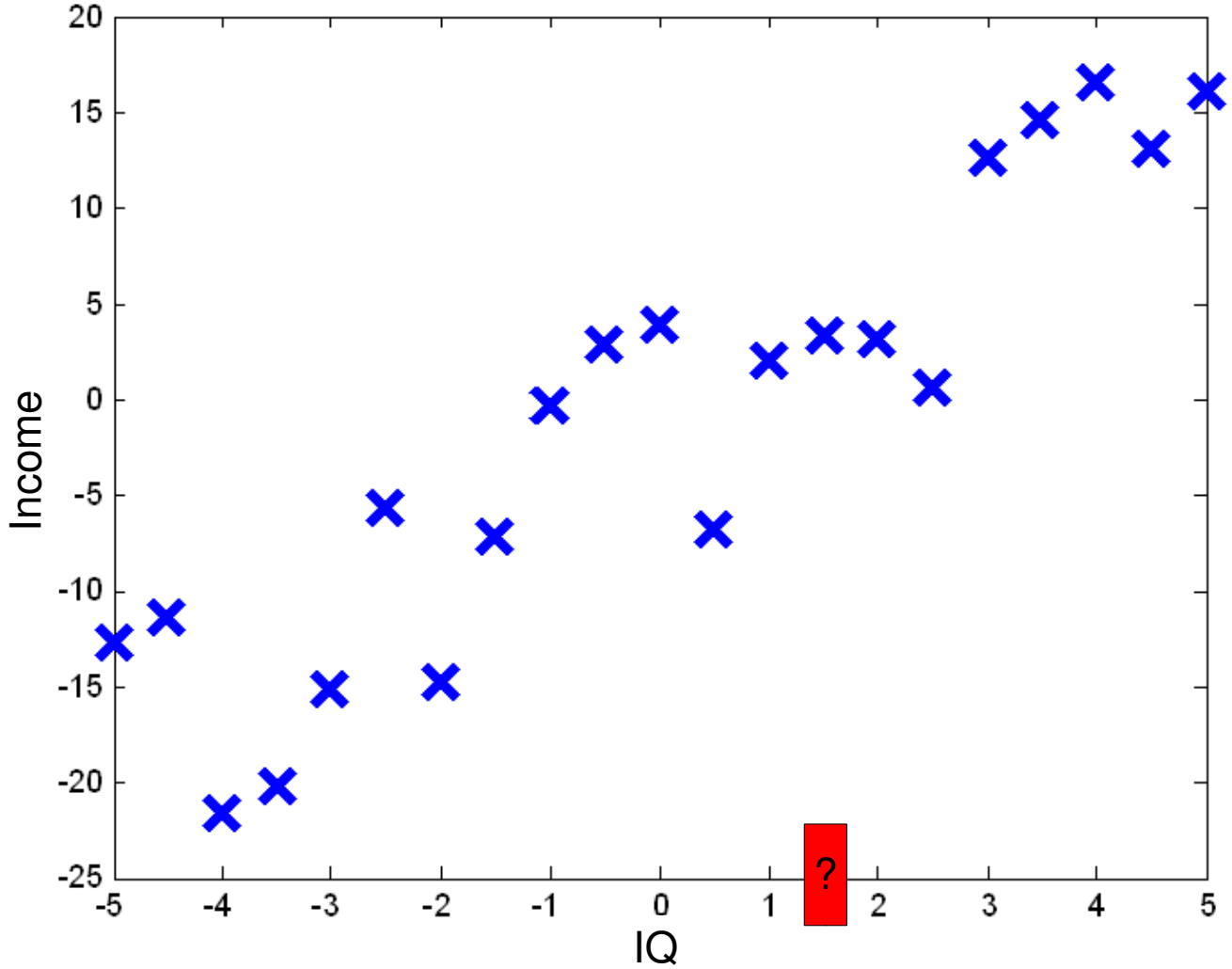
Prediction

An example

- The idea is to predict the income of somebody only by knowing this person's IQ
- Assumption: The income is a function of the IQ
- The samples were collected with a questionnaire among empoyees
- Can be used to predict and even forecast the income of somebody by knowing the person's IQ

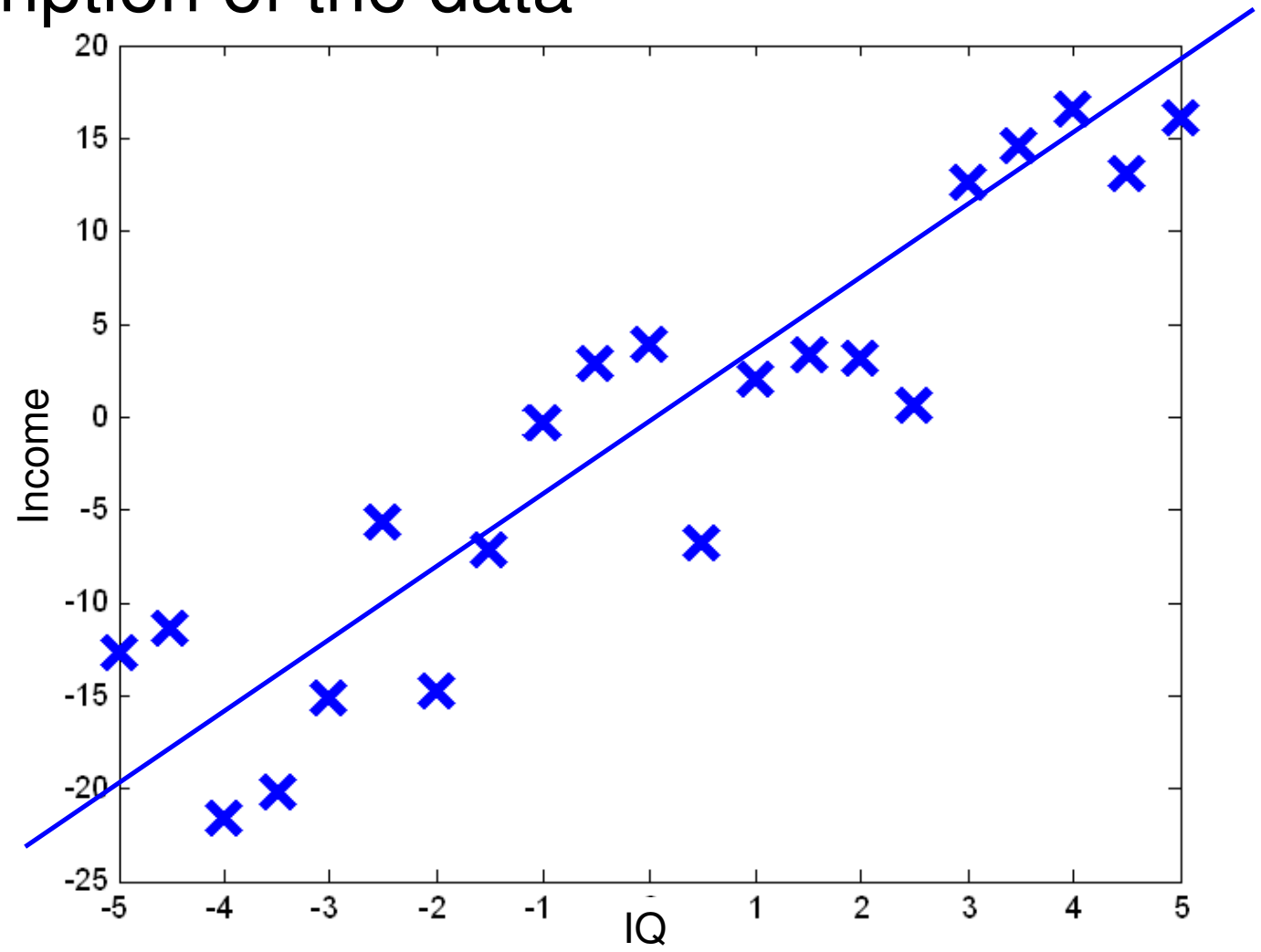
Prediction of new sample

- Predict the dependent variable for a new sample with Attribute 1 = 1.5



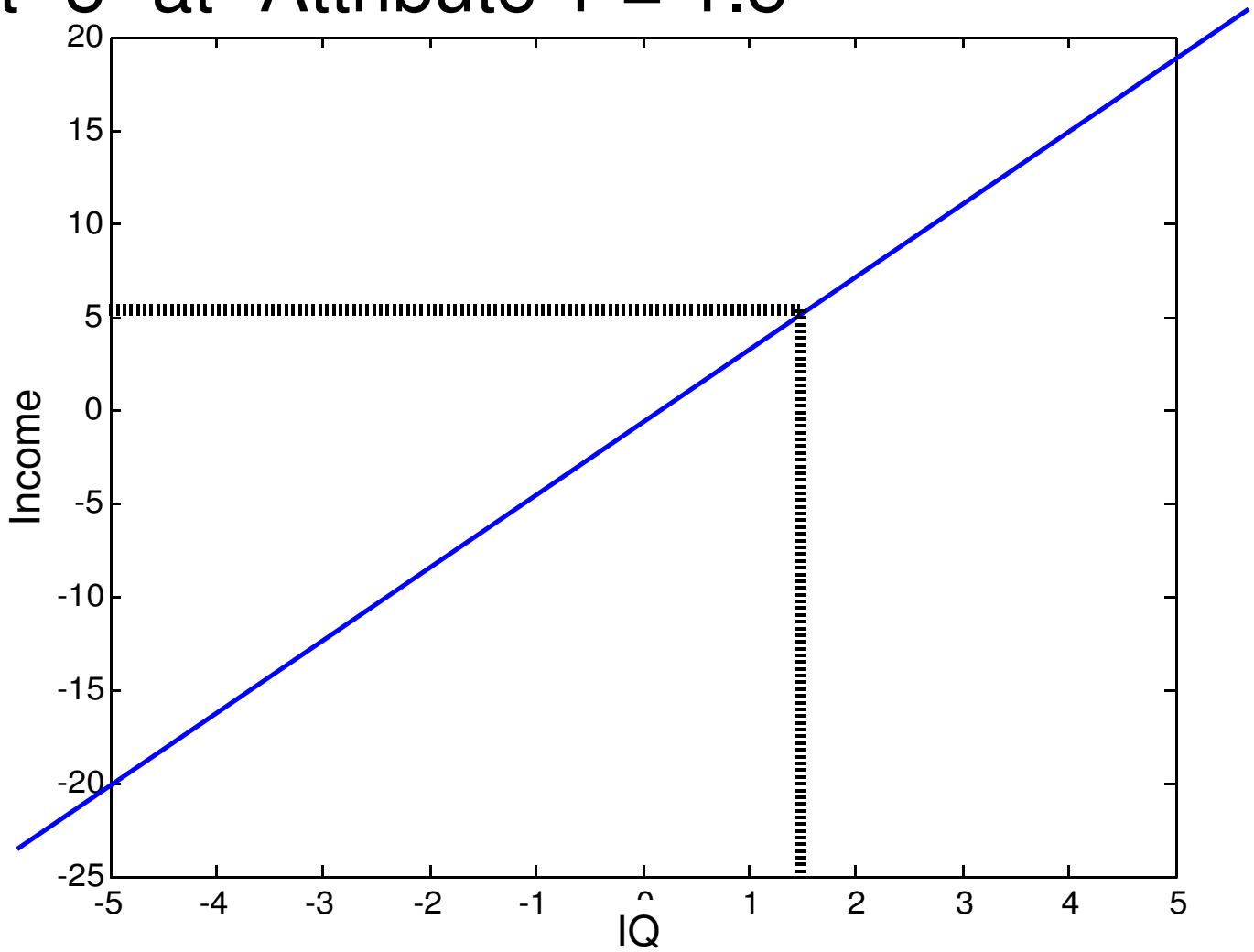
Step 1 Build a Model

- The Model (a straight line) is a more abstract description of the data

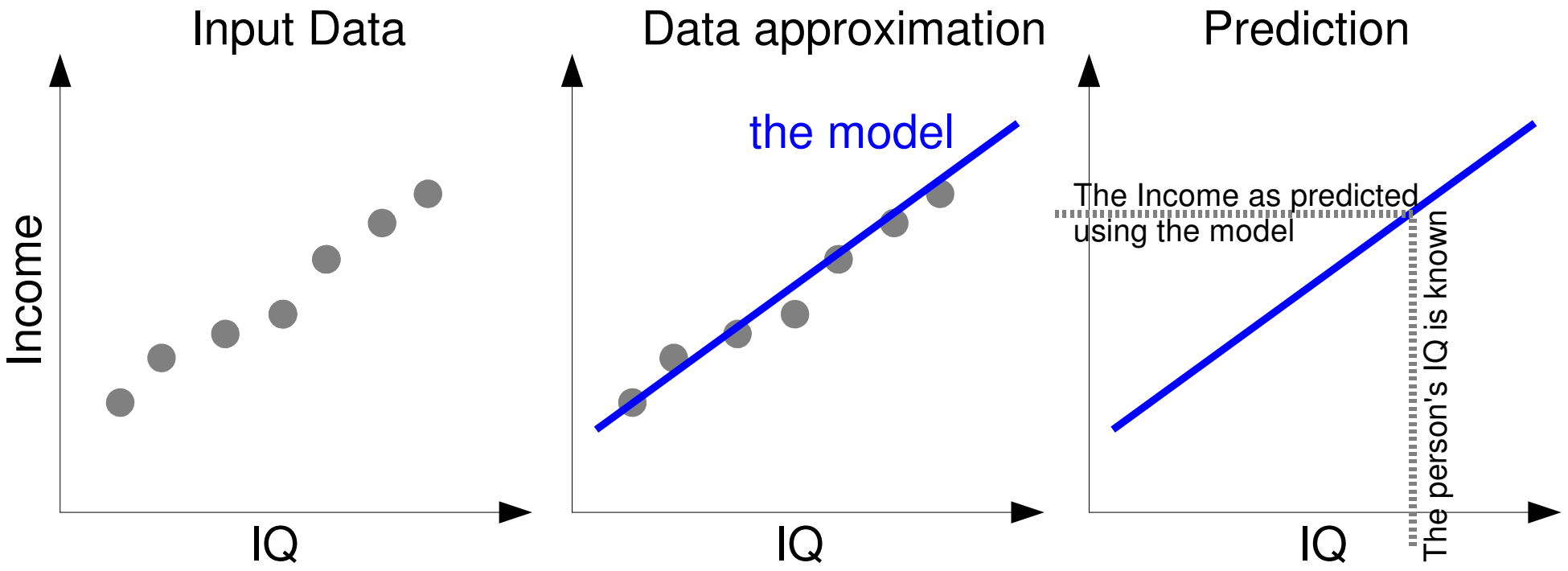


Step 2 Prediction using the Model

- The dependent variable taken the model gives about “5” at “Attribute 1 = 1.5”



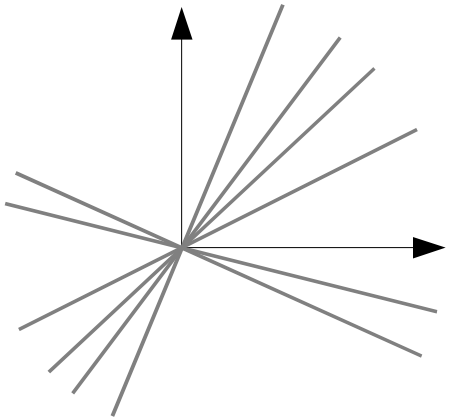
A non-abstract example



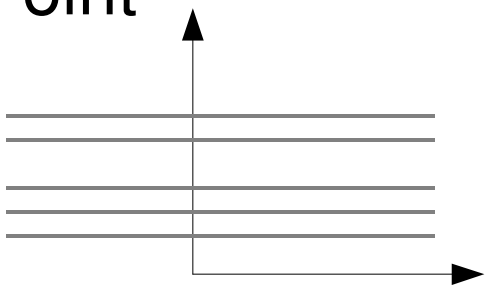
- The model is a line with two parameters
  - Ascent and Y-Intersection-Point

A) Decide about the model's freedom

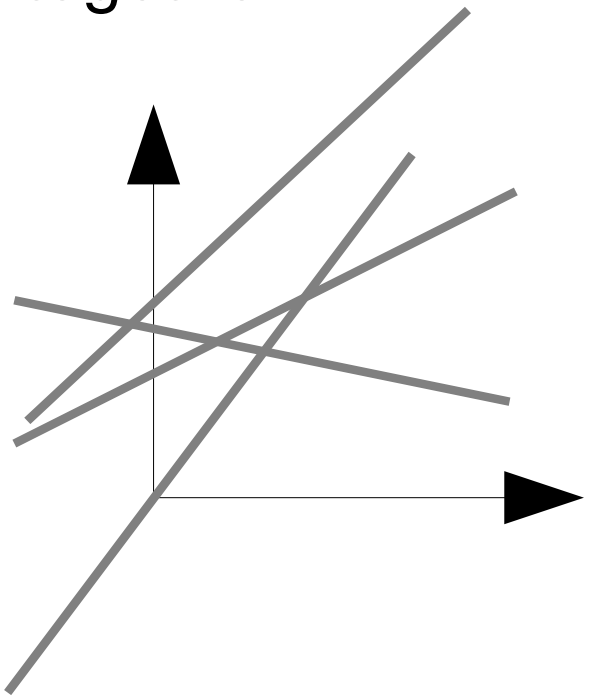
- Only one parameter
  - Ascent



  - null-Point

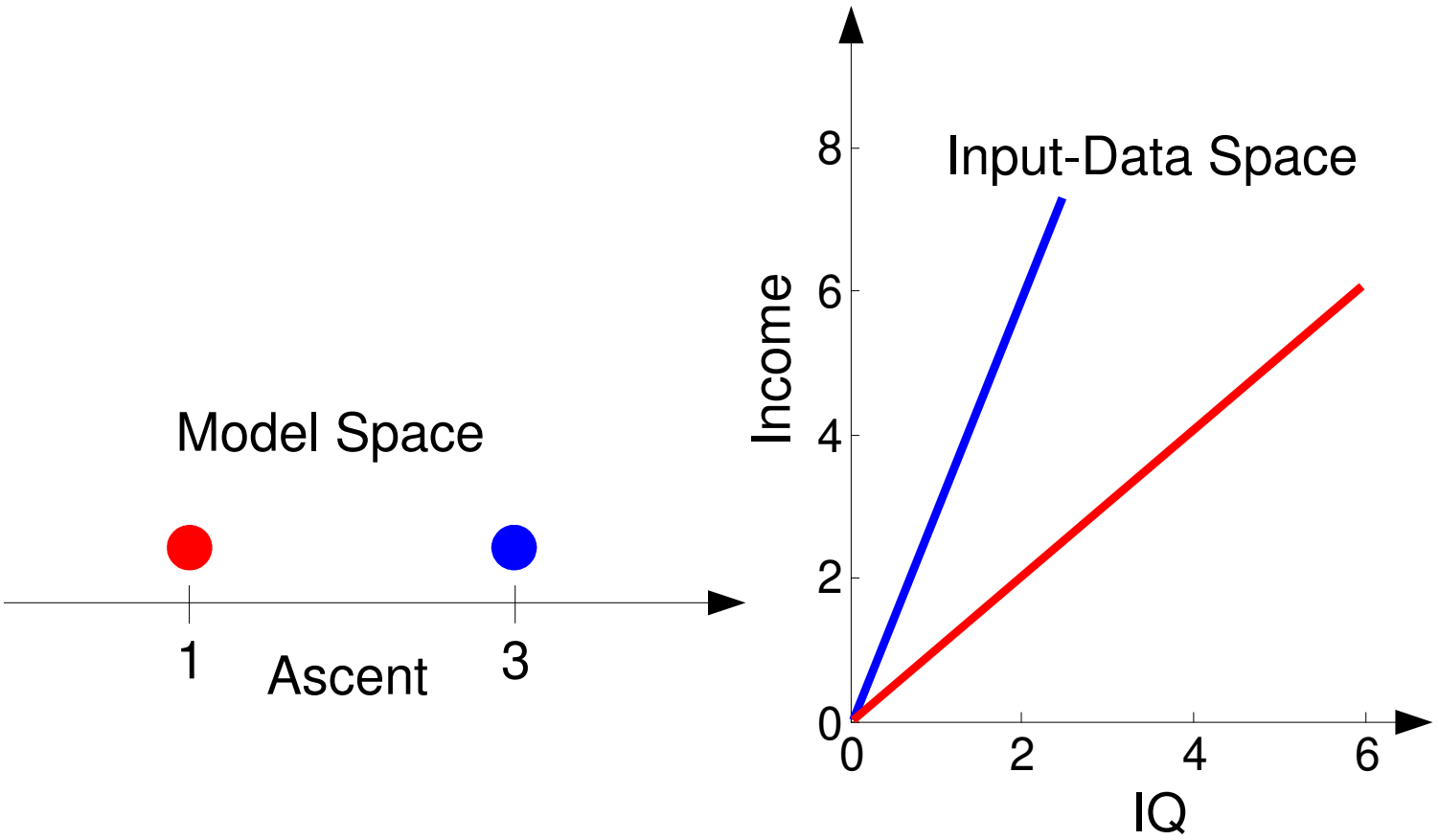


- Two parameters
  - Ascent and null-Point together



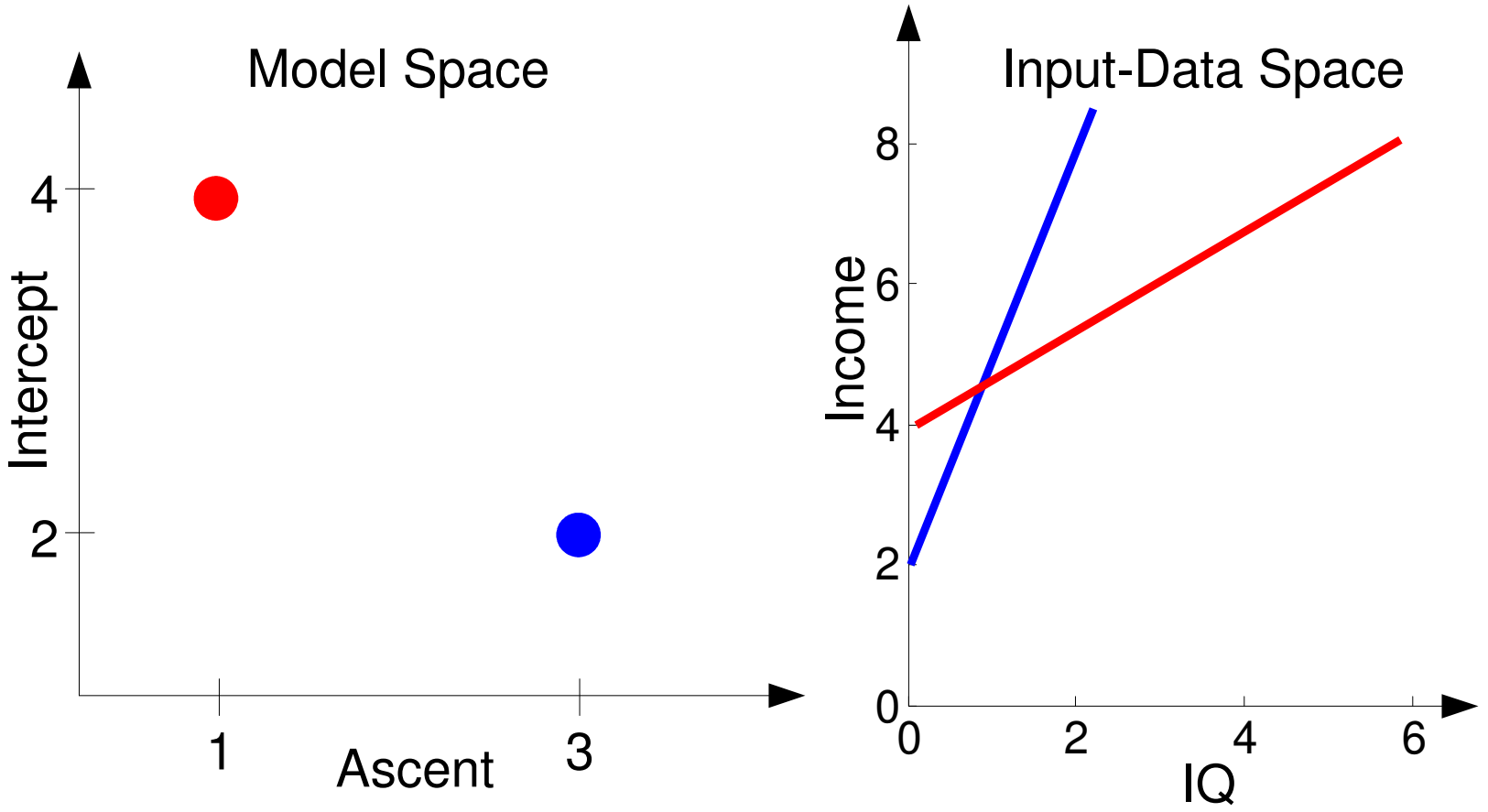
A Model with only one parameter

- One parameter (ascent of a line) means one degree of freedom



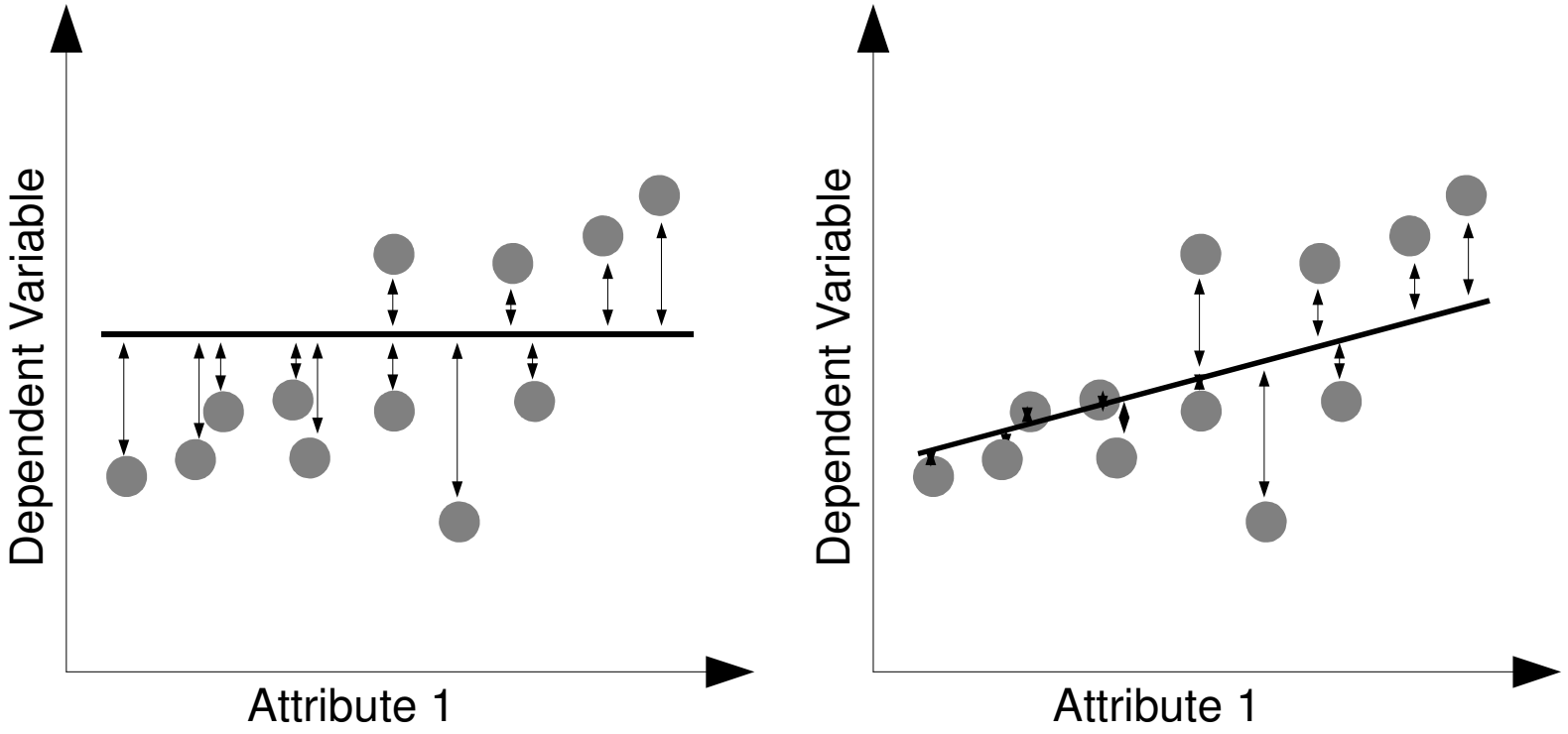
A Model with two parameters

- Two degrees of freedom (ascent and interception point)
- The model parameters span a 2D-model space

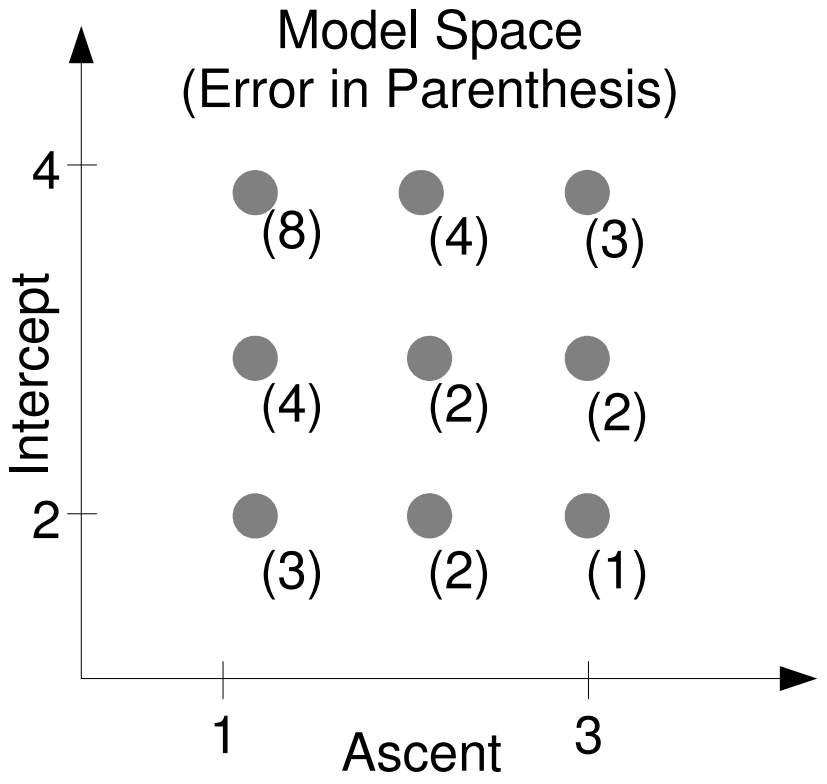


B) Define an error Function

- How well does the model fit the input data
- Sum of squared distances between model prediction and input data

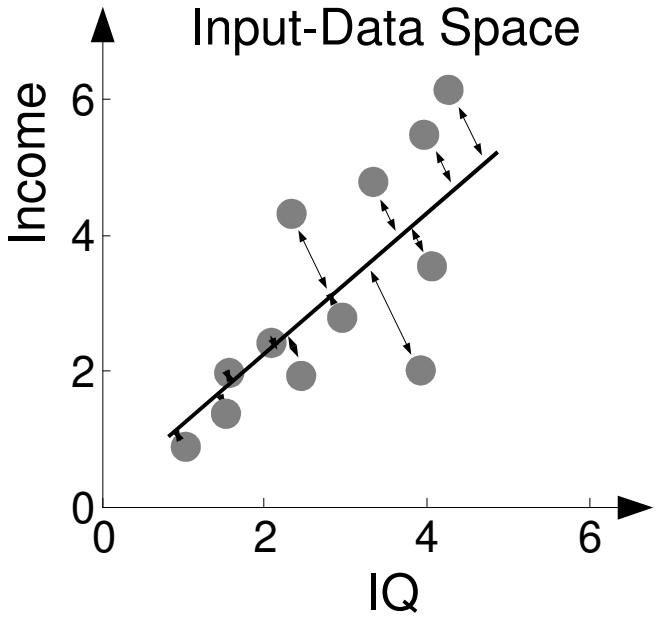


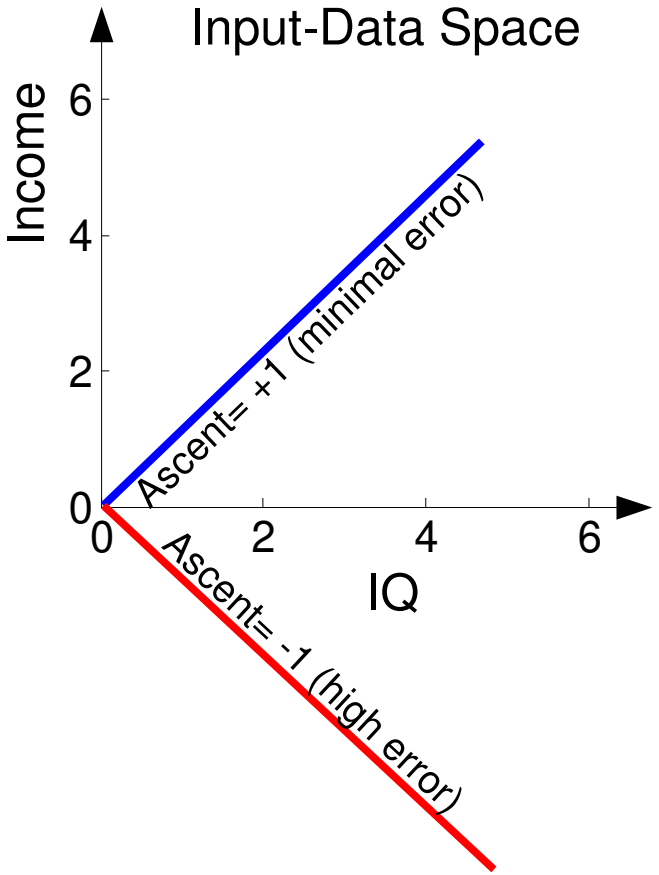
C) How to build the model automatically – scan the model space

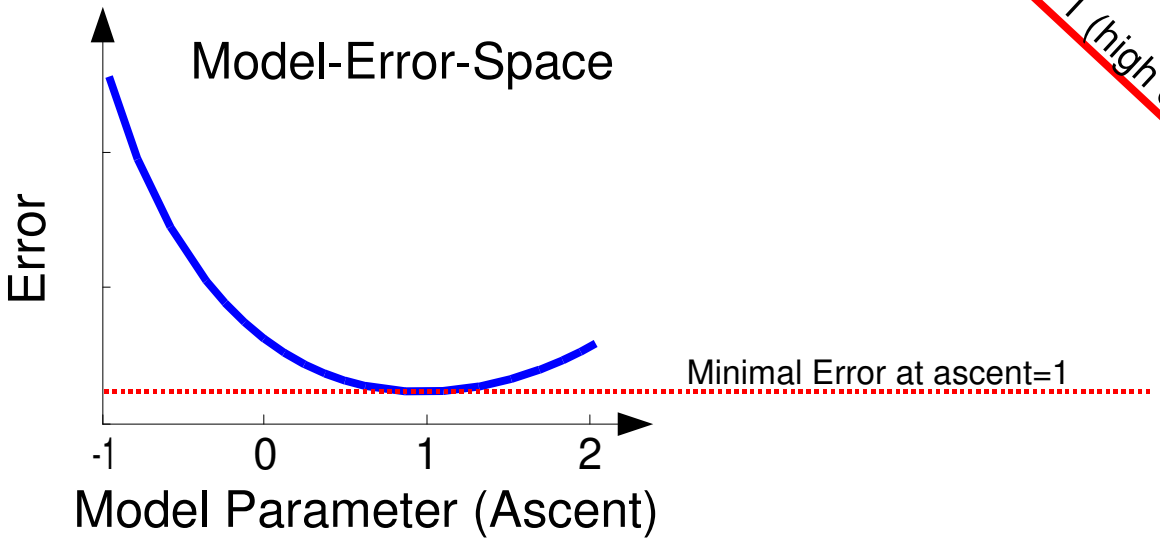


- The task is to find the point in the model space that optimally approximates the input data

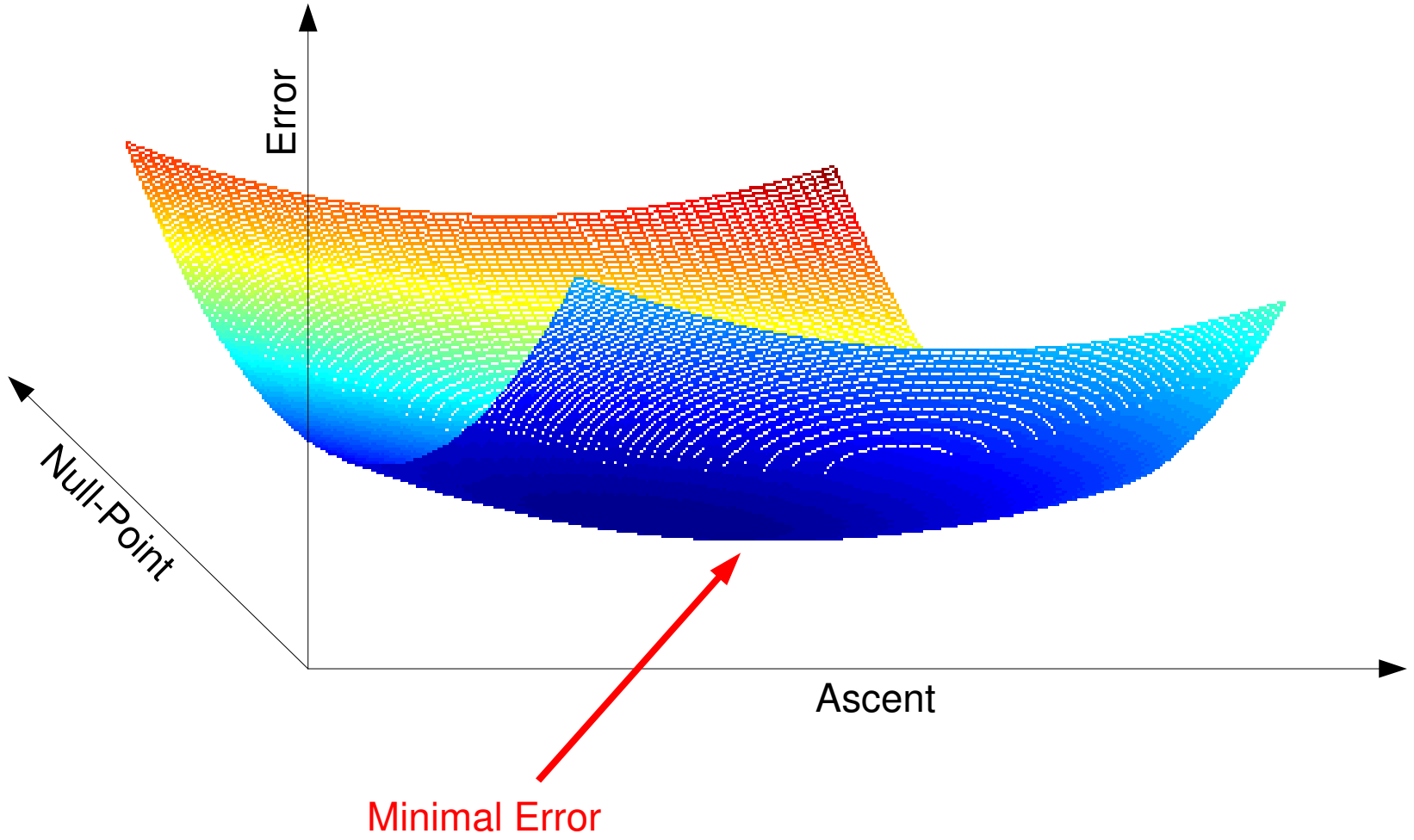
The model-error function





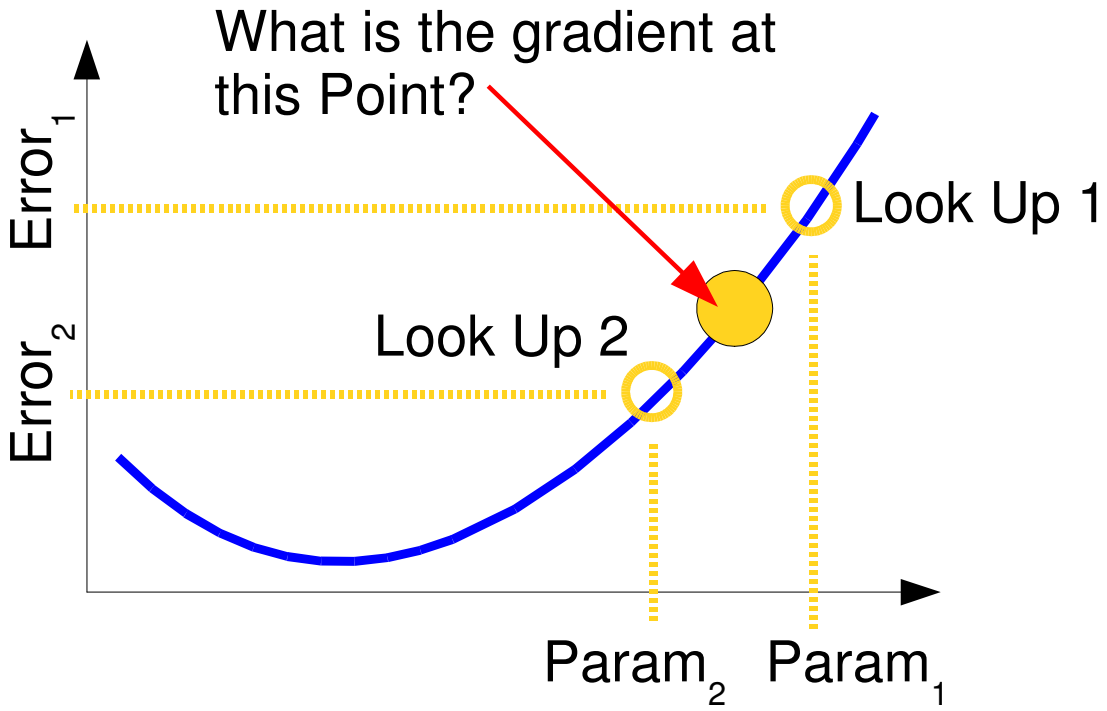


The model parameters and their mutual error function



A fast algorithm...

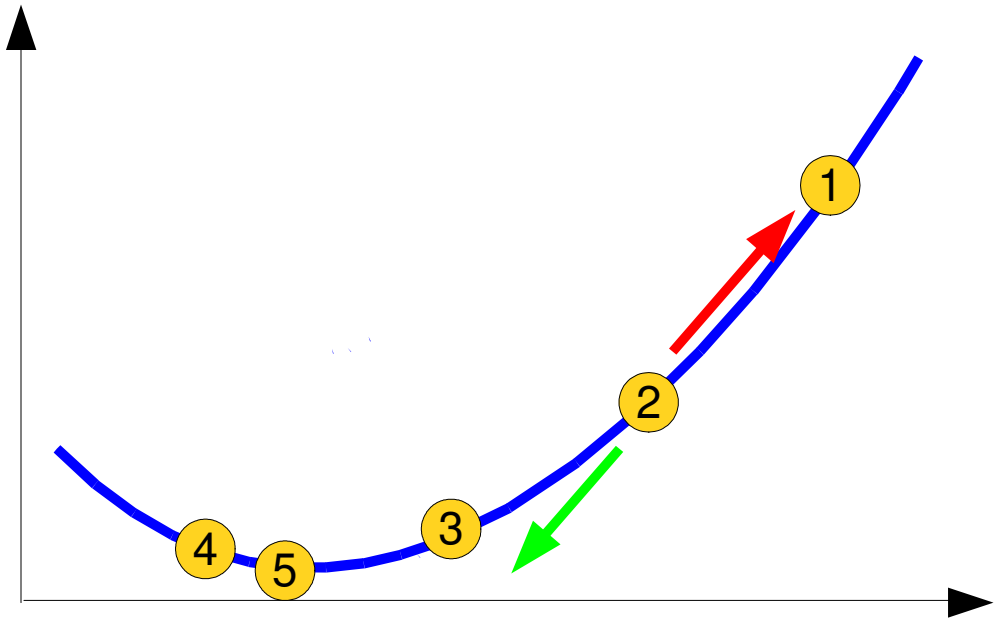
- ...for finding the minimal error of models fast
- Gradient Descent (gradient is the direction of steepest ascent)



$$= \frac{Error_2 - Error_1}{Param_2 - Param_1}$$

Gradient Descent...

- ...is walking in opposite direction of the gradient



- The walk distance is a function of the gradient ascent ...
- ...multiplied by a small factor (step size)
- What happens with small or big step sizes?

Gradient Descent

- Update function for model parameter x

$$x^{new} = x^{old} - \gamma \cdot \frac{\Delta E}{\Delta x}$$

Gradient approximated

  - “x” the model parameter
  - With E the Error of the Model given the input data
  - With $\Delta E/\Delta x$ the approximated gradient
  - And γ the step size (should be small ≈ 0.001)

Gradient Descent Practice

- Matlab functions in Material.zip/Regression
  - Create and show random Data data=generateData
  - Watch the error for a point in the search space error_Data_Model(data,ascent,interception)
  - Visualize the search space (watch the error at many points in the search space
  - [ascents,nullPoints,mat]=createSearchSpace(data)
  - mesh(ascents,nullPionts,mat)
  - See how gradient descent works (gradientDescent)
    - Try several initial positions for the model parameters
    - Try different stepSizes (be careful suggested is 0.001)

Gradient Descent Limits

- inappropriate StepSize -> strange effects
  - Oscillating behavior
  - Slowly progressing versus to fast and missing global minimum
  - Exploration versus exploitation
    - The stepSize can change after time (decreases)
- The final solution depends on initial position of model parameters (use heuristics)
  - Problems with local minima