Signal Modeling

The idea of *signal modeling* is to represent the signal via (some) model parameters.

Signal modeling is used for *signal compression*, *prediction*, *reconstruction* and *understanding*.

Two generic model classes will be considered:

- ARMA, AR, and MA models,
- low-rank models.

AR, MA, and ARMA equations

General ARMA equations:

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \quad \text{ARMA.}$$

Particular cases:

$$y(n) = \sum_{k=0}^{M} b_k x(n-k) \quad \text{MA}, \quad \sum_{k=0}^{N} a_k y(n-k) = x(n) \quad \text{AR.}$$

Taking Z-transforms of both sides of the ARMA equation

$$\sum_{k=0}^{N} a_k \mathcal{Z}\{y(n-k)\} = \sum_{k=0}^{M} b_k \mathcal{Z}\{x(n-k)\}$$

and using time-shift property, we obtain

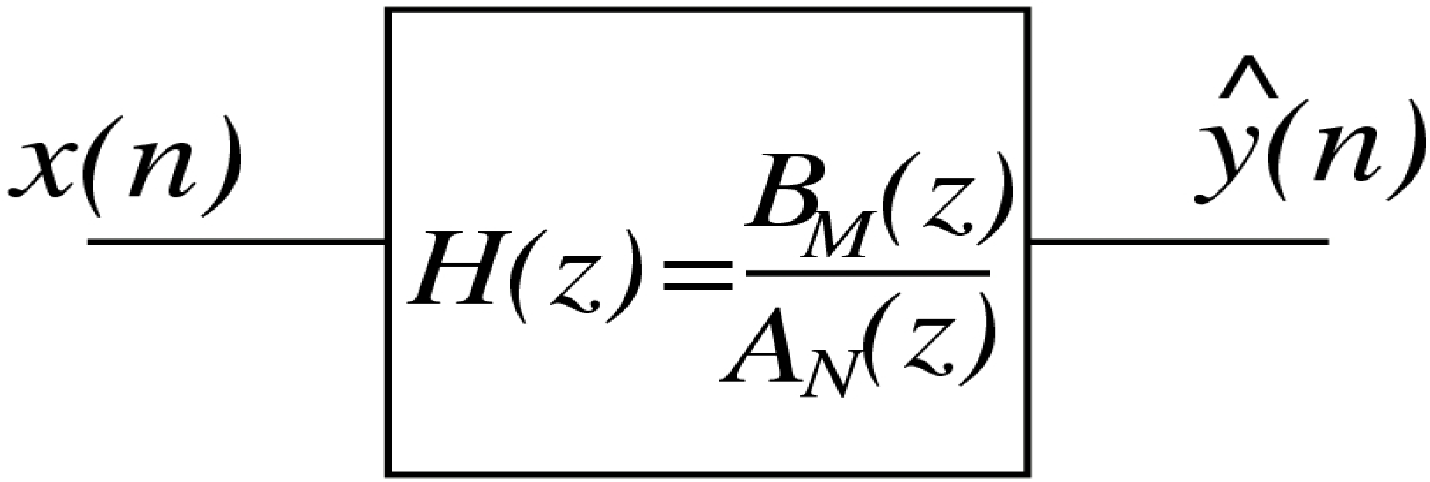
$$Y(z)\sum_{k=0}^{N} a_k z^{-k} = X(z)\sum_{k=0}^{M} b_k z^{-k}.$$

Therefore, the frequency response of causal LTI system (filter):

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} = \frac{B_M(z)}{A_N(z)}.$$

Pole-zero Model

Modeling a signal $y(n)$ as the response of a LTI filter to an input $x(n)$. The goal is to find the filter coefficients and the input $x(n)$ that make the modeled signal $\widehat{y}(n)$ as close as possible to $y(n)$. The most general model is the so-called pole-zero model:



All-pole Modeling: Yule-Walker Equations

All-pole model:

$$H(z) = \frac{\sigma}{A(z)}.$$

Consider the *real* AR equation:

$$y(n) + a_1 y(n-1) + \cdots + a_N y(n-N) = x(n)$$

(with $a_0 = 1$) and assume that $\mathrm{E}\{x(n)x(n-k)\} = \sigma^2\delta(k)$. Since the AR model implies that

$$y(n) = x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2) + \cdots$$

we get

$$\mathrm{E}\{y(n)x(n)\} = \mathrm{E}\{[x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2) + \cdots]x(n)\} = \sigma^2.$$

All-pole Modeling: Yule-Walker Equations

Similarly,

$$\begin{aligned}
\mathrm{E}\{y(n-k)x(n)\} &= \mathrm{E}\{[x(n-k)+\alpha_1 x(n-k-1)+\cdots]\, x(n)\} \\
&= 0 \quad \text{for } k > 0.
\end{aligned}$$

Represent the AR equation in the vector form:

$$[y(n), y(n-1), \ldots, y(n-N)] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = x(n).$$

All-pole Modeling: Yule-Walker Equations

$$\mathrm{E}\{y(n)x(n)\} = \mathrm{E}\left\{ y(n)\,[y(n), y(n-1), \ldots, y(n-N)] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} \right\}$$

$$= [r_0, r_1, \ldots, r_N] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \sigma^2.$$

All-pole Modeling: Yule-Walker Equations

$$\mathrm{E}\{y(n-k)x(n)\} = \mathrm{E}\left\{ y(n-k)\,[y(n),\ldots,y(n-N)] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} \right\}$$

$$= [r_k, r_{k-1}, \ldots, r_{k-N}] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = 0 \quad \text{for } k > 0.$$

All-pole Modeling: Yule-Walker Equations

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & r_0 & \cdots & r_{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ r_N & r_{N-1} & \cdots & r_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

If we *omit* the first equation, we get

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_{N-1} \\ r_1 & r_0 & \cdots & r_{N-2} \\ \vdots & \vdots & \vdots & \vdots \\ r_{N-1} & r_{N-2} & \cdots & r_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix},$$

or, in matrix notation

$$R\boldsymbol{a} = -\boldsymbol{r} \quad \text{Yule-Walker equations.}$$

Levinson Recursion

For this purpose, let us introduce a slightly different notation:

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & r_0 & \cdots & r_{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ r_N & r_{N-1} & \cdots & r_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{N,1} \\ \vdots \\ a_{N,N} \end{bmatrix} = \begin{bmatrix} \sigma_N^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

which, in matrix notation is

$$R_N \boldsymbol{a}_N = \sigma_N^2 \boldsymbol{e}_1,$$

where $\boldsymbol{e}_1 = [1, 0, 0, \ldots, 0]^T$.

For $N = 1$, we have:

$$\begin{aligned} r_0 + r_1 a_{1,1} &= \sigma_1^2, \\ r_1 + r_0 a_{1,1} &= 0, \end{aligned}$$

and thus

$$\begin{aligned} a_{1,1} &= -\frac{r_1}{r_0}, \\ \sigma_1^2 &= r_0 \Big\{ 1 - \Big[\frac{r_1}{r_0} \Big]^2 \Big\}. \end{aligned}$$

Levinson Recursion (cont.)

Goal: Given $\boldsymbol{a}_N$, we want to find the solution to the $(N+1)$st-order equations $R_{N+1}\boldsymbol{a}_{N+1} = \sigma^2_{N+1}\boldsymbol{e}_1$.

Append a zero to $\boldsymbol{a}_N$ and multiply the resulting vector by R_{N+1}:

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_N & r_{N+1} \\ r_1 & r_0 & \cdots & r_{N-1} & r_N \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ r_N & r_{N-1} & \cdots & r_0 & r_1 \\ r_{N+1} & r_N & \cdots & r_1 & r_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{N,1} \\ \vdots \\ a_{N,N} \\ 0 \end{bmatrix} = \begin{bmatrix} \sigma^2_N \\ 0 \\ \vdots \\ 0 \\ \gamma_N \end{bmatrix},$$

where $\gamma_N = r_{N+1} + \sum_{k=1}^{N} a_{N,k} r_{N+1-k}$. Use the symmetric Toeplitz property of R_{N+1} to rewrite

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_N & r_{N+1} \\ r_1 & r_0 & \cdots & r_{N-1} & r_N \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ r_N & r_{N-1} & \cdots & r_0 & r_1 \\ r_{N+1} & r_N & \cdots & r_1 & r_0 \end{bmatrix} \begin{bmatrix} 0 \\ a_{N,N} \\ \vdots \\ a_{N,1} \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma_N \\ 0 \\ \vdots \\ 0 \\ \sigma_N^2 \end{bmatrix}.$$

Now, make a weighted sum of the above two equations.

Levinson Recursion (cont.)

$$R_{N+1}\left\{\begin{bmatrix}1\\a_{N,1}\\\vdots\\a_{N,N}\\0\end{bmatrix}+\Gamma_{N+1}\begin{bmatrix}0\\a_{N,N}\\\vdots\\a_{N,1}\\1\end{bmatrix}\right\}=\begin{bmatrix}\sigma_N^2\\0\\\vdots\\0\\\gamma_N\end{bmatrix}+\Gamma_{N+1}\begin{bmatrix}\gamma_N\\0\\\vdots\\0\\\sigma_N^2\end{bmatrix}.$$

Now, pick

$$\Gamma_{N+1}=-\frac{\gamma_N}{\sigma_N^2},$$

which reduces the above equation to $R_{N+1}\boldsymbol{a}_{N+1}=\sigma_{N+1}^2\boldsymbol{e}_1$, where

$$\boldsymbol{a}_{N+1}=\begin{bmatrix}1\\a_{N,1}\\\vdots\\a_{N,N}\\0\end{bmatrix}+\Gamma_{N+1}\begin{bmatrix}0\\a_{N,N}\\\vdots\\a_{N,1}\\1\end{bmatrix}\quad\text{and}\quad\sigma_{N+1}^2=\sigma_N^2+\Gamma_{N+1}\gamma_N=\sigma_N^2[1-\Gamma_{N+1}^2].$$

All-pole Modeling: Prony's Method

Yule-Walker equations do not show an explicit way of finding the AR model coefficients from the data.

Consider the AR equation:

$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + x(n),$$

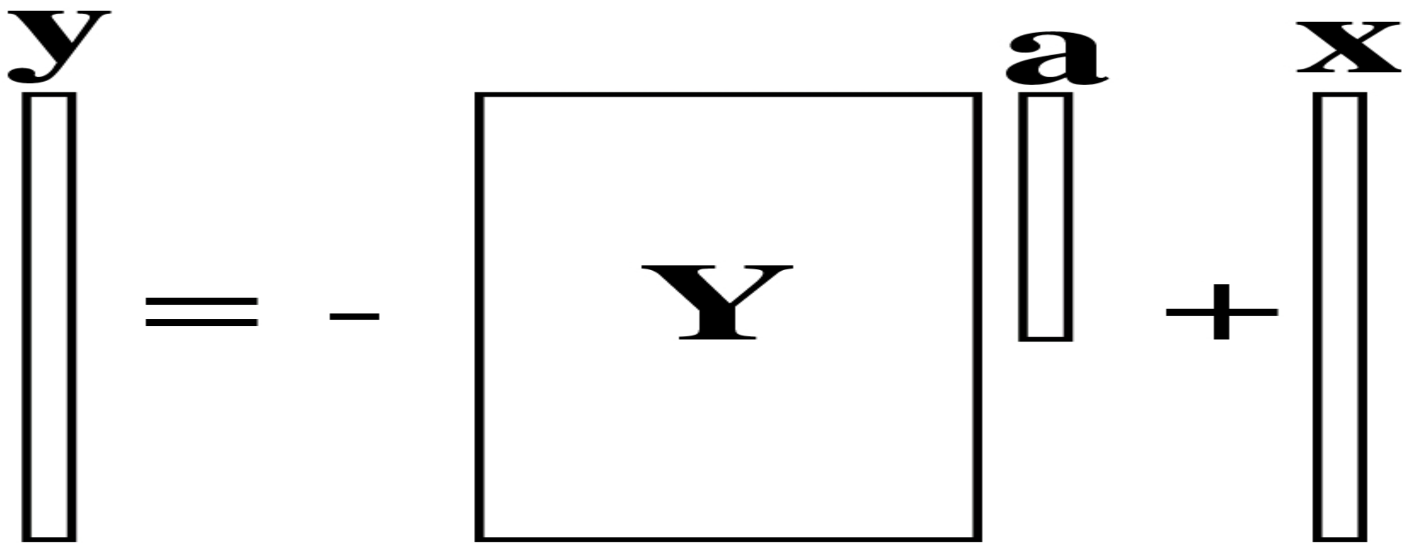
written for $L - N$ measured data points $\{y(n)\}_N^{L-1}$. In matrix form:

$$\begin{bmatrix} y(N) \\ y(N+1) \\ \vdots \\ y(L-1) \end{bmatrix} = -\begin{bmatrix} y(N-1) & y(N-2) & \cdots & y(0) \\ y(N) & y(N-1) & \cdots & y(1) \\ \vdots & \vdots & \vdots & \vdots \\ y(L-2) & & \cdots & y(L-N-1) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} + \begin{bmatrix} x(N) \\ x(N+1) \\ \vdots \\ x(L-1) \end{bmatrix}$$

All-pole Modeling: Prony's Method

In matrix notation, the overdetermined system:

$$\boldsymbol{y} = -Y\boldsymbol{a} + \boldsymbol{x}.$$



All-pole Modeling: Prony's Method

To find a solution, use LS, i.e. minimize

$$\|\boldsymbol{x}\|^2 = (\boldsymbol{y} + Y\boldsymbol{a})^H(\boldsymbol{y} + Y\boldsymbol{a}).$$

The solution is given by the *normal equations*:

$$Y^H Y\boldsymbol{a} = -Y^H\boldsymbol{y}.$$

Solving normal equations, we obtain $\boldsymbol{a}$. Formally, solution can be written as

$$\boldsymbol{a} = -(Y^H Y)^{-1} Y^H\boldsymbol{y}.$$

Relationship between the Yule-Walker and normal (Prony) equations:

$$R\boldsymbol{a} = -\boldsymbol{r}, \qquad Y^H Y\boldsymbol{a} = -Y^H\boldsymbol{y},$$

i.e.

$$R \leftrightarrow Y^H Y \qquad r \leftrightarrow Y^H \boldsymbol{y}.$$

In practice,

$$\widehat{R} = Y^H Y \qquad \widehat{r} = Y^H \boldsymbol{y}$$

represent *sample estimates* of the exact covariance matrix R and covariance vector $\boldsymbol{r}$, respectively!

Linear Prediction $\leftrightarrow$ All-pole Models

Consider the problem of prediction of the future nth value $y(n)$ of the process using the linear predictor based on the previous values $y(n-1), \ldots, y(n-N)$:

$$\widehat{y}(n) = \sum_{i=1}^{N} w_i y(n-i) = \boldsymbol{w}^T \boldsymbol{y},$$

where $\boldsymbol{w}$ is the predictor weight vector and $\boldsymbol{y}$ is the signal vector

$$\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}, \quad \boldsymbol{y} = \begin{bmatrix} y(n-1) \\ y(n-2) \\ \vdots \\ y(n-N) \end{bmatrix}.$$

Linear Prediction $\leftrightarrow$ All-pole Models

Minimize the Mean Square Error (MSE)

$$\begin{aligned}
\epsilon^2 &= \mathrm{E}\left\{[y(n) - \widehat{y}(n)]^2\right\} \\
&= \mathrm{E}\left\{[y - \widehat{y}]^2\right\} \\
&= \mathrm{E}\left\{[y - \boldsymbol{w}^T \boldsymbol{y}]^2\right\} \\
&= \mathrm{E}\left\{y^2 - 2\boldsymbol{w}^T \boldsymbol{y} y + \boldsymbol{w}^T \boldsymbol{y}\boldsymbol{y}^T \boldsymbol{w}\right\} \\
&= \mathrm{E}\left\{y^2\right\} - 2\boldsymbol{w}^T \boldsymbol{r} + \boldsymbol{w}^T R \boldsymbol{w}.
\end{aligned}$$

Taking the gradient, we obtain

$$\frac{\partial \epsilon^2}{\partial \boldsymbol{w}} = -2\boldsymbol{r} + 2R\boldsymbol{w} = 0 \quad \Longrightarrow \quad R\boldsymbol{w} = \boldsymbol{r}$$

and we obtain Yule-Walker equations ($\boldsymbol{w} = -\boldsymbol{a}$)!

- Order-recursive *Levinson-Durbin algorithm* can be used to compute solutions to Yule-Walker (normal) equations (Toeplitz systems).

- The covariance (Prony) method can be modified to minimize the *forward-backward* prediction errors (improved performance).

- AR (all-pole) models are very good for modeling *narrowband* (peaky) signals.

- All-pole modeling is somewhat *simpler* than pole-zero modeling.

All-zero Modeling

All-zero model

$$H(z) = B(z).$$

Consider the *real* MA equation:

$$y(n) = \sum_{i=0}^{M} b_i x(n-i).$$

How to find the MA coefficients?

All-zero Modeling

One idea: find the MA coefficients through the coefficients of an *auxiliary higher-order AR model*. We know that finite MA model can be approximated by an infinite AR model:

$$B_M(z) = \sum_{k=0}^{M} b_k z^{-k} = \frac{1}{A_\infty(z)}.$$

Since AR(∞) is an idealization, let us take an *auxiliary finite* AR(N) model with large $N \gg M$ to find an *approximation* to the above equation:

$$B_M(z) \approx \frac{1}{\sum_{k=0}^{N} a_{k,\mathrm{aux}} z^{-k}}.$$

Clearly, the reverse equation also holds

$$A_{N,\mathrm{aux}}(z) \approx \frac{1}{B_M(z)}.$$

When the auxiliary AR coefficients are obtained, the last step is to find the MA coefficients of the original MA model. This can be done by

$$\min_{\boldsymbol{b}} \left\{ \int_{-\pi}^{\pi} \|A_{N,\mathrm{aux}}(e^{j\omega})B_M(e^{j\omega}) - 1\|^2 d\omega \right\}.$$

Durbin's Method

- **Step 1**: Given the MA(M) signal $y(n)$, find for it an auxiliary high-order AR(N) model with $N \gg M$ using Yule-Walker or normal equations.

- **Step 2**: Using the AR coefficients obtained in the previous step, find the coefficients of the MA(M) model for the signal $y(n)$.

Pole-zero Modeling: Modified Yule-Walker Method

Pole-zero model (for $b_0 = 1$ and $a_0 = 1$):

$$H(z) = \sigma \frac{B(z)}{A(z)}.$$

Let us consider the *real* ARMA equation:

$$y(n) + \sum_{i=1}^{N} a_i y(n-i) = x(n) + \sum_{i=1}^{M} b_i x(n-i)$$

and assume that

$$\mathrm{E}\left\{x(n)x(n-k)\right\} = \sigma^2 \delta(k) :$$

Write the ARMA(N, M) model as MA(∞) equation:

$$y(n) = x(n) + \beta_1 x(n-1) + \beta_2 x(n-2) + \cdots$$

Similar to the all-pole modeling case, we obtain that

$$\mathrm{E}\{y(n)x(n)\} = \sigma^2, \qquad \mathrm{E}\{y(n-k)x(n)\} = 0 \quad \text{for } k > 0.$$

ARMA equation can be rewritten in the vector form:

$$\begin{aligned} & [y(n), y(n-1), \ldots, y(n-N)] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} \\ = \quad & [x(n), x(n-1), \ldots, x(n-M)] \begin{bmatrix} 1 \\ b_1 \\ \vdots \\ b_M \end{bmatrix}. \end{aligned}$$

Pole-zero Modeling: Modified Yule-Walker Method

Multiply both sides of the last equation with $y(n-k)$ and take $\mathrm{E}\{\cdot\}$:

$k = 0$:

$$[r_0, r_1, \ldots, r_N] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = [\sigma^2, \sigma^2\beta_1, \ldots, \sigma^2\beta_M] \begin{bmatrix} 1 \\ b_1 \\ \vdots \\ b_M \end{bmatrix}.$$

Pole-zero Modeling: Modified Yule-Walker Method

$k = 1$:

$$[r_{-1}, r_0, \ldots, r_{N-1}] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = [0, \sigma^2, \sigma^2\beta_1, \ldots, \sigma^2\beta_{M-1}] \begin{bmatrix} 1 \\ b_1 \\ \vdots \\ b_M \end{bmatrix},$$

... so on until $k = M$.

$k \geq M + 1$:

$$[r_{-k}, r_{-k+1}, \ldots, r_{-k+N}] \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = [0, 0, \ldots, 0] \begin{bmatrix} 1 \\ b_1 \\ \vdots \\ b_M \end{bmatrix} = \mathbf{0}.$$

Pole-zero Modeling: Modified Yule-Walker Method

Therefore, we obtain the *modified* Yule-Walker equations:

$$\begin{bmatrix} r_{-(M+1)} & r_{-(M)} & \cdots & r_{-(M+1)+N} \\ r_{-(M+2)} & r_{-(M+1)} & \cdots & r_{-(M+2)+N} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \mathbf{0}.$$

To solve for $a_1, \ldots, a_N$, we need N equations:

$$\begin{bmatrix} r_{M+1} & r_M & \cdots & r_{M-N+1} \\ r_{M+2} & r_{M+1} & \cdots & r_{M-N+2} \\ \vdots & \vdots & \vdots & \vdots \\ r_{M+N} & r_{M+N-1} & \cdots & r_M \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \mathbf{0},$$

where we use $r_{-k} = r_k$. The matrix is $N \times (N+1)$. Equivalent to

$$\begin{bmatrix} r_M & r_{M-1} & \cdots & r_{M-N+1} \\ r_{M+1} & r_{M+1} & \cdots & r_{M-N+2} \\ \vdots & \vdots & \vdots & \vdots \\ r_{M+N-1} & r_{M+N-2} & \cdots & r_M \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} = - \begin{bmatrix} r_{M+1} \\ r_{M+2} \\ \vdots \\ r_{M+N} \end{bmatrix},$$

with the square $N \times N$ matrix. In matrix notation:

$$R\boldsymbol{a} = -\boldsymbol{r} \qquad \leftarrow \quad \text{(modified Yule-Walker equations).}$$

Pole-zero Modeling (cont.)

Once the AR coefficients are determined, it remains to obtain the MA part of the considered ARMA model. Write the ARMA power spectrum as

$$P_y(z) = \sigma^2 \frac{B(z)B(1/z)}{A(z)A(1/z)} \overset{z=\exp(j\omega)}{=} \sigma^2 \left| \frac{B(e^{j\omega})}{A(e^{j\omega})} \right|^2 .$$

Hence, filtering the ARMA process $y(n)$ with the LTI filter having a transfer function $A(z)$ gives the MA part of the process, having the spectrum:

$$P(z) = B(z)B(1/z).$$

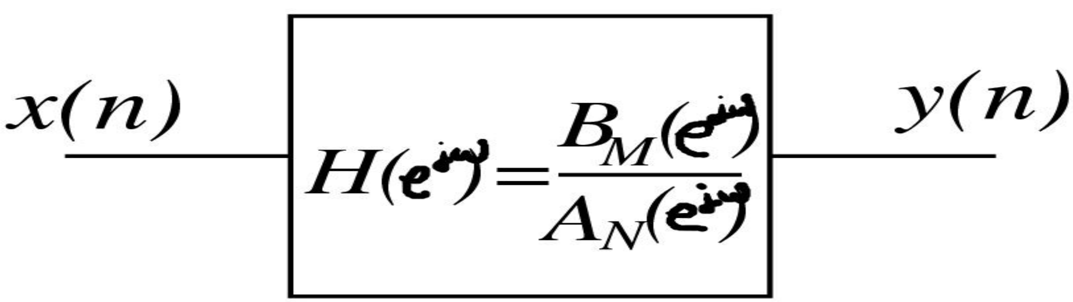
Then, the MA parameters of the ARMA process $y(n)$ can be estimated from this (filtered) MA process using all-zero modeling techniques (for example, Durbin's method).

Digression: Rational Spectra

$$P(e^{j\omega}) = \sigma^2 \left| \frac{B(e^{j\omega})}{A(e^{j\omega})} \right|^2 .$$

Recall: we consider real-valued signals here.

- $a_1, \ldots, a_N, b_1, \ldots, b_M$ are real coefficients.
- Any continuous power spectral density (PSD) can be approximated arbitrarily close by a rational PSD. Consider passing $x(n) \equiv$ zero-mean white noise of variance σ^2 through filter H.



Digression: Rational Spectra (cont.)

The rational spectra can be associated with a signal obtained by filtering white noise of power σ^2 through a rational filter with $H(e^{j\omega}) = B(e^{j\omega})/A(e^{j\omega})$. ARMA model: ARMA(M,N)

$$P(e^{j\omega}) = \sigma^2 \left| \frac{B(e^{j\omega})}{A(e^{j\omega})} \right|^2 .$$

AR model: AR(N)

$$P(e^{j\omega}) = \sigma^2 \left| \frac{1}{A(e^{j\omega})} \right|^2 .$$

MA model: MA(M)

$$P(e^{j\omega}) = \sigma^2 \left| B(e^{j\omega}) \right|^2 .$$

Remarks:

- AR models peaky PSD better,
- MA models valley PSD better,
- ARMA is used for PSD with both peaks and valleys.

Spectral Factorization

$$H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})}.$$

$$P(e^{j\omega}) = \sigma^2 \left|\frac{B(e^{j\omega})}{A(e^{j\omega})}\right|^2 = \frac{\sigma^2 B(e^{j\omega}) B(e^{-j\omega})}{A(e^{j\omega}) A(e^{-j\omega})}$$

$$A(e^{j\omega}) = 1 + a_1 e^{-j\omega} + \ldots + a_M e^{-jM\omega}.$$

$a_1, \ldots, a_N, b_1, \ldots, b_M$ are real coefficients.

Remark: If $a_1, \ldots, a_N, b_1, \ldots, b_M$ are complex,

$$P(z) = \sigma^2 \frac{B(z) B^*(\frac{1}{z^*})}{A(z) A^*(\frac{1}{z^*})}$$

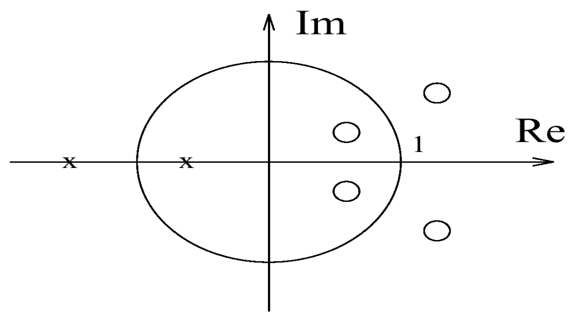
Spectral Factorization

Consider real case:

$$P(z) = \sigma^2 \frac{B(z)B(\frac{1}{z})}{A(z)A(\frac{1}{z})}$$

Remarks:

- If α is zero of $P(z)$, so is $\frac{1}{\alpha}$.
- If β is a pole of $P(z)$, so is $\frac{1}{\beta}$.
- Since $a_1, \ldots, a_N, b_1, \ldots, b_M$ are real, the poles and zeroes of $P(z)$ occur in complex conjugate pairs.



Spectral Factorization

Remarks:

- If poles of $\frac{1}{A(z)}$ inside unit circle, $H(z) = \frac{B(z)}{A(z)}$ is BIBO stable.

- If zeroes of $B(z)$ inside unit circle, $H(z) = \frac{B(z)}{A(z)}$ is minimum phase.

We choose $H(z)$ so that both its zeroes and poles are inside the unit circle.

Low-rank Models

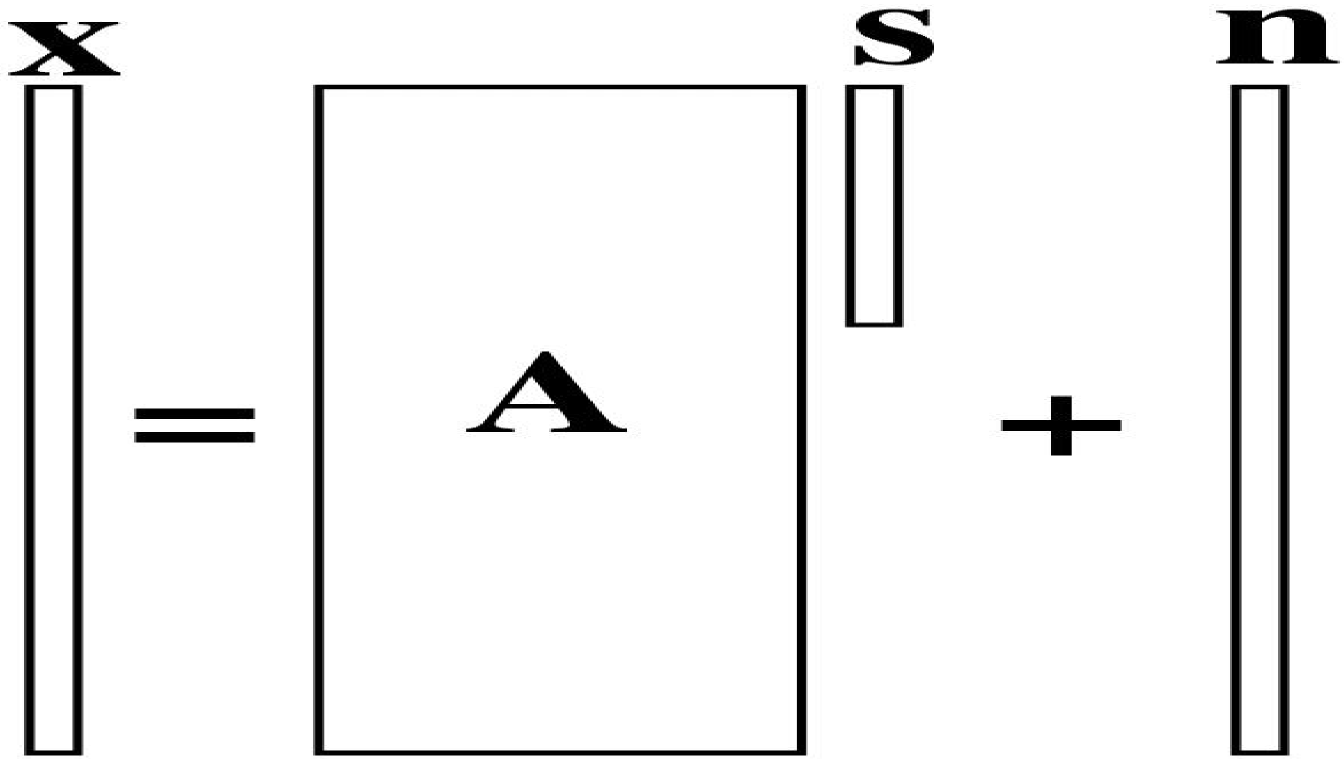
A low-rank model for the data vector $\boldsymbol{x}$:

$$\boldsymbol{x} = A\boldsymbol{s} + \boldsymbol{n}.$$

where A is the model basis matrix, $\boldsymbol{s}$ is the vector of model basis parameters, and $\boldsymbol{n}$ is noise.

$\boldsymbol{s}$ is unknown, A is sometimes completely known (unknown), and sometimes is known up to an unknown parameter vector $\boldsymbol{\theta}$.

Low-rank Models



Low-rank Models (cont.)

Case 1: A completely known. Then, conventional linear LS:

$$\min_{\boldsymbol{s}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2 = \min_{\boldsymbol{s}} \|\boldsymbol{x} - A\boldsymbol{s}\|^2 \quad \Longrightarrow$$

$$\widehat{\boldsymbol{s}} = (A^H A)^{-1} A^H \boldsymbol{x}.$$

This approach can be generalized for *multiple snapshot* case:

$$X = AS + N, \quad X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K],$$

$$S = [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_K], \quad N = [\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_K].$$

$$\min_{S} \|X - \widehat{X}\|^2 = \min_{S} \|X - AS\|^2 \quad \Longrightarrow$$

$$\widehat{S} = (A^H A)^{-1} A^H X.$$

Low-rank Models (cont.)

Case 2: A known up to unknown $\boldsymbol{\theta}$. Nonlinear LS:

$$\min_{\boldsymbol{s},\boldsymbol{\theta}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2 = \min_{\boldsymbol{s},\boldsymbol{\theta}} \|\boldsymbol{x} - A(\boldsymbol{\theta})\boldsymbol{s}\|^2 \quad \Longrightarrow$$

For fixed $\boldsymbol{\theta}$: $\widehat{\boldsymbol{s}} = [A^H(\boldsymbol{\theta})A(\boldsymbol{\theta})]^{-1}A^H(\boldsymbol{\theta})\boldsymbol{x}$. Substituting this back into the LS criterion:

$$\begin{aligned}
& \min_{\boldsymbol{\theta}} \left\| \boldsymbol{x} - A(\boldsymbol{\theta})[A^H(\boldsymbol{\theta})A(\boldsymbol{\theta})]^{-1}A^H(\boldsymbol{\theta})\boldsymbol{x} \right\|^2 \\
= \quad & \min_{\boldsymbol{\theta}} \left\| \left\{ I - A(\boldsymbol{\theta})[A^H(\boldsymbol{\theta})A(\boldsymbol{\theta})]^{-1}A^H(\boldsymbol{\theta}) \right\} \boldsymbol{x} \right\|^2 \\
= \quad & \min_{\boldsymbol{\theta}} \left\| P_A^{\perp}(\boldsymbol{\theta})\boldsymbol{x} \right\|^2 \quad \Longleftrightarrow \quad \max_{\boldsymbol{\theta}} \boldsymbol{x}^H P_A(\boldsymbol{\theta})\boldsymbol{x}.
\end{aligned}$$

Low-rank Models (cont.)

Generalization to the *multiple snapshot* case:

$$\min_{S,\boldsymbol{\theta}} \|X - \widehat{X}\|^2 = \min_{S,\boldsymbol{\theta}} \|X - A(\boldsymbol{\theta})S\|^2 \quad \Longrightarrow$$

$$\widehat{S} = [A^H(\boldsymbol{\theta})A(\boldsymbol{\theta})]^{-1}A^H(\boldsymbol{\theta})X.$$

Substituting this back into the LS criterion:

$$\begin{aligned}
& \min_{\boldsymbol{\theta}} \left\| X - A(\boldsymbol{\theta})[A^H(\boldsymbol{\theta})A(\boldsymbol{\theta})]^{-1}A^H(\boldsymbol{\theta})X \right\|^2 \\
= \quad & \min_{\boldsymbol{\theta}} \left\| P_A^{\perp}(\boldsymbol{\theta})X \right\|^2 = \min_{\boldsymbol{\theta}} \operatorname{tr}\{P_A^{\perp}(\boldsymbol{\theta})XX^H P_A^{\perp}(\boldsymbol{\theta})\} \\
= \quad & \min_{\boldsymbol{\theta}} \operatorname{tr}\{P_A^{\perp}(\boldsymbol{\theta})^2 XX^H\} \\
= \quad & \min_{\boldsymbol{\theta}} \operatorname{tr}\{P_A^{\perp}(\boldsymbol{\theta})XX^H\} \Longleftrightarrow \max_{\boldsymbol{\theta}} \operatorname{tr}\big\{P_A(\boldsymbol{\theta})XX^H\big\}.
\end{aligned}$$

Low-rank Models (cont.)

Note that

$$XX^H = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \begin{bmatrix} \boldsymbol{x}_1^H \\ \boldsymbol{x}_2^H \\ \vdots \\ \boldsymbol{x}_K^H \end{bmatrix} = \sum_{k=1}^{K} \boldsymbol{x}_k \boldsymbol{x}_k^H = K\widehat{R},$$

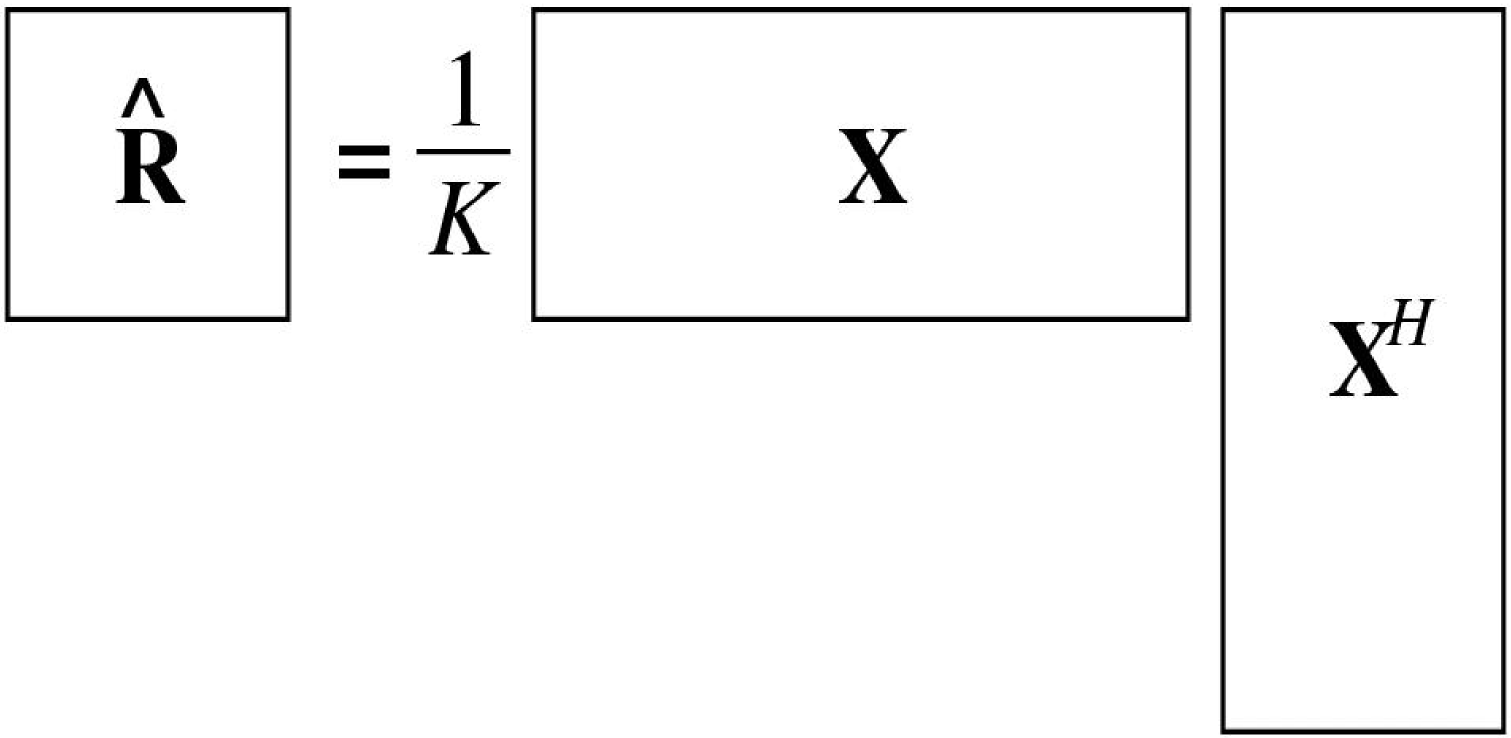
where

$$\widehat{R} = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{x}_k \boldsymbol{x}_k^H \quad \text{sample covariance matrix!}$$

Therefore, the nonlinear LS objective functions can be rewritten as

$$\min_{\boldsymbol{\theta}} \mathrm{tr}\{P_A^{\perp}(\boldsymbol{\theta})\widehat{R}\} \quad \Longleftrightarrow \quad \max_{\boldsymbol{\theta}} \mathrm{tr}\big\{P_A(\boldsymbol{\theta})\widehat{R}\big\}.$$

Low-rank Models (cont.)



Low-rank Models (cont.)

Case 3: A completely unknown.

In this case, a nice result exists, enabling low-rank modeling.

Theorem (Eckart and Young, 1936): Given arbitrary $N \times K$ $(N > K)$ matrix X with the SVD

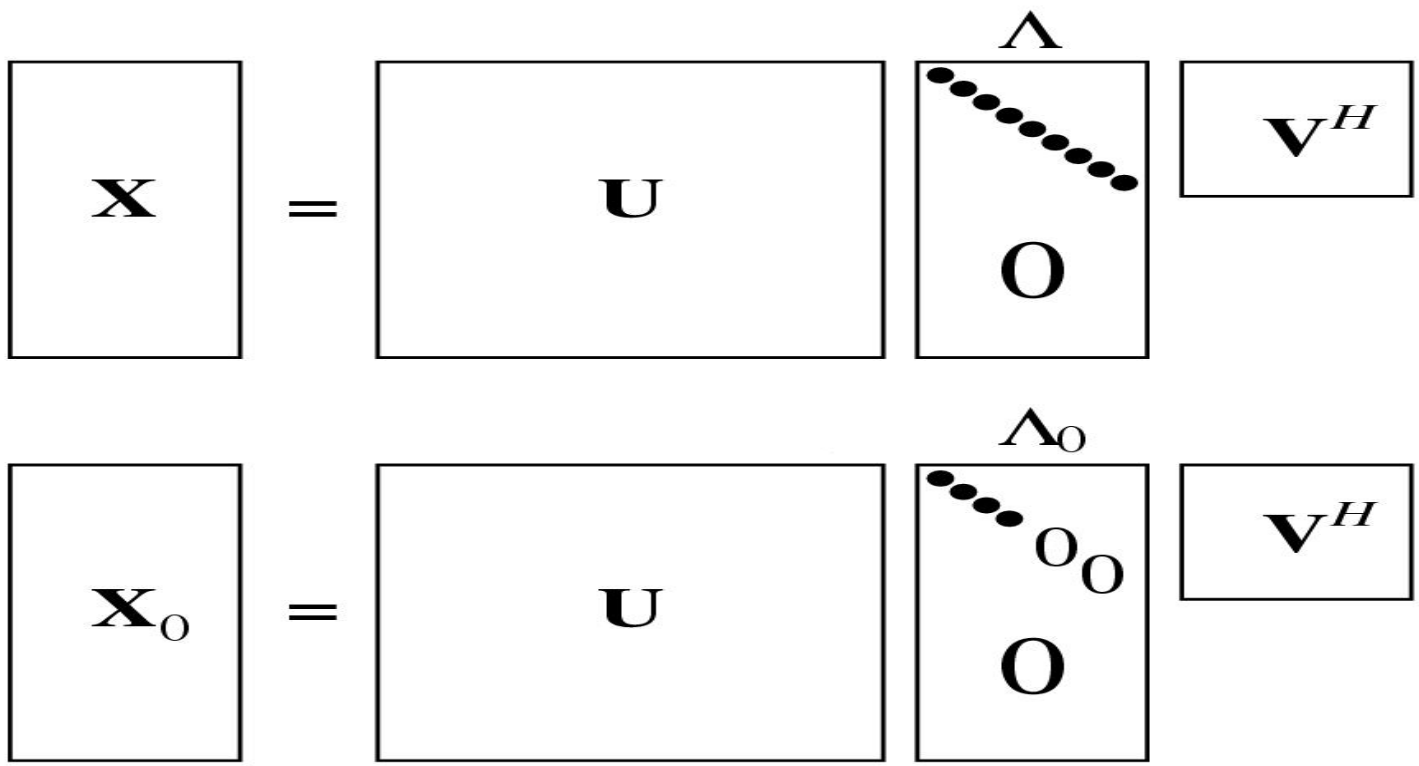
$$X = U \Lambda V^H,$$

the best LS approximation of this matrix by a low-rank matrix X_0 $(L = \mathrm{rank}\{X_0\} \le K)$ is given by

$$\widehat{X}_0 = U \Lambda_0 V^H$$

where the matrix Λ_0 is built from the matrix Λ by replacing the lowest $K - L$ singular values by zeroes.

Low-rank Models (cont.)



Low-rank Modeling of Data Matrix

1. Compute SVD of a given data matrix X,

2. Specify the model order L,

3. From the computed SVD, obtain the low-rank representation using the Eckart and Young's decomposition X_0.