Lecture 6
Least-squares applications

- least-squares data fitting
- growing sets of regressors
- system identification
- growing sets of measurements and recursive least-squares

Least-squares data fitting

we are given:

- functions $f_1, \ldots, f_n : S \to \mathbf{R}$, called *regressors* or *basis functions*
- *data* or *measurements* (s_i, g_i), $i = 1, \ldots, m$, where $s_i \in S$ and (usually) $m \gg n$

problem: find coefficients $x_1, \ldots, x_n \in \mathbf{R}$ so that

$$x_1 f_1(s_i) + \cdots + x_n f_n(s_i) \approx g_i, \quad i = 1, \ldots, m$$

$i.e.$, find linear combination of functions that fits data

least-squares fit: choose x to minimize total square fitting error:

$$\sum_{i=1}^{m} \left(x_1 f_1(s_i) + \cdots + x_n f_n(s_i) - g_i\right)^2$$

- using matrix notation, total square fitting error is $\|Ax - g\|^2$, where $A_{ij} = f_j(s_i)$

- hence, least-squares fit is given by

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

  (assuming A is skinny, full rank)

- corresponding function is

$$f_{\mathrm{lsfit}}(s) = x_1 f_1(s) + \cdots + x_n f_n(s)$$

- applications:
  - interpolation, extrapolation, smoothing of data
  - developing simple, approximate model of data

Least-squares polynomial fitting

problem: fit polynomial of degree $< n$,

$$p(t) = a_0 + a_1 t + \cdots + a_{n-1} t^{n-1},$$

to data (t_i, y_i), $i = 1, \ldots, m$

- basis functions are $f_j(t) = t^{j-1}$, $j = 1, \ldots, n$
- matrix A has form $A_{ij} = t_i^{j-1}$

$$A = \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \cdots & t_2^{n-1} \\ & \vdots & & & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^{n-1} \end{bmatrix}$$

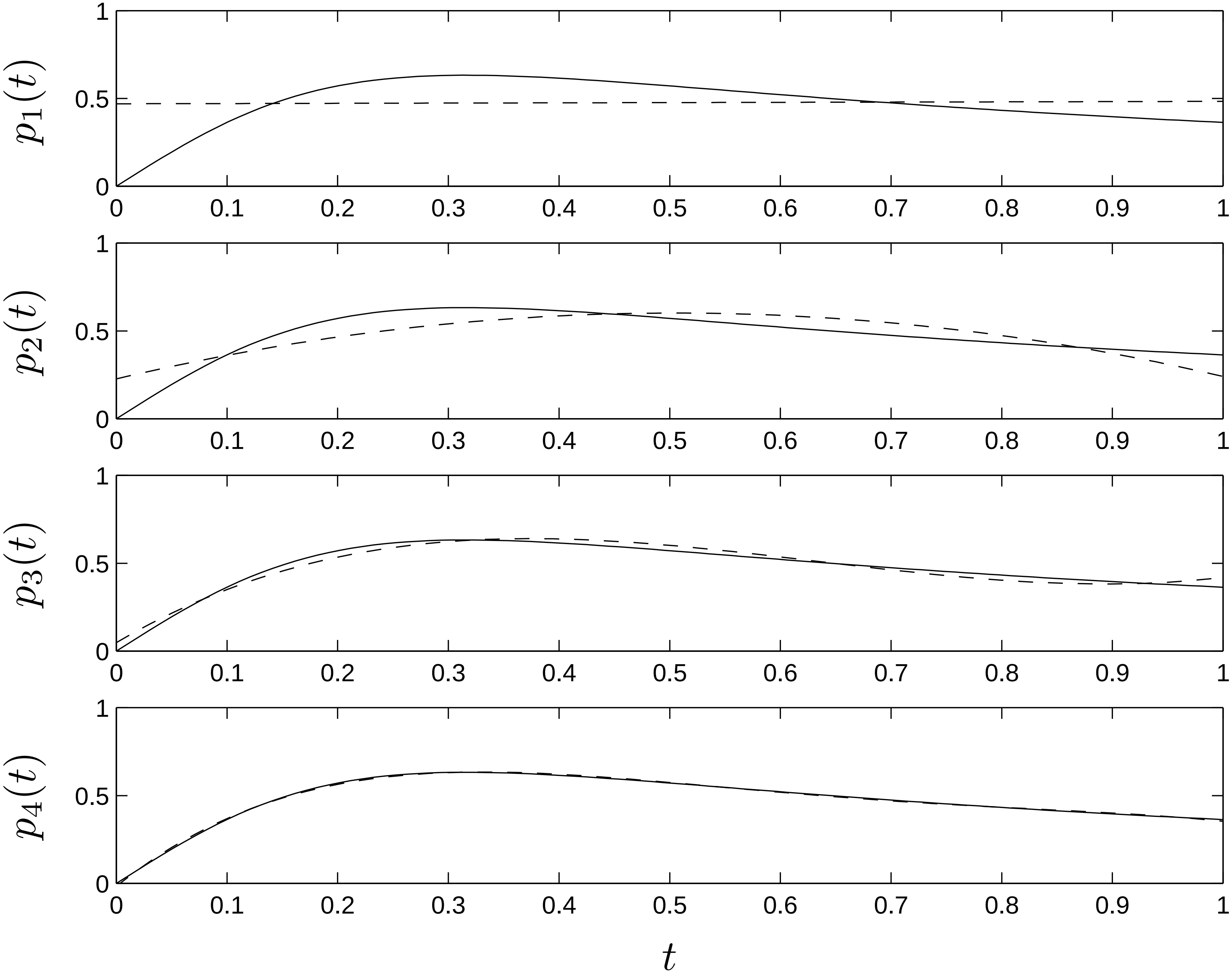
(called a *Vandermonde matrix*)

assuming $t_k \neq t_l$ for $k \neq l$ and $m \geq n$, A is full rank:

- suppose $Aa = 0$
- corresponding polynomial $p(t) = a_0 + \cdots + a_{n-1}t^{n-1}$ vanishes at m points $t_1, \ldots, t_m$
- by fundamental theorem of algebra p can have no more than $n - 1$ zeros, so p is identically zero, and $a = 0$
- columns of A are independent, *i.e.*, A full rank

Example

- fit $g(t) = 4t/(1 + 10t^2)$ with polynomial
- $m = 100$ points between $t = 0$ & $t = 1$
- least-squares fit for degrees 1, 2, 3, 4 have RMS errors $.135$, $.076$, $.025$, $.005$, respectively



$p_1(t)$
$p_2(t)$
$p_3(t)$
$p_4(t)$
1
0.5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
t

Growing sets of regressors

consider *family* of least-squares problems

$$\text{minimize} \quad \|\textstyle\sum_{i=1}^{p} x_i a_i - y\|$$

for $p = 1, \ldots, n$

($a_1, \ldots, a_p$ are called *regressors*)

- approximate y by linear combination of $a_1, \ldots, a_p$
- project y onto $\mathrm{span}\{a_1, \ldots, a_p\}$
- regress y on $a_1, \ldots, a_p$
- as p increases, get better fit, so optimal residual decreases

solution for each $p \leq n$ is given by

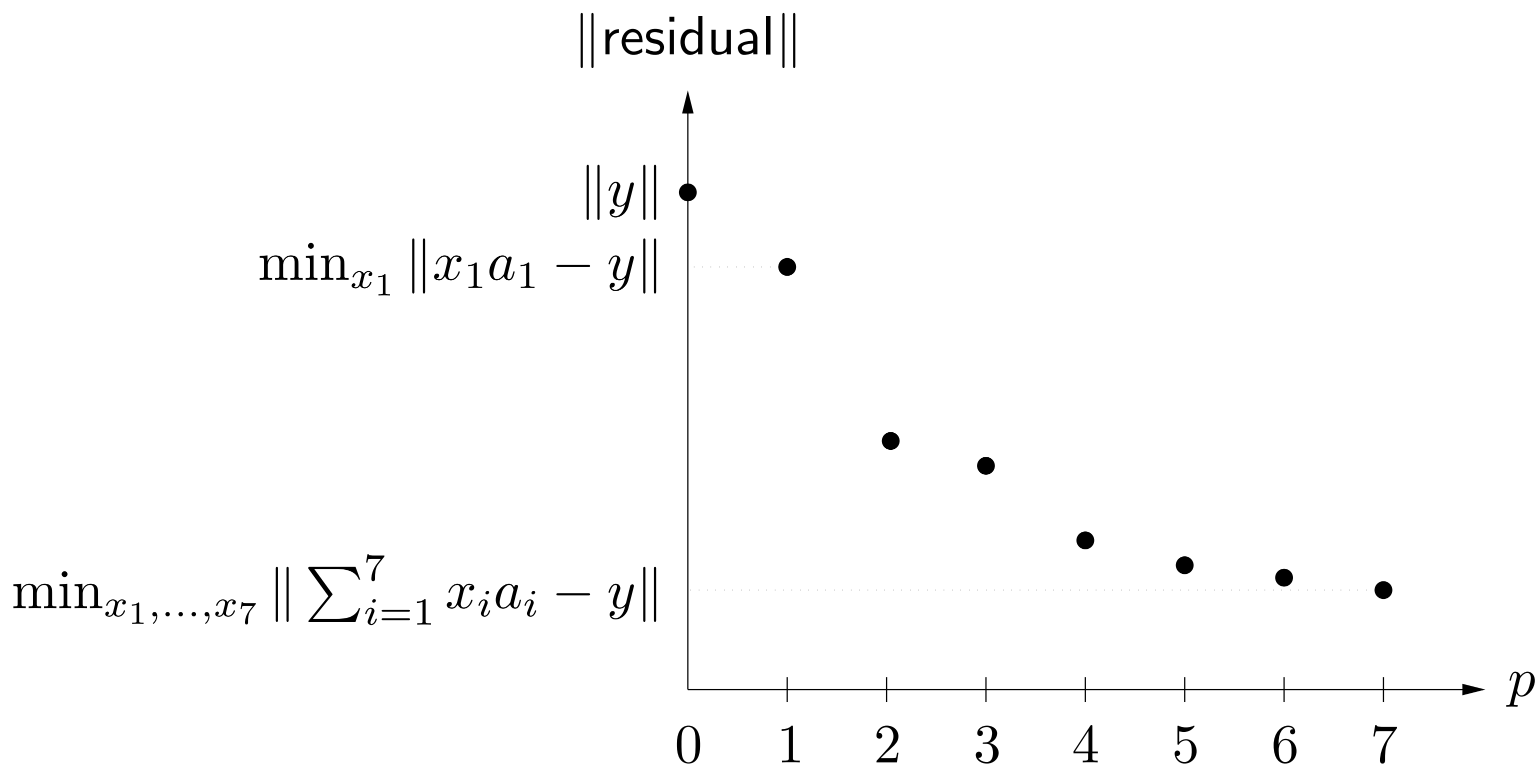
$$x_{\mathrm{ls}}^{(p)} = (A_p^T A_p)^{-1} A_p^T y = R_p^{-1} Q_p^T y$$

where

- $A_p = [a_1 \cdots a_p] \in \mathbf{R}^{m \times p}$ is the first p columns of A
- $A_p = Q_p R_p$ is the QR factorization of A_p
- $R_p \in \mathbf{R}^{p \times p}$ is the leading $p \times p$ submatrix of R
- $Q_p = [q_1 \cdots q_p]$ is the first p columns of Q

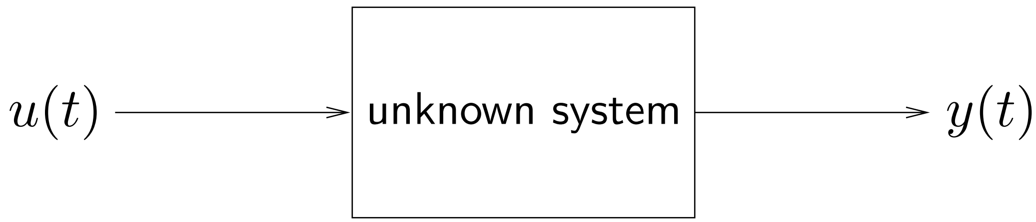
Norm of optimal residual versus p

plot of optimal residual versus p shows how well y can be matched by linear combination of $a_1, \ldots, a_p$, as function of p



Least-squares system identification

we measure input $u(t)$ and output $y(t)$ for $t = 0, \ldots, N$ of unknown system



system identification problem: find reasonable model for system based on measured I/O data u, y

example with scalar u, y (vector u, y readily handled): fit I/O data with moving-average (MA) model with n delays

$$\hat{y}(t) = h_0 u(t) + h_1 u(t-1) + \cdots + h_n u(t-n)$$

where $h_0, \ldots, h_n \in \mathbf{R}$

we can write model or predicted output as

$$\begin{bmatrix} \hat{y}(n) \\ \hat{y}(n+1) \\ \vdots \\ \hat{y}(N) \end{bmatrix} = \begin{bmatrix} u(n) & u(n-1) & \cdots & u(0) \\ u(n+1) & u(n) & \cdots & u(1) \\ \vdots & \vdots & & \vdots \\ u(N) & u(N-1) & \cdots & u(N-n) \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_n \end{bmatrix}$$

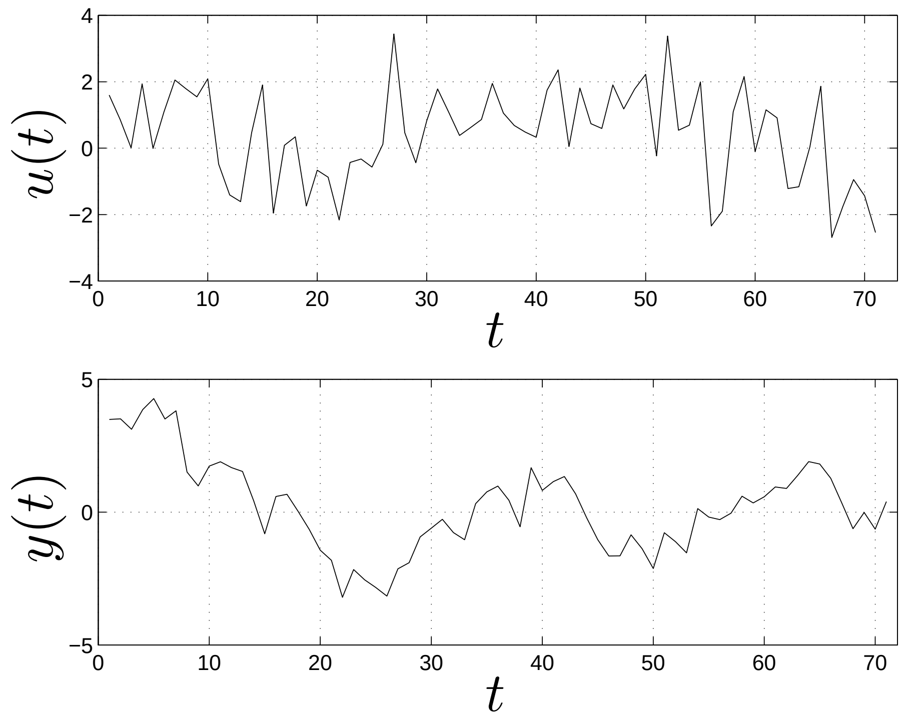
model prediction error is

$$e = (y(n) - \hat{y}(n), \ldots, y(N) - \hat{y}(N))$$

least-squares identification: choose model ($i.e.$, h) that minimizes norm of model prediction error $\|e\|$

. . . a least-squares problem (with variables h)

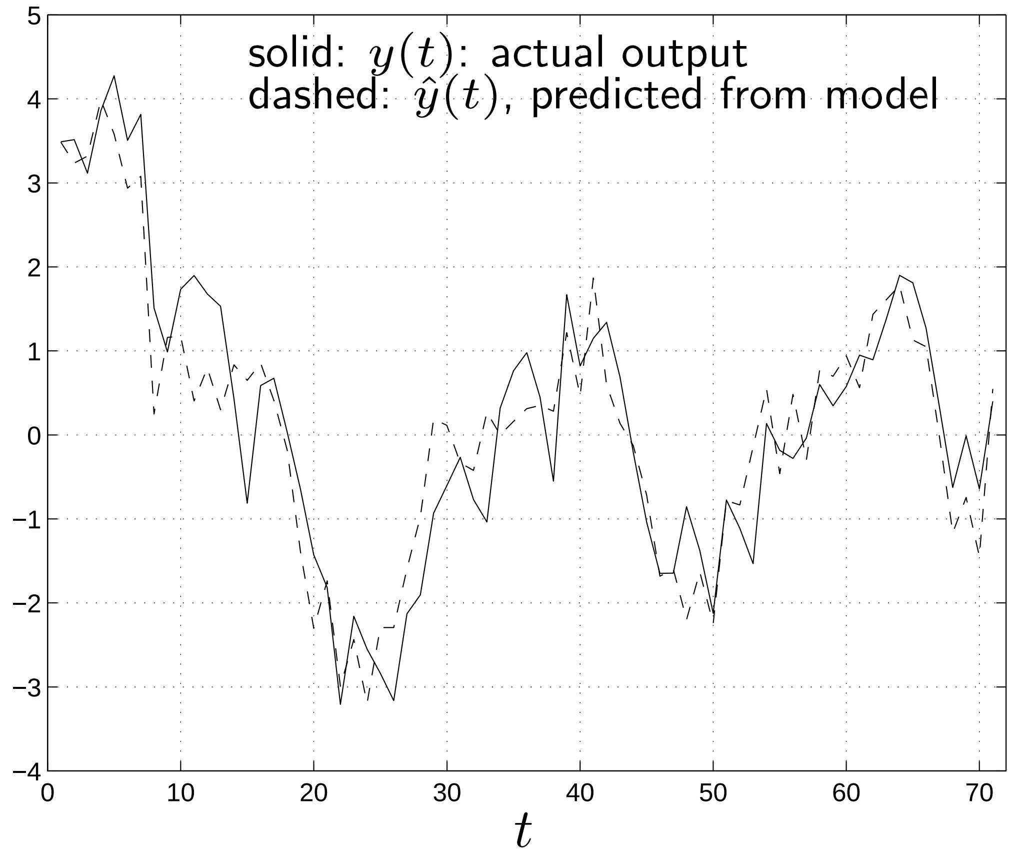
Example



for $n = 7$ we obtain MA model with

$$(h_0, \ldots, h_7) = (.024,\ .282,\ .418,\ .354,\ .243,\ .487,\ .208,\ .441)$$

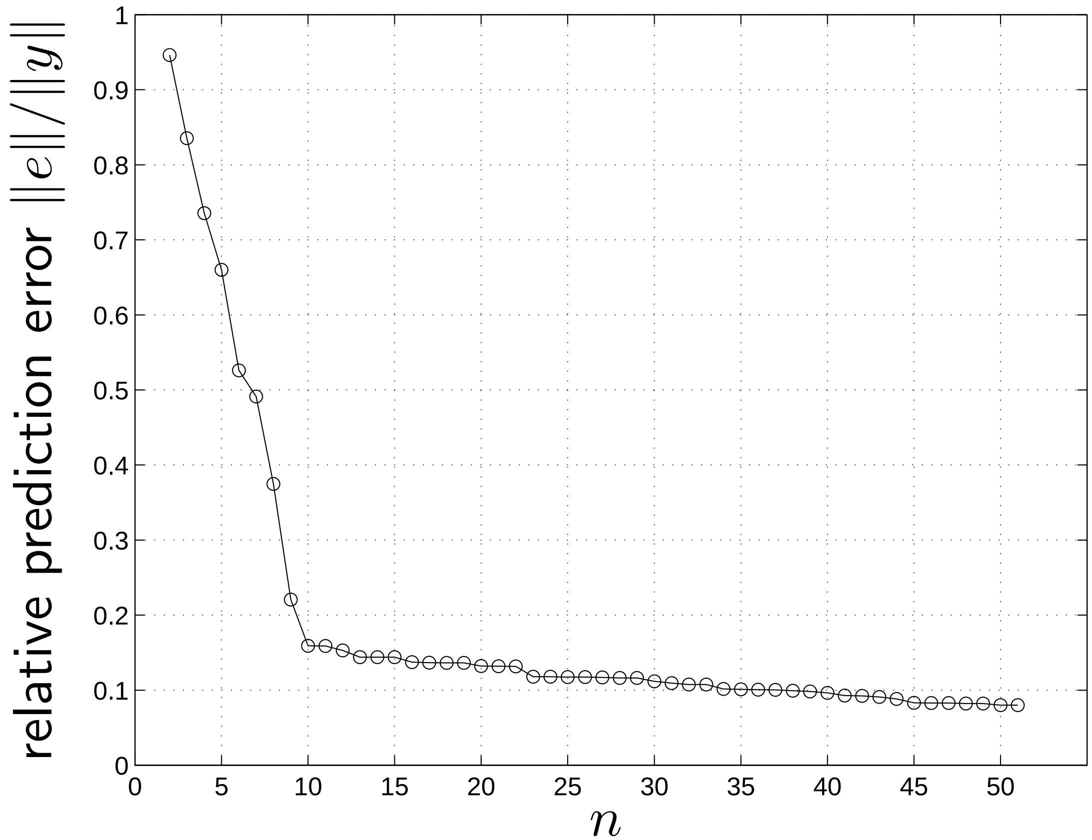
with relative prediction error $\|e\|/\|y\| = 0.37$


solid: $y(t)$: actual output
dashed: $\hat{y}(t)$, predicted from model
5
4
3
2
1
0
−1
−2
−3
−4
0
10
20
30
40
50
60
70
t

Model order selection

question: how large should n be?

- obviously the larger n, the smaller the prediction error *on the data used to form the model*
- suggests using largest possible model order for smallest prediction error

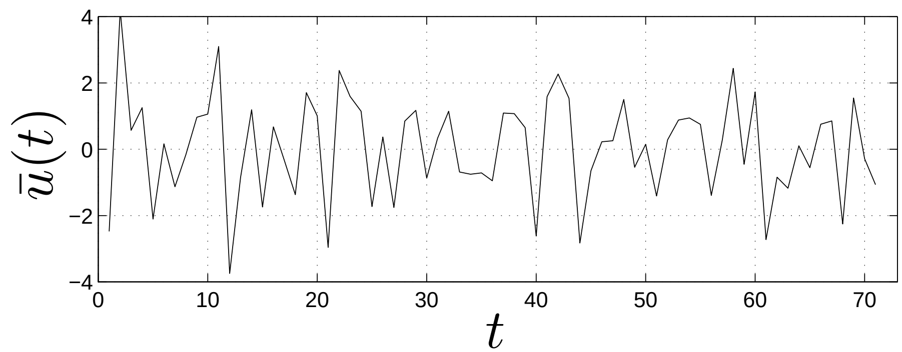


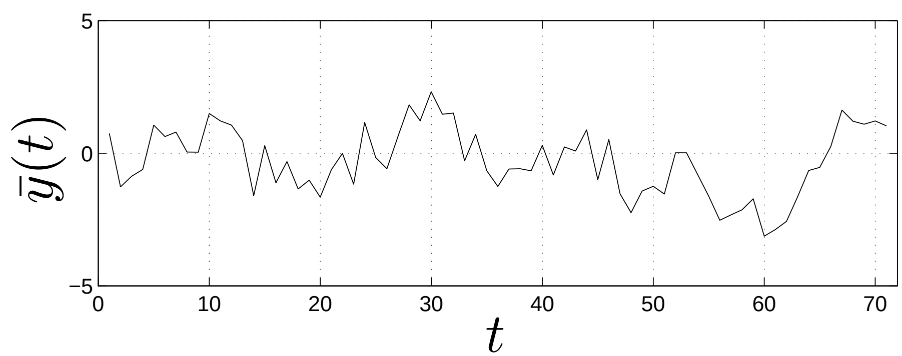
difficulty: for n too large the *predictive ability* of the model on *other I/O data* (from the same system) becomes worse

Out of sample validation

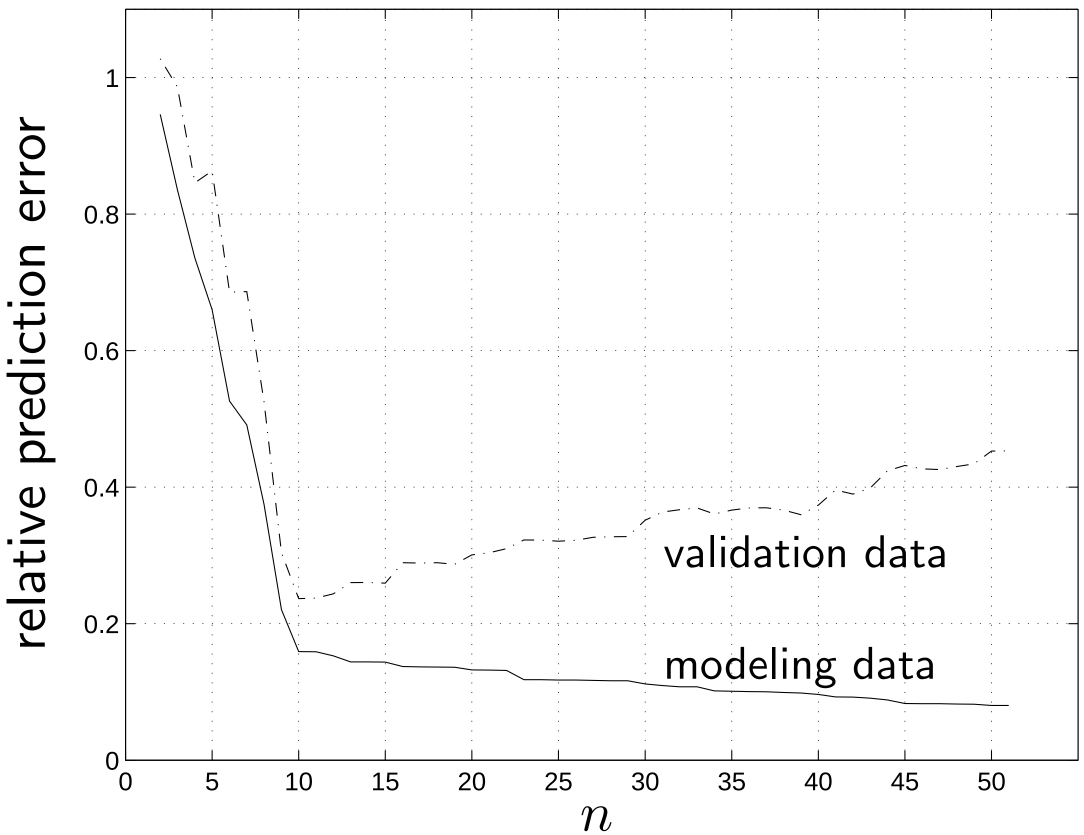
evaluate model predictive performance on another I/O data set *not used to develop model*

model validation data set:



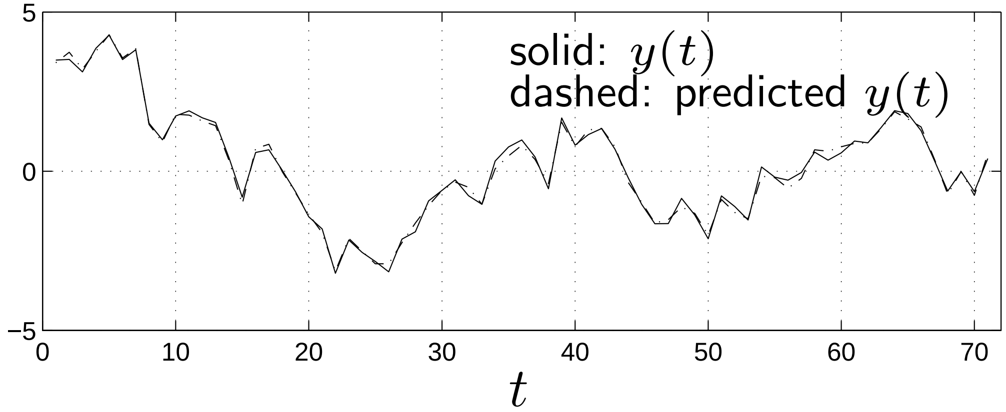


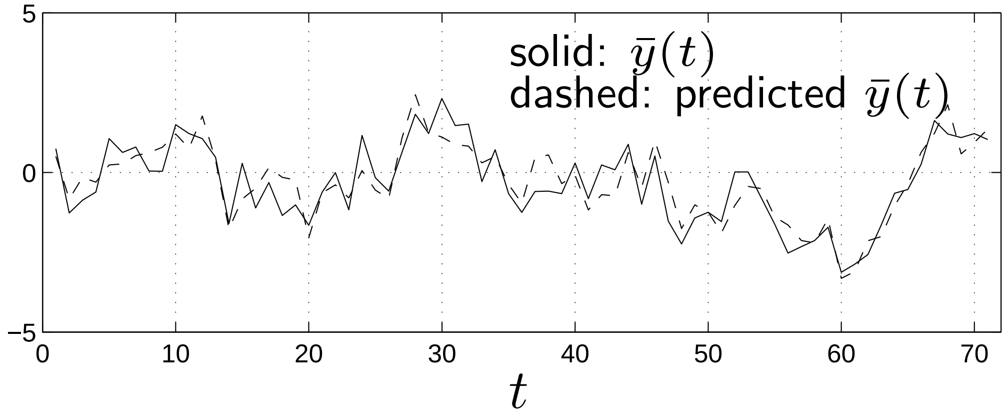
now check prediction error of models (developed using *modeling data*) on *validation data:*



plot suggests $n = 10$ is a good choice

for $n = 50$ the actual and predicted outputs on system identification and model validation data are:





loss of predictive ability when n too large is called *model overfit* or *overmodeling*

Growing sets of measurements

least-squares problem in 'row' form:

$$\text{minimize} \quad \|Ax - y\|^2 = \sum_{i=1}^{m} (\tilde{a}_i^T x - y_i)^2$$

where $\tilde{a}_i^T$ are the rows of A ($\tilde{a}_i \in \mathbf{R}^n$)

- $x \in \mathbf{R}^n$ is some vector to be estimated
- each pair $\tilde{a}_i$, y_i corresponds to one measurement
- solution is
$$x_{\mathrm{ls}} = \left(\sum_{i=1}^{m} \tilde{a}_i \tilde{a}_i^T \right)^{-1} \sum_{i=1}^{m} y_i \tilde{a}_i$$
- suppose that $\tilde{a}_i$ and y_i become available sequentially, $i.e.$, m increases with time

Recursive least-squares

we can compute $x_{\rm ls}(m) = \left(\sum_{i=1}^{m} \tilde{a}_i \tilde{a}_i^T\right)^{-1} \sum_{i=1}^{m} y_i \tilde{a}_i$ recursively

- initialize $P(0) = 0 \in \mathbf{R}^{n\times n}$, $q(0) = 0 \in \mathbf{R}^n$
- for $m = 0, 1, \ldots,$

$$P(m+1) = P(m) + \tilde{a}_{m+1}\tilde{a}_{m+1}^T \qquad q(m+1) = q(m) + y_{m+1}\tilde{a}_{m+1}$$

- if $P(m)$ is invertible, we have $x_{\rm ls}(m) = P(m)^{-1}q(m)$
- $P(m)$ is invertible $\Longleftrightarrow \tilde{a}_1, \ldots, \tilde{a}_m$ span $\mathbf{R}^n$
(so, once $P(m)$ becomes invertible, it stays invertible)

Fast update for recursive least-squares

we can calculate

$$P(m+1)^{-1} = \left(P(m) + \tilde{a}_{m+1}\tilde{a}_{m+1}^T\right)^{-1}$$

efficiently from $P(m)^{-1}$ using the *rank one update formula*

$$\left(P + \tilde{a}\tilde{a}^T\right)^{-1} = P^{-1} - \frac{1}{1 + \tilde{a}^T P^{-1}\tilde{a}}(P^{-1}\tilde{a})(P^{-1}\tilde{a})^T$$

valid when $P = P^T$, and P and $P + \tilde{a}\tilde{a}^T$ are both invertible

- gives an $O(n^2)$ method for computing $P(m+1)^{-1}$ from $P(m)^{-1}$
- standard methods for computing $P(m+1)^{-1}$ from $P(m+1)$ are $O(n^3)$

Verification of rank one update formula

$$
\begin{aligned}
&(P+\tilde{a}\tilde{a}^T)\left(P^{-1}-\frac{1}{1+\tilde{a}^TP^{-1}\tilde{a}}(P^{-1}\tilde{a})(P^{-1}\tilde{a})^T\right)\\
&= \quad I+\tilde{a}\tilde{a}^TP^{-1}-\frac{1}{1+\tilde{a}^TP^{-1}\tilde{a}}P(P^{-1}\tilde{a})(P^{-1}\tilde{a})^T\\
&\qquad\quad -\frac{1}{1+\tilde{a}^TP^{-1}\tilde{a}}\tilde{a}\tilde{a}^T(P^{-1}\tilde{a})(P^{-1}\tilde{a})^T\\
&= \quad I+\tilde{a}\tilde{a}^TP^{-1}-\frac{1}{1+\tilde{a}^TP^{-1}\tilde{a}}\tilde{a}\tilde{a}^TP^{-1}-\frac{\tilde{a}^TP^{-1}\tilde{a}}{1+\tilde{a}^TP^{-1}\tilde{a}}\tilde{a}\tilde{a}^TP^{-1}\\
&= \quad I
\end{aligned}
$$