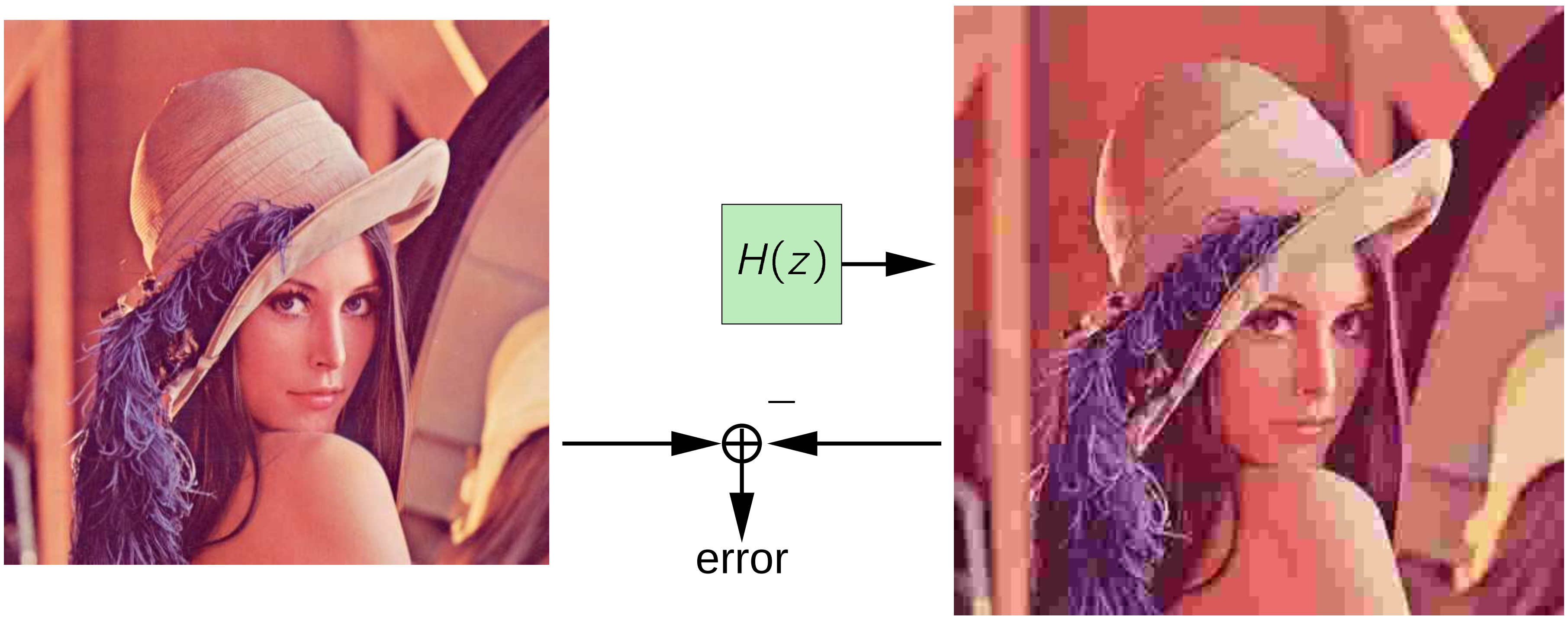
Chapter 4: Signal Modeling

Given a signal (set of samples), how can it be modeled using a filter?

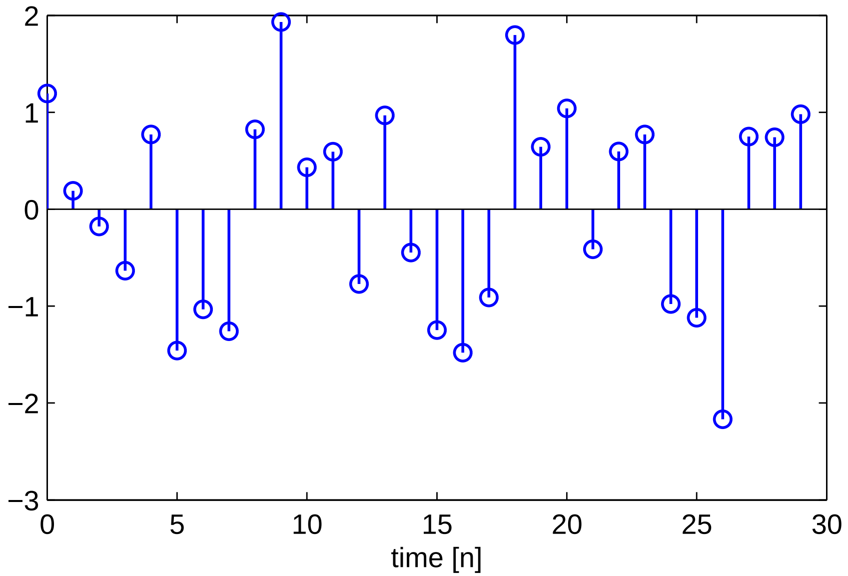


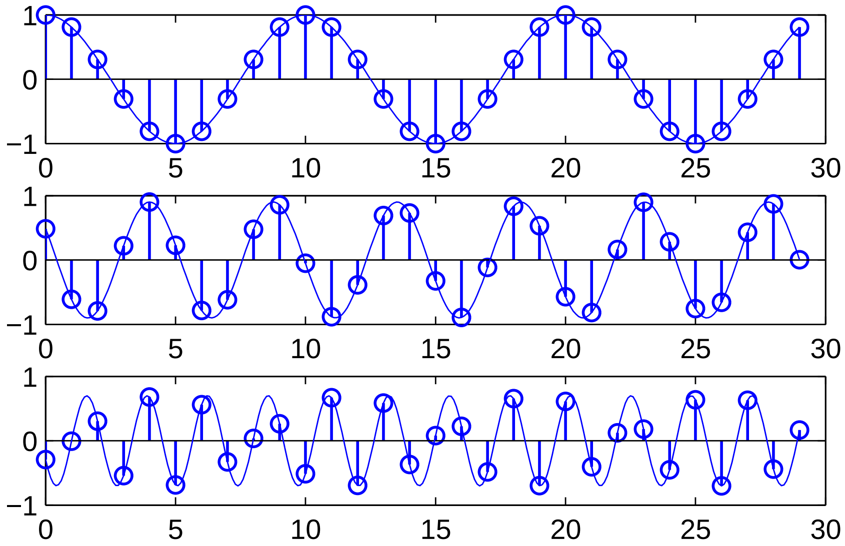
Original (bmp, 780 kB)

Parametrized model (jpg 10%, 5 kB)

Signal Modeling—Motivation

Motivation 1: Efficient transmission/storage



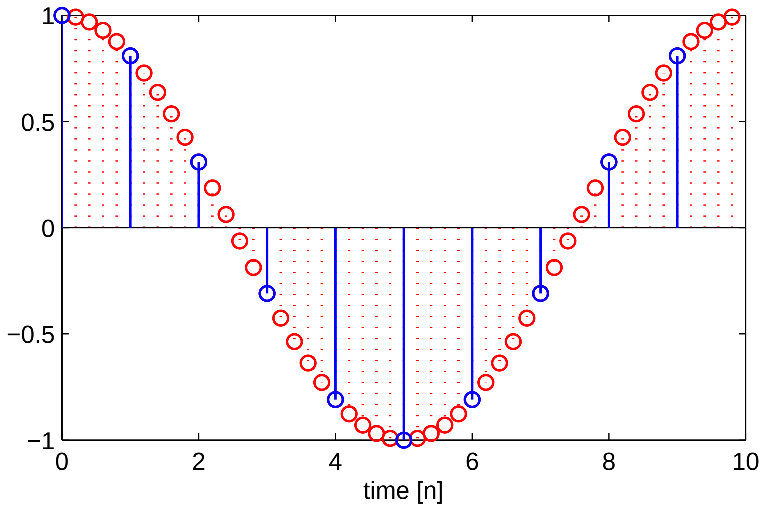


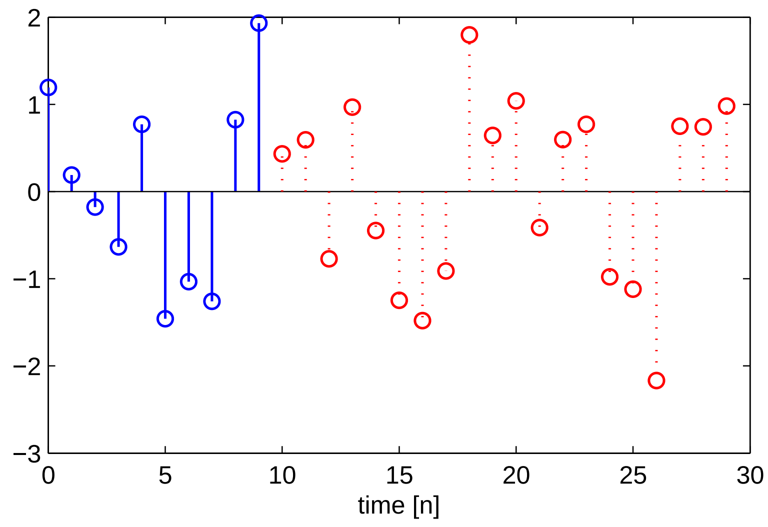
- Direct: store all samples
- Coded:
  - Model the signal, e.g., sum of sinusoids: $\sum \alpha_k \cos(n\omega_k + \phi_k)$
  - Estimate the parameters of the signal, $\{\alpha_k, \omega_k, \phi_k\}$
  - Store the parameters instead of the original samples.

Example: GSM speech coding, MP3 audio coding, JPEG image coding, $\cdots$

Motivation 2: Interpolation/extrapolation

Interpolation/extrapolation requires a model, e.g., bandlimited/lowpass, sum of sinusoids, etc.





Motivation

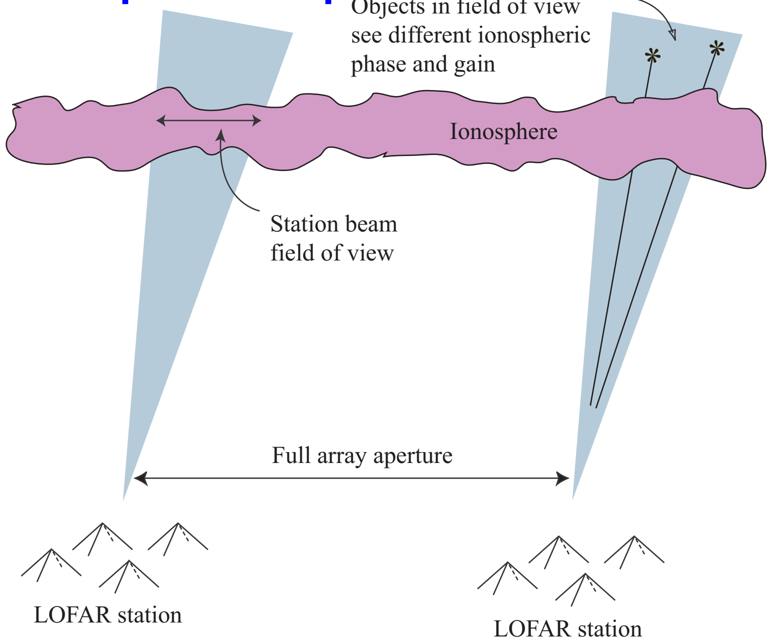
Extrapolation of Lena

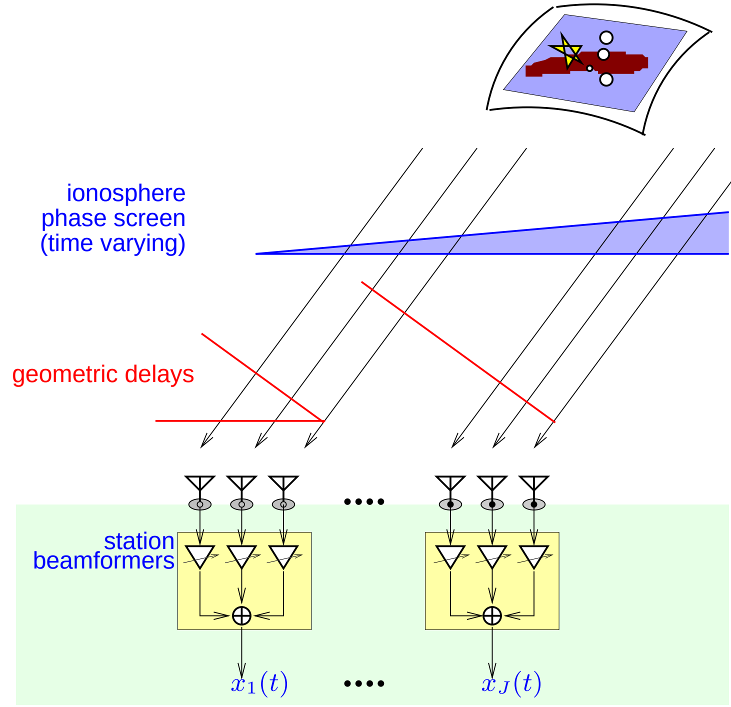
Horizontal extrapolation: 

(In fact, there are stationarity requirements that would prohibit this. . .)

Motivation

Example: ionospheric calibration



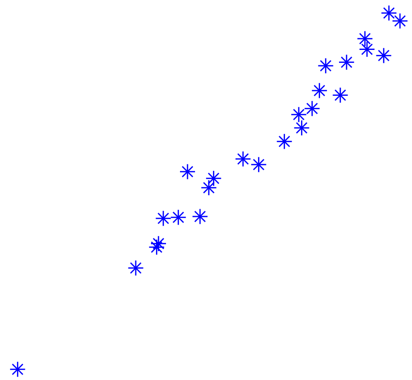


- The ionosphere causes small delays in the reception of signals that modify the apparent direction of astronomical sources.
- Low frequency radio telescopes can ‘sample’ the ionosphere in the direction of calibration sources. For other directions, we rely on interpolation.
- A simple ionospheric model specifies correlations in phase delay τ as function of distance between points: $\mathbf{C}_\tau(\mathbf{x}_1, \mathbf{x}_2) = 1 - \alpha\|\mathbf{x}_1 - \mathbf{x}_2\|^\beta$

Motivation

Example: Interpolation of correlated variables

Suppose we have samples of random variables (x, y) that are correlated. Given a new sample x, can we predict the corresponding y?



- Pose the model: $y = \alpha x$, and estimate α by minimizing $E|y - \alpha x|^2$:

$$E\{xy\} = \alpha E\{xx\} \quad \Leftrightarrow \quad r_{xy} = \alpha r_{xx} \quad \Leftrightarrow \quad \alpha = \frac{r_{xy}}{r_{xx}}$$

- If we stack the measured samples in vectors **x** and **y**, then we obtain the estimate

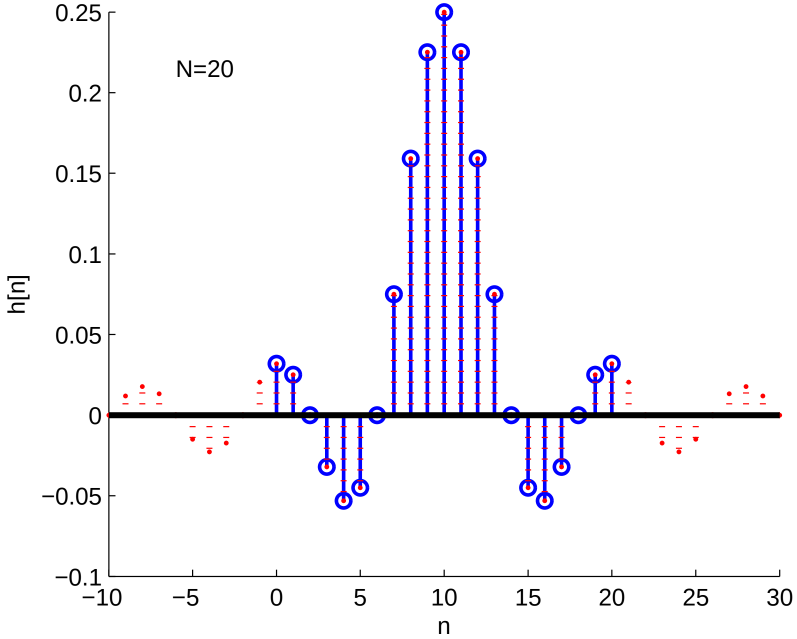
$$\hat{r}_{xx} = \frac{1}{N}\mathbf{x}^H\mathbf{x}\,, \qquad \hat{r}_{xy} = \frac{1}{N}\mathbf{x}^H\mathbf{y} \quad \Rightarrow \quad \hat{\alpha} = \frac{\mathbf{x}^H\mathbf{y}}{\mathbf{x}^H\mathbf{x}}$$

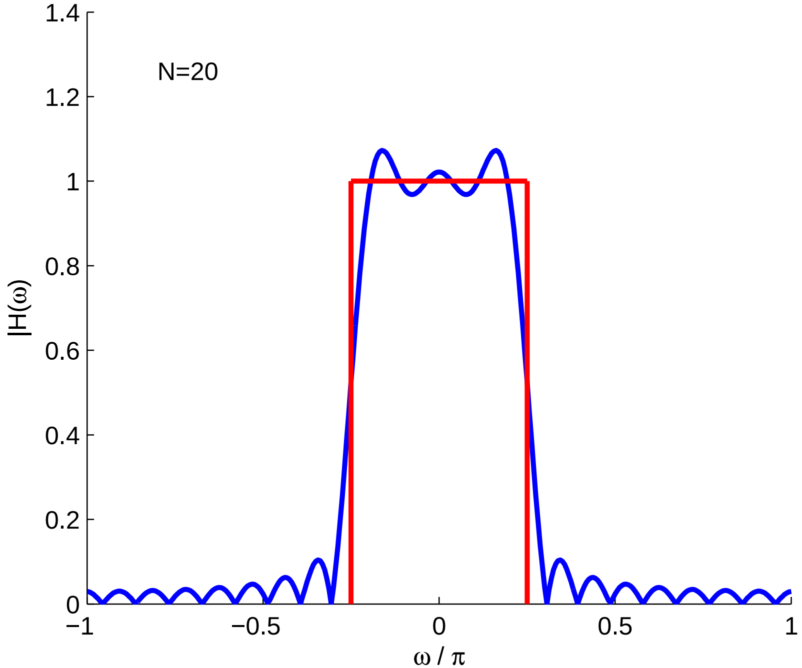
(cf. the Wiener filter in Ch. 7.) This is the solution of the Least Squares problem

$$\min_{\alpha} \|\mathbf{y} - \alpha\mathbf{x}\|^2$$

Motivation 3: Filter design

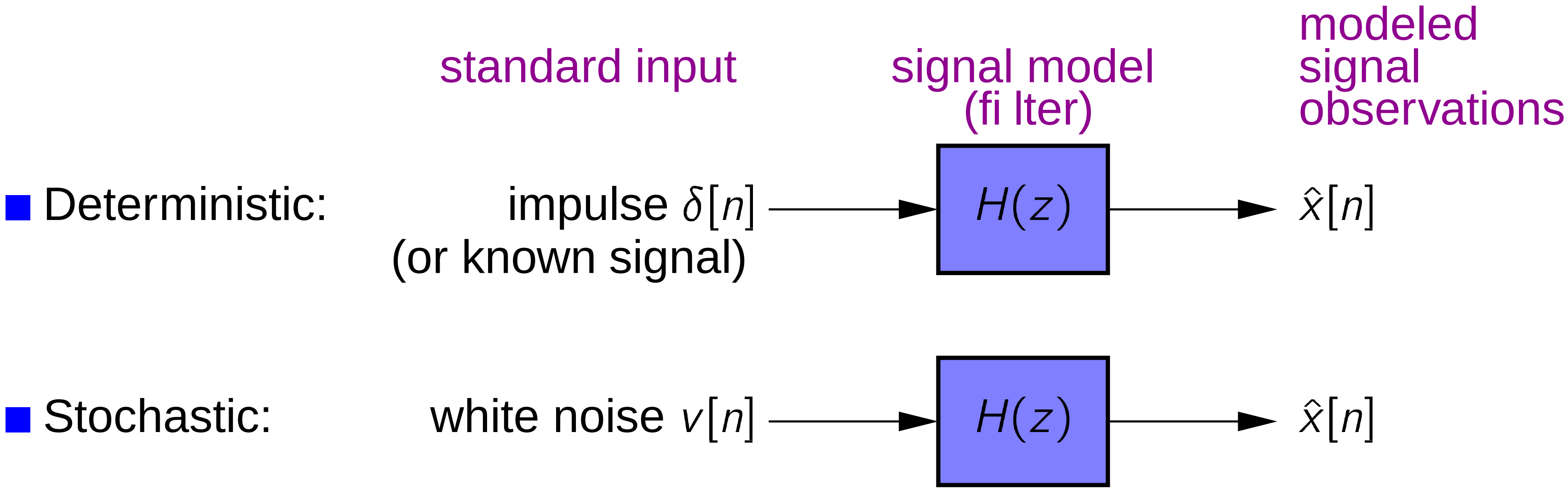
Design a filter that approximates a desired response (e.g., ideal lowpass)





The design depends on the filter model (FIR, IIR), filter order, error criteria, etc.

Signal models



Models for $H(z)$

ARMA(p, q): $$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{q} b(k) z^{-k}}{1 + \sum_{k=1}^{p} a(k) z^{-k}}$$

Special cases: AR(p), MA(q)

Signal models

Model identification

Given observations $x[n]$, $n = 0, \cdots, N-1$ and filter order p, q, find the parameters of $H(z)$ such that the modeled output signal $\hat{x}[n] = h[n]$ best matches the observations.

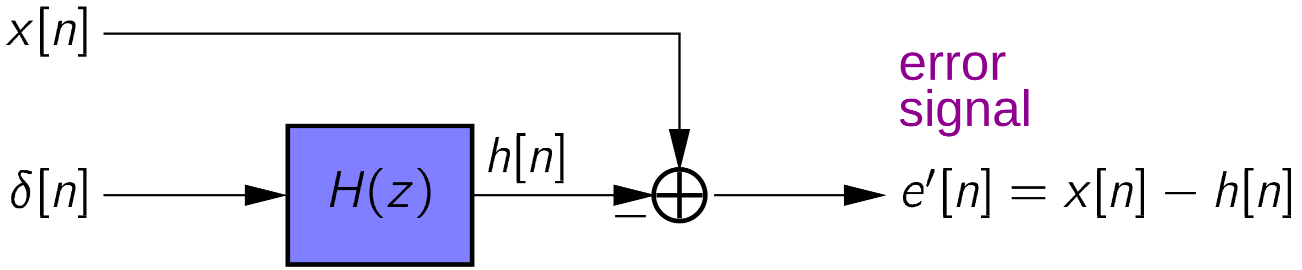
For stochastic signals, we will try to match the correlation sequences: $r_x[k] = r_h[k]$.

Issues:

- stability of the filter (in case $p > 0$)
- computational complexity for parameter estimation
- model order selection
- error criterion for the approximation

Model identification via Least Squares

In the next slides, we consider deterministic input signals (impulse $\delta[n]$). The model is a filter $H(z)$: LTI, causal, rational. The desired signal has $x[n] = 0$, $n < 0$.



- "Minimize the error" depends on the definition of error, and the norm.

  Least squares: $\mathcal{E}_{LS} = \sum_{n=0}^{\infty} |e'[n]|^2$

- The minimization of $\mathcal{E}_{LS} = \sum |x[n] - h[n]|^2$ with $H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{q} b[k]z^{-k}}{1+\sum_{k=1}^{p} a[k]z^{-k}}$ requires

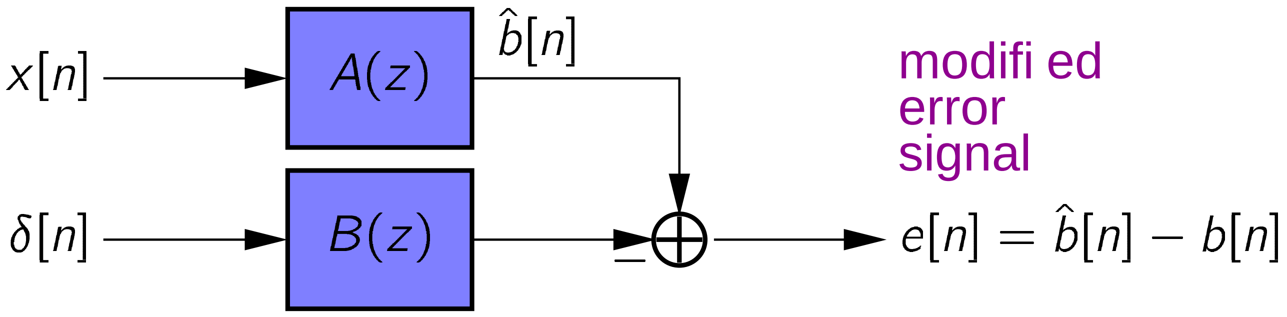
$$\begin{cases} \dfrac{\partial \mathcal{E}_{LS}}{\partial a^*[k]} = 0, & k = 1, \cdots, p \\ \dfrac{\partial \mathcal{E}_{LS}}{\partial b^*[k]} = 0, & k = 0, \cdots, q \end{cases}$$

The resulting $p + q + 1$ equations are nonlinear because of the division by $A(z)$.

Model identification via Least Squares

- Alternative that leads to tractable results: consider "weighted" error

$$E(z) = A(z)E'(z) = A(z)X(z) - B(z)$$



- Now $A(z)$ is in the nominator, the error equation is linear.
- Techniques that are based on this: Pade Approximation, Prony's Method, Shank's Method.

Model identification via Pade Approximation

Pade approximation

We have $p+q+1$ model parameters: Can match $p+q+1$ signal samples exactly:

$$h[n] = x[n], \quad n = 0, \cdots, p+q$$

- How to find $h[n]$ in terms of the parameters:

$$H(z) = \frac{B(z)}{A(z)} \quad \Rightarrow \quad H(z)A(z) = B(z) \quad \Rightarrow \quad h[n] * a[n] = b[n]$$

$$\Rightarrow \quad h[n] + \sum_{k=1}^{p} a[k]h[n-k] = b[n]$$

  where $h[n] = 0, \; n < 0$, and $b[n] = 0, \; n < 0$ or $n > q$

- Match exactly with $x[n]$ for $n = 0, \cdots, p+q$:

$$x[n] + \sum_{k=1}^{p} a[k]x[n-k] = \begin{cases} b[n]\,, & n = 0, \cdots, q \\ 0\,, & n = q+1, \cdots, p+q \end{cases}$$

Model identification via Pade Approximation

- Write these equations in matrix form:

$$\begin{bmatrix} x[0] & 0 & \cdots & 0 \\ x[1] & x[0] & \ddots & 0 \\ x[2] & x[1] & \ddots & x[0] \\ \vdots & \vdots & \ddots & \vdots \\ x[q] & x[q-1] & \ddots & x[q-p] \\ \hline x[q+1] & x[q] & \ddots & x[q-p+1] \\ \vdots & \vdots & \ddots & \vdots \\ x[q+p] & x[q+p-1] & & x[q] \end{bmatrix} \begin{bmatrix} 1 \\ a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = \begin{bmatrix} b[0] \\ b[1] \\ b[2] \\ \vdots \\ b[q] \\ \hline 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Take the submatrix that does not involve the $b[k]$, and first solve for the $a[k]$:

$$\begin{bmatrix} x[q] & x[q-1] & \ddots & x[q-p+1] \\ x[q+1] & x[q] & \ddots & x[q-p+2] \\ \ddots & \ddots & \ddots & \vdots \\ x[q+p-1] & x[q+p-2] & & x[q] \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} x[q+1] \\ x[q+2] \\ \vdots \\ x[q+p] \end{bmatrix}$$

This is a square $p \times p$ matrix equation: $\mathbf{X}_q \bar{\mathbf{a}} = -\mathbf{x}_{q+1}$.

The solution is $\bar{\mathbf{a}} = -\mathbf{X}_q^{-1} \mathbf{x}_{q+1}$.

Model identification via Pade Approximation

- Plug the solution back to find the $b[k]$:

$$\begin{bmatrix} b[0] \\ b[1] \\ b[2] \\ \vdots \\ b[q] \end{bmatrix} = \begin{bmatrix} x[0] & 0 & \cdots & 0 \\ x[1] & x[0] & & 0 \\ x[2] & x[1] & & x[0] \\ \vdots & \vdots & \ddots & \vdots \\ x[q] & x[q-1] & \ddots & x[q-p] \end{bmatrix} \begin{bmatrix} 1 \\ a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix}$$

$$\mathbf{b} = \mathbf{X}_0 \mathbf{a}$$

Disadvantages of the Pade approximation

- The resulting model doesn't have to be stable
- Outside the used interval $[0, p+q]$, the approximation may be very bad
- If $\mathbf{X}_q$ is singular, then a solution does not always exist that matches all signal values. (Sometimes a solution exists of lower order for $A(z)$, $B(z)$, i.e., smaller p, q, but then no guarantee on matching for samples beyond $p+q$.)

Model identification via Prony's Method

Derivation of Prony's Method

As before, $E(z) = X(z)A(z) - B(z)$, i.e.,

$$e[n] = \begin{cases} x[n] + \sum_{k=1}^{p} a[k]x[n-k] - b[n]\,, & n = 0, \cdots, q \\ x[n] + \sum_{k=1}^{p} a[k]x[n-k]\,, & n > q \end{cases}$$

- For Pade, we first solved $e[n] = 0, \;\; n = q+1, \cdots, q+p$ to find $A(z)$.
- Prony: solve $\min\limits_{\{a[k]\}} \sum\limits_{n=q+1}^{\infty} |e[n]|^2 = \min\limits_{\{a[k]\}} \sum\limits_{n=q+1}^{\infty} |x[n] + \sum\limits_{k=1}^{p} a[k]x[n-k]|^2$

  In matrix form:

$$\min \left\| \begin{bmatrix} x[q+1] & x[q] & x[q-1] & \cdots & x[q-p+1] \\ x[q+2] & x[q+1] & x[q] & \ddots & x[q-p+2] \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ x[q+p] & x[q+p-1] & x[q+p-2] & \cdots & x[q] \\ \hline x[q+p+1] & x[q+p] & \ddots & \ddots & x[q+1] \\ x[q+p+2] & x[q+p+1] & \ddots & \ddots & x[q+2] \\ \vdots & \vdots & \ddots & \ddots & \vdots \end{bmatrix} \begin{bmatrix} 1 \\ a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} \right\|^2$$

Model identification via Prony's Method

$$\min \left\| \begin{bmatrix} x[q] & x[q-1] & \ddots & x[q-p+1] \\ x[q+1] & x[q] & \ddots & x[q-p+2] \\ \ddots & \ddots & \ddots & \vdots \\ x[q+p-1] & x[q+p-2] & \ddots & x[q] \\ x[q+p] & \ddots & \ddots & x[q+1] \\ x[q+p+1] & \ddots & \ddots & x[q+2] \\ \vdots & \ddots & \ddots & \vdots \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} + \begin{bmatrix} x[q+1] \\ x[q+2] \\ \vdots \\ x[q+p] \\ x[q+p+1] \\ x[q+p+2] \\ \vdots \end{bmatrix} \right\|^2$$

$$\min \|\mathbf{X}_q \bar{\mathbf{a}} - (-\mathbf{x}_{q+1})\|^2$$

(Now, $\mathbf{X}_q$ refers to the infinite-dimensional matrix.)

- This is a Least-Squares problem of an overdetermined system of equations.
- The solution is $\bar{\mathbf{a}} = -\mathbf{X}_q^{\dagger}\mathbf{x}_{q+1} = -(\mathbf{X}_q^H\mathbf{X}_q)^{-1}\mathbf{X}_q^H\mathbf{x}_{q+1} = -\mathbf{R}_x^{-1}\mathbf{r}_x$.

  This assumes that $\mathbf{R}_x := \mathbf{X}_q^H\mathbf{X}_q$ is not singular.
- In practice, $\mathbf{X}_q$ and $\mathbf{x}_{q+1}$ are of finite size as determined by the available data.

Model identification via Prony's Method

$$\mathbf{R}_x := \mathbf{X}_q^H \mathbf{X}_q \qquad (p \times p \text{ matrix})$$

- $\mathbf{R}_x$ is positive (semi)definite by construction. We will see later that this makes $A(z)$ (marginally) stable.
- If $\mathbf{R}_x$ is singular, then this indicates that the filter order can be reduced.
- After **a** is known, we find **b** (i.e., $B(z)$) precisely as in Pade's method: $\mathbf{b} = \mathbf{X}_0\mathbf{a}$. This makes $e[n] = 0, \; n = 0, \cdots, q$.

Alternatively, we can find the numerator by minimizing $e[n]$ over the entire data record. For the error criterion based on $e[n]$, this does not make a difference. Shank's method switches back to $e'[n]$ and minimizes over the entire data.

Example: filter design

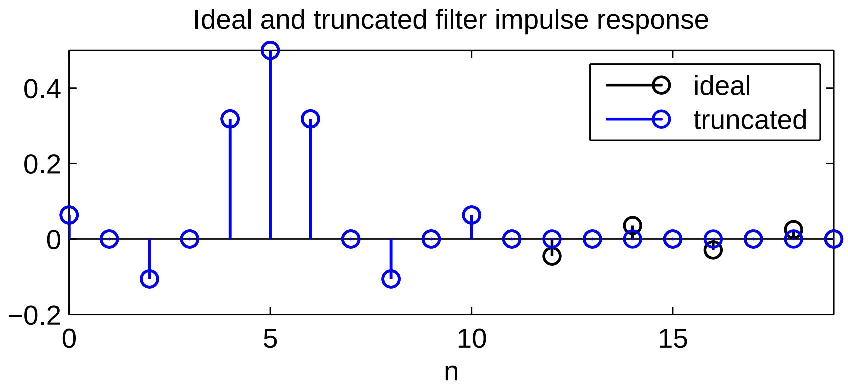
Suppose we want to design an ideal linear phase lowpass filter:

$$G(e^{j\omega}) = \begin{cases} e^{-jn_d\omega}, & |\omega| < \pi/2 \\ 0, & \text{otherwise} \end{cases}$$

n_d is the filter delay. The corresponding impulse response is

$$g[n] = \frac{\sin[(n - n_d)\pi/2]}{(n - n_d)\pi}$$

- We will match $p + q + 1 = 11$ values, and choose $n_d = 5$.



- Two cases: FIR filter ($p = 0$, $q = 10$), ARMA filter ($p = 5$, $q = 5$).

Example: filter design

Pade approximation

- Filter coefficients to match:

$$\mathbf{g} = \Big[\ 0.064 \quad -0.000 \quad -0.106 \quad 0.000 \quad 0.318 \quad 0.5 \quad 0.318 \quad 0.000 \quad -0.106 \quad -0.000 \quad 0.064$$

- FIR filter ($p = 0$, $q = 10$):

$$h[n] = \begin{cases} g[n], & 0 \leq n \leq 10 \\ 0, & \text{otherwise} \end{cases}$$

- ARMA filter ($p = q = 5$): solving the Pade equations for $\mathbf{a}$ gives

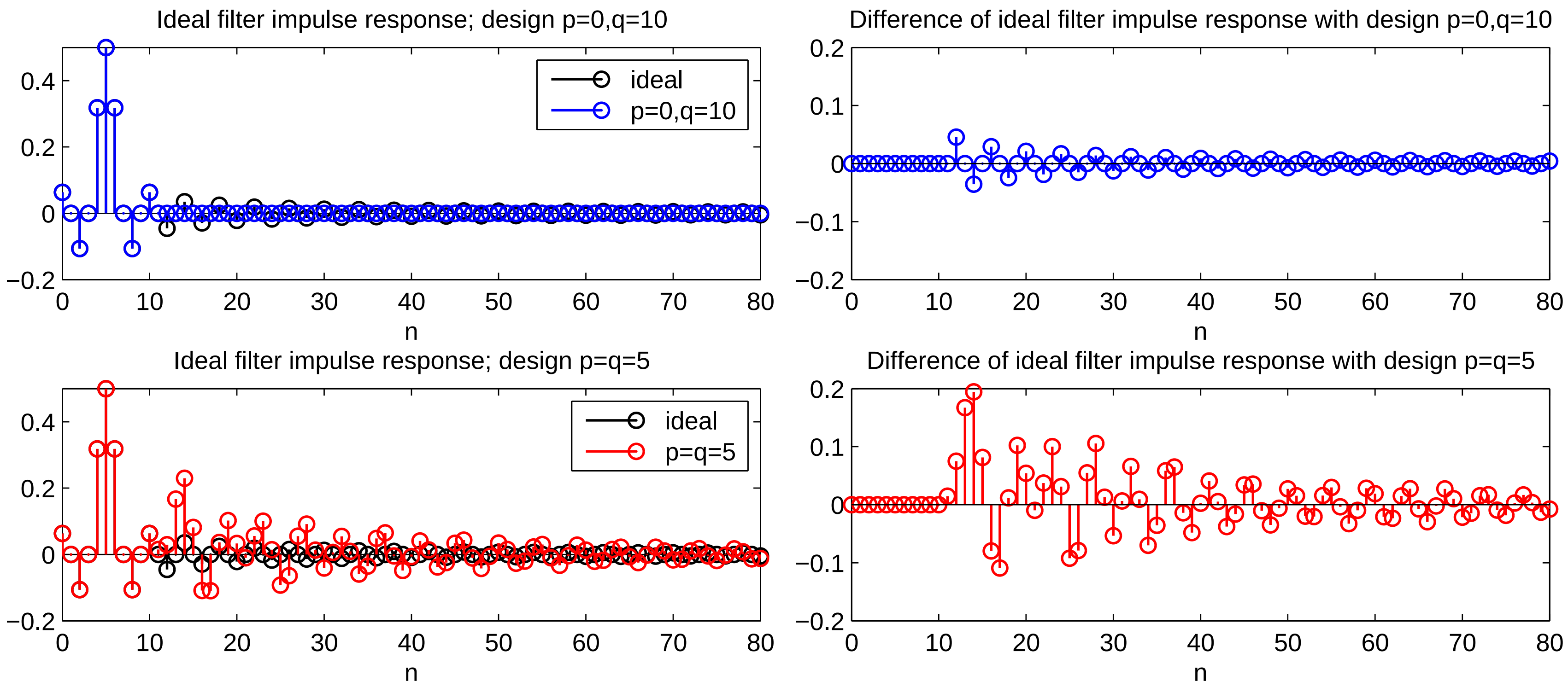
$$\mathbf{a} = \begin{bmatrix} 1.000 & -2.526 & 3.677 & -3.485 & 2.131 & -0.703 \end{bmatrix}$$

  and subsequently the numerator coefficients are found as

$$\mathbf{b} = \begin{bmatrix} 0.064 & -0.161 & 0.128 & 0.046 & 0.064 & 0.021 \end{bmatrix}$$

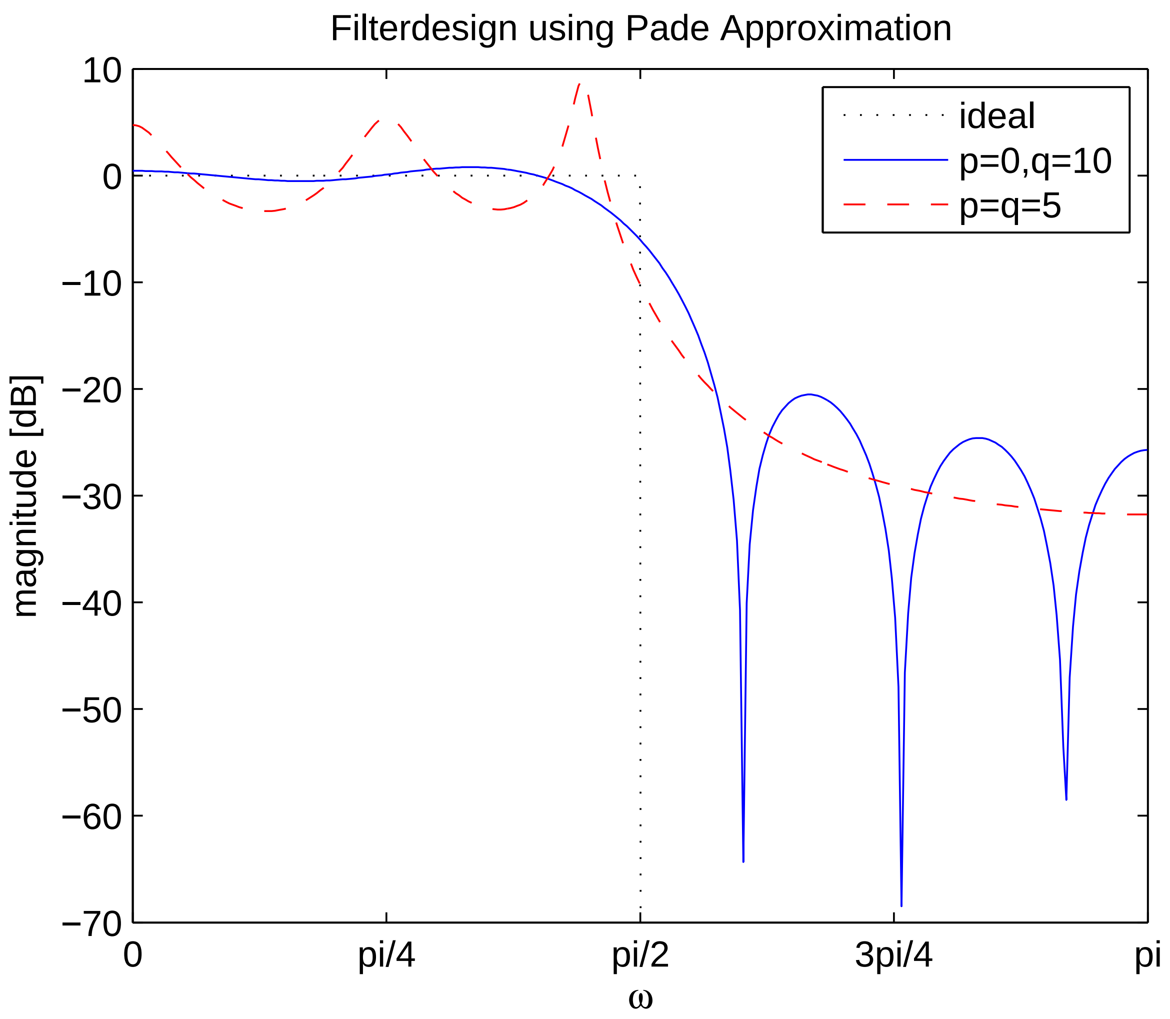
(Use Matlab `[b,a] = prony(g,q,p)` to find the design)

Example: filter design



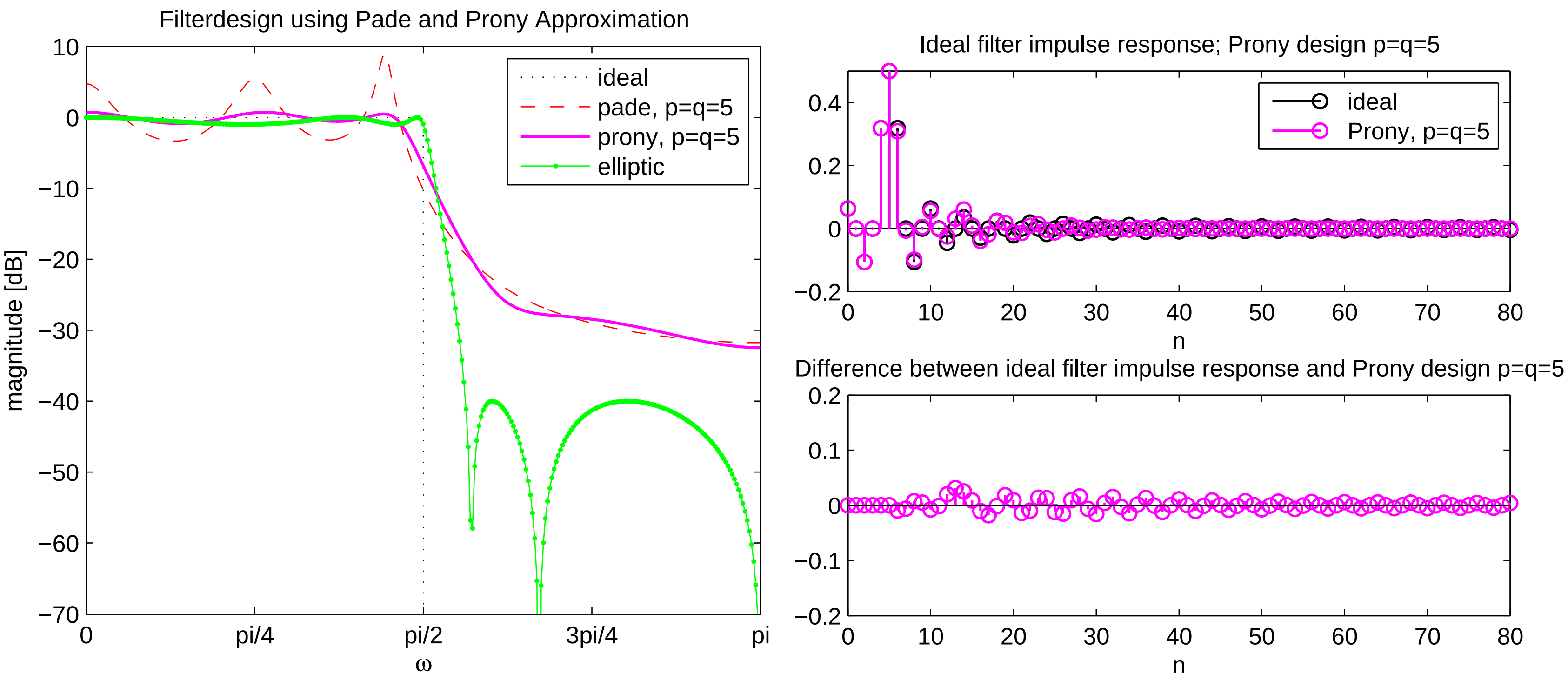
- The ARMA(5,5) filter gives a larger error outside the specified interval. Also, this filter is not linear phase (no symmetry).

Example: filter design



- The ARMA(5,5) filter does not have a good frequency response in the passband.

Example: filter design



- The ARMA(5,5) design using Prony's method is much better than the Pade approximation, since $A(z)$ is designed to minimize the error over the entire domain
- A specialistic design (elliptic filter of 5th order) can still be better, with full control over the passband error and stopband attenuation.

Model identification via Prony's Method

Alternative writing of the equations

Recall: The solution is $\bar{\mathbf{a}} = -\mathbf{R}_x^{-1}\mathbf{r}_x$ with

$$\mathbf{R}_x = \mathbf{X}_q^H \mathbf{X}_q \,, \qquad \mathbf{r}_x = \mathbf{X}_q^H \mathbf{x}_{q+1}$$

Thus, the entries of $\mathbf{R}_x$ and $\mathbf{r}_x$ are

$$r_x(k,\ell) := [\mathbf{R}_x]_{k,\ell} = \sum_{n=q+1}^{\infty} x^*[n-k]x[n-\ell] \,, \qquad r_x(k,0) := [\mathbf{r}_x]_k = \sum_{n=q+1}^{\infty} x^*[n-k]x[n]$$

The equation $\mathbf{R}_x\bar{\mathbf{a}} = -\mathbf{r}_x$ can be written as

$$\begin{bmatrix} r_x(1,1) & r_x(1,2) & \cdots & r_x(1,p) \\ r_x(2,1) & r_x(2,2) & \cdots & r_x(2,p) \\ \vdots & \ddots & & \vdots \\ r_x(p,1) & r_x(p,2) & \cdots & r_x(p,p) \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} r_x(1,0) \\ r_x(2,0) \\ \vdots \\ r_x(p,0) \end{bmatrix}$$

The entries of $\mathbf{R}_x$ are considered *autocorrelations*. In a stochastic context (discussed later), we arrive at very similar equations.

Model identification via Prony's Method

- These equations can also be summarized as

$$\sum_{\ell=1}^{p} a[\ell] r_x(k,\ell) = -r_x(k,0), \quad k = 1, \cdots, p$$

  These are called the *Prony normal equations*.

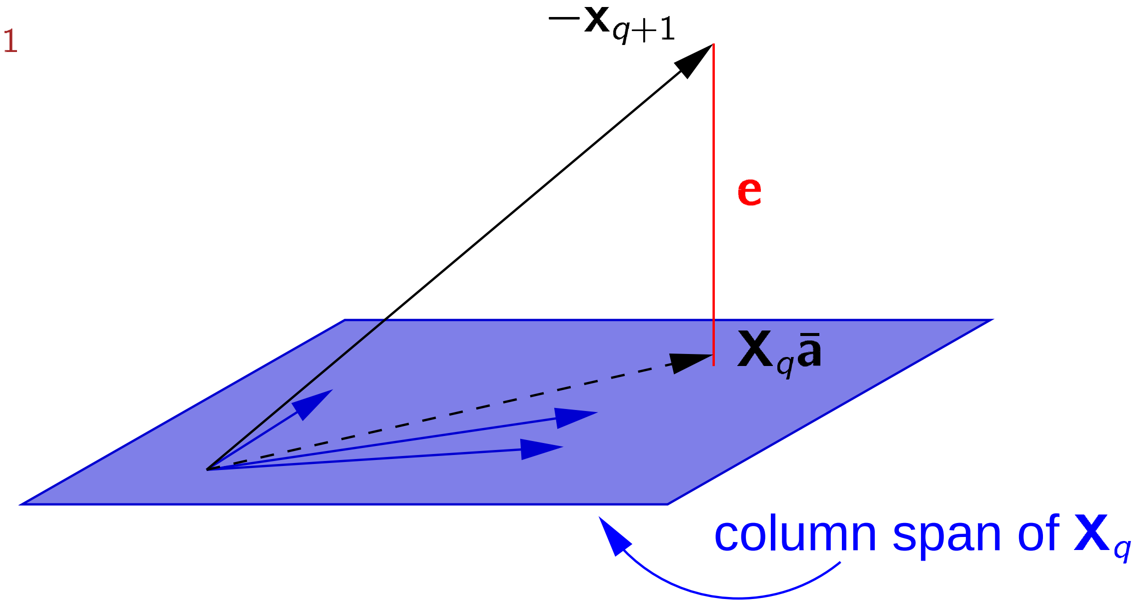
- Yet another writing form: $r_x(k,0) + \sum_{\ell=1}^{p} a[\ell] r_x(k,\ell) = 0, \quad k = 1, \cdots, p$

$$\begin{bmatrix} r_x(1,0) & r_x(1,1) & r_x(1,2) & \cdots & r_x(1,p) \\ r_x(2,0) & r_x(2,1) & r_x(2,2) & \cdots & r_x(2,p) \\ \vdots & \vdots & \ddots & & \vdots \\ r_x(p,0) & r_x(p,1) & r_x(p,2) & \cdots & r_x(p,p) \end{bmatrix} \begin{bmatrix} 1 \\ a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Model identification via Prony's Method

Orthogonality principle

Consider the Least Squares problem $\mathbf{X}_q\bar{\mathbf{a}} \approx -\mathbf{x}_{q+1}$. After solving, the error vector is $\mathbf{e} = \mathbf{X}_q\bar{\mathbf{a}} + \mathbf{x}_{q+1}$



The orthogonality principle states, in general, that the error vector for the optimal $\bar{\mathbf{a}}$ is orthogonal to all the columns of $\mathbf{X}_q$, i.e., $\mathbf{e} \perp \mathrm{colspan}(\mathbf{X}_q)$. If it was not the case, we could take a linear combination of these columns to reduce the error!

$$\mathbf{X}_q^H\mathbf{e} = 0 \quad \Leftrightarrow \quad \mathbf{X}_q^H(\mathbf{X}_q\bar{\mathbf{a}} + \mathbf{x}_{q+1}) = 0 \quad \Leftrightarrow \quad \mathbf{R}_x\bar{\mathbf{a}} = -\mathbf{r}_x$$

Model identification via Prony's Method

The minimum error

- At the minimum, the total error is (due to the orthogonality principle)

$$\epsilon_{p,q} = \|\mathbf{e}\|^2 = \mathbf{e}^H\mathbf{e} = (\mathbf{X}_q\bar{\mathbf{a}} + \mathbf{x}_{q+1})^H\mathbf{e} = \mathbf{x}_{q+1}^H\mathbf{e}$$

  This can further be written as

$$\epsilon_{p,q} = \mathbf{x}_{q+1}^H(\mathbf{X}_q\bar{\mathbf{a}} + \mathbf{x}_{q+1}) = (\mathbf{x}_{q+1}^H\mathbf{X}_q)\bar{\mathbf{a}} + \mathbf{x}_{q+1}^H\mathbf{x}_{q+1}$$

- In terms of the autocorrelation sequence,

$$r_x(0,0) = \sum_{n=q+1}^{\infty} x^*[n]x[n] = \mathbf{x}_{q+1}^H\mathbf{x}_{q+1}\,, \qquad r_x(0,k) = \sum_{n=q+1}^{\infty} x^*[n]x[n-k] = [\mathbf{x}_{q+1}^H\mathbf{X}_q]_k$$

  this can be written as

$$\epsilon_{p,q} = r_x(0,0) + \sum_{k=1}^{p} a[k]r_x(0,k)$$

Model identification via Prony's Method

This can be combined with the equation $\mathbf{R}_x\bar{\mathbf{a}} = -\mathbf{r}_x$ as

$$\left[\begin{array}{c|cccc} r_x(0,0) & r_x(0,1) & r_x(0,2) & \cdots & r_x(0,p) \\ \hline r_x(1,0) & r_x(1,1) & r_x(1,2) & \cdots & r_x(1,p) \\ r_x(2,0) & r_x(2,1) & r_x(2,2) & \cdots & r_x(2,p) \\ \vdots & \ddots & & \vdots & \\ r_x(p,0) & r_x(p,1) & r_x(p,2) & \cdots & r_x(p,p) \end{array}\right] \left[\begin{array}{c} 1 \\ \hline a[1] \\ a[2] \\ \vdots \\ a[p] \end{array}\right] = \left[\begin{array}{c} \epsilon_{p,q} \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right]$$

or also as

$$\mathbf{R}_x\mathbf{a}_p = \epsilon_{p,q}\mathbf{u}_1$$

where $\mathbf{u}_1 = [1, 0 \cdots 0]^T$ is a unit vector. These are the *augmented normal equations*.

All-pole modeling

Special case: all-pole modeling

If $q = 0$, then we have an all-pole model:

$$H(z) = \frac{b[0]}{1 + \sum_{k=1}^{p} a[k] z^{-k}}$$

- In some cases, this is an accurate physical model (e.g., speech)
- Even if it is not a valid physical model, it is attractive because it leads to fast computational algorithms to find $a[k]$ (the Levinson algorithm)

Recall that the solution is given by solving the overdetermined system ($q = 0$)

$$\begin{bmatrix} x[0] & 0 & 0 & \cdots & 0 \\ x[1] & x[0] & 0 & \cdots & 0 \\ x[2] & x[1] & x[0] & \ddots & 0 \\ x[3] & x[2] & x[1] & \ddots & x[0] \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ a[3] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ \vdots \\ \vdots \end{bmatrix}$$

All-pole modeling

The normal equations become (premultiply with $\mathbf{X}_0^H$)

$$\begin{bmatrix} x^*[0] & x^*[1] & x^*[2] & x^*[3]\cdots \\ 0 & \ddots & \ddots & \ddots \;\cdots \\ 0 & 0 & x^*[0] & x^*[1]\cdots \end{bmatrix} \begin{bmatrix} x[0] & 0 & 0 \\ x[1] & \ddots & 0 \\ x[2] & \ddots & x[0] \\ x[3] & \ddots & x[1] \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} a[1] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} x^*[0] & x^*[1] & x^*[2] & x^*[3]\cdots \\ 0 & \ddots & \ddots & \ddots \;\cdots \\ 0 & 0 & x^*[0] & x^*[1]\cdots \end{bmatrix} \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ \vdots \end{bmatrix}$$

From this structure it is seen that the normal equations become

$$\begin{bmatrix} r_x(0) & r_x^*(1) & \cdots & r_x^*(p-1) \\ r_x(1) & r_x(0) & \cdots & r_x^*(p-2) \\ \vdots & \ddots & & \vdots \\ r_x(p-1) & r_x(p-2) & \cdots & r_x(0) \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} r_x(1) \\ r_x(2) \\ \vdots \\ r_x(p) \end{bmatrix}$$

where the autocorrelation sequence is in fact “stationary”:

$$r_x(k-\ell) = r_x(k,\ell) = \sum_{n=0}^{\infty} x[n-k]^* x[n-\ell]$$

All-pole modeling

- For an all-pole model, the matrix $\mathbf{R}_x$ has a Toeplitz structure (constant along diagonals); it can be efficiently inverted using the Levinson algorithm (later in Ch.5).

- The numerator will be chosen as $b[0] = \sqrt{\epsilon_p}$. It can be shown that then

$$r_x(k) = r_h(k)\,, \qquad |k| \leq p$$

  Meaning: the autocorrelation sequence of the filter matches the first p lags of the autocorrelation sequence of the specified data (“moment matching”).

All-pole modeling with finite data

Suppose we have only $N+1$ samples, $x[0], \cdots, x[N]$. What changes?

Autocorrelation method

Here, the entire data set is used, and extended with zeros:

$$\begin{bmatrix} x[0] & 0 & 0 \\ x[1] & \ddots & 0 \\ \vdots & \ddots & x[0] \\ x[N] & \ddots & x[1] \\ 0 & \ddots & \vdots \\ 0 & 0 & x[N] \end{bmatrix} \begin{bmatrix} a[1] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} x[1] \\ \vdots \\ x[N] \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

We will have the same normal equations as before, but with a different $r_x(k)$:

$r_x(k) = \sum_{n=k}^{N} x[n]x^*[n-k], \;\; k = 0, \cdots, p.$

$$\begin{bmatrix} r_x(0) & \cdots & r_x^*(p-1) \\ \vdots & & \vdots \\ r_x(p-1) & \cdots & r_x(0) \end{bmatrix} \begin{bmatrix} a[1] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} r_x(1) \\ \vdots \\ r_x(p) \end{bmatrix}$$

All-pole modeling with finite data

Covariance method

If we know that $x[n]$ is not zero for $n < 0$ and $n > N$, it is more accurate to omit those equations that contain an extension with zeros:

$$\begin{bmatrix} x[p-1] & \ddots & x[0] \\ x[p] & \ddots & x[1] \\ \vdots & \ddots & \vdots \\ x[N-1] & \ddots & x[N-p] \end{bmatrix} \begin{bmatrix} a[1] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} x[p] \\ x[p+1] \\ \vdots \\ x[N] \end{bmatrix}$$

We will have slightly different normal equations as before, with $r_x(k, \ell)$ defined as $r_x(k, \ell) = \sum_{n=p}^{N} x[n-\ell]x^*[n-k], \; k, \ell = 0, \cdots, p.$

$$\begin{bmatrix} r_x(1,1) & \cdots & r_x(1,p) \\ \vdots & & \vdots \\ r_x(p,1) & \cdots & r_x(p,p) \end{bmatrix} \begin{bmatrix} a[1] \\ \vdots \\ a[p] \end{bmatrix} = - \begin{bmatrix} r_x(1,0) \\ \vdots \\ r_x(p,0) \end{bmatrix}$$

The Toeplitz property is lost.

Example: pole estimation

Consider the finite data sequence $\mathbf{x} = [1, \beta, \beta^2, \cdots, \beta^N]^T$. Estimate an AR(1) model for this signal ($p = 1$):

$$H(z) = \frac{b(0)}{1 + a[1]z^{-1}}$$

Autocorrelation method

The normal equations collapse to $r_x(0)a[1] = -r_x(1)$, where

$$r_x(0) = [1, \beta, \beta^2, \cdots, \beta^N] \begin{bmatrix} 1 \\ \beta \\ \beta^2 \\ \vdots \\ \beta^N \end{bmatrix} = \sum_{n=0}^{N} |\beta|^{2n} = \frac{1 - |\beta|^{2N+2}}{1 - |\beta|^2}$$

$$r_x(1) = [0, 1, \beta, \cdots, \beta^N] \begin{bmatrix} 1 \\ \beta \\ \vdots \\ \beta^N \\ 0 \end{bmatrix} = \sum_{n=0}^{N-1} \beta|\beta|^{2n} = \beta \frac{1 - |\beta|^{2N}}{1 - |\beta|^2}$$

$$\Rightarrow \quad a[1] = -\beta \frac{1 - |\beta|^{2N}}{1 - |\beta|^{2N+2}}$$

Example: pole estimation

Covariance method

The normal equations collapse to $r_x(1,1)a[1] = -r_x(1,0)$, where

$$r_x(1,1) = [1, \beta, \beta^2, \cdots, \beta^{N-1}] \begin{bmatrix} 1 \\ \beta \\ \beta^2 \\ \vdots \\ \beta^{N-1} \end{bmatrix} = \sum_{n=0}^{N-1} |\beta|^{2n} = \frac{1-|\beta|^{2N}}{1-|\beta|^2}$$

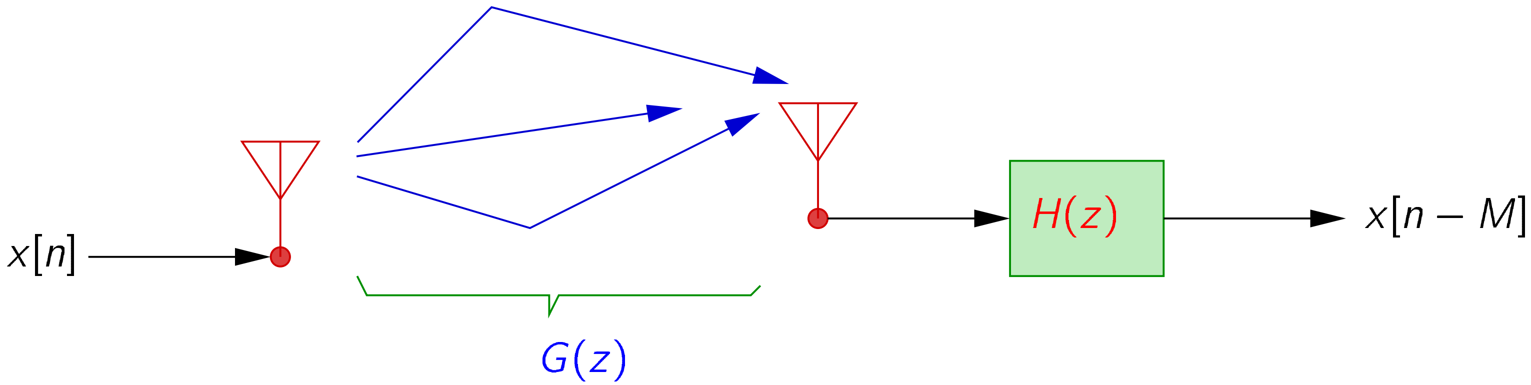
$$r_x(1,0) = [1, \beta, \cdots, \beta^{N-1}] \begin{bmatrix} \beta \\ \beta^2 \\ \vdots \\ \beta^{N} \end{bmatrix} = \sum_{n=0}^{N-1} \beta|\beta|^{2n} = \beta\frac{1-|\beta|^{2N}}{1-|\beta|^2}$$

Thus, $a[1] = -\beta$. Set $b(0) = 1$, then

$$H(z) = \frac{1}{1-\beta z^{-1}}$$

The pole location is found exactly.

Example: channel inversion



Consider a communication channel with a known (estimated) transfer function $G(z)$.

- At the receiver, we wish to equalize (invert) the channel using an equalizer $H(z)$:

$$G(z)H(z) = 1 \qquad \Leftrightarrow \qquad g[n] * h[n] = \delta[n]$$

  Further $H(z)$ must be causal and stable, typically FIR.

- If $G(z)$ is not minimum-phase (has zeros outside the unit circle), causal inversion is not possible, and we allow for a delay:

$$G(z)H(z) = z^{-M} \qquad \Leftrightarrow \qquad g[n] * h[n] = \delta[n-M] =: d[n]$$

Example: channel inversion

Design of an FIR equalizer $H(z)$ of length N:

$$e[n] = d[n] - h[n] * g[n] = d[n] - \sum_{\ell=0}^{N-1} h[\ell] g[n-\ell]$$

- Minimize $\mathcal{E} = \sum_{n=0}^{\infty} |e[n]|^2 = \sum_{n=0}^{\infty} |d[n] - \sum_{\ell=0}^{N-1} h[\ell] g[n-\ell]|^2$:

$$\min_{\{h[\ell]\}} \left\| \begin{bmatrix} g[0] & 0 & 0 \\ g[1] & \ddots & 0 \\ \vdots & \ddots & g[0] \\ g[N] & \ddots & g[1] \\ \vdots & \ddots & \vdots \end{bmatrix} \begin{bmatrix} h[0] \\ \vdots \\ h[N-1] \end{bmatrix} - \begin{bmatrix} d[0] \\ \vdots \\ d[N-1] \\ d[N] \\ \vdots \end{bmatrix} \right\|^2$$

The LS solution of this overdetermined system $\mathbf{Gh} = \mathbf{d}$ is $\mathbf{h} = \mathbf{G}^\dagger \mathbf{d} = (\mathbf{G}^H\mathbf{G})^{-1}\mathbf{G}^H\mathbf{d}$.

- Also, the corresponding normal equations are $(\mathbf{G}^H\mathbf{G})\mathbf{h} = \mathbf{G}^H\mathbf{d}$, or

$$\begin{bmatrix} r_g(0) & \cdots & r_g^*(N-1) \\ \vdots & & \vdots \\ r_g(N-1) & \cdots & r_g(0) \end{bmatrix} \begin{bmatrix} h[0] \\ \vdots \\ h[N-1] \end{bmatrix} = - \begin{bmatrix} r_{dg}(0) \\ \vdots \\ r_{dg}(N-1) \end{bmatrix}$$

where $r_g(k) = \sum_{n=0}^{\infty} g[n]g^*[n-k]$ and $r_{dg}(k) = \sum_{n=0}^{\infty} d[n]g^*[n-k]$. Because the matrix $\mathbf{R}_g = \mathbf{G}^H\mathbf{G}$ is Toeplitz, it can be inverted efficiently.