Optimal Signal Processing

Lesson 2

Chapter 4. Signal Modeling

LTH

September 2013

Nedelko Grbic
(Mtrl from Bengt Mandersson)

Electrical and Information Technology, Lund University

Lund University

Chapter 4 Signal modeling

In Chapter 2, we have given a brief review of digital signal processing and some basic matrix definitions.

Then, in chapter 3, the basics of random processes was given, specially autocorrelation sequence, power spectra (power spectral density) and filtering random processes.

Now, we will use our knowledge of random processes to analyze signals which could be described as random processes such as speech signals.

We assume that we have a random process such as speech signals and we want to describe this process in terms of the output from digital filters.

We will have matrix equations and then, in chapter 5, we will describe a well-known algorithm (the Levinson-Durbin algorithm) to solve the equations.

Applications:

Speech coding in Mobile phones

Synthetic speech

Seismology

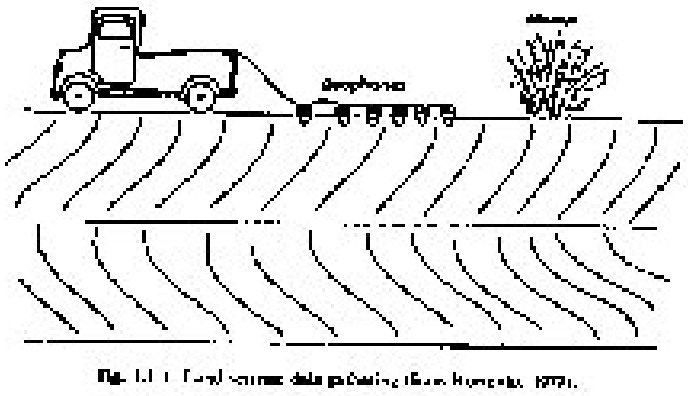
Biomedical applications

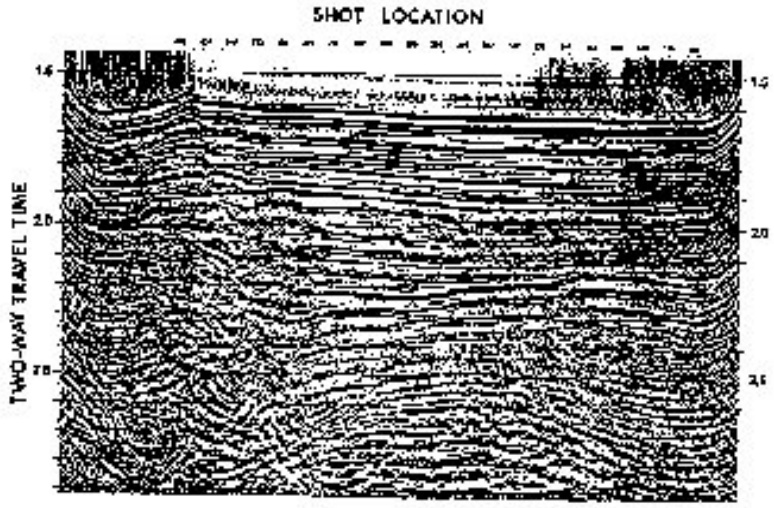
Radar

Sonar

Designing optimum filters for noise reduction

Seismology





Deterministic signals.

Padé	**chapter 4.3 page 134-138**
Prony	**chapter 4.4 page 145-148**
Shanks method	**chapter 4.4.2 page 154-158**
All-pole model	**chapter4.4.3**

Random signals.

All-pole model	**chapter 4.7.2 page 194**

The all-pole model is the most commonly used method and we will concentrate on the use of this method.

Padés approximation (chap 4.3, page 133 - 141)

Start with the difference equation and let the input be *δ(n)* and the output *x(n)*. Then,

$$x(n)+\sum_{k=1}^{p} a(k)\ x(n-k)=\sum_{k=0}^{q} b(k)\delta(n-k)=b(n)$$

This can be written in matrix forms,

$$\begin{bmatrix} x(0) & 0 & \cdots & 0 \\ x(1) & x(0) & \cdots & 0 \\ & . & & \\ x(q) & x(q-1) & \cdots & x(q-p) \\ & \cdots\cdots\cdots\cdots & & \\ x(q+1) & x(q) & \cdots & x(q-p+1) \\ & . & & \\ x(q+p) & x(q+p-1) & \cdots & x(q) \\ & . & & \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a_p(1) \\ a_p(2) \\ . \\ a_p(p) \end{bmatrix} = \begin{bmatrix} b_q(0) \\ b_q(1) \\ . \\ b_q(q) \\ \cdots. \\ 0 \\ 0 \end{bmatrix}$$

We divide the equation in two parts (row *1* to *q* and *q+1* to *q+p*)

$$X\ a_p = b_q \quad \textbf{or} \quad \begin{bmatrix} X_0 \\ X_{q+1} \end{bmatrix} \begin{bmatrix} 1 \\ \bar{a}_p \end{bmatrix} = \begin{bmatrix} b_q \\ 0 \end{bmatrix}$$

Now, we use the lower part to determine *a(n)*. Then, we use these values of *a(n)* to determine *b(n)* from the upper part.

We illustrate the method with an example.

Example of Pade's approximation

Use Padé to determine $H(z)$ **for** $p=2,\ q=2$ **for the signal**

$$x(n)=n\,0.5^n\ \ u(n)=[\underset{\uparrow}{0}\ \ 1/2\ \ 1/2\ \ 3/8\ \ 1/4\ \ 5/32\ \ 3/32\ \ 7/128\ \ ...]$$

We know the system function (use table for z-transform)

$$H(z)=\frac{0.5\ z^{-1}}{1-z^{-1}+0.25\,z^{-2}}$$

We now use method of Pade' and see if we get the same solution.

In matrix form, we have

$$\begin{bmatrix} x(0) & 0 & 0 \\ x(1) & x(0) & 0 \\ x(2) & x(1) & x(0) \\ \cdots & \cdots & \cdots \\ x(3) & x(2) & x(1) \\ x(4) & x(3) & x(2) \\ x(5) & x(4) & x(3) \\ x(6) & x(5) & x(4) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a(1) \\ a(2) \end{bmatrix} = \begin{bmatrix} b(0) \\ b(1) \\ b(2) \\ \cdots \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Pade': Use the rows 4 and 5 to solve *a(1),a(2),* Then rows 1,2,3 to solve *b(0),b(1),b(2.)*

$$\begin{bmatrix} x(3) & x(2) & x(1) \\ x(4) & x(3) & x(2) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a(1) \\ a(2) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Longrightarrow \underbrace{\begin{bmatrix} x(2) & x(1) \\ x(3) & x(2) \end{bmatrix}}_{X_q} \cdot \underbrace{\begin{bmatrix} a(1) \\ a(2) \end{bmatrix}}_{\bar{a}} = -\begin{bmatrix} x(3) \\ x(4) \end{bmatrix}$$

This gives

$$\begin{bmatrix} 1/2 & 1/2 \\ 3/8 & 1/2 \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \end{bmatrix} = -\begin{bmatrix} 3/8 \\ 1/4 \end{bmatrix} \Rightarrow \begin{bmatrix} a(1) \\ a(2) \end{bmatrix} = \begin{bmatrix} -1 \\ 1/4 \end{bmatrix}$$

Then, use row 1,2 and 3 to determine *b(n).*

$$b(0)=x(0)=0$$

$$b(1)=x(1)+a(1)x(0)=0.5$$

$$b(2)=x(2)+a(1)x(1)+a(2)x(0)=0$$

which gives the filter $H(z)=\dfrac{0.5\ z^{-1}}{1-z^{-1}+0.25\,z^{-2}}$

Prony's method (chap 4.4, page 144 – 149)

In Pade's approximation, we use a square matrix to determine *a(n)*. If we use more equations, then we got an overdetermined equation system but we know from the first session how to solve this. This method is called Prony's method. We use the same example to illustrate this.

Example of the Prony method.

We restrict us to use 3 rows because we solve it manually.
Then use the row 4,5,6 and solve it as an overdetermined equation system.
The formula for this from chapter 2.

$$Ax = b \quad (n > m) \Longrightarrow x = (A^H A)^{-1} A^H b$$

Now, we use this formula for row 4,5 and 6.

$$X_q = \begin{bmatrix} x(2) & x(1) \\ x(3) & x(2) \\ x(4) & x(3) \end{bmatrix}, \quad X_q^T X_q = \begin{bmatrix} x(2) & x(3) & x(4) \\ x(1) & x(2) & x(3) \end{bmatrix} \begin{bmatrix} x(2) & x(1) \\ x(3) & x(2) \\ x(4) & x(3) \end{bmatrix} = \begin{bmatrix} 0.45 & 0.53 \\ 0.53 & 0.64 \end{bmatrix}$$

gives $$\begin{bmatrix} a(1) \\ a(2) \end{bmatrix} = -(X_q^T X_q)^{-1} X_q^T \begin{bmatrix} x(3) \\ x(4) \\ x(5) \end{bmatrix} = \ldots = \begin{bmatrix} -1 \\ .25 \end{bmatrix}$$

Then *b(n)* the same as in a), which gives $H(z) = \dfrac{0.5\, z^{-1}}{1 - z^{-1} + 0.25\, z^{-2}}$**.**

Comment: The z-transform of *x(n)* can be found in a formula table to be just $H(z) = \dfrac{z^{-1}}{1 - z^{-1} + 0.25\, z^{-2}}$ **and due to no noise , both methods gives the exact solution.**
The disadvantage of these two methods is that the correlation matrix is not a Toeplitz matrix. Now, we restrict us to use an all-pole model.

All-pole model. (chap 4.4.3, page 162 – 165)

This is the most common model used in practical applications (synthetic speech, speech coding in mobile phones). We assume that the signal *x(n)* can be modeled as output from an *p*-order all-pole filter.

The difference equation for the input $\delta(n)$ is

$$x(n) + \sum_{k=1}^{p} a_p(k)\ x(n-k) = b(0)\delta(n)$$

and the system function

$$H(z) = \frac{b(0)}{1 + a_p(1)z^{-1} + a_p(2)z^{-2} + \cdots + a_p(p)z^{-p}} = \frac{b(0)}{1 + \sum_{k=1}^{p} a_p(k)z^{-k}}$$

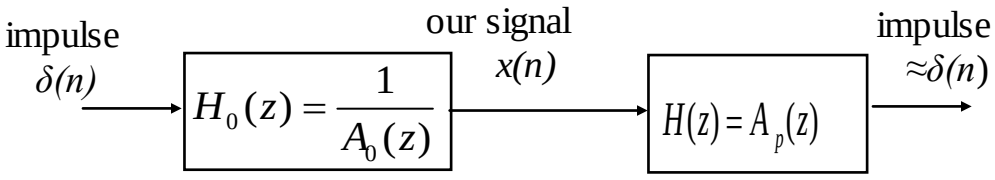
The output should be zero for all $n \neq 0$. We define an error

$$e(n) = x(n) + \sum_{k=1}^{p} a_p(k)\ x(n-k)$$

and we minimize

$$\varepsilon_p = \sum_{n=0}^{\infty} |\, e(n) \,|^2$$

This can be described by the following figure (*b(0)=1*).



$$A_p(z) = 1 + \sum_{k=1}^{p} a_p(k)\, z^{-k}$$

The filter $A_p(z)$ is called the predicting error filter (PEF).

We use a least squares solution to solve the problem.
Take the derivative (for simplicity, we assume real valued signals).

$$\frac{\partial \mathcal{E}_p}{\partial a_p(k)} = \frac{\partial}{\partial a_p(k)} \sum_{n=0}^{\infty} |\, e(n) \,|^2 = 2 \sum_{n=0}^{\infty} e(n) \frac{\partial}{\partial a_p(k)} e(n) =$$

$$= 2 \sum_{n=0}^{\infty} e(n) \frac{\partial}{\partial a_p(k)} [\, x(n) + \sum_{l=1}^{p} a_p(l)\; x(n-l) \,] =$$

$$= 2 \underbrace{\sum_{n=0}^{\infty} e(n) x(n-k)}_{\substack{e(n) \text{ and given data} \\ \text{orthogonal}}} = 0 \qquad k = 1, 2, .., p$$

Then
$$\sum_{n=0}^{\infty} [\, x(n) + \sum_{l=1}^{p} a_p(l)\; x(n-l) \,] x(n-k) = 0$$

With

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

we got the result

$$r\,(k) + \sum_{l=1}^{p} a_p(l)\; \underbrace{r_x(l-k)}_{r_x(k-l)} = 0$$

or rewritten

$$\sum_{l=1}^{p} a_p(l)\; r_x(k-l) = -r_x(k) \qquad k = 1, ..., p$$

This equation is called the normal equation or the Yule-Walker equation.

In matrix form

$$\underbrace{\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & \dots & r_x(p-1) \\ r_x(1) & r_x(0) & r_x(1) & \dots & r_x(p-2) \\ r_x(2) & r_x(1) & r_x(0) & \dots & r_x(p-3) \\ \dots & & & & \\ r_x(p-1) & r_x(p-2) & r_x(p-3) & \dots & r_x(0) \end{bmatrix}}_{R_x} \underbrace{\begin{bmatrix} a_p(1) \\ a_p(2) \\ a_p(3) \\ \\ a_p(p) \end{bmatrix}}_{\bar{a}_p} = -\begin{bmatrix} r_x(1) \\ r_x(2) \\ r_x(3) \\ \\ r_x(p) \end{bmatrix}$$

Orthogonality principle.

We can derive the filter in a slightly different way.

Writing

$$\varepsilon_p = \sum_{n=0}^{\infty} |e(n)|^2 = \sum_{n=0}^{\infty} e(n)e(n) = \sum_{n=0}^{\infty} e(n)[x(n) + \sum_{k=1}^{p} a_p(k)x(n-k)] =$$

$$= \underbrace{\sum_{n=0}^{\infty} e(n)\,x(n)}_{\varepsilon_{p,\min}} + \sum_{k=1}^{p} a_p(k) \underbrace{\sum_{n=0}^{\infty} e(n)\,x(n-k)}_{=0}$$

called model *error*

e(*n*) *and given data must be orthogonal*

The minimum error (model error) is now found as

$$\varepsilon_{p,\min} = \varepsilon_p = \sum_{n=0}^{\infty} e(n)x(n) = \sum_{n=0}^{\infty} [x(n) + \sum_{k=1}^{p} a_p(k)x(n-k)]x(n) =$$

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

$$\varepsilon_{p,\min} = \varepsilon_p = r_x(0) + \sum_{k=1}^{p} a(k)\, r_x\,(k)$$

This equation can be added to the matrix equation described above.

Then, we get (for real signals $r_x^*(k) = r_x(k)$)

$$\underbrace{\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & \dots & r_x(p) \\ r_x(1) & r_x(0) & r_x(1) & \dots & r_x(p-1) \\ r_x(2) & r_x(1) & r_x(0) & \dots & r_x(p-2) \\ r_x(3) & r_x(2) & r_x(1) & \dots & r_x(p-3) \\ \dots & & & & \\ r_x(p) & r_x(p-1) & r_x(p-2) & \dots & r_x(0) \end{bmatrix}}_{R_x} \underbrace{\begin{bmatrix} 1 \\ a_p(1) \\ a_p(2) \\ a_p(3) \\ \\ a_p(p) \end{bmatrix}}_{a_p} = \underbrace{\begin{bmatrix} \varepsilon_p \\ 0 \\ 0 \\ 0 \\ \\ 0 \end{bmatrix}}_{\varepsilon_p u_1}$$

$$R_x \; a_p = \varepsilon_p u_1$$

This is a symmetrical Toeplitz matrix equation system and can be solve with the method described in chapter 5.

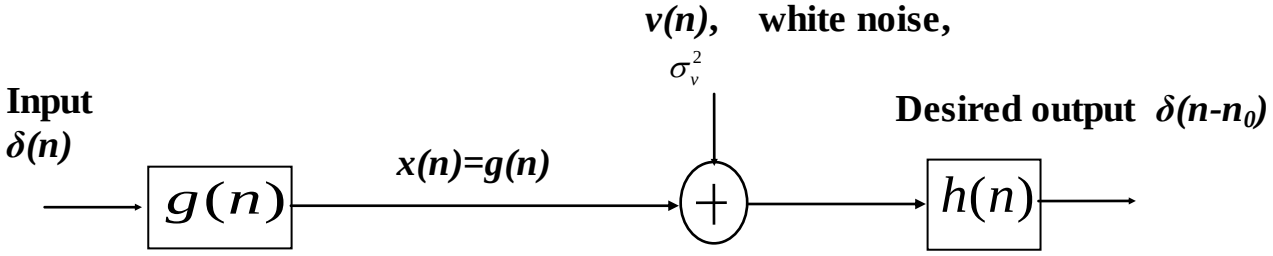
This all-pole model is often called Prediction Error Filter (PEF) or Linear Prediction Coding (LPC).

Shank's method (4.4.2, see Hayes)

Application: FIR Least Squares Inverse Filters: Chap. 4.4.5

Exercise 3, problem 4.19 , Exercise 4 (Computer exercise 1)

The following system is given



The input signal is an impulse,

$$input = \delta(n),$$

and the desired output from our receiver is a delayed version of the input impulse,

$$desired \;\; output = \delta(n - n_0).$$

This means that we want to have the overall impulse response

$$g(n) * h(n) \approx \delta(n - n_0)$$

We define the error signal

$$e(n) = \delta(n - n_0) - g(n) * h(n)$$

Determine the receiver impulse response $h(n)$, which we will minimize

$$\varepsilon_A = \sum_{n=0}^{\infty} e^2(n)$$

Solution: See Exercise 3 and exercise4 (Computer exercise)

Finite Data Records, all-pole modeling (see Hayes ch 4.6)

The error is defined as

$$\varepsilon_p = \sum_{n=0}^{\infty} |e(n)|^2$$

but $x(n)$ **is normally known only for n in the interval [0 N].**

Autocorrelation Method (most common used method)

Determine $r_x(k)$ **assuming** $x(n) = 0$ **outside the interval [0 N].**

Exactly as the Prony's all-pole method with the autocorrelation matrix a Toeplitz matrix.

Covariance Method (used sometimes)

Use only values of $x(n)$ **in the interval [0 N]. Like the Prony's method but the autocorrelation matrix is now not a Toeplitz matrix.**
Then, we only use the *e(n)* that not involves *x(n)* outside the interval.
Then, we use

$$\varepsilon_p^C = \sum_{n=p}^{N} |e(n)|^2$$

Some methods in chapter 6 uses the above error, especially for very short data sequences

Stochastic model 4.7, All-pole model 4.7.2 page 194.

An all-pole stochastic model is called an autoregressive model (AR). The equations are identical to the all-pole model we had before. The only difference is the definition of the autocorrelation sequence.

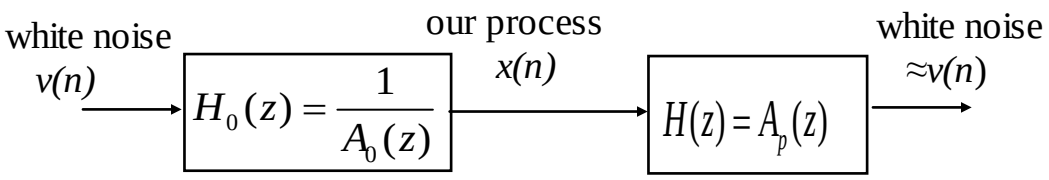
$$\sum_{l=1}^{p} a_p(l)\; r_x(n-l) = -r_x(k) \qquad k = 1,\ldots, p$$

$$r_x(k) = E\{x(n)\, x(n-l)\}$$

The minimum error (model error) is

$$\varepsilon_{p,\min} = \varepsilon_p = r_x(0) + \sum_{k=1}^{p} a(k)\; r_x(k)$$

We now write the model as (predicting error filter, PEF)



$$A_p(z) = 1 + \sum_{k=1}^{p} a_p(k)\, z^{-k}$$

Stochastic model with both poles and zeroes. ARMA-model (4.7 page 189)

The solution with both poles and zeroes is more difficult.

Solve the problem in 2 steps like before (first $a_p(k)$, then $b_q(k)$)

The differential equation is $(white\ \ input\ \ noise\ with\ \ \sigma_v^2 = 1)$

$$x(n) + \sum_{l=1}^{p} a_p(l)x(n-l) = \sum_{l=0}^{q} b_q(l)\, v(n-l)$$

Multiply with $x^*(n-k)$ **and take** $E\{....\}$ **gives** $(a_p(0) = 1)$

$$\sum_{l=0}^{p} a_p(l) r_x(k-l) = \underbrace{\sum_{l=0}^{q} b_q(l) \underbrace{r_{vx}(k-l)}_{r_{vx}(k-l)=h^*(l-k)\,\sigma_v^2}}_{c_q(k)}$$

The right side is $(l \geq k)$

$$c_q(k) = \sum_{l=k}^{q} b_q(l) h^*(l-k)$$

This gives the equations

$$\sum_{l=0}^{p} a_p(l)\, r_x(k-l) = \begin{cases} c_q(k) & 0 \leq k \leq q \\ 0 & k > q \end{cases}$$

or in matrix form

$$\begin{bmatrix} R_a \\ R_b \end{bmatrix} a_p = \begin{bmatrix} c_q \\ 0 \end{bmatrix}$$

*** Determine** a_p **from the lower part, then** c_q **from the upper part.**

*** From** c_q **back to** b_q **we use spectral factorization.**

From chapter 3, we have

$$P_x(z) = \sigma_v^2 \, Q(z) \; Q^*\left(\frac{1}{z^*}\right)$$

Take the transform of the YWE:

$$A_p(z) \, P_x(z) = C_q(z) = B_q(z) \; H^*\left(\frac{1}{z^*}\right)$$

$$A_p(z) \, P_x(z) = C_q(z) = B_q(z) \; \frac{B_q{}^*(1/z^*)}{A_p{}^*(1/z^*)}$$

An finally

$$C_q(z) A_p{}^*(1/z^*) = B_q(z) B_q{}^*(1/z^*)$$

The left side is known and hopefully we can identify the factors in the right side.

Example - ARMA model

Problem: **We will estimate a first order ARMA model from**

$$r_x(0) = 3,\ r_x(1) = 2,\ r_x(2) = 1,$$

Solution: We have $p = q = 1$ **which gives the equations**

$$\begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \end{bmatrix} = \begin{bmatrix} c_1(0) \\ c_1(1) \\ 0 \end{bmatrix}$$

Then, we find $a_1 = -1/2$ **from the lower equation**

and

$$\begin{bmatrix} c_1(0) \\ c_1(1) \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ -1/2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1/2 \end{bmatrix}$$

$$\begin{pmatrix} c_1(k) = 0 & k \geq 2 \\ c_1(k) & k < 0 \quad not \quad used \end{pmatrix}$$

Now, we have to identify $B_q(z)$

We have

$$C_q(z)A_p^*(1/z^*) = \underbrace{(\ldots + 2 + \frac{1}{2}z^{-1})}_{C_q(z)}\underbrace{(1-\frac{1}{2}z)}_{A_p^*(1/z^*)}$$

$$\ldots \ \frac{7}{4} + \frac{1}{2}z^{-1}$$

But

$B_q(z)B_q^*(1/z^*)$ **must be symmetrical so we can write**

$$B_q(z)B_q^*(1/z^*) = \frac{1}{2}z + \frac{7}{4} + \frac{1}{2}z^{-1} =$$

$$= (c_1 + c_2 z^{-1})(c_1 + c_2 z) =$$

$$= \begin{cases} (1.26 + 0.4z^{-1})(1.26 + 0.4z) & \textit{minimum phase} \\ (0.4 + 1.26z^{-1})(0.4 + 1.26z) & \textit{not minimum phase} \end{cases}$$

which gives the filter (choose minimum phase)

$$H(z) = \frac{1.26 + 0.4z^{-1}}{1 - \frac{1}{2}z^{-1}} = 1.26\ \frac{1 + 0.31z^{-1}}{1 - \frac{1}{2}z^{-1}}$$