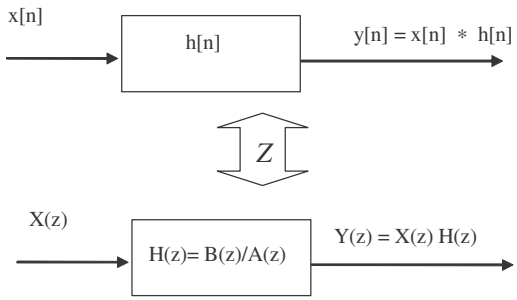
ECGR 6114
Computer Project: All-Pole Prony Models
Student Name:_____

Part 1
Approximating a system response



The Prony method can be used to create all-pole models (or autoregressive models) models for a wide variety of systens where such models are effective. A system is represented by its impulse response h[n] in the time domain, or by its Z-transform H(z) = B(z)/A(z) in the z-domain. As indicated above, the z-transform may be taken across the entire block diagram.

Suppose that we want to approximate a signal g[n] as the output of the above system for some input signal x[n]. For example, we may wish to do this to find some more compact representation of g[n] that may require less memory for storage or less buts for some data transmission.

In most cases, the input signal is presumed to be the unit sample, x[n] = δ[n]. Therefore, the focus is on finding some compact polynomial H(z) = B(z)/A(z) that represents g[n] when x[n] = δ[n].

The trick is then to solve for polnomials A(z) and B(z), given some signal g[n] to be approximated, and given that x[n] = δ[n]. In the Prony method, a least squares approach is used to minimize the error e[n] = Σ (g[n] = h[n])^2

Finally, note that:

1. Prony all-pole autocorrelation method DOES guarantee a stable solution for H(z)
2. Prony all-pole covariance method DOES NOT guarantee a stable solution for H(z)
3. Prony does not necessarily exactly match the points of g[n],
3. Prony uses more than the first na + nb + 1 points of g[n], where na and nb are the order of the polynomials A(z) and B(z) ... a least squares error measure is used.
4. Lower order models are used to approximate a large nunber of points in g[n]
5. Using more points from g[n], i.e., using more data, should result in a better approximation h[n].

As for the Pade method, to solve for H(z), first rearrange the equatons:

$$H(z) = B(z)/A(z)$$

To

$$H(z)A(z) = B(z)$$

But $H(z)=Y(z)/X(z)$ and we want $Y(z) = G(z)$
and so,

$$H(z))A(z) = Y(z))A(z)/X(z) = G(z))A(z)/X(z) = B(z)$$

and since $x[n] = \delta[n]$, and so $X(z) = 1$, then

$$G(z))A(z) = B(z)$$

Finally, taking the inverse z-transform

$$g[n] * a[n] = b[n]$$

And using the definition of convolution

$$\sum_{p=0}^{Na} g[p]\, a[n-p] = b[n]$$

Which can be written in matrix form as follows, with the main difference between Prony and Pade that the number of rows in the g-matrix are greater than the Na + Nb +1 rows used in Pade method. Hence, the system of equations are overdetermined, and will most commonly not have an exact solution. Therefore a least-squares solution will be needed. The order of A(z) is Na, and the order of B(z) is Nb.

All-Pole autocorrelation Prony form of problem shown below.

$$R \cdot A = G^T B$$

$$\begin{pmatrix} g_0 & g_1 & g_2 & g_3 & g_4 & 0 & 0 \\ 0 & g_0 & g_1 & g_2 & g_3 & g_4 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 \end{pmatrix} \cdot \begin{pmatrix} g_0 & 0 & 0 \\ g_1 & g_0 & 0 \\ g_2 & g_1 & g_0 \\ g_3 & g_2 & g_1 \\ g_4 & g_3 & g_2 \\ 0 & g_4 & g_3 \\ 0 & 0 & g_4 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} g_0 & g_1 & g_2 & g_3 & g_4 & 0 & 0 \\ 0 & g_0 & g_1 & g_2 & g_3 & g_4 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 \end{pmatrix} \begin{pmatrix} b_0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Q1. Is the all-pole autocorrelation form of Prony approximation IIR?

Part 2
All-Pole Autocorrelation Method Prony H(z)

Next, we will use the left psuedo-inverse solve for the coefficients a1, a2, b0 as follows. As before, set the order of numerator as Nb, order of denominator as Na where a0, a1, ... aNa, bo

$$Na := 2$$

First define the numerator coefficient as b0, and the vector representing and denominator coefficients as a. Also, load these vectors with initial values (i.e., 0) for the coefficients and define the desired output response g[n]

$$na := 0 .. Na$$

$$a_{na} := 0 \qquad b0 := 0$$

Then, define the desired output response g[n]

$$g := \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$$

For the autocorrelation Prony method the system of equations must be overdetermined, so the Prony method requires rows > Na + Nb +1.

So define Ncmin as the minimum number of rows required in the matrix G.

$$rows\_in\_G\_min := Na + 2 \qquad rows\_in\_G\_min = 4$$

Since the number of rows in vector g will equal the number of rows in matrix G, set vector g equal to zero to force an error in the rest of the program if the number of rows does not satisfy the requirement rows > Na + Nb +1

$$g := if(rows(g) < rows\_in\_G\_min, 0, g) \qquad g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$$

Then set up the autocorrelation form of the matrix G that corresponds to the convolution g[n] * a[n] when G is multioplied by column vector a.

$grows := rows(a) + rows(g) - 1 \qquad gcols := rows(a) \qquad grows = 8$

$gr := 0 .. grows - 1 \qquad gc := 0 .. gcols - 1 \qquad gcols = 3$

$G_{gr,\,gc} := if\left[(gr - gc \geq 0) \wedge (gr - gc \leq rows(g) - 1),\, g_{gr-gc}, 0\right]$

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 0.74 & 1 & 0 \\ 0.45 & 0.74 & 1 \\ 0.2 & 0.45 & 0.74 \\ 0.1 & 0.2 & 0.45 \\ 0.05 & 0.1 & 0.2 \\ 0 & 0.05 & 0.1 \\ 0 & 0 & 0.05 \end{pmatrix}$$

Then compute the autocorrelation matrix R

$R := \left(\overline{G}\right)^{T} \cdot G$

$$R = \begin{pmatrix} 1.803 & 1.188 & 0.653 \\ 1.188 & 1.803 & 1.188 \\ 0.653 & 1.188 & 1.803 \end{pmatrix}$$

The Prony method using split G-matrix and matrix inverse is:

Step 1. Solve for a1, a2 using lower portion of matrix where there are zeroes in the vector on the right hand side of the equation (below the b2 term).
This is done using the least-squares method, i.e., left psuedo-inverse.

Step 2. Solve for b0, using the top of the matrix and the solutions for a1, a2.

All pole autocorrelation method Prony Step 1 is to first solve for a1, a2 values.
First get the lower right G submatrix, G_br, and lower left column, g_left_col

$a\_order := rows(a) - 1$

$R\_br := submatrix(R, 1, rows(R) - 1, 1, cols(R) - 1)$

$r\_left\_col := submatrix(R, 1, rows(R) - 1, 0, 0)$

$$R\_br = \begin{pmatrix} 1.803 & 1.188 \\ 1.188 & 1.803 \end{pmatrix} \qquad r\_left\_col = \begin{pmatrix} 1.188 \\ 0.653 \end{pmatrix}$$

$$aa := -R\_br^{-1} \cdot r\_left\_col \qquad aa = \begin{pmatrix} -0.743 \\ 0.127 \end{pmatrix}$$

Solution:

$$nna := 0 .. rows(aa) - 1$$

$$a_0 := 1 \qquad a_{nna+1} := aa_{nna} \qquad a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix}$$

$$G\_br := submatrix(G, 1, rows(G) - 1, 1, cols(G) - 1)$$

$$g\_left\_col := submatrix(G, 1, rows(G) - 1, 0, 0)$$

And the error vector ε is

$$\varepsilon := G\_br \cdot aa + g\_left\_col$$

And the error is

$$E := \left(\overline{\varepsilon}\right)^T \cdot \varepsilon \qquad E = 3.09 \times 10^{-3}$$

Q2. Is E the same measure of error as used for Prony method in the previous Prony project? yes/no

Q3. Is G_br in the above example full rank? yes/no

And check that e is orthogonal to the column space of A, and that ε is orthogonal to the approximation Ax_s of -y.

$$\left[\overline{(G\_br)}\right]^T \cdot \varepsilon = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \left[\overline{(G\_br \cdot aa)}\right]^T \cdot \varepsilon = 0$$

Summarizing:
We have now used all-pole autocorrelation method to solve for the coefficients of the denominator, A(z)

$$a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix}$$

with error E being

$$E = 3.09 \times 10^{-3}$$

All pole autocorrelation method Prony Step 2 is to solve for b0 value.

$R\_top := submatrix(R, 0, 0, 0, cols(R) - 1)$

$R\_top = (1.803 \quad 1.188 \quad 0.653)$

$bb := R\_top \cdot a$

$bsquared := bb$

$b0 := \sqrt{bsquared}$

Solution:

$b0 = 1.002$

Summarizing steps 1 and 2, the all-pole autocorrelation method solution is:

$$a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix} \qquad b0 = 1.002 \qquad E = 3.09 \times 10^{-3}$$

Where the desired function t be matched was vector g:

$$g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix} \qquad G = \begin{pmatrix} 1 & 0 & 0 \\ 0.74 & 1 & 0 \\ 0.45 & 0.74 & 1 \\ 0.2 & 0.45 & 0.74 \\ 0.1 & 0.2 & 0.45 \\ 0.05 & 0.1 & 0.2 \\ 0 & 0.05 & 0.1 \\ 0 & 0 & 0.05 \end{pmatrix}$$

And the system response is H(z)=N(z)/D(z)=Num(z)/Den(z) where

$$Num(b, z) := b0 \qquad Den(a, z) := \sum_{nn=0}^{rows(a)-1} \left(a_{nn} \cdot z^{-nn} \right)$$

or

$$H(a, b0, z) := \frac{b0}{\sum_{nn=0}^{rows(a)-1} \left(a_{nn} \cdot z^{-nn} \right)}$$

Note also, that the all-pole autocrrelation method has a nonzero error in the approximation of g[n] by the impulse response of the system, y[n] = h[n].

This is because the Prony method minimizes the mean square error of an overdetermined system of equations, it does not guarantee zero. However, note that the overdetermined system resulted from more data, i.e., more samples of g[n]. Hence, it should result in a better model since the Prony method uses more data in estimating the model.

Part 3
Checking the impulse response of the autocorrelation method solution

Compute the first Np points of y[n]=h[n]:

$Np := rows(g)$ $\quad a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix}$ $\quad b0 = 1.002$

$k := 0 .. Np - 1$ $\quad a1 := a_1$ $\quad a2 := a_2$

$y_k := 0$

$y_k := if(k = 0, b0, 0)$

$y_k := if(k = 1, 0 - a_1 \cdot y_{k-1}, y_k)$

$y_k := if(k = 2, 0 - a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

$y_k := if(k > 2, -a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

So, our model y[n] is compared to the desired response g[n]:

$$y = \begin{pmatrix} 1.001544 \\ 0.744187 \\ 0.42532 \\ 0.221187 \\ 0.110146 \\ 0.053654 \end{pmatrix} \quad g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix} \quad g - y = \begin{pmatrix} -1.544 \times 10^{-3} \\ -4.187 \times 10^{-3} \\ 0.025 \\ -0.021 \\ -0.01 \\ -3.654 \times 10^{-3} \end{pmatrix}$$

With error

$\overline{(g - y)}^T \cdot (g - y) = 1.194 \times 10^{-3}$ $\quad g^T g = 1.803$

Note that the error here, $(g-y)^H(g-y)$ is not the same error measure e^He that was used in the least-squares algorithm. This is because the least-squares algorthm used an "indirect error measure" since it was based on an "indirect method" for finding y[n]=h[n] to model g[n].

Q4. Is the approximation y[n] nearly equal to g[n] for n=0 to n=5 (is it within 1% energy difference)?
yes/no

.

Part 4
Checking the stability response of the Autocorrelation method solution

$$a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix} \qquad b0 = 1.002$$

Find poles and zeroes

$$a\_roots := polyroots(a) \qquad a\_roots = \begin{pmatrix} 2.108 \\ 3.722 \end{pmatrix}$$

$$Hpoles := \frac{1}{a\_roots} \qquad Hpoles = \begin{pmatrix} 0.474 \\ 0.269 \end{pmatrix} \qquad \overrightarrow{|Hpoles|} = \begin{pmatrix} 0.474 \\ 0.269 \end{pmatrix}$$

$$b\_roots := polyroots(b0) \qquad b\_roots = \blacksquare$$

$$Hzeroes := \frac{1}{b\_roots} \qquad Hzeroes = \blacksquare$$

Lets plot poles and Zeroes in z-plane

$$ImagZeroes := Im(Hzeroes) \qquad RealZeroes := Re(Hzeroes)$$

Lets just force a zero way off plot to let us plot below

$$ImagZeroes := 1000 \qquad RealZeroes := 1000$$

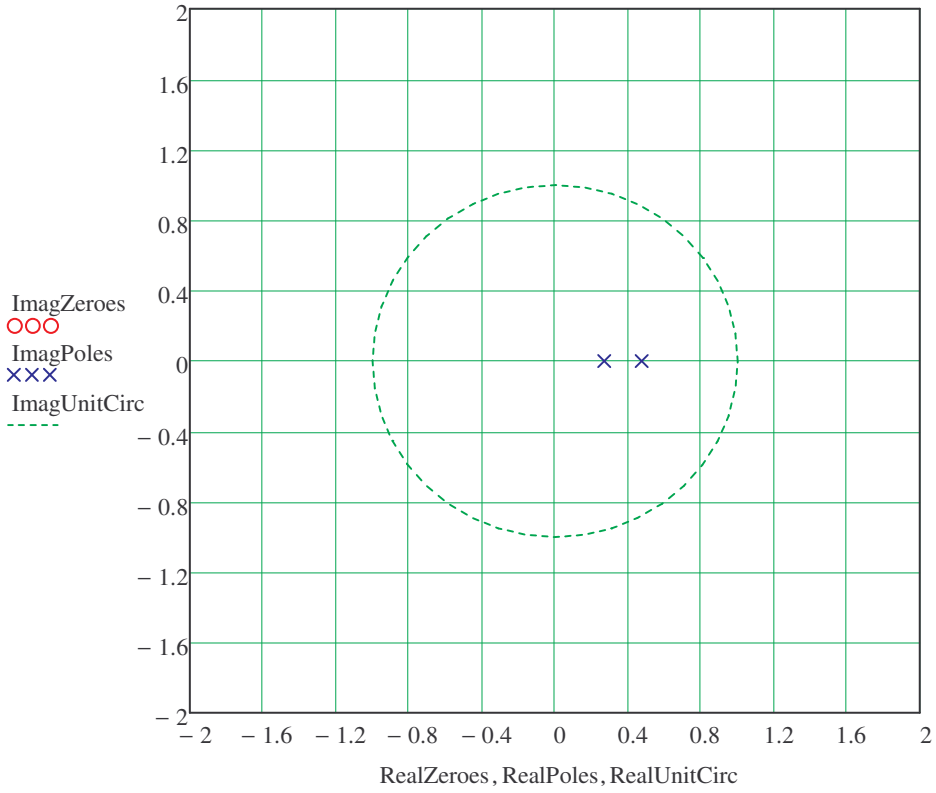
$$ImagPoles := Im(Hpoles) \qquad RealPoles := Re(Hpoles)$$

Lets plot a unit circle too

$un := 0 .. 40$

$ImagUnitCirc_{un} := \sin\left(2 \cdot \pi \cdot \frac{un}{40}\right)$ $RealUnitCirc_{un} := \cos\left(2 \cdot \pi \cdot \frac{un}{40}\right)$

$pmax := round(|Hpoles| + 1)$



Q5. Is the autocorreltion method approximation stable? yes/no

Q6. Is the autocorreltion method approximation invertible ? yes/no

Part 5
Frequency response of the Prony solution

Finally, we have Hbilin(z)=B(z)/N(z)

$$\text{Hprony}(bb, den, z) := \frac{bb}{\sum_{k=0}^{\text{rows}(den)-1} \left(den_k \cdot z^{-k}\right)}$$

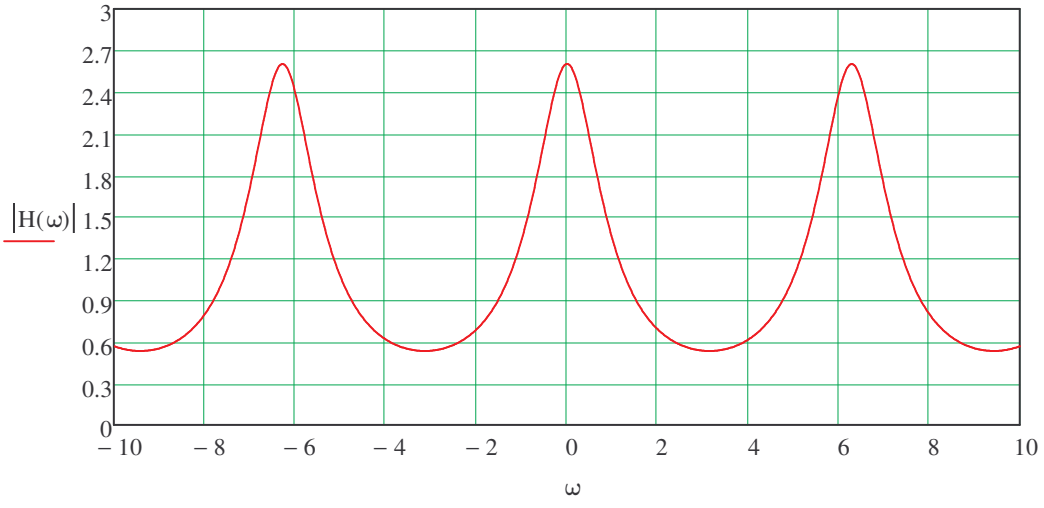
So check a few important points:

$$\text{Hprony}\left(b0, a, e^{i\cdot 0.25}\right) = 2.327 - 0.729i \qquad \text{Hprony}(b0, a, -1) = 0.535$$

Plot the frequency reponse of the bilinear transform filter

$$H(\omega) := \text{Hprony}\left(b0, a, e^{i\cdot\omega}\right) \qquad H(0) = 2.605$$

$$a = \begin{pmatrix} 1 \\ -0.743 \\ 0.127 \end{pmatrix} \qquad b0 = 1.002 \qquad E = 3.09 \times 10^{-3}$$



Q7. Is the autocorrelation method approximation lowpass, highpass, or bandpass ?

Part 6
All-Pole Covariance Method Prony H(z)

Next, we will use the left psuedo-inverse solve for the coefficients a1, a2, b0 as follows. As before, set the order of numerator as Nb, order of denominator as Na where a0, a1, ... aNa, bo

$Na := 2$

First define the numerator coefficient as b0, and the vector representing and denominator coefficients as a. Also, load these vectors with initial values (i.e., 0) for the coefficients and define the desired output response g[n]

$na := 0 .. Na$

$a_{na} := 0 \qquad b0 := 0$

Then, define the desired output response g[n]

$$g := \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$$

For the autocorrelation Prony method the system of equations must be overdetermined, so the Prony method requires rows > Na + Nb +1.

So define Ncmin as the minimum number of rows required in the matrix G.

$rows\_in\_G\_min := Na + 2 \qquad rows\_in\_G\_min = 4$

Since the number of rows in vector g will equal the number of rows in matrix G, set vector g equal to zero to force an error in the rest of the program if the number of rows does not satisfy the requirement rows > Na + Nb +1

$g := if(rows(g) < rows\_in\_G\_min, 0, g)$

$$g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$$

Then set up the covariance form of the matrix G that corresponds to the convolution g[n] * a[n] when G is multiplied by column vector a.

$grows := rows(g)$ $gcols := rows(a)$

$gr := 0 .. grows - 1$ $gc := 0 .. gcols - 1$

$$GG_{gr,gc} := if\left[(gr - gc \geq 0) \wedge (gr - gc \leq rows(g) - 1), g_{gr-gc}, 0\right]$$

$$G := submatrix(GG, Na, rows(GG) - 1, 0, cols(GG) - 1)$$

$$G = \begin{pmatrix} 0.45 & 0.74 & 1 \\ 0.2 & 0.45 & 0.74 \\ 0.1 & 0.2 & 0.45 \\ 0.05 & 0.1 & 0.2 \end{pmatrix}$$

Then compute the autocorrelation matrix R

$$R := \left(\overline{G}\right)^T \cdot G \qquad R = \begin{pmatrix} 0.255 & 0.448 & 0.653 \\ 0.448 & 0.8 & 1.183 \\ 0.653 & 1.183 & 1.79 \end{pmatrix}$$

The Prony method using split G-matrix and matrix inverse is:

Step 1. Solve for a1, a2 using lower portion of matrix where there are zeroes in the vector on the right hand side of the equation (below the b2 term).
This is done using the least-squares method, i.e., left psuedo-inverse.

Step 2. Solve for b0, using the top of the matrix and the solutions for a1, a2.

All pole covariance method Prony Step 1 is to first solve for a1, a2 values.
First get the lower right G submatrix, G_br, and lower left column, g_left_col

$a\_order := rows(a) - 1$

$$R\_br := submatrix(R, 1, rows(R) - 1, 1, cols(R) - 1)$$

$$r\_left\_col := submatrix(R, 1, rows(R) - 1, 0, 0)$$

$$R\_br = \begin{pmatrix} 0.8 & 1.183 \\ 1.183 & 1.79 \end{pmatrix} \qquad r\_left\_col = \begin{pmatrix} 0.448 \\ 0.653 \end{pmatrix}$$

$$aa := -R\_br^{-1} \cdot r\_left\_col \qquad aa = \begin{pmatrix} -0.899 \\ 0.229 \end{pmatrix}$$

Solution:

$nna := 0 .. rows(aa) - 1$

$a_0 := 1 \qquad a_{nna+1} := aa_{nna}$

$$a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix}$$

$G\_br := submatrix(G, 1, rows(G) - 1, 1, cols(G) - 1)$

$g\_left\_col := submatrix(G, 1, rows(G) - 1, 0, 0)$

And the error vector ε is

$$\varepsilon := G\_br \cdot aa + g\_left\_col$$

And the error is

$$E := \left(\overline{\varepsilon}\right)^T \cdot \varepsilon \qquad E = 1.798 \times 10^{-3}$$

And check that e is orthogonal to the column space of A, and that ε is orthogonal to the approximation Ax_s of -y.

$$\overline{(G\_br)}^{T} \cdot \varepsilon = \begin{pmatrix} -0.01 \\ -0.014 \end{pmatrix} \qquad \overline{(G\_br \cdot aa)}^{T} \cdot \varepsilon = 6.126 \times 10^{-3}$$

Summarizing:
We have now used all-pole covariance method to solve for the coefficients of the denominator, A(z)

$$a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix}$$

with error E being

$$E = 1.798 \times 10^{-3}$$

All pole covariance method Prony Step 2 is to solve for b0 value.

$b0 := g_0$

Solution:

$$b0 = 1$$

Summarizing steps 1 and 2, the all-pole autocorrelation method solution is:

$$a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix} \qquad b0 = 1 \qquad E = 1.798 \times 10^{-3}$$

Where the desired function t be matched was vector g:

$$g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix} \qquad G = \begin{pmatrix} 0.45 & 0.74 & 1 \\ 0.2 & 0.45 & 0.74 \\ 0.1 & 0.2 & 0.45 \\ 0.05 & 0.1 & 0.2 \end{pmatrix}$$

And the system response is H(z)=N(z)/D(z)=Num(z)/Den(z) where

$$Num(b, z) := b0 \qquad Den(a, z) := \sum_{nn=0}^{rows(a)-1} \left(a_{nn} \cdot z^{-nn} \right)$$

or

$$H(a, b0, z) := \frac{b0}{\sum_{nn=0}^{rows(a)-1} \left(a_{nn} \cdot z^{-nn} \right)}$$

Note also, that the all-pole autocrrelation method has a nonzero error in the approximation of g[n] by the impulse response of the system, y[n] = h[n].

This is because the Prony method minimizes the mean square error of an overdetermined system of equations, it does not guarantee zero. However, note that the overdetermined system resulted from more data, i.e., more samples of g[n]. Hence, it should result in a better model since the Prony method uses more data in estimating the model.

Part 7
Checking the impulse response of the covariance method solution

Compute the first Np points of y[n]=h[n]:

$Np := rows(g)$ $\quad a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix}$ $\quad b0 = 1$

$k := 0 .. Np - 1$ $\quad a1 := a_1$ $\quad a2 := a_2$

$y_k := 0$

$y_k := if(k = 0, b0, 0)$

$y_k := if(k = 1, 0 - a_1 \cdot y_{k-1}, y_k)$

$y_k := if(k = 2, 0 - a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

$y_k := if(k > 2, -a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

So, our model y[n] is compared to the desired response g[n]:

$y = \begin{pmatrix} 1 \\ 0.89917 \\ 0.579068 \\ 0.314376 \\ 0.149817 \\ 0.062581 \end{pmatrix}$ $\quad g = \begin{pmatrix} 1 \\ 0.74 \\ 0.45 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$ $\quad g - y = \begin{pmatrix} 0 \\ -0.159 \\ -0.129 \\ -0.114 \\ -0.05 \\ -0.013 \end{pmatrix}$

With error

$\overline{(g - y)}^T \cdot (g - y) = 0.058$ $\quad g^T g = 1.803$

Note that the error here, $(g-y)^H(g-y)$ is not the same error measure e^He that was used in the least-squares algorithm. This is because the least-squares algorthm used an "indirect error measure" since it was based on an "indirect method" for finding y[n]=h[n] to model g[n].

Q8. Is the approximation y[n] nearly equal to g[n] for n=0 to n=5 (is it within 5% energy difference)?
yes/no

.

Part 8
Checking the stability response of the covariance method solution

$$a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix} \qquad b0 = 1$$

Find poles and zeroes

$$a\_roots := polyroots(a) \qquad a\_roots = \begin{pmatrix} 1.96 - 0.72i \\ 1.96 + 0.72i \end{pmatrix}$$

$$Hpoles := \frac{1}{a\_roots} \qquad Hpoles = \begin{pmatrix} 0.45 + 0.165i \\ 0.45 - 0.165i \end{pmatrix} \qquad \overrightarrow{|Hpoles|} = \begin{pmatrix} 0.479 \\ 0.479 \end{pmatrix}$$

$$b\_roots := polyroots(b0) \qquad b\_roots = \blacksquare$$

$$Hzeroes := \frac{1}{b\_roots} \qquad Hzeroes = \blacksquare$$

Lets plot poles and Zeroes in z-plane

$$ImagZeroes := Im(Hzeroes) \qquad RealZeroes := Re(Hzeroes)$$

Lets just force a zero way off plot to let us plot below

$$ImagZeroes := 1000 \qquad RealZeroes := 1000$$

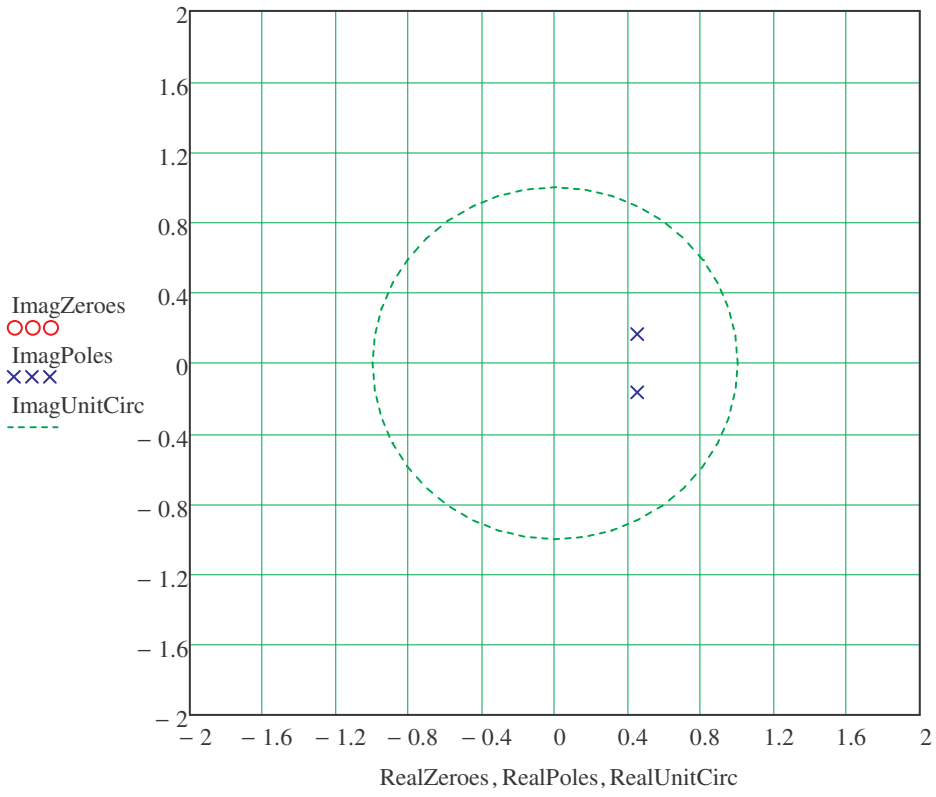
$$ImagPoles := Im(Hpoles) \qquad RealPoles := Re(Hpoles)$$

Lets plot a unit circle too

$un := 0 .. 40$

$ImagUnitCirc_{un} := \sin\left(2 \cdot \pi \cdot \frac{un}{40}\right)$ $RealUnitCirc_{un} := \cos\left(2 \cdot \pi \cdot \frac{un}{40}\right)$

$pmax := round(|Hpoles| + 1)$



Q8. Is the covariance method approximation stable? yes/no

Q9. Is the covariance method minimum phase? yes/no

Part 9
Frequency response of the Prony solution

Finally, we have Hbilin(z)=B(z)/N(z)

$$Hprony(bb, den, z) := \frac{bb}{\sum_{k=0}^{rows(den)-1} \left(den_k \cdot z^{-k}\right)}$$

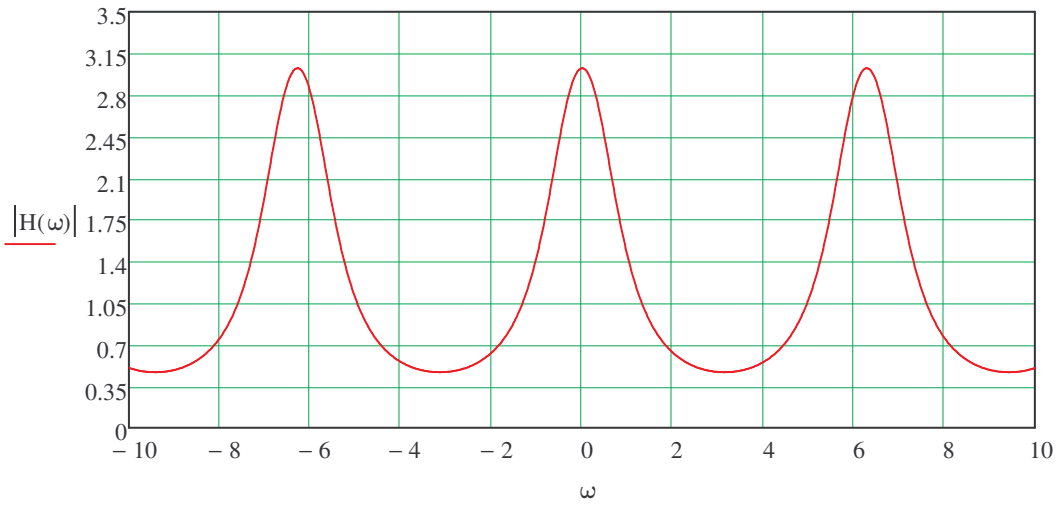
So check a few important points:

$$Hprony\left(b0, a, e^{i \cdot 0.25}\right) = 2.714 - 0.925i \qquad Hprony(b0, a, -1) = 0.47$$

Plot the frequency reponse of the bilinear transform filter

$$H(\omega) := Hprony\left(b0, a, e^{i \cdot \omega}\right) \qquad H(0) = 3.028$$

$$a = \begin{pmatrix} 1 \\ -0.899 \\ 0.229 \end{pmatrix} \qquad b0 = 1 \qquad E = 1.798 \times 10^{-3}$$



Q10. Does the covariance method or the autocorrelation method provide a better match between the approximation y[n] and the desired response g[n]?

autocorrelation/covariance

Part 11
Linear prediction using covariance method

Next, we will use the left psuedo-inverse solve for the coefficients a1, a2, b0 as follows. As before, set the order of numerator as Nb, order of denominator as Na where a0, a1, ... aNa, bo

$Na := 2$

First define the numerator coefficient as b0, and the vector representing and denominator coefficients as a. Also, load these vectors with initial values (i.e., 0) for the coefficients and define the desired output response g[n]

$na := 0 .. Na$

$a_{na} := 0 \qquad b0 := 0$

Then, define the desired output response g[n]

$$g := \begin{pmatrix} 1.01 \\ 2 \\ 2.98 \\ 3.94 \\ 5.03 \\ 6 \end{pmatrix}$$

For the autocorrelation Prony method the system of equations must be overdetermined, so the Prony method requires rows > Na + Nb +1.

So define Ncmin as the minimum number of rows required in the matrix G.

$rows\_in\_G\_min := Na + 2 \qquad rows\_in\_G\_min = 4$

Since the number of rows in vector g will equal the number of rows in matrix G, set vector g equal to zero to force an error in the rest of the program if the number of rows does not satisfy the requirement rows > Na + Nb +1

$g := if(rows(g) < rows\_in\_G\_min, 0, g)$

$$g = \begin{pmatrix} 1.01 \\ 2 \\ 2.98 \\ 3.94 \\ 5.03 \\ 6 \end{pmatrix}$$

Then set up the covariance form of the matrix G that corresponds to the convolution g[n] * a[n] when G is multiplied by column vector a.

$grows := rows(g)$ $gcols := rows(a)$

$gr := 0 .. grows - 1$ $gc := 0 .. gcols - 1$

$$GG_{gr,gc} := if\left[(gr - gc \geq 0) \wedge (gr - gc \leq rows(g) - 1), g_{gr-gc}, 0\right]$$

$$G := submatrix(GG, Na, rows(GG) - 1, 0, cols(GG) - 1)$$

$$G = \begin{pmatrix} 2.98 & 2 & 1.01 \\ 3.94 & 2.98 & 2 \\ 5.03 & 3.94 & 2.98 \\ 6 & 5.03 & 3.94 \end{pmatrix}$$

Then compute the autocorrelation matrix R

$$R := \left(\overline{G}\right)^T \cdot G \qquad R = \begin{pmatrix} 85.705 & 67.699 & 49.519 \\ 67.699 & 53.705 & 39.539 \\ 49.519 & 39.539 & 29.424 \end{pmatrix}$$

The Prony method using split G-matrix and matrix inverse is:

Step 1. Solve for a1, a2 using lower portion of matrix where there are zeroes in the vector on the right hand side of the equation (below the b2 term). This is done using the least-squares method, i.e., left psuedo-inverse.

Step 2. Solve for b0, using the top of the matrix and the solutions for a1, a2.

All pole covariance method Prony Step 1 is to first solve for a1, a2 values.
First get the lower right G submatrix, G_br, and lower left column, g_left_col

$a\_order := rows(a) - 1$

$R\_br := submatrix(R, 1, rows(R) - 1, 1, cols(R) - 1)$

$r\_left\_col := submatrix(R, 1, rows(R) - 1, 0, 0)$

$$R\_br = \begin{pmatrix} 53.705 & 39.539 \\ 39.539 & 29.424 \end{pmatrix} \qquad r\_left\_col = \begin{pmatrix} 67.699 \\ 49.519 \end{pmatrix}$$

$$aa := -R\_br^{-1} \cdot r\_left\_col \qquad aa = \begin{pmatrix} -2.019 \\ 1.031 \end{pmatrix}$$

Solution:

$nna := 0 .. rows(aa) - 1$

$a_0 := 1 \qquad a_{nna+1} := aa_{nna}$

$$a = \begin{pmatrix} 1 \\ -2.019 \\ 1.031 \end{pmatrix}$$

$G\_br := submatrix(G, 1, rows(G) - 1, 1, cols(G) - 1)$

$g\_left\_col := submatrix(G, 1, rows(G) - 1, 0, 0)$

And the error vector ε is

$$\varepsilon := G\_br \cdot aa + g\_left\_col$$

And the error is

$$E := \left(\overline{\varepsilon}\right)^T \cdot \varepsilon \qquad E = 0.031$$

And check that e is orthogonal to the column space of A, and that ε is orthogonal to the approximation Ax_s of -y.

$$\left[\overline{(G\_br)}\right]^T \cdot \varepsilon = \begin{pmatrix} 0.036 \\ 0.018 \end{pmatrix} \qquad \left[\overline{(G\_br \cdot aa)}\right]^T \cdot \varepsilon = -0.053$$

Summarizing:
We have now used all-pole covariance method to solve for the coefficients of the denominator, A(z)

$$a = \begin{pmatrix} 1 \\ -2.019 \\ 1.031 \end{pmatrix}$$

with error E being

$$E = 0.031$$

All pole covariance method Prony Step 2 is to solve for b0 value.

$$b0 := g_0$$

Solution:

$$b0 = 1.01$$

Summarizing steps 1 and 2, the all-pole autocorrelation method solution is:

$$a = \begin{pmatrix} 1 \\ -2.019 \\ 1.031 \end{pmatrix} \qquad b0 = 1.01 \qquad E = 0.031$$

Where the desired function t be matched was vector g:

$$g = \begin{pmatrix} 1.01 \\ 2 \\ 2.98 \\ 3.94 \\ 5.03 \\ 6 \end{pmatrix} \qquad G = \begin{pmatrix} 2.98 & 2 & 1.01 \\ 3.94 & 2.98 & 2 \\ 5.03 & 3.94 & 2.98 \\ 6 & 5.03 & 3.94 \end{pmatrix}$$

And the system response is H(z)=N(z)/D(z)=Num(z)/Den(z) where

$$\mathrm{Num}(b, z) := b0 \qquad \mathrm{Den}(a, z) := \sum_{nn=0}^{\mathrm{rows}(a)-1} \left(a_{nn} \cdot z^{-nn} \right)$$

or

$$H(a, b0, z) := \frac{b0}{\sum_{nn=0}^{\mathrm{rows}(a)-1} \left(a_{nn} \cdot z^{-nn} \right)}$$

Note also, that the all-pole autocrrelation method has a nonzero error in the approximation of g[n] by the impulse response of the system, y[n] = h[n].

This is because the Prony method minimizes the mean square error of an overdetermined system of equations, it does not guarantee zero. However, note that the overdetermined system resulted from more data, i.e., more samples of g[n]. Hence, it should result in a better model since the Prony method uses more data in estimating the model.

Part 7
Checking the impulse response of the covariance method solution

Compute the first Np points of y[n]=h[n]:

$$Np := \mathrm{rows}(g) \qquad a = \begin{pmatrix} 1 \\ -2.019 \\ 1.031 \end{pmatrix} \qquad b0 = 1.01$$

$k := 0 .. Np - 1$ $a1 := a_1$ $a2 := a_2$

$y_k := 0$

$y_k := if(k = 0, b0, 0)$

$y_k := if(k = 1, 0 - a_1 \cdot y_{k-1}, y_k)$

$y_k := if(k = 2, 0 - a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

$y_k := if(k > 2, -a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k)$

So, our model y[n] is compared to the desired response g[n]:

$$y = \begin{pmatrix} 1.01 \\ 2.03954 \\ 3.077629 \\ 4.112845 \\ 5.133447 \\ 6.1275 \end{pmatrix} \quad g = \begin{pmatrix} 1.01 \\ 2 \\ 2.98 \\ 3.94 \\ 5.03 \\ 6 \end{pmatrix} \quad g - y = \begin{pmatrix} 0 \\ -0.04 \\ -0.098 \\ -0.173 \\ -0.103 \\ -0.127 \end{pmatrix}$$

With error

$$\overline{(g - y)}^T \cdot (g - y) = 0.068 \qquad g^T g = 90.725$$

Note that the error here, $(g-y)^H(g-y)$ is not the same error measure e^He that was used in the least-squares algorithm. This is because the least-squares algorthm used an "indirect error measure" since it was based on an "indirect method" for finding y[n]=h[n] to model g[n].

Finally, we may use the coeffificients to predict the future price:

$$-a_1 \cdot y_{rows(y)-1} - a_2 \cdot y_{rows(y)-2} = 7.083$$

Q8. Do you think that the above predicted future value is a good prediction, based on the known past values of g[n]?

yes/no

.