

# **ORF307 – Optimization**

## **4. Least squares data fitting**

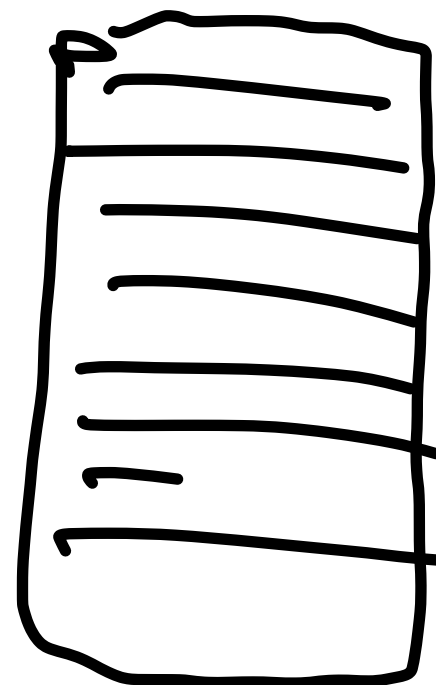
# Ed Forum

$$\|Ax - b\|^2$$

$$Ax = \sum x_i a_i \quad \Bigg| \quad r_i = (Ax - b)_i$$

- Why do we look at both rows and columns interpretation of least squares problems?
- How does the derivation of the least squares function gradient work?
- In some cases,  $(2mn^2 > 2mn)$  it never makes sense to run slower method. However, in the real world, may we always assume that  $n$  is so large (the only times when complexity likely matters) so our only goal is to reduce the exponent on  $n$  when we construct the code?

$$m \gg n$$



**Recap**

$$(Ax)^T = x^T A^T$$

# Calculus derivation in vector form

$$\rightarrow (Ax)^T(Ax) + (Ax)^T(-b) + (-b)^T Ax + b^T b$$

$$f(x) = \|Ax - b\|^2 = (Ax - b)^T (Ax - b) = x^T A^T Ax - 2(A^T b)^T x + b^T b$$

$$\nabla f(x^*) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x^*) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x^*) \end{bmatrix} = 2A^T Ax^* - 2A^T b = 2A^T (Ax^* - b) = 0$$

$n \times n$   
square  
linear system



**normal equations**

$$(A^T A)x^* = A^T b$$

$$\begin{aligned} & \underbrace{-2b^T Ax}_{-x^T A^T b - b^T Ax} \\ & = -b^T Ax - b^T Ax \end{aligned}$$

# Solving normal equations

$$(A^T A)x^* = A^T b$$

## Inversion

$$x^* = (A^T A)^{-1} A^T b$$



## Pseudo-inverse

$$A^\dagger = (A^T A)^{-1} A^T$$

## Factor-solve method

$A$  has linearly independent columns



$A^T A$  is **symmetric positive-definite**



## Cholesky factorization

$$A^T A = LL^T$$

# Optimal advertising

$m$  demographic groups  
we want to advertise to



$v^{\text{des}}$  is the  $m$ -vector  
of desired views/impressions

$n$  advertising channels  
(web publishers, radio, print, etc.)



$s$  is the  $n$ -vector  
of purchases

$m \times n$  matrix  $A$  gives  
demographic reach of channels



$A_{ij}$  is the number of views  
for group  $i$  and dollar spent  
on channel  $j$  (1000/\$)

## Views across demographic groups

$$v = As$$

### Goal

minimize  $\|As - v^{\text{des}}\|^2$

# Optimal advertising

## Reusing factorization on large example

$m = 100,000$  groups,  $n = 5,000$  channels

$$\text{minimize } \|As - v^{\text{des}}\|^2$$

### First solve

desired views  $v^{\text{des},1} = (10^3)\mathbf{1}$

1. Form linear system  $Mx = q$   
where  $M = A^T A$ ,  $q = A^T b$
2. Factor  $M = LL^T$
3. Solve  $LL^T x = q$

### Complexity

$$2mn^2$$

**Time:** 9 sec

### Second solve

desired views  $v^{\text{des},2} = 500\mathbf{1}$

1. Form  $q = A^T b$
2. Solve  $LL^T x = q$

### Complexity

$$2mn$$

**Time:** 0.37 sec

**Pseudoinverse**

**Time:** 263 sec

# Today's lecture

## Least squares data fitting

- Least squares model fitting
- Univariate regression
- Multivariate regression
- Validation
- Example



# Least squares model fitting

# Setup

We believe a scalar  $y$  and a  $n$ -vector are related by a model

$$y \approx f(x)$$

- $x$  is the *independent variable* or *feature vector*
- $y$  is the *outcome* or *response variable*
- $f : \mathbf{R}^n \rightarrow \mathbf{R}$  maps  $x$  to  $y$

# Setup

We believe a scalar  $y$  and a  $n$ -vector are related by a model

$$y \approx f(x)$$

- $x$  is the *independent variable* or *feature vector*
- $y$  is the *outcome* or *response variable*
- $f : \mathbf{R}^n \rightarrow \mathbf{R}$  maps  $x$  to  $y$

**We don't know  $f$  and we want to estimate it from data**

# Data

All we have is data

$n$ -vectors  $x^{(1)}, \dots, x^{(N)}$ , and scalars  $y^{(1)}, \dots, y^{(N)}$

also called *observations, examples, samples* or *measurements*.

$(x^{(i)}, y^{(i)})$  is the  $i$ th data pair

$x_j^{(i)}$  is the  $j$ th component of  $i$ th data point  $x^{(i)}$

# Model

Guess a model  $\hat{f} : \mathbf{R}^n \rightarrow \mathbf{R}$  to approximate  $f$

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

- $f_i : \mathbf{R}^n \rightarrow \mathbf{R}$  are *feature mappings* or *basis functions*
- $\theta_i$  are *model parameters* to choose
- $\hat{y}^{(i)} = \hat{f}(x^{(i)})$  is the model's *prediction* for  $y^{(i)}$
- Goal:  $\hat{y}^{(i)} \approx y^{(i)}$  (consistent with observed data)

# Least squares data fitting

**Prediction error (residual)**  $r^{(i)} = y^{(i)} - \hat{y}^{(i)}$

**Goal:** choose model parameters  $\theta_i$  to minimize mean squared error (MSE)

$$\frac{(r^{(1)})^2 + \dots + (r^{(N)})^2}{N}$$

# Least squares data fitting

**Prediction error (residual)**

$$r^{(i)} = y^{(i)} - \hat{y}^{(i)}$$

**Goal:** choose model parameters  $\theta_i$  to minimize mean squared error (MSE)

$$\frac{(r^{(1)})^2 + \dots + (r^{(N)})^2}{N}$$



**This can be formulated  
as a least squares problem**

# Least squares data fitting

**Prediction error (residual)**

$$r^{(i)} = y^{(i)} - \hat{y}^{(i)}$$

**Goal:** choose model parameters  $\theta_i$  to minimize mean squared error (MSE)

$$\frac{(r^{(1)})^2 + \dots + (r^{(N)})^2}{N}$$



**This can be formulated as a least squares problem**

**Note.** we sometimes compute the root mean squared error  $\text{RMS} = \sqrt{\text{MSE}}$  because it has the same units as  $y^{(i)}$



# Least squares data fitting

## Vector form

Express problems with  $N$ -vectors

- $y^d = (y^{(1)}, \dots, y^{(N)})$ , vector of outcomes
- $\hat{y}^d = (\hat{y}^{(1)}, \dots, \hat{y}^{(N)})$ , vector of predictions
- $r^d = (r^{(1)}, \dots, r^{(N)})$ , vector of residuals

# Least squares data fitting

## Vector form

Express problems with  $N$ -vectors

- $y^d = (y^{(1)}, \dots, y^{(N)})$ , vector of outcomes
- $\hat{y}^d = (\hat{y}^{(1)}, \dots, \hat{y}^{(N)})$ , vector of predictions
- $r^d = (r^{(1)}, \dots, r^{(N)})$ , vector of residuals



**Goal**  
minimize  $\|r^d\|^2$

# Least squares data fitting

## Vector form

Express problems with  $N$ -vectors

- $y^d = (y^{(1)}, \dots, y^{(N)})$ , vector of outcomes
- $\hat{y}^d = (\hat{y}^{(1)}, \dots, \hat{y}^{(N)})$ , vector of predictions
- $r^d = (r^{(1)}, \dots, r^{(N)})$ , vector of residuals



**Goal**

minimize  $\|r^d\|^2$

We can write  $\hat{y}^{(i)} = \hat{f}(x^{(i)})$  in terms of parameters  $\theta_i$

$$\hat{y}^{(i)} = A_{i1}\theta_1 + \dots + A_{ip}\theta_p, \quad A_{ij} = f_j(x^{(i)})$$



$$\hat{y}^d = A\theta$$

# Least squares data fitting

## Vector form

Express problems with  $N$ -vectors

- $y^d = (y^{(1)}, \dots, y^{(N)})$ , vector of outcomes
- $\hat{y}^d = (\hat{y}^{(1)}, \dots, \hat{y}^{(N)})$ , vector of predictions
- $r^d = (r^{(1)}, \dots, r^{(N)})$ , vector of residuals



**Goal**

minimize  $\|r^d\|^2$

We can write  $\hat{y}^{(i)} = \hat{f}(x^{(i)})$  in terms of parameters  $\theta_i$

$$\hat{y}^{(i)} = A_{i1}\theta_1 + \dots + A_{ip}\theta_p, \quad A_{ij} = f_j(x^{(i)})$$



$$\hat{y}^d = A\theta$$

## Least squares problem

$$\text{minimize } \|r^d\|^2 = \|y^d - \hat{y}^d\|^2 = \|y^d - A\theta\|^2 = \|A\theta - y^d\|^2$$

# Least squares data fitting

$$\theta \in \mathbb{R}^p \quad A \in \mathbb{R}^{n \times p}$$

## Vector form

Express problems with  $N$ -vectors

- $y^d = (y^{(1)}, \dots, y^{(N)})$ , vector of outcomes
- $\hat{y}^d = (\hat{y}^{(1)}, \dots, \hat{y}^{(N)})$ , vector of predictions
- $r^d = (r^{(1)}, \dots, r^{(N)})$ , vector of residuals



**Goal**  
minimize  $\|r^d\|^2$

We can write  $\hat{y}^{(i)} = \hat{f}(x^{(i)})$  in terms of parameters  $\theta_i$

$$\hat{y}^{(i)} = A_{i1}\theta_1 + \dots + A_{ip}\theta_p, \quad A_{ij} = f_j(x^{(i)}) \quad \longrightarrow \quad \hat{y}^d = A\theta$$

## Least squares problem

$$\text{minimize } \|r^d\|^2 = \|y^d - \hat{y}^d\|^2 = \|y^d - A\theta\|^2 = \|A\theta - y^d\|^2$$

## Solution

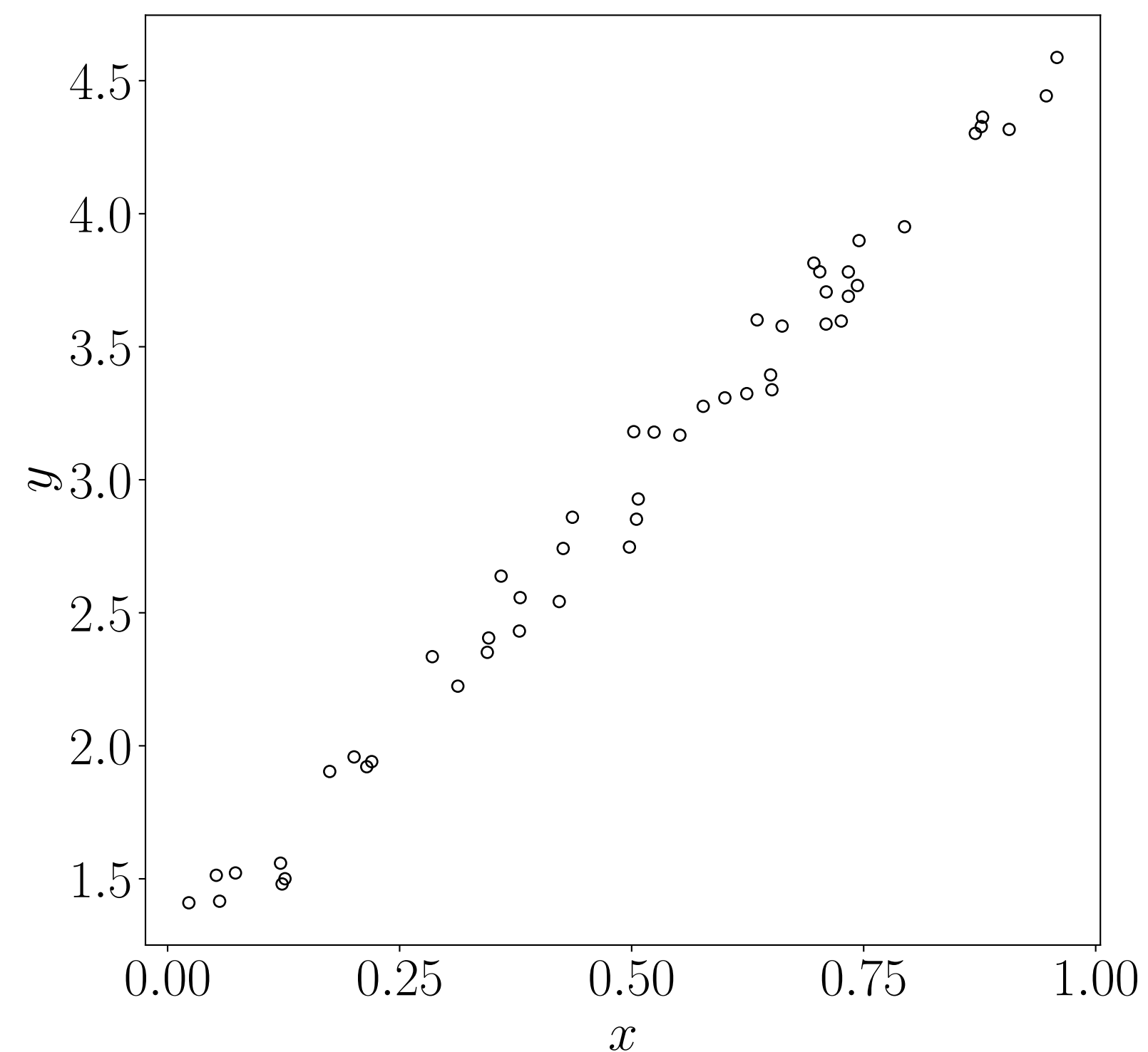
$$(A^T A)\theta^* = A^T y^d$$

# Univariate fitting

# Fitting univariate functions

We seek to approximate a function  $f : \mathbf{R} \rightarrow \mathbf{R}$  ( $n = 1$ )

**Data points**  $(x^{(i)}, y^{(i)})$



# Straight line fit

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x$$

- Parameters:  $\theta = (\theta_1, \theta_2)$  ( $p = 2$ )
- Features:  $f_1(x) = 1, f_2(x) = x$



# Straight line fit

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x$$

- Parameters:  $\theta = (\theta_1, \theta_2)$  ( $p = 2$ )
- Features:  $f_1(x) = 1$ ,  $f_2(x) = x$

## Least squares data

$$A = \begin{bmatrix} 1 & x^{(1)} \\ 1 & x^{(2)} \\ \vdots & \vdots \\ 1 & x^{(N)} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & x^d \end{bmatrix}$$

# Straight line fit

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x$$

- Parameters:  $\theta = (\theta_1, \theta_2)$  ( $p = 2$ )
- Features:  $f_1(x) = 1$ ,  $f_2(x) = x$

## Least squares data

$$A = \begin{bmatrix} 1 & x^{(1)} \\ 1 & x^{(2)} \\ \vdots & \vdots \\ 1 & x^N \end{bmatrix} = \begin{bmatrix} \mathbf{1} & x^d \end{bmatrix}$$

## Goal

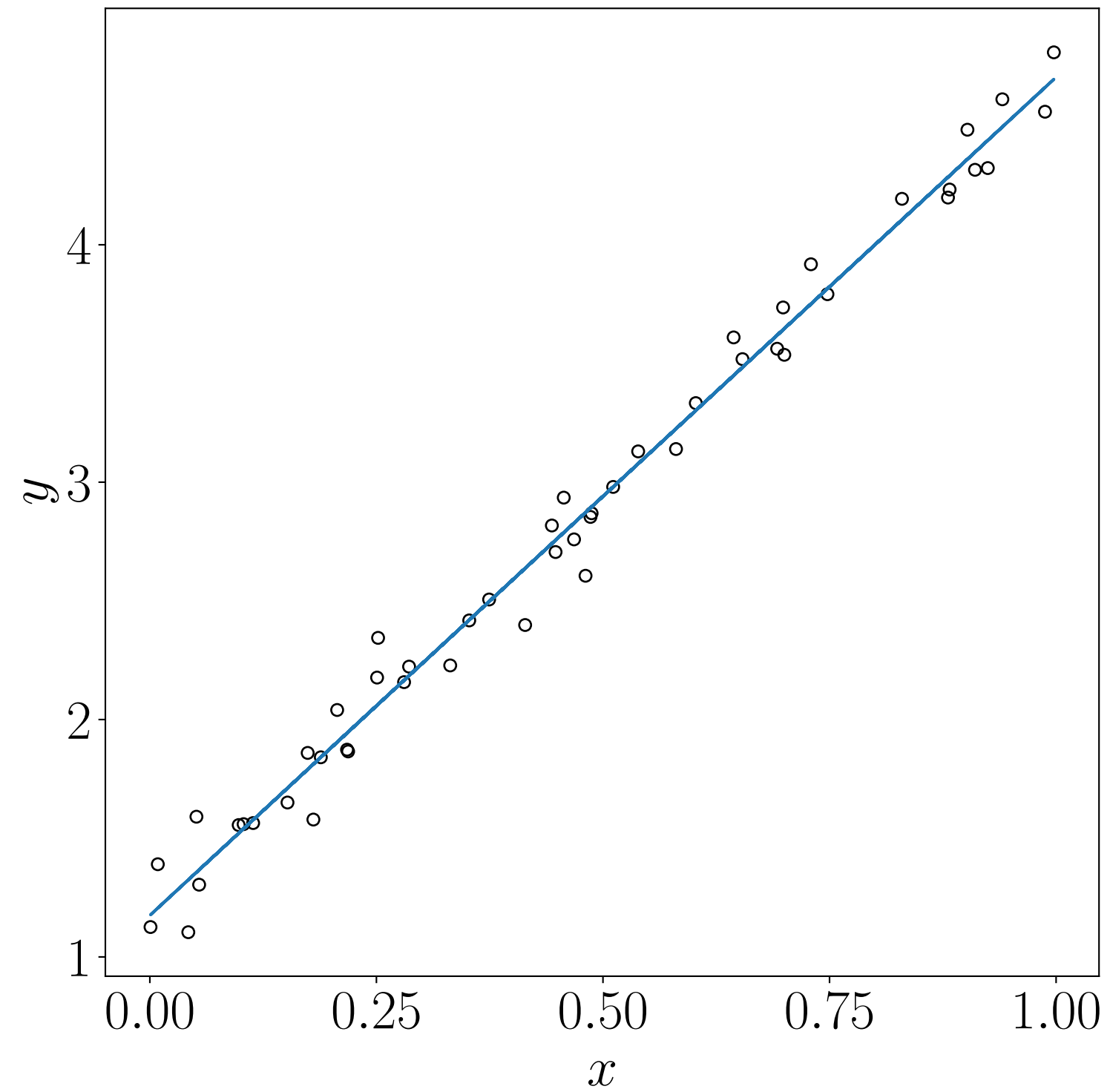
$$\text{minimize } \|r^d\|^2 = \|A\theta - y^d\|^2$$

# Straight line fit

## Example

$$\theta^* = (1.75, 3.53)$$

$$\hat{f}(x) = \theta_1 + \theta_2 x$$



# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

$r_t^{\text{mkt}}$  is the return  
of the *whole market*  
at time  $t$

# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

$r_t^{\text{mkt}}$  is the return  
of the *whole market*  
at time  $t$

**individual asset returns**

$$y^d = (r_1^{\text{ind}}, \dots, r_T^{\text{ind}})$$

$r_t^{\text{ind}}$  is the return  
of an *individual asset*  
at time  $t$

# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

**individual asset returns**

$$y^d = (r_1^{\text{ind}}, \dots, r_T^{\text{ind}})$$

$r_t^{\text{mkt}}$  is the return  
of the *whole market*  
at time  $t$

$r_t^{\text{ind}}$  is the return  
of an *individual asset*  
at time  $t$



**Goal**  
predict individual  
asset return  
from whole market  
return

# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

**individual asset returns**

$$y^d = (r_1^{\text{ind}}, \dots, r_T^{\text{ind}})$$

$r_t^{\text{mkt}}$  is the return  
of the *whole market*  
at time  $t$

$r_t^{\text{ind}}$  is the return  
of an *individual asset*  
at time  $t$



**Goal**  
predict individual  
asset return  
from whole market  
return

**Linear model**

$$\hat{y} = (r^{\text{rf}} + \alpha) + \beta(x - \mu^{\text{mkt}})$$

- $\mu^{\text{mkt}}$  is the average market return over period
- $r^{\text{rf}}$  is the risk-free interest rate over the period

# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

$r_t^{\text{mkt}}$  is the return of the *whole market* at time  $t$

**individual asset returns**

$$y^d = (r_1^{\text{ind}}, \dots, r_T^{\text{ind}})$$

$r_t^{\text{ind}}$  is the return of an *individual asset* at time  $t$



**Goal**  
predict individual asset return from whole market return

**Linear model**

$$\hat{y} = (r^{\text{rf}} + \alpha) + \beta(x - \mu^{\text{mkt}})$$

asset  $\alpha$   
average asset return above  $r^{\text{rf}}$

- $\mu^{\text{mkt}}$  is the average market return over period
- $r^{\text{rf}}$  is the risk-free interest rate over the period



# Asset $\alpha$ and $\beta$ in finance

**whole market returns**

$$x^d = (r_1^{\text{mkt}}, \dots, r_T^{\text{mkt}})$$

$r_t^{\text{mkt}}$  is the return of the *whole market* at time  $t$

**individual asset returns**

$$y^d = (r_1^{\text{ind}}, \dots, r_T^{\text{ind}})$$

$r_t^{\text{ind}}$  is the return of an *individual asset* at time  $t$



**Goal**  
predict individual asset return from whole market return

**Linear model**

$$\hat{y} = (r^{\text{rf}} + \alpha) + \beta(x - \mu^{\text{mkt}}) = \overbrace{r^{\text{rf}} + \alpha}^{\mathcal{J}_1} - \beta \mu^{\text{mkt}} + \beta x \quad \downarrow \mathcal{J}_2$$

asset  $\alpha$   
average asset return above  $r^{\text{rf}}$

asset  $\beta$   
relates market return fluctuations to asset return

- $\mu^{\text{mkt}}$  is the average market return over period
- $r^{\text{rf}}$  is the risk-free interest rate over the period

# Time series trend

$y^{(i)}$  is the value of quantity at time  $x^{(i)} = i$

$y^d = (y^{(1)}, \dots, y^{(N)})$  is the *time series*

# Time series trend

$y^{(i)}$  is the value of quantity at time  $x^{(i)} = i$

$y^d = (y^{(1)}, \dots, y^{(N)})$  is the *time series*

## Model (trend line)

$$\hat{y}^{(i)} = \theta_1 + \theta_2 i, \quad i = 1, \dots, N$$

# Time series trend

$y^{(i)}$  is the value of quantity at time  $x^{(i)} = i$

$y^d = (y^{(1)}, \dots, y^{(N)})$  is the *time series*

## Model (trend line)

$$\hat{y}^{(i)} = \theta_1 + \theta_2 i, \quad i = 1, \dots, N$$

↑  
trend  
coefficient

# Time series trend

$y^{(i)}$  is the value of quantity at time  $x^{(i)} = i$

$y^d = (y^{(1)}, \dots, y^{(N)})$  is the *time series*

## Model (trend line)

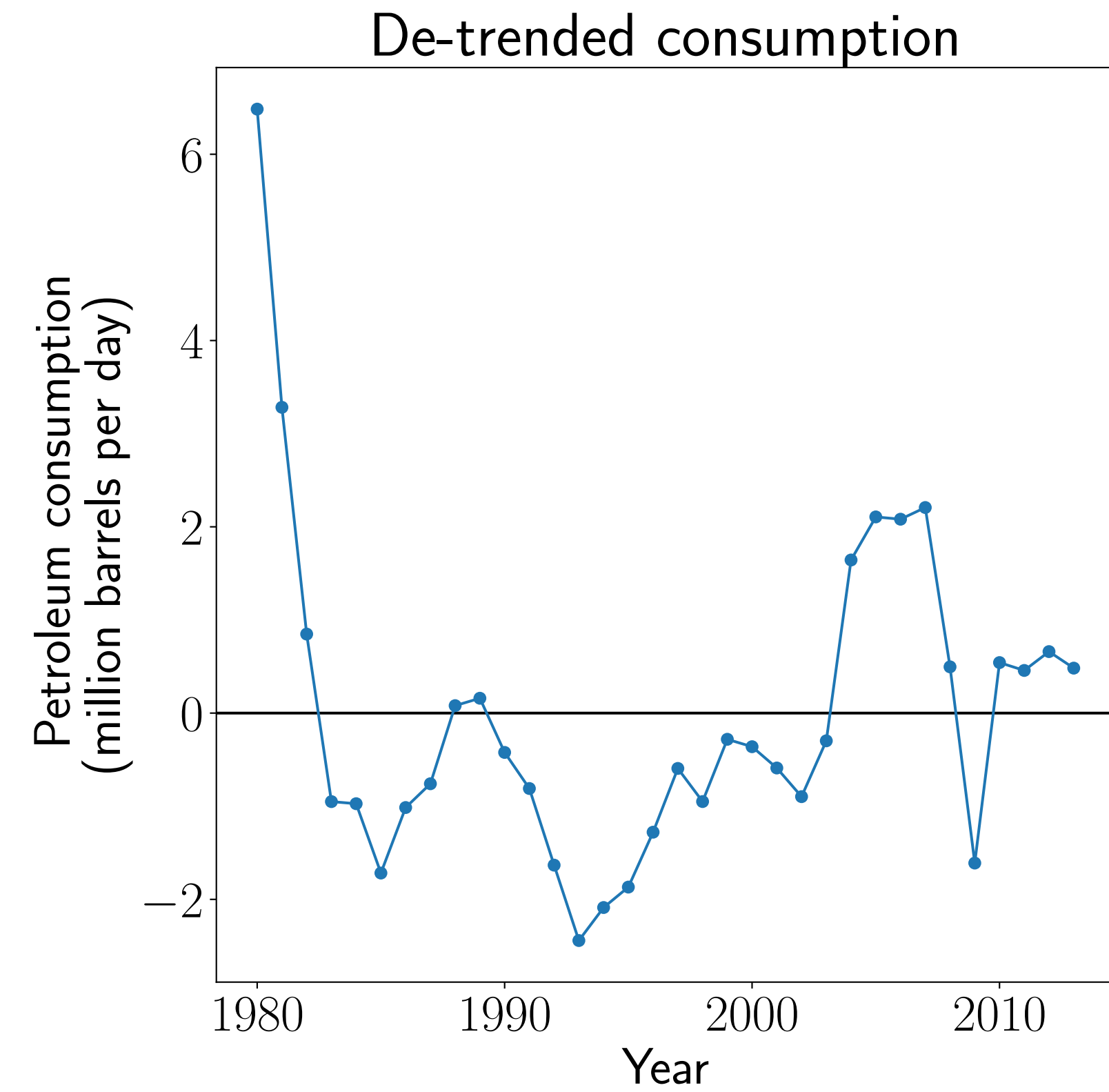
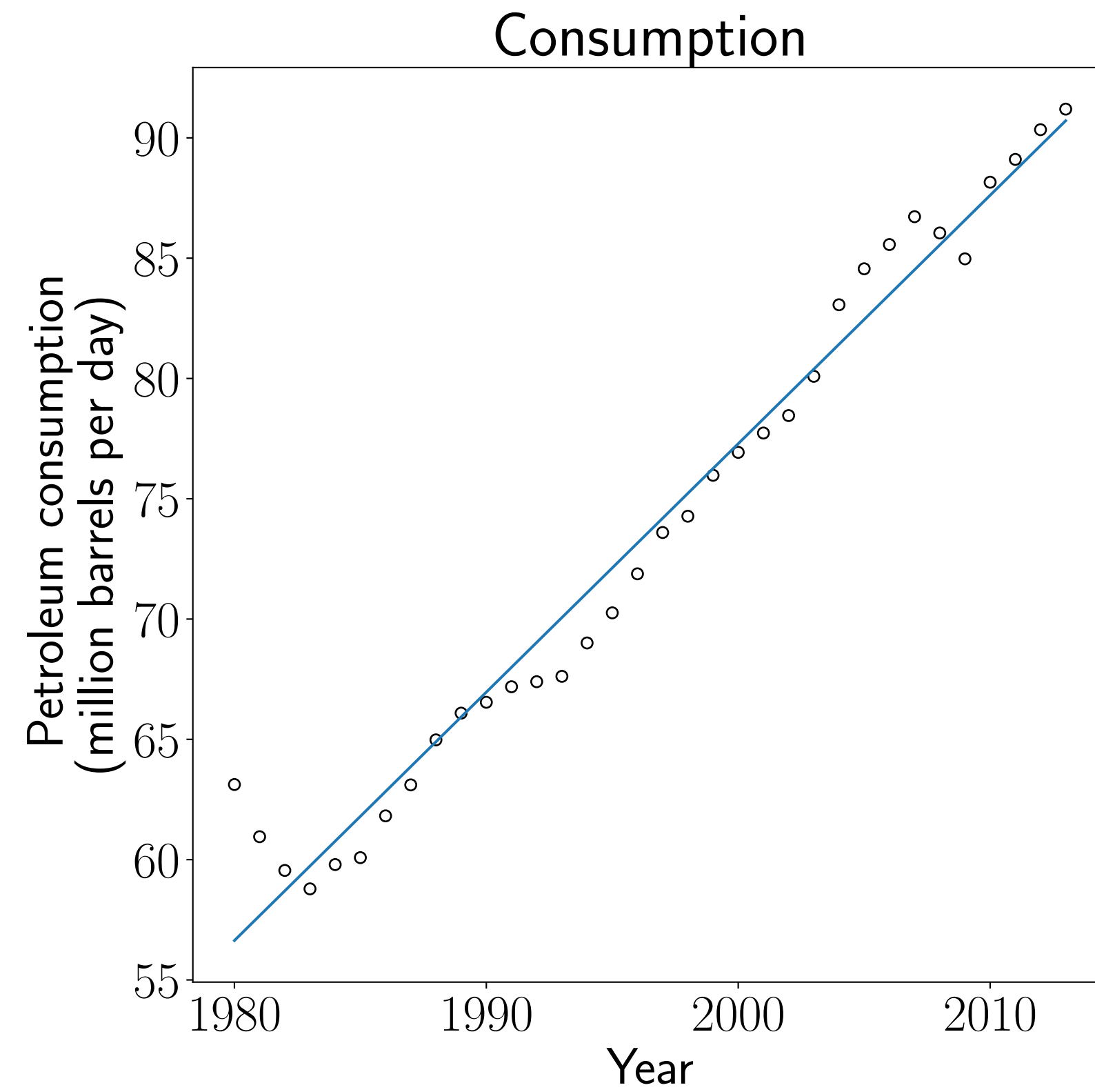
$$\hat{y}^{(i)} = \theta_1 + \theta_2 i, \quad i = 1, \dots, N$$

↑  
trend  
coefficient

$y^d - \hat{y}^d$  is the *de-trended time series*

# Time series trend

## Petroleum consumption



# Polynomial fit

## Features

$$f_i(x) = x^{i-1}, \quad i = 1, \dots, p$$

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x + \dots + \theta_p x^{p-1}$$

degree at most  
 $p - 1$

## Notation remark

$x^i$  means scalar  
to  $i$ th power

$x^{(i)}$  means  
 $i$ th data point

# Polynomial fit

## Features

$$f_i(x) = x^{i-1}, \quad i = 1, \dots, p$$

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x + \dots + \theta_p x^{p-1}$$

degree at most  
 $p - 1$

## Notation remark

$x^i$  means scalar  
to  $i$ th power

$x^{(i)}$  means  
 $i$ th data point

## Least squares data

*Vandermonde matrix*

$$A = \begin{bmatrix} 1 & x^{(1)} & \dots & (x^{(1)})^{p-1} \\ 1 & x^{(2)} & \dots & (x^{(2)})^{p-1} \\ \vdots & \vdots & & \vdots \\ 1 & x^{(N)} & \dots & (x^{(N)})^{p-1} \end{bmatrix}$$



# Polynomial fit

## Features

$$f_i(x) = x^{i-1}, \quad i = 1, \dots, p$$

## Model

$$\hat{f}(x) = \theta_1 + \theta_2 x + \dots + \theta_p x^{p-1}$$

degree at most  
 $p - 1$

## Notation remark

$x^i$  means scalar  
to  $i$ th power

$x^{(i)}$  means  
 $i$ th data point

## Least squares data *Vandermonde matrix*

$$A = \begin{bmatrix} 1 & x^{(1)} & \dots & (x^{(1)})^{p-1} \\ 1 & x^{(2)} & \dots & (x^{(2)})^{p-1} \\ \vdots & \vdots & & \vdots \\ 1 & x^{(N)} & \dots & (x^{(N)})^{p-1} \end{bmatrix}$$

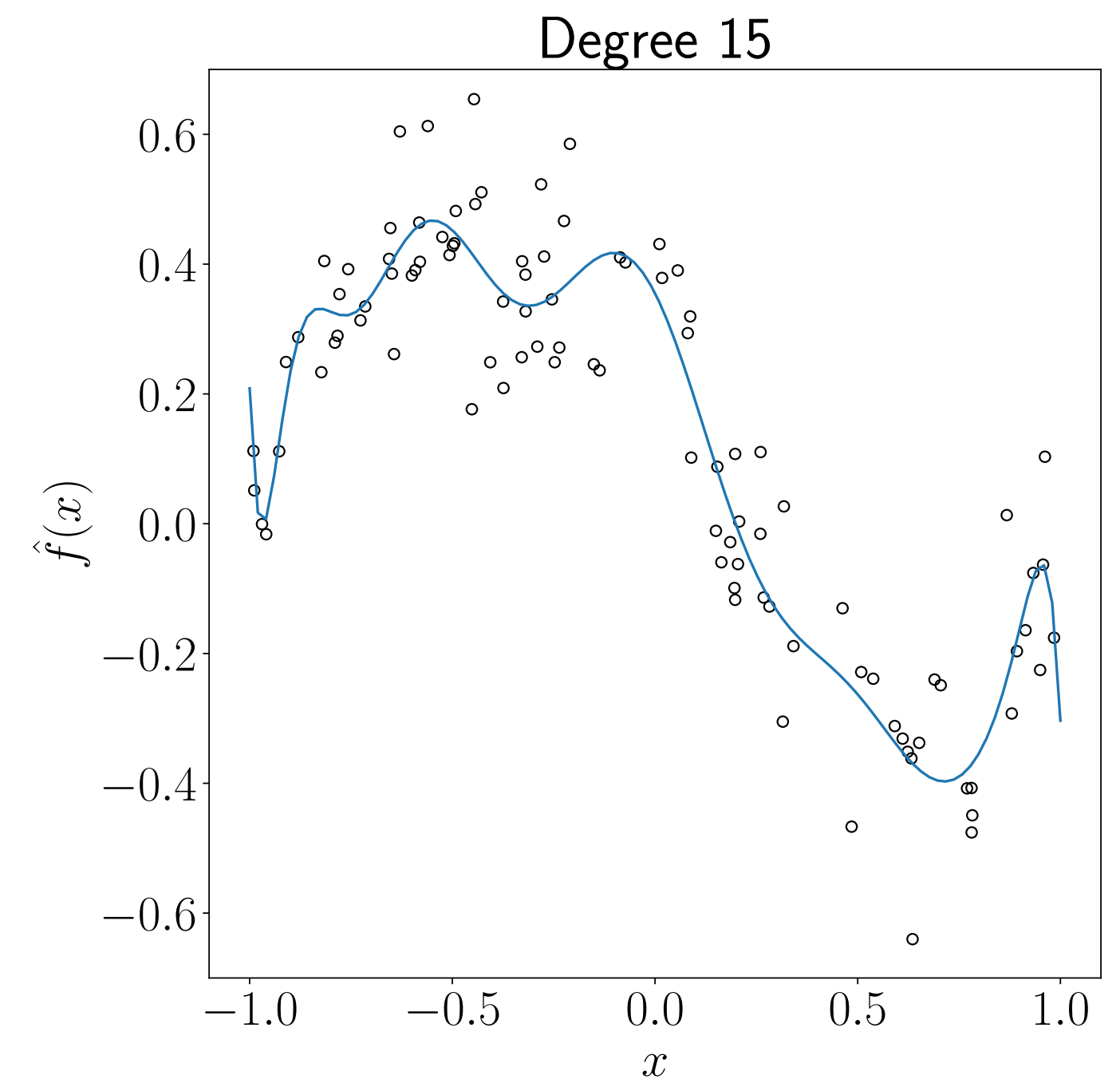
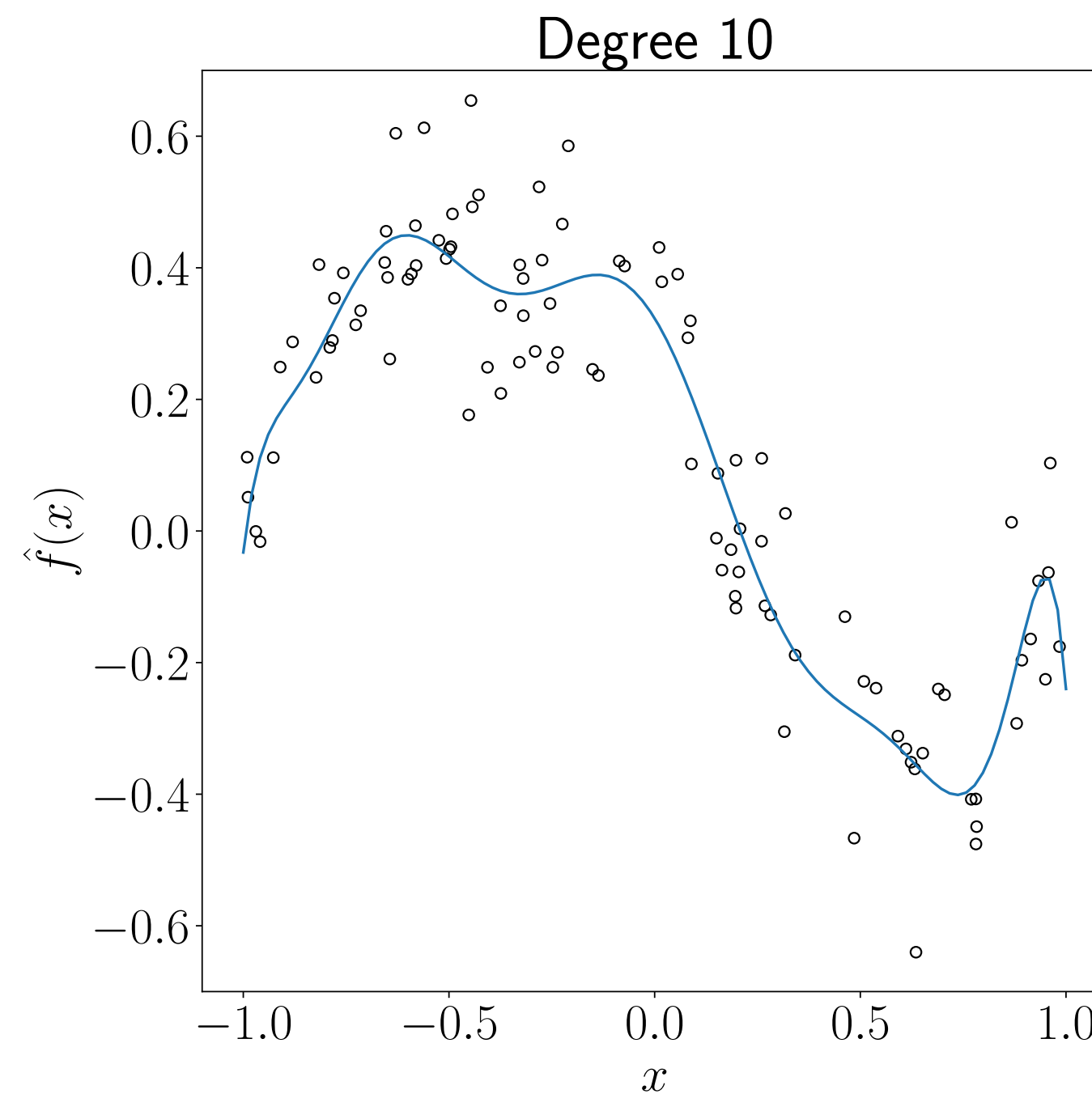
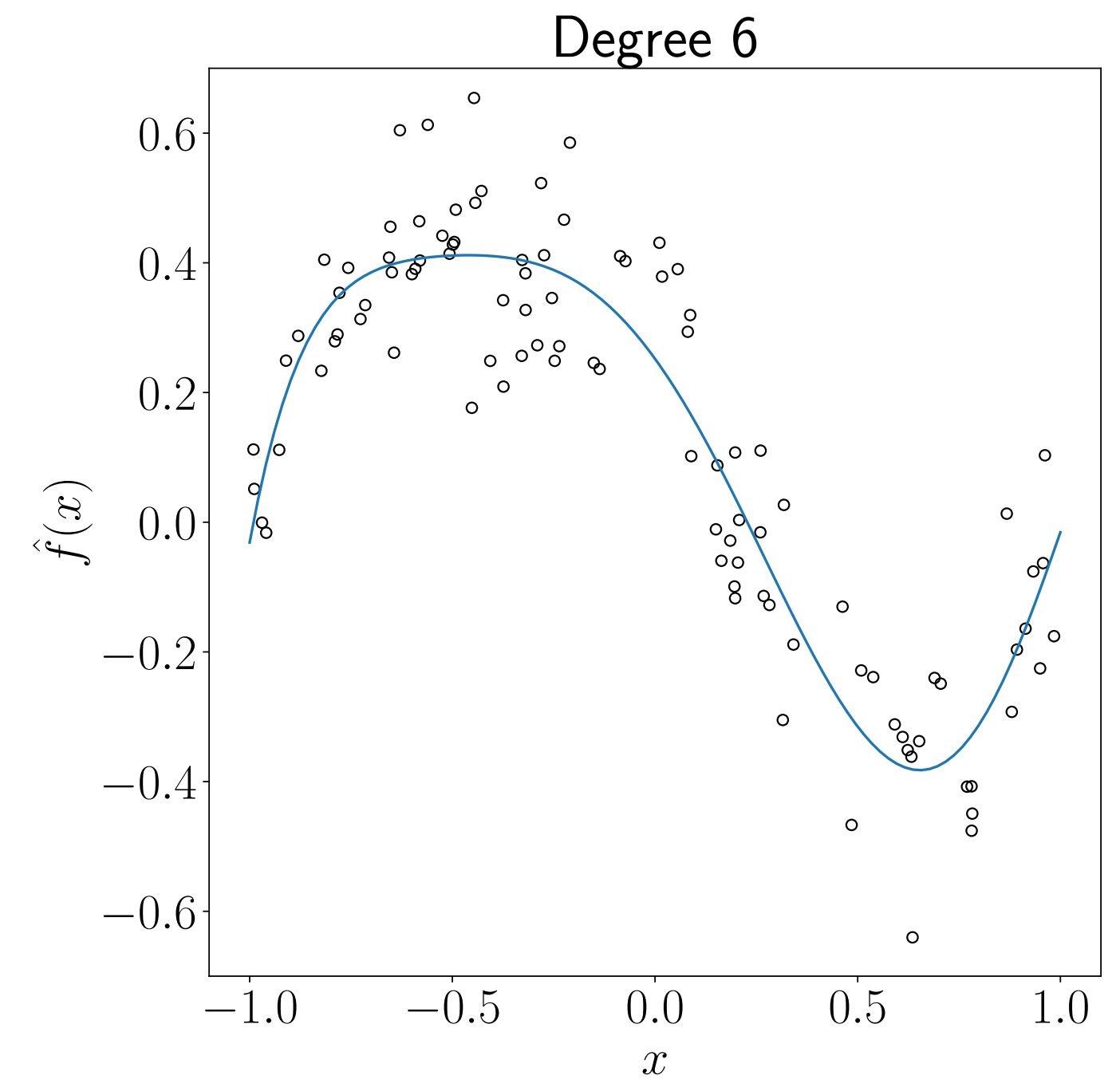
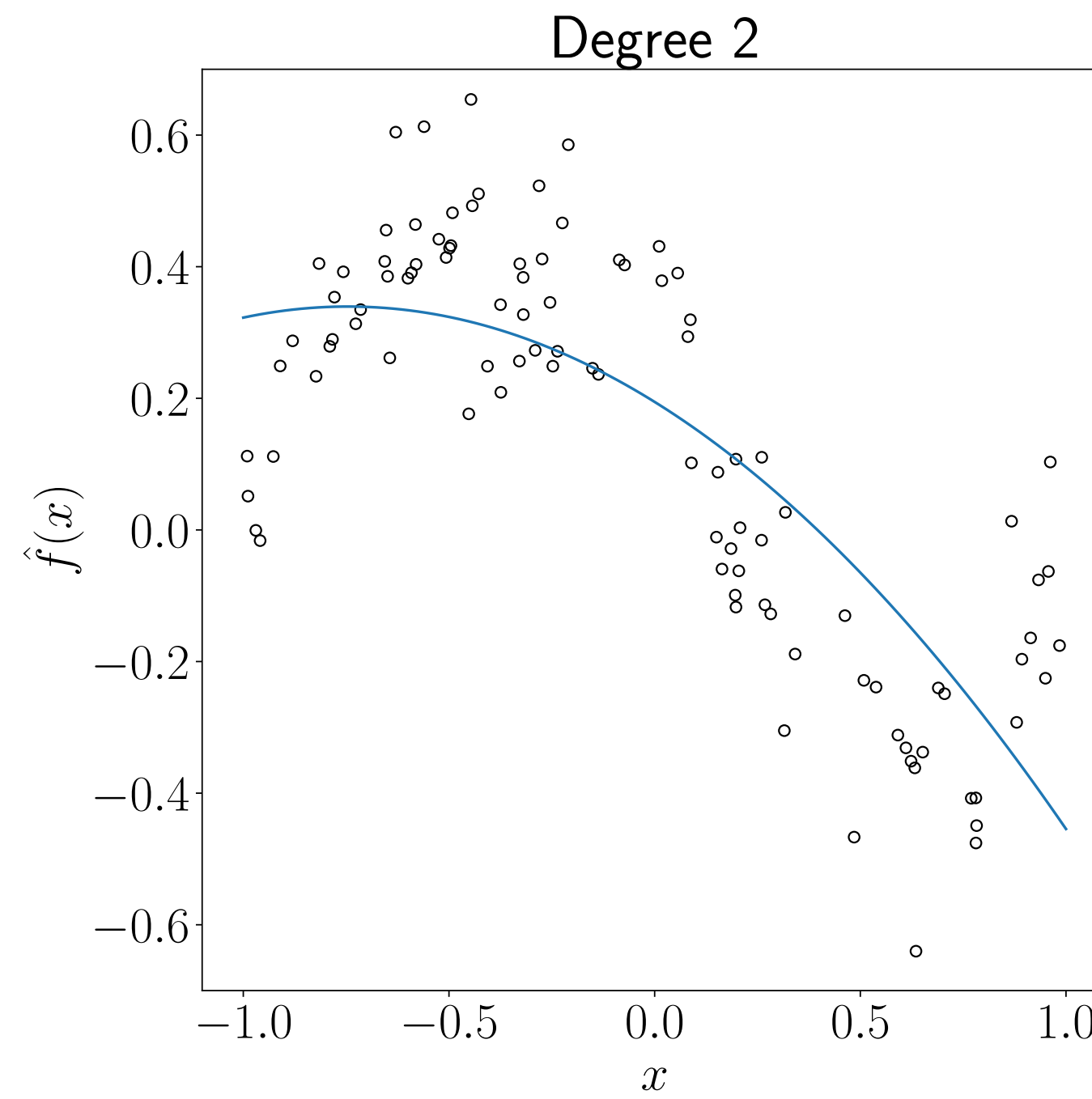
## Goal

minimize  $\|r^d\|^2 = \|A\theta - y^d\|^2$

# Polynomial fit

$N = 100$  data points

Which model is better?



# Auto-regressive time series model

$z_1, z_2, \dots$  is a time series

# Auto-regressive time series model

$z_1, z_2, \dots$  is a time series

**auto-regressive (AR) prediction model**

$$\hat{z}_{t+1} = \theta_1 z_t + \dots + \theta_M z_{t-M+1}, \quad t = M, M+1, \dots$$

(predict  $\hat{z}_{t+1}$  based on previous  $M$  values, where  $M$  is the memory)

# Auto-regressive time series model

$z_1, z_2, \dots$  is a time series

**auto-regressive (AR) prediction model**

$$\hat{z}_{t+1} = \theta_1 z_t + \dots + \theta_M z_{t-M+1}, \quad t = M, M+1, \dots$$

(predict  $\hat{z}_{t+1}$  based on previous  $M$  values, where  $M$  is the memory)

**Goal:** Chose  $\theta$  to minimize sum of squares of prediction errors

$$(\hat{z}_{M+1} - z_{M+1})^2 + \dots + (\hat{z}_T - z_T)^2$$

# Auto-regressive time series model

$z_1, z_2, \dots$  is a time series

**auto-regressive (AR) prediction model**

$$\hat{z}_{t+1} = \theta_1 z_t + \dots + \theta_M z_{t-M+1}, \quad t = M, M+1, \dots$$

(predict  $\hat{z}_{t+1}$  based on previous  $M$  values, where  $M$  is the memory)

**Goal:** Chose  $\theta$  to minimize sum of squares of prediction errors

$$(\hat{z}_{M+1} - z_{M+1})^2 + \dots + (\hat{z}_T - z_T)^2$$

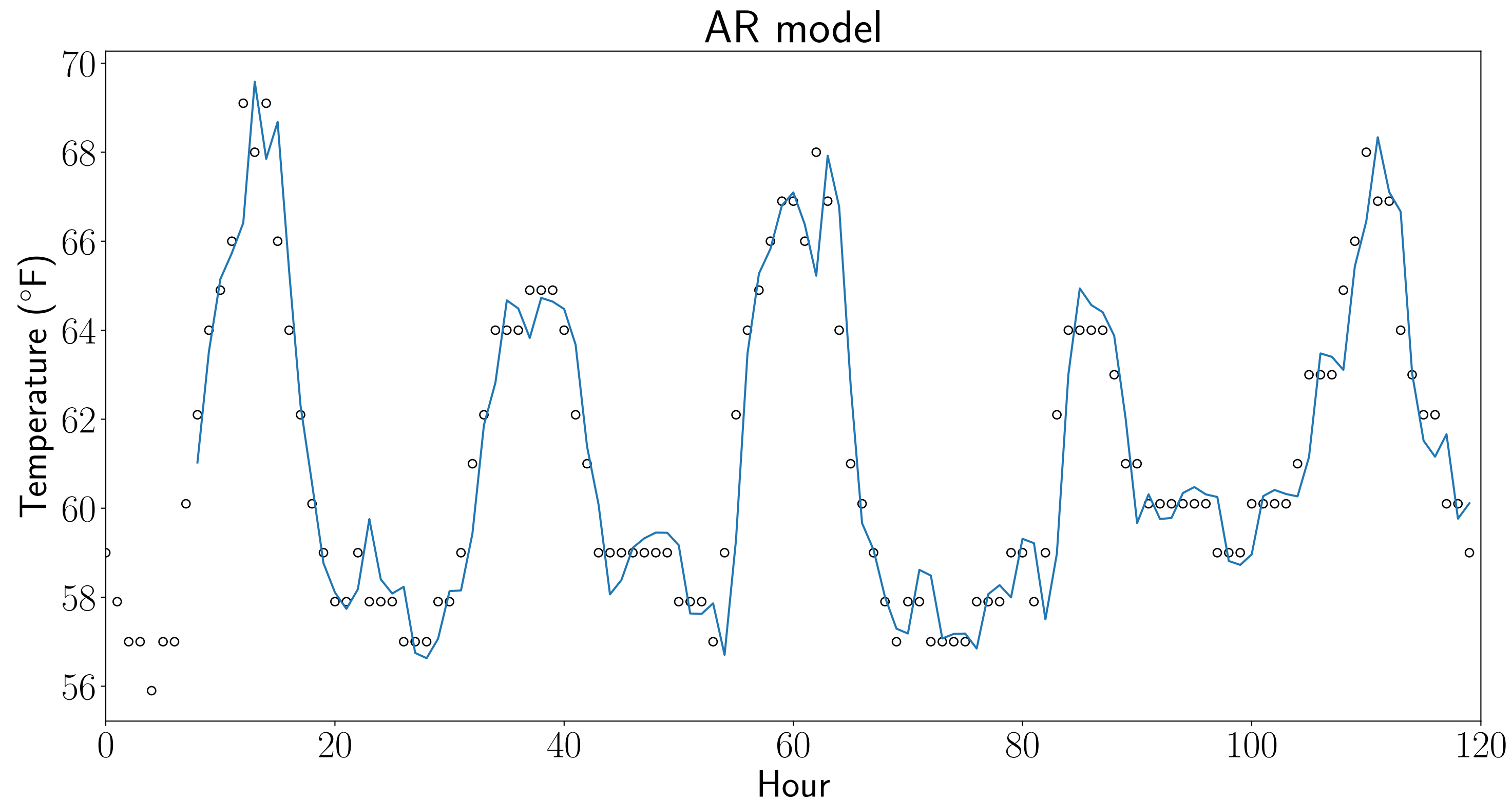
**General data fitting form**

$$y^{(i)} = z_{M+i}, \quad x^{(i)} = (z_{M+i-1}, \dots, z_i), \quad i = 1, \dots, T - M$$

# Auto-regressive time series model

**5 days hourly temperature at  
Los Angeles International  
Airport (LAX)**

- Previous hour:  $\hat{z}_{t+1} = z_t$ , MSE 1.35
- 24 hours before:  $\hat{z}_{t+1} = z_{t-23}$ , MSE = 3.00
- AR model with  $M = 8$ , MSE = 1.02



# Multivariate regression



# Linear regression as general data fitting

**Standard linear regression model**

$$\hat{y} = \hat{f}(x) = x^T \beta + v$$

# Linear regression as general data fitting

## Standard linear regression model

$$\hat{y} = \hat{f}(x) = x^T \beta + v$$

## Equivalent general data fitting model

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

with basis functions

$$f_1(x) = 1, \quad f_i(x) = x_{i-1}, \quad i = 2, \dots, n - 1$$

# Linear regression as general data fitting

## Standard linear regression model

$$\hat{y} = \hat{f}(x) = x^T \beta + v$$

## Equivalent general data fitting model

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

with basis functions

$$f_1(x) = 1, \quad f_i(x) = x_{i-1}, \quad i = 2, \dots, n - 1$$

Therefore, we can write the linear regression model as

$$\hat{y} = \hat{f}(x) = \theta_1 + \theta_2 x_1 + \cdots + \theta_{n+1} x_n = x^T \theta_{2:n} + \theta_1$$

with  $\beta = \theta_{2:n}$ ,  $v = \theta_1$

# General data fitting as linear regression

**General data fitting model**

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

# General data fitting as linear regression

## General data fitting model

$$\hat{f}(x) = \theta_1 f_1(x) + \dots + \theta_p f_p(x)$$

## Equivalent linear regression model

$$\hat{y} = \hat{f}(\tilde{x}) = \tilde{x}^T \beta + v$$

- $f_1(x) = 1$  (common assumption)
- $\tilde{x} = (f_2(x), \dots, f_p(x))$  (transformed features)
- $v = \theta_1$  and  $\beta = \theta_{2:p}$  (linear regression parameters)

# General data fitting as linear regression

## General data fitting model

$$\hat{f}(x) = \theta_1 f_1(x) + \dots + \theta_p f_p(x)$$

## Equivalent linear regression model

$$\hat{y} = \hat{f}(x) = \tilde{x}^T \beta + v$$

- $f_1(x) = 1$  (common assumption)
- $\tilde{x} = (f_2(x), \dots, f_p(x))$  (transformed features)
- $v = \theta_1$  and  $\beta = \theta_{2:p}$  (linear regression parameters)

Our general data fitting framework is nothing more than  
**linear regression on transformed data**

**Validation**

# Generalization

## Main goal

- The goal of model is not to predict outcome for *given data*
- Instead, it is to predict the outcome on *new, unseen data*



# Generalization

## Main goal

- The goal of model is not to predict outcome for *given data*
- Instead, it is to predict the outcome on *new, unseen data*



## Seen/Unseen data

- A model that makes reasonable predictions on new, unseen data has **generalization ability** or it **generalizes**
- A model that makes poor predictions on new, unseen data is said to suffer from **over-fitting**

# Generalization

## Main goal

- The goal of model is not to predict outcome for *given data*
- Instead, it is to predict the outcome on *new, unseen data*



## Seen/Unseen data

- A model that makes reasonable predictions on new, unseen data has **generalization ability** or it **generalizes**
- A model that makes poor predictions on new, unseen data is said to suffer from **over-fitting**

**(Almost) always true in decision making**

The objective function (here, the training error) is just a “surrogate” of the true goal

# Overfitting example



# Validation

Simple and effective method to guess if a model generalize



1. Split data in *training* and *test set* (typical 80%/20% or 90%/10%)
2. Build (train) model on training data set (i.e., compute  $\theta^*$ )
3. Check model's prediction on test data set

# Validation

Simple and effective method to guess if a model generalize



1. Split data in *training* and *test set* (typical 80%/20% or 90%/10%)
2. Build (train) model on training data set (i.e., compute  $\theta^*$ )
3. Check model's prediction on test data set

Compare the MSE prediction error on test vs train data set



If similar, we can *guess* that the model will generalize

# Validation



Useful to choose among different candidate models

- Polynomials of different degrees
- Models with different transformed features

# Validation



Useful to choose among different candidate models

- Polynomials of different degrees
- Models with different transformed features



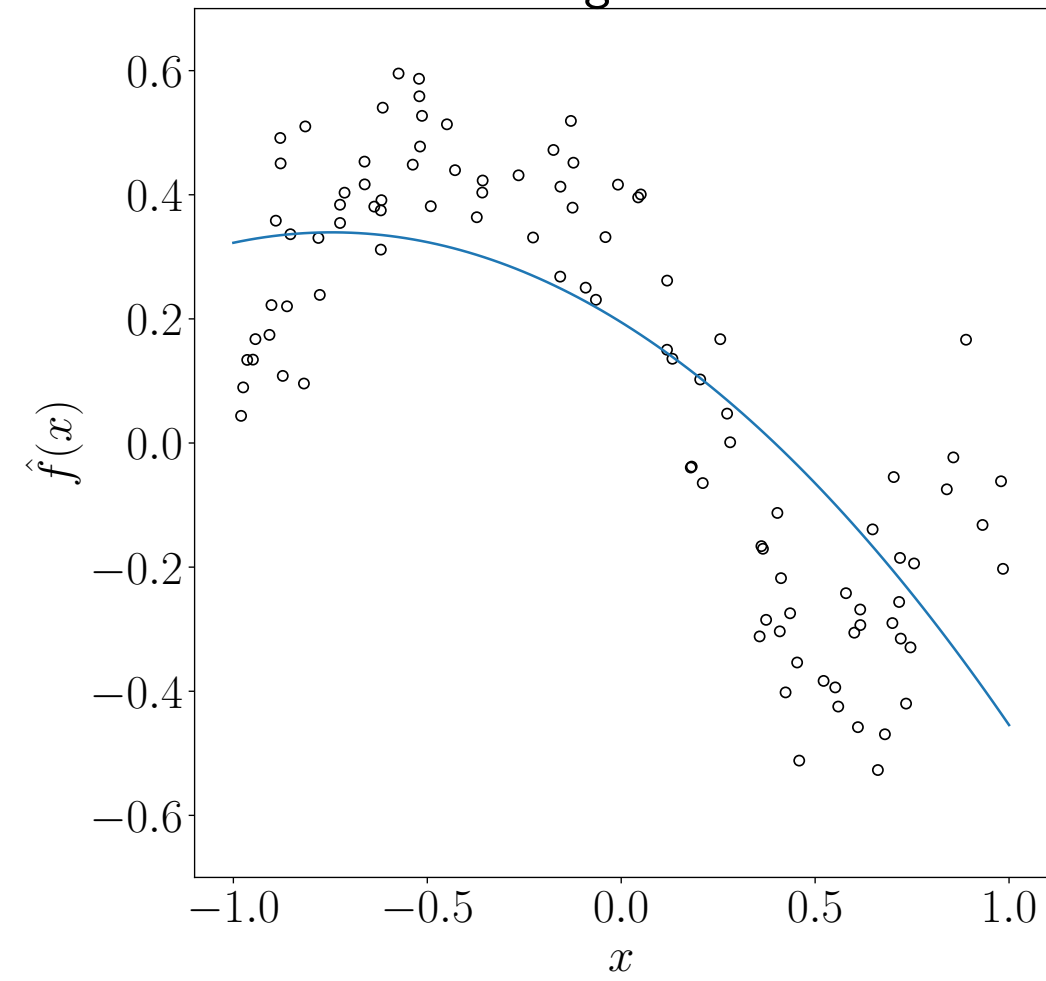
We choose the one with ***lowest test error***

# Example

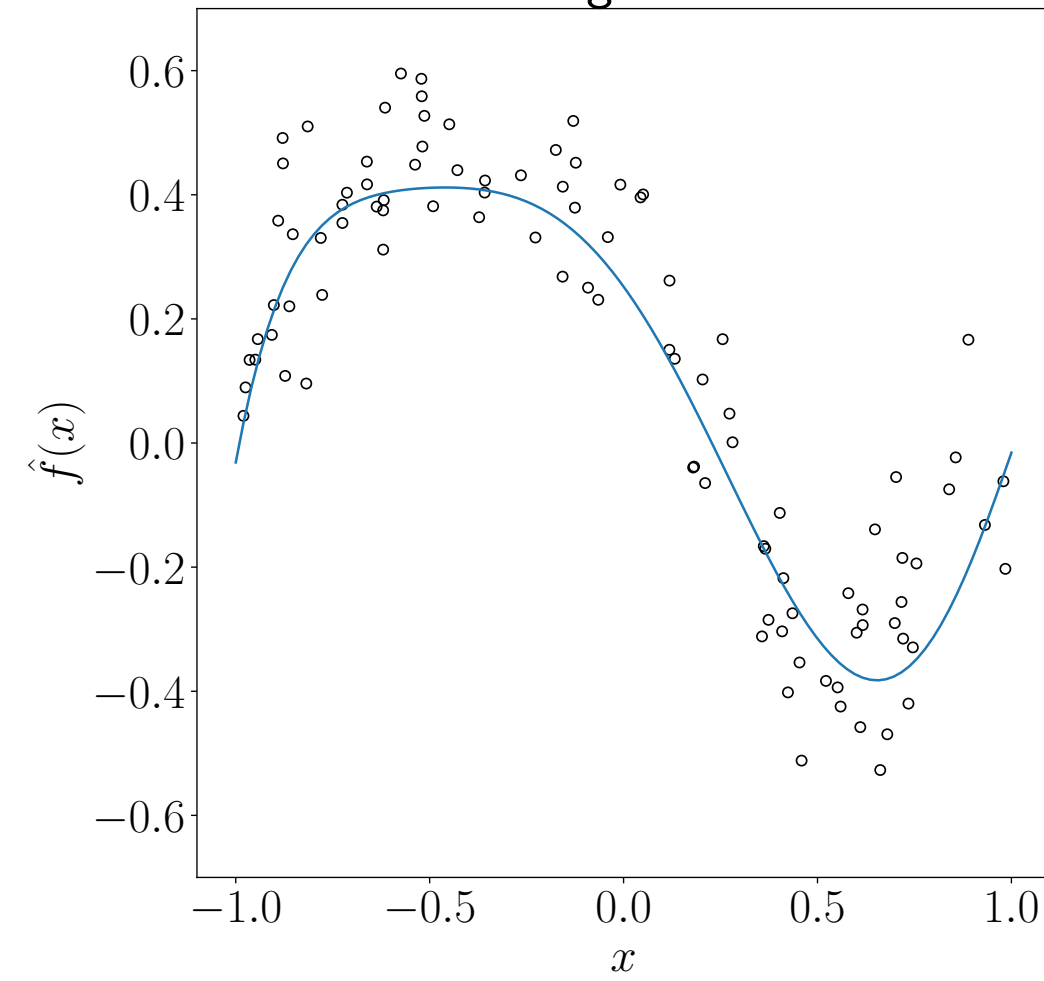
## Polynomial fit

100 points *training set*,  
100 points *test set*

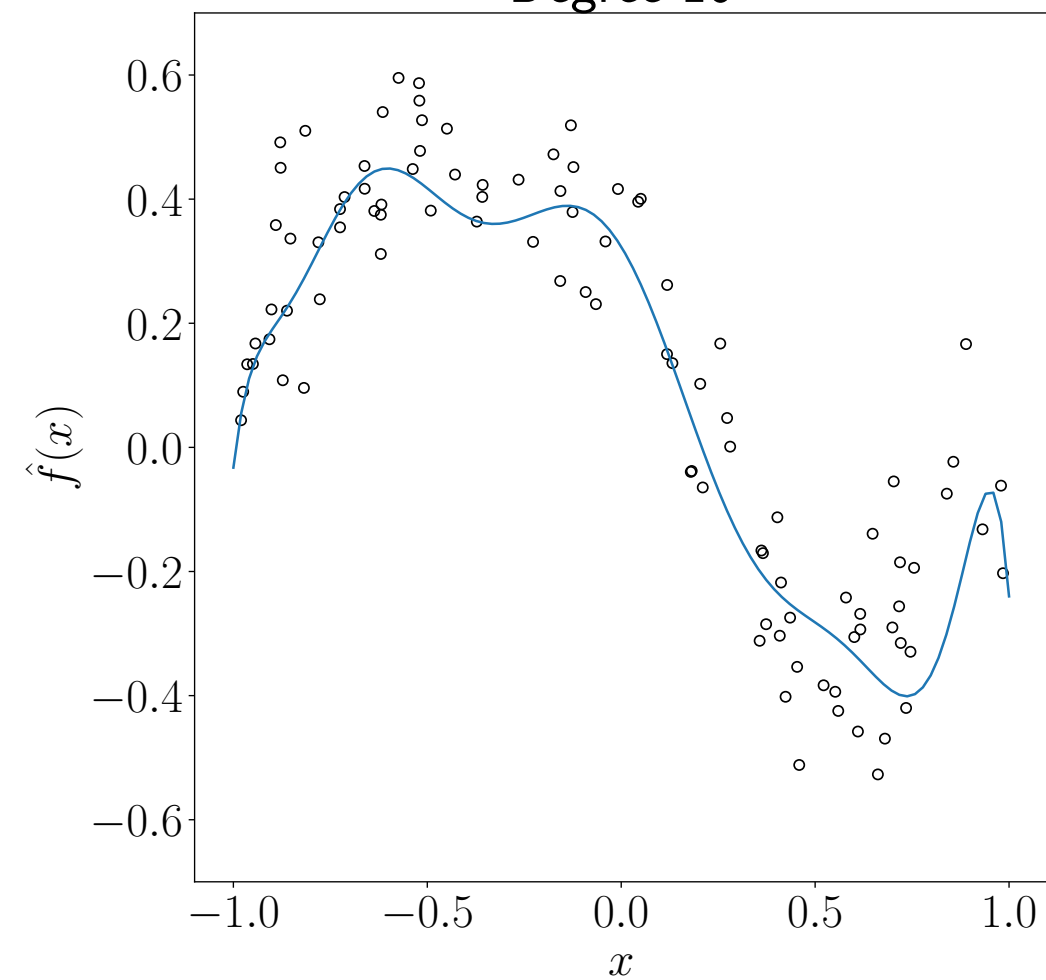
Degree 2



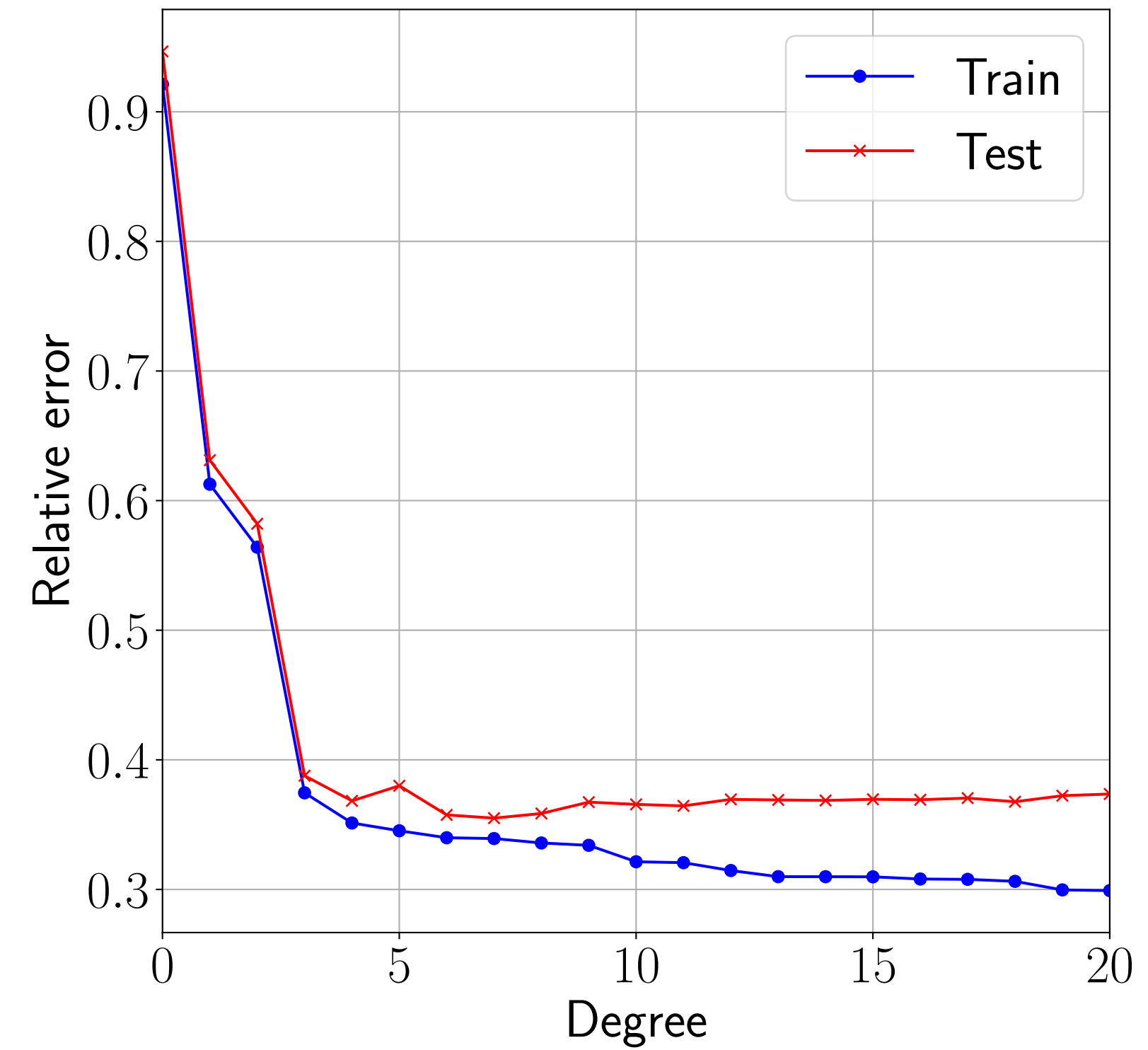
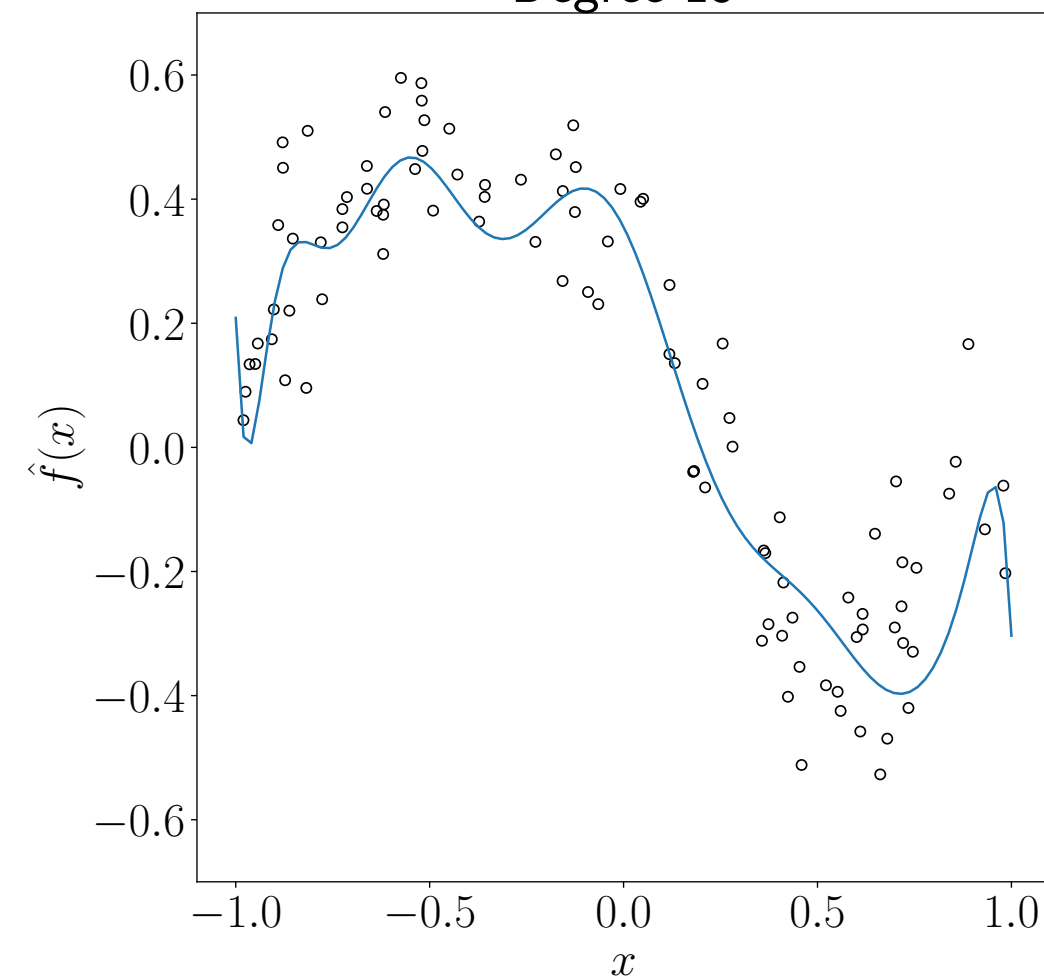
Degree 6



Degree 10



Degree 15

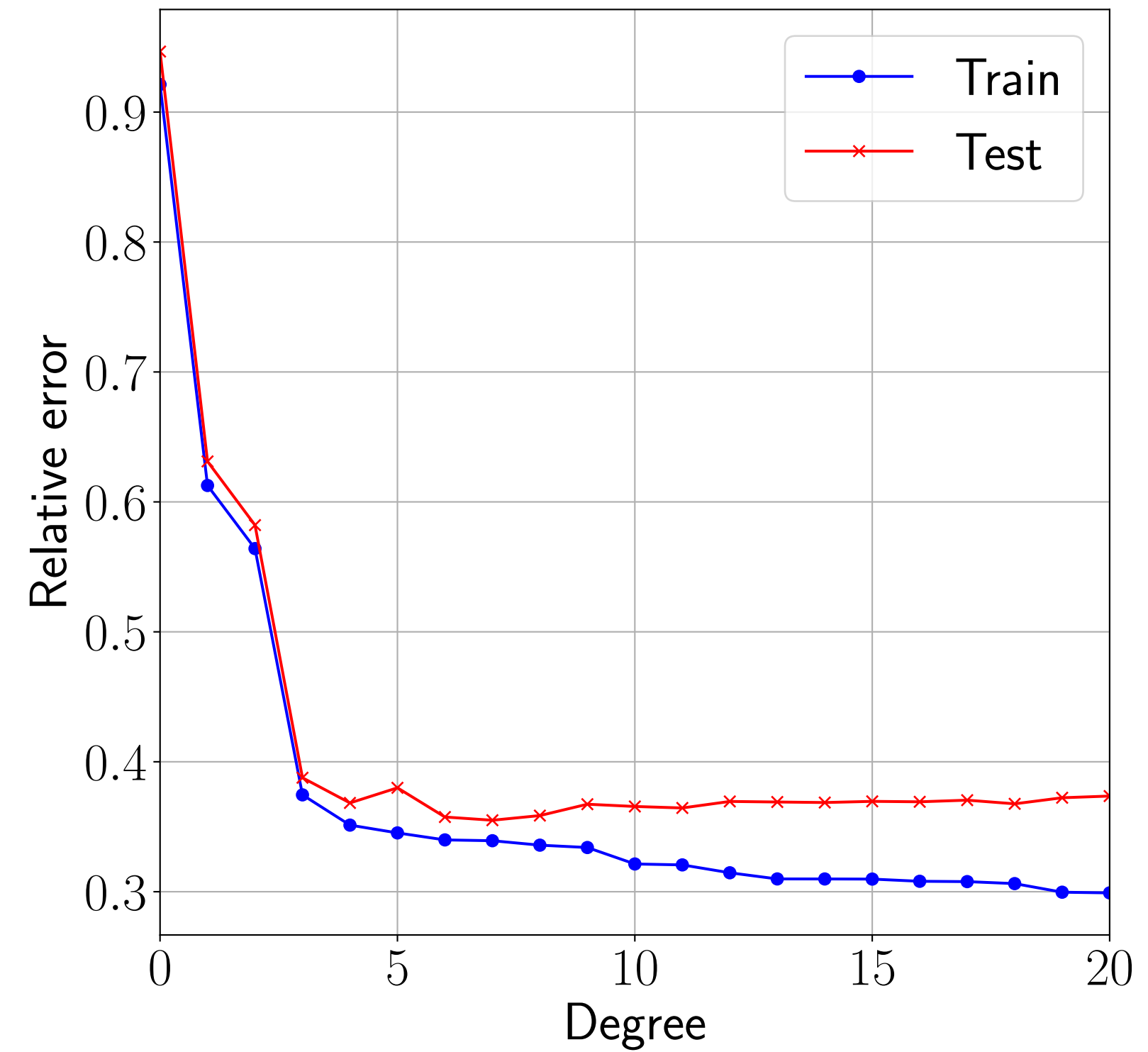
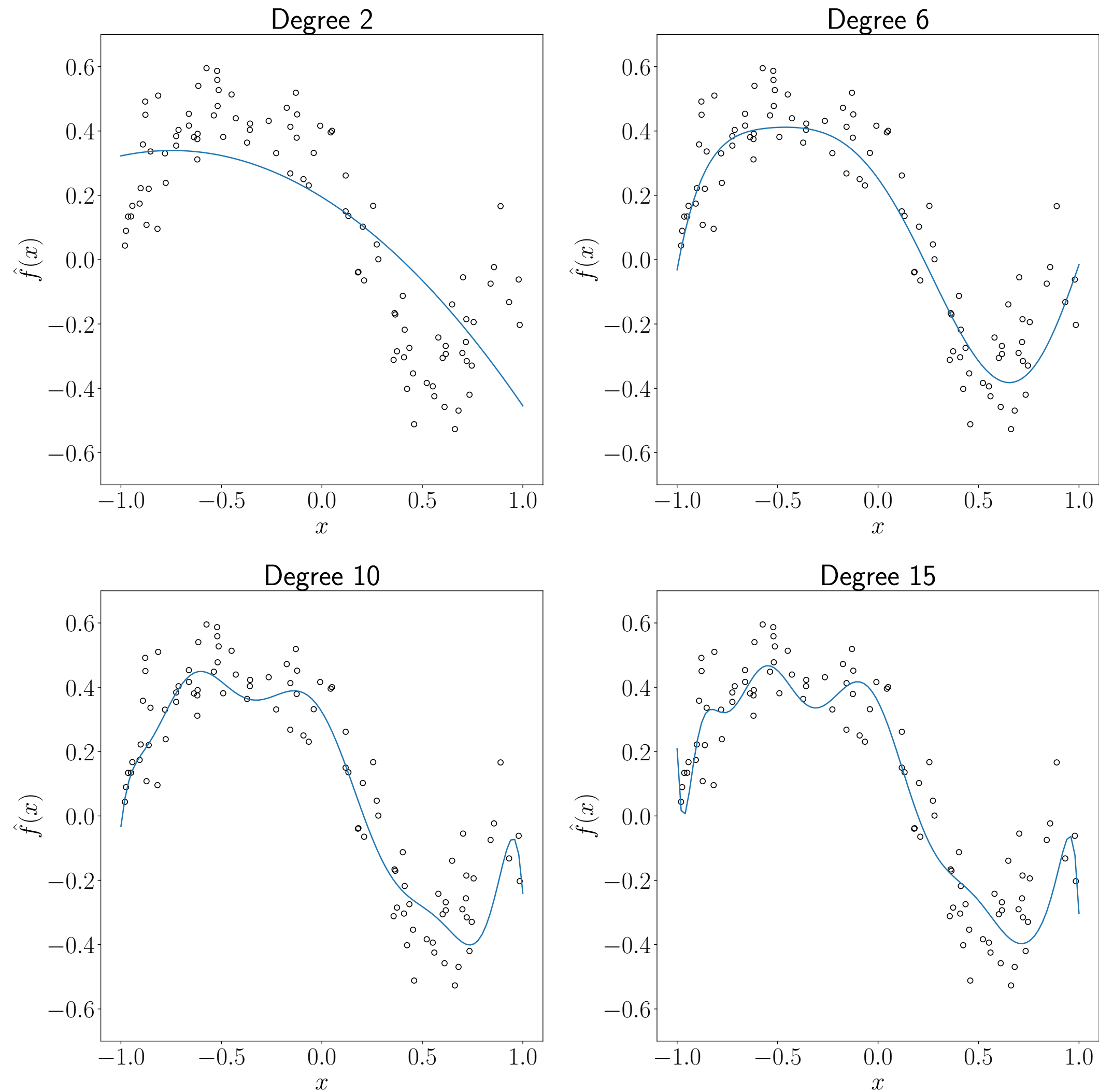




# Example

## Polynomial fit

100 points *training set*,  
100 points *test set*



**It suggests degrees 6/7/8 are reasonable choices**

# Cross validation

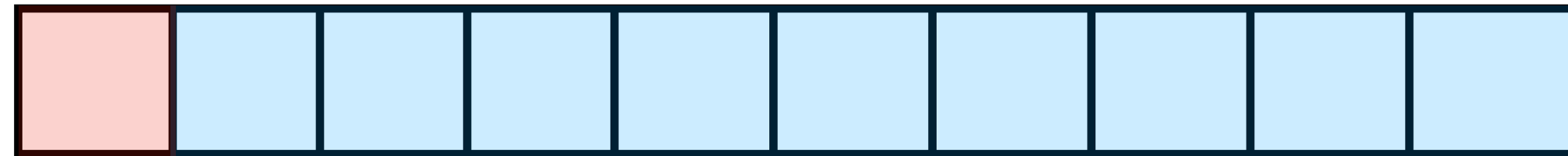
Check **which method** provides the better models  
(e.g., choice of features, basis functions, etc.)

# Cross validation

Check **which method** provides the better models  
(e.g., choice of features, basis functions, etc.)

## Data

**1 fold**



**2 fold**



⋮

⋮

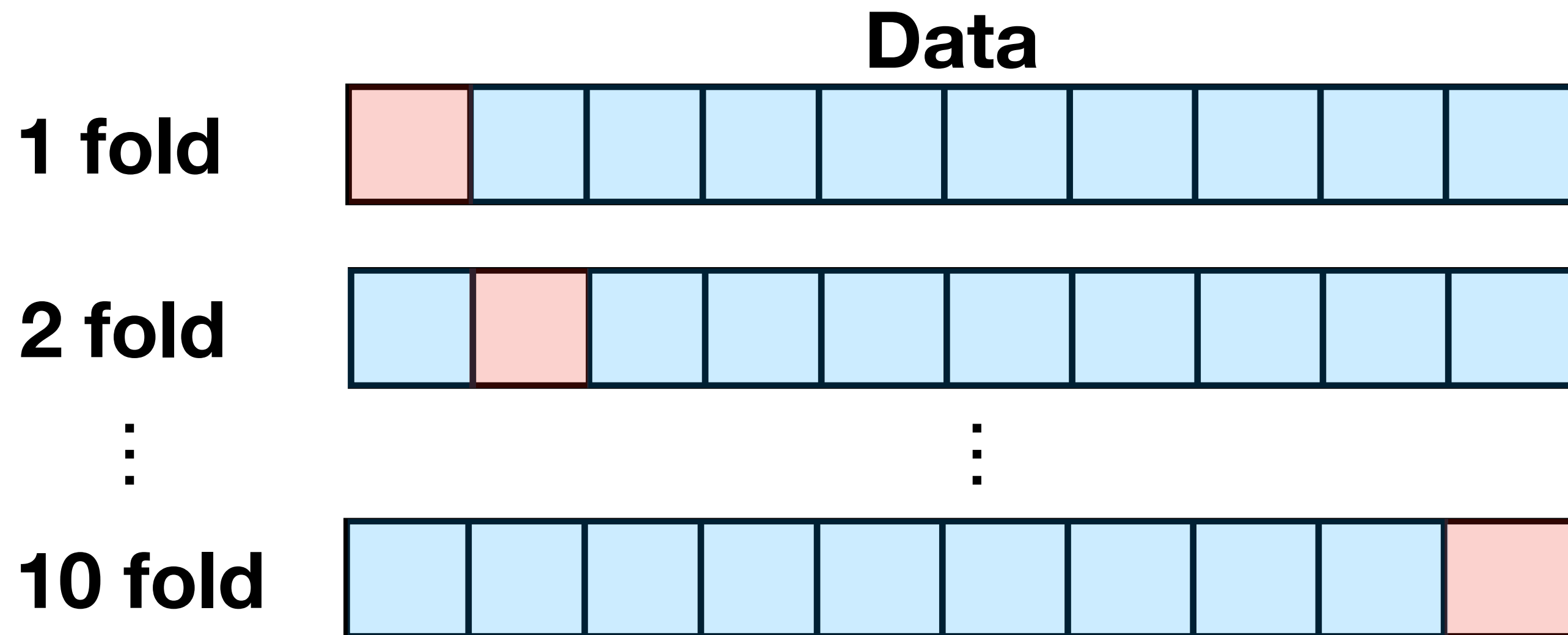
**10 fold**



1. Divide data in 10 folds
2. For  $i = 1, \dots, 10$  build (train) model using all folds except  $i$
3. Test model on data in fold  $i$

# Cross validation

Check **which method** provides the better models  
(e.g., choice of features, basis functions, etc.)



1. Divide data in 10 folds
2. For  $i = 1, \dots, 10$  build (train) model using all folds except  $i$
3. Test model on data in fold  $i$

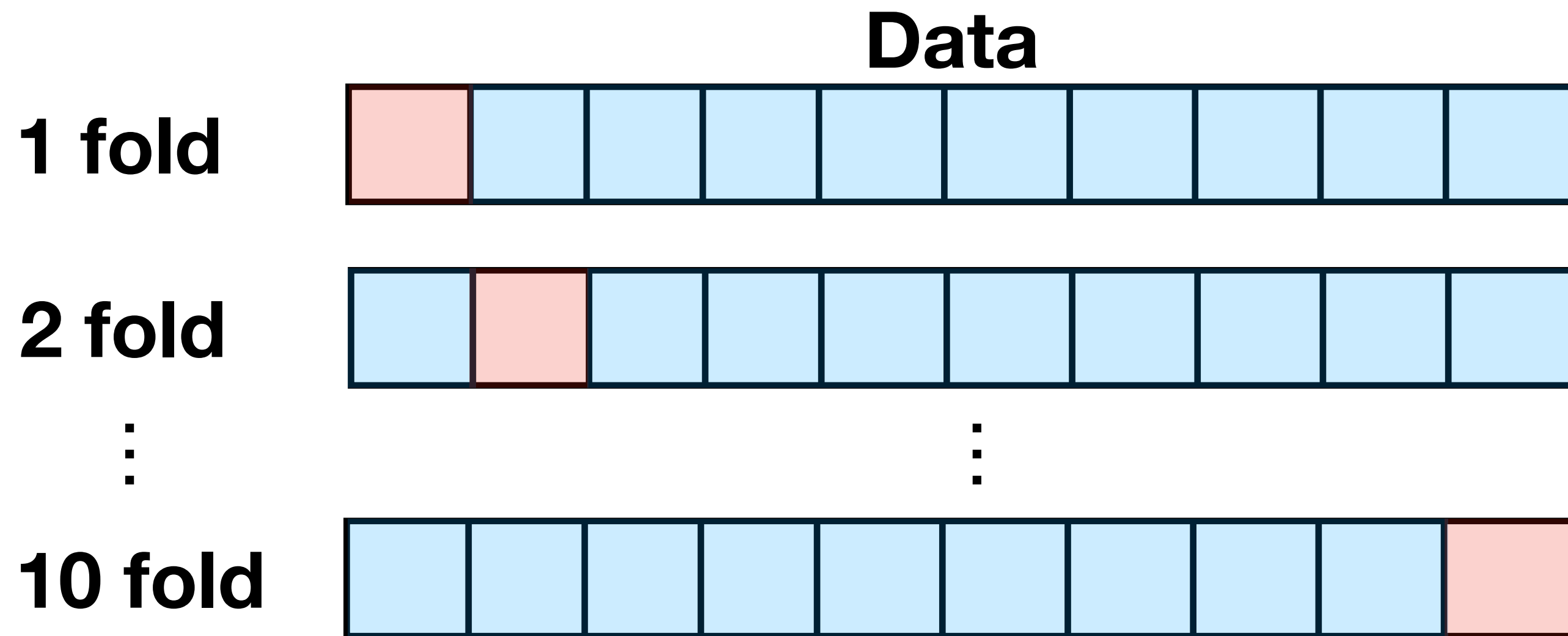
For each fold, compare the MSE  
prediction error on test vs train data



If test vs train MSE are similar and consistent,  
we can *guess* the model will generalize

# Cross validation

Check **which method** provides the better models  
(e.g., choice of features, basis functions, etc.)



1. Divide data in 10 folds
2. For  $i = 1, \dots, 10$  build (train) model using all folds except  $i$
3. Test model on data in fold  $i$

For each fold, compare the MSE prediction error on test vs train data



If test vs train MSE are similar and consistent, we can *guess* the model will generalize

## Remark

This returns many models, to get a model, we have to afterwards train over the whole data set

**Example**

# House price regression

774 house sales in Sacramento area



## Base features

- $x_1$  is the area of the house (in 1000ft<sup>2</sup>)
- $x_2$  is the number of bedrooms
- $x_3$  is 1 for condo and 0 for house (boolean)
- $x_4$  is the ZIP code (62 values)

# House price regression

774 house sales in Sacramento area

## Model

$$\hat{f}(x) = \theta_1 f_1(x) + \dots + \theta_8 f_8(x)$$



## Base features

- $x_1$  is the area of the house (in 1000ft<sup>2</sup>)
- $x_2$  is the number of bedrooms
- $x_3$  is 1 for condo and 0 for house (boolean)
- $x_4$  is the ZIP code (62 values)

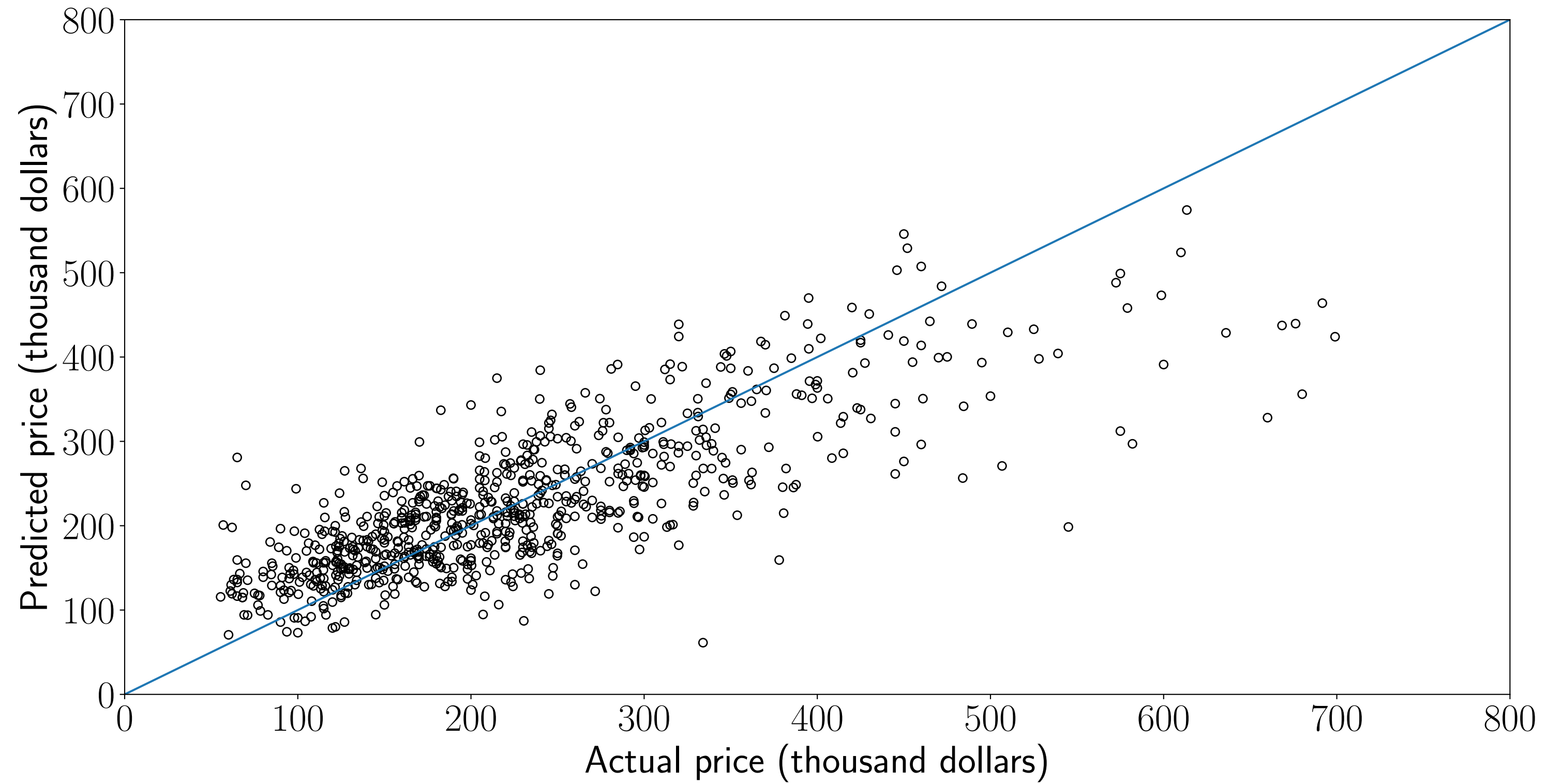
## Transformed features

- $f_1(x) = 1$  (offset)
- $f_2(x) = x_1$
- $f_3(x) = \max\{x_1 - 1.5, 0\}$  is the house area above 1500ft<sup>2</sup>
- $f_4(x) = x_2$
- $f_5(x) = x_3$
- $f_6(x), f_7(x), f_8(x)$  are Boolean functions of  $x_4$  to encode 4 groups of nearby zip codes (i.e., neighborhood)



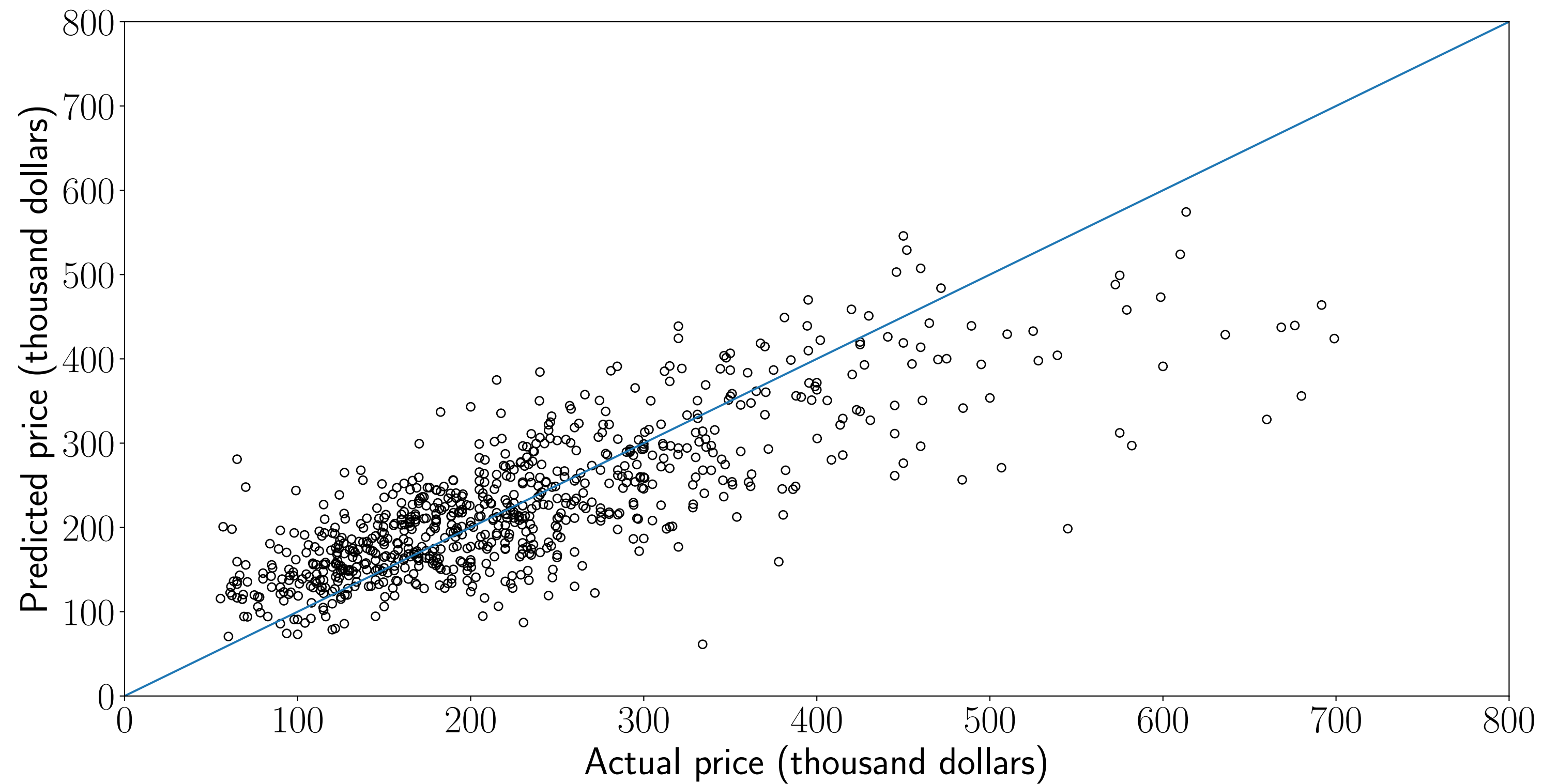
# House price prediction

## Fitting over the whole dataset



# House price prediction

## Fitting over the whole dataset



**How can we be sure it generalizes?**

# House price regression

## Crossvalidation

5 folds of 155 sales each

Fold	Train error	Test error	$\theta_1$	$\theta_2$	$\theta_3$	$\theta_4$	$\theta_5$	$\theta_6$	$\theta_7$	$\theta_8$
1	0.26	0.30	115.60	177.53	-47.05	-14.64	-13.90	-112.00	-122.59	-36.51
2	0.26	0.29	121.59	165.48	-30.35	-17.74	-20.21	-95.22	-103.67	-7.94
3	0.27	0.25	117.83	181.18	-49.78	-19.30	-18.68	-106.50	-112.76	-32.37
4	0.27	0.25	104.00	174.54	-41.96	-18.81	-17.13	-85.68	-92.09	-6.99
5	0.27	0.27	119.40	178.57	-44.82	-19.50	-25.98	-103.95	-111.56	-36.89

# House price regression

## Crossvalidation

5 folds of 155 sales each

Fold	Train error	Test error	$\theta_1$	$\theta_2$	$\theta_3$	$\theta_4$	$\theta_5$	$\theta_6$	$\theta_7$	$\theta_8$
1	0.26	0.30	115.60	177.53	-47.05	-14.64	-13.90	-112.00	-122.59	-36.51
2	0.26	0.29	121.59	165.48	-30.35	-17.74	-20.21	-95.22	-103.67	-7.94
3	0.27	0.25	117.83	181.18	-49.78	-19.30	-18.68	-106.50	-112.76	-32.37
4	0.27	0.25	104.00	174.54	-41.96	-18.81	-17.13	-85.68	-92.09	-6.99
5	0.27	0.27	119.40	178.57	-44.82	-19.50	-25.98	-103.95	-111.56	-36.89

### Good feature choice

- Models (parameters) reasonably stable across folds
- Similar train and test errors

# Least squares data fitting

Today, we learned to:

- **Formulate** many data fitting problems as least squares
- **Avoid** overfitting by keeping our models simple
- **Compare** our models using validation

# References

- S. Boyd, L. Vandenberghe: Introduction to Applied Linear Algebra – Vectors, Matrices, and Least Squares
  - Chapter 13: least squares data fitting

# Next lecture

- Multi-objective least squares