3D geoinformation
Department of Urbanism
Faculty of Architecture and the Built Environment
Delft University of Technology

GEO5017
Machine Learning for the Built Environment
https://3d.bk.tudelft.nl/courses/geo5017/

Linear Regression

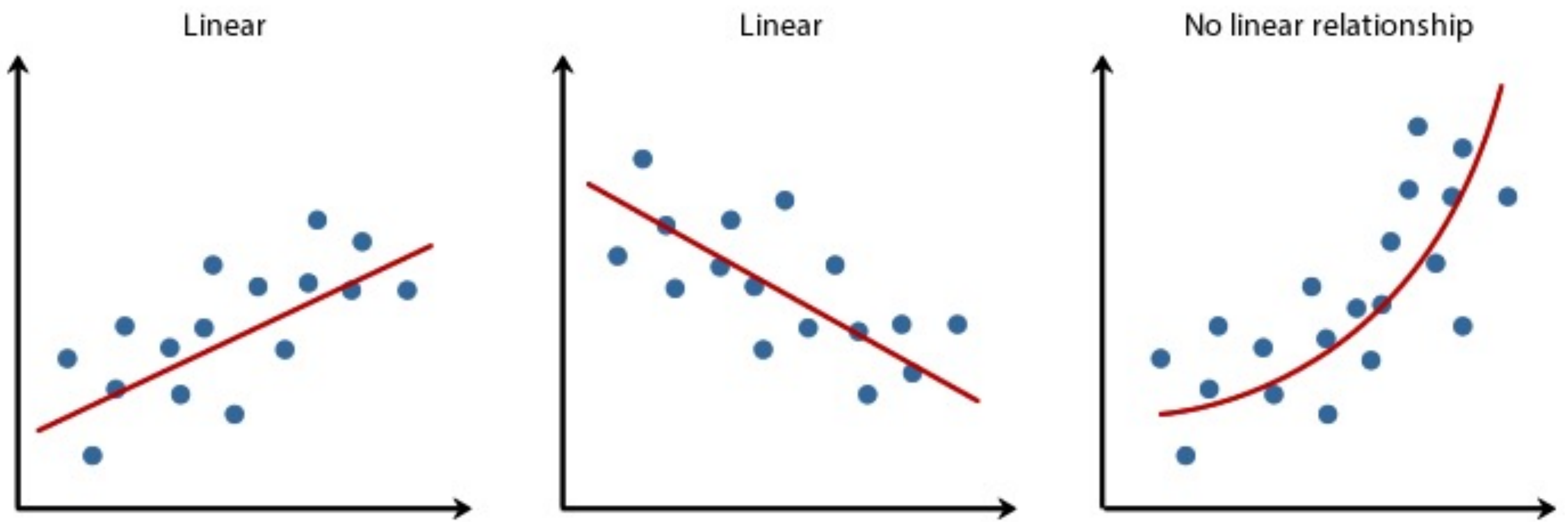
Liangliang Nan
https://3d.bk.tudelft.nl/liangliang/

Agenda

- Linear regression
- The closed-form solution
  - Simple linear regression
  - Polynomial regression
  - Multivariate linear regression
- Solve linear regression by optimization
  - Gradient descent

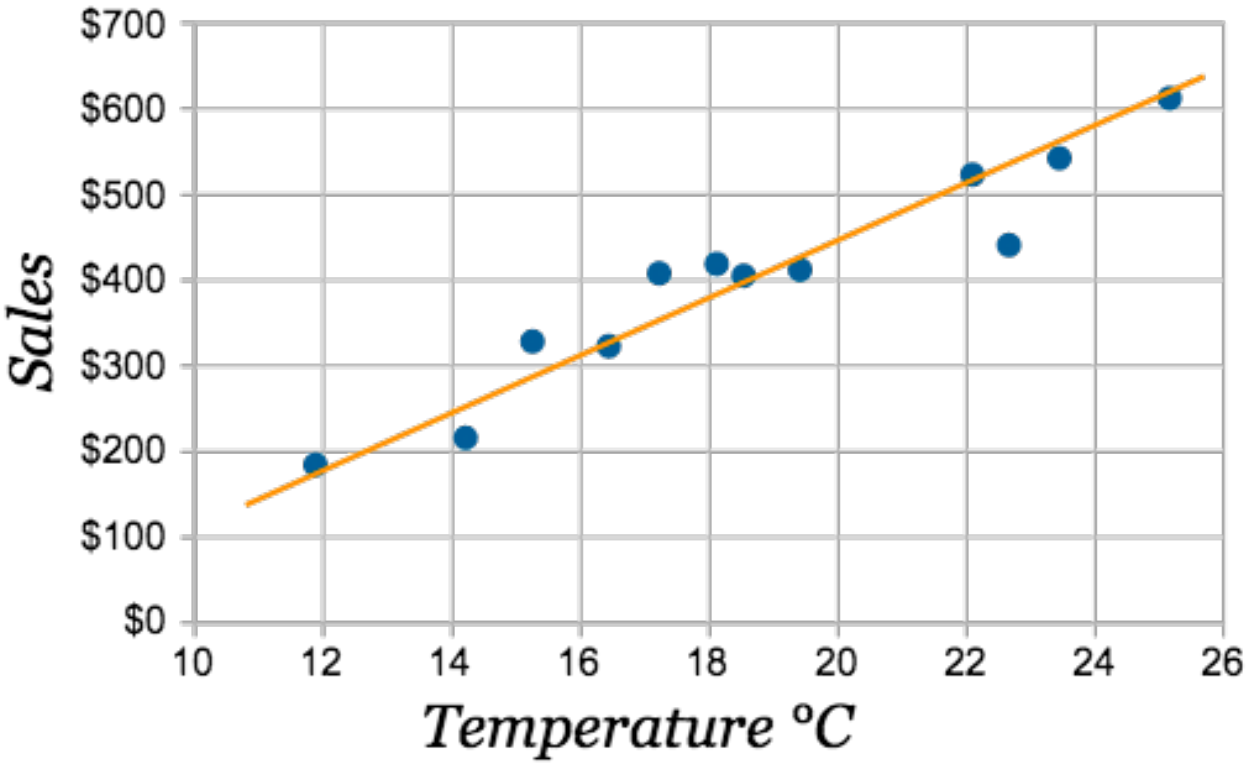
What is linear regression?

- Given a set of observed values of the **independent (input) variables** and the corresponding values of the **dependent (output) variable**, **determine a relation** between the independent variable(s) and a continuous output variable



Linear regression

- Examples



Linear regression

- Examples

Prices of used cars: example data for regression

Price (US$)	Age (years)	Distance (km)	Weight (pounds)
13500	23	46986	1165
13750	23	72937	1165
13950	24	41711	1165
14950	26	48000	1165
13750	30	38500	1170
12950	32	61000	1170
16900	27	94612	1245
18600	30	75889	1245
21500	27	19700	1185
12950	23	71138	1105

Linear regression

- General approach
  - Regression function

$$y = f(x, \theta)$$

  - Objective
    - Optimize θ such that the approximation error is minimized

$$E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Example

$$\text{Price} = a_0 + a_1 \cdot \text{Age} + a_2 \cdot \text{Distance} + a_3 \cdot \text{Weight}$$

$$x = \{\text{Age}, \text{Distance}, \text{Weight}\}$$

$$\theta = \{a_0, a_1, a_2, a_3\}$$

Prices of used cars: example data for regression

Price (US$)	Age (years)	Distance (km)	Weight (pounds)
13500	23	46986	1165
13750	23	72937	1165
13950	24	41711	1165
14950	26	48000	1165
13750	30	38500	1170
12950	32	61000	1170
16900	27	94612	1245
18600	30	75889	1245
21500	27	19700	1185
12950	23	71138	1105

Different linear regression models

- Simple linear regression
  - Only one continuous independent variable

$$y = a + bx$$

- Polynomial regression
  - Only one continuous independent variable

$$y = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

- Multivariate linear regression
  - More than one independent variable

$$y = a_0 + a_1 x_1 + \ldots + a_n x_n$$

Agenda

- Linear regression
- The closed-form solution
    - Simple linear regression
    - Polynomial regression
    - Multivariate linear regression
- Solving linear regression using optimization techniques
    - Gradient descent

Simple linear regression

- Ordinary least squares

$$y = \alpha + \beta x$$

x	x_1	x_2	$\dots$	x_n
y	y_1	y_2	$\dots$	y_n

$$E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$
$$= \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)]^2$$

$$\Rightarrow$$

$$\sum_{i=1}^{n} y_i = n\alpha + \beta \sum_{i=1}^{n} x_i$$
$$\sum_{i=1}^{n} x_i y_i = \alpha \sum_{i=1}^{n} x_i + \beta \sum_{i=1}^{n} x_i^2$$

Simple linear regression

- Ordinary least squares

$$y = \alpha + \beta x$$

$$\beta = \frac{\mathrm{Cov}(x, y)}{\mathrm{Var}(x)}$$

$$\alpha = \bar{y} - \beta\bar{x}$$

$$\mathrm{Var}(x) = \frac{1}{n-1}\sum (x_i - \bar{x}_i)^2$$

$$\mathrm{Cov}(x, y) = \frac{1}{n-1}\sum (x_i - \bar{x})(y_i - \bar{y})$$

$$\bar{x} = \frac{1}{n}\sum x_i$$

$$\bar{y} = \frac{1}{n}\sum y_i$$

Simple linear regression

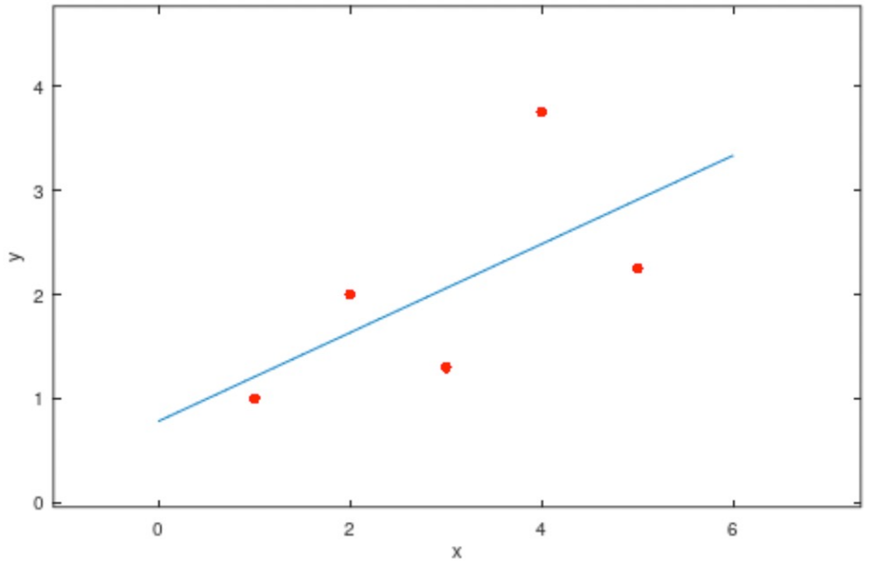
- Example

$$y = \alpha + \beta x$$

$$n = 5$$
$$\bar{x} = \frac{1}{5}(1.0 + 2.0 + 3.0 + 4.0 + 5.0) = 3.0$$
$$\bar{y} = \frac{1}{5}(1.00 + 2.00 + 1.30 + 3.75 + 2.25) = 2.06$$
$$\mathrm{Cov}(x, y) = \frac{1}{4}[(1.0 - 3.0)(1.00 - 2.06) + \cdots + (5.0 - 3.0)(2.25 - 2.06)] = 1.0625$$
$$\mathrm{Var}(x) = \frac{1}{4}\left[(1.0 - 3.0)^2 + \cdots + (5.0 - 3.0)^2\right] = 2.5$$
$$b = \frac{1.0625}{2.5} = 0.425$$
$$a = 2.06 - 0.425 \times 3.0 = 0.785$$

$$y = 0.785 + 0.425x$$

x	1.0	2.0	3.0	4.0	5.0
y	1.00	2.00	1.30	3.75	2.25



Polynomial regression

- Model

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_k x^k$$

x	x_1	x_2	$\dots$	x_n
y	y_1	y_2	$\dots$	y_n

- Ordinary least squares
  - Objective

$$E = \sum_{i=1}^{n} \left[y_i - (\alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \cdots + \alpha_k x_i^k) \right]^2$$

  - Solution can be obtained by solving

$$\frac{\partial E}{\partial \alpha_i} = 0, \quad \forall i = 0, 1, \ldots, k.$$

Polynomial regression

- Ordinary least squares

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_k x^k$$

$$\frac{\partial E}{\partial \alpha_i} = 0, \quad \forall i = 0, 1, \ldots, k.$$

$$\Downarrow$$

$$\begin{aligned}
\sum y_i &= \alpha_0 n + \alpha_1 \left(\sum x_i\right) + \cdots + \alpha_k \left(\sum x_i^k\right) \\
\sum y_i x_i &= \alpha_0 \left(\sum x_i\right) + \alpha_1 \left(\sum x_i^2\right) + \cdots + \alpha_k \left(\sum x_i^{k+1}\right) \\
\sum y_i x_i^2 &= \alpha_0 \left(\sum x_i^2\right) + \alpha_1 \left(\sum x_i^3\right) + \cdots + \alpha_k \left(\sum x_i^{k+2}\right) \\
&\vdots \\
\sum y_i x_i^k &= \alpha_0 \left(\sum x_i^k\right) + \alpha_1 \left(\sum x_i^{k+1}\right) + \cdots + \alpha_k \left(\sum x_i^{2k}\right)
\end{aligned}$$

Polynomial regression

- Ordinary least squares

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_k x^k$$

$$\frac{\partial E}{\partial \alpha_i} = 0, \quad \forall i = 0, 1, \ldots, k.$$

$$\begin{aligned}
\sum y_i &= \alpha_0 n + \alpha_1 \left(\sum x_i\right) + \cdots + \alpha_k \left(\sum x_i^k\right) \\
\sum y_i x_i &= \alpha_0 \left(\sum x_i\right) + \alpha_1 \left(\sum x_i^2\right) + \cdots + \alpha_k \left(\sum x_i^{k+1}\right) \\
\sum y_i x_i^2 &= \alpha_0 \left(\sum x_i^2\right) + \alpha_1 \left(\sum x_i^3\right) + \cdots + \alpha_k \left(\sum x_i^{k+2}\right) \\
&\vdots \\
\sum y_i x_i^k &= \alpha_0 \left(\sum x_i^k\right) + \alpha_1 \left(\sum x_i^{k+1}\right) + \cdots + \alpha_k \left(\sum x_i^{2k}\right)
\end{aligned}$$

$$\vec{y} = D\vec{\alpha}$$

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad D = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^k \\ 1 & x_2 & x_2^2 & \cdots & x_2^k \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^k \end{bmatrix}, \text{ and } \vec{\alpha} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_k \end{bmatrix}$$

Polynomial regression

- Ordinary least squares

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_k x^k$$

$$\frac{\partial E}{\partial \alpha_i} = 0, \quad \forall i = 0, 1, \ldots, k.$$

$$\begin{aligned}
\sum y_i &= \alpha_0 n + \alpha_1 \left(\sum x_i\right) + \cdots + \alpha_k \left(\sum x_i^k\right) \\
\sum y_i x_i &= \alpha_0 \left(\sum x_i\right) + \alpha_1 \left(\sum x_i^2\right) + \cdots + \alpha_k \left(\sum x_i^{k+1}\right) \\
\sum y_i x_i^2 &= \alpha_0 \left(\sum x_i^2\right) + \alpha_1 \left(\sum x_i^3\right) + \cdots + \alpha_k \left(\sum x_i^{k+2}\right) \\
&\vdots \\
\sum y_i x_i^k &= \alpha_0 \left(\sum x_i^k\right) + \alpha_1 \left(\sum x_i^{k+1}\right) + \cdots + \alpha_k \left(\sum x_i^{2k}\right)
\end{aligned}$$

$$\vec{y} = D\vec{\alpha}$$

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad D = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^k \\ 1 & x_2 & x_2^2 & \cdots & x_2^k \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^k \end{bmatrix}, \text{ and } \vec{\alpha} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_k \end{bmatrix}$$

$$\vec{\alpha} = \left(D^T D\right)^{-1} D^T \vec{y}$$

Polynomial regression

- Example

x	3.0	4.0	5.0	6.0	7.0
y	2.5	3.2	3.8	6.5	11.5

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2$$

$$\sum y_i = n\alpha_0 + \alpha_1 \left(\sum x_i\right) + \alpha_2 \left(\sum x_i^2\right)$$
$$\sum y_i x_i = \alpha_0 \left(\sum x_i\right) + \alpha_1 \left(\sum x_i^2\right) + \alpha_2 \left(\sum x_i^3\right)$$
$$\sum y_i x_i^2 = \alpha_0 \left(\sum x_i^2\right) + \alpha_1 \left(\sum x_i^3\right) + \alpha_2 \left(\sum x_i^4\right)$$

$$27.5 = 5\alpha_0 + 25\alpha_1 + 135\alpha_2$$
$$158.8 = 25\alpha_0 + 135\alpha_1 + 775\alpha_2$$
$$966.2 = 135\alpha_0 + 775\alpha_1 + 4659\alpha_2$$

$$\alpha_0 = 12.4285714$$
$$\alpha_1 = -5.5128571$$
$$\alpha_2 = 0.7642857$$

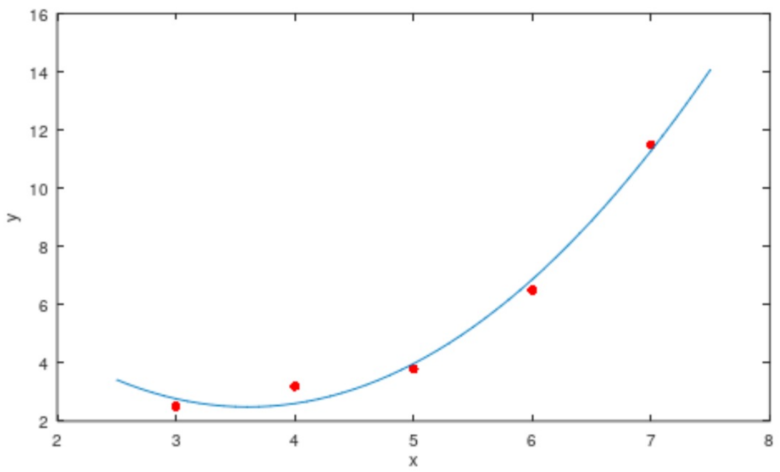
$$y = 12.4285714 - 5.5128571x + 0.7642857x^2$$

Polynomial regression

- Example

x	3.0	4.0	5.0	6.0	7.0
y	2.5	3.2	3.8	6.5	11.5

$$y = \alpha_0 + \alpha_1 x + \alpha_2 x^2$$



$$\sum y_i = n\alpha_0 + \alpha_1 \left(\sum x_i\right) + \alpha_2 \left(\sum x_i^2\right)$$
$$\sum y_i x_i = \alpha_0 \left(\sum x_i\right) + \alpha_1 \left(\sum x_i^2\right) + \alpha_2 \left(\sum x_i^3\right)$$
$$\sum y_i x_i^2 = \alpha_0 \left(\sum x_i^2\right) + \alpha_1 \left(\sum x_i^3\right) + \alpha_2 \left(\sum x_i^4\right)$$

$$27.5 = 5\alpha_0 + 25\alpha_1 + 135\alpha_2$$
$$158.8 = 25\alpha_0 + 135\alpha_1 + 775\alpha_2$$
$$966.2 = 135\alpha_0 + 775\alpha_1 + 4659\alpha_2$$

$$\alpha_0 = 12.4285714$$
$$\alpha_1 = -5.5128571$$
$$\alpha_2 = 0.7642857$$

$$y = 12.4285714 - 5.5128571x + 0.7642857x^2$$

Multivariate linear regression

- Model

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_N x_N$$

- Ordinary least squares

Variables	Values (examples)			
	Example 1	Example 2	...	Example n
x_1	x_{11}	x_{12}	...	x_{1n}
x_1	x_{21}	x_{22}	...	x_{2n}
...				
x_N	x_{N1}	x_{N2}	...	x_{Nn}
y (outcomes)	y_1	y_2	...	y_n

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad X = \begin{bmatrix} 1 & x_{11} & x_{21} & \cdots & x_{N1} \\ 1 & x_{12} & x_{22} & \cdots & x_{N2} \\ \vdots & & & & \\ 1 & x_{1n} & x_{2n} & \cdots & x_{Nn} \end{bmatrix}, \text{ and } B = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_N \end{bmatrix}$$

$$B = \left(X^T X\right)^{-1} X^T Y$$

Multivariate linear regression

- Example

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

x_1	1	1	2	0
x_2	1	2	2	1
y	3.25	6.5	3.5	5.0

Multivariate linear regression

- Example

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

x_1	1	1	2	0
x_2	1	2	2	1
y	3.25	6.5	3.5	5.0

$$\boxed{B = \left(X^T X\right)^{-1} X^T Y}$$

$$Y = \begin{bmatrix} 3.25 \\ 6.5 \\ 3.5 \\ 5.0 \end{bmatrix}, \quad X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \\ 1 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

$$y = 2.0625 - 2.3750 x_1 + 3.2500 x_2$$

$$X^T X = \begin{bmatrix} 4 & 4 & 6 \\ 4 & 6 & 7 \\ 6 & 7 & 10 \end{bmatrix} \Rightarrow \left(X^T X\right)^{-1} = \begin{bmatrix} \frac{11}{4} & \frac{1}{2} & -2 \\ \frac{1}{2} & 1 & -1 \\ -2 & -1 & 2 \end{bmatrix} \Rightarrow B = \left(X^T X\right)^{-1} X^T Y = \begin{bmatrix} 2.0625 \\ -2.3750 \\ 3.2500 \end{bmatrix}$$

Agenda

- Linear regression
- The closed-form solution
  - Simple linear regression
  - Polynomial regression
  - Multivariate linear regression
- Solve linear regression by optimization
  - Gradient descent

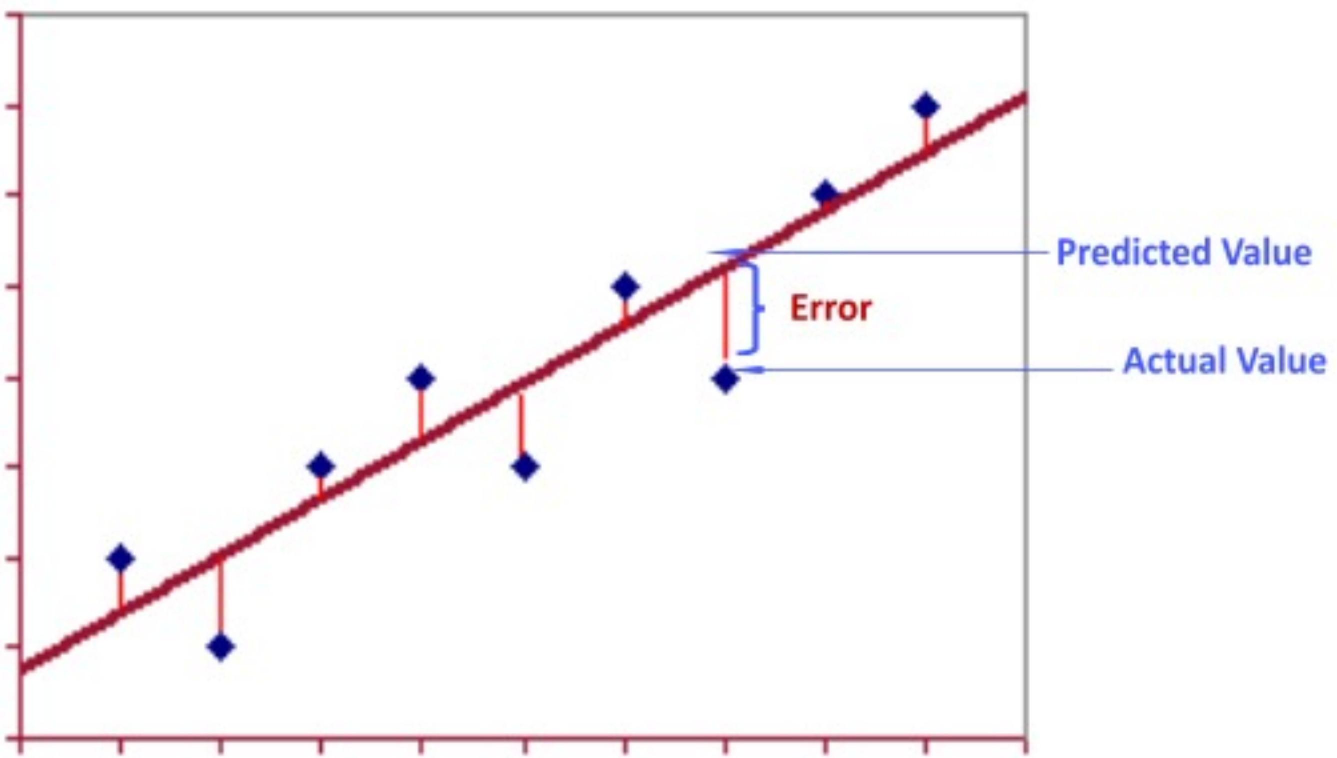
Solve linear regression by optimization

- Linear regression

$$y = f(x, \theta)$$

- Objective function
  - Sum of squared error

$$min \sum_{i=0}^{n} (y_i - \widehat{y}_i)^2$$



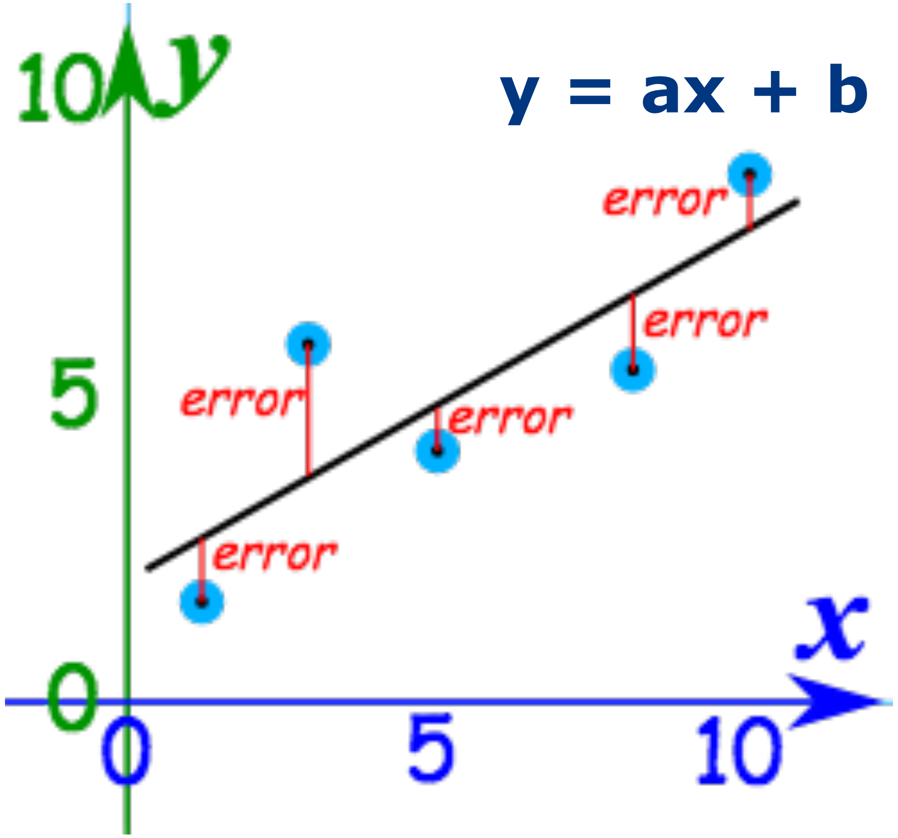
Solve linear regression by optimization

- Example
  - Objective function
    - Sum of squared error

$$E = \sum_{i=0}^{n} (y_i - (ax_i + b))^2$$

    - Solution

$$min \sum_{i=0}^{n} (y_i - (ax_i + b))^2$$



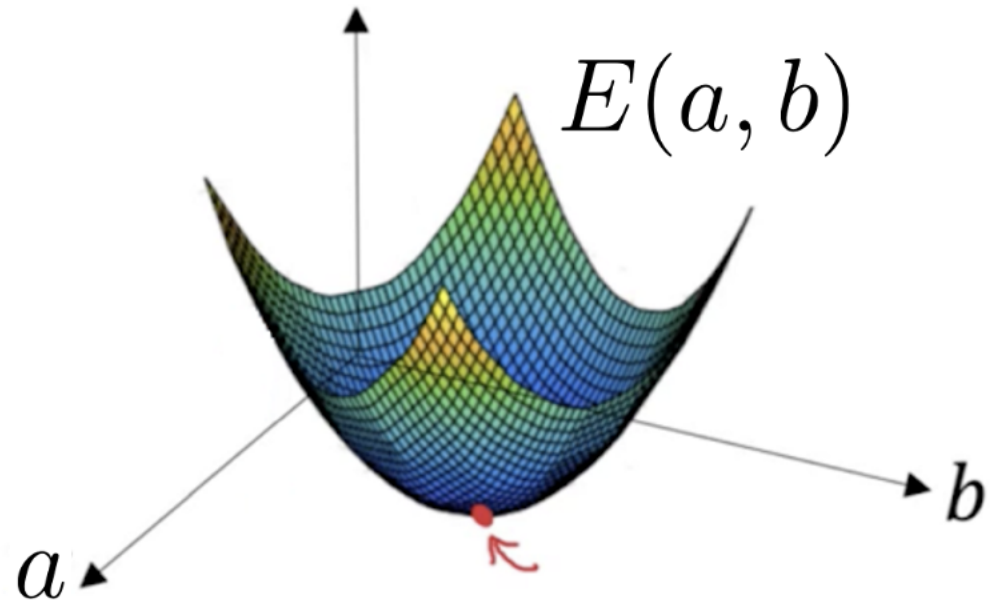
Solve linear regression by optimization

- Example
  - Objective function
    - Sum of squared error

$$E = \sum_{i=0}^{n} (y_i - (ax_i + b))^2$$

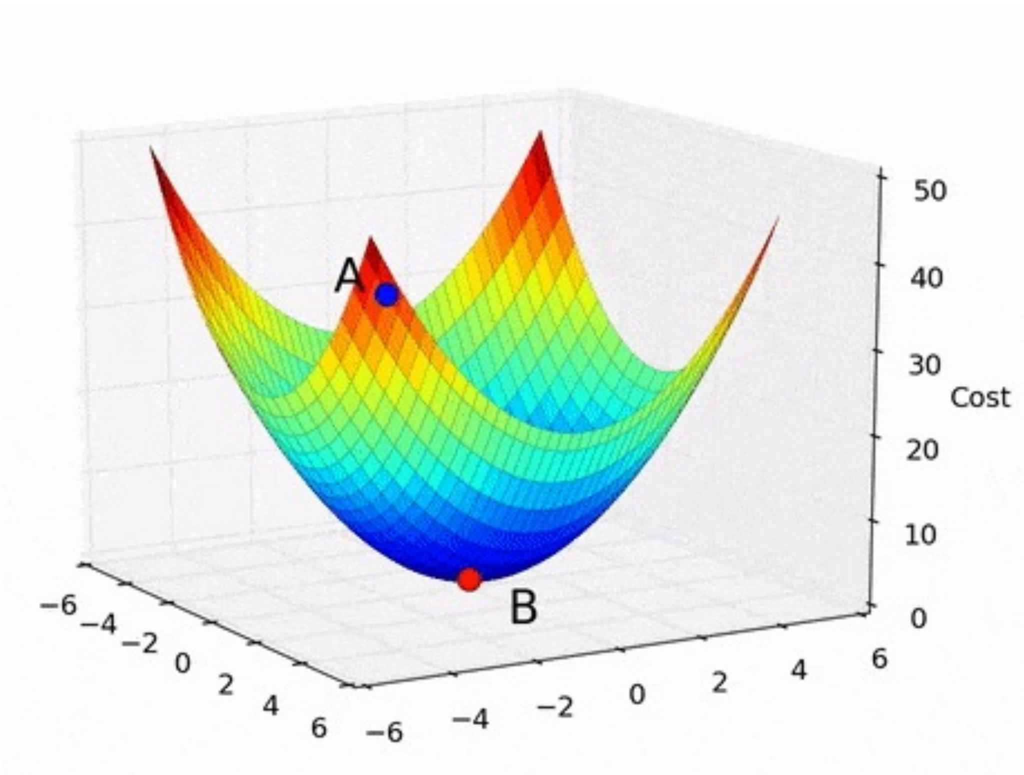
    - Solution

$$min \sum_{i=0}^{n} (y_i - (ax_i + b))^2$$



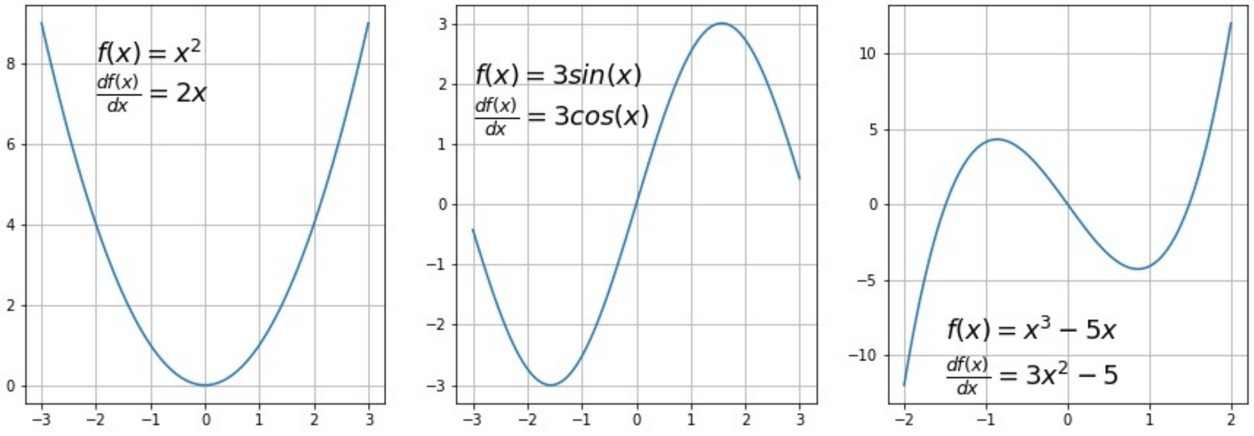
Gradient descent

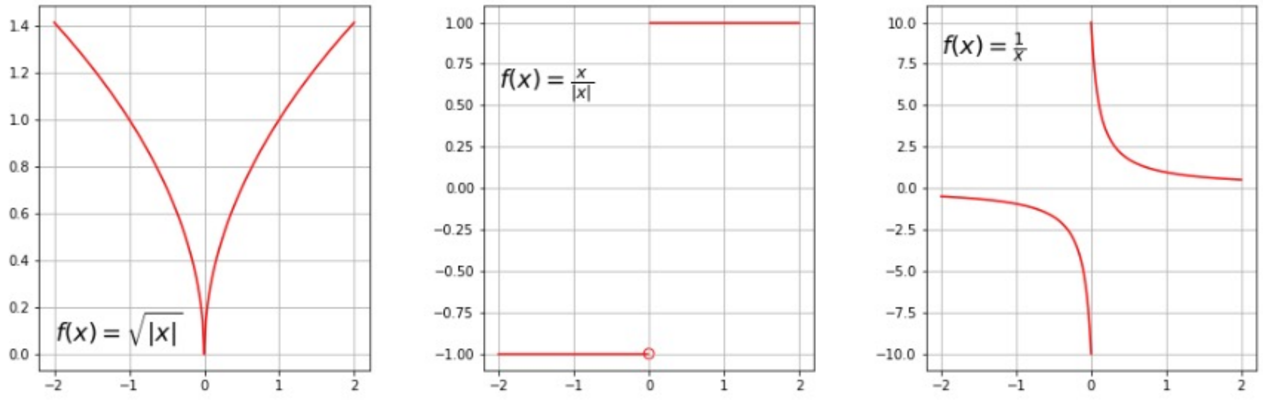
- Basic idea
  - Take repeated steps in steepest descent direction until the lowest point is reached



Gradient descent

- Function requirements
  - Differentiable



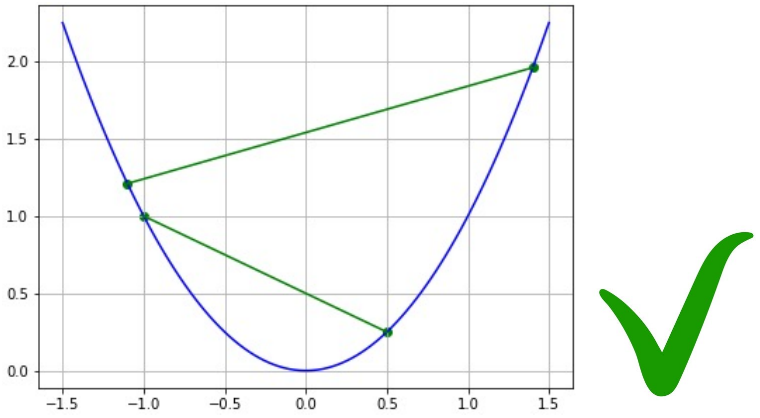


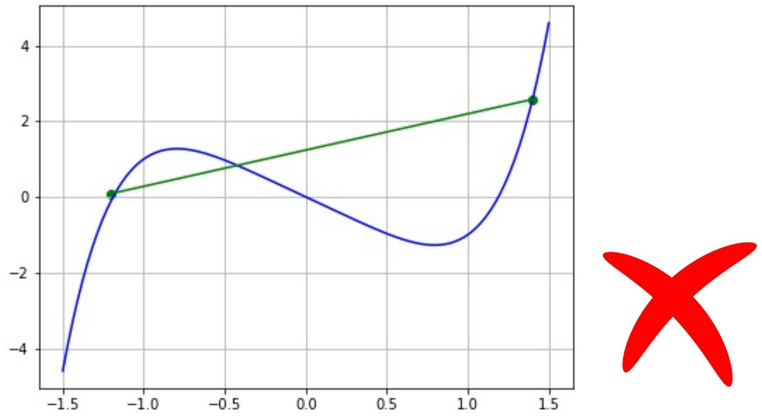
(a) Cusp (b) Jump discontinuity (c) Infinite discontinuity

Gradient descent

- Function requirements
  - Differentiable
  - Convex

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2)$$





Gradient descent

- Function requirements
  - Differentiable
  - Convex

Example $f(x) = x^2 - x + 3$

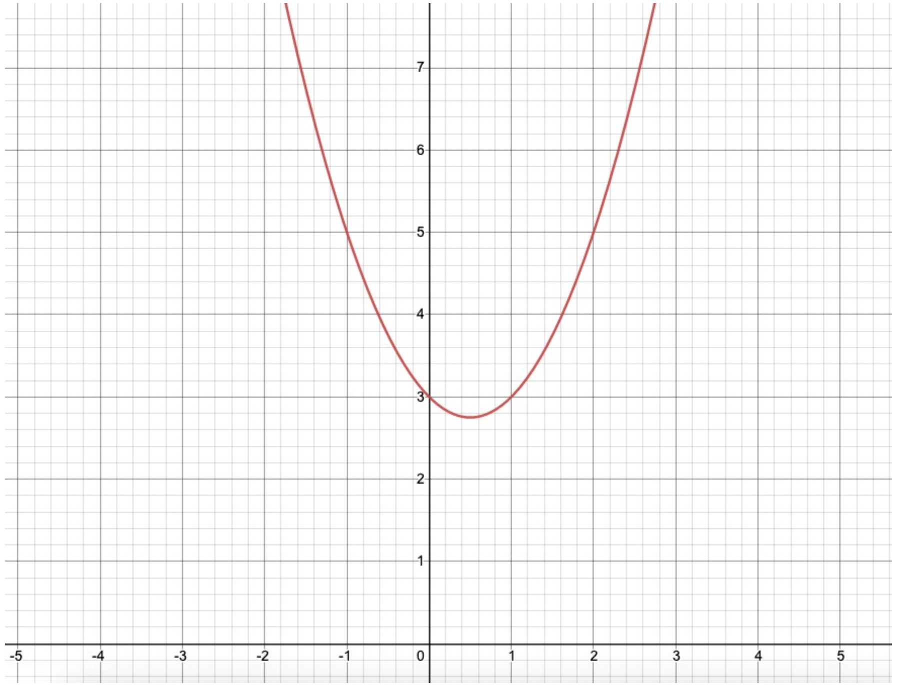
Gradient descent

- Function requirements
  - Differentiable
  - Convex

Example $f(x) = x^2 - x + 3$

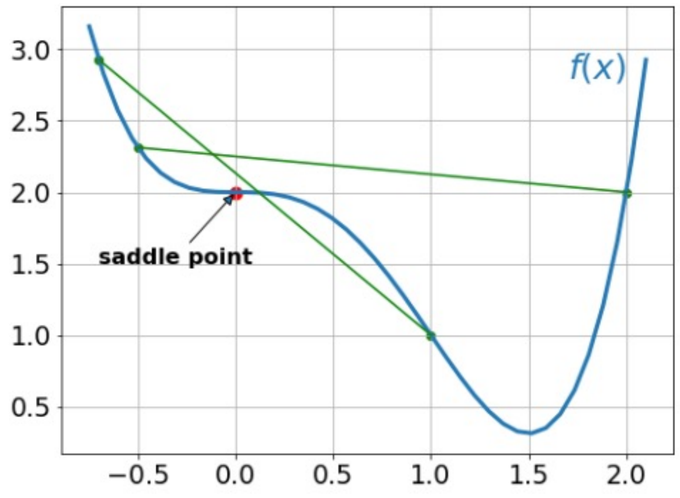
$$\frac{df(x)}{dx} = 2x - 1, \quad \frac{d^2 f(x)}{dx^2} = 2$$

The function has derivative everywhere
The second derivative is always > 0



Gradient descent

- Function requirements
  - Differentiable
  - Convex



Example of a semi-convex function with a saddle point

Example $f(x) = x^4 - 2x^3 + 2$

$$\frac{df(x)}{dx} = 4x^3 - 6x^2 = x^2(4x - 6)$$

$$\frac{d^2 f(x)}{dx^2} = 12x^2 - 12x = 12x(x - 1)$$

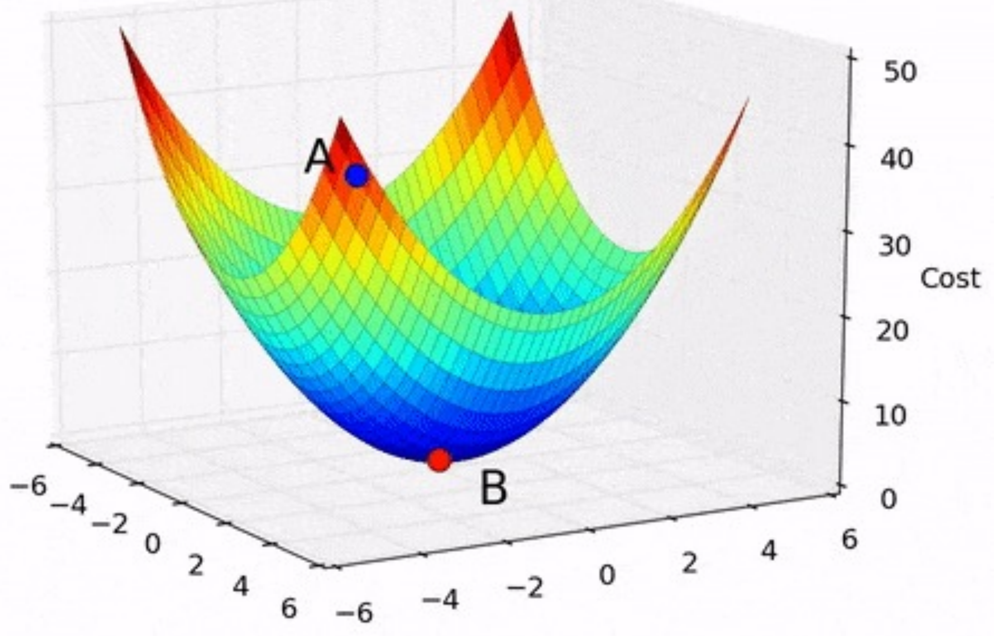
- for $x < 0$: function is convex
- for $0 < x < 1$: function is concave
- for $x > 1$: function is convex again

$x = 0$: saddle point
both first and second derivatives equal to zero

Gradient descent

- Basic idea
  - Take repeated steps in steepest descent direction until the lowest point is reached
    - The opposite direction of the **gradient** (or approximate gradient) of the function at the current point

$$\nabla f(\vec{p}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{p}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{p}) \end{bmatrix}$$



Gradient

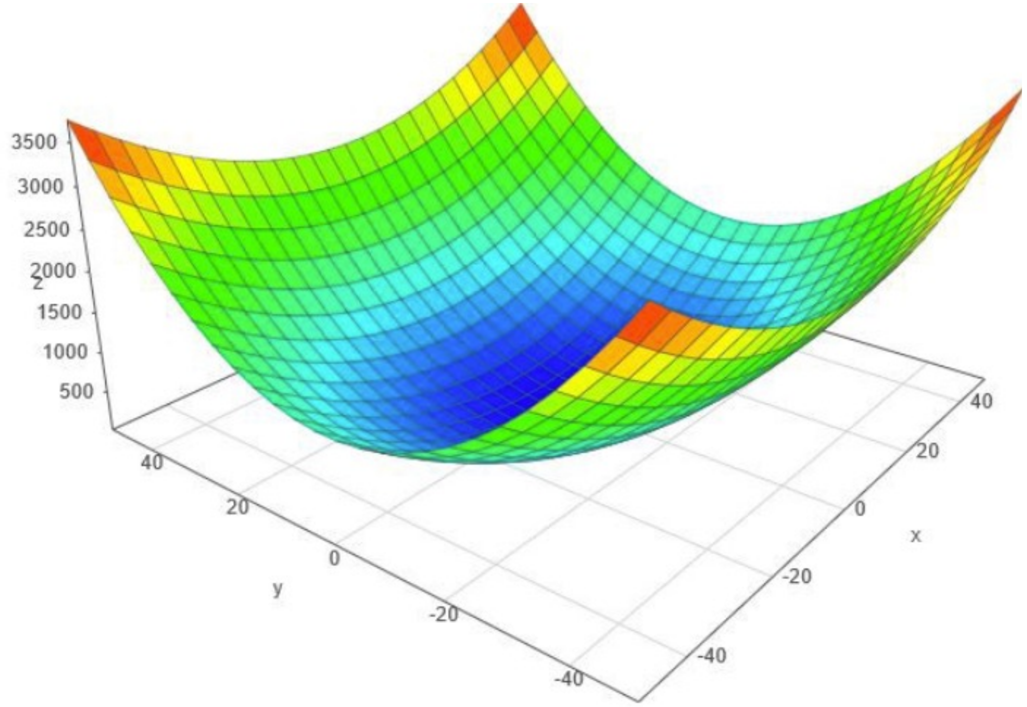
- Example

$$f(x, y) = 0.5x^2 + y^2$$

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x}(x, y) \\ \\ \frac{\partial f}{\partial y}(x, y) \end{bmatrix} = \begin{bmatrix} x \\ \\ 2y \end{bmatrix}$$

The gradient at point *p*(10, 10)

$$\nabla f(10, 10) = \begin{bmatrix} 10 \\ 20 \end{bmatrix}$$

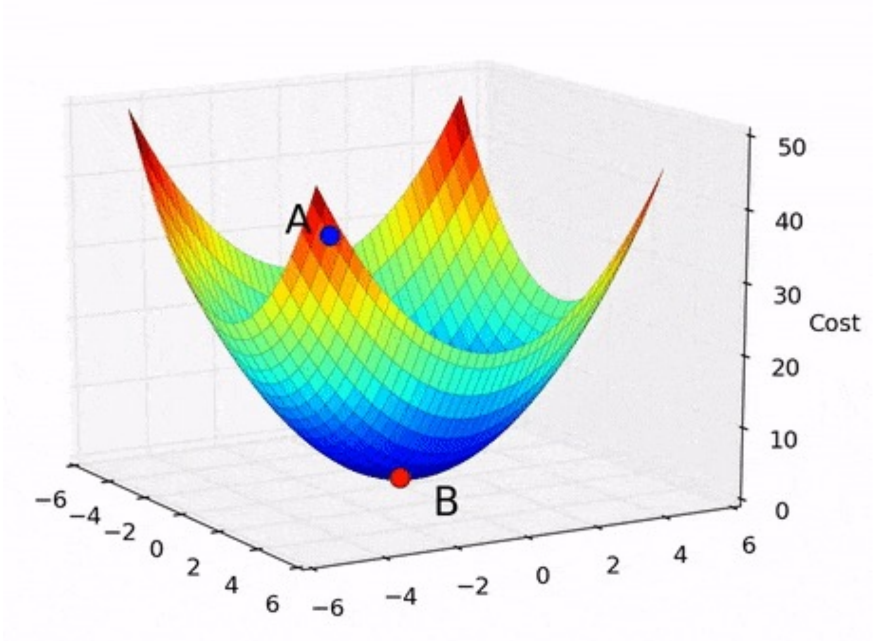


Gradient descent algorithm

- Main steps
  1) Start from an initial guess (or even randomly)
  2) Calculate the the gradient of the function at current point
  3) Make a scaled step in the opposite direction
     to the gradient

$$\vec{p}_{n+1} = \vec{p}_n - \eta \nabla f\left(\vec{p}_n\right)$$

  1) Repeat 2) and 3) until one of the criteria is met
     -) maximum number of iterations reached
     -) step size (or the change of the function value)
        is smaller than a given tolerance



Gradient descent algorithm

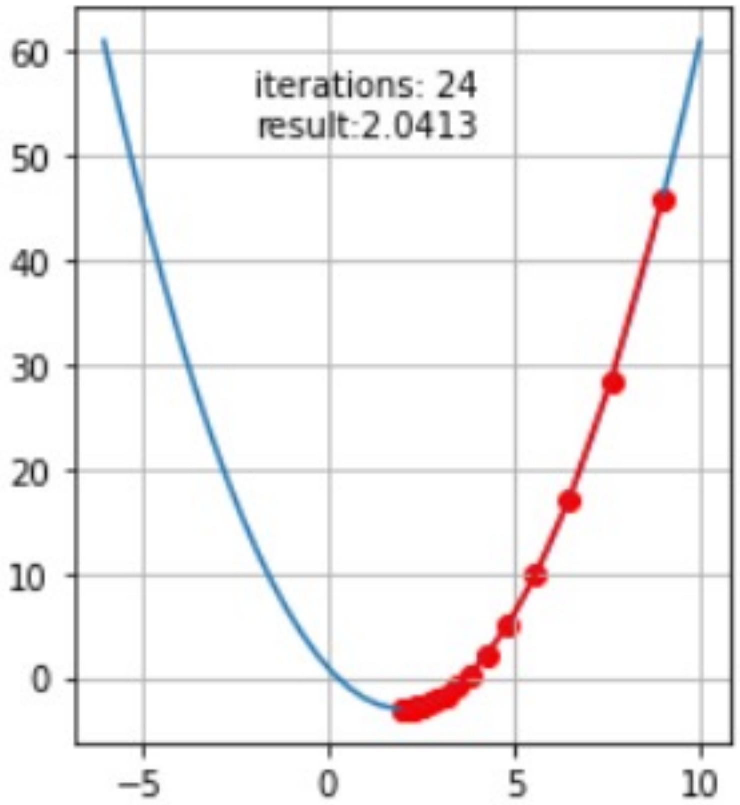
- Example: a 1D function

$$f(x) = x^2 - 4x + 1$$

$$\frac{df(x)}{dx} = 2x - 4$$

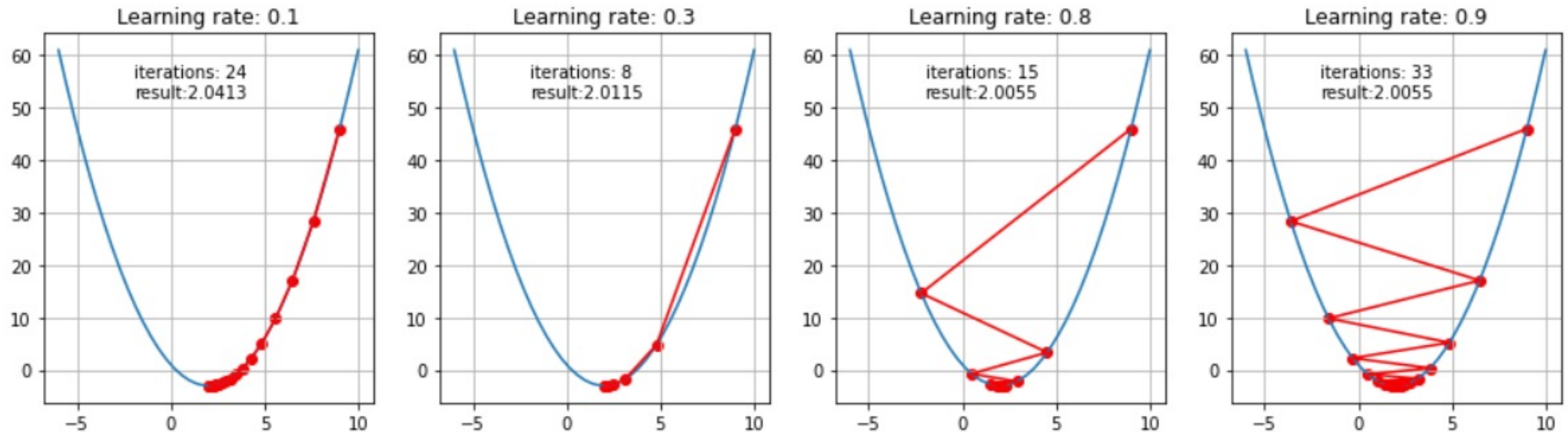
The first few steps

$$x_0 = 9, \qquad f(9) = 46$$
$$x_1 = 9 - 0.1 \times (2 \times 9 - 4) = 7.6, \qquad f(7.6) = 28.36$$
$$x_2 = 7.6 - 0.1 \times (2 \times 7.6 - 4) = 6.48, \qquad f(6.48) = 17.07$$
$$x_3 = 6.48 - 0.1 \times (2 \times 6.48 - 4) = 5.584, \qquad f(5.584) = 9.845$$
$$\ldots$$
$$x_{21} = 2.065, \quad f(2.065) = -2.996$$
$$x_{22} = 2.052, \quad f(2.052) = -2.997$$



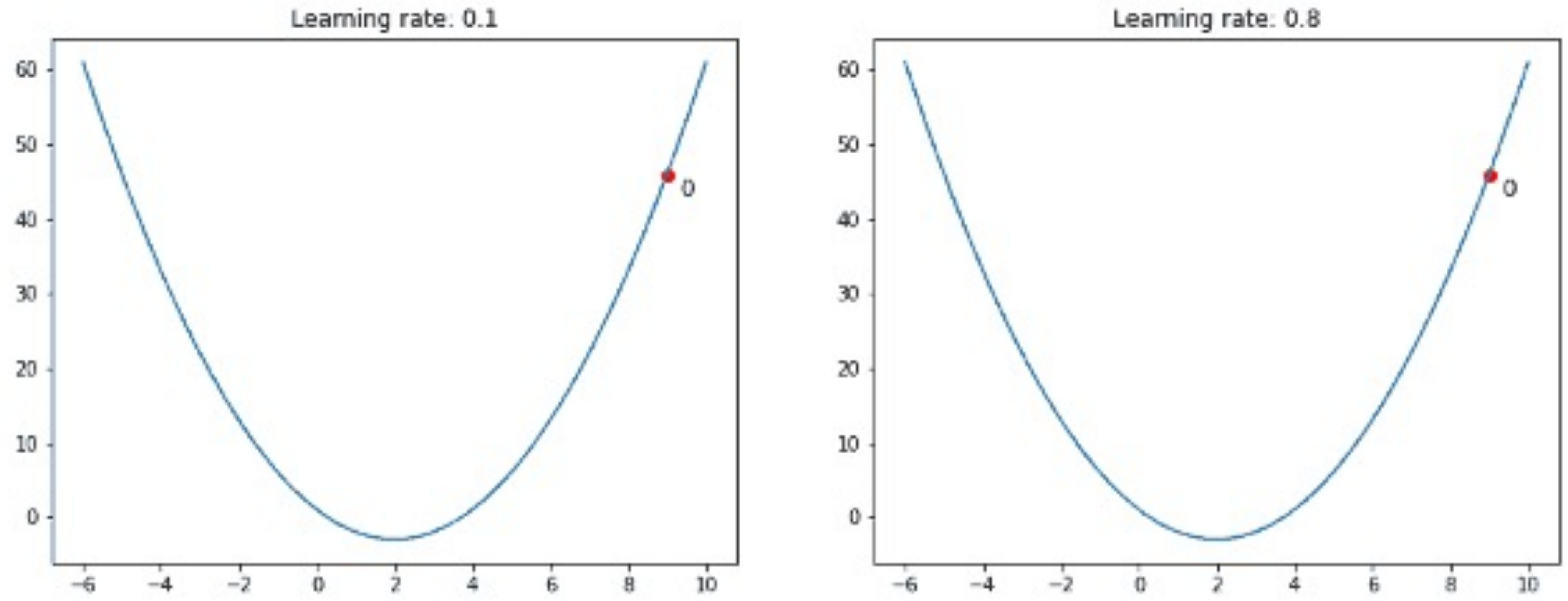
Gradient descent algorithm

- Parameter update $$\vec{p}_{n+1} = \vec{p}_n - \eta \nabla f\left(\vec{p}_n\right)$$
- Learning rate ***η***: scales the gradient and thus controls the step size
  - Too small
    - Too slow to converge; may reach maximum iteration before convergence
  - Too big
    - May not converge to the optimal point (jump around) or even to diverge completely



Gradient descent algorithm

- Learning rate

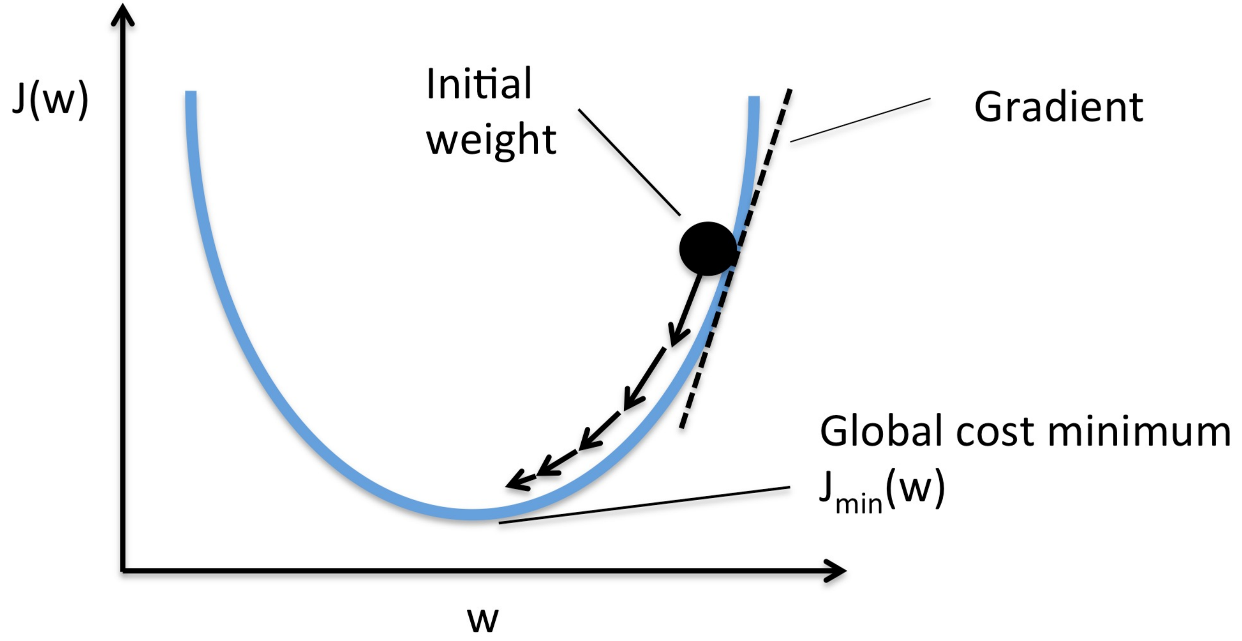


Gradient descent algorithm

- Use a fixed learning rat $\vec{p}_{n+1} = \vec{p}_n - \eta \nabla f\left(\vec{p}_n\right)$
  - Try with a large value like 0.1
  - Try exponentially lower values: 0.01, 0.001, etc.

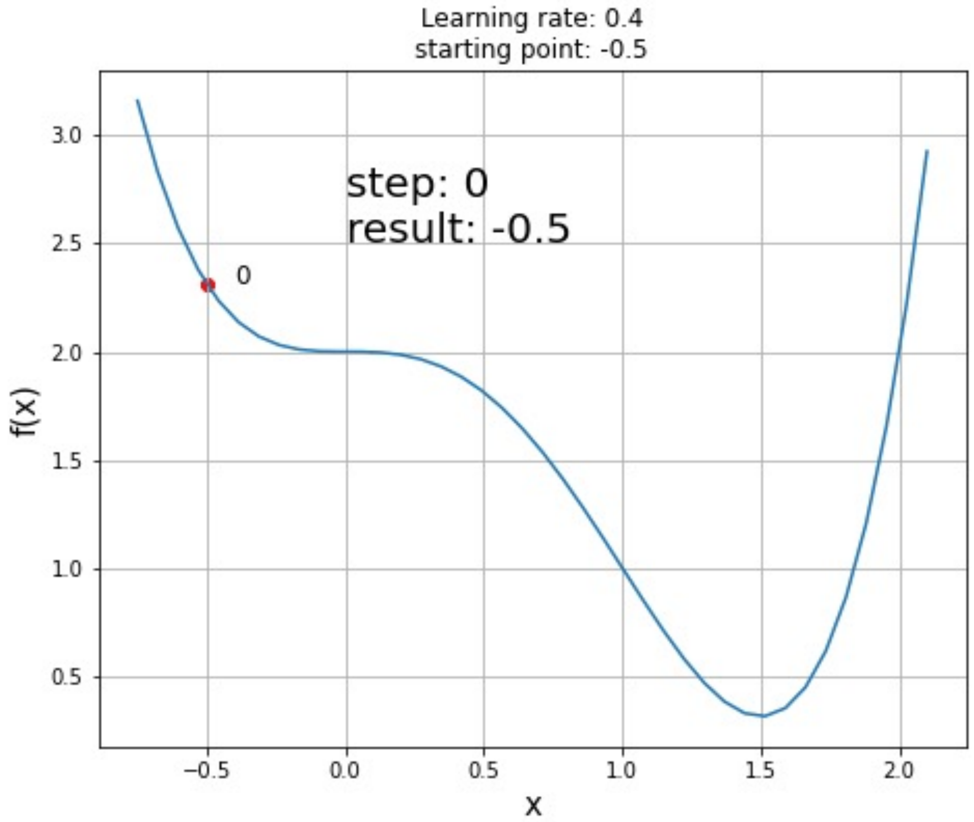
Gradient descent algorithm

- Use a fixed learning rat $\vec{p}_{n+1} = \vec{p}_n - \eta \nabla f(\vec{p}_n)$
  - Try with a large value like 0.1
  - Try exponentially lower values: 0.01, 0.001, etc.
- Use an adaptive learning rate
  - Start with a larger value
  - Gradual decrease it



Gradient descent algorithm

- Challenges
  - Learning rate
  - Saddle points



Advanced methods

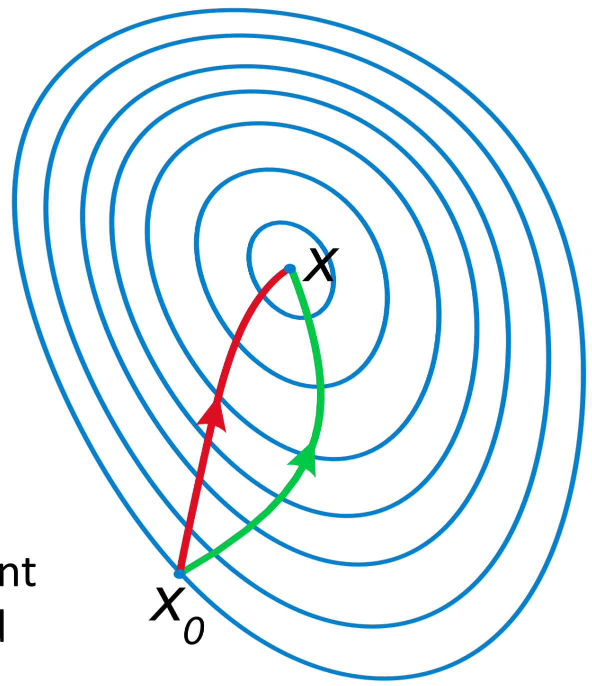
- Newton's method
  - Second-order derivative is used
  - Take a more direct route

Gradient descent

$$f(x_k + t) \approx f(x_k) + f'(x_k)t$$

Newton's method

$$f(x_k + t) \approx f(x_k) + f'(x_k)t + \frac{1}{2}f''(x_k)t^2$$

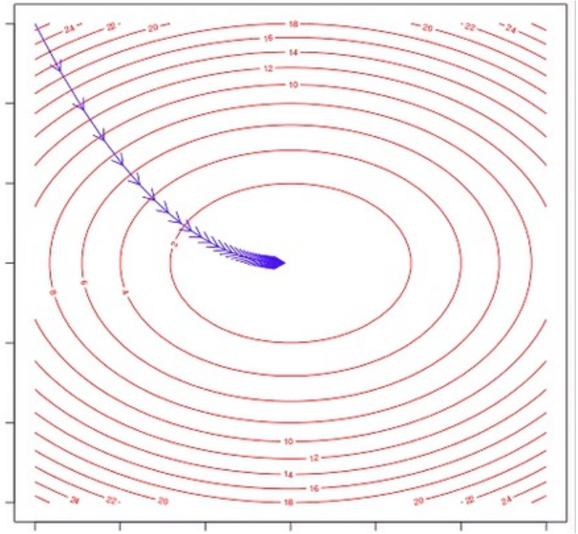


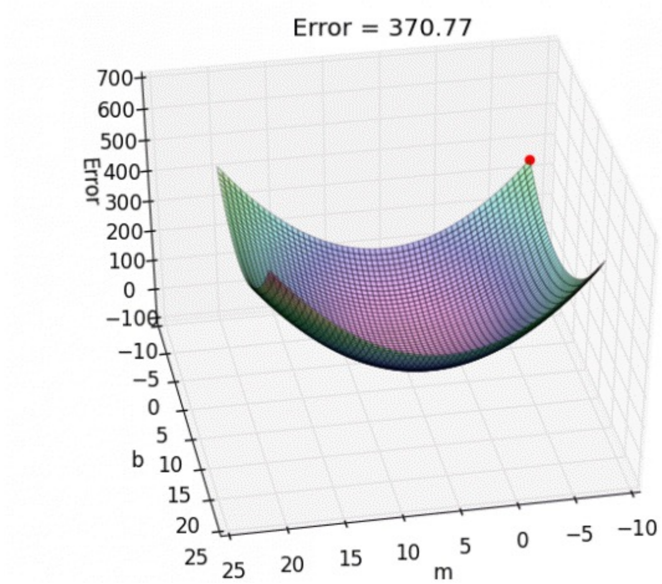
Green: Gradient descent
Red: Newton's method

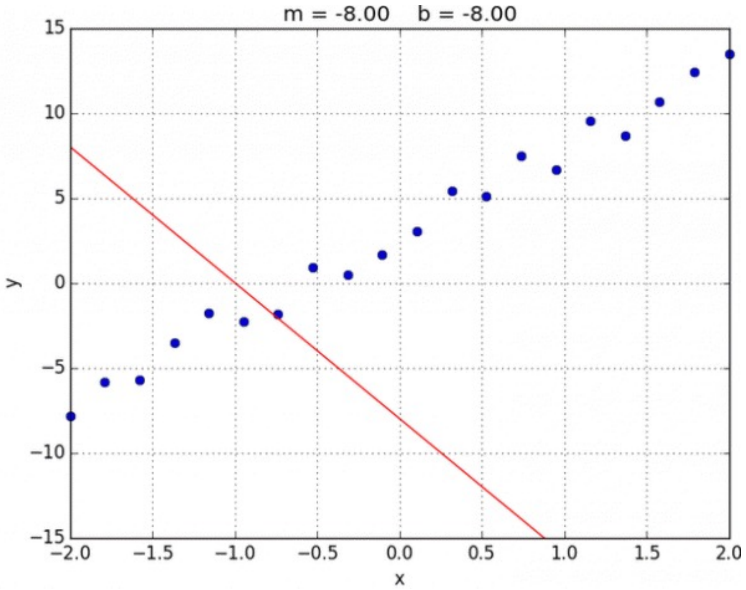
Solve linear regression using GD

- Objective function
  - Always convex

$$f(a,b) = \sum_{i=0}^{n} (y_i - (ax_i + b))^2$$

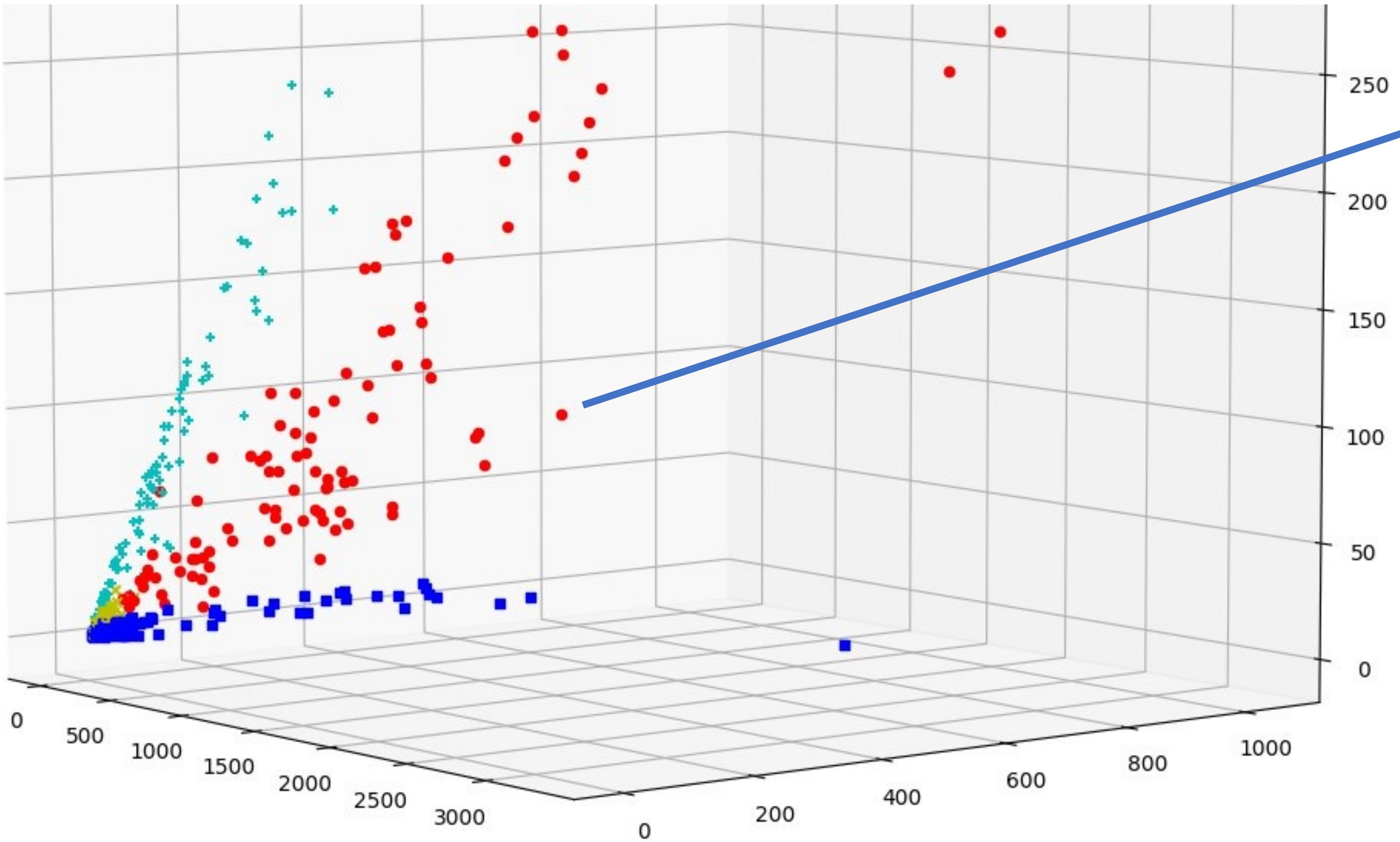






About A1 - Clustering

- The points/input to the clustering algorithm



$$\begin{bmatrix} height \\ volume \\ aspect\ ratio \\ \vdots \end{bmatrix}$$

About A1 - Clustering

- The points/input to the clustering algorithm
  - Each point is an *N*-dimensional vector denoting the features/attributes of an object (an “object” is a point cloud). For example, *p* = {height, volume, … }
  - We have 500 point clouds, thus the input to the clustering algorithm are 500 points
- Goal
  - Put the 500 objects into different groups, such that the same type of objects are in the same group. The result will not be perfect.
- Evaluation
  - After clustering, we manually assign each group a label, so
    - To better understand/compare the performance of the clustering algorithms
    - To be able to compare the performance with supervised techniques (in Q2 and Q3)

What’s next?

- Lab: Gradient descent
  - Application of gradient descent in geometry processing
  - Python code of gradient descent
    https://3d.bk.tudelft.nl/courses/geo5017/code/gradient_descent.py

- Next lecture: Bayesian classification & logistic regression