University of Moratuwa

Department of Electronic and Telecommunication Engineering

BM4152

Bio-signal Processing

Assignment 2

K. P. Tharuka

200641T

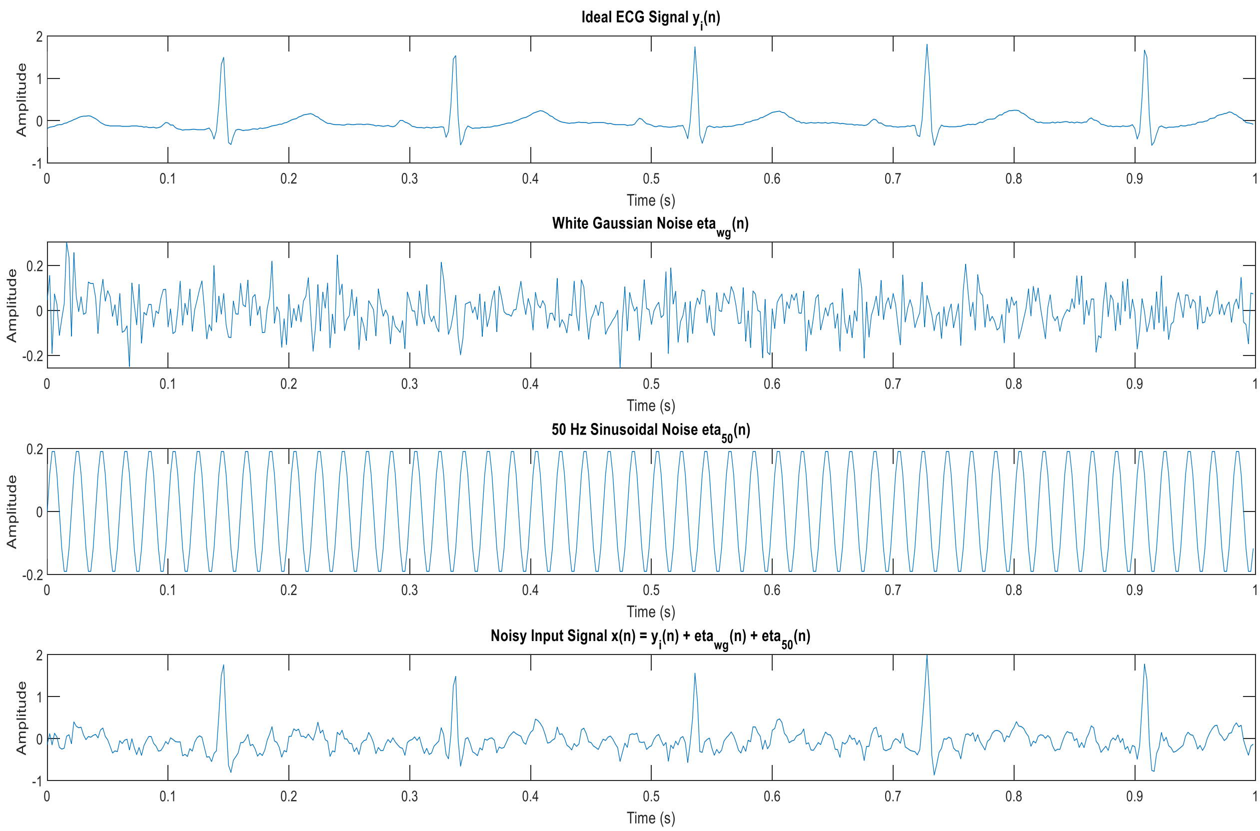
Contents

1. Weiner filtering

1.1. Discrete time-domain implementation of the Wiener filter

Data construction

- Ideal signal $yi(n)$:
- Input signal $x(n) = yi\ (n) + \eta(n)$
- Where $\eta(n) = \eta wg(n) + \eta 50(n)$
- $\eta wg(n)$: white Gaussian noise such that SNR is 10 dB with respect to $yi\ (n)$
- $\eta 50(n) = 0.2 \sin\ (2\pi 50 n)$



Part1

a)

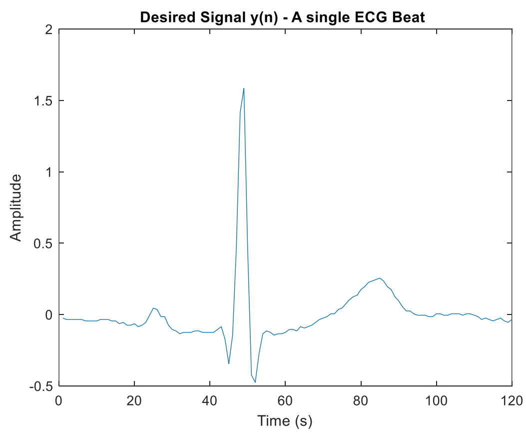
Here we have to calculate the optimum weight vector using the equation $W_0 = (\Phi Y + \Phi N)^{-1}\Theta Y y$ for an arbitrary filter order.

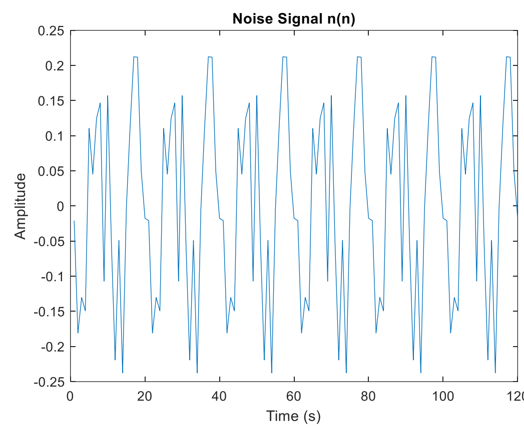
For that we need to specify the desired signal $y_i(n)$ and the noise signal $n(n)$.

For the desired signal I selected a random ECG beat from the given ideal ECG signal. Also, noisy signal was extracted by obtaining a signal segment from the T wave of any chosen ECG beat to the P wave of the next ECG beat (isoelectric segment) of $x(n)$.

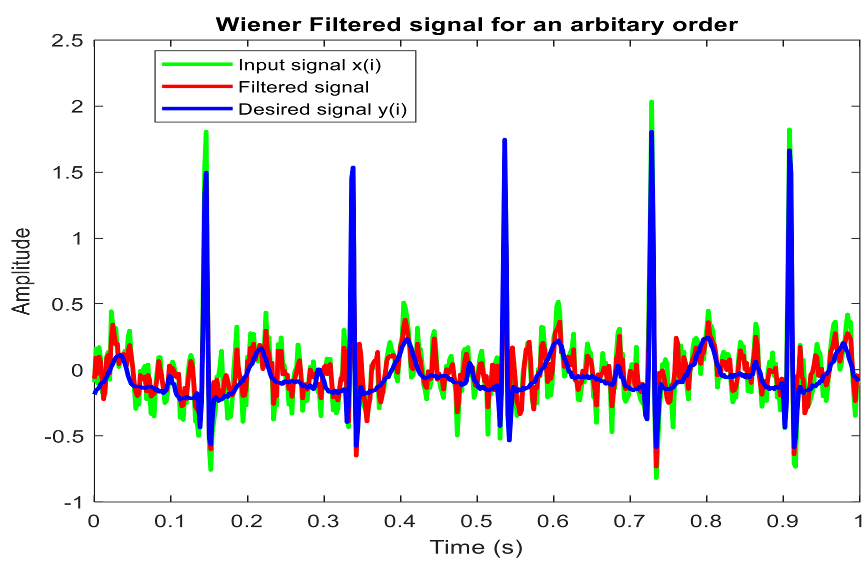
Length of the desired ECG signal = 120 samples.

Length of the noise signal segment (iso electric segment) = 20 samples (therefore, this segment is repeated 6 times to get a noisy signal with the same length of desired ECG signal).

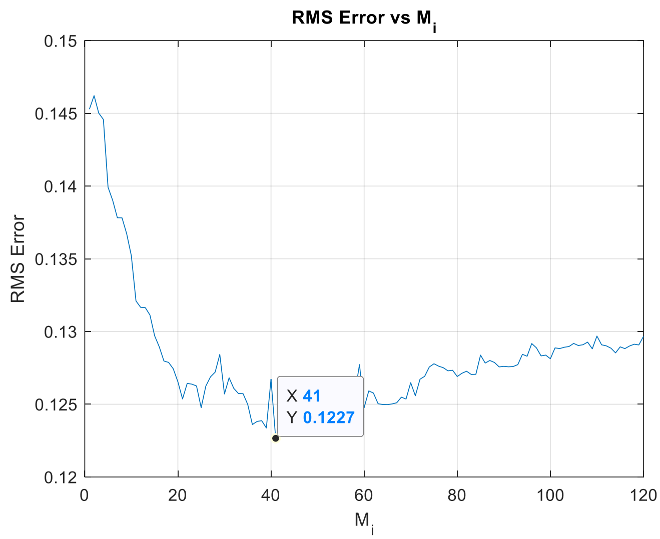




Then the filter order is selected as 20 - the arbitary filter order.

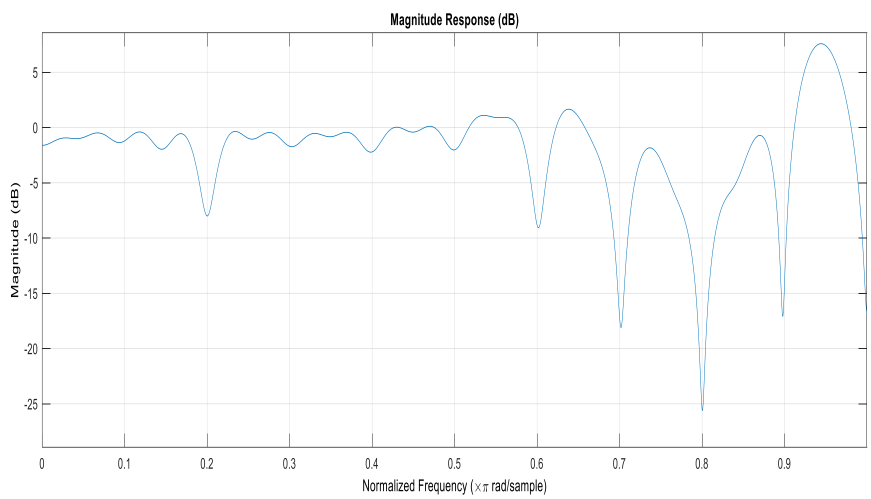


b) Then the optimum filter order is obtained by evaluating the Mean Square Error of the filtered signal in comparison to the ideal signal.



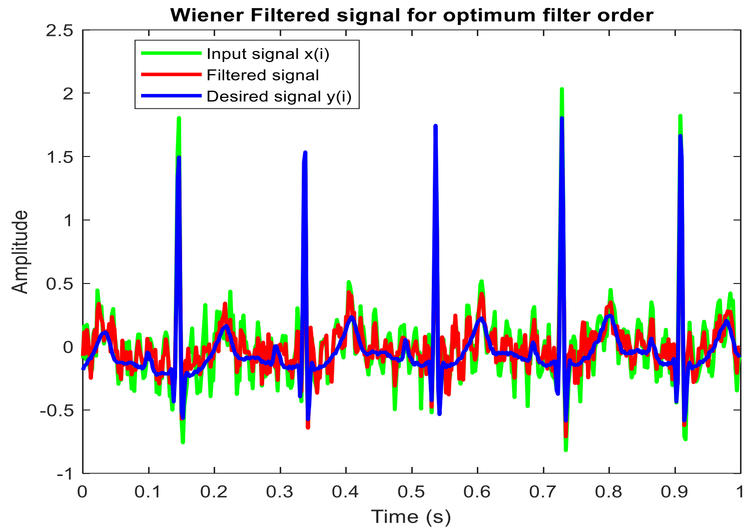
Therefore, Optimum filter order is 41.

Magnitude response of the optimum filter order is as follows.

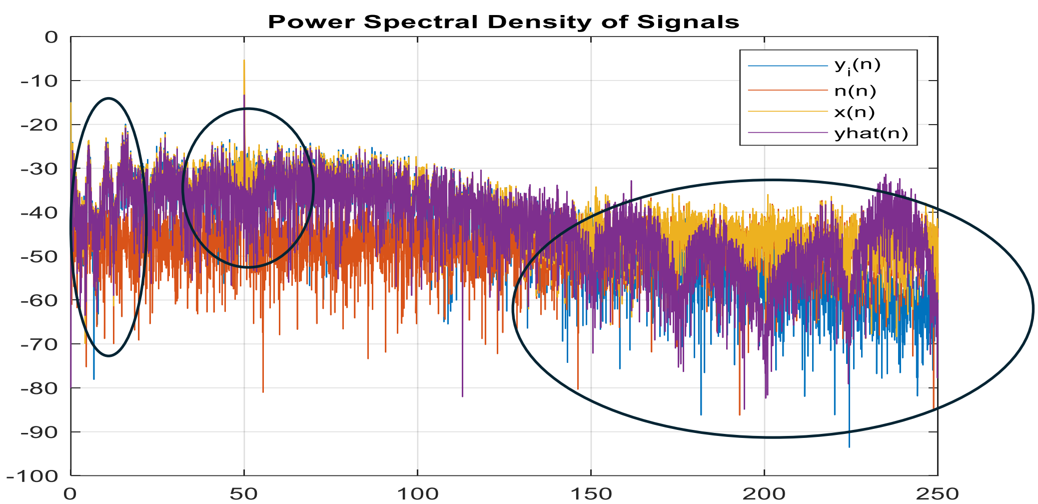


Here we can clearly see a notch at 0.2*250 =50Hz. And clear attenuations of the amplitude after 150Hz. Which are reasonable values for ECG signals and with the added noise.

c)



d)



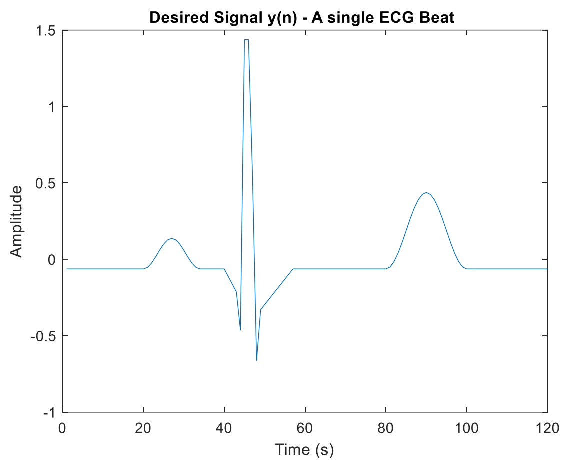
e)

We can clearly observe that the optimum wiener filter has successfully filtered the high frequencies (frequencies higher than the 150Hz) low frequencies and the 50 Hz noise interference.

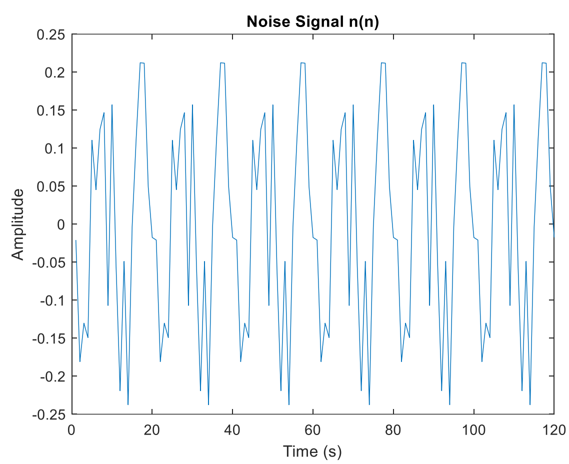
Part 2

Here we have to construct manually a model of an ideal ECG beat. Because in practice we don't have the ideal noise free ECG signal.

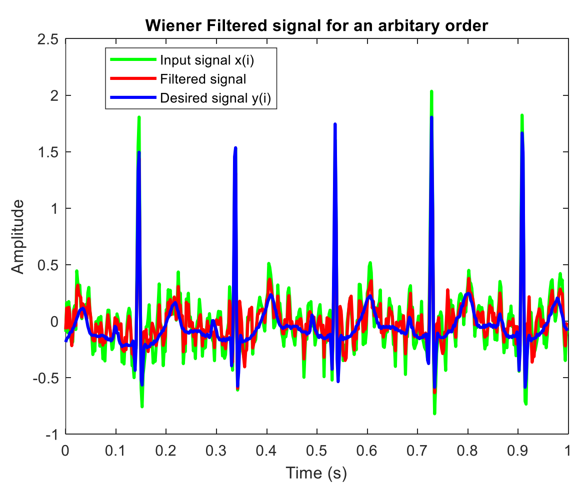
Therefore, I constructed a model of an ECG signal (which is not a linear though).



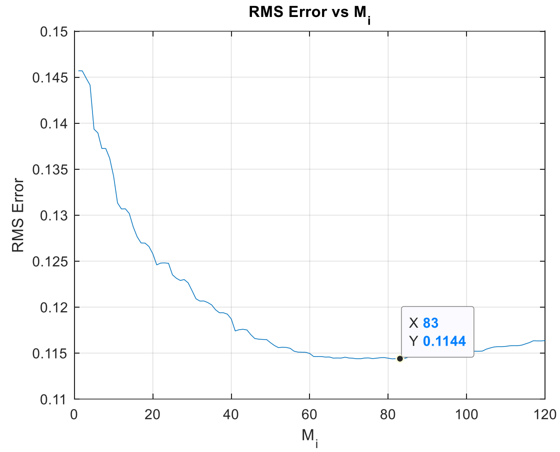
Whereas we can extract the noisy signal as previous, Therefore, noisy signal was extracted by obtaining a signal segment from the T wave of any chosen ECG beat to the P wave of the next ECG beat (isoelectric segment) of $x(n)$.



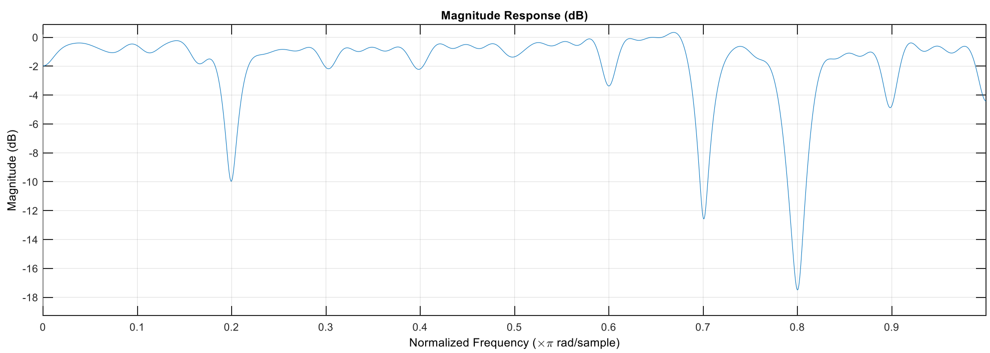
Then for an arbitrary filter order(20) the filtered signal is like below;



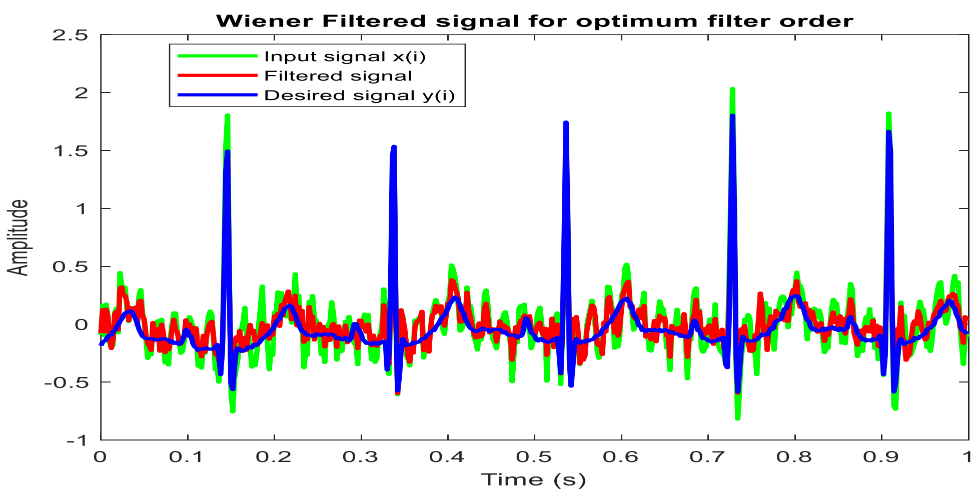
Then the optimum filter order was found similar to the previous one.

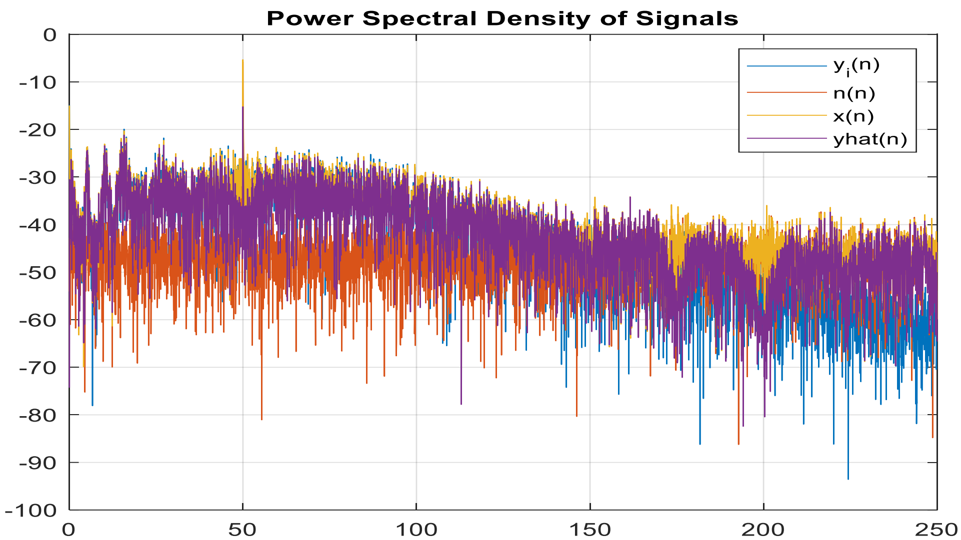


According to that the optimum filter order is 83. And the frequency response of the filter is;



Here also, we can clearly see a large notch at 0.2*250 =50Hz. And clear attenuations of the amplitude after 150Hz.





Here also we can clearly observe that the optimum wiener filter has successfully filtered the high frequencies (frequencies higher than the 150Hz) low frequencies and the 50 Hz noise interference similar to the previous regardless of the use of a modeled ECG signal rather an actual one.

1.2. Frequency domain implementation of the Wiener filter

Weights of the wiener filter is obtained using,

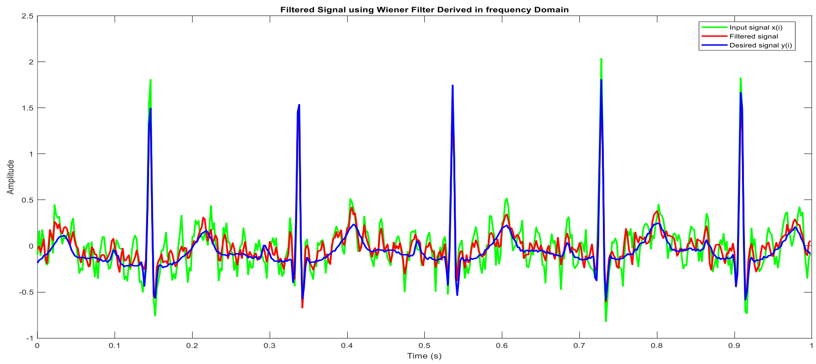
$$W(f) = \frac{S_{YY}(f)}{S_{YY}(f) + S_{NN}(f)}$$

Where, $S_{ZZ}(f)$ is the PSD of the signal $z(n)$. Alternatively, $S_{ZZ}(f)$ is the power of the fourier transform ($\mathcal{F}\{.\}$) of the template $z(n)$. Therefore, $S_{ZZ}(f) = |\mathcal{F}\{z(n)\}|^2$.

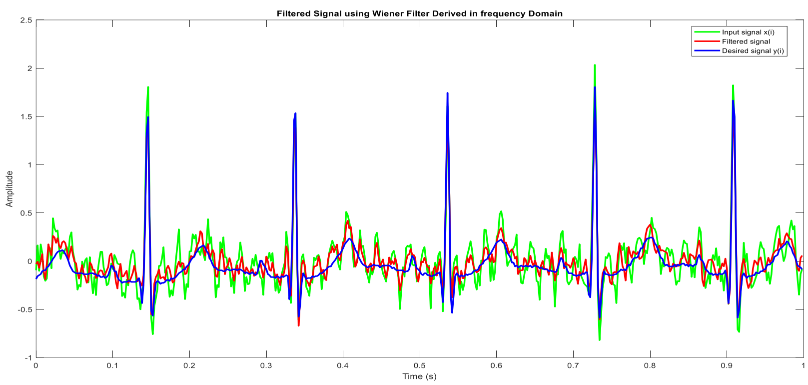
After finding the weights the filtered signal is also obtained in frequency domain and then converted back into the time domain after filtering.

a) This question requires to implement the frequency domain version of the Wiener filter for the two cases (Part 1 and Part 2 above).

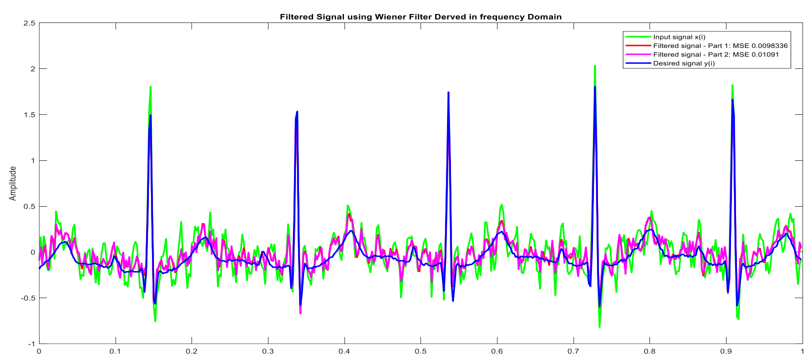
Part 1



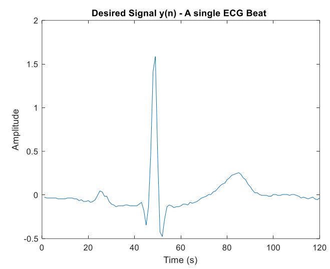
Part 2

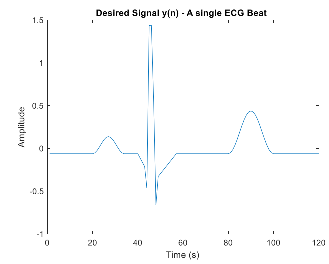


b)



In part 1 we used an ideal ECG signal (a real ECG signal) as the desired ECG signal whereas in real world we can't get such ground truth noise free ECG. Therefore, the solution is to use a mathematically modeled ECG signal as the desired signal; therefore in part 2 a mathematically modeled ECG signal model is used.



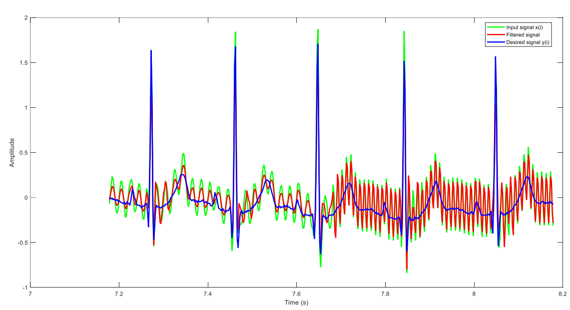


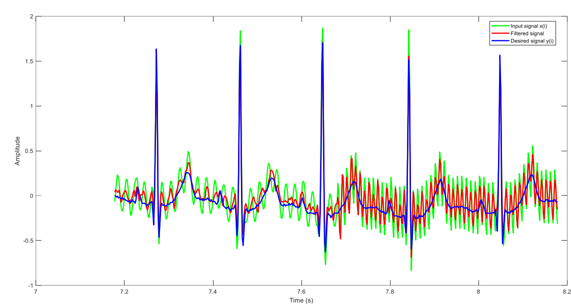
Part 1 part 2

According to the results both filtered (according to part1 and part2) signal are almost identical. Whereas, if we observe the MSE of filtered signals we can infer that the desired signal used in part1 performs slightly better than the part 2 desired signal which is expected because the ideal ECG signal always contains more characteristics of an ECG beat than a modeled ECG signal and also the results should vary with the fact that how closely the modeled ECG beat is similar to an actual ECG beat.

1.3 Effect on non-stationary noise on Wiener filtering

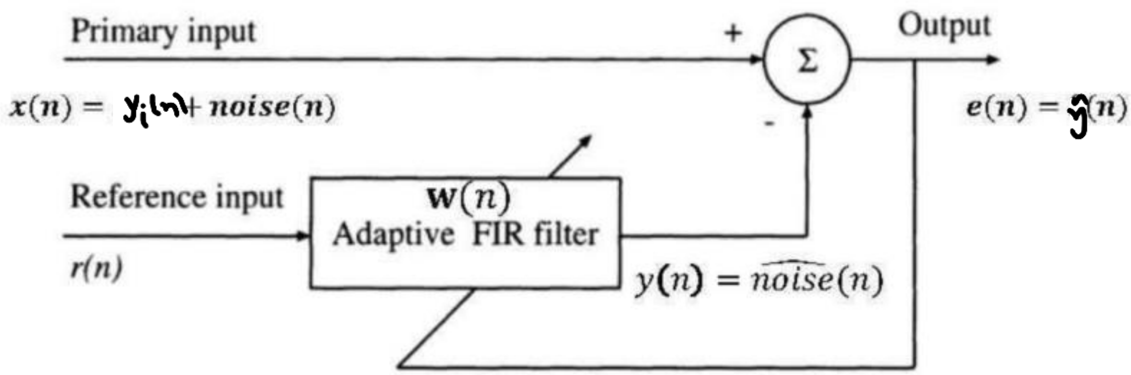
Part 1 1.1 Part 1 1.2





We can observe that the wiener filter is unable to filter the 100Hz noise as effective as 50Hz noise(we can observe that in magnitude response of the filter as well.) that is the wiener filter fails to filter nonstationary noise. Because it assumes the noise and the signal are stationary. Solution for that is the adaptive filter which changes it weights over time.

2. Adaptive filtering



Here, **x(n) is the primary input signal** which we need to filter, and simply it is $y_i(n) + n(n)$. Where, $y_i(n)$ is the desired (noise free) signal and n(n) is the noise. Whereas the signal and the noise don't need to be stationary as in Wiener filter.

However, the processes that generate the desired signal $y_i(n)$ and the noise n(n) should be uncorrelated.

In adaptive filtering we use another input named **Reference input signal r(n),** it is a signal that is closely related to the noise n(n); because we can't exactly find the actual noise signal in practice, so we get a signal that is closely related to the noise. But it should be uncorrelated with the desired signal $y_i(n)$.

Adaptive FIR filter modifies the reference input r(n) s.t. it is as close to the actual noise as possible . then that modified reference signal or the estimation for the noisy signal is subtracted from the primary input signal. Which will give an error signal e(n) s.t. it is close to the desired signal , $y_i(n)$. How close it is can be measured using the absolute error or mean square error.

The word adaptive filter referred to that it is capable of changing its filter coefficients according to the reference signal and the feedback from the output signal y(n)

There is a closed form solution as well as an iterative method to estimate the filter coefficients(tap-weights).

Closed form solution is given by; $\mathbf{w}(n) = \Phi_{\mathbf{R}}^{-1}(n)\Theta_{\mathbf{R}x}(n)$. But matrix inversion is not a healthy option for a higher order matrix. Therefore, the most commonly used adaptive filters are

LMS filter and the RLS filter.

2.1 LMS method

a)

In adaptive filtering algorithm tap weights are updated so that it minimizes the squared error e(n)

$$e^2(n) = \left(x(n) - \mathbf{w}^T(n)\mathbf{R}(n)\right)^2$$

We can consider the squared error as a loss function. Therefore, we can update tap -weights or minimize the loss function using the gradient descent algorithm.

$$J\big(\mathbf{w}(n)\big) = x^2(n) - 2x(n)\mathbf{w}^T(n)\mathbf{R}(n) + \mathbf{w}^T(n)\mathbf{R}(n)\mathbf{R}^T(n)\mathbf{w}(n)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla(n)$$

Here, $\nabla(n)$ = gradient of the squared error

And it is equals to $-2e(n)\mathbf{R}^T(n)$ then.

$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu e(n)\mathbf{R}^T(n)$$

This is what it does in LMS algorithm.

In this particular case.

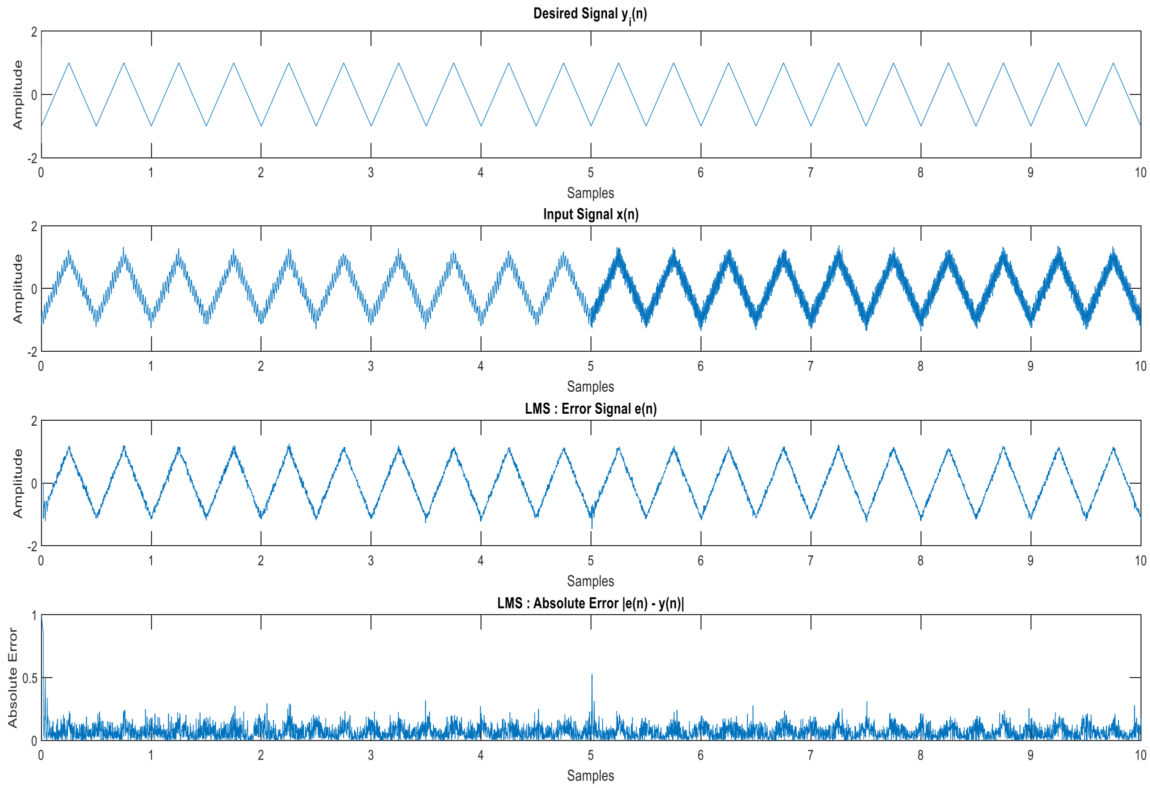
- $x(n) = y_i(n) + n(n)$.
- $y_i(n)$ is sawtooth waveform with a width of 0.5. It has 5000 samples and sampling rate is 500Hz.
- $\eta(n) = \begin{cases} \eta_{wg}(n) + \eta_{50}(n) & 0 \le n < T/2 \\ \eta_{wg}(n) + \eta_{100}(n) & T/2 \le n < T \end{cases}$ where, T = 2500.
- $r(n) = a(\eta_{wg}(n) + \sin(2\pi 50n + \phi_1) + \sin(2\pi 100n + \phi_2))$ where a, ϕ_1, ϕ_2 are arbitrary constants

Selected those arbitrary constants as.

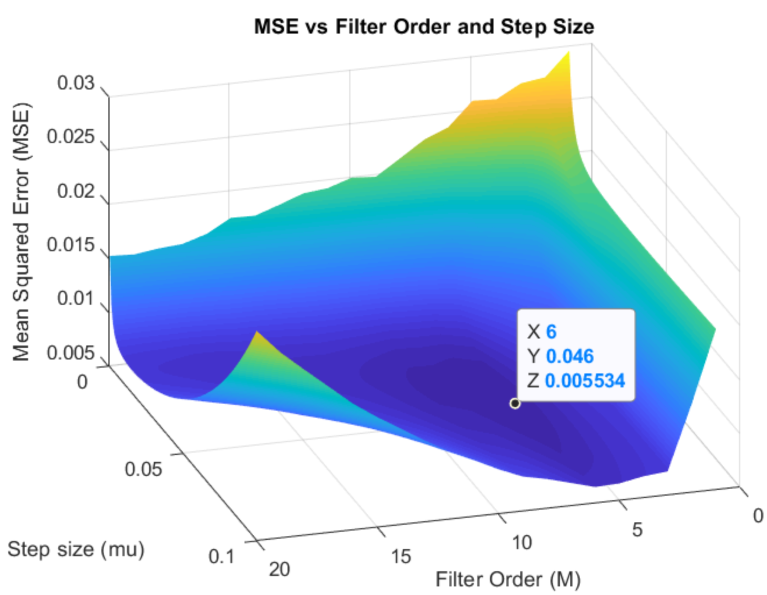
a =0.3, $\emptyset_1 = \pi/6$, $\emptyset_2 = \pi/4$

b)

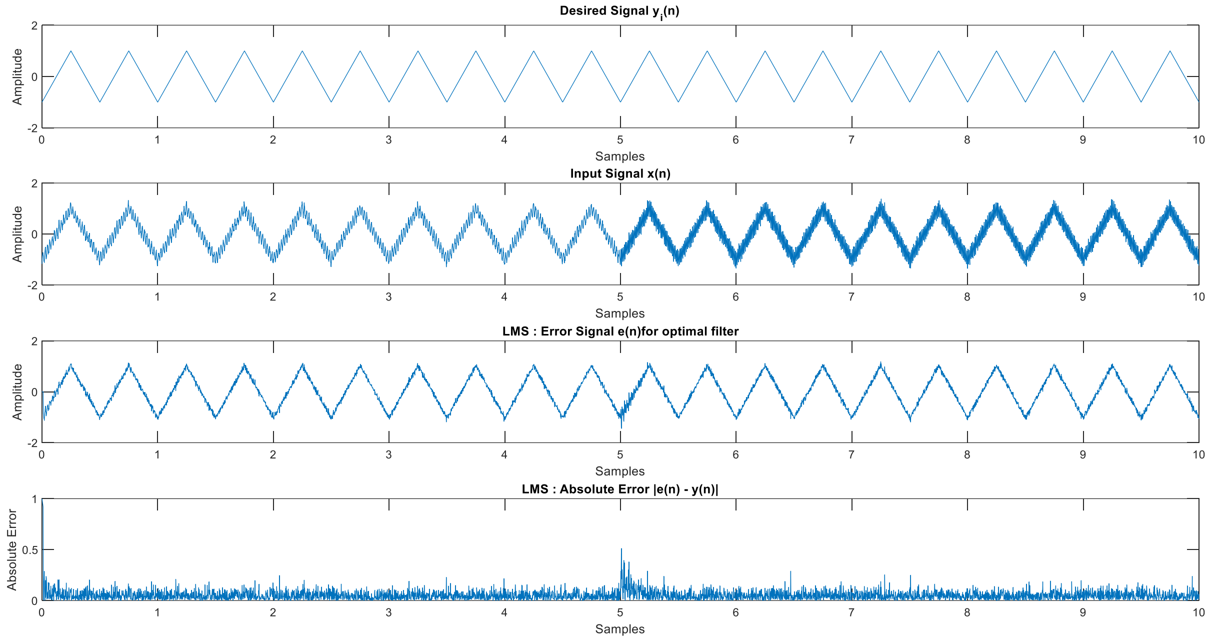
Arbitrary filter values: order = 10 and μ(step size) = 0.1



b)



Optimum filter order = 6 and μ(step size) = 0.046 => MSE = 0.005534



2.1 RLS method

a)

Similar to the LMS method, the RLS method is also an algorithm that minimizes the square error function. Whereas in this algorithm it also considers the errors of previous square errors as well instead of just using the current square error. So, it introduces a forgetting factor which determines the weight that gives to the previous square errors.

It is also an iterative method, and the weights are updated according to the following algorithm.

for each n:

$$\mathbf{k}(n) = \frac{\lambda^{-1}\mathbf{P}(n-1)\mathbf{r}(n)}{1 + \lambda^{-1}\mathbf{r}^T(n)\mathbf{P}(n-1)\mathbf{r}(n)}$$

$$\mathbf{P}(n) = \lambda^{-1}\mathbf{P}(n-1) - \lambda^{-1}\mathbf{k}(n)\mathbf{r}^T(n)\mathbf{P}(n-1)$$

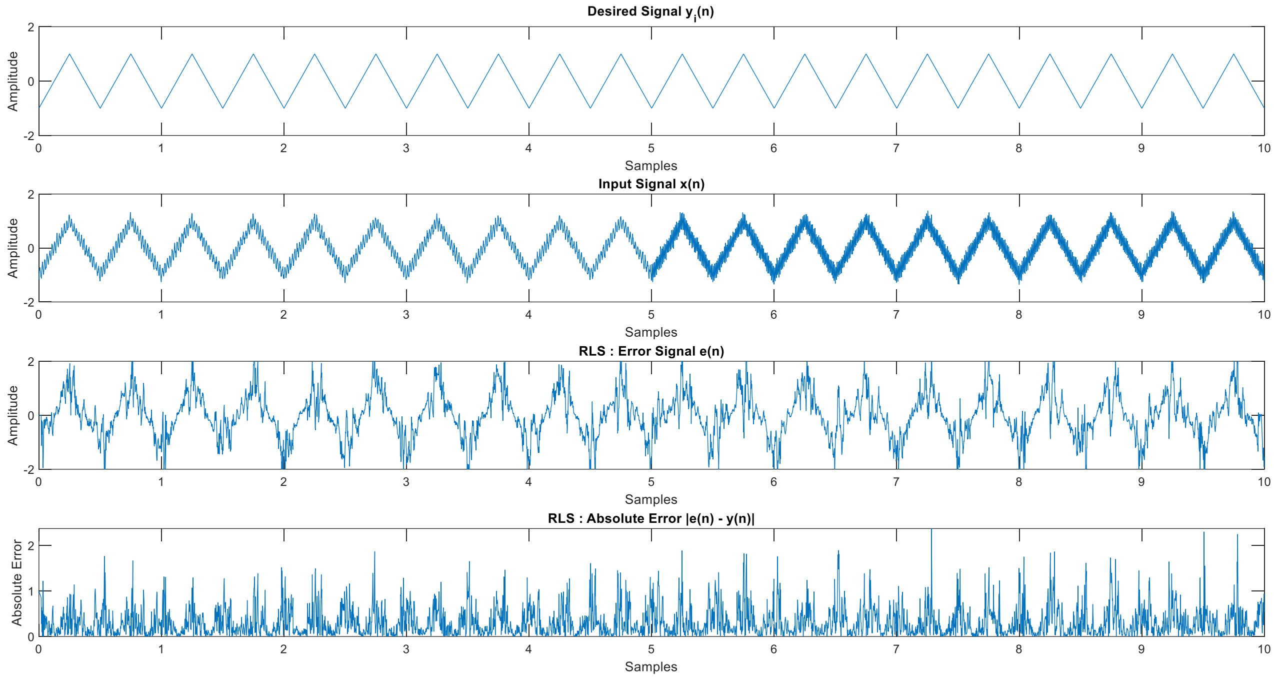
$$\alpha(n) = x(n) - \mathbf{w}^T(n-1)\mathbf{r}(n)$$

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mathbf{k}(n)\alpha(n)$$

$$\alpha(n) = \hat{s}(n) = x(n) - \mathbf{w}^T(n)\mathbf{r}(n)$$

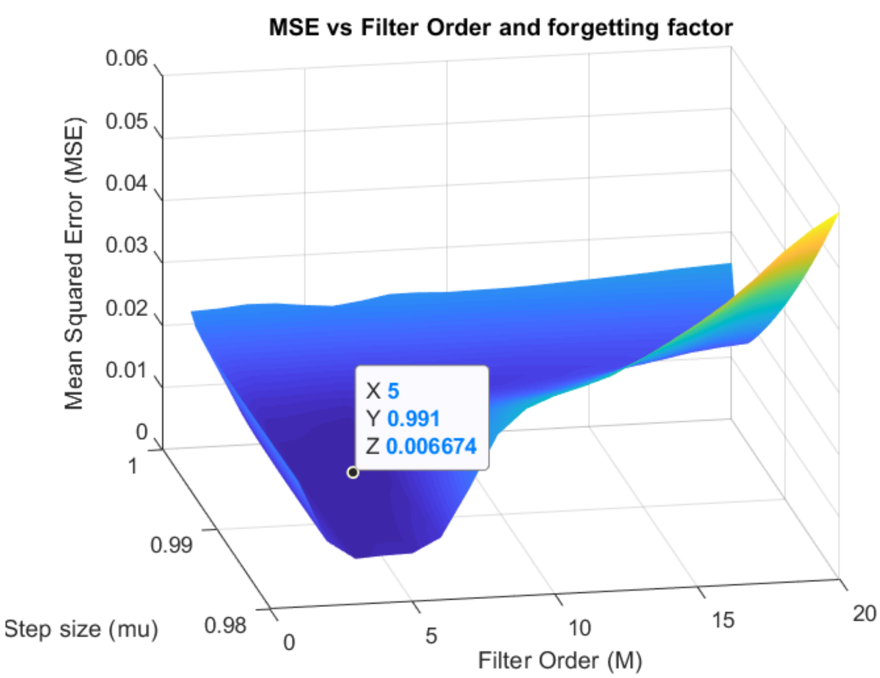
b)

Arbitrary filter values: order = 10 and μ(step size) = 0.1

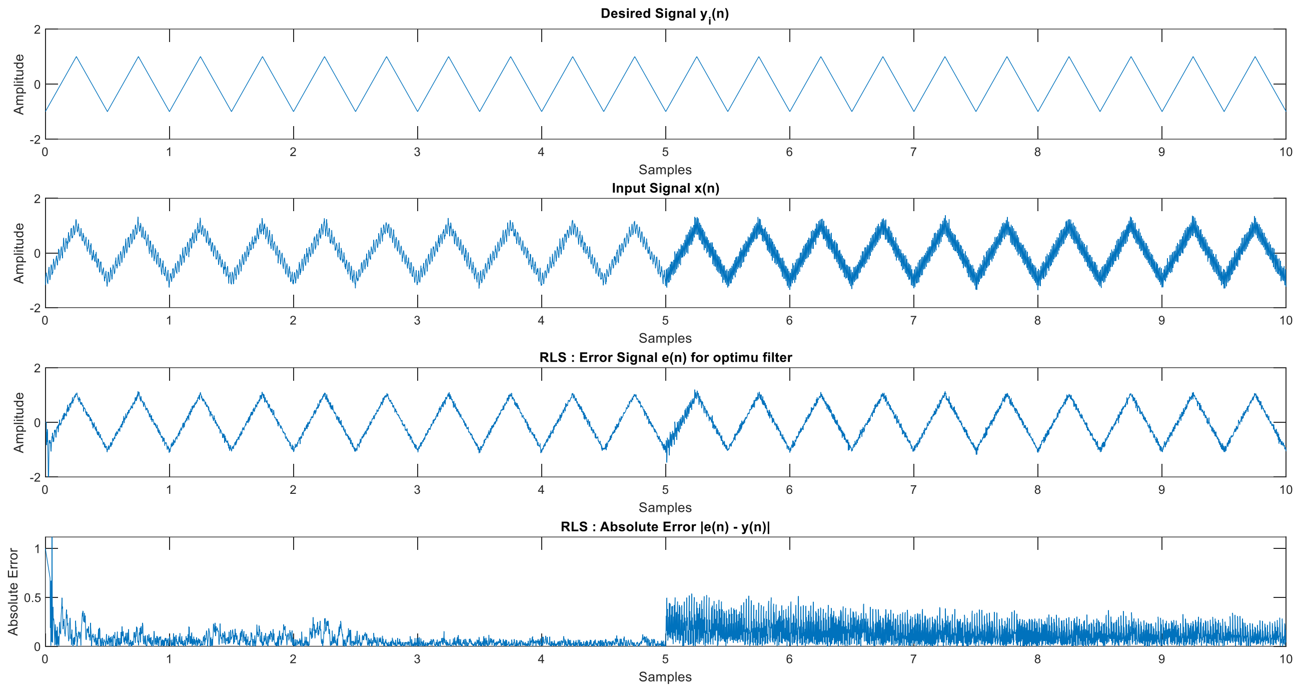


Here if we use the same order as we used in LMS filter the filtered signal is heavily distorted.

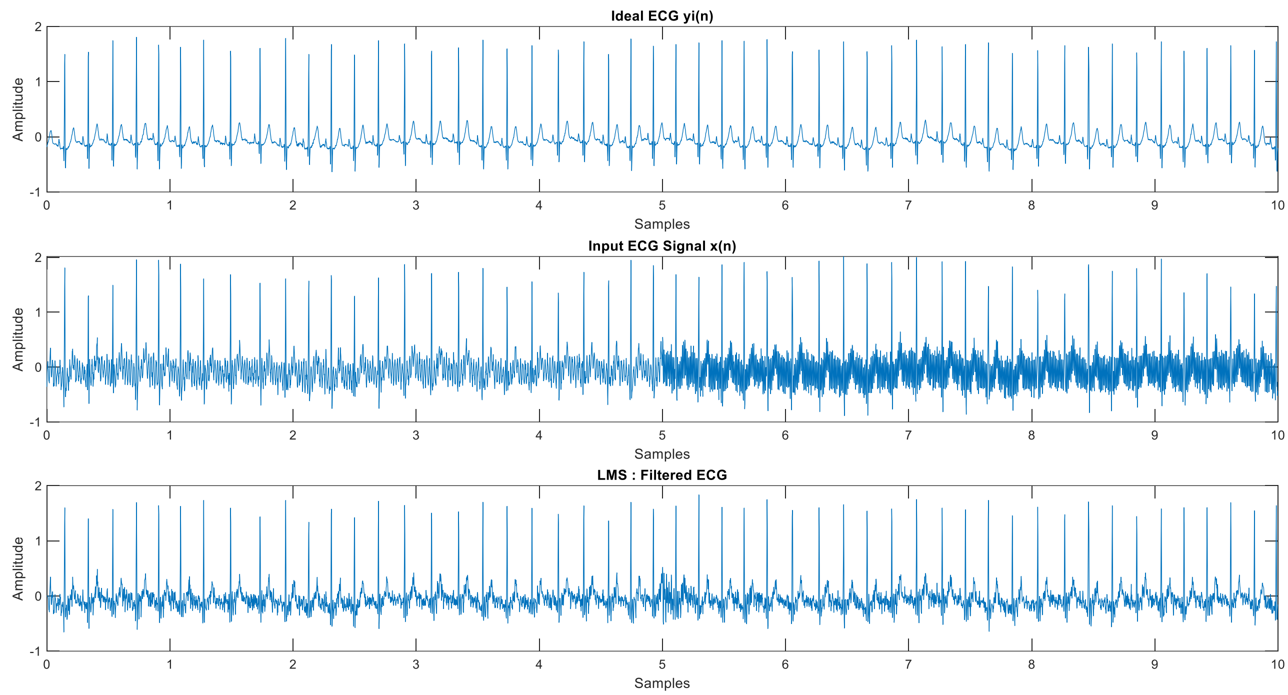
c)



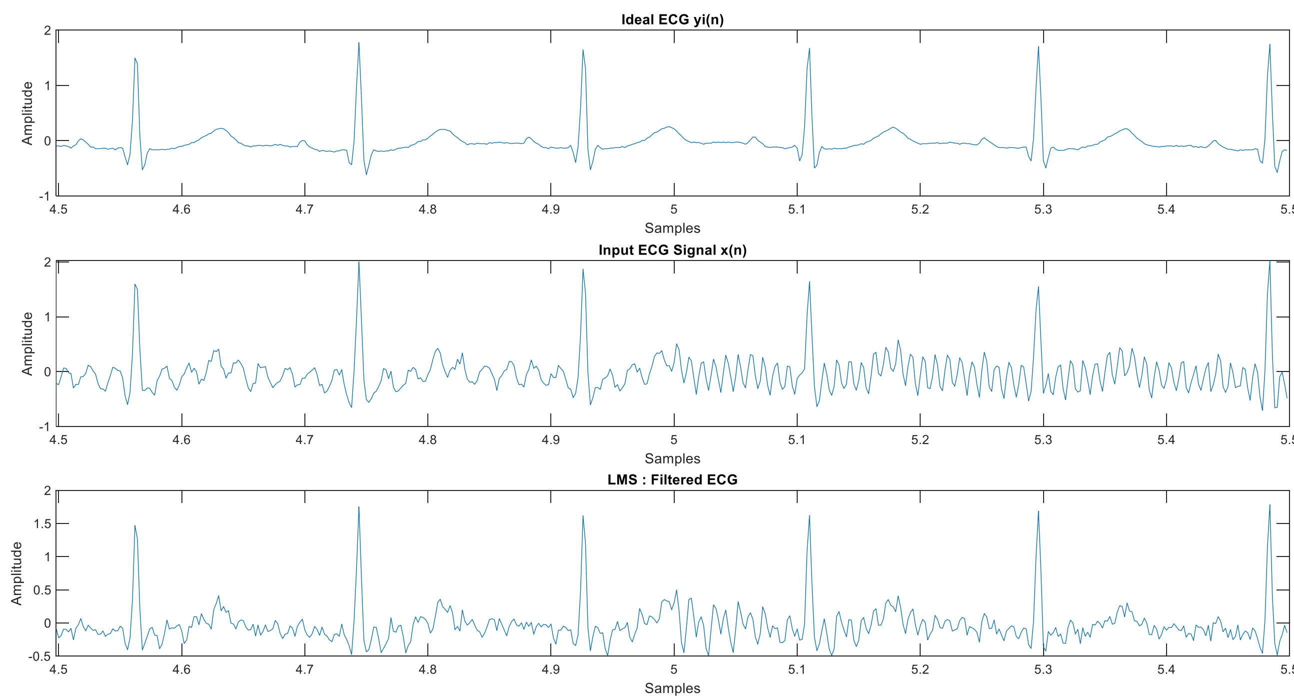
Optimum filter order = 5 and λ(forgetting factor) = 0.991 => MSE = 0.006674



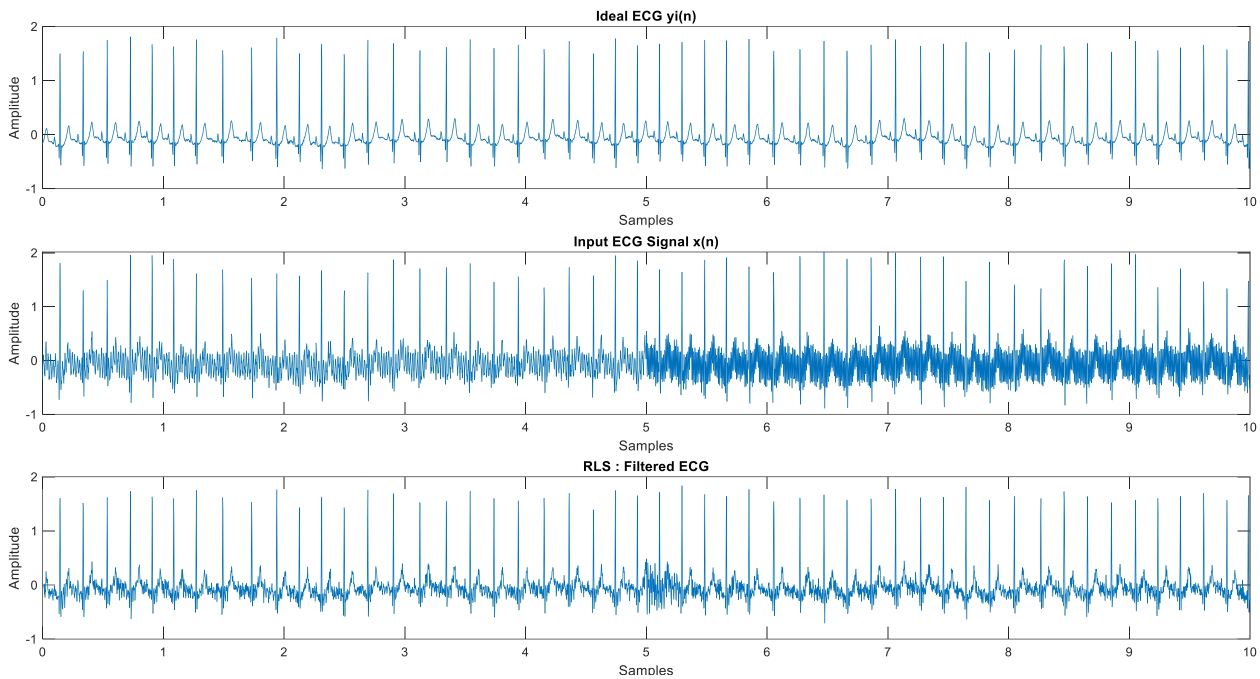
d) Optimum LMS filter for ECG filtering



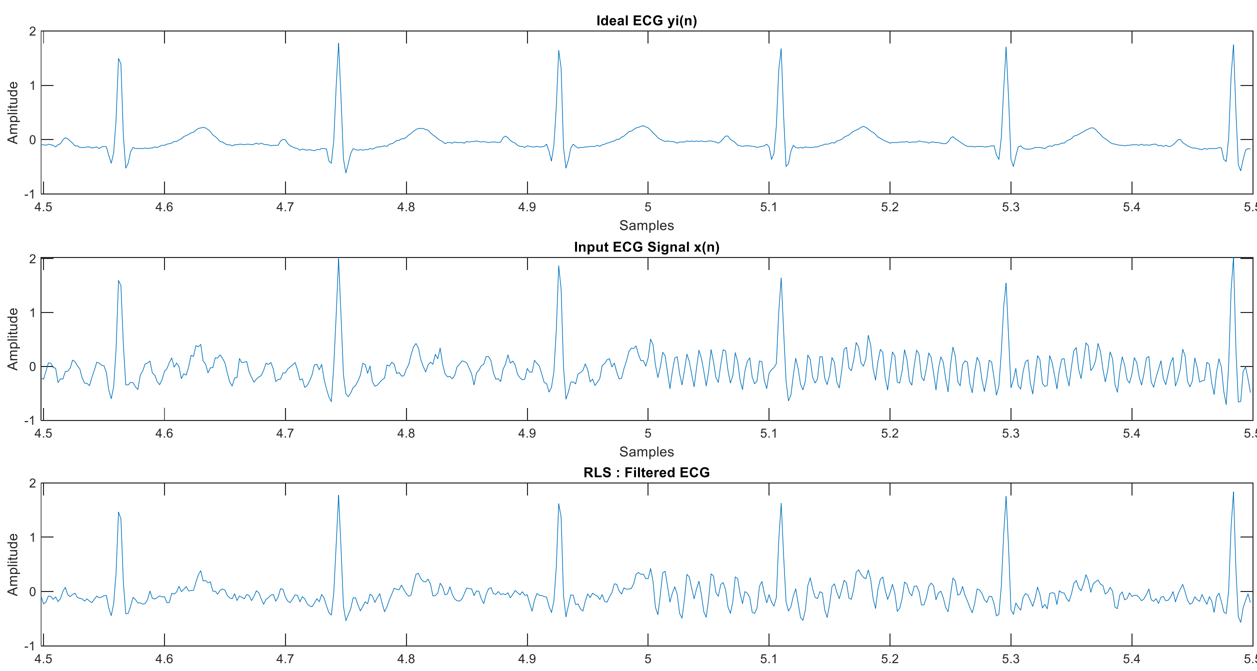
Zoomed Version



Optimum RLS filter for ECG filtering



Zoomed Version



Here we can see not as in wiener filter the LMS filter and RLS filter adapts to filter both 50Hz noise and 100Hz noise as well because here we don’t need to have non stationary noise or a signal.