Prony approximation

M.Tognolini HEIG-VD 2022 MA-StatDig

Example 4.3.2

Test Prony approximation with the signal x(n) defined below with the same example as in the Padé.

Test with AR(2) model, AM(2) and ARMA(1,1).

```
N = 41;
n = 0:N-1;
%xn = [ones(1,21),zeros(1,N-21)]

xn = [1, 3* (0.5).^n(2:end)]
```

```
xn = 1×41
    1.0000    1.5000    0.7500    0.3750    0.1875    0.0938    0.0469    0.0234 ···
```

a) Test with AR(2) : p=2 and q = 0

```
p = 2
q = 0
xn = 41×1
    1.0000
    1.5000
    0.7500
    0.3750
    0.1875
    0.0938
    0.0469
    0.0234
    0.0117
    0.0059
      ⋮
```

Compute the Xq matrix using the Matlab X= convmtx(xn,...) function and chose the right index to extract the Xq as in theory eq (4.50)

```
X = 43×3
    1.0000         0         0
    1.5000    1.0000         0
    0.7500    1.5000    1.0000
    0.3750    0.7500    1.5000
    0.1875    0.3750    0.7500
    0.0938    0.1875    0.3750
    0.0469    0.0938    0.1875
    0.0234    0.0469    0.0938
    0.0117    0.0234    0.0469
    0.0059    0.0117    0.0234
      ⋮
```

```
Xq = 42×2
    1.0000         0
    1.5000    1.0000
    0.7500    1.5000
    0.3750    0.7500
    0.1875    0.3750
    0.0938    0.1875
    0.0469    0.0938
    0.0234    0.0469
    0.0117    0.0234
    0.0059    0.0117
      ⋮
```

Compute the vector x_{q+1} as well:

```
xq_1 = 42×1
    1.5000
    0.7500
    0.3750
    0.1875
    0.0938
    0.0469
    0.0234
    0.0117
    0.0059
    0.0029
      ⋮
```

And now compute the ap coefficients with the pseudo inverse of Xq:

```
ap = 2×1
   -1.0714
    0.4286
```

The autocorrelation marrix Rx is :

```
Rx = 2×2
     4     3
     3     4

rx = 2×1
    3.0000
    1.5000

ap = 3×1
    1.0000
   -1.0714
    0.4286
```

The bq coefficients could be found as:

```
X0 = 1×3
     1     0     0

bq = 1

en = 41×1
    0.4286
   -0.4286
    0.2143
    0.1071
    0.0536
    0.0268
```

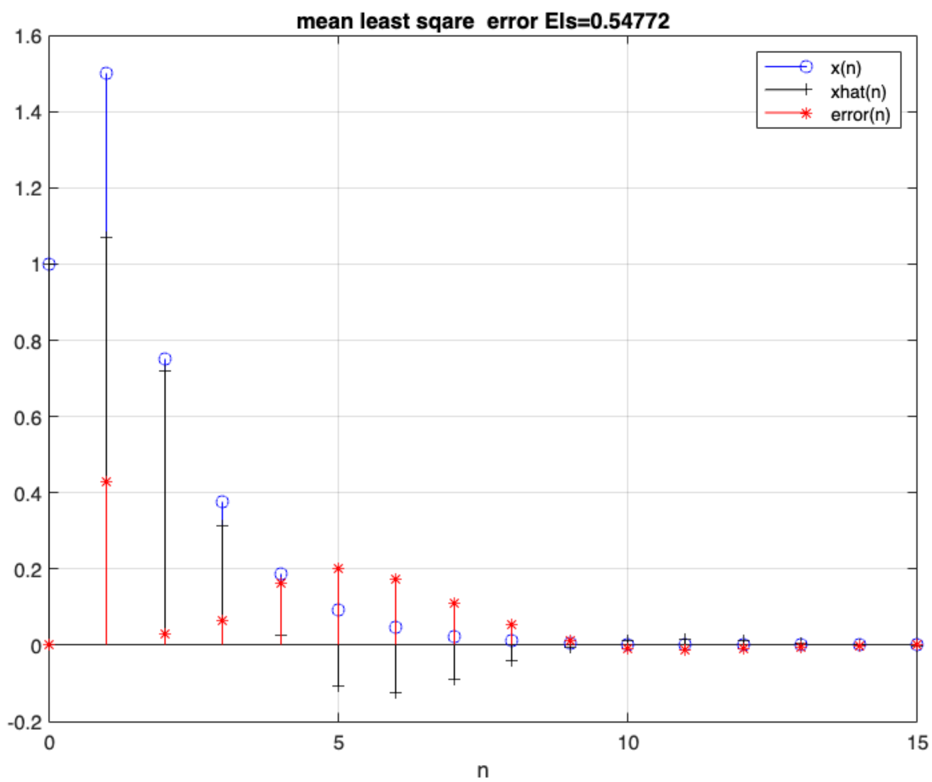
```
      0.0134
      0.0067
      0.0033
      0.0017
        ⋮

  epq = 0.4286
```

Now compute the impulse response that is $\hat{x}(n) = h(n)$

```
  xhat = 41×1
      1.0000
      1.0714
      0.7194
      0.3116
      0.0255
     -0.1062
     -0.1247
     -0.0881
     -0.0410
     -0.0061
        ⋮
```

Compute the error of t he approximation and the least square error:

```
  Els = 0.5477
```



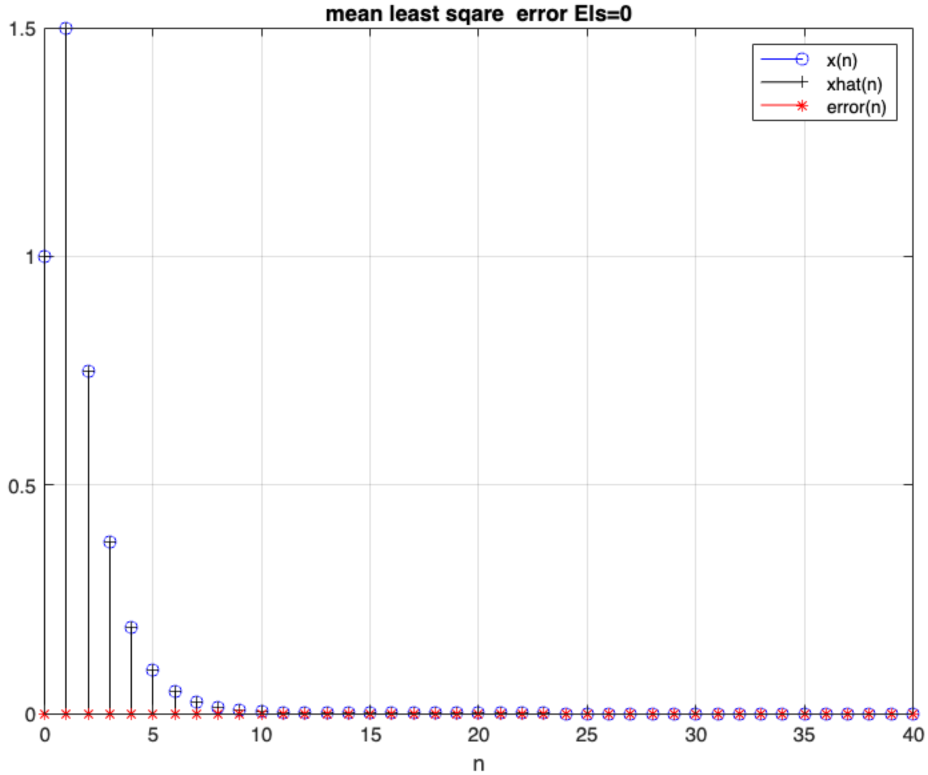
Write a function that compute the Prony approximation

Use the template like this ;

```
prony compute the Prony approximation of the coeff of  ARMA  system with p
poles  and q zeros usage: [ap,bq,epq,xhat] = prony(x,p,q)
epq is optional and is the L2  norm of the approximation error
xhat is optional  and it is the  approximated signal.
(c) M.Tognolini HEIG-VD 2022 r1.0
```

Test the function with the previous exemple:

```
p = 1

q = 1

Calculating the  squared error of  the approximation
and the approximation signal xhat
ap = 2×1
    1.0000
   -0.5000
bq = 2×1
     1
     1
epq = 0
xhat = 41×1
    1.0000
    1.5000
    0.7500
    0.3750
    0.1875
    0.0938
    0.0469
    0.0234
    0.0117
    0.0059
     ⋮

Els = 0
```



Example 4.4.2) Filter design with Prony approximation of the impulse response h(n)

The proble of the filter approximation is to find the impulse response h(n) of the filter from a frequency constraints.

For this exemple to be as simple as possible we want to synthetize a filter with a frequency response $|H(jf)| = 1$ for f < Fp and 0 for Fp < f < 0.5,

aditionally we want a linear phase response then $H(jf) = e^{-jn_d 2\pi f}$ for f < Fp and 0 for Fp < f < 0.5 , the constant nd is the time delay of the filter to assure the causality of the filter.

$$H(jf) = \begin{cases} e^{jn_d 2\pi f} & ; \ |f| < 0.25 \\ 0 & ; \ \text{otherwise} \end{cases}$$

Since the frequency response is a rectangular function the inverse transform is a sinc function of the variable n, the the impulse response is :

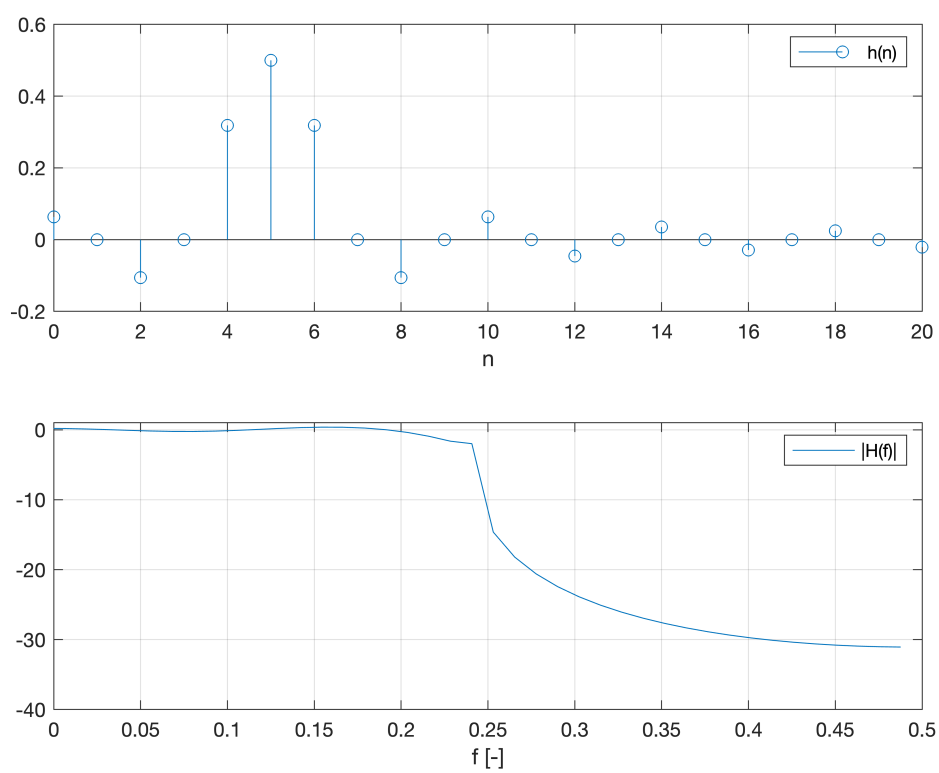
$$h(n) = \frac{sin[(n - n_d)\pi/2]}{(n - n_d)\pi}$$

With the function sinc we write it as: hn = 1/2 * sinc((n-nd)*2*Fp)

```
N = 81
```

```
Fp = 0.25
n= 0:N-1;
nd = 5

hn = 1/2 * sinc((n-nd)*2*Fp)
H = fftshift(fft(hn));
f = -0.5:1/N:0.5-0.5/N;
figure;
subplot(2,1,1)
stem(n,hn)
axis([0,20,-0.2,0.6])
grid;
xlabel('n')
legend('h(n)')
subplot(2,1,2)
plot(f,20*log10(abs(H)))
axis([0,0.5,-40,1])
grid;
xlabel('f [-]')
legend('|H(f)|')
```

```
N = 81

Fp = 0.2500

nd = 5

hn = 1×81
    0.0637   -0.0000   -0.1061    0.0000    0.3183    0.5000    0.3183    0.0000 ···
```

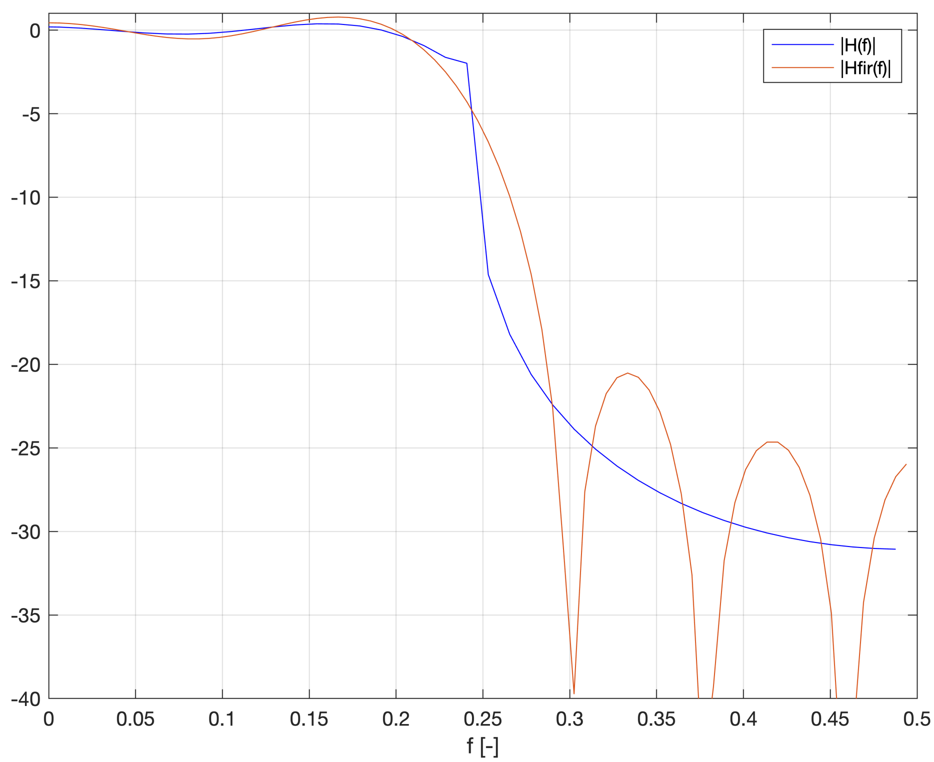


FIR design (MA) with Prony appoximation

If we design a FIR filter with q= 10 and p=0 (MA) the Prony approximation wil give the 11 first values of the h(n). In this case bq(n) = h(n) for n = 0 ..10.

```
p = 0

q = 10

bqFIR = 1×11
    0.0637   -0.0000   -0.1061    0.0000    0.3183    0.5000    0.3183    0.0000 ···

apFIR = 1×11
     1     0     0     0     0     0     0     0     0     0     0

Hfir =


  0.06366 z^10 - 1.949e-17 z^9 - 0.1061 z^8 + 1.949e-17 z^7 + 0.3183 z^6 + 0.5 z^5 + 0.3183 z^4

                                              + 1.949e-17 z^3 - 0.1061 z^2 - 1.949e-17 z + 0.06366

  -------------------------------------------------------------------------------------------------

                                               z^10


Sample time: 1 seconds
Discrete-time transfer function.
```

We could plot the frequency response of this filter with the function freqz():



Pade Approximation

We want to find the transfer fuction of an ARMA system with p=5 and q=5. First test with Padé:

```
p=5
q=5
[ap_pa,bq_pa,Els_pa] = pade(hn(1:p+q+1),p,q)
```

```
p = 5

q = 5

Calculating the  squared error of  the approximation
ap_pa = 6×1
    1.0000
   -2.5256
    3.6774
   -3.4853
    2.1307
   -0.7034
bq_pa = 6×1
    0.0637
   -0.1608
    0.1280
    0.0461
    0.0638
    0.0211
Els_pa = 3.2126e-14

Hpa =

  0.06366 z^5 - 0.1608 z^4 + 0.128 z^3 + 0.04609 z^2 + 0.06377 z + 0.02111
  ------------------------------------------------------------------------
         z^5 - 2.526 z^4 + 3.677 z^3 - 3.485 z^2 + 2.131 z - 0.7034

Sample time: 1 seconds
Discrete-time transfer function.
```

Next test the same with Prony:

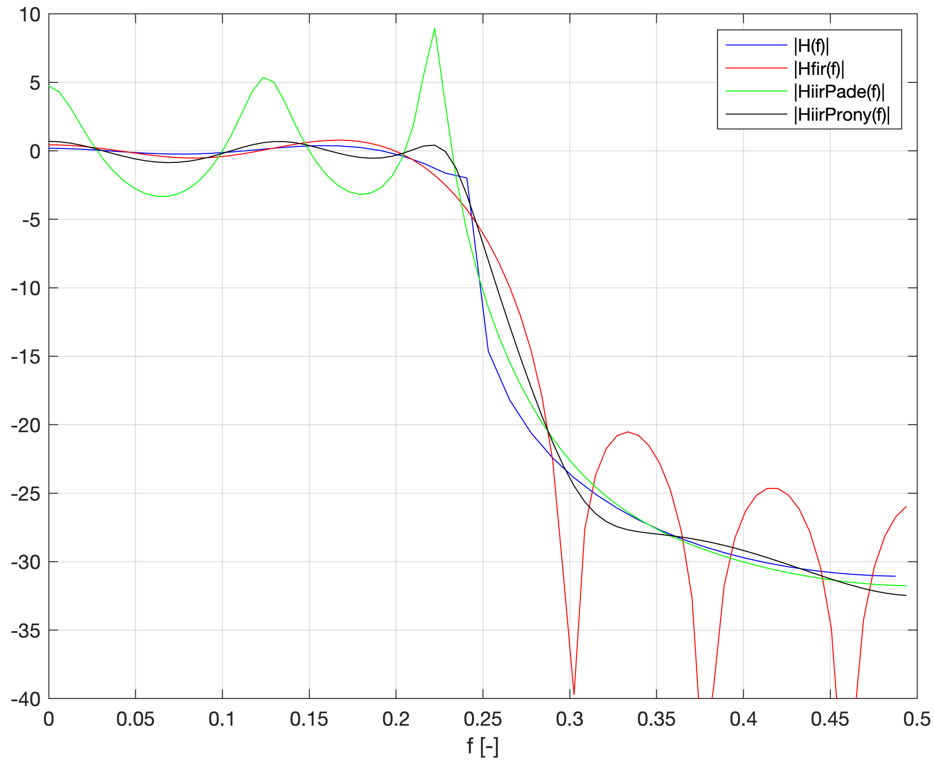
```
[ap_pr,bq_pr,Els_pr] = prony(hn,p,q)
```

Prony Approximation:

```
Calculating the  squared error of  the approximation
ap_pr = 6×1
    1.0000
   -1.9093
    2.3740
   -1.9531
    1.0350
   -0.2893
bq_pr = 6×1
    0.0637
   -0.1216
    0.0450
    0.0783
    0.1323
    0.0810
Els_pr = 0.0011

Hpr =
```

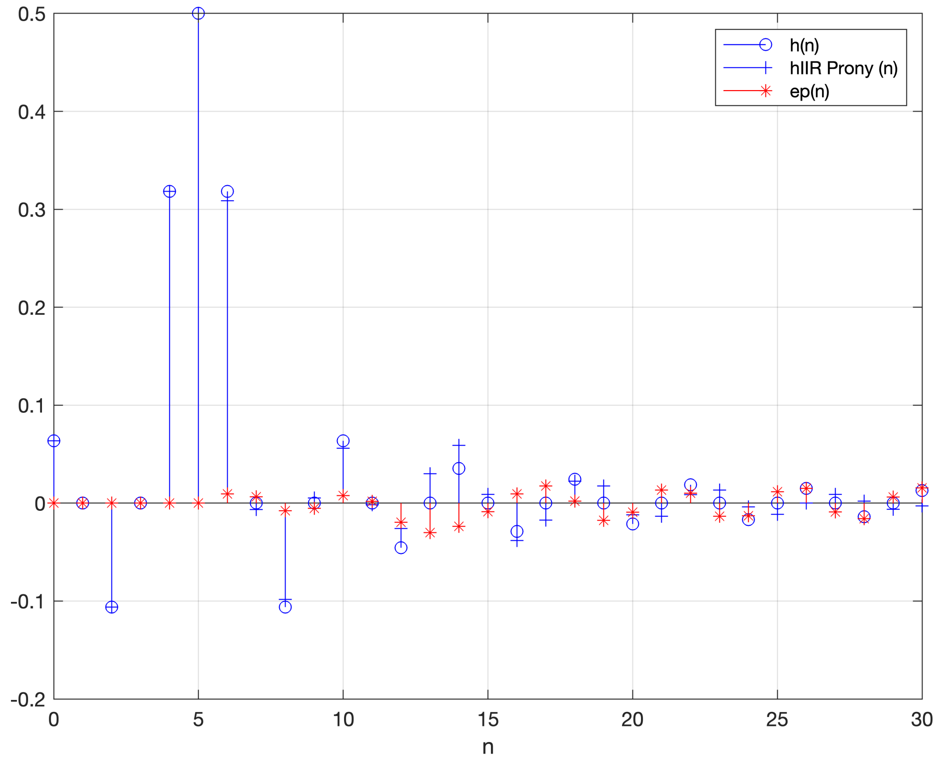
```
  0.06366 z^5 - 0.1216 z^4 + 0.04503 z^3 + 0.07825 z^2 + 0.1323 z + 0.08105
  -------------------------------------------------------------------------
        z^5 - 1.909 z^4 + 2.374 z^3 - 1.953 z^2 + 1.035 z - 0.2893

Sample time: 1 seconds
Discrete-time transfer function.
```

As before we will plot the frequency response compared to the desired frequency response H(f):



The impulse response of the synthetized $\widehat{h}(n)$ filter is compared to the desired impulse response h(n):

```
hpr = 1×81
    0.0637   -0.0000   -0.1061    0.0000    0.3183    0.5000    0.3088   -0.0064 ···

Els = 0.0790
```



Shanks Approximation:

Write the shanks function with the heading as follows:

```
shanks Scompute the Shanks approximation of  a  filter of  ARMA  system with p
 poles  and q zeros usage: [ap,bq,epq,xhat] = shanks(x,p,q)
 epq is optional and is the L2  norm of the approximation error
 xhat is optional  and it is the  approximated signal.
 (c) M.Tognolini HEIG-VD 2022 r1.0
```

Apply of the same example as before for the Prony filter design with p=5 and q=5.

```
[ap_sh,bq_sh,Els_sh] = shanks(hn,p,q)
```

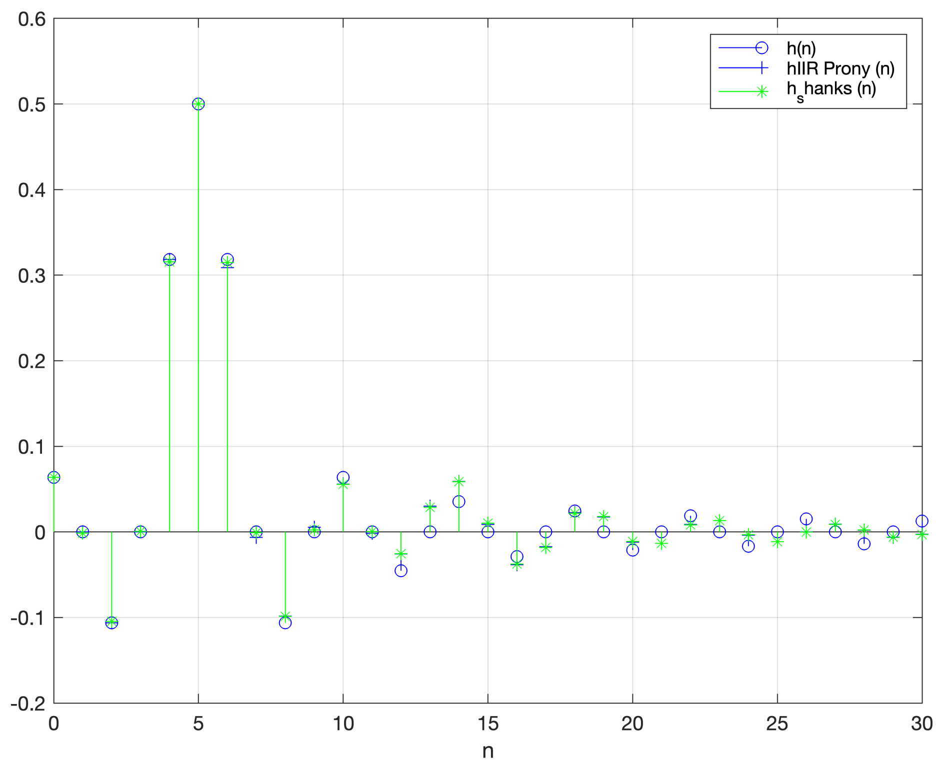
```
Calculating the  squared error of  the approximation
ap_sh = 6×1
    1.0000
   -1.9093
    2.3740
   -1.9531
    1.0350
   -0.2893
bq_sh = 6×1
    0.0637
   -0.1228
    0.0490
    0.0729
    0.1345
```

```
    0.0835
Els_sh = 0.0061
```

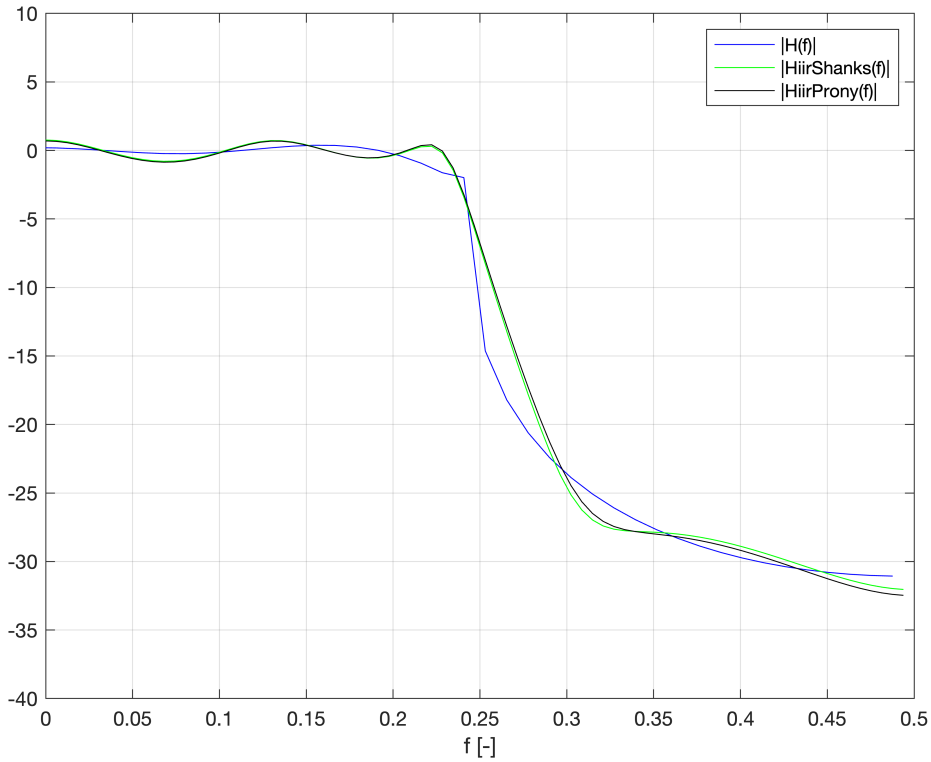
We observe the coefficiens are a bit different compared those obtained with the Prony method.

```
[bq_pr, bq_sh]
```

```
ans = 6×2
    0.0637    0.0637
   -0.1216   -0.1228
    0.0450    0.0490
    0.0783    0.0729
    0.1323    0.1345
    0.0810    0.0835

hsh = 1×81
    0.0637   -0.0012   -0.1044    0.0007    0.3156    0.5000    0.3147   -0.0009 ···

Els = 0.0784
```



As before we will plot the frequency response compared to the desired frequency response H(f):


|H(f)|
|HiirShanks(f)|
|HiirProny(f)|
f [-]