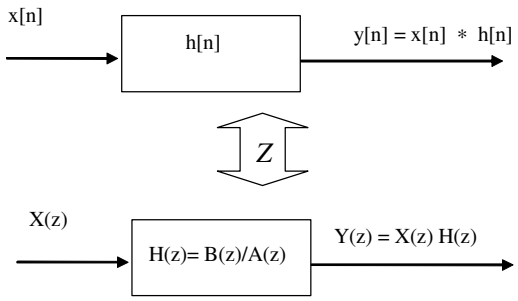
ECGR 6114
Computer Project: Prony method and Least Squares
Student Name:_____

Part 1
Approximating a system response



The Prony method follows the same general idea as the Pade method, except that the derivation of H(z) incorporates more than the first few points in g[n].

As for Pade method, 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, $x[n] = \delta[n]$. Therefore, the focus is on finding some compact polynomial H(z) = B(z)/A(z) that represents g[n] when $x[n] = \delta[n]$.

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

Finally, note that:

1. Prony method does not guarantee a stable solution for H(z)
2. 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.

Prony form of problem shown below.
(The number of rows > Na + Nb +1)

$$\begin{pmatrix} g_0 & 0 & 0 \\ g_1 & g_0 & 0 \\ g_2 & g_1 & g_0 \\ g_3 & g_2 & g_1 \\ g_4 & g_3 & g_2 \\ g_5 & g_4 & g_3 \\ g_6 & g_5 & g_4 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ 0 \\ 0 \end{pmatrix}$$

Pade form of problem shown below.
(The number of rows = Na + Nb +1)

$$\begin{pmatrix} g_0 & 0 & 0 \\ g_1 & g_0 & 0 \\ g_2 & g_1 & g_0 \\ g_3 & g_2 & g_1 \\ g_4 & g_3 & g_2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ 0 \\ 0 \end{pmatrix}$$

Q1. Is the Prony approximation IIR?

Part 2
Least squares and left psuedo-inverse

A commonly encountered problem is the solution of an overdetermined system of equations, where the number of equations exxceeds the number of variables. Since an exact soultion is not usually possible, a common approach is to minimize the error.

In matrix form, let the matrix A have m rows and n columns, where m > n to be overdetermined, and let the rank of A be m (meaning that the columns of A are independent). Let vector x represent the variables to be solved for, and y represent measured output data, or equivalently, let -y represent the vector which is to be approximated by Ax. Then, the least squares problem is writen as

$$Ax \overset{?}{=} -y$$

(An alternative form of the problem would be Ax = y, and olnly differs by a change of sign in y.) Since A is not a square matrix, it is not generallly invertible. So define an error vector ε

$$\varepsilon = Ax + y$$

where ε is a measure of how much Ax + y differs from the ideal value of 0. The least squares error measure is the L2 norm of ε:

$$E = \varepsilon^H \varepsilon$$

where e^H is the hermitian transpose, i.e., the complex conjugate transpose operator. It can be shown that E is minimiized when the solution x_s is given as

$$x = x_s = -\{ A^H A \}^{-1} A^H y$$

where $\{ A^H A \}^{-1} A^H$ is called the left psuedo-inverse of A.

Two important properties of the least squares solution are:

1. The error e is orthogonal to the column space of A, and hence lies completely outside the column space of A. Mathematically, $A^H \varepsilon = 0$.
2. The vector Ax_s (i.e., the approximation of -y) lies entirely within the column space of A, and hence, Ax_s is orthogonal to e. $(Ax_s)^H \varepsilon = x_s{}^H A^H \varepsilon = 0$.

As a simple least squares example, consider the following overdetermined problem:

$$A := \begin{pmatrix} 2 & 1 \\ 1 & 2 \\ 2 & 2 \end{pmatrix} \qquad y := \begin{pmatrix} -11 \\ -11 \\ -11 \end{pmatrix}$$

Then, create an appropriately sized x-vector:

$Nx := \text{cols}(A) - 1 \qquad nx := 0 .. Nx \qquad x_{nx} := 0 \qquad x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Then, make sure the rank of A equals the number of columns: $\text{rank}(A) = 2$

Finally, get the solution:

$$xs := -\left[\left(\left(\overline{A}\right)^{T} \cdot A\right)^{-1} \cdot \left(\overline{A}\right)^{T} \cdot y\right] \qquad xs = \begin{pmatrix} 3.235 \\ 3.235 \end{pmatrix}$$

Note: in Mathcad, the bar over the A indicates complex conjugate and is entered using the double quote key when the cursor is just behind the A.

Also, note that psuedoinverse acts as an inverse:

$$\left[\left(\left(\overline{A}\right)^{T} \cdot A\right)^{-1} \cdot \left(\overline{A}\right)^{T}\right] \cdot A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

And the error vector e is

$$\varepsilon := A \cdot xs + y \qquad \varepsilon = \begin{pmatrix} -1.294 \\ -1.294 \\ 1.941 \end{pmatrix}$$

And the error is

$$E := \left(\overline{\varepsilon}\right)^{T} \cdot \varepsilon \qquad E = (7.118)$$

and compare the approximation Ax_s with -y

$$A \cdot xs = \begin{pmatrix} 9.706 \\ 9.706 \\ 12.941 \end{pmatrix} \qquad -y = \begin{pmatrix} 11 \\ 11 \\ 11 \end{pmatrix}$$

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

$$\left(\overline{A}\right)^{T} \cdot \varepsilon = \begin{pmatrix} 1.599 \times 10^{-14} \\ 1.599 \times 10^{-14} \end{pmatrix} \qquad \left(\overline{(A \cdot xs)}\right)^{T} \cdot \varepsilon = \left(1.034 \times 10^{-13}\right)$$

Q2. Change the above least-squares problem so that the last row of A is [2, 1], and explain why the error E becomes zero.

Part 3
Using left psuedoinverse to solve for Prony H(z)

Next, we will use the left psuedo-inverse solve for the coefficients a1, a2, b0, b1, b2 as follows. As before, set the order of numerator as Nb, order of denominator as Na where a0, a1, ... aNa, bo, b1, b2, ... bNb

$Na := 2$ $Nb := 2$

First define the vectors representing the numberator coefficients as b, 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]

$na := 0..Na$ $nb := 0..Nb$

$a_{na} := 0$ $b_{nb} := 0$

Then, define the desired output response g[n]

$$g := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \end{pmatrix}$$

For the Pade method, the number of rows in the matrix was rows = Na + Nb +1, for the 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.

$rows\_in\_G\_min := Na + Nb + 2$ $rows\_in\_G\_min = 6$

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

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

$$g = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \end{pmatrix}$$

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

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

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

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

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \\ 4 & 3 & 2 \\ 3 & 4 & 3 \\ 2 & 3 & 4 \end{pmatrix}$$

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, b1, b2, using the top of the matrix and the solutions for a1, a2.

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

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

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

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

$$G\_br = \begin{pmatrix} 3 & 2 \\ 4 & 3 \\ 3 & 4 \end{pmatrix} \qquad g\_left\_col = \begin{pmatrix} 4 \\ 3 \\ 2 \end{pmatrix}$$

then for this example we have to solve the following overdetermined system for vector a

$$G\_br \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = -g\_left\_col \qquad \text{or} \qquad \begin{pmatrix} 3 & 2 \\ 4 & 3 \\ 5 & 4 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = -\begin{pmatrix} 4 \\ 5 \\ 0 \end{pmatrix}$$

The least-squares soultion (if G_br has full rank) using the left psuedo-inverse is

$$\text{rank}(G\_br) = 2$$

$$as := -\left[\left(\left(\overline{G\_br}\right)^{T} \cdot G\_br\right)^{-1} \cdot \left(\overline{G\_br}\right)^{T} \cdot g\_left\_col\right] \qquad as = \begin{pmatrix} -1.395 \\ 0.581 \end{pmatrix}$$

Note: in Mathcad, the bar over the A indicates complex conjugate and is entered using the double quote key when the cursor is just behind the A.

Checking the psuedoinverse:

$$\left(\left(\overline{G\_br}\right)^{T} \cdot G\_br\right)^{-1} \cdot \left(\overline{G\_br}\right)^{T} \cdot G\_br = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Next, load the solution into vector a containing the coefficients of H(z):

$$a_0 := 1$$

$$cnt := 1 .. \text{rows}(a) - 1$$

$$a_{cnt} := as_{cnt-1}$$

And the error vector ε is

$$\varepsilon := G\_br \cdot as + g\_left\_col \qquad \varepsilon = \begin{pmatrix} 0.977 \\ -0.837 \\ 0.14 \end{pmatrix}$$

And the error is

$$E := \left(\overline{\varepsilon}\right)^{T} \cdot \varepsilon \qquad E = (1.674) \qquad \left(\overline{g\_left\_col}\right)^{T} \cdot \varepsilon = (1.674)$$

and compare the approximation Ax_s with -y

$$G\_br \cdot as = \begin{pmatrix} -3.023 \\ -3.837 \\ -1.86 \end{pmatrix} \quad g\_left\_col = \begin{pmatrix} -4 \\ -3 \\ -2 \end{pmatrix}$$

Q3. Is G_br in the above example full rank?

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

$$\left(\overline{(G\_br)}\right)^T \cdot \varepsilon = \begin{pmatrix} -1.132 \times 10^{-14} \\ -9.77 \times 10^{-15} \end{pmatrix} \qquad \left(\overline{(G\_br \cdot as)}\right)^T \cdot \varepsilon = \left(1.034 \times 10^{-14}\right)$$

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

$$a = \begin{pmatrix} 1 \\ -1.395 \\ 0.581 \end{pmatrix}$$

with error E being

$$E = (1.674)$$

Prony Step 2 is to solve for b0, b1, b2 values.
First get the top G submatrix, G_t, and solve for the upper right side of the equations

$$G\_t := \text{submatrix}(G, 0, b\_order, 0, \text{cols}(G) - 1)$$

$$G\_t = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix}$$

Then the solution for the vector b containing the coefficients of B(z) is

$$b := G\_t \cdot a \qquad b = \begin{pmatrix} 1 \\ 0.605 \\ 0.791 \end{pmatrix}$$

Summarizing steps 1 and 2, the Prony solution is:

$$a = \begin{pmatrix} 1 \\ -1.395 \\ 0.581 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 0.605 \\ 0.791 \end{pmatrix} \qquad E = (1.674) \qquad \varepsilon = \begin{pmatrix} 0.977 \\ -0.837 \\ 0.14 \end{pmatrix}$$

Where the desired function t be matched was vector g:

$$g = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \end{pmatrix} \qquad G = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \\ 4 & 3 & 2 \\ 3 & 4 & 3 \\ 2 & 3 & 4 \end{pmatrix}$$

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

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

or

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

Note also, that the Prony 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 4
Checking the impulse response of the Prony solution

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

$Np := rows(g)$

$$a = \begin{pmatrix} 1 \\ -1.395 \\ 0.581 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 0.605 \\ 0.791 \end{pmatrix}$$

$k := 0 .. Np - 1$ $\quad a1 := a_1$ $\quad a2 := a_2$ $\quad b0 := b_0$ $\quad b1 := b_1$ $\quad b2 := b_2$

$y_k := 0$

$$y_k := if\left(k = 0, b_0, 0\right)$$

$$y_k := if\left(k = 1, b_1 - a_1 \cdot y_{k-1}, y_k\right)$$

$$y_k := if\left(k = 2, b_2 - a_1 \cdot y_{k-1} - a_2 \cdot y_{k-2}, y_k\right)$$

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

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

$$y = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 3.023256 \\ 2.47431 \\ 1.694819 \end{pmatrix} \qquad g = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \end{pmatrix} \qquad g - y = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0.977 \\ 0.526 \\ 0.305 \end{pmatrix}$$

With error

$$\left(\overline{(g - y)}\right)^T \cdot (g - y) = (1.324)$$

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 Prony approximation y[n] nearly equal to g[n] for n=0 to n=5 when when g[n] = { 1, 2, 3, 4, 3, 2 }?

Q5. Add a new plot right below this question, showing the first 20 points of the prony approximation (i.e., the impulse response of H(z)) for for g[n] = { 1, 2, 3, 4, 3, 2 }.

Part 5
Checking the stability response of the Prony solution

$$a = \begin{pmatrix} 1 \\ -1.395 \\ 0.581 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 0.605 \\ 0.791 \end{pmatrix}$$

Find poles and zeroes

$$a\_roots := polyroots(a) \qquad a\_roots = \begin{pmatrix} 1.2 - 0.529i \\ 1.2 + 0.529i \end{pmatrix}$$

$$Hpoles := \frac{1}{a\_roots} \qquad Hpoles = \begin{pmatrix} 0.698 + 0.308i \\ 0.698 - 0.308i \end{pmatrix} \qquad \overrightarrow{|Hpoles|} = \begin{pmatrix} 0.762 \\ 0.762 \end{pmatrix}$$

$$b\_roots := polyroots(b) \qquad b\_roots = \begin{pmatrix} -0.382 + 1.058i \\ -0.382 - 1.058i \end{pmatrix}$$

$$Hzeroes := \frac{1}{b\_roots} \qquad Hzeroes = \begin{pmatrix} -0.302 - 0.836i \\ -0.302 + 0.836i \end{pmatrix}$$

Lets plot poles and Zeroes in z-plane

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

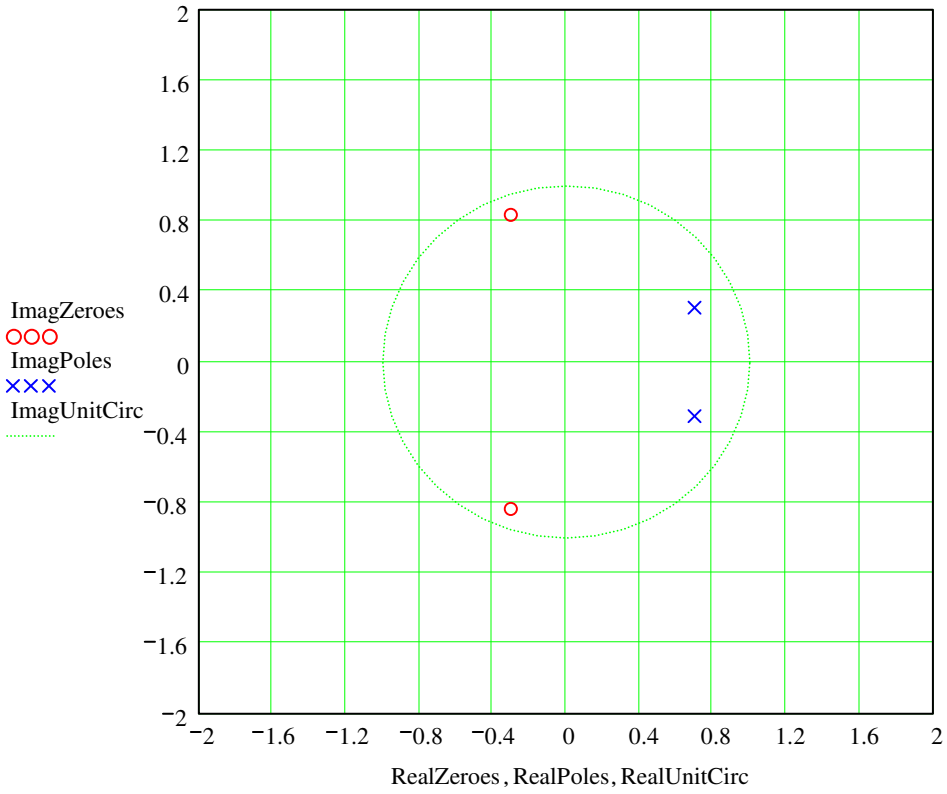
$$\text{ImagPoles} := \text{Im}(\text{Hpoles}) \qquad \text{RealPoles} := \text{Re}(\text{Hpoles})$$

Lets plot a unit circle too

$$un := 0 .. 40$$

$$\text{ImagUnitCirc}_{un} := \sin\left(2\cdot\pi\cdot\frac{un}{40}\right) \qquad \text{RealUnitCirc}_{un} := \cos\left(2\cdot\pi\cdot\frac{un}{40}\right)$$

$$pmax := \text{round}(|\text{Hpoles}| + 1)$$



Q6. Is the Prony approximation stable ***and*** minimum phase for g[n] = { 1, 2, 3, 4, 3, 2 }?

Part 6
Frequency response of the Prony solution

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

$$\text{Hprony}(\text{num},\text{den},z) := \frac{\sum_{k=0}^{\text{rows}(\text{num})-1} \text{num}_k \cdot z^{-k}}{\sum_{k=0}^{\text{rows}(\text{den})-1} \text{den}_k \cdot z^{-k}}$$

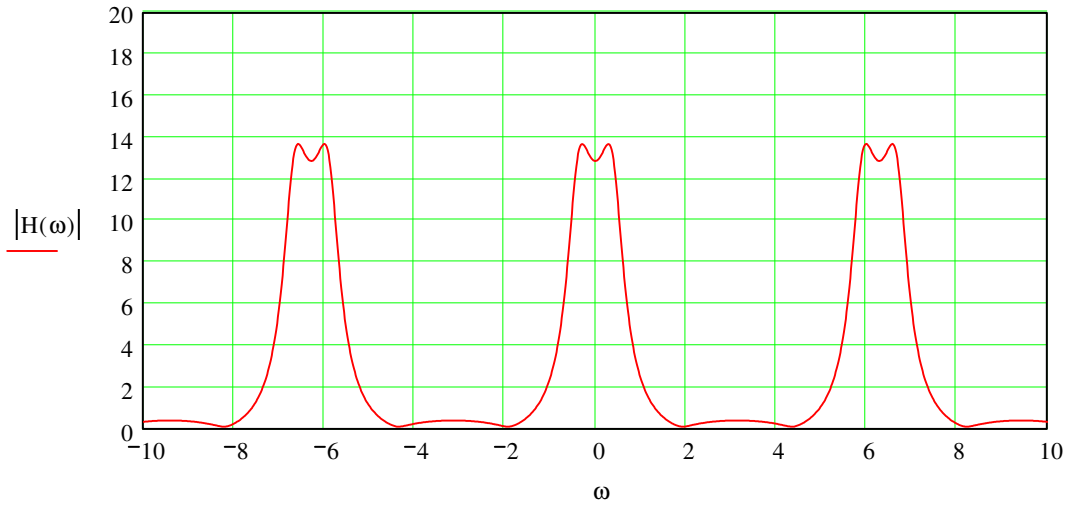
So check a few important points:

$$\text{Hprony}\left(b, a, e^{i\cdot 0.25}\right) = 11.052 - 7.984i \qquad \text{Hprony}(b, a, -1) = 0.398$$

Plot the frequency reponse of the bilinear transform filter

$$H(\omega) := \text{Hprony}\left(b, a, e^{i\cdot\omega}\right) \qquad H(0) = 12.875$$

$$a = \begin{pmatrix} 1 \\ -1.395 \\ 0.581 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 0.605 \\ 0.791 \end{pmatrix} \qquad E = (1.674)$$



Q7. Is the Prony approximation lowpass, highpass, or bandpass for g[n] = { 1, 2, 3, 4, 3, 2 }?

See the last page for additional questions.

Appendix

Using Mathcad Minerr Block to solve for Prony H(z)

THIS MAY GIVE ERRONEOUS RESULTS AS SHOWN BELOW

Mathcad should be able to directly solve for the coefficients a1, a2, b0, b1, b2 as follows using a solve block. Set the order of numerator as Nb, order of denominator as Na where a0, a1, ... aNa, bo, b1, b2, ... bNb

$Na := 2$ $Nb := 2$

First define the vectors representing the numberator coefficients as b, 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]

$na := 0 .. Na$ $nb := 0 .. Nb$

$a_{na} := 0$ $b_{nb} := 0$

Then, define the desired output response g[n]

$$g := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 0 \end{pmatrix}$$

For the Pade method, the number of rows in the matrix was rows = Na + Nb +1, for the 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.

$rows\_in\_G\_min := Na + Nb + 2$ $rows\_in\_G\_min = 6$

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

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

$$g = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 0 \end{pmatrix}$$

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

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

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

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

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \\ 4 & 3 & 2 \\ 5 & 4 & 3 \\ 0 & 5 & 4 \end{pmatrix}$$

Next set up the equations
and use the Mathcad solve block to find the solutionn
(Given is a keyword to start a solve block)

$a1 := 0$ $a2 := 0$ $b0 := 0$ $b1 := 0$ $b2 := 0$

Given

$$G \cdot \begin{pmatrix} 1 \\ a1 \\ a2 \end{pmatrix} = \begin{pmatrix} b0 \\ b1 \\ b2 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \text{where} \qquad G = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \\ 4 & 3 & 2 \\ 5 & 4 & 3 \\ 0 & 5 & 4 \end{pmatrix}$$

Note: if no soultion exists using Find() below, see the Minerr() help page in the manual.

$prony\_solution := Minerr(a1, a2, b0, b1, b2)$

$$prony\_solution = \begin{pmatrix} -2 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

In some editions of Matlab, this prony solution may be wrong, depending on how the Minerr algorithm is working.

And place the solutions back in vectors a and b

$$a_{na} := \text{if}\left(na = 0, 1, \text{prony\_solution}_{na-1}\right) \qquad a = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} \qquad a_2 = 1 \times 10^0$$

$$b_{nb} := \text{prony\_solution}_{nb+\text{rows}(a)-1} \qquad b = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

and check the approximation

$$G \cdot a = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ -6 \end{pmatrix} \qquad \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

and the error vector e

$$ne := 0 .. \text{rows}(G) - \text{rows}(b) - 1$$

$$\varepsilon_{ne} := \left[G \cdot a - \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right]_{ne+\text{rows}(b)} \qquad \varepsilon = \begin{pmatrix} 0 \\ 0 \\ -6 \end{pmatrix}$$

In some editions of Matlab, this error is higher that the psuedo-inverse method solution

and the error E

$$E := \left(\overline{\varepsilon}\right)^T \cdot \varepsilon \qquad E = (36)$$

Q8-13. Do not change the results above. In the following pages, let g[n] = { 1, 2, 3, 4, 3, 2, 1 }, repeat the Prony approximation for this new g[n] below this page, and let **Na = 3** and Nb = 2. Do not change the foregoing system that was solved on the previous pages above this point.

Q8. Solve for the new vectors a and b determining A(z) and B(z)

Q9. Find the new error E and error vector ε

Q10. Find the first 7 points of the impulse response and compare them to g[n] = { 1, 2, 3, 4, 3, 2, 1 } and compute the error from g[n]

Q11. Find and plot the poles and zeroes of the new Prony approximation

Q12. Find and plot $|H(\omega)|$ of the new Prony approximation

Q13. Is this new design stable and minimum phase?

$$g := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}$$