\theta} = (\theta_0, \theta_1)$$

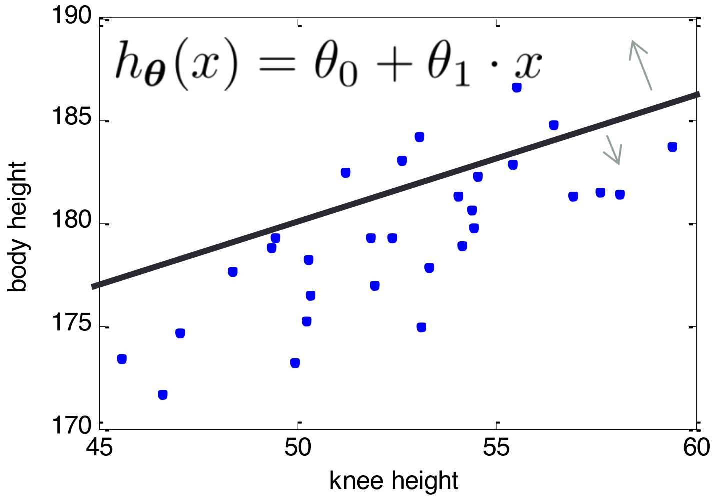
Formalization of problem

$x^{(i)}$

Knee Height [cm]	Height [cm]
50	171
56	175
52	168
…	…

$y^{(i)}$

m=30 data points



- Given *m* training examples

$$\langle x^{(1)}, y^{(1)} \rangle \; \dots \; \langle x^{(m)}, y^{(m)} \rangle$$

- Goal: learn parameters

$$\boldsymbol{\theta} = (\theta_0, \theta_1)$$

such that

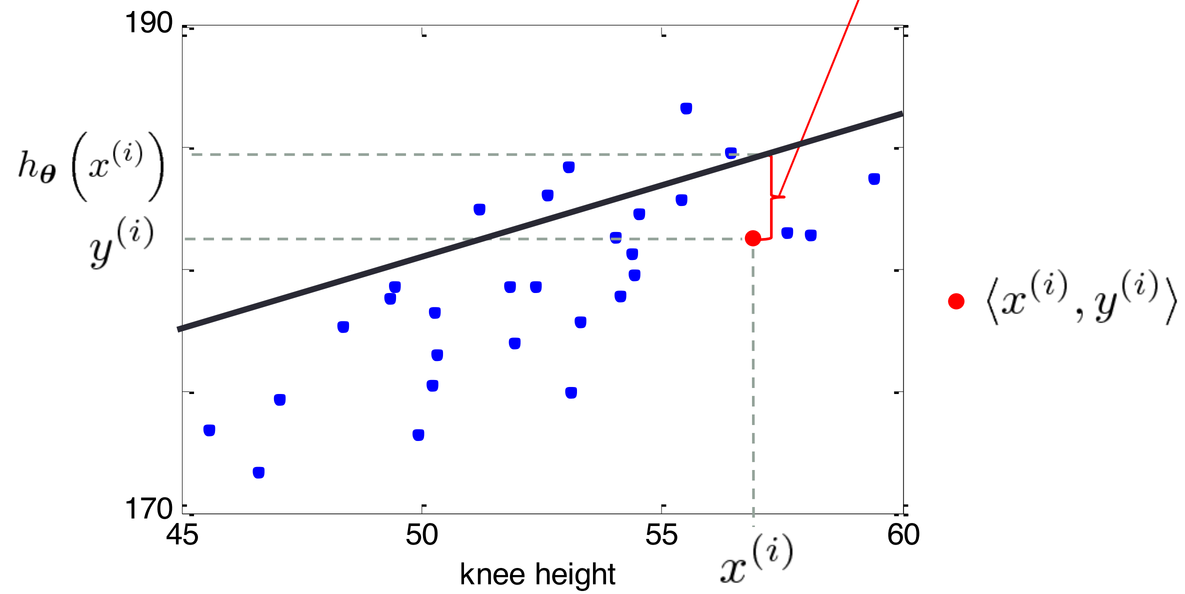
$$h_{\boldsymbol{\theta}}(x^{(i)}) = \theta_0 + \theta_1 \cdot x^{(i)} \approx y^{(i)}$$

for all training examples *i=1…30.*

Least Squares Objective

- Minimize Error

$$J(\theta_0, \theta_1) = \underbrace{\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}\right)}_{}^2$$



$\theta_0 = \ 150$

$\theta_1 = \ 0.6$

Least Squares Objective

- Minimize Error $$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(\underline{h_{\boldsymbol{\theta}} \left(x^{(i)} \right) - y^{(i)}} \right)^2$$

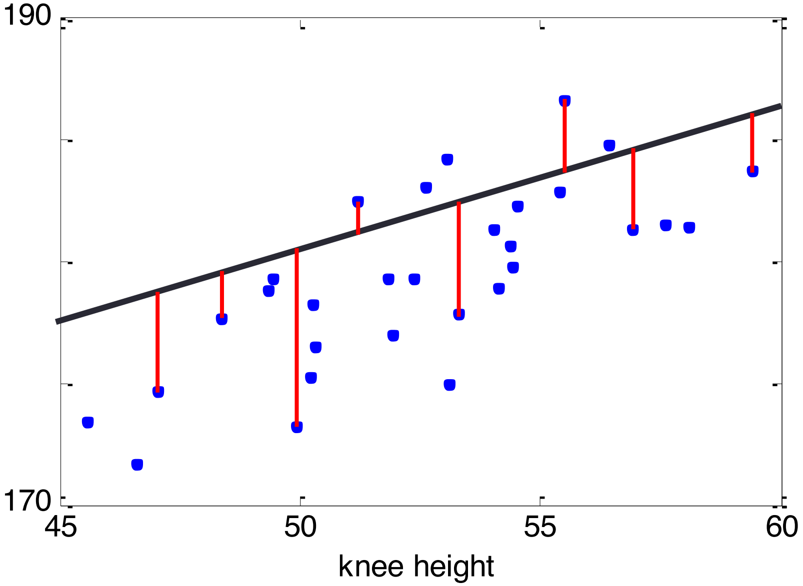
cost function

mean squared error

$\theta_0 = 150$

$\theta_1 = 0.6$

$$J(\theta_0, \theta_1) = 10.77$$



Least Squares Objective

- Minimize Error $J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(\underline{h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}} \right)^2$

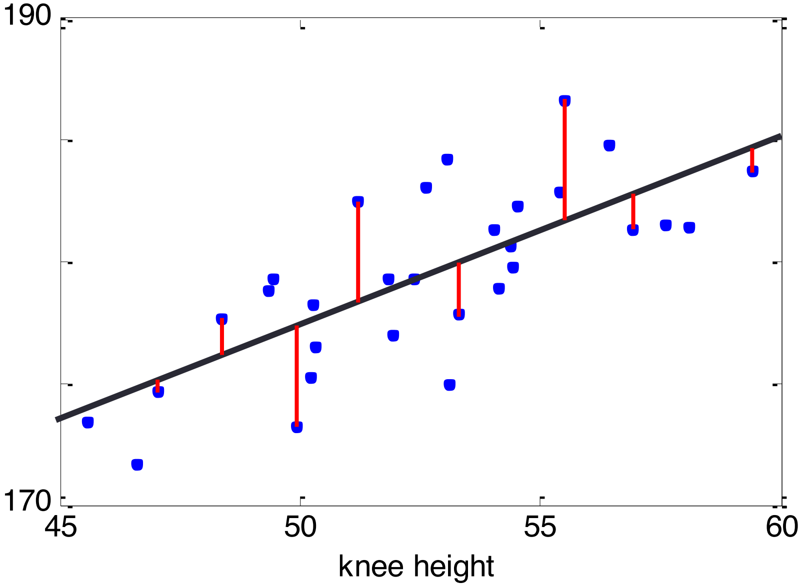
cost function

mean squared error

$\theta_0 = 140$

$\theta_1 = 0.75$

$J(\theta_0, \theta_1) = 5.94$

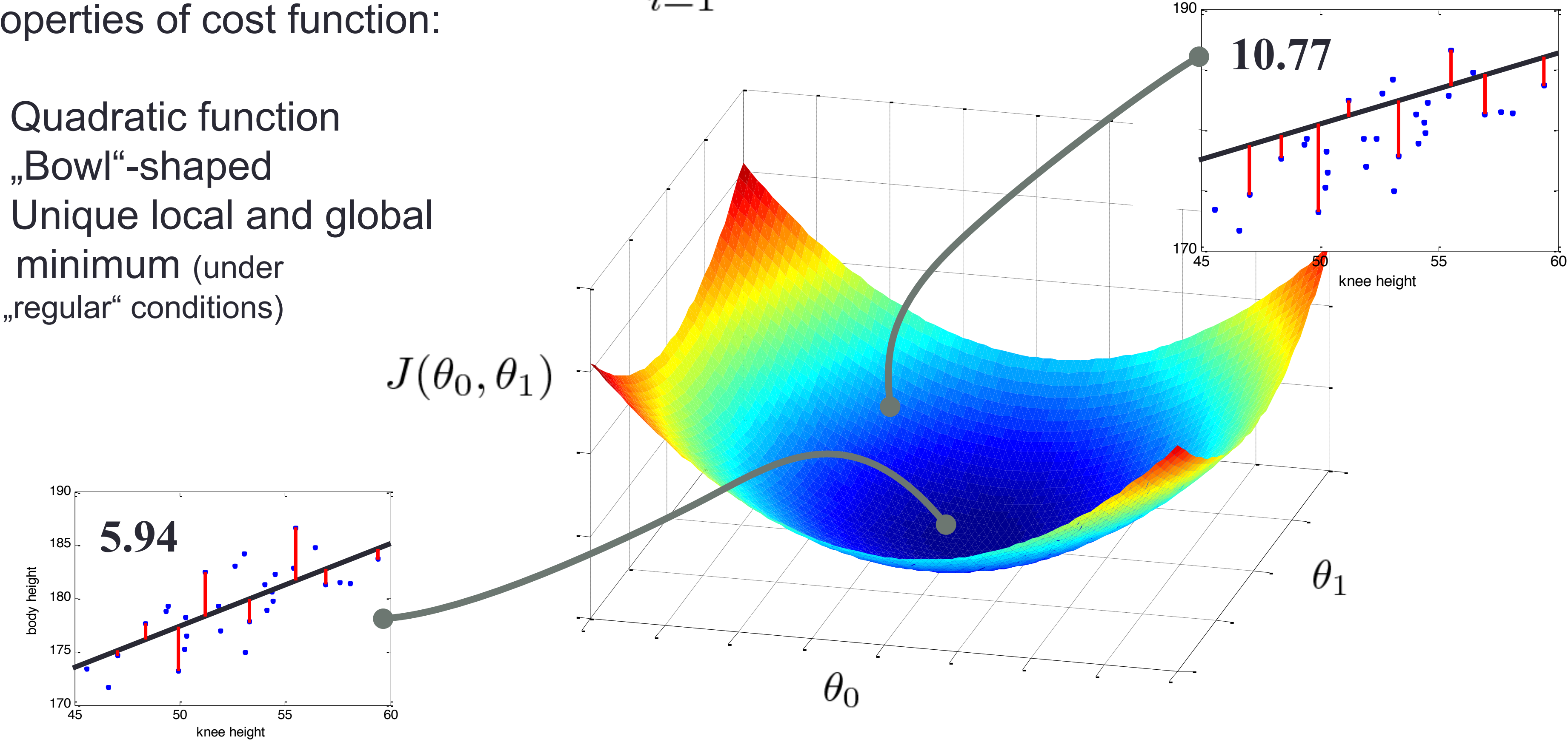


Cost function illustrated

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\theta}} \left(x^{(i)} \right) - y^{(i)} \right)^2$$

Properties of cost function:

- Quadratic function
- „Bowl“-shaped
- Unique local and global minimum (under „regular“ conditions)



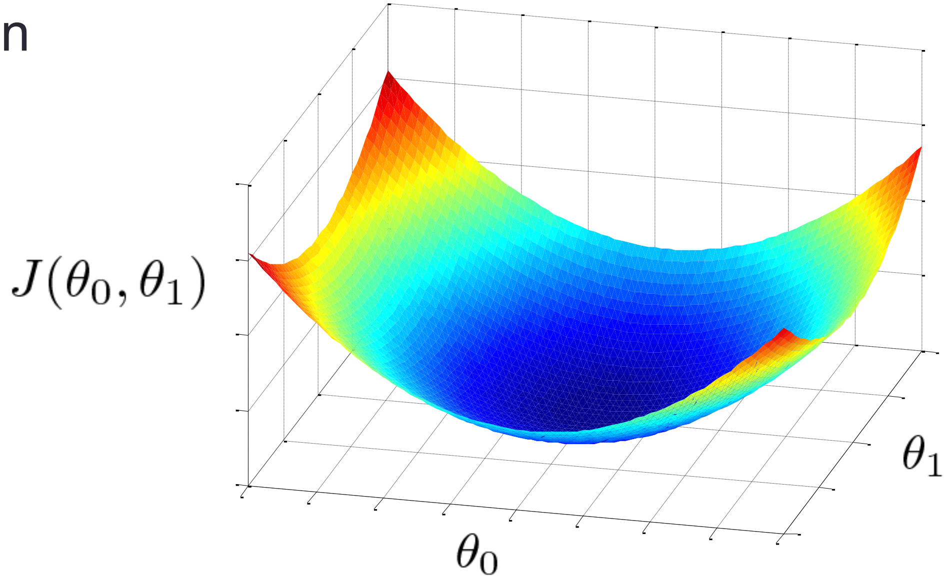
Minimizing the cost

- Two ways to find the parameters $\boldsymbol{\theta} = (\theta_0, \theta_1)$ minimizing

$$J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}\right)^2$$

  - Gradient descent
  - Direct analytical solution

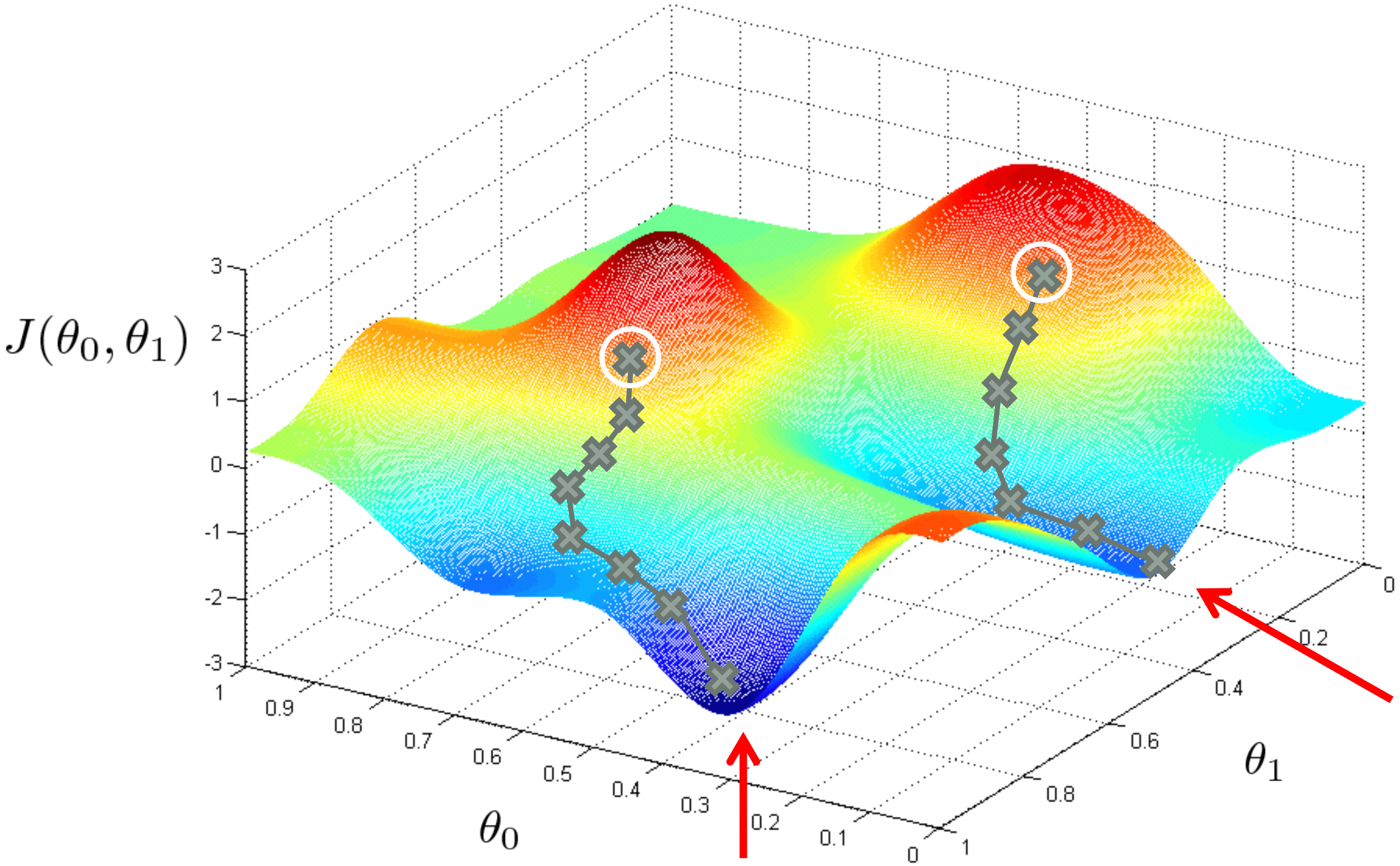
(setting derivatives = 0)



GRADIENT DESCENT

Descending in the steepest direction

Gradient descent on some arbitrary cost function $J(\theta_0, \theta_1)$ …



Gradient descent algorithm

- Repeat until convergence

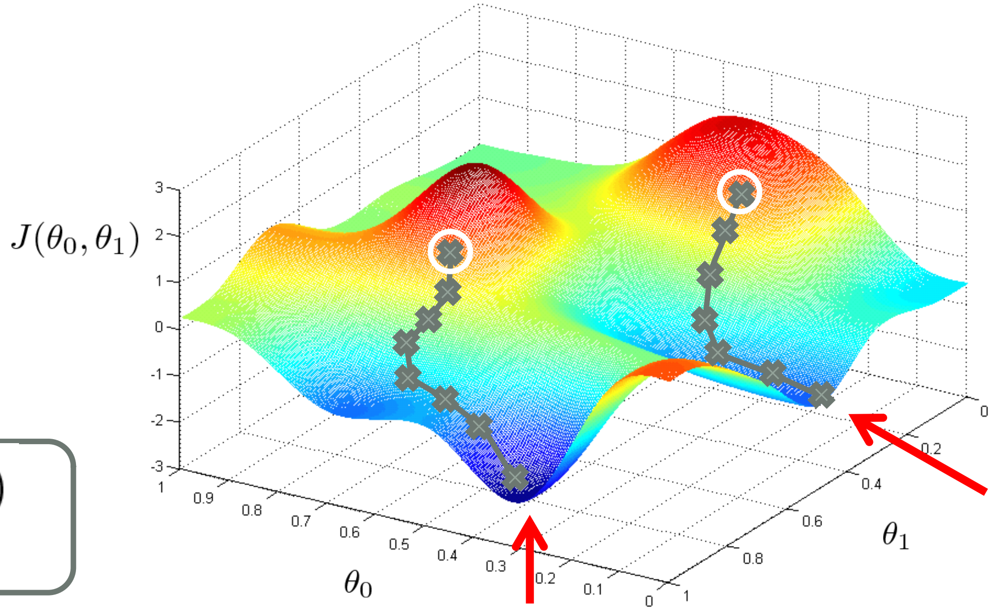
$$\theta_j := \theta_j - \eta \cdot \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(***simultaneously*** *updating* θ_0 *and* θ_1)

negative gradient = descent

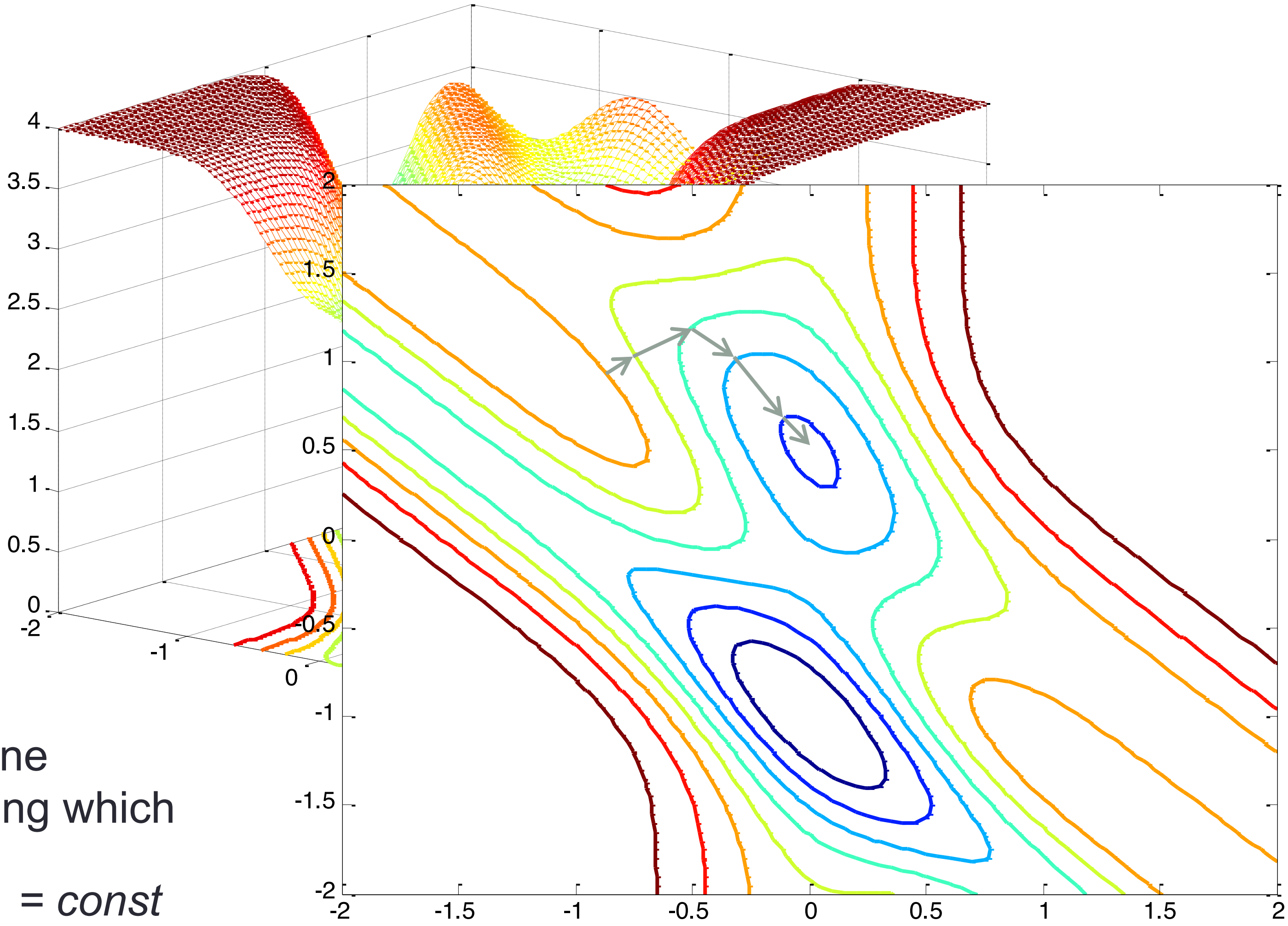
learning rate („eta“)

partial derivative of $J(\theta_0, \theta_1)$ with respect to θ_j



Gradient is orthogonal to contour lines

$J(\theta_0, \theta_1)$

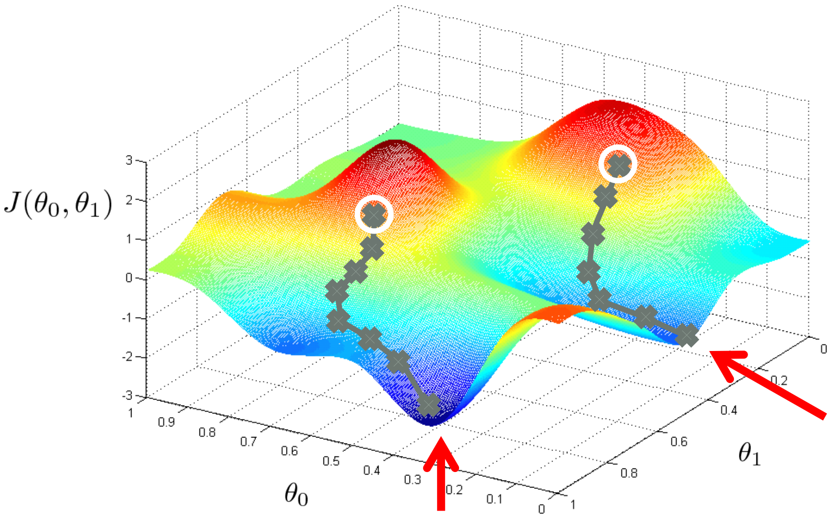


A contour line
is a line along which

$J(\theta_0, \theta_1)$ = *const*

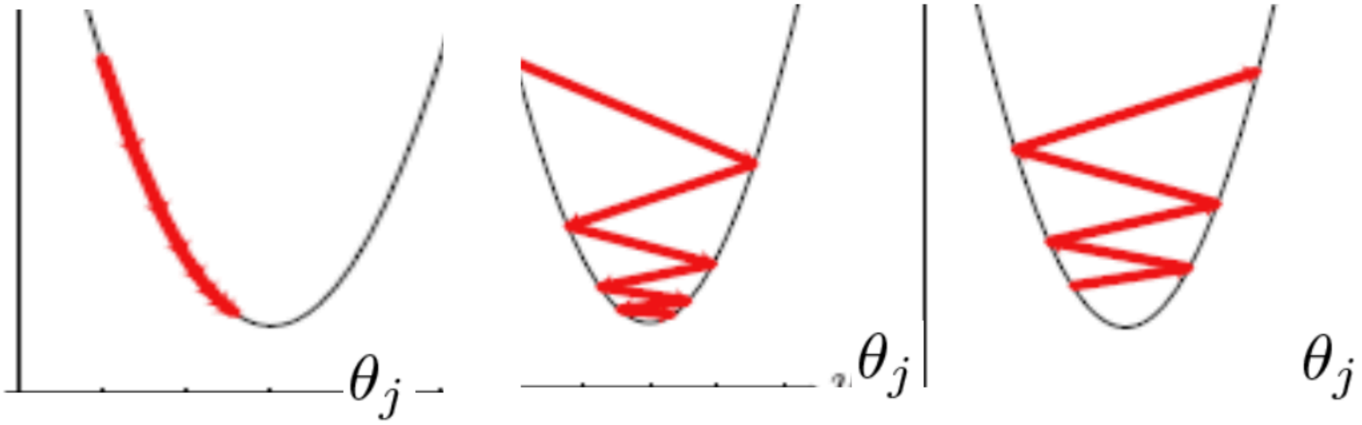
Potential issues with gradient descent

- May get stuck in local minima
- Learning rate too small: slow convergence
- Learning rate too large: oscillations, divergence



η too small

η too large



LINEAR REGRESSION WITH GRADIENT DESCENT

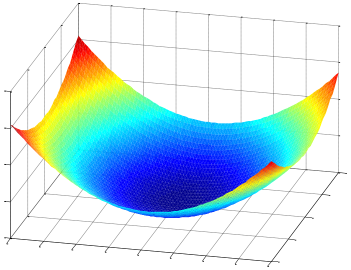
(ONE INPUT)

Application of gradient descent

- Linear regression cost

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)} \right)^2$$

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot x$$



- Gradient descent

$$\theta_j := \theta_j - \eta \cdot \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

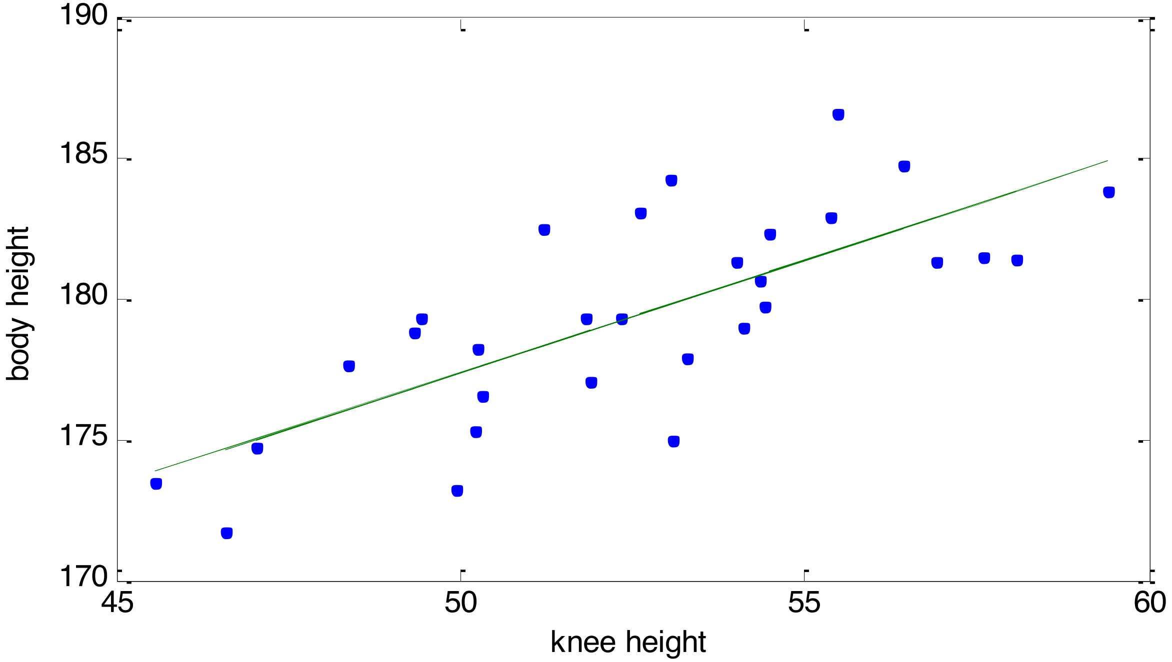
(simultaneous update)

$$\left.\begin{aligned} \theta_0 &:= \theta_0 - \underbrace{2\eta}_{\text{"learning rate"}} \cdot \frac{1}{m} \sum_{i=1}^{m} \underline{\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)} \right)} \\ \theta_1 &:= \theta_1 - \underline{2\eta} \cdot \frac{1}{m} \sum_{i=1}^{m} \underbrace{\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)} \right)}_{\text{"error"}} \cdot \underbrace{x^{(i)}}_{\text{"input"}} \end{aligned}\right\}$$

(simultaneous update)

Predicting height from knee height

- Optimal fit to training data



$$\theta_0 = 137.4$$

$$\theta_1 = 0.8$$

LINEAR REGRESSION

MORE GENERAL FORMULATION: MULTIPLE FEATURES

Multiple inputs (features)

Knee Height x_1	Arm span x_2	Age x_3	Height y
50	166	32	171
56	172	17	175
52	174	62	168
…	…	…	…

m

n = 3

$$\boldsymbol{x}^{(2)} = \begin{pmatrix} 56 \\ 172 \\ 17 \end{pmatrix}$$

$$x_3^{(2)} = 17$$

- Notation:

m … number of training examples
n … number of features

$\boldsymbol{x}^{(i)}$ … input features of *i*'th training example (vector-valued)
$x_j^{(i)}$ … value of feature *j* in *i*'th training example

Linear hypothesis

- Hypothesis (one input):

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot x$$

- Hypothesis (multiple input features):

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 \cdot x_1 + \cdots + \theta_n \cdot x_n$$

Example: h(x) = 50 + 0.5*kneeheight + 0.3*armspan + 0.1*age

- More compact notation:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{\theta}$$

$$\boldsymbol{x} = \begin{pmatrix} \boxed{x_0} \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \quad \boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix}$$

Introduce $\boxed{x_0 = 1}$

Why? Notation convenience!

Multiple inputs (features) revisited

x_0	Knee Height x_1	Arm span x_2	Age x_3	Height y
1	50	166	32	171
1	56	172	17	175
1	52	174	62	168
1	…	…	…	…

m

$n = 3$

$$\boldsymbol{x}^{(2)} = \begin{pmatrix}\boxed{\begin{matrix}1\\56\\172\\17\end{matrix}}\end{pmatrix}$$

$$x_0^{(2)} = 1$$

$$x_3^{(2)} = 17$$

- Notation:

m … number of training examples

n … number of features

$\boldsymbol{x}^{(i)}$ … input features of *i*'th training example (vector-valued)

$x_j^{(i)}$ … value of feature *j* in *i*'th training example

Matrix and vector notation

x_0	Knee Height x_1	Arm span x_2	Age x_3	Height y
1	50	166	32	171
1	56	172	17	175
1	52	174	62	168

$$\boldsymbol{X} = \begin{pmatrix} \boxed{\begin{matrix} 1 & 50 & 166 & 32 \end{matrix}} \\ \begin{matrix} 1 & 56 & 172 & 17 \end{matrix} \\ \begin{matrix} 1 & 52 & 174 & 62 \end{matrix} \end{pmatrix}$$

$$\boldsymbol{y} = \begin{pmatrix} 171 \\ 175 \\ 168 \end{pmatrix}$$

$$\boldsymbol{x}^{(i)} = \begin{pmatrix} \boxed{\begin{matrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ x_n^{(i)} \end{matrix}} \end{pmatrix} \qquad \boldsymbol{X} = \begin{pmatrix} \boxed{\text{———} \left(\boldsymbol{x}^{(1)}\right)^T \text{———}} \\ \text{———} \left(\boldsymbol{x}^{(2)}\right)^T \text{———} \\ \vdots \\ \text{———} \left(\boldsymbol{x}^{(m)}\right)^T \text{———} \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{pmatrix}$$

features of i‘th training example

(n+1) × 1

design matrix

m × (n+1)

output/target vector

m × 1

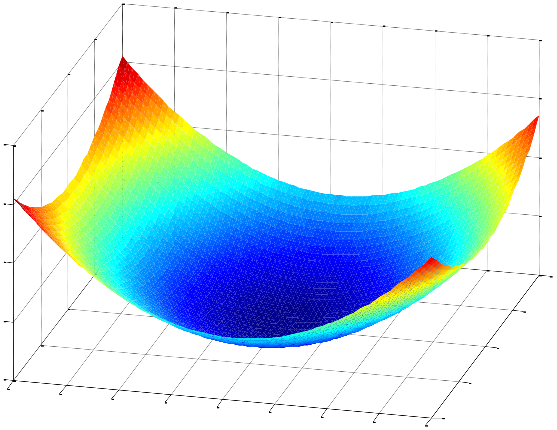
LINEAR REGRESSION WITH GRADIENT DESCENT

(GENERAL FORMULATION)

Linear regression problem statement

- Hypothesis: $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\theta}$
- Cost function: $J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$

high-dimensional quadratic („bowl“-shaped) function



Goal is to find parameters which minimize the cost

Gradient descent (multiple features)

with ***one*** input feature:

$$\left.\begin{aligned}\theta_0 &:= \theta_0 - \underbrace{2\eta}_{\text{"learning rate"}} \cdot \frac{1}{m} \sum_{i=1}^{m} \underline{\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}\right)} \\ \theta_1 &:= \theta_1 - \underline{2\eta} \cdot \frac{1}{m} \sum_{i=1}^{m} \underbrace{\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}\right)}_{\text{"error"}} \cdot \underbrace{x^{(i)}}_{\text{"input"}}\end{aligned}\right\}$$

(simultaneous update)

with ***n*** input features:

$$\theta_j := \theta_j - \underbrace{2\eta}_{\text{"learning rate"}} \cdot \frac{1}{m} \sum_{i=1}^{m} \underbrace{\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)}_{\text{"error"}} \cdot \underbrace{x_j^{(i)}}_{\text{"input"}}$$

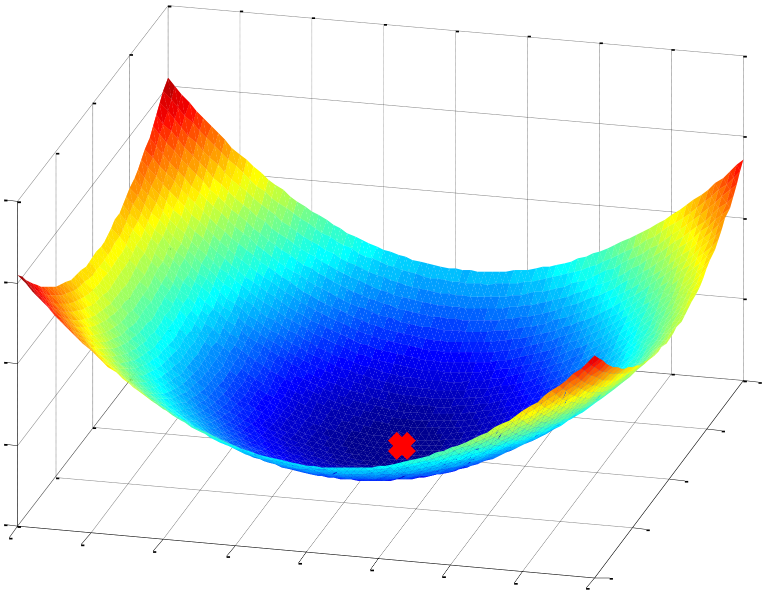
(simultaneous update for j=0…n)

For j = 0: define for convenience $x_0^{(i)} = 1$

LINEAR REGRESSION ANALYTICAL SOLUTION

Analytical solution

- Set all partial derivatives of cost function $J(\boldsymbol{\theta})$ = 0



- Solving system of linear equations yields:

$$\boxed{\boldsymbol{\theta}^* = \underbrace{\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T}_{\text{Moore-Penrose Pseudoinverse of } \boldsymbol{X}}\boldsymbol{y}}$$

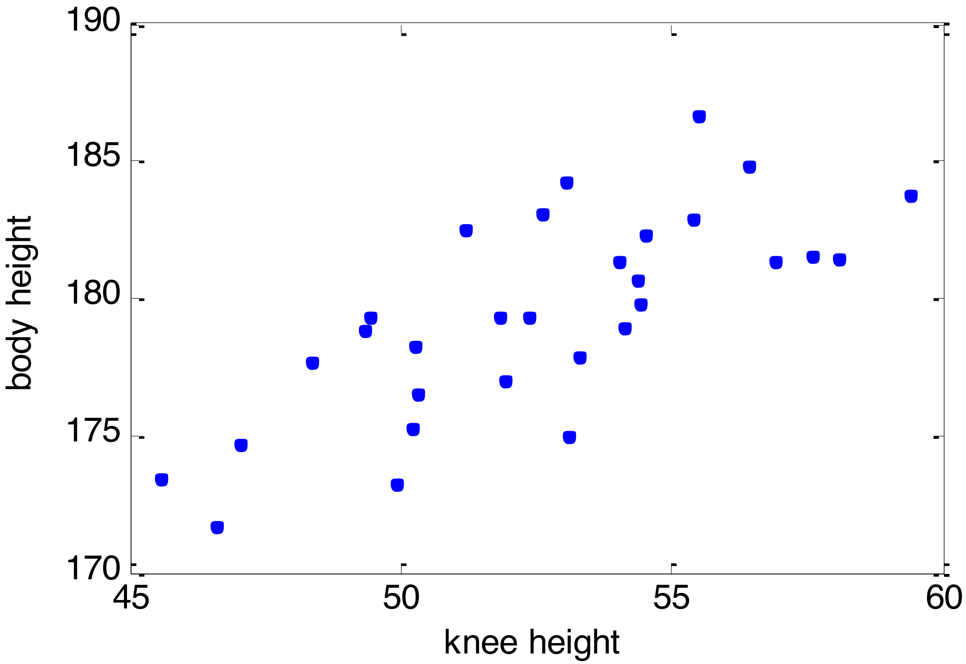
$\boldsymbol{X}$ … design matrix

$\boldsymbol{y}$ … output/target vector

- *Note: This analytical solution requires that columns of* $\boldsymbol{X}$ *are linearly independent („regular“ conditions)*

Example: analytical solution applied to problem with one input

Knee Height [cm]	Height [cm]
50	171
56	175
52	168
…	…



Example: analytical solution applied to problem with one input

Knee Height [cm]	Height [cm]
50	171
56	175
52	168
…	…

$$\boldsymbol{X} = \begin{pmatrix} 1 & 50 \\ 1 & 56 \\ 1 & 52 \\ \vdots & \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} 171 \\ 175 \\ 168 \\ \vdots \end{pmatrix}$$

30 × 2 *30 × 1*

$$\boldsymbol{\theta}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

$$= \begin{pmatrix} 137.4 \\ 0.8 \end{pmatrix}$$

$$\boldsymbol{X}^T\boldsymbol{X} = \begin{pmatrix} 30 & 1577 \\ 1577 & 83222 \end{pmatrix}$$

2 × 2

$$(\boldsymbol{X}^T\boldsymbol{X})^{-1} = \begin{pmatrix} 7.994 & -0.152 \\ -0.152 & 0.003 \end{pmatrix}$$

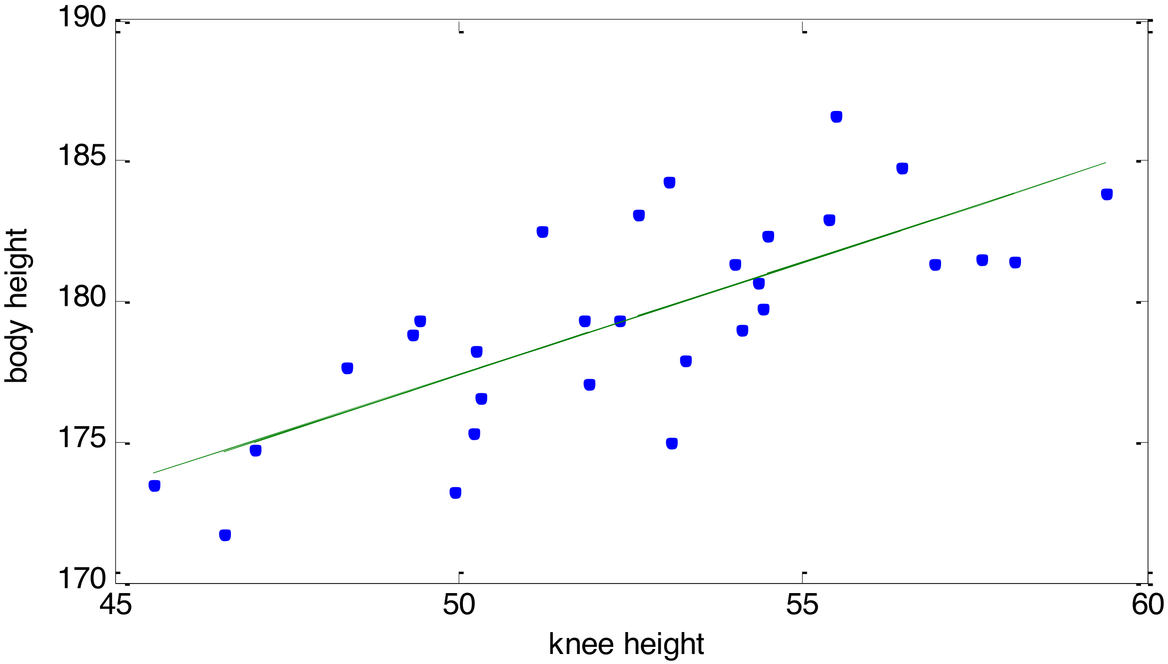
2 × 2

$$\boldsymbol{X}^T\boldsymbol{y} = \begin{pmatrix} 5383 \\ 283210 \end{pmatrix}$$

2 × 1

Predicting height from knee height

$$\theta_0 = 137.4$$
$$\theta_1 = 0.8$$



$$\boldsymbol{\theta}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$
$$= \begin{pmatrix} 137.4 \\ 0.8 \end{pmatrix}$$

Gradient descent	Analytical solution
Need to choose learning rate η	No need to choose η
Iterative algorithm (needs many iterations to converge)	Direct solution (no iteration)
Works well even when number of input features is large n	Slow if n is too large (inverting n x n matrix)

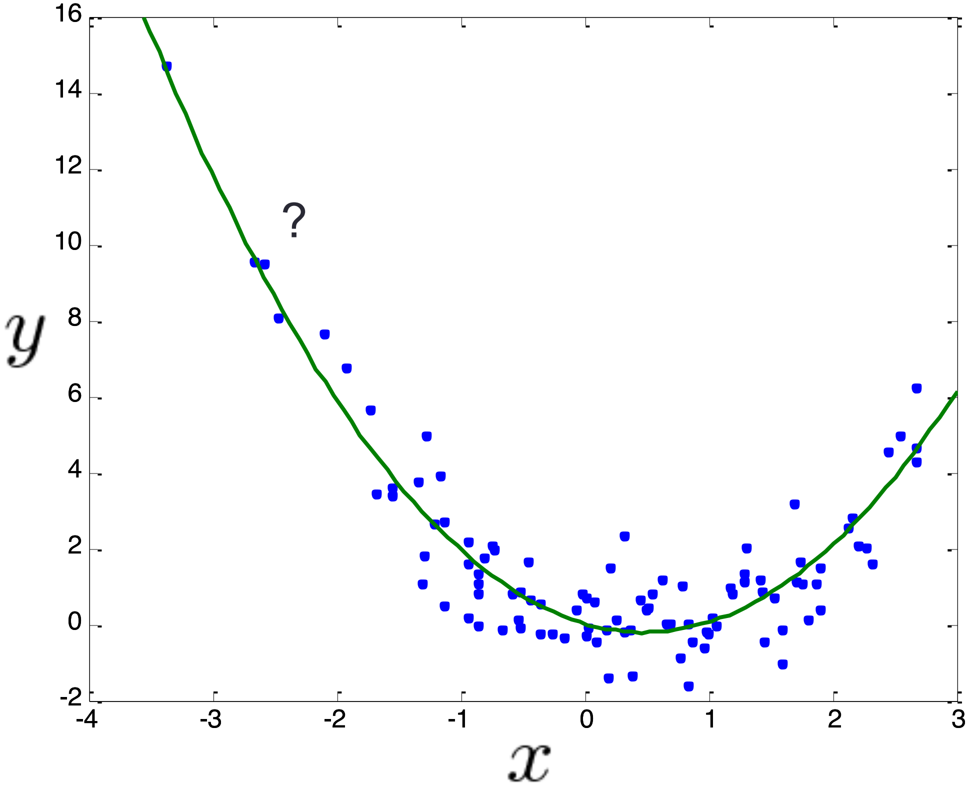
NON-LINEAR FEATURES

(NON-LINEAR BASIS FUNCTIONS)

Non-linear trends in data

- How can we learn non-linear hypotheses?

x	y
0.01	-0.27
-1.22	2.63
0.17	-0.13
…	…



$$h_{\boldsymbol{\theta}}(x) = \overset{?}{\theta_0} + \overset{?}{\theta_1} \cdot x + \overset{?}{\theta_2} \cdot x^2$$

Linear fit to this “non-linear” data

x	y
0.01	-0.27
-1.22	2.63
0.17	-0.13
…	…

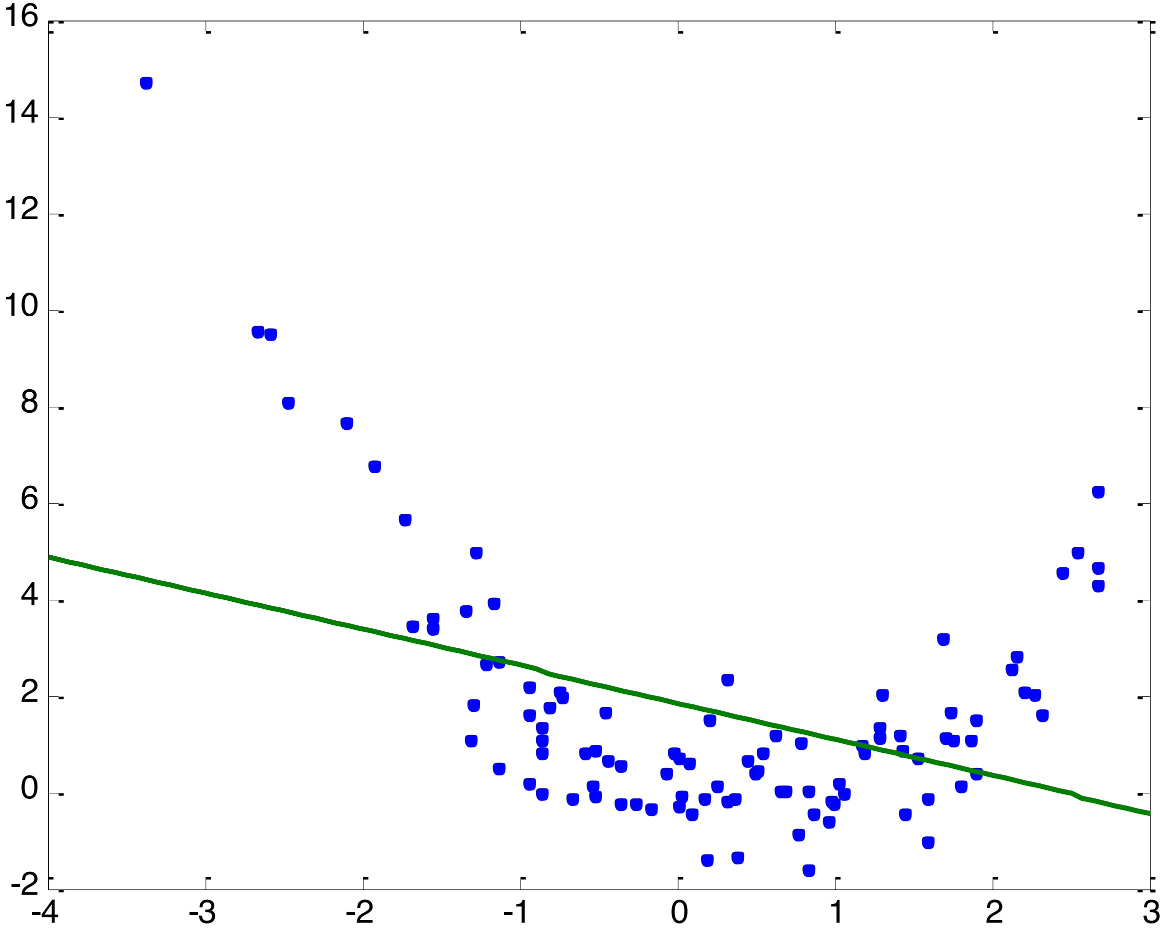
$$\boldsymbol{X} = \begin{pmatrix} 1 & 0.01 \\ 1 & -1.22 \\ 1 & 0.17 \\ & \vdots \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} -0.27 \\ 2.63 \\ -0.13 \\ \vdots \end{pmatrix}$$

standard design matrix

Hypothesis: $h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot x$

Optimal parameters: $\boldsymbol{\theta}^* = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$

Linear fit to this "non-linear" data



$$h_{\boldsymbol{\theta}}(x) = 1.85 - 0.76 \cdot x$$

Non-linear (quadratic) fit

x	y
0.01	-0.27
-1.22	2.63
0.17	-0.13
…	…

$$\phi_0 = 1 \qquad \phi_1 = x \qquad \phi_2 = x^2$$

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & 0.01 & 0.01^2 \\ 1 & -1.22 & (-1.22)^2 \\ 1 & 0.17 & (0.17)^2 \\ & \vdots & \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} -0.27 \\ 2.63 \\ -0.13 \\ \vdots \end{pmatrix}$$

design matrix with non-linear features

Hypothesis: $h_{\boldsymbol{\theta}}(\boldsymbol{\phi}) = \theta_0 + \theta_1 \cdot \phi_1 + \theta_2 \cdot \phi_2$

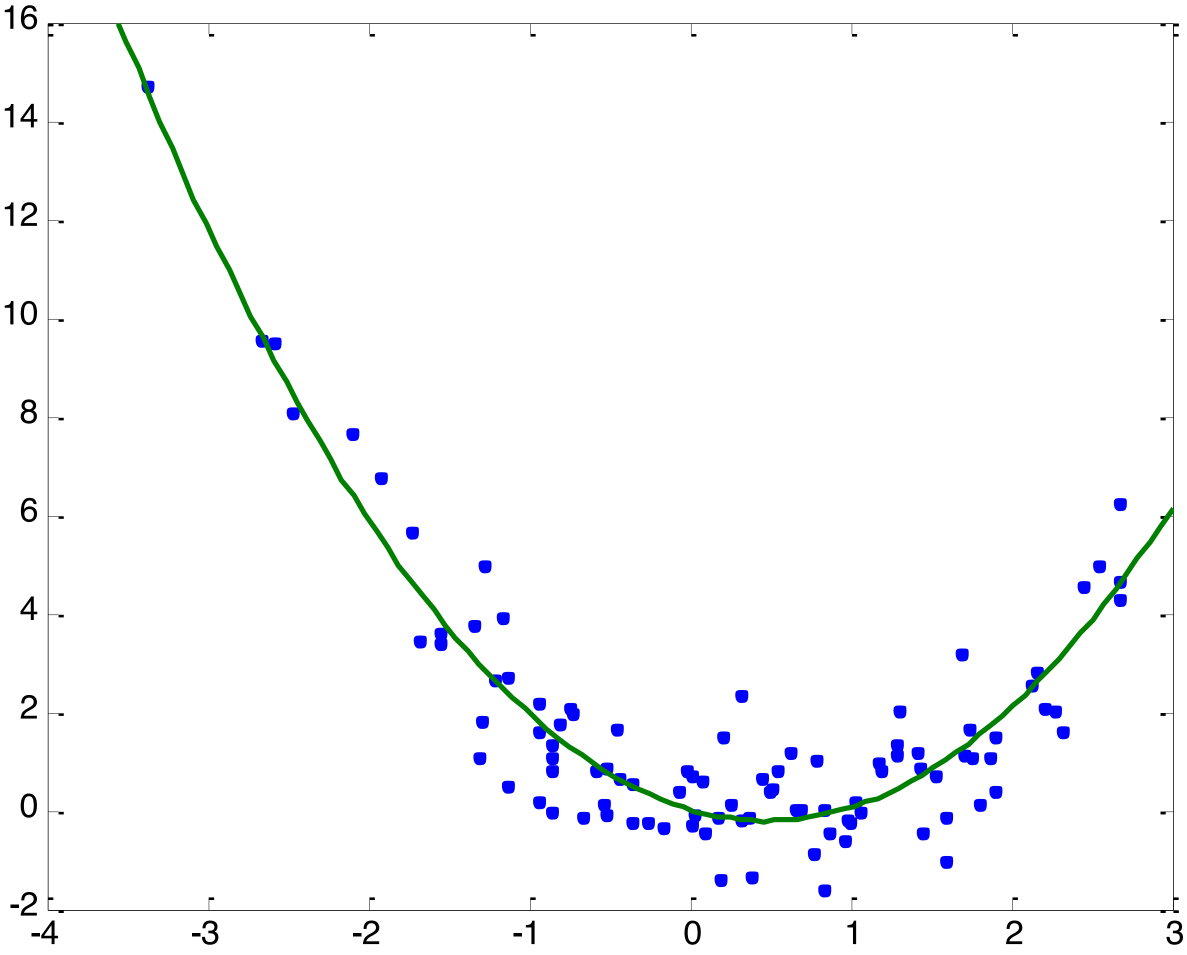
Optimal parameters: $\boldsymbol{\theta}^* = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{y}$

Non-linear (quadratic) fit

$$\phi_0 = 1$$

$$\phi_1 = x$$

$$\phi_2 = x^2$$



$$h_{\boldsymbol{\theta}}(x) = 0.02 \cdot 1 - 0.95 \cdot x + 0.99 \cdot x^2$$

Non-linear (sinusoid) fit

x	y
0.01	-0.27
-1.22	2.63
0.17	-0.13
…	…

$$\phi_0 = 1 \qquad \phi_1 = x \qquad \phi_2 = \cos(x)$$

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & 0.01 & \cos(0.01) \\ 1 & -1.22 & \cos(-1.22) \\ 1 & 0.17 & \cos(0.17) \\ & \vdots & \end{pmatrix} \quad \boldsymbol{y} = \begin{pmatrix} -0.27 \\ 2.63 \\ -0.13 \\ \vdots \end{pmatrix}$$

design matrix with non-linear features

Hypothesis: $h_{\boldsymbol{\theta}}(\boldsymbol{\phi}) = \theta_0 + \theta_1 \cdot \phi_1 + \theta_2 \cdot \phi_2$

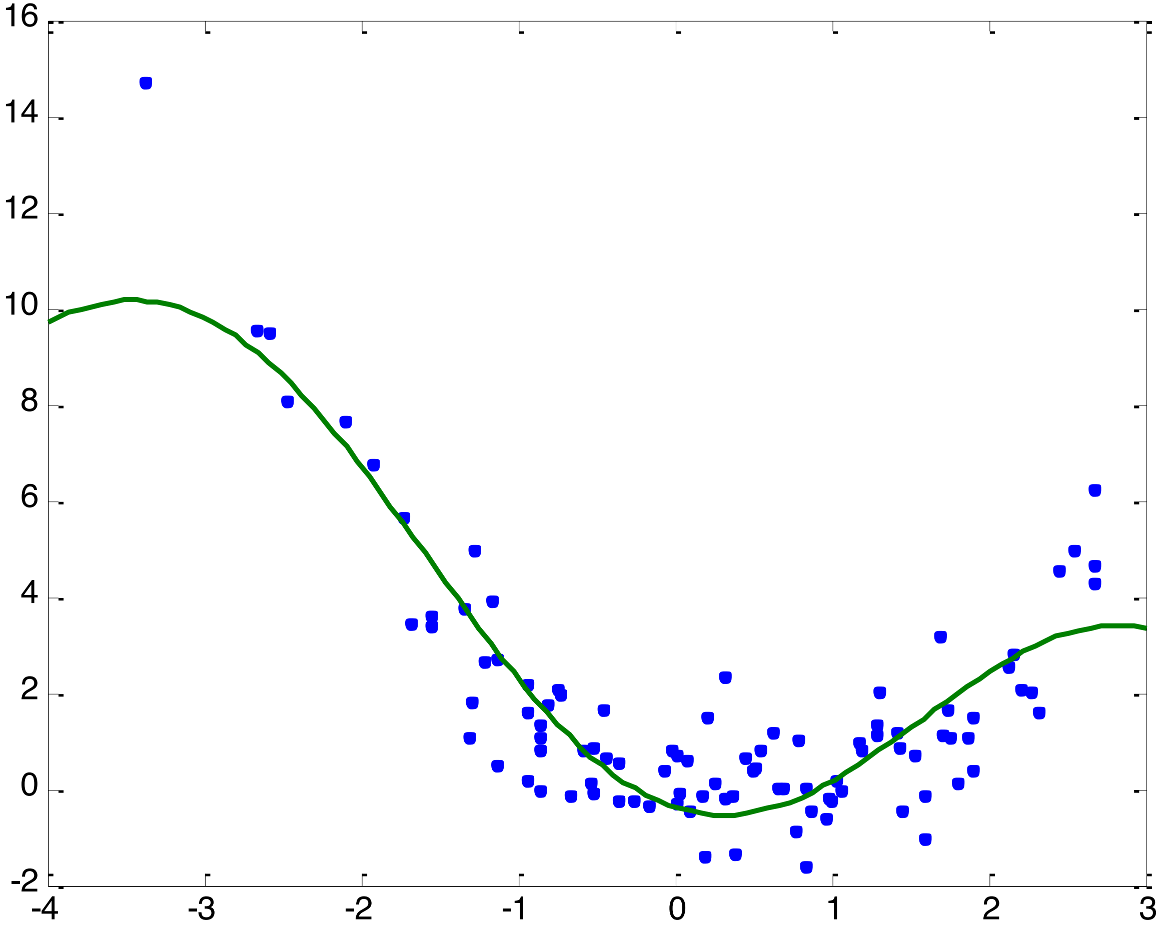
Optimal parameters: $\boldsymbol{\theta}^* = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{y}$

Non-linear (sinusoidal) fit

$$\phi_0 = 1$$

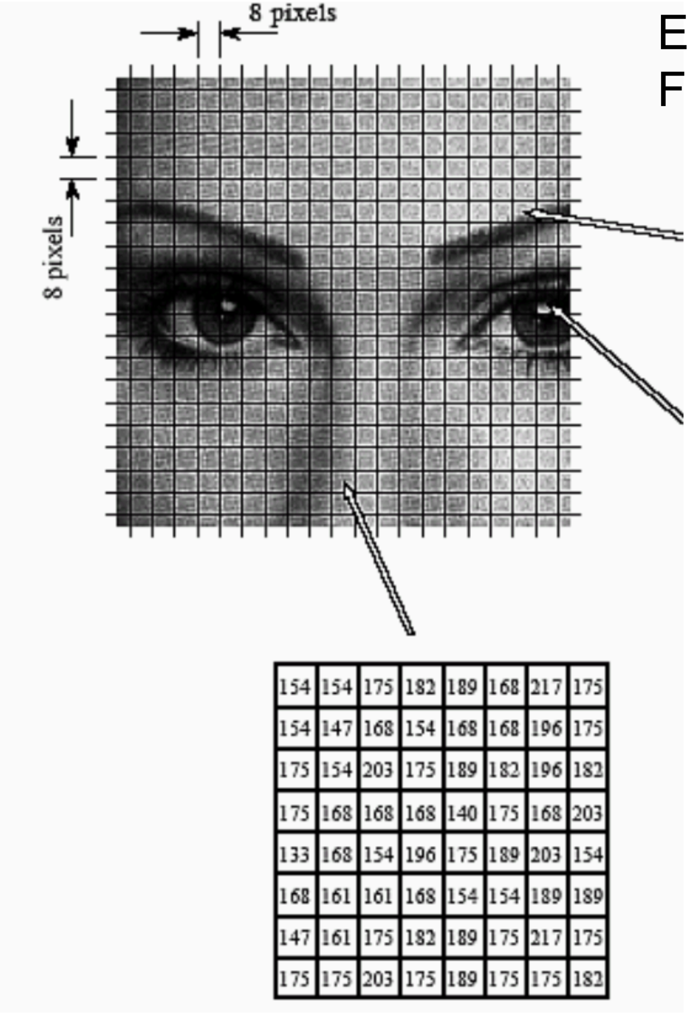
$$\phi_1 = x$$

$$\phi_2 = \cos(x)$$



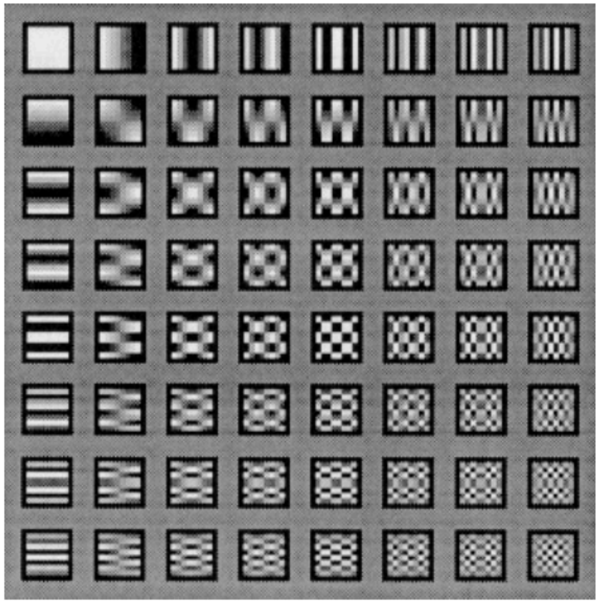
$$h_{\boldsymbol{\theta}}(x) = 3.12 \cdot 1 - 1.07 \cdot x - 3.5 \cdot \cos(x)$$

Image: JPEG = cosine-basis



154	154	175	182	189	168	217	175
154	147	168	154	168	168	196	175
175	154	203	175	189	182	196	182
175	168	168	168	140	175	168	203
133	168	154	196	175	189	203	154
168	161	161	168	154	154	189	189
147	161	175	182	189	175	217	175
175	175	203	175	189	175	175	182

Each block of 8x8 pixels is represented in a Fourier basis of cosine filters



Better representation of edges and corners Allows for compression

Non-linear input features (in general)

feature 2 of all training examples

all features of 1st training example

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & \phi_1^{(1)} & \phi_2^{(1)} & \cdots & \phi_n^{(1)} \\ 1 & \phi_1^{(2)} & \phi_2^{(2)} & \cdots & \phi_n^{(2)} \\ & & \vdots & & \\ 1 & \phi_1^{(m)} & \phi_2^{(m)} & \cdots & \phi_n^{(m)} \end{pmatrix}$$

- Feature 2 for each training example *i* is computed by applying a **non-linear basis function**:

$$\phi_2^{(i)} = \phi_2(\boldsymbol{x}^{(i)})$$

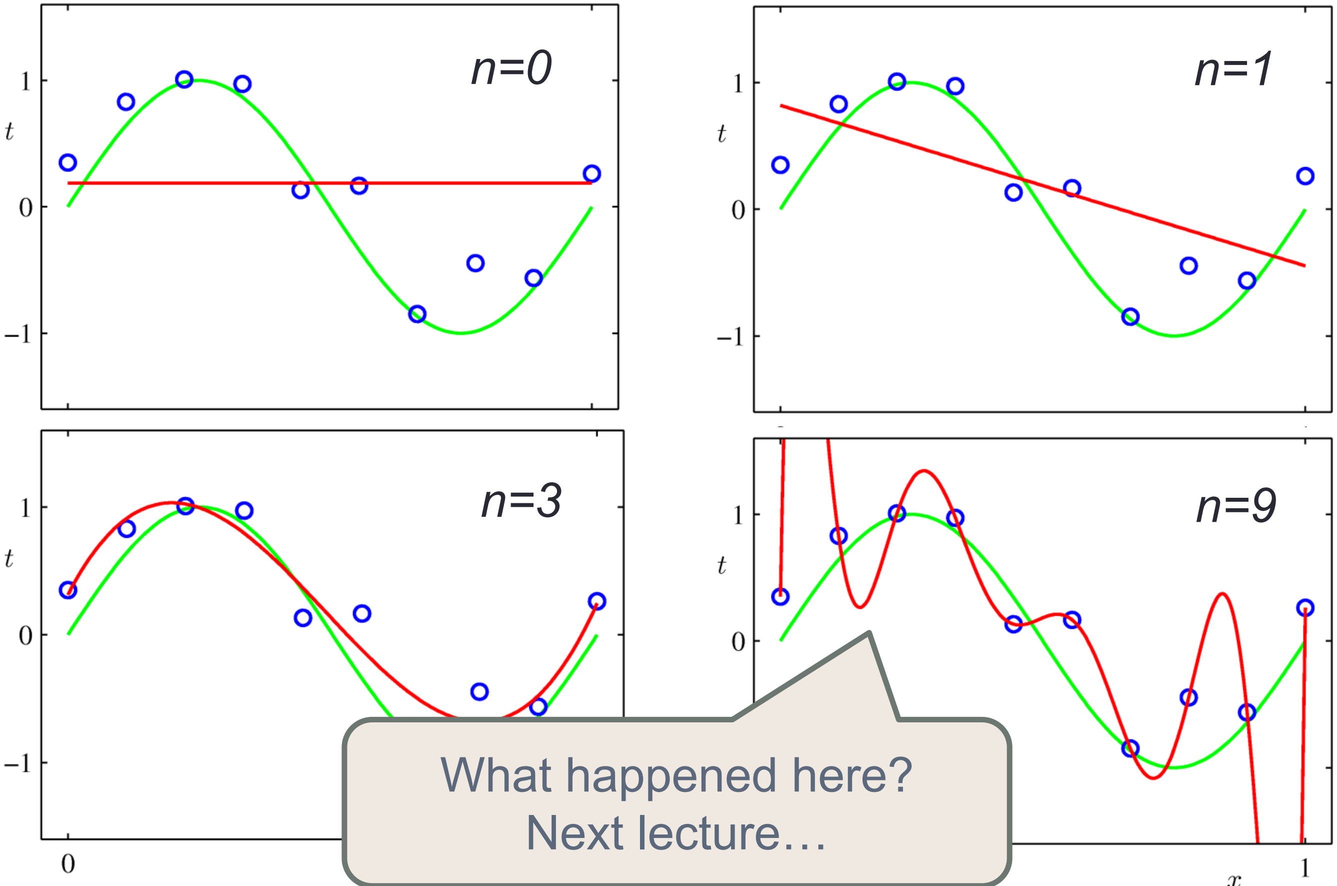
- Allows to learn a variety of non-linear functions with the same technique(s):
  - Analytical $\boldsymbol{\theta}^* = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{y}$ *or* gradient descent

Polynomial regression

n = degree of polynome
to be learned

- Features are powers of x

$$\phi_0 = x^0, \phi_1 = x^1, \phi_2 = x^2, \ldots, \phi_n = x^n$$

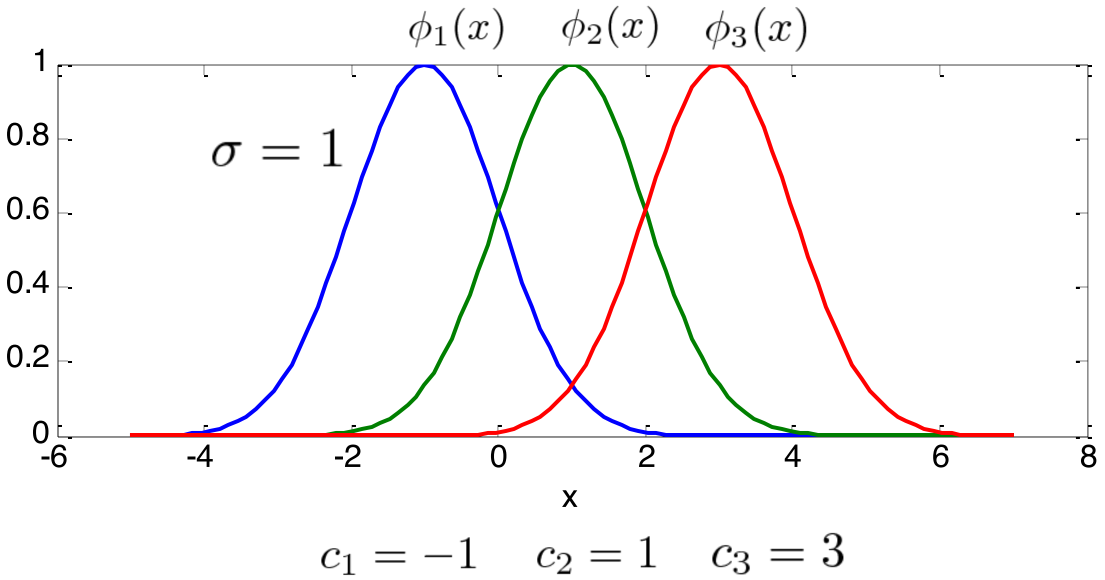


What happened here?
Next lecture…

Radial basis functions

- „Gaussian“-shaped RBFs:
  - Each basis function j has a **center** $\boldsymbol{c}_j$ in the input space
  - The **width** of the basis functions is determined by σ.

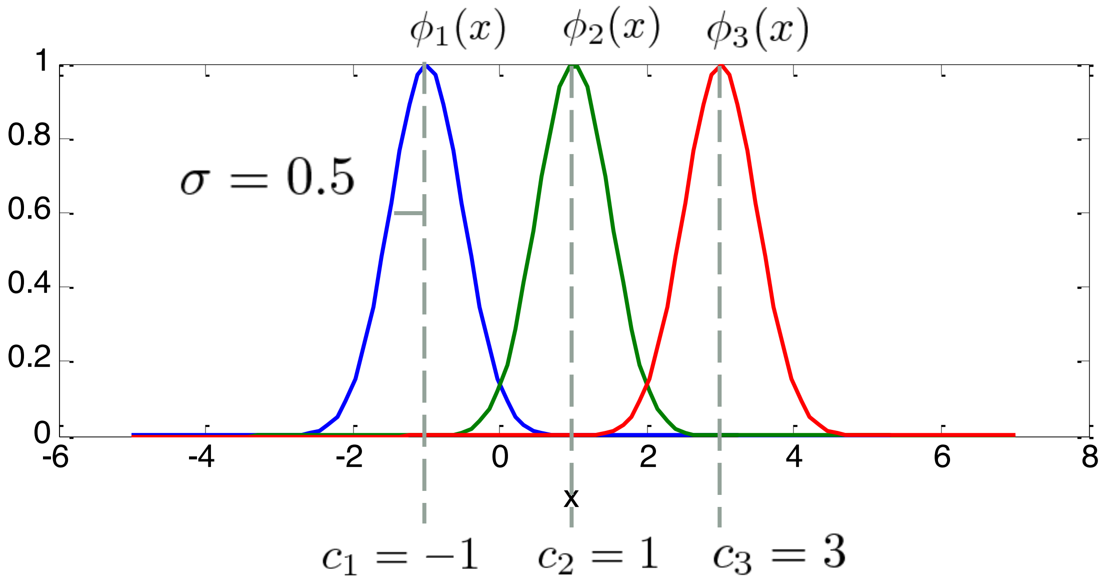
$$\phi_j(\boldsymbol{x}) = \exp\left(-\frac{1}{2\sigma^2} \cdot \|\boldsymbol{x} - \boldsymbol{c}_j\|^2\right)$$



Radial basis functions

- „Gaussian"-shaped RBFs:
  - Each basis function j has a **center** $\boldsymbol{c}_j$ in the input space
  - The **width** of the basis functions is determined by σ.

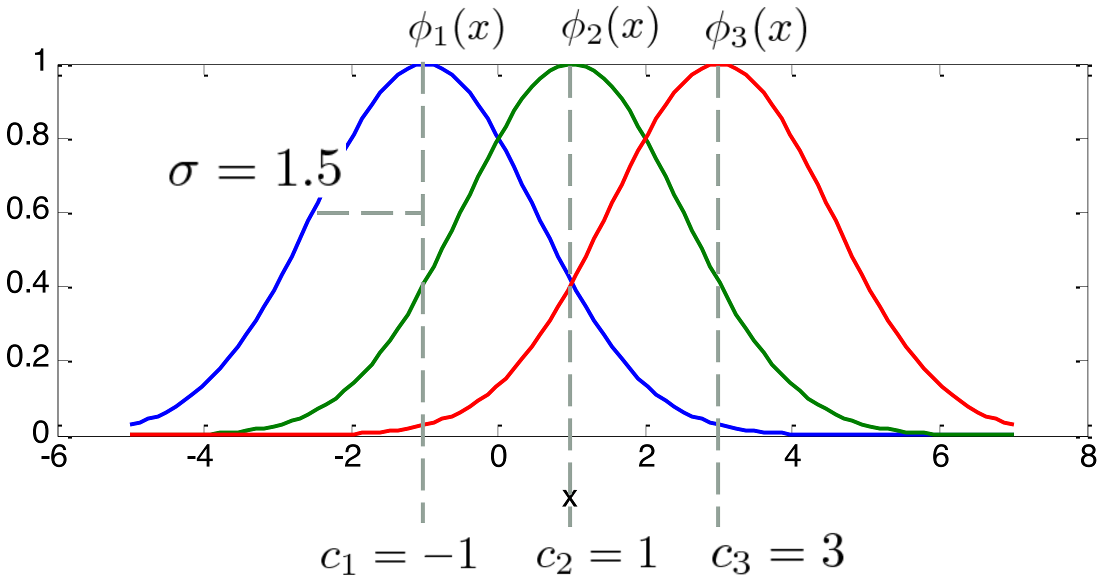
$$\phi_j(\boldsymbol{x}) = \exp\left(-\frac{1}{2\sigma^2} \cdot \|\boldsymbol{x} - \boldsymbol{c}_j\|^2\right)$$



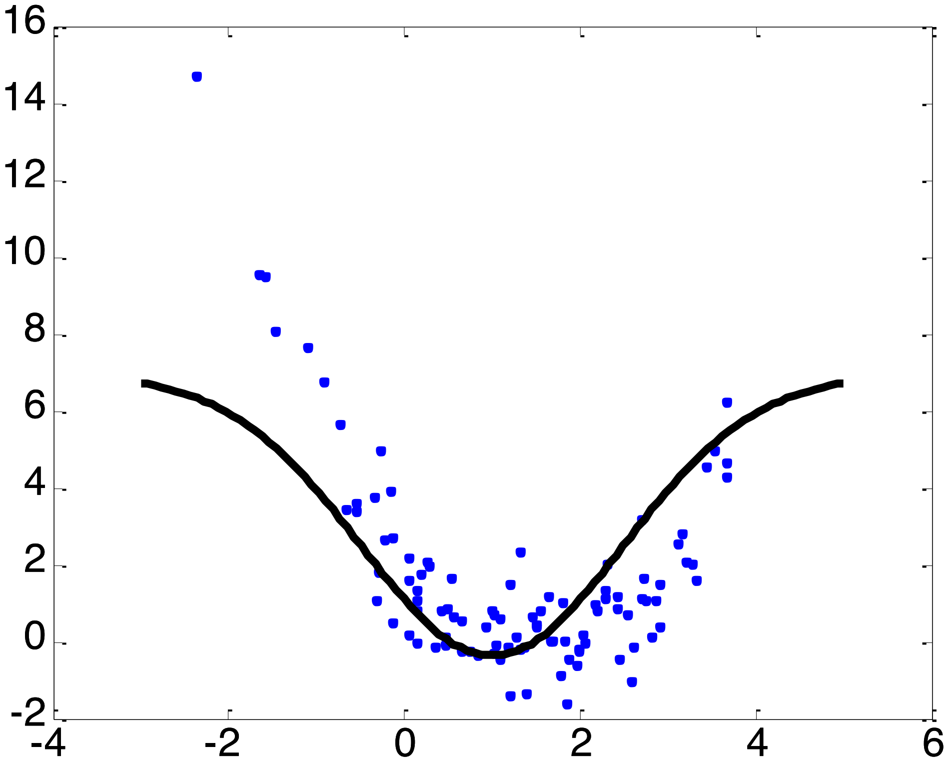
Radial basis functions

- „Gaussian“-shaped RBFs:
  - Each basis function j has a **center** $\boldsymbol{c}_j$ in the input space
  - The **width** of the basis functions is determined by σ.

$$\phi_j(\boldsymbol{x}) = \exp\left(-\frac{1}{2\sigma^2} \cdot \|\boldsymbol{x} - \boldsymbol{c}_j\|^2\right)$$



Fitting a single RBF to data

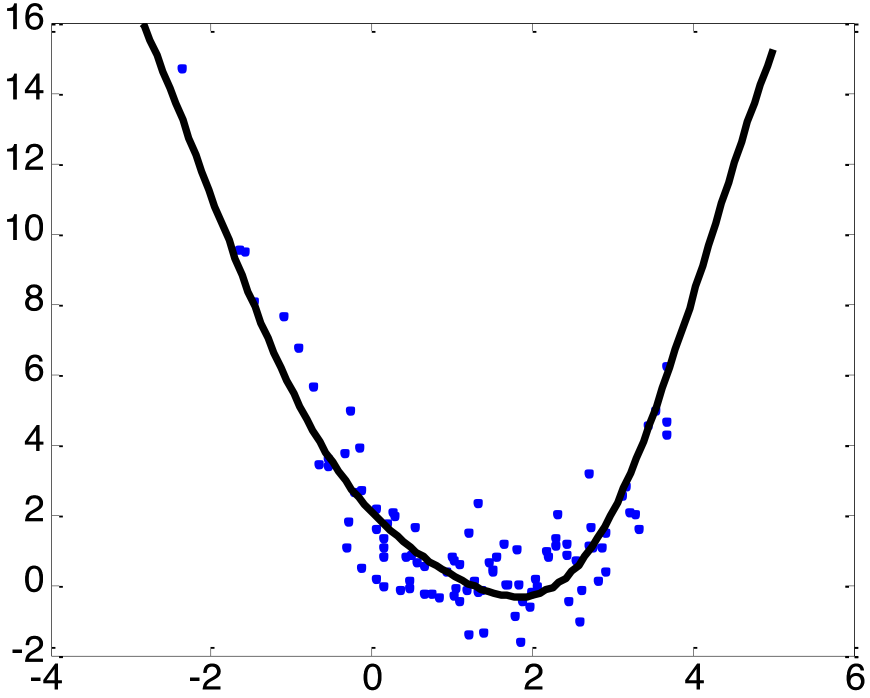


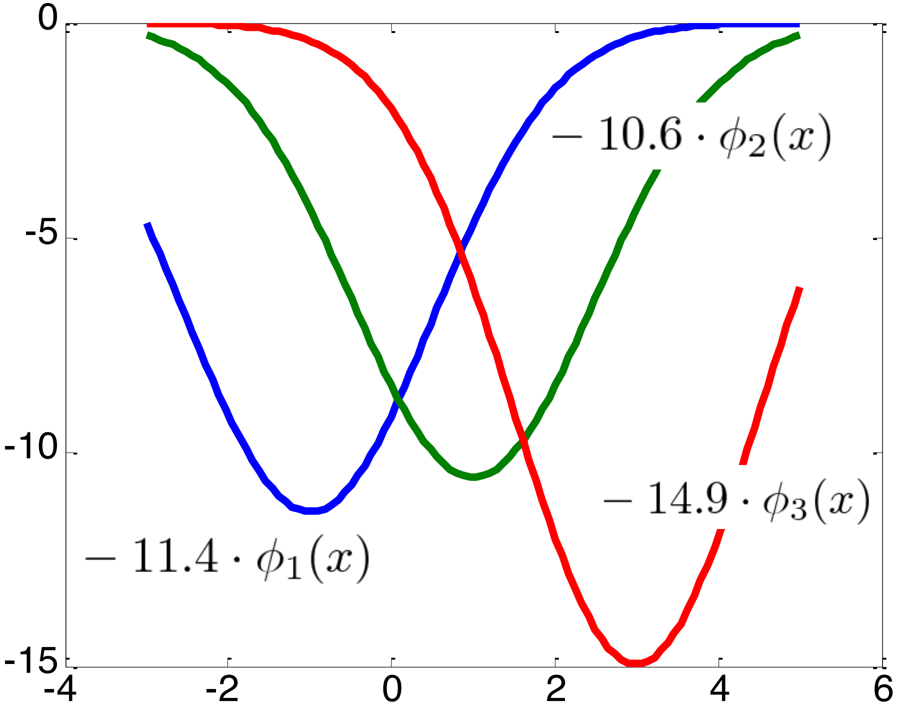
RBF with $c_1 = 1$ $\sigma = 1.5$

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot \phi_1(x)$$

$$h_{\boldsymbol{\theta}}(x) = 6.9 - 7.3 \cdot \phi_1(x)$$

Fitting RBFs to data





RBFs with $\sigma = 1.5$

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 \cdot \phi_1(x) + \theta_2 \cdot \phi_2(x) + \theta_3 \cdot \phi_3(x)$$

$$h_{\boldsymbol{\theta}}(x) = 21.7 - 11.4 \cdot \phi_1(x) - 10.6 \cdot \phi_2(x) - 14.9 \cdot \phi_3(x)$$

SUMMARY (QUESTIONS)

Some questions…

- Hypothesis for linear regression = ?
- Cost function for linear regression = ?
- How many local minima may the cost function for lin. reg. have (under regular conditions)?
- Name two ways to minimize the cost function?
- General gradient descent formula?
- Linear regression with gradient descent formula?
- What issues can arise during gradient descent?
- What is the design matrix? What are its dimensions?
- Analytical solution for linear regression = ?
  - What are the components of the solution?
- Pros and Cons of gradient descent vs. analytical solution?
- How can one learn non-linear hypotheses with linear regression?
- What is polynomial regression?
- What are radial basis functions?

What is next?

- Classification with Logistic Regression
- Gradient descent tricks & more advanced optimization techniques
- Underfitting & Overfitting
- Model selection (Training, Validation and test set)