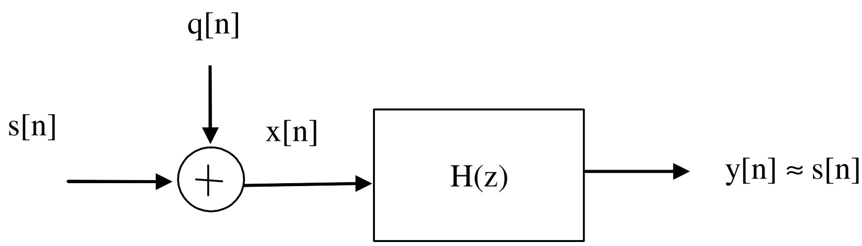
ECGR 6114
Computer Project: Adaptive RLS Filtering
Student Name:___________

Part1
Overview



An adaptive filter is useful for non-stationary systems. Consider the case of designing a filter H(z) to recover a signal s(n) from a corrupted signal
x[n]=s[n]+q[n] where q[n] is noise. In the above adaptive filter formulation, the filter weights are not presumed constant.

As before, the filter 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.

So,

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

then

$$X(z)B(z) = Y(z)A(z)$$

and

$$x[n] * b[n] = y[n] * a[n]$$

in matrix form

$$\mathbf{X}[n]\ \mathbf{b}[n] = \mathbf{Y}[n]\ \mathbf{A}[n]$$

RLS, Recursive Least Squares, is an adaptive filtering implementation of the above problem. Similar to LMS, RLS is useful for finding the filter solution when the process may not be stationary.

The for of RLS we will use is "sliding window" RLS,
only using chunks of data x[n] and y[n] that are of length L

Summary of algorithm:

Initialization:

assume unity filter

$y[m] = x[m]$ for $-1 < m < Nb$

$s[m] = D\{ y[m] \}$ for $-1 < m < Nb$ if using decision-directed s[n]

where D{ } is the slicer/threshold/quatizer function

compute

$$\mathbf{R}[Nb] = \mathbf{X}^H[Nb]\, \mathbf{X}[Nb]$$

$$(\mathbf{R}[Nb])^{-1}$$

$$\mathbf{r}_{sx}[Nb] = \mathbf{X}^H[Nb]\, \mathbf{s}[Nb]$$

$$\mathbf{b}[Nb] = (\mathbf{R}[Nb])^{-1}\, \mathbf{r}_{sx}[Nb]$$

update (a posteriori)

$$y[Nb] = \mathbf{b}^T[Nb]\, \mathbf{x}[Nb]$$

$$s[Nb] = D\{ y[Nb] \}$$

Iteration for n > Nb:

compute

$$\mathbf{R}^{-1}[n] = \mathbf{R}^{-1}[n-1] - (\mathbf{R}^{-1}[n-1]\, \mathbf{x}[n]\mathbf{x}^H[n]\, \mathbf{R}^{-1}[n-1]) / (1 + \mathbf{x}^H[n]\, \mathbf{R}^{-1}[n-1]\, \mathbf{x}[n])$$

$d[n] = D\{ \mathbf{b}^T[n-1]\, \mathbf{x}[n] \}$ a priori aproximation for decision directed

$$\mathbf{r}_{sx}[n] = \mathbf{r}_{sx}[n-1] + s[n]\, \mathbf{x}^*[n]$$

$$\mathbf{b}[n] = (\mathbf{R}[n])^{-1}\, \mathbf{r}_{sx}[n]$$

$s[n] = D\{ \mathbf{b}^T[n]\, \mathbf{x}[n] \}$ a pposteriori final value for decision directed

Part 2
Exponentially weighted RLS

Create a random binary (+/-1) signal s, for FIR filter of length Nb, with signal pulsewidth Nb/4, noise q with uniform pdf from -noise/2 to noise/2,

$N := 2^{12}$ $Nb := 16$ $mu := 0.01$ $pulsewidth := trunc\left(\frac{Nb}{4}\right)$

$noise := 1.5$ $signal := 1$

$n := 0..N-1$ $bb := 0..Nb$ $s_n := signal$ $pulsewidth = 4$

$pp := 0..round\left(\frac{N}{pulsewidth}\right) + 10$ $ss_{pp} := rnd(signal) - \frac{signal}{2}$

$ss := 2 \cdot signal \cdot (rbinom(length(s), 1, 0.5) - 0.5)$

$s_n := ss_{trunc\left(\frac{n}{pulsewidth}\right)}$ $q_n := rnd(noise) - \frac{noise}{2}$

$x_n := s_n + q_n$

x =

	0
0	-1.553
1	-0.729
2	-0.762
3	-1.051
4	-1.103
5	-1.432
6	-1.302
7	-1.01
8	-0.69
9	-0.63
10	-1.004
11	-0.616
12	1.032
13	0.841
14	1.129
15	0.868

$\lambda := 0.95^{\frac{1}{Nb}}$ ZZZZZ

$\lambda = 0.997$ $\lambda^{Nb} = 0.95$

$\lambda = 0.997$ $Nb = 16$ $pulsewidth = 4$

$halfPoint := \frac{\ln(0.5)}{\ln(\lambda)}$ $halfPoint = 216.215$ $\frac{halfPoint}{pulsewidth} = 54.054$

Exponential Weighted FIR-RLS Algorithm
With Decision-Directed s(n)

$RLS(N, Nb, x, s, \lambda) :=$

```
tt ← 1
for bb ∈ 0..Nb
   B_bb ← 0
for nn ∈ 0..Nb
   for bb ∈ 0..Nb
      XALL_nn,bb ← if[(nn − bb) ≥ 0, x_nn−bb, 0]        load all data into XALL
for nn ∈ 0..N
   for bb ∈ 0..Nb                                       Initialize algorithm
      B_nn,bb ← 0
yout ← x·0
sout ← x·0
xn ← submatrix(x, 0, Nb, 0, 0)
Xn ← submatrix(XALL, 0, Nb, 0, Nb)
Rn ← (conj(Xn))^T · Xn
iRn ← Rn^−1
yn ← xn
sn ← yn
for nn ∈ 0..Nb
   sn_nn ← if(yn_nn ≥ 0, 1, −1)
rsxn ← (conj(Xn))^T · sn                                End Initialize algorithm
bn ← iRn·rsxn
yn ← bn·xn
tt ← 1
sn_Nb ← if(yn ≥ 0, 1, −1)
```

```
Nb
for nn ∈ 0..Nb
    yout_nn ← xn_nn
    sout_nn ← sn_nn
tt ← 1
for bb ∈ 0..Nb
    B_Nb,bb ← bn_bb
tt ← 1
for nn ∈ Nb + 1..N − 1
    tt ← 1
    iRold ← iRn
    rsxold ← rsxn
    bold ← bn                                          bold is old value of b
    xn ← submatrix(x, nn, nn − Nb, 0, 0)
    yn ← bold·xn
    sn ← if(yn ≥ 0, 1, −1)
    rsxn ← λ^2·rsxold + sn·conj(xn)                    Recursion for rsx
    Rtop ← λ^(−2)·iRold·(xn (conj(xn))^T)·iRold        Recursion for R
    Rbot ← 1 + λ^(−2)·[conj(xn)·(iRold·xn)]
    iRn ← λ^(−2)·(iRold − Rtop/Rbot)
                                                       compute b
    bn ← iRn·rsxn
    yout_nn ← bn·xn                                    yout is filtered input x
    sout_nn ← if(yout_nn ≥ 0, 1, −1)                   sout is decision
                                                       thresholded output
    for bb ∈ 0..Nb
        B_nn,bb ← bn_bb
return (B, yout, sout)
```

$N = 4.096 \times 10^{3}$ $Nb = 16$

$LL := RLS(N, Nb, x, s, \lambda)$

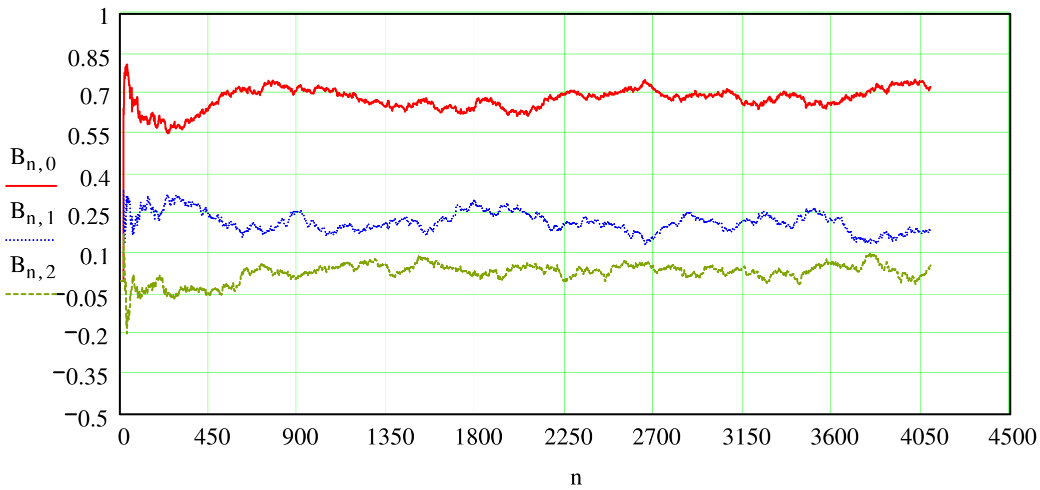
$B := LL_0$ B is array of all b vectors for each timestep

$yout := LL_1$ yout is filter analog output for each timestep

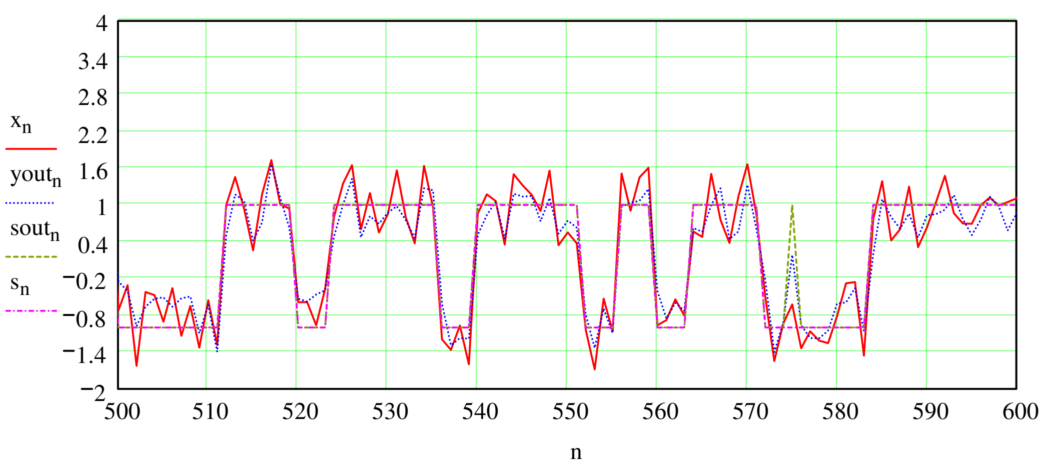
$sout := LL_2$ sout is thresholded yout for each timestep

$Nb = 16$ $\lambda = 0.997$ $signal = 1$ $noise = 1.5$ $N = 4.096 \times 10^{3}$

Below: Plot of coefficient convergence



Below: Plot of output vs input



Q1. What is yout in the plot above?

Q2. What is sout in the plot above?

Part 2
Checking the zeroes of the solution

$$b := \mathrm{submatrix}(B, N-1, N-1, 0, Nb)^T$$

$b =$

	0
0	0.726
1	0.188
2	0.057
3	-0.041
4	-0.087
5	0.028
6	0.012
7	-0.025
8	0.016
9	$-5.506 \cdot 10^{-3}$
10	-0.057
11	$5.475 \cdot 10^{-3}$
12	0.085
13	-0.032
14	-0.071
15	-0.031

$$b_0 = 0.726$$

A better way to compute b, since it bounces around, is to take an average

$$b := \left(\frac{1}{\mathrm{trunc}\left(\frac{N}{8}\right)}\right) \cdot \sum_{k = N-\mathrm{trunc}\left(\frac{N}{8}\right)}^{N-1} \mathrm{submatrix}(B, k, k, 0, Nb)^T$$

$b =$

	0
0	0.713
1	0.179
2	0.045
3	-0.022
4	-0.09
5	0.059
6	$-3.772 \cdot 10^{-4}$
7	-0.013
8	$2.889 \cdot 10^{-3}$
9	$-1.038 \cdot 10^{-3}$
10	-0.026
11	0.01
12	0.014
13	$6.361 \cdot 10^{-3}$
14	-0.015
15	-0.032

Find zeroes

b_roots := polyroots(b)

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

b_roots =

	0
0	-1.157+0.185i
1	-1.157-0.185i
2	-0.997-0.655i
3	-0.997+0.655i
4	-0.62-1.03i
5	-0.62+1.03i
6	-0.241+1.119i
7	-0.241-1.119i
8	0.262+1.159i
9	0.262-1.159i
10	0.796-0.963i
11	0.796+0.963i
12	1.173+0.602i
13	1.173-0.602i
14	1.375
15	1.687

Hzeroes =

	0
0	-0.843-0.135i
1	-0.843+0.135i
2	-0.701+0.461i
3	-0.701-0.461i
4	-0.429+0.712i
5	-0.429-0.712i
6	-0.184-0.854i
7	-0.184+0.854i
8	0.186-0.821i
9	0.186+0.821i
10	0.51+0.617i
11	0.51-0.617i
12	0.675-0.346i
13	0.675+0.346i
14	0.727
15	0.593

Lets plot Zeroes in z-plane

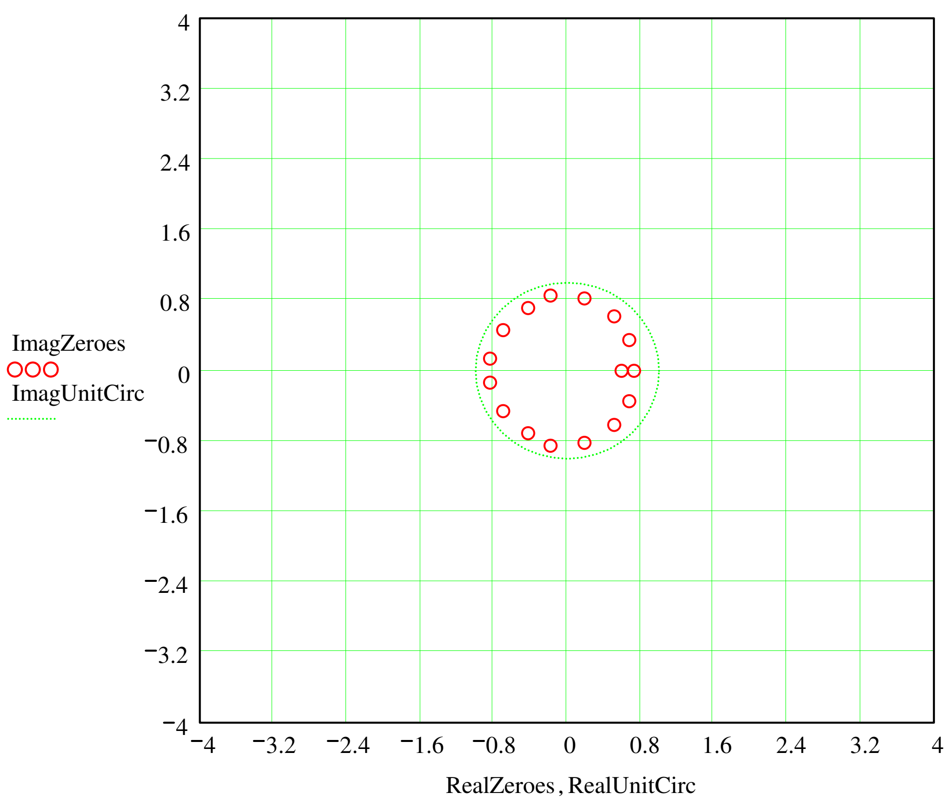
$ImagZeroes := Im(Hzeroes)$ $RealZeroes := Re(Hzeroes)$

Lets plot a unit circle too

$un := 0..40$

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

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



Q3. Is the RLS solution stable?

Part 3
Frequency response of the solution

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

$$\mathrm{Hrls}(num, z) := \sum_{k=0}^{rows(num)-1} num_k \cdot z^{-k}$$

So check a few important points:

$$\mathrm{Hrls}\left(b, e^{i \cdot 0.25}\right) = 0.899 - 0.018i$$

Plot the frequency reponse of the bilinear transform filter

$$\mathrm{H\_rls}(\omega) := \mathrm{Hrls}\left(b, e^{i \cdot \omega}\right)$$

$\mathrm{H\_rls}(0) = 0.853$

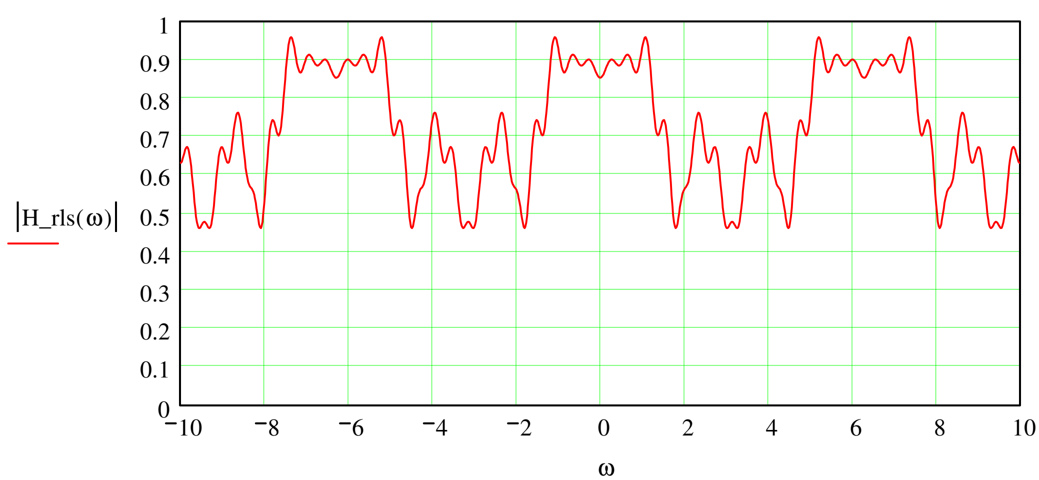
$\mathrm{Hrls}(b, -1) = 0.478$

b =

	0
0	0.713
1	0.179
2	0.045
3	-0.022
4	-0.09
5	0.059
6	$-3.772 \cdot 10^{-4}$
7	-0.013
8	$2.889 \cdot 10^{-3}$
9	$-1.038 \cdot 10^{-3}$
10	-0.026
11	0.01
12	0.014
13	$6.361 \cdot 10^{-3}$
14	-0.015
15	-0.032

FIR RLS filter

$\lambda = 0.997$ $\quad Nb = 16$ $\quad signal = 1$ $\quad noise = 1.5$ $\quad N = 4.096 \times 10^3$



Part 4
Comparison to FIR Wiener filter

FIR wiener filter is set by autororrelation matrix

B= $\mathbf{R}^{-1}\,\mathbf{r}_{sx}$

$maxlag := Nb$

$bb := 0..maxlag \qquad bbb := 0..maxlag$

$$rx_{bb} := \left(\frac{1}{N-bb}\right)\cdot\sum_{k=bb}^{N-1} x_k\cdot x_{k-bb}$$

$$rsx_{bb} := \left(\frac{1}{N-bb}\right)\cdot\sum_{k=bb}^{N-1} s_k\cdot x_{k-bb}$$

$$rs_{bb} := \left(\frac{1}{N-bb}\right)\cdot\sum_{k=bb}^{N-1} s_k\cdot s_{k-bb}$$

$$rq_{bb} := \left(\frac{1}{N-bb}\right)\cdot\sum_{k=bb}^{N-1} q_k\cdot q_{k-bb}$$

rx =

	0
0	1.18
1	0.739
2	0.489
3	0.236
4	-0.016
5	-0.015
6	$-9.581\cdot10^{-3}$
7	$6.721\cdot10^{-3}$
8	$-3.56\cdot10^{-3}$
9	$4.456\cdot10^{-3}$
10	$3.381\cdot10^{-4}$
11	-0.012
12	$4.945\cdot10^{-3}$
13	$-1.826\cdot10^{-3}$
14	-0.021
15	-0.032

rsx =

	0
0	0.993
1	0.732
2	0.482
3	0.227
4	-0.02
5	$-8.56\cdot10^{-3}$
6	$-5.074\cdot10^{-3}$
7	$6.811\cdot10^{-3}$
8	$7.094\cdot10^{-3}$
9	$9.312\cdot10^{-3}$
10	$4.732\cdot10^{-3}$
11	$-1.184\cdot10^{-3}$
12	$3.208\cdot10^{-3}$
13	-0.012
14	-0.025
15	-0.031

rs =

	0
0	1
1	0.747
2	0.493
3	0.239
4	-0.015
5	$-9.044\cdot10^{-3}$
6	$-3.423\cdot10^{-3}$
7	$2.201\cdot10^{-3}$
8	$7.828\cdot10^{-3}$
9	$4.649\cdot10^{-3}$
10	$1.468\cdot10^{-3}$
11	$-1.714\cdot10^{-3}$
12	$-4.897\cdot10^{-3}$
13	-0.012
14	-0.019
15	-0.026

Note rsx equals rs as we expect for this problem

$R_{bb,bbb} := \text{if}\left(bb - bbb > 0, rx_{bb-bbb}, rx_{bbb-bb}\right)$

R =

	0	1	2	3	4	5
0	1.18	0.739	0.489	0.236	-0.016	-0.015
1	0.739	1.18	0.739	0.489	0.236	-0.016
2	0.489	0.739	1.18	0.739	0.489	0.236
3	0.236	0.489	0.739	1.18	0.739	0.489
4	-0.016	0.236	0.489	0.739	1.18	0.739
5	-0.015	-0.016	0.236	0.489	0.739	1.18
6	$-9.581\cdot10^{-3}$	-0.015	-0.016	0.236	0.489	0.739
7	$6.721\cdot10^{-3}$	$-9.581\cdot10^{-3}$	-0.015	-0.016	0.236	0.489
8	$-3.56\cdot10^{-3}$	$6.721\cdot10^{-3}$	$-9.581\cdot10^{-3}$	-0.015	-0.016	0.236
9	$4.456\cdot10^{-3}$	$-3.56\cdot10^{-3}$	$6.721\cdot10^{-3}$	$-9.581\cdot10^{-3}$	-0.015	-0.016
10	$3.381\cdot10^{-4}$	$4.456\cdot10^{-3}$	$-3.56\cdot10^{-3}$	$6.721\cdot10^{-3}$	$-9.581\cdot10^{-3}$	-0.015
11	-0.012	$3.381\cdot10^{-4}$	$4.456\cdot10^{-3}$	$-3.56\cdot10^{-3}$	$6.721\cdot10^{-3}$	$-9.581\cdot10^{-3}$
12	$4.945\cdot10^{-3}$	-0.012	$3.381\cdot10^{-4}$	$4.456\cdot10^{-3}$	$-3.56\cdot10^{-3}$	$6.721\cdot10^{-3}$
13	$-1.826\cdot10^{-3}$	$4.945\cdot10^{-3}$	-0.012	$3.381\cdot10^{-4}$	$4.456\cdot10^{-3}$	$-3.56\cdot10^{-3}$
14	-0.021	$-1.826\cdot10^{-3}$	$4.945\cdot10^{-3}$	-0.012	$3.381\cdot10^{-4}$	$4.456\cdot10^{-3}$
15	-0.032	-0.021	$-1.826\cdot10^{-3}$	$4.945\cdot10^{-3}$	-0.012	$3.381\cdot10^{-4}$

$Bwiener := R^{-1}\cdot rsx$

Bwiener =

	0
0	0.724
1	0.168
2	0.04
3	-0.02
4	-0.083
5	0.049
6	0.022
7	$-7.347\cdot10^{-3}$
8	-0.015
9	$6.58\cdot10^{-3}$
10	$1.239\cdot10^{-3}$
11	0.013
12	$1.598\cdot10^{-3}$
13	-0.013
14	$-7.39\cdot10^{-3}$
15	$2.53\cdot10^{-3}$

Comparison of Bwiener to B

b =

	0
0	0.713
1	0.179
2	0.045
3	-0.022
4	-0.09
5	0.059
6	$-3.772 \cdot 10^{-4}$
7	-0.013
8	$2.889 \cdot 10^{-3}$
9	$-1.038 \cdot 10^{-3}$
10	-0.026
11	0.01
12	0.014
13	$6.361 \cdot 10^{-3}$
14	-0.015
15	-0.032

Bwiener =

	0
0	0.724
1	0.168
2	0.04
3	-0.02
4	-0.083
5	0.049
6	0.022
7	$-7.347 \cdot 10^{-3}$
8	-0.015
9	$6.58 \cdot 10^{-3}$
10	$1.239 \cdot 10^{-3}$
11	0.013
12	$1.598 \cdot 10^{-3}$
13	-0.013
14	$-7.39 \cdot 10^{-3}$
15	$2.53 \cdot 10^{-3}$

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

$$\mathrm{Hrls}(\mathrm{num}, z) := \sum_{k=0}^{\mathrm{rows(num)}-1} b_k \cdot z^{-k}$$

$$\mathrm{Hwien}(\mathrm{num}, z) := \sum_{k=0}^{\mathrm{rows(num)}-1} \mathrm{Bwiener}_k \cdot z^{-k}$$

So check a few important points:

$$\mathrm{Hrls}\left(b, e^{i \cdot 0.25}\right) = 0.899 - 0.018i$$

$$\mathrm{Hrls}(b, -1) = 0.478$$

Plot the frequency reponse of the filter

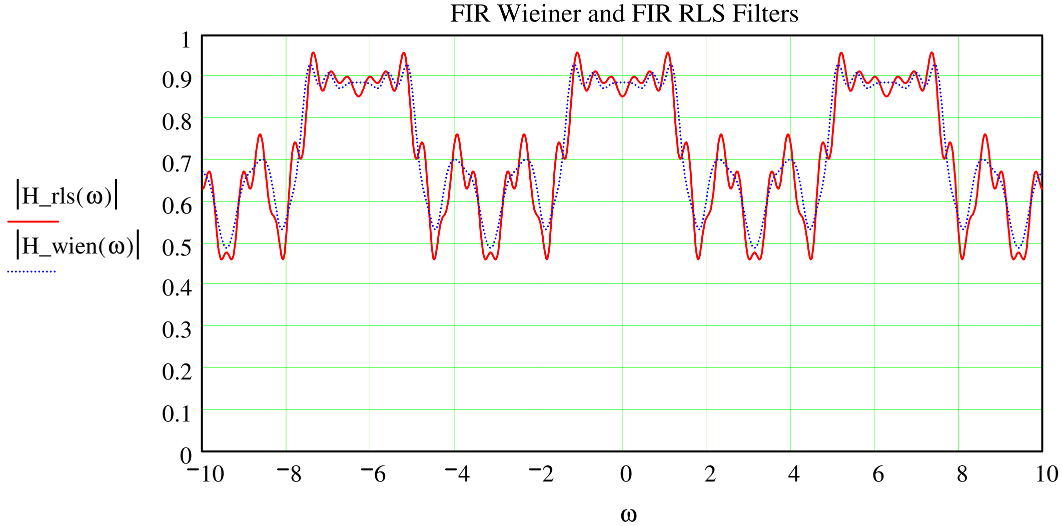
$$\mathrm{H\_rls}(\omega) := \mathrm{Hrls}\left(b, e^{i \cdot \omega}\right)$$

$$\mathrm{H\_rls}(0) = 0.853$$

$$\mathrm{H\_wien}(\omega) := \mathrm{Hwien}\left(\mathrm{Bwiener}, e^{i \cdot \omega}\right)$$

Comparison of Ideal wieiner filter to RLS filter response $\quad$ pulsewidth = 4

$\lambda = 0.997 \quad Nb = 16 \quad signal = 1 \quad noise = 1.5 \quad N = 4.096 \times 10^{3}$



Q4. Momentarily change $\lambda := 0.95^{\frac{1}{Nb}}$ to $\lambda := 0.7^{\frac{1}{Nb}}$ above at the **ZZZZZ** marker, and state a) whether the coefficients converge to the same values as before and b) whether errors increase. Return λ to the original settings.

Q5. Return the value of $\lambda := 0.95^{\frac{1}{Nb}}$ Does the RLS filter approximate the FIR Wiener frequency response?

Q6. Reset the filter order to Nb=12 above (leave it this way for the report that you turn in).

Q7. With Nb=12, is the RLS filter a better or worse approximation of the FIR Wiener frequency response?