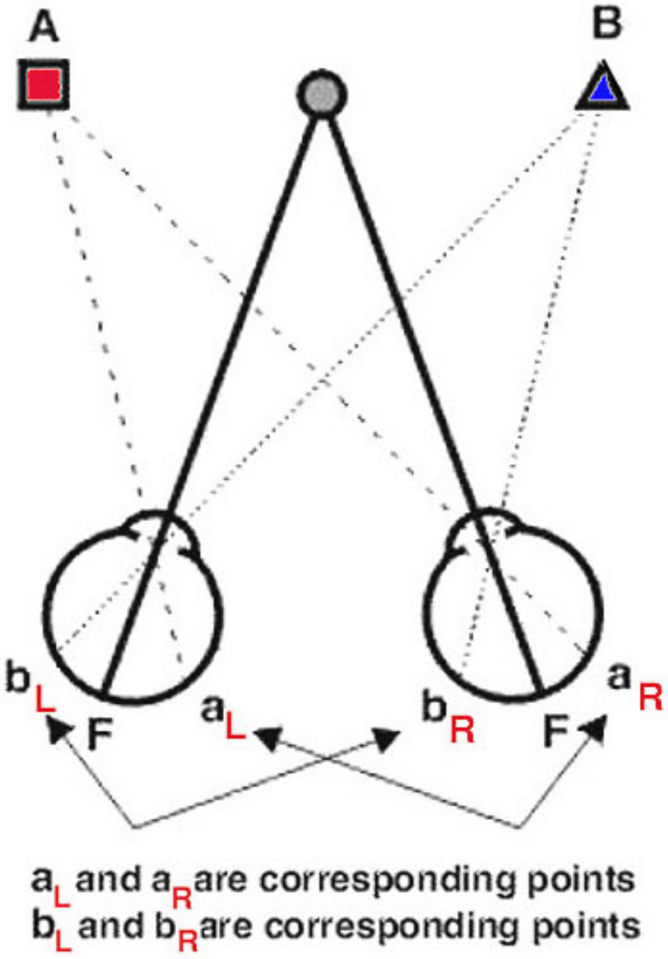
Recap

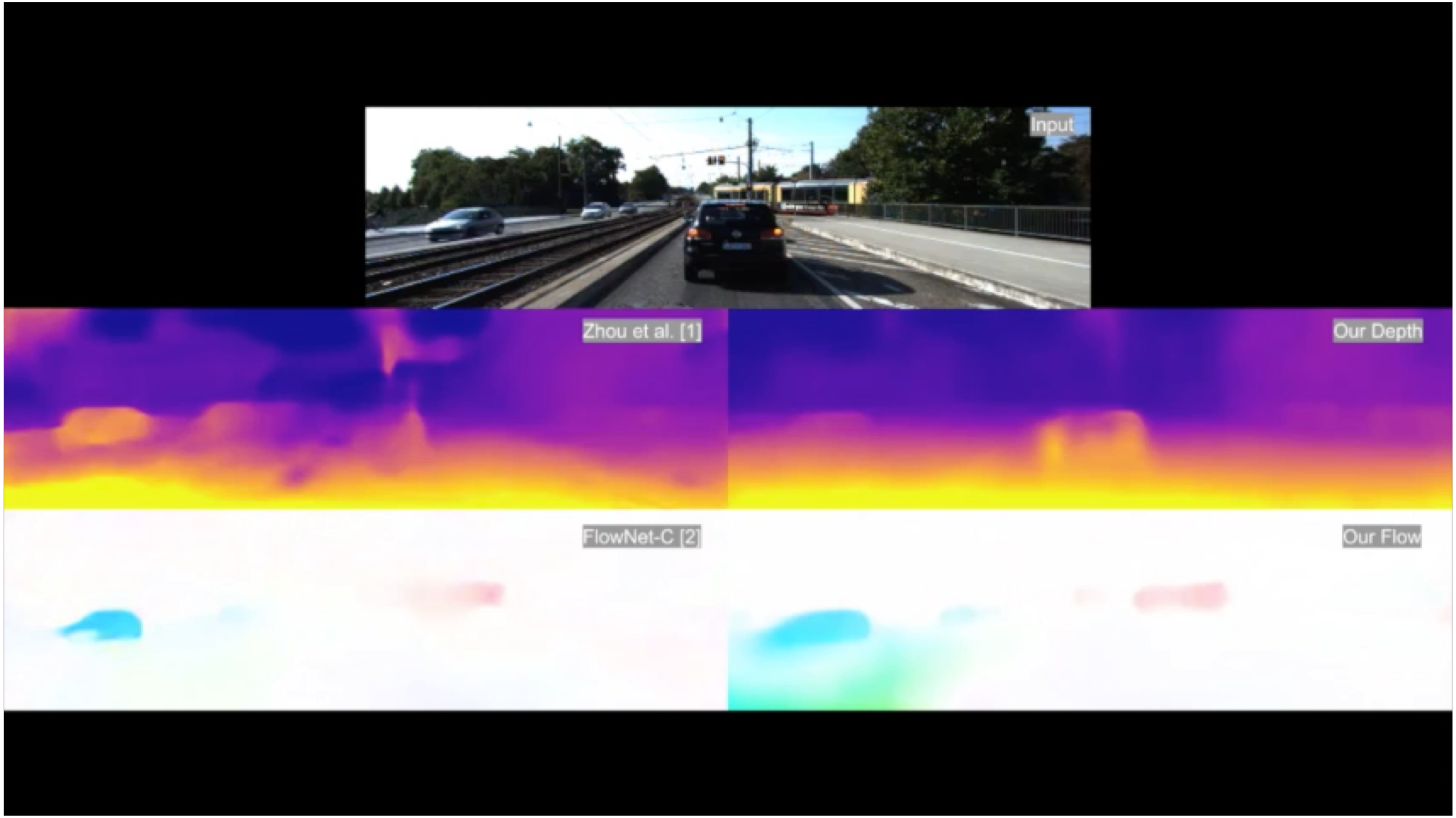


Source: Space Perception, Michael Kalloniatis and Charles Luu

3D Visual Cues

Willem van de Velde the Elder's The Capture of the Royal Prince during the Four Days' Battle (1666)

Find 4 different depth/shape cues

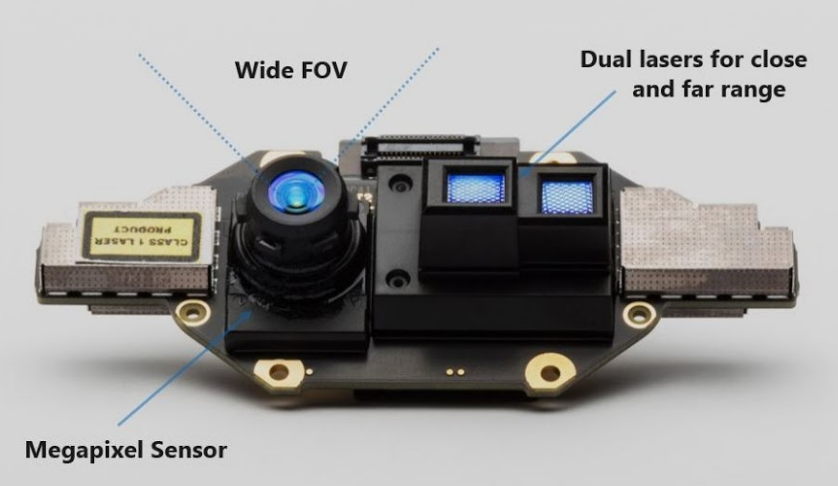


DF-Net: Unsupervised Joint Learning of Depth and Flow using Cross-Task Consistency
Yuliang Zou, Zelun Luo, Jia-Bin Huang

3D Vision

- ❑ Depth sensors

KINECT



IR sensors

Singular Value Decomposition

- ❑ Relation with Rank?

Lets go to the board!

Optimization

Short Introduction

20 December, 2018

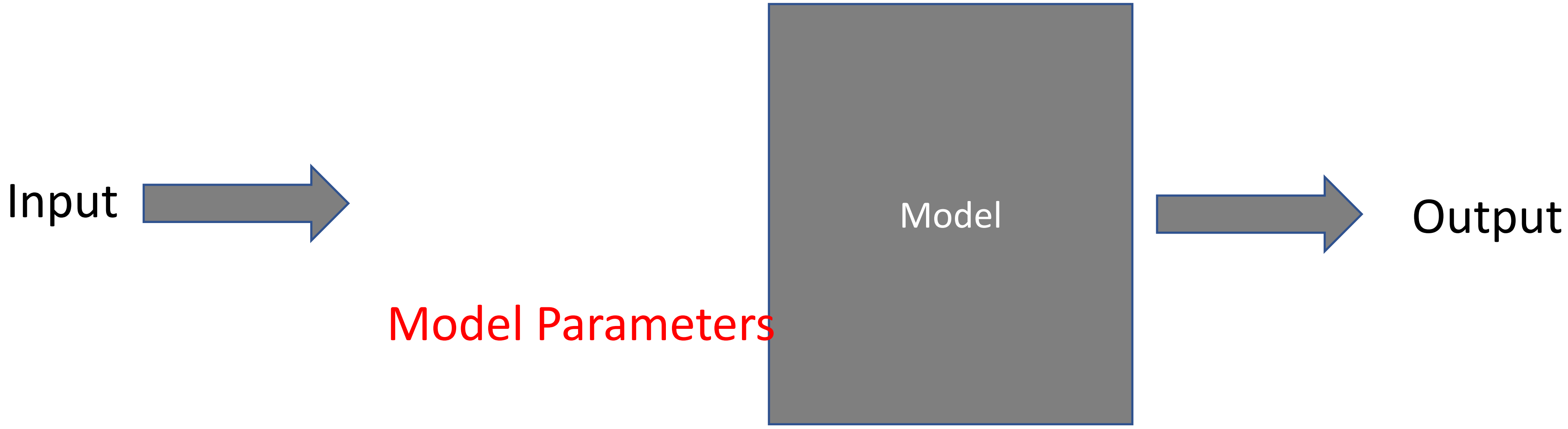
Ajad Chhatkuli[1]

1 ETH Zürich, Switzerland; NAAMII, Nepal

2 King's/Imperial College London, UK; NAAMII, Nepal

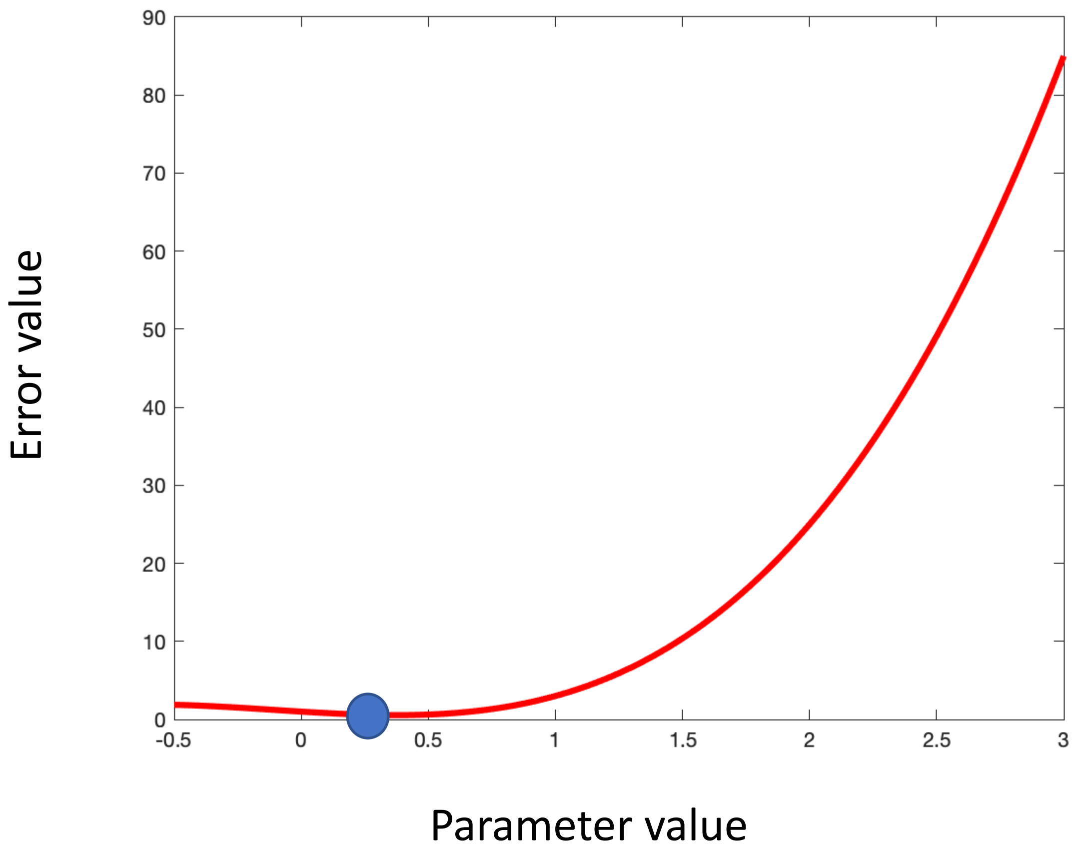
What do we optimize?

- We want to optimize **for** our model by minimizing **error**



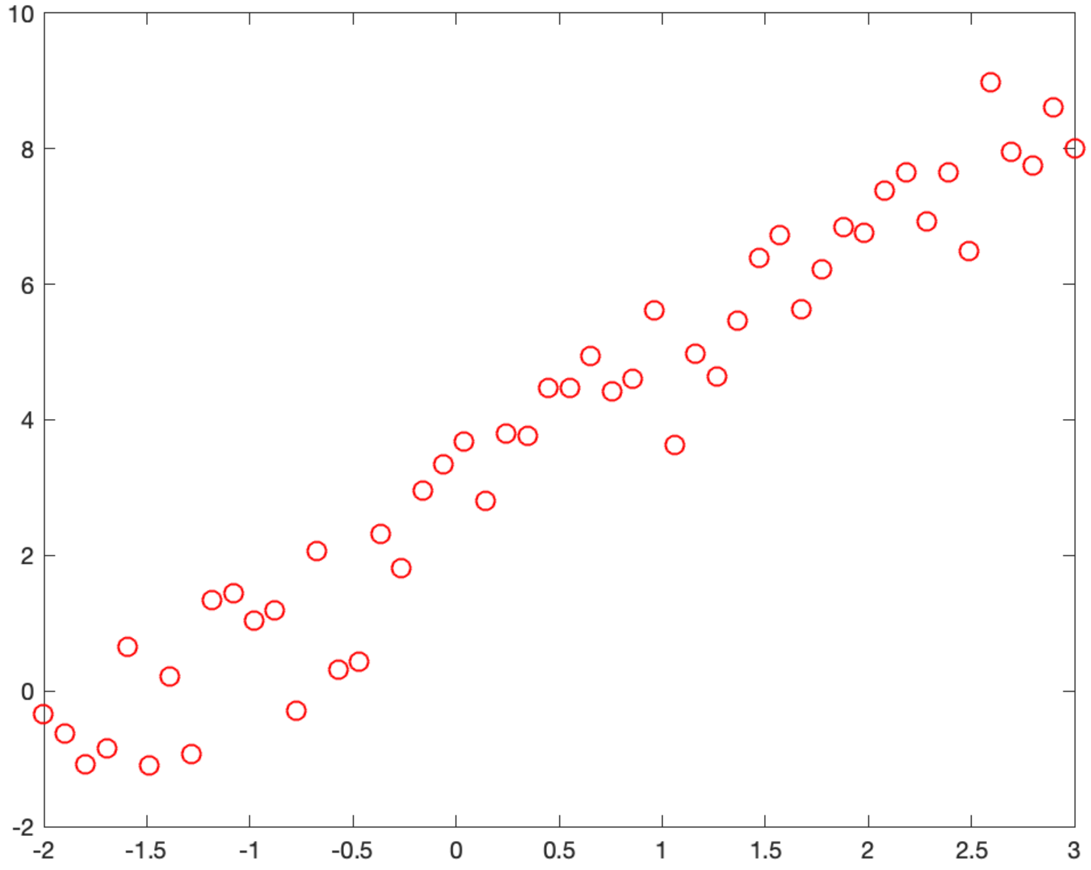
What do we optimize?

- ❑ Minimize errors



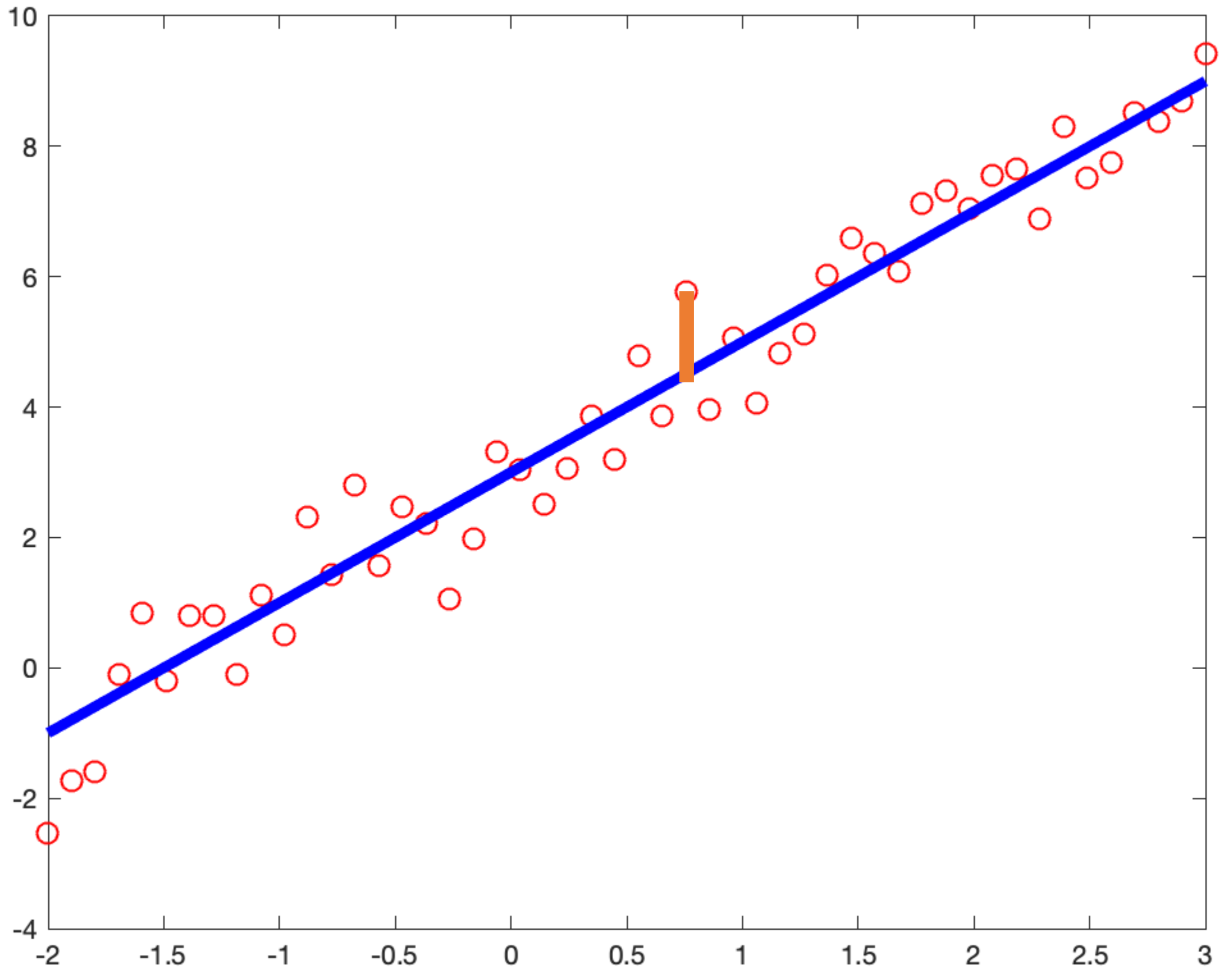
Line fitting

- ❑ Noisy Data points



Line fitting

- ❑ Fitting a line



Minimize for Model:

Sum(Model_Prediction – Data_point)

Line fitting

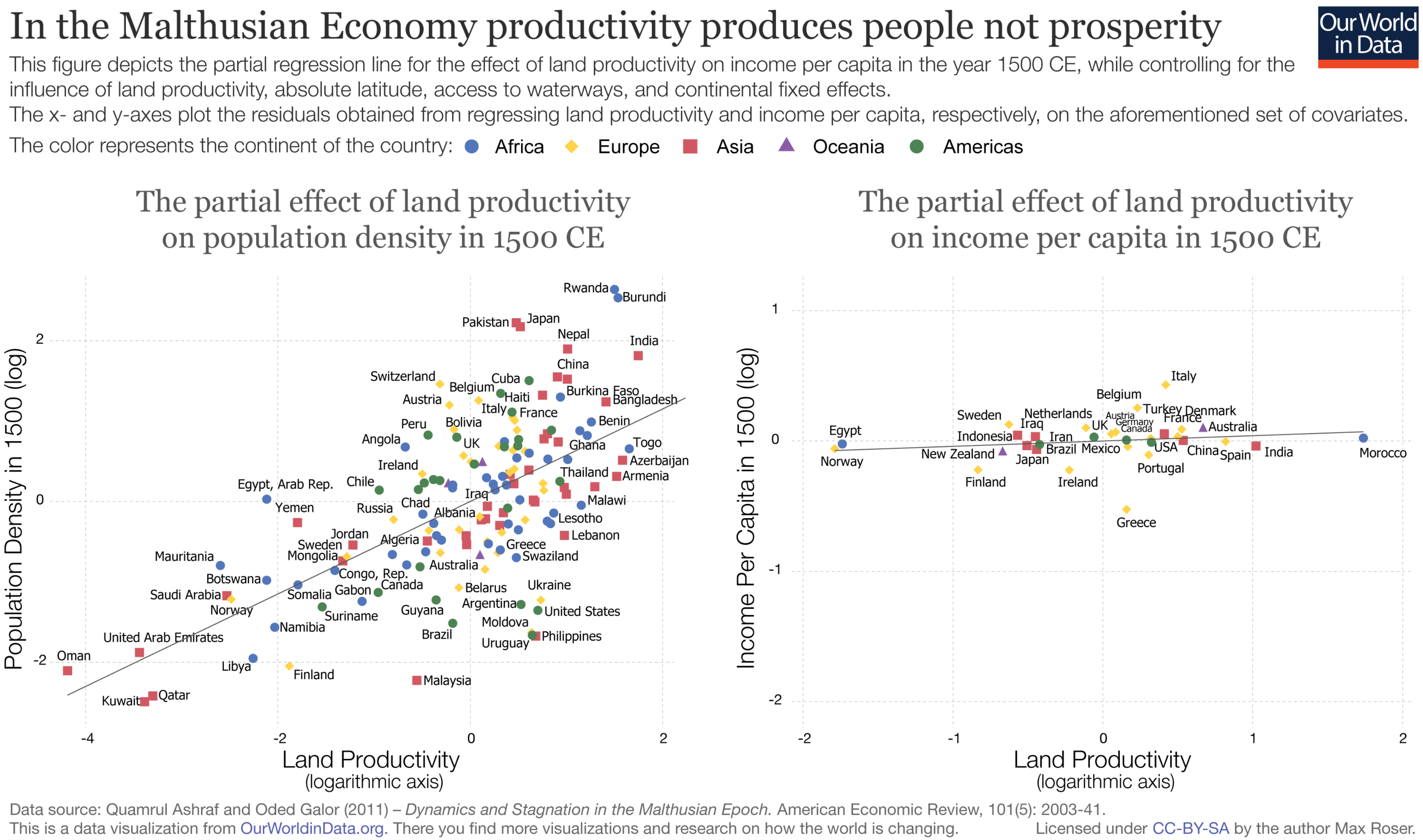
- Fitting a line

$$A\mathbf{w} = \mathbf{y}$$

A is the linear model

- Write down the equations: Input data points (x,y)

Line fitting



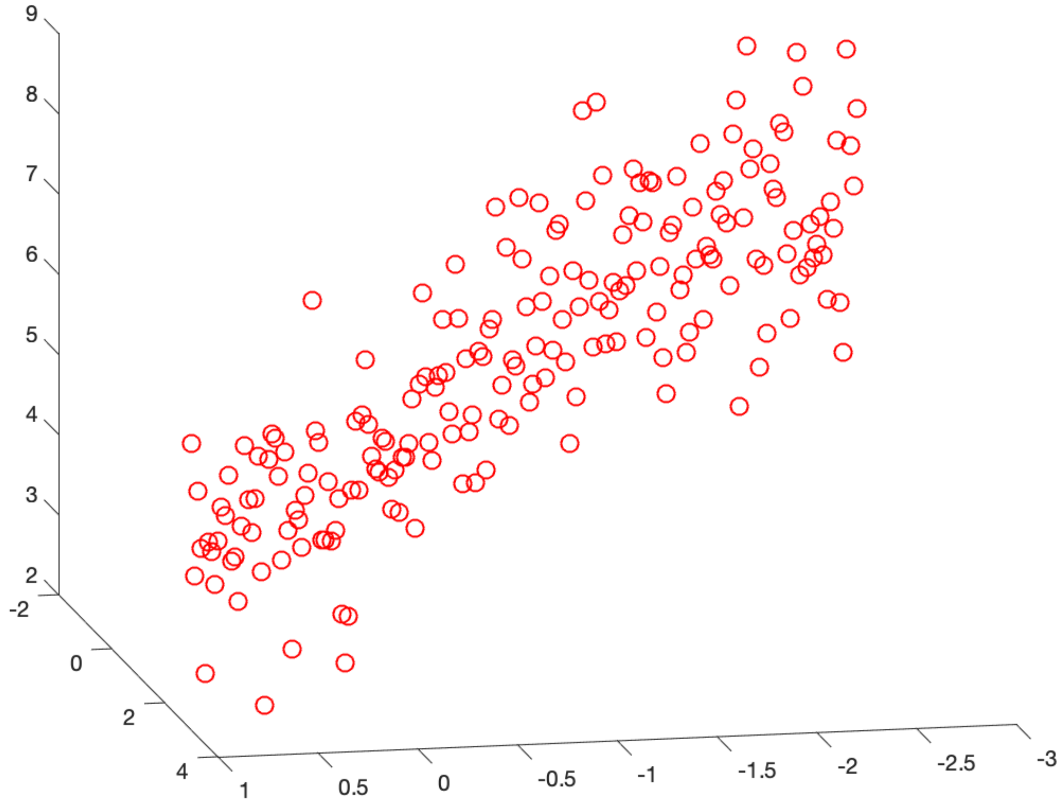
Source: Our World in data

Model fitting

- ❑ Least squares fitting

Plane fitting ⟶ hyperplane

Sphere fitting



Fundamental Matrix fitting

- Equation and scale

$$u_2u_1f_{11} + u_2v_1f_{12} + u_2f_{13} + v_2u_1f_{21} + v_2v_1f_{22} + v_2f_{23} + u_1f_{31} + v_1f_{32} + f_{33} = 0$$

$$\vdots$$

$f_{33} = 1$ $\Rightarrow$ Scale does not matter

Fundamental Matrix fitting

- Equation and scale

  - Recall SVD.
  - How is the scale fixed?

Wrong Measurements

- What are the measurements?

Recall experiments

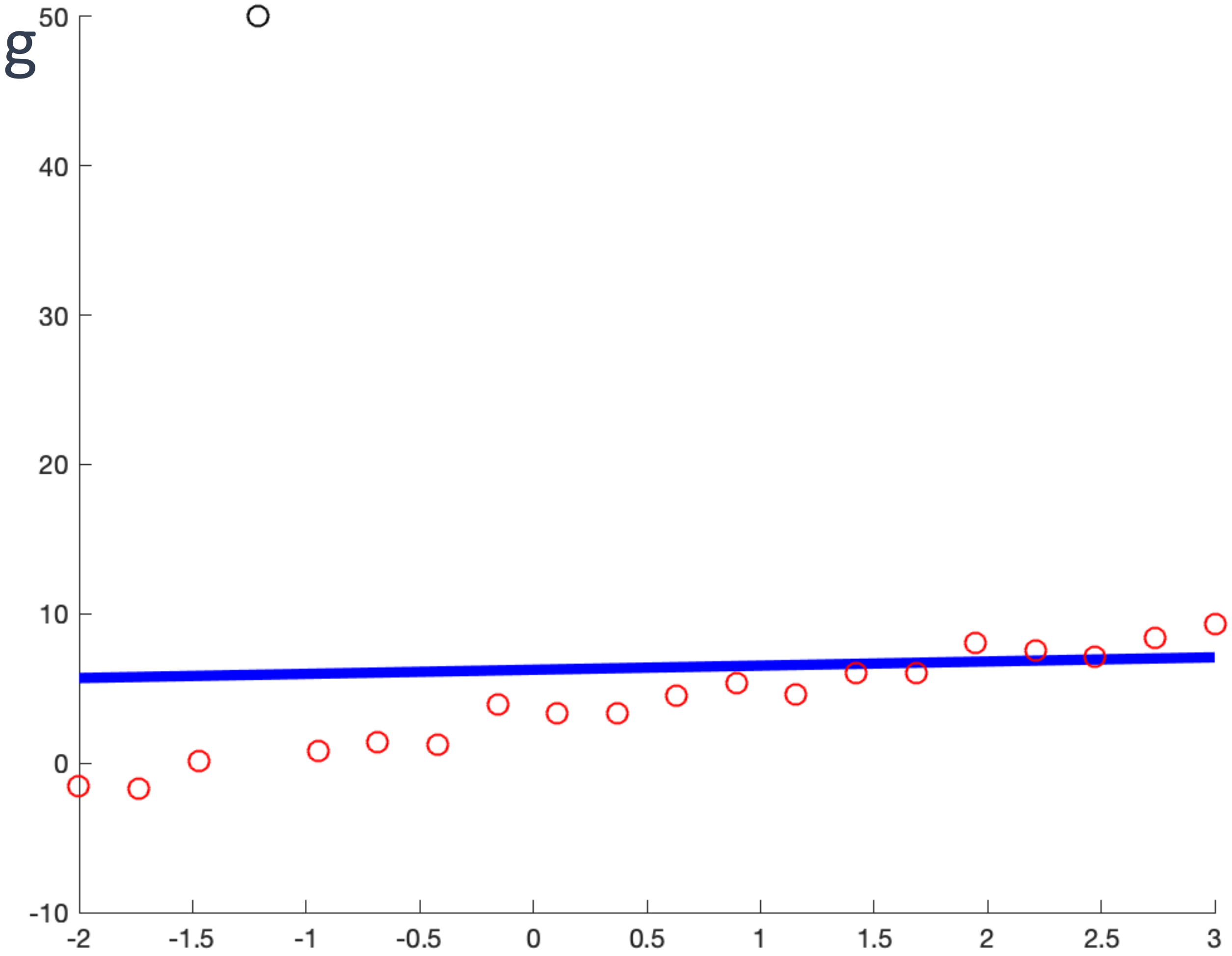
E.g., Young’s Modulus = Stress/Strain

1. 7.0e10
2. 7.3e10
3. 7.1e10
4. 6.7e10
5. 3.0e10

→ 6.2e10

Effects of Outliers

- Example: Line fitting



Effects of Outliers

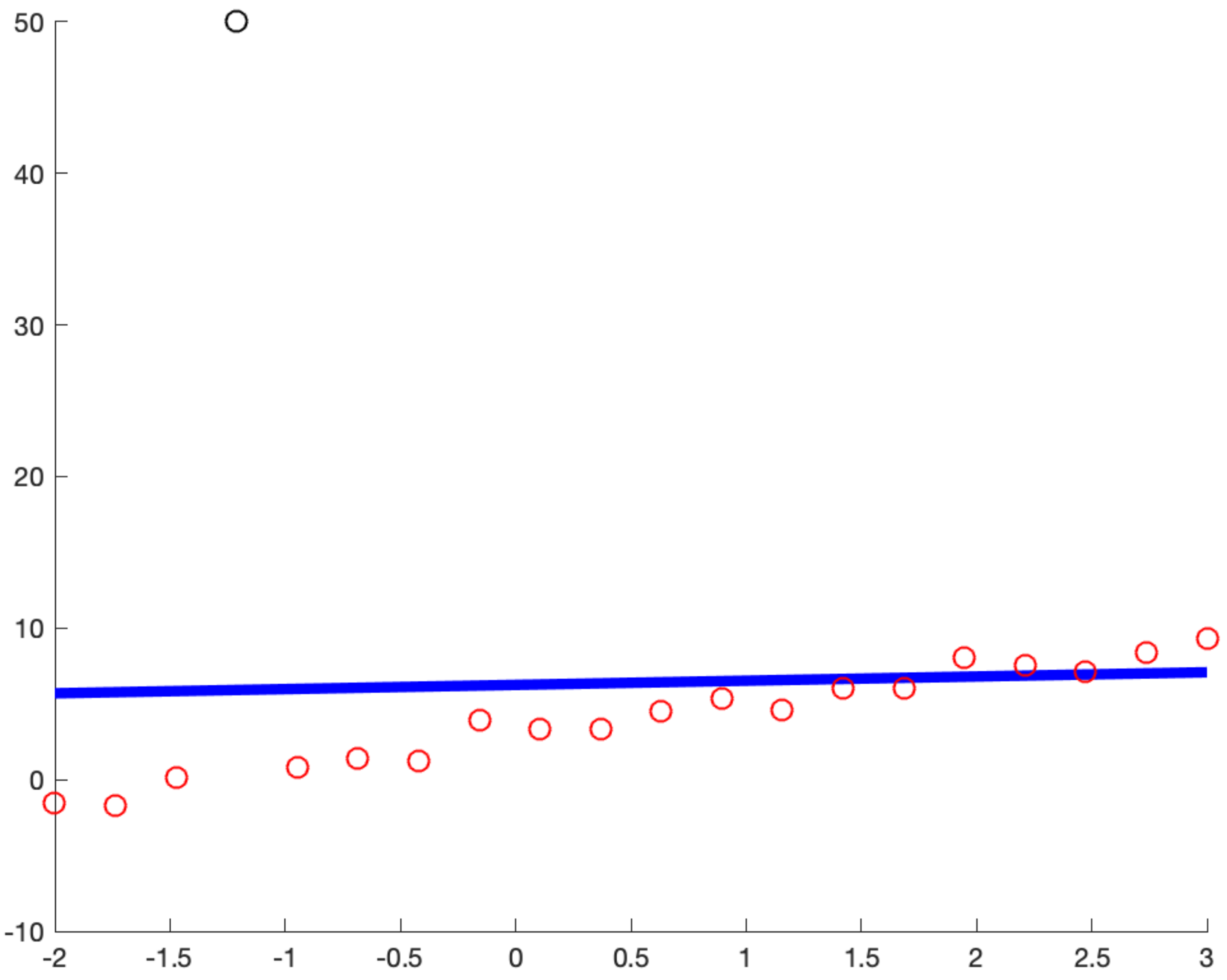
- Fundamental/Essential matrix
  - What could go wrong?

$$u_2 u_1 f_{11} + u_2 v_1 f_{12} + u_2 f_{13} + v_2 u_1 f_{21} + v_2 v_1 f_{22} + v_2 f_{23} + u_1 f_{31} + v_1 f_{32} + f_{33} = 0$$

How do you solve it?

- ❑ Back to line fitting

Remove the wrong measurements



Solution

- ❑ Line fitting

Method: Random Sampling and Consensus [Fischler et al., 1981]

Random Sampling

Consensus

Solution

- Fitting the Fundamental Matrix

$$u_2u_1f_{11} + u_2v_1f_{12} + u_2f_{13} + v_2u_1f_{21} + v_2v_1f_{22} + v_2f_{23} + u_1f_{31} + v_1f_{32} + f_{33} = 0$$

RANSAC

Take randomly ? number of points

Compute model F or E.

Compute residual

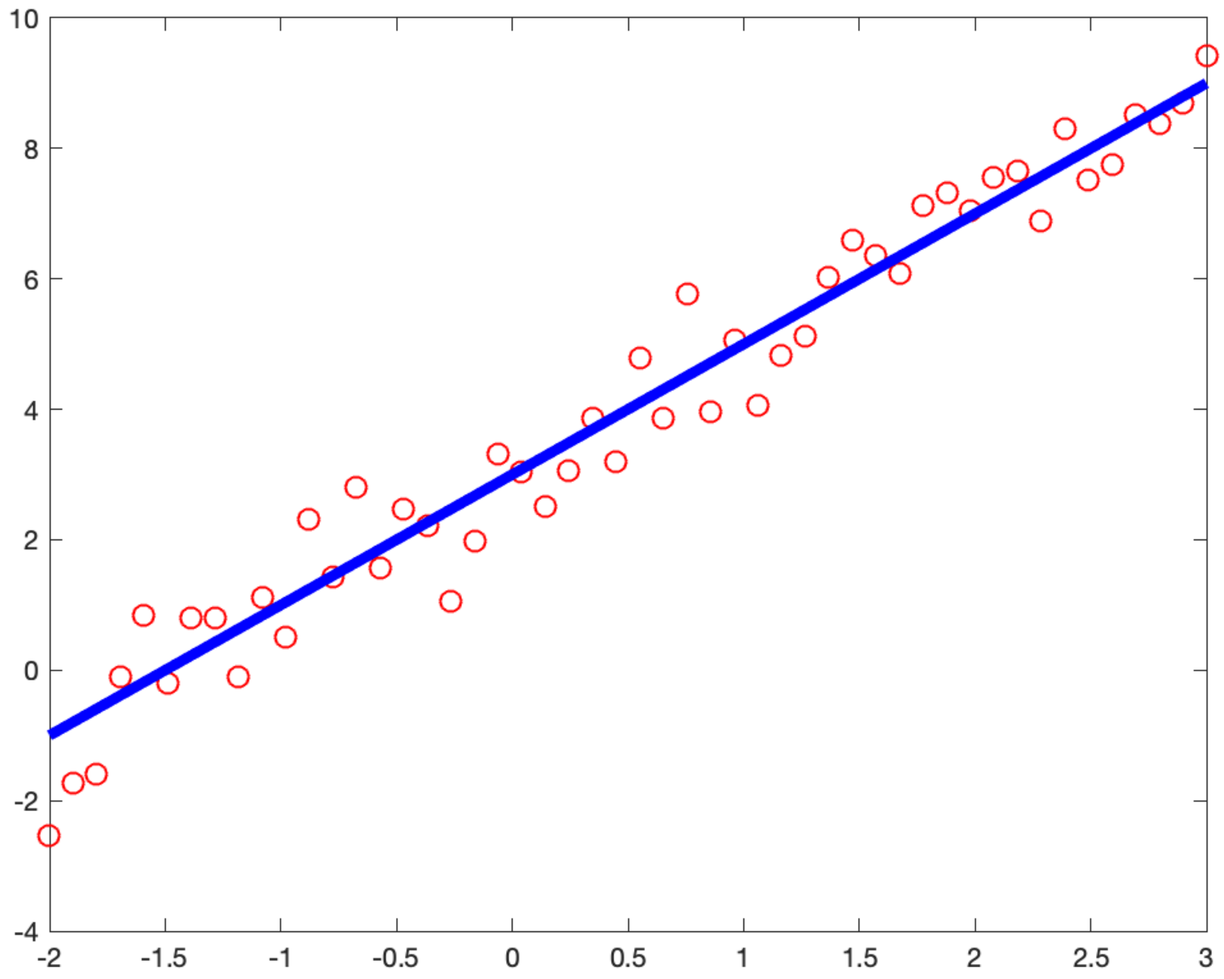
Compute how many inliers are there? Residual < ε

Optimization problem

- ❑ Expressing optimization

$$\underset{\mathbf{w}}{\text{minimize}} \|A\mathbf{w} - \mathbf{y}\|$$

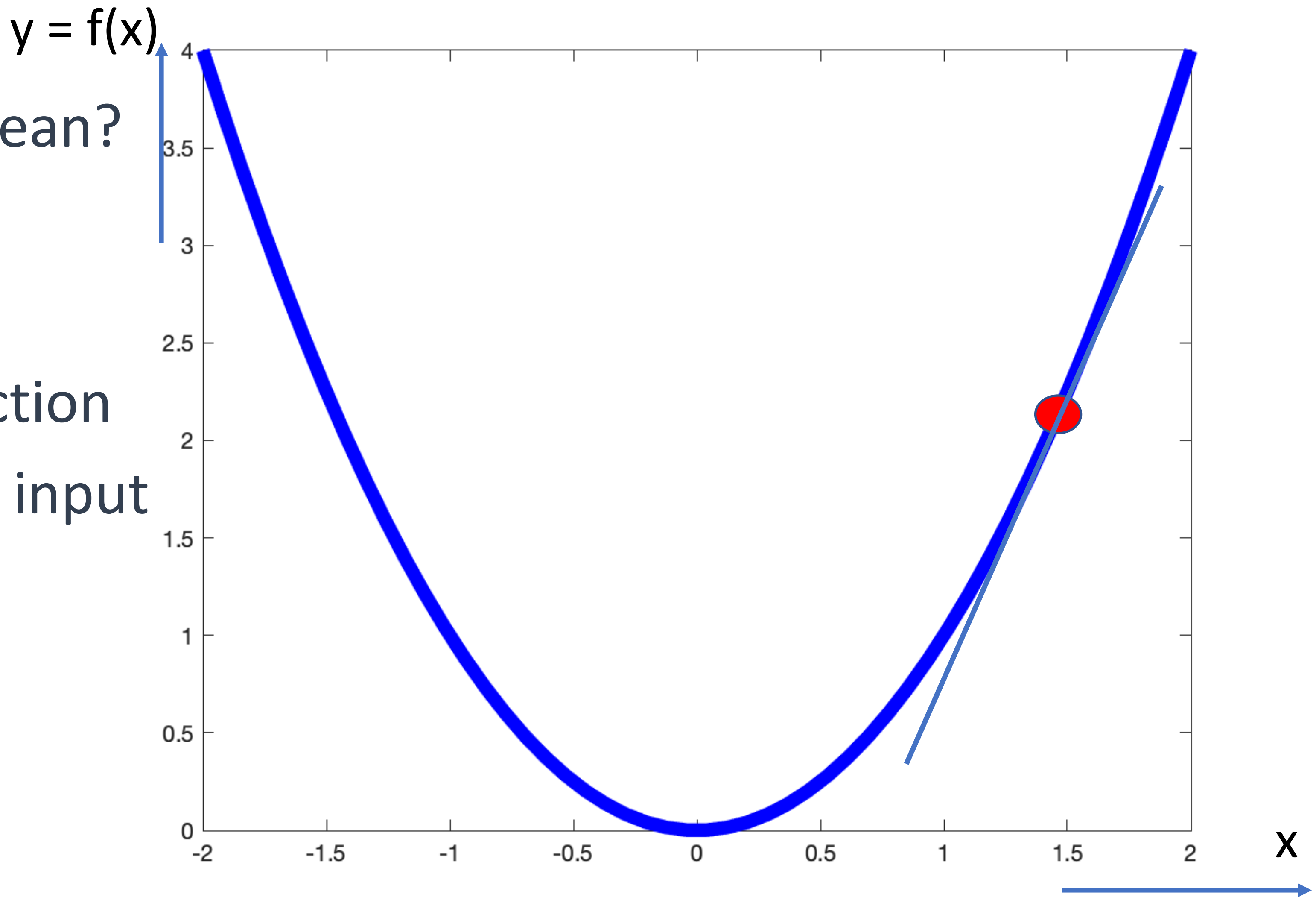
- ❑ Optimization example



Function Derivatives

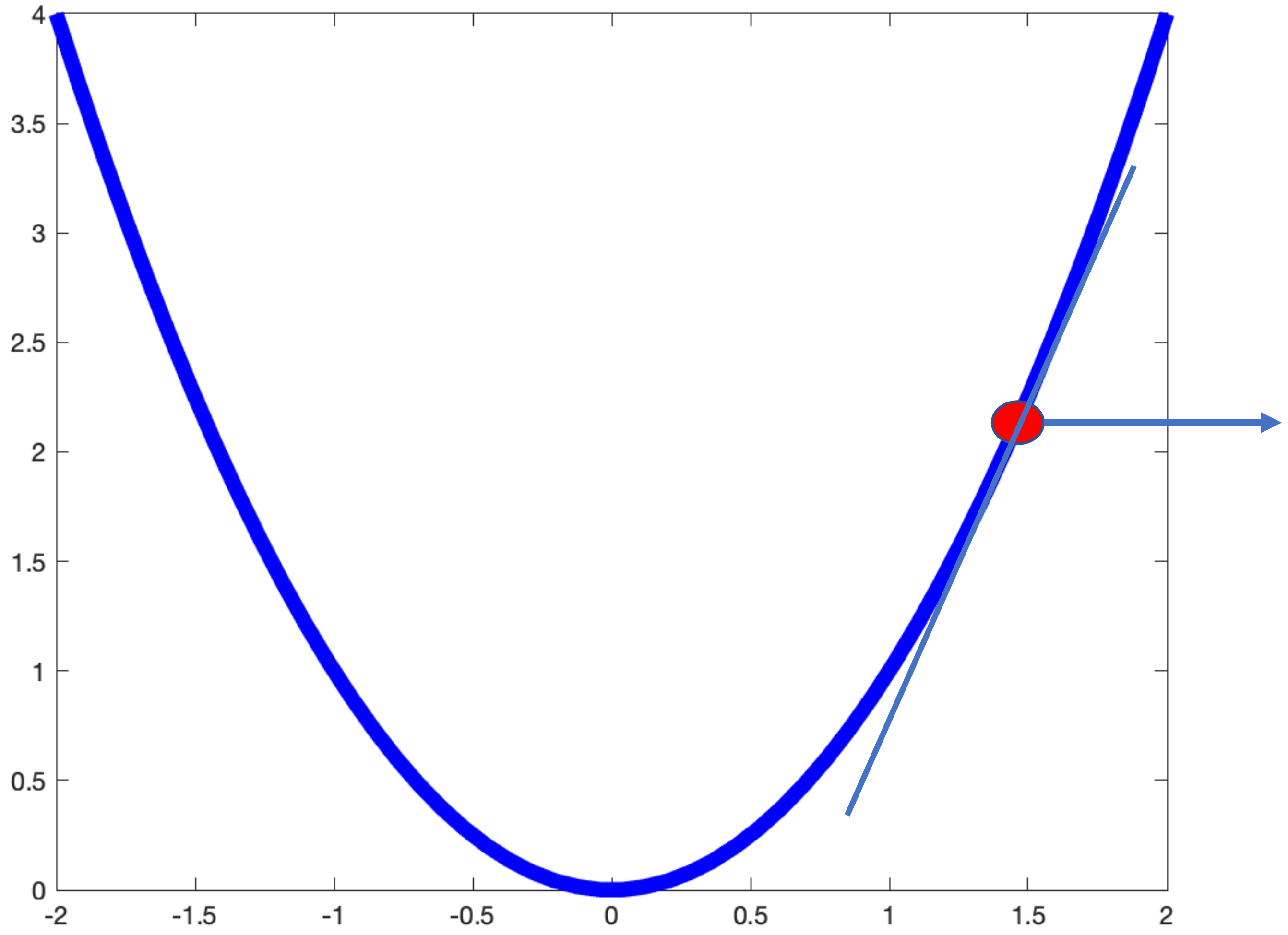
- ❑ What does it mean?

- ❑ Increase in function per unit change in input



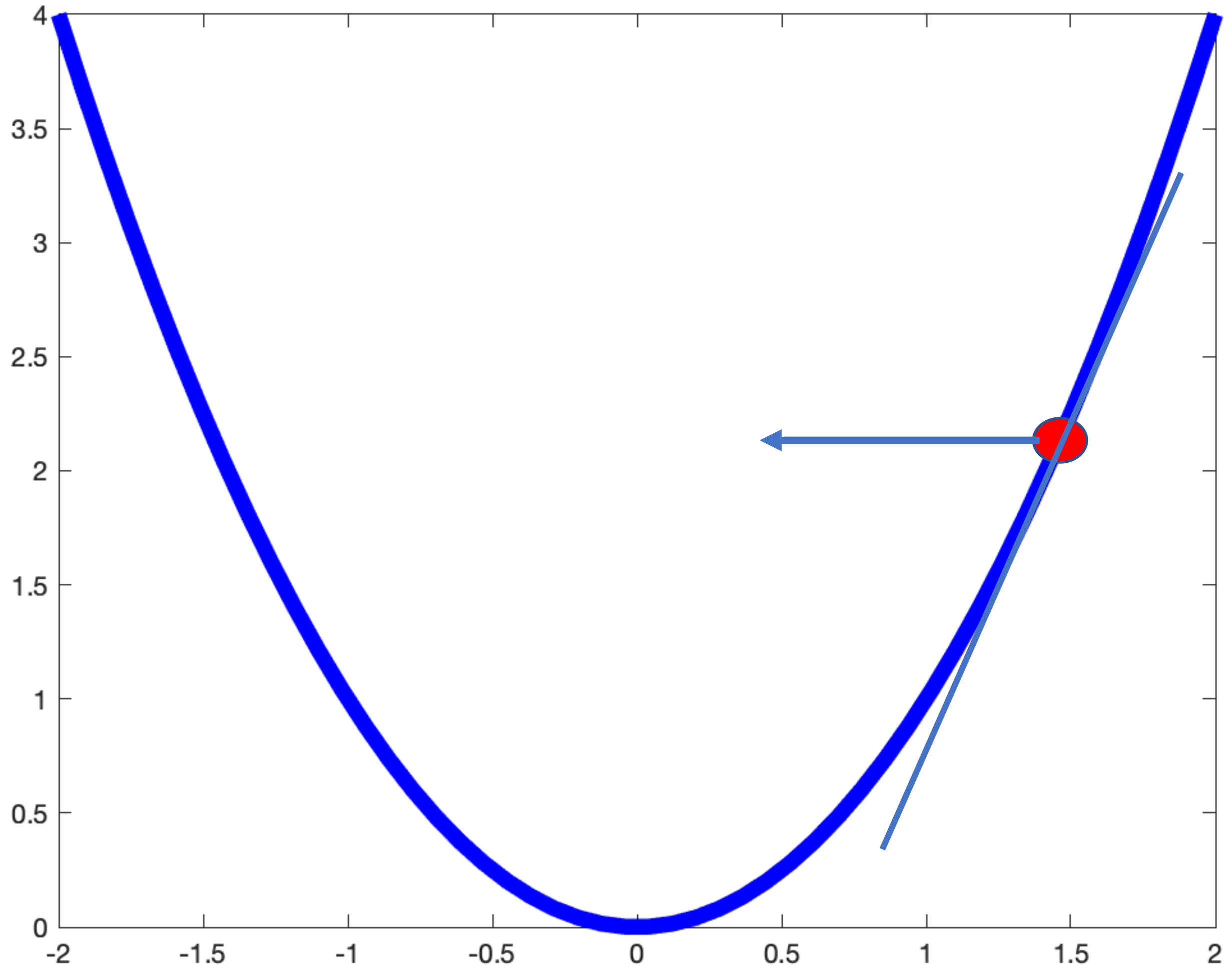
Direction of Derivatives

- ❑ What direction?
- ❑ Single component:
  - Direction
  - Magnitude



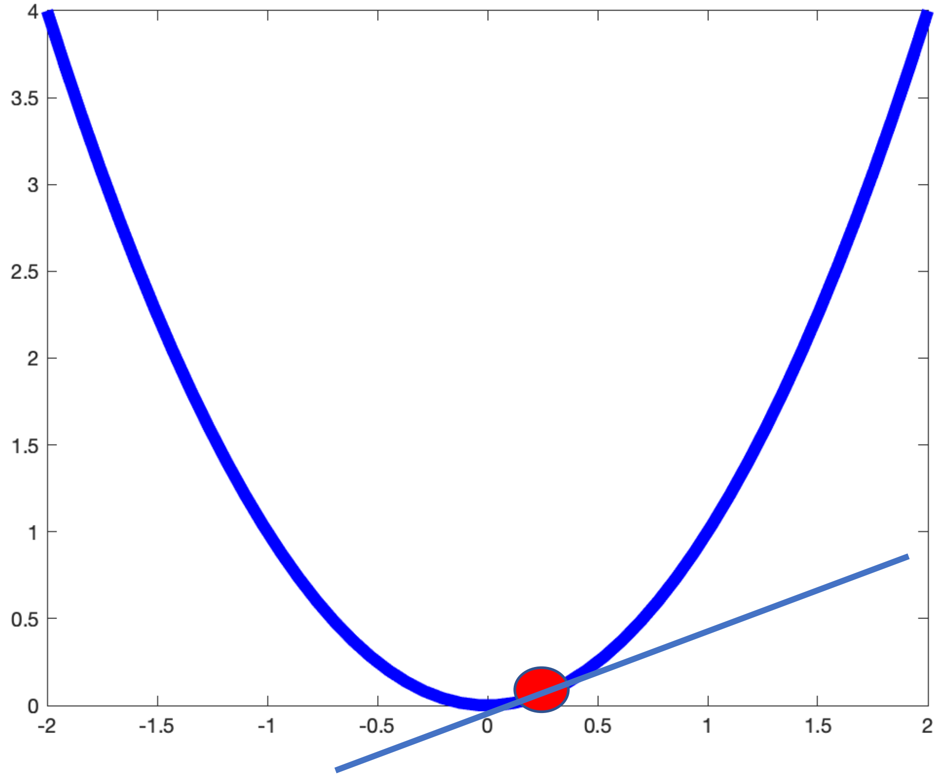
Gradient Descent

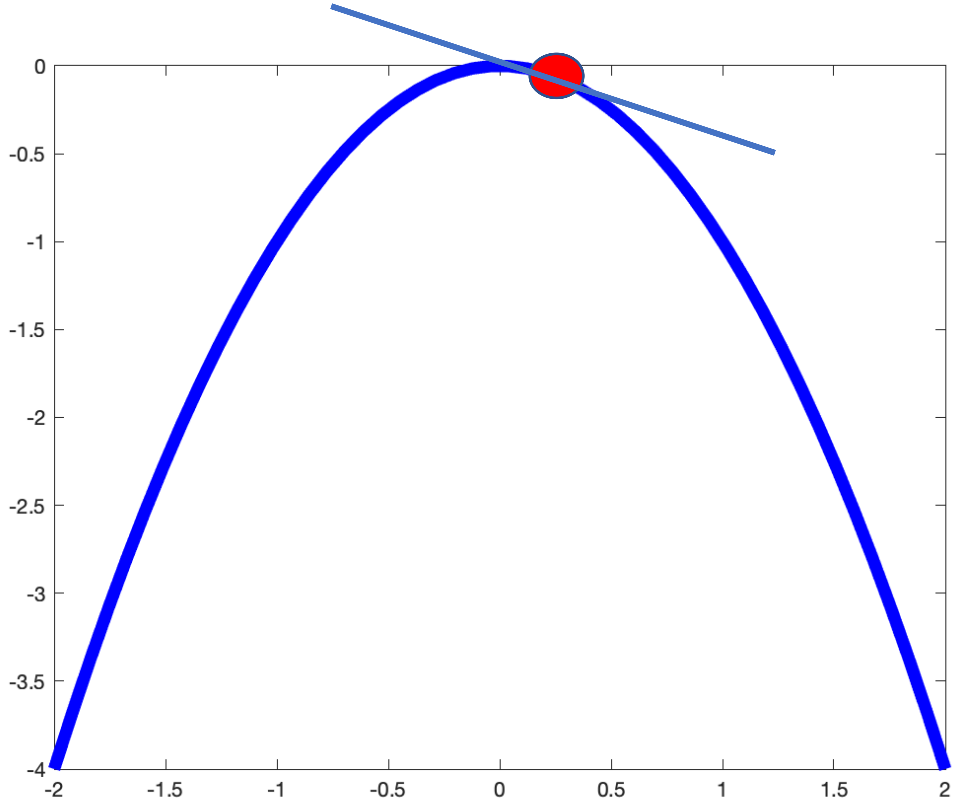
- ❑ Follow the direction opposite to gradients



Going against the Gradient

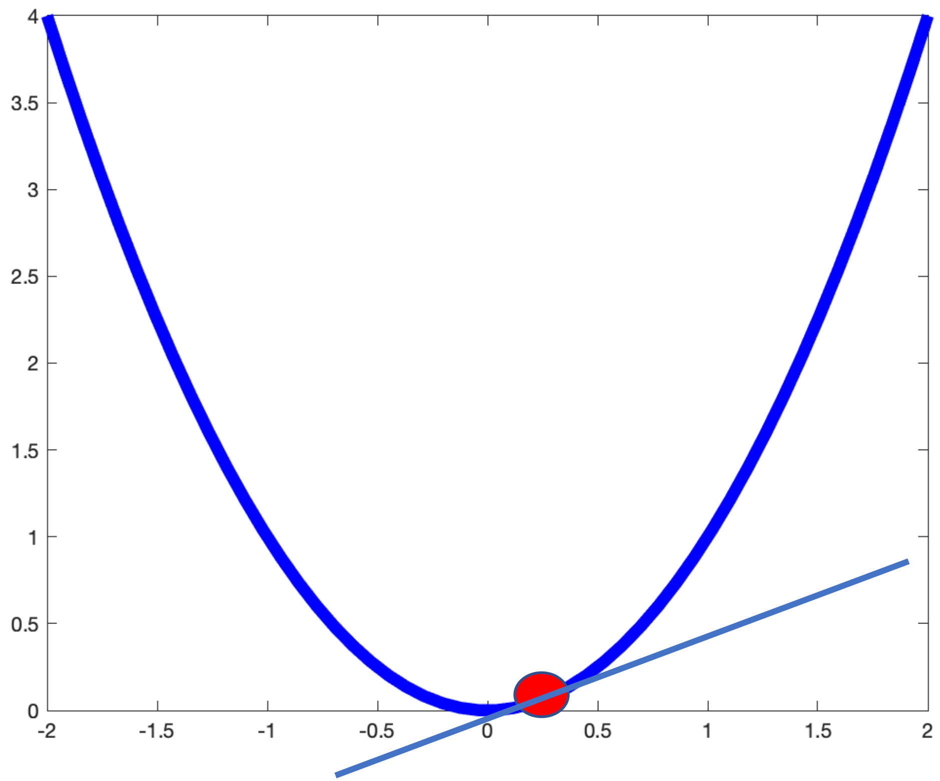
- Where does it lead to?

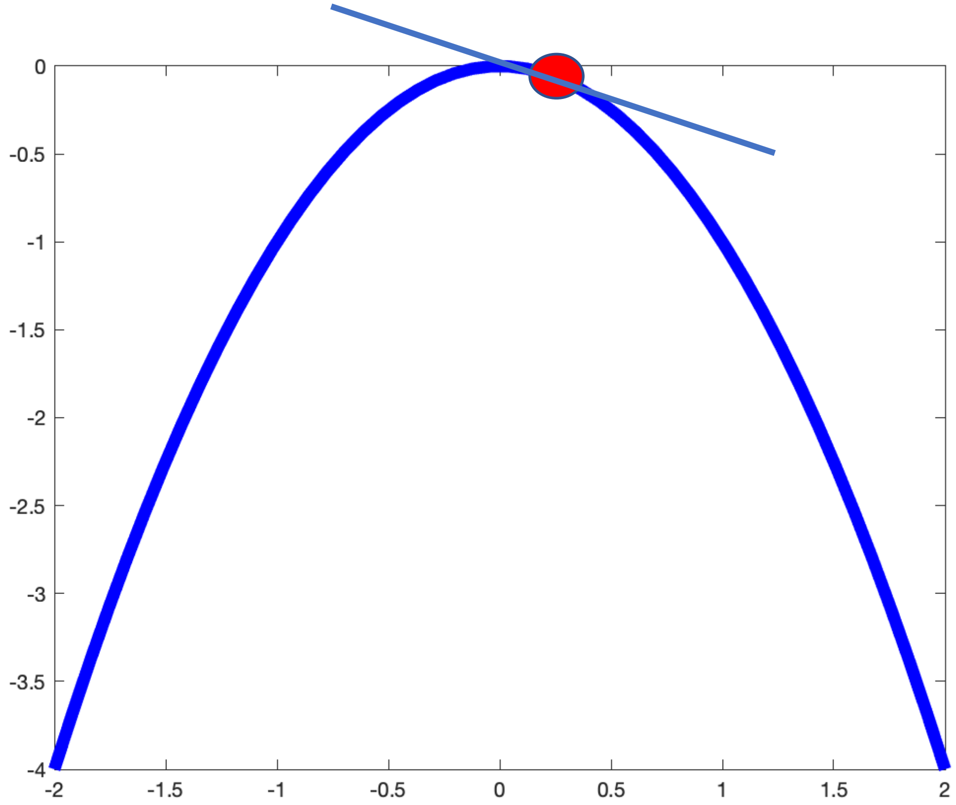




Going against the Gradient

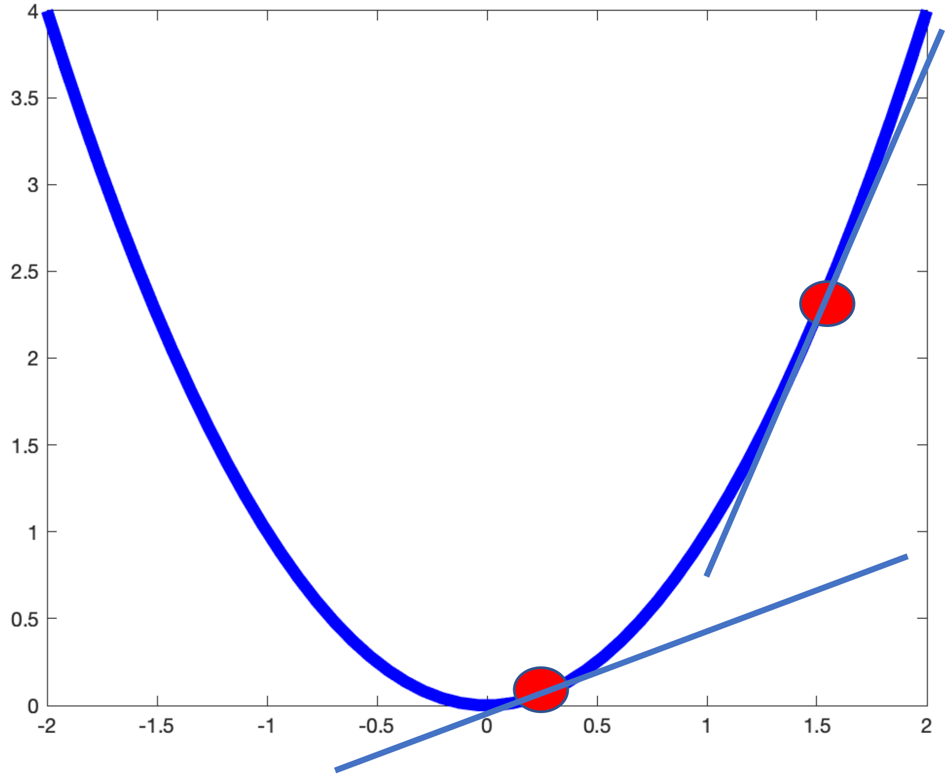
- Minimization and Maximization





Going against the Gradient

- Minimization

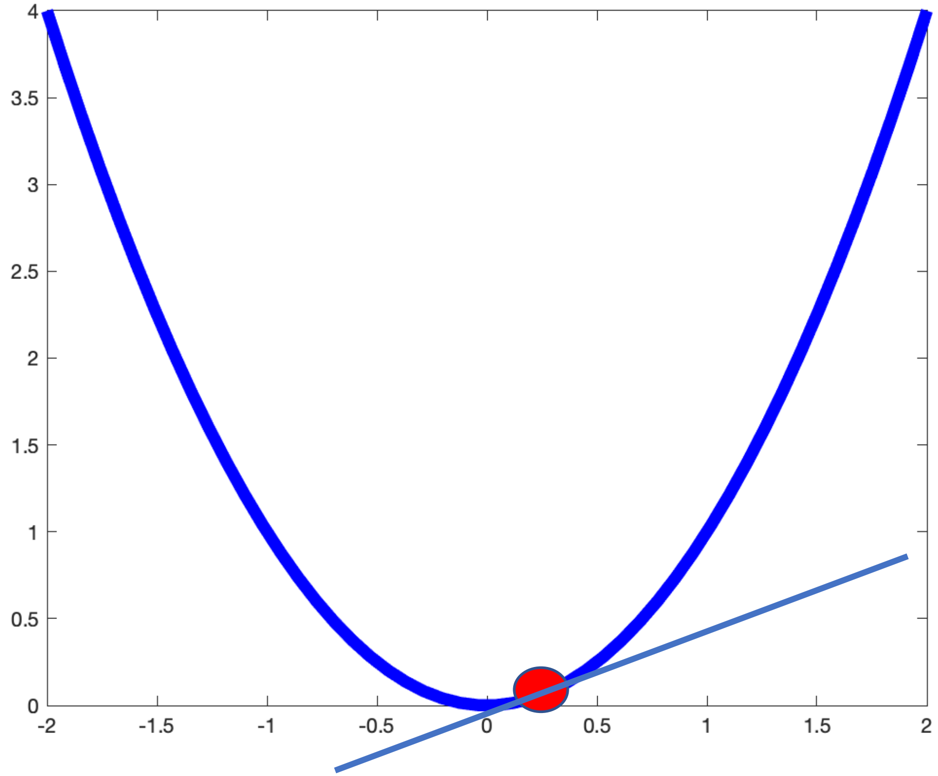


$$x^* = x^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x})$$

$$x^* = x^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x}) \ \mathrm{magnitude}(\frac{\partial y}{\partial x})$$

Going against the Gradient

- Minimization



$$x^* = x^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x})$$

$$x^* = x^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x}) \ \mathrm{magnitude}(\frac{\partial y}{\partial x})$$

$$y = x^2$$

$$\frac{\partial y}{\partial x} = \frac{\partial x^2}{\partial x} = 2x$$

Going against the Gradient

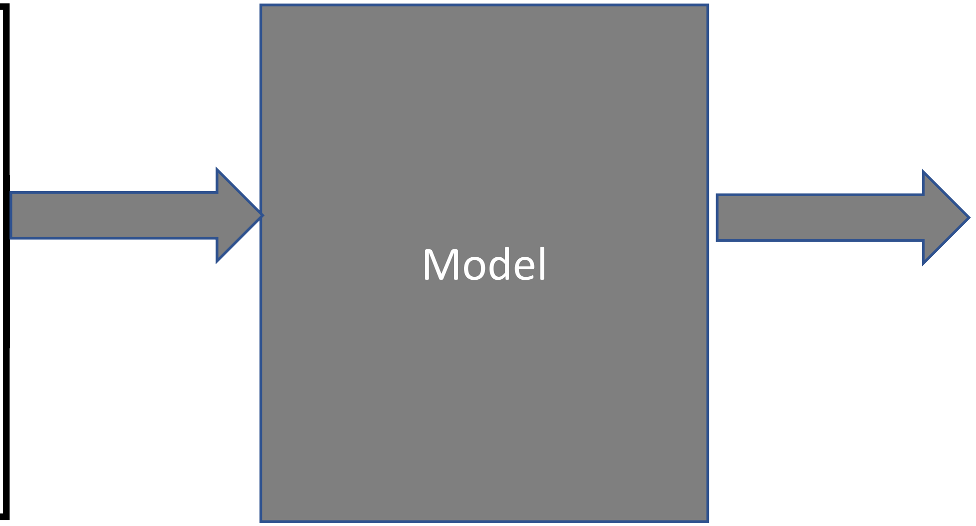
https://www.youtube.com/watch?v=kJgx2RcJKZY

Multivariate functions

❑ Multiple input variables

Model Parameters

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



$$y = f(\mathbf{x})$$

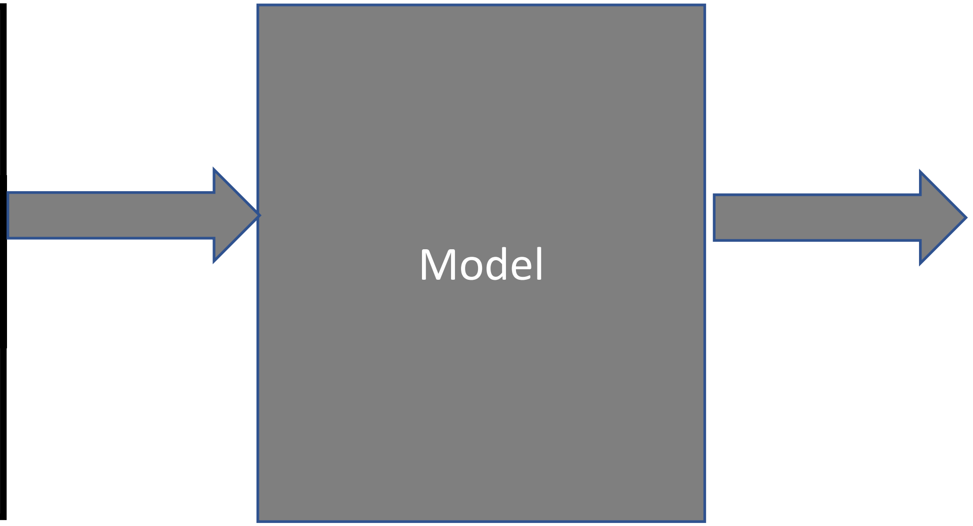
https://www.youtube.com/watch?v=kJgx2RcJKZY

Derivatives for multiple variables

- ❑ The Jacobian

Model Parameters

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



$$y = f(\mathbf{x})$$

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}^\top$$

Update equations

- Gradient descent

$$x_1^* = x_1^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x_1}) \ \mathrm{magnitude}(\frac{\partial y}{\partial x_1})$$

$$x_2^* = x_2^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x_2}) \ \mathrm{magnitude}(\frac{\partial y}{\partial x_2})$$

$$\vdots$$

$$x_n^* = x_n^0 - \lambda \ \mathrm{dir}(\frac{\partial y}{\partial x_n}) \ \mathrm{magnitude}(\frac{\partial y}{\partial x_n})$$

Update equations

- Gradient descent

$$x_1^* = x_1^0 - \lambda \, \frac{\partial y}{\partial x_1}$$

$$x_2^* = x_2^0 - \lambda \, \frac{\partial y}{\partial x_2}$$

$$\vdots$$

$$x_n^* = x_n^0 - \lambda \, \frac{\partial y}{\partial x_n}$$

$$\mathbf{x}^* = \mathbf{x}^0 - \lambda \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

Update equations

- Gradient descent

$$x_1^* = x_1^0 - \lambda \, \frac{\partial y}{\partial x_1}$$

$$x_2^* = x_2^0 - \lambda \, \frac{\partial y}{\partial x_2}$$

$$\vdots$$

$$x_n^* = x_n^0 - \lambda \, \frac{\partial y}{\partial x_n}$$

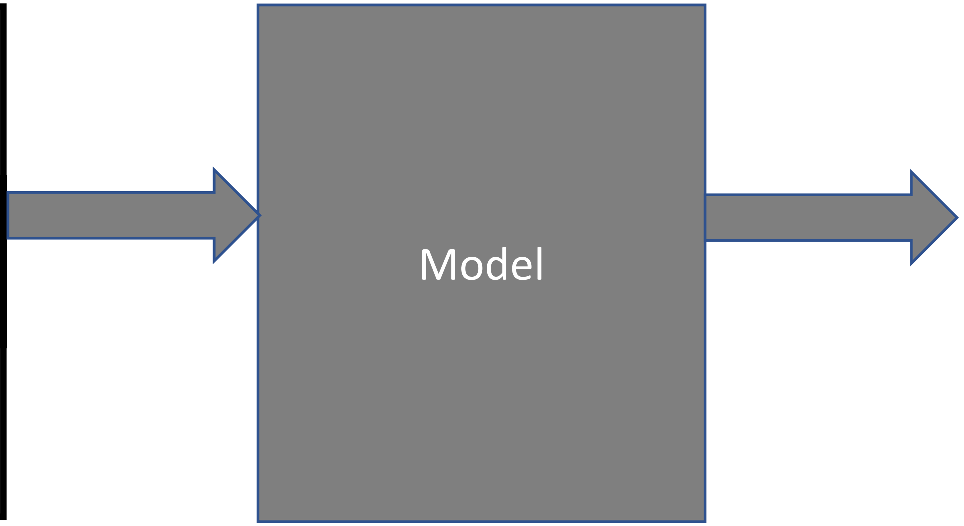
$$\mathbf{x}^* = \mathbf{x}^0 - \lambda J(\mathbf{x}^0)^\top$$

Multiple outputs

- ❑ How do you minimize a vector?

Model Parameters

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



$$\mathbf{y} = f(\mathbf{x}) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Multiple outputs

- What does the vector represent? Residuals/ Differences between predictions and training labels

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \longrightarrow \boxed{\text{Model}} \longrightarrow \mathbf{y} = f(\mathbf{x}) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

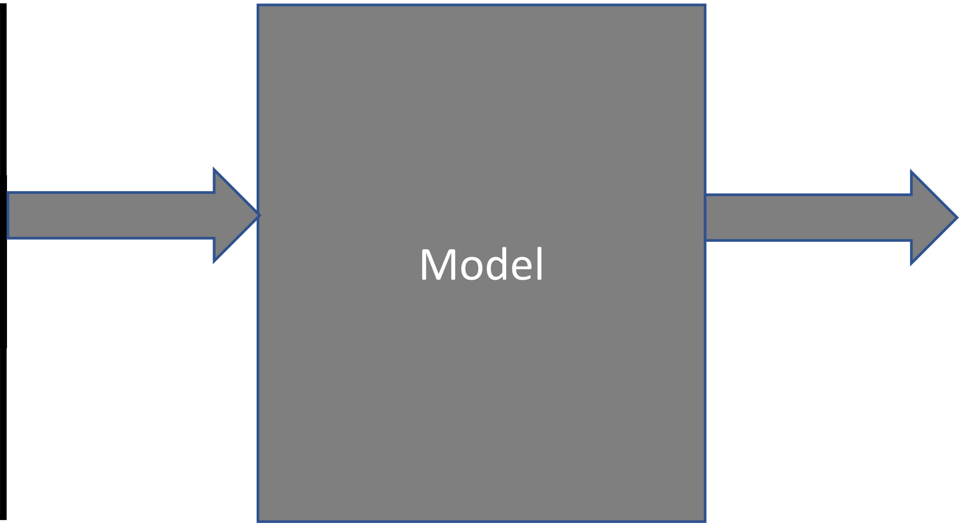
Model Parameters

Multivariate functions

- Minimize the square of the norm $\|\mathbf{y}\|^2 = \mathbf{y}^\top \mathbf{y}$

Model Parameters

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



$$\mathbf{y} = f(\mathbf{x}) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Derivatives for Multiple outputs

- Derivatives

$$\frac{\partial \|\mathbf{y}\|^2}{\partial \mathbf{x}} = 2\mathbf{y}^\top \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = 2\mathbf{y}^\top J(\mathbf{x})$$

$$J = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_1} & \cdots & \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Optimizing with Gradient Descent

- Equations – minimize single scalar y

$$\mathbf{x}^* = \mathbf{x}^0 - \lambda \frac{\partial y}{\partial \mathbf{x}} \longrightarrow \underset{\mathbf{a}}{\text{minimize}}\ y = f(\mathbf{x})$$

$$\mathbf{x}^* = \mathbf{x}^0 - \lambda J(\mathbf{x}^0)^\top$$

- Equations – minimize vector norm $\|\mathbf{y}\|^2 = \mathbf{y}^\top \mathbf{y}$

$$\mathbf{x}^* = \mathbf{x}^0 - \lambda \frac{\partial \|\mathbf{y} - \mathbf{y}^0\|^2}{\partial \mathbf{x}}$$

$$\mathbf{x}^* = \mathbf{x}^0 - \lambda 2 J(\mathbf{x}^0)^\top (\mathbf{y} - \mathbf{y}^0)$$

Convergence of Gradient Descent

- What happens if the gradient magnitude is very high?

- What happens if the gradient magnitude is very low?

Gauss Newton's method

- Motivation

For faster convergence assume second-order approximation of cost

Taylor's series:

$$\mathbf{y} = f(\mathbf{x}) = f(\mathbf{x}^0) + (\mathbf{x} - \mathbf{x}^0)^\top \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} + (\mathbf{x} - \mathbf{x}^0)^\top \frac{\partial f(\mathbf{x})^2}{\partial^2 \mathbf{x}} (\mathbf{x} - \mathbf{x}^0)$$

0

Gauss Newton's method

❑ Motivation

For faster convergence assume second-order approximation

$$\mathbf{y} = \mathbf{y}^0 + J(\mathbf{x})^0(\mathbf{x}^* - \mathbf{x}^0)$$

$$\underset{\mathbf{x}^*-\mathbf{x}^0}{\text{minimize}} \; \|\mathbf{y}^0 + J(\mathbf{x})^0(\mathbf{x}^* - \mathbf{x}^0)\|^2$$

$$\mathbf{x}^* = \mathbf{x}^0 - \left(J^\top J\right)^{-1} J^\top \mathbf{y}^0$$

Gauss Newton’s method

- ❑ Discuss problems

Back to the board!

Levenberg Marquardt Method

- ❑ Motivation

Combine both Gradient Descent and Gauss - Newton

$$\mathbf{x}^* = \mathbf{x}^0 - \left(J^\top J + \lambda I\right)^{-1} J^\top \mathbf{y}^0$$

Damping or regularizer

Levenberg Marquardt Method

- ❑ More about LM

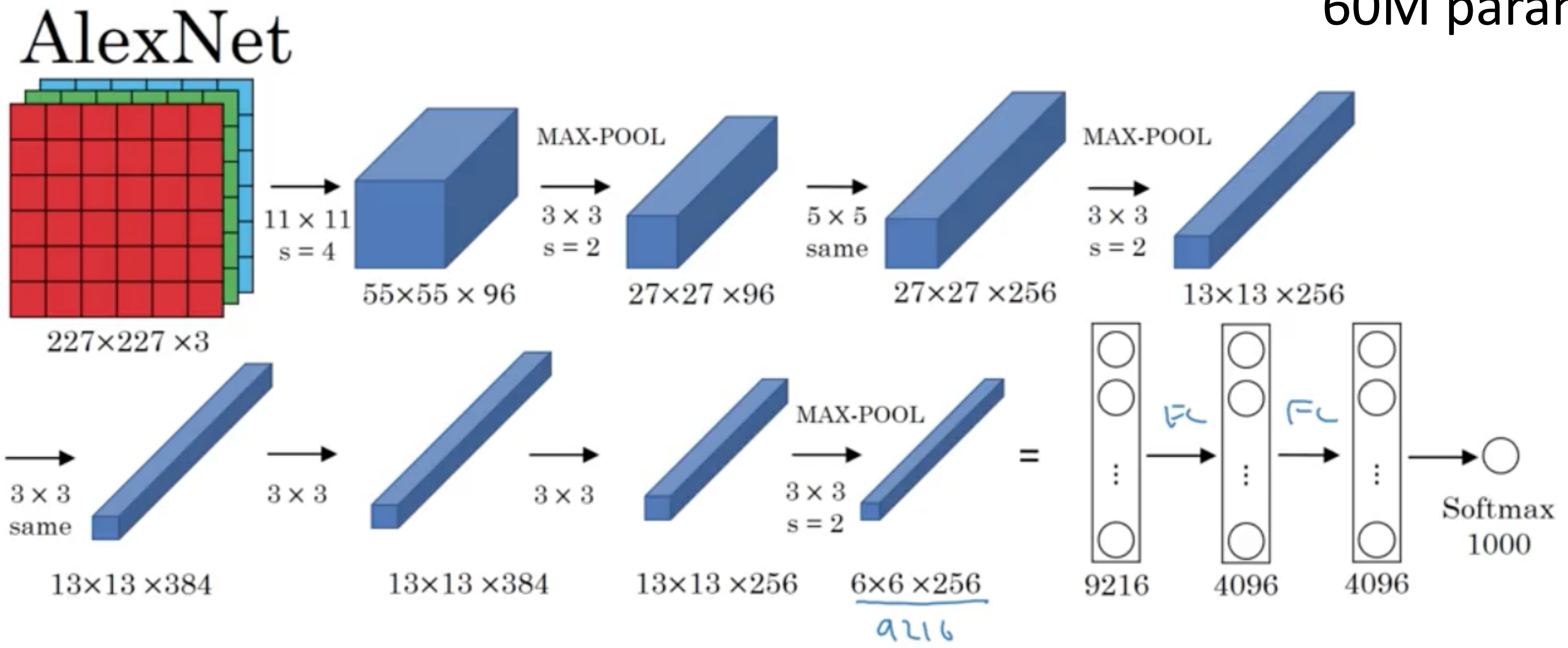
Often first method of choice

Why isn’t it used in Deep Learning?

Sneak - peak into current deep learning

- Size and computation of Jacobian

60M parameters



[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

Andrew Ng

Stochastic Gradient Descent

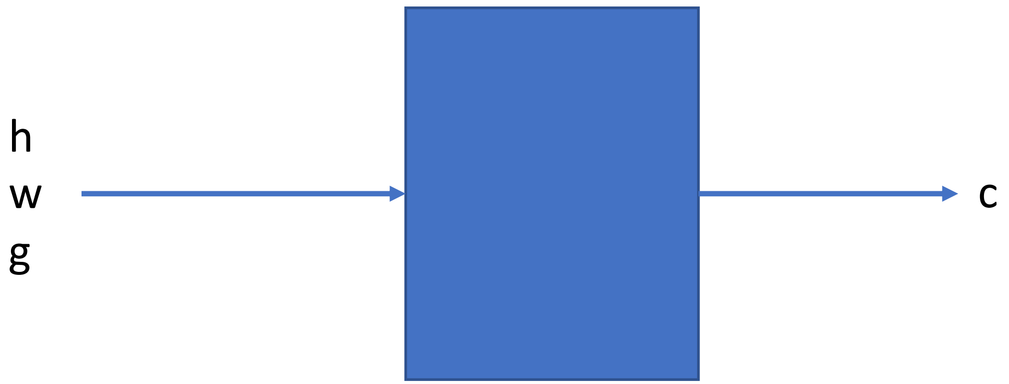
- ❑ Dummy example

$$J_{\text{train}} = \sum^{50} J$$

Training dataset:

50 people: each person is represented by height, weight and gender

50 people: predict calories spent



- ❑ Gradient Descent evaluates all samples for each update

Stochastic Gradient Descent

$$J_{\text{sgd}} = J_i$$

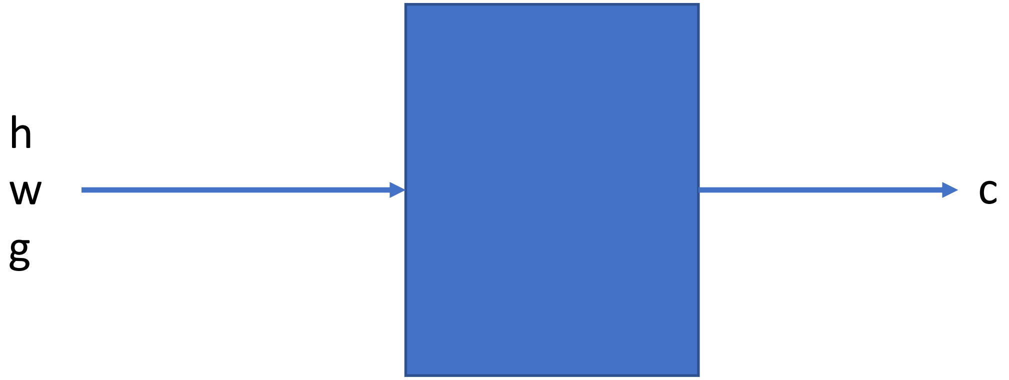
$$i = \text{random\_index}$$

- Dummy example

Training dataset:

50 people: each person is represented by height, weight and gender

50 people: predict calories spent



- Stochastic Gradient Descent takes only one random sample

Objective functions

- How do we design them?

Errors or residuals

They are usually vectors

Objective functions

- ❑ How do we design them?

- ❑ Examples:

Distance functions

- Norm of differences

Used to measure distances

Properties of norms

- L - 1 norm

$$\|\mathbf{x}\|_1$$

- L -2 norm

$$\|\mathbf{x}\|_2$$

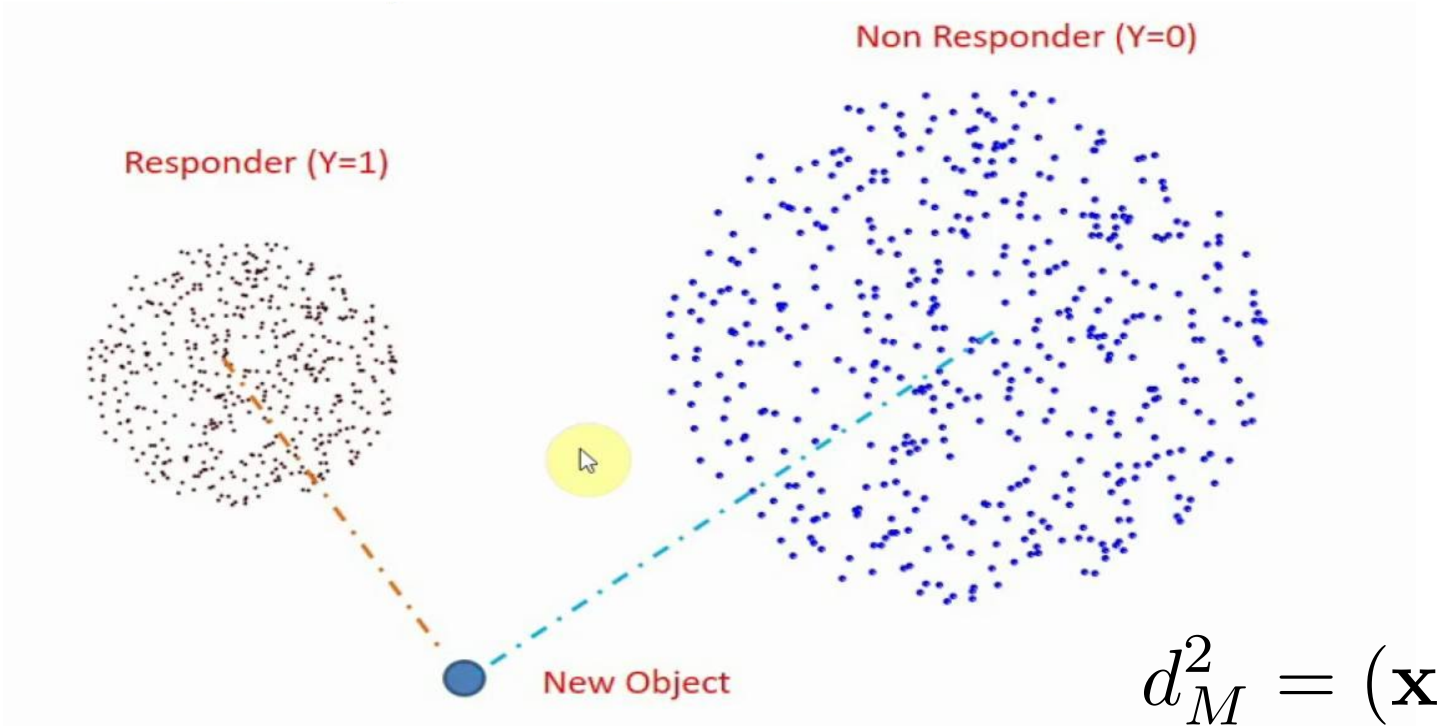
Real Examples of Optimization

- Bundle Adjustment

Other distance functions

- The Mahalonobis distance

$$C = \frac{1}{n-1}\sum(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^\top$$



$$d_M^2 = (\mathbf{x} - \mu)^\top C^{-1}(\mathbf{x} - \mu)$$

Convexity