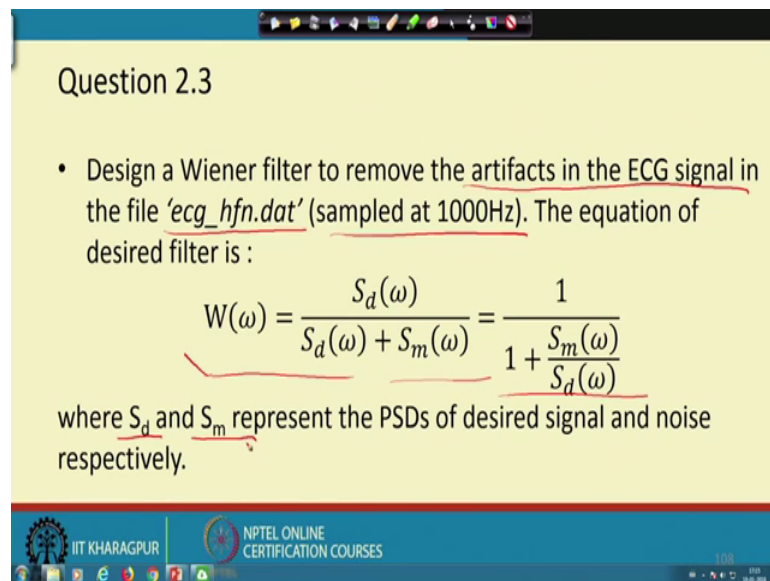


**Biomedical Signal Processing**  
**Prof. Sudipta Mukhopadhyay**  
**Department of Electrical and Electronics Communication Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 52**  
**Tutorial - II (Contd.)**

So, now we start the third problem of the tutorial two.

(Refer Slide Time: 00:26)



Question 2.3

- Design a Wiener filter to remove the artifacts in the ECG signal in the file 'ecg\_hfn.dat' (sampled at 1000Hz). The equation of desired filter is :

$$W(\omega) = \frac{S_d(\omega)}{S_d(\omega) + S_m(\omega)} = \frac{1}{1 + \frac{S_m(\omega)}{S_d(\omega)}}$$

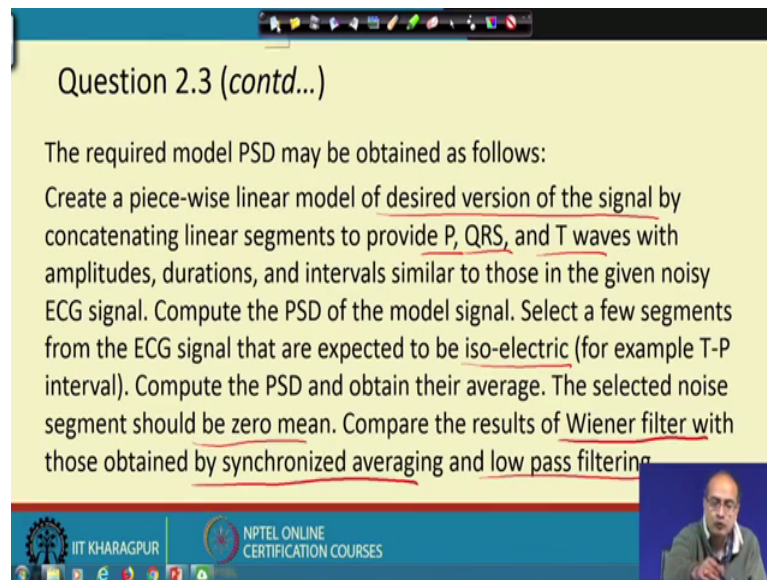
where  $S_d$  and  $S_m$  represent the PSDs of desired signal and noise respectively.

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, here we have to look for a Wiener filter design and here is the form of the Wiener filter is given that we have a signal we need to remove the artifacts in the ECG signal; ecg underscore hfn sampled at 1000 hertz the equation of the Wiener filter is given here, it is expressed in the form of in the frequency domain and it is expressed with the help of the PSD of the desired signal and the noisy noise signal ok.

So, first we need to be actually acquainted with them that what would be the desired spectrum as well as the noise spectrum; we need to look at that to design the Wiener filter ok. So, Wiener filter is the optimal filter. So, it is a very important filter. So, let us move forward to see that.

(Refer Slide Time: 01:32)



Question 2.3 (contd...)

The required model PSD may be obtained as follows:

Create a piece-wise linear model of desired version of the signal by concatenating linear segments to provide P, QRS, and T waves with amplitudes, durations, and intervals similar to those in the given noisy ECG signal. Compute the PSD of the model signal. Select a few segments from the ECG signal that are expected to be iso-electric (for example T-P interval). Compute the PSD and obtain their average. The selected noise segment should be zero mean. Compare the results of Wiener filter with those obtained by synchronized averaging and low pass filtering.

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

Now, we get some more description or some more instruction that how we can actually build that PSDs.

For that the summation is here that we can create a piecewise linear model of the desired version of the signal; why that is told that if we have the desired model of the signal in the time domain or the frequency domain, then I think that filtering is not required, if we already know that how the signal looks like without noise, then the problem is already solved. We would not take the pen to design the filter as we do not have that we are trying to approximate that with the help of piecewise linear model.

The idea is in that way we are able to remove or suppress the noise to a good extent whereas, we still keep the overall characteristics of the signal at hand and we concatenated the piecewise linear model of different segments P segment, QRS segment and T segment and with that; we can get actually the that the PSD of the approximate desired signal and in between there are actually that places where the isothermal line is there these are the activities are completely noise.

So, we can take those terms that is the space between specially between the T and the next P wave that part, we actually collect that and that gives us the idea about the noise ok. So, we will select a few segments of the ECG signal and isothermal lines that part we take the that T to P segment couple of the these and we get the noise PSD in that way by averaging these PSDs ok.

And we select the noise segment that is that which are which should be 0 mean and then once we have the PSD of the signal and the noise, then we are ready to create the Wiener filter and after that we need to compare the Wiener filter output with that of synchronous averaging and low pass filtering ok. So, that is the overall goal of this experiment.

(Refer Slide Time: 04:36)

Solution 2.3 Cont....

- Input ECG signal is available at:  
[http://people.ucalgary.ca/~ranga/enel563/SIGNAL\\_DATA\\_FILE/S/ecg\\_hfn.dat](http://people.ucalgary.ca/~ranga/enel563/SIGNAL_DATA_FILE/S/ecg_hfn.dat)
- Sample MATLAB code to display the input ECG is available at:  
[http://people.ucalgary.ca/~ranga/enel563/SIGNAL\\_DATA\\_FILE/S/ecg\\_hfn.m](http://people.ucalgary.ca/~ranga/enel563/SIGNAL_DATA_FILE/S/ecg_hfn.m)

Note: Keep the input signal and the MATLAB codes in the same directory.

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, we start from the beginning the first task is to collect the data, we get it from this link, we need to collect the that the MATLAB file to read that data and we need to keep the input data and the MATLAB code in the same directory that is the working directory of the MATLAB.

(Refer Slide Time: 05:07)

**Solution 2.3**

- Wiener filtering is the statistical approach to reduce noise in signal
- The Wiener filter minimizes the mean square error between the estimated random process and the desired process
- Wiener filter requires the assumption that the signal and noise processes are weak-sense stationary
- PSD of the desired signal and noise is required for Wiener filtering

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, once we are ready with that we proceed we get the Wiener filter, we need to first design that. So, the first thing that here the Wiener filter is a statistical approach, it is looking at the PSD of both the signal and the noise PSD and we are trying to actually get a match between these two where the signal strength is more compared to the noise, we would like to keep more input from there where the noise input is more, then we would like to suppress all such components. So, that we can avoid the noise and when both are in intermediate range, then we need to judiciously select that what should be the gain.

So, how the Wiener filter does it it minimises the mean square error between the estimated random process and the desired process. So, desired process have a particular spectrum. So, what we try to do that whatever the input signal is given we try to find out a filter which minimises, the error between the desired process and the input process and thereby we that whatever the output; why we get we like to reduce that and that that is the way Wiener filter work.

Wiener filter; it requires the assumption that about the signal and the noise that which should be weak sense stationary, it should be at least weak sense stationary; that means, the characteristics should be stable, it should not change with time. So, for the non stationary signal, Wiener filter is not a good choice.

Next thing that we need to have the PSD of the desired signal as well as for the noise to design the Wiener filter without that information that we are unable to design the Wiener

filter; so, the first task becomes now to find out the PSD of the desired signal and the noise. So, we have to go for that. So, for that that we proceed now.

(Refer Slide Time: 07:55)

Solution 2.3

- The input Signal with High frequency noise

```
%% Load and Display Signal
% load Input ECG Signal
ecg = load('ecg_hfn.dat');
fs = 1000; % sampling freq
L = length(ecg); %Signal length
t = [1:L]/fs;
figure;
plot(t,ecg);
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

We try to first look at that signal that how the signal looks like, we heard that this is corrupted with the high frequency noise. So, we would like to plot the signal for that first we load that data and we load the ECG data in the variable name called ECG and we initialise the variable fs with the sampling frequency 1000 L gives the number of samples in the vector ECG or the number of data points available in the signal ECG using the command length.

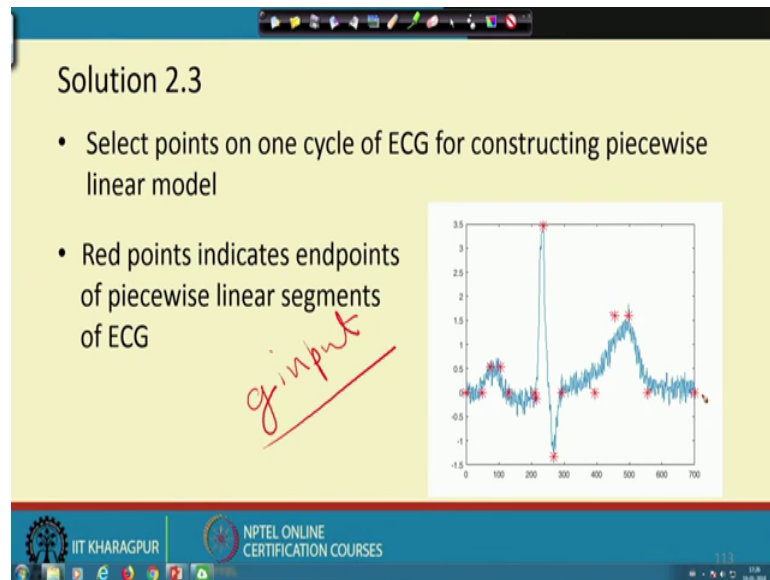
And then we create a time index instead of sample, we would like to plot it against the time to appreciate it better. So, we create a vector in the form of a ram increasing from 1 to L; that is the sample numbers and divided by fs each of this sample which will give us actually that the time in stunt of each of this sample. In fact, we can look at that; we are multiplying it with 1 by fs which is the, that the signal that the time interval between two samples.

Next is we create the pen for creating the new image using the command figure and we plot the signal ECG with respect to the x axis which is the time axis here t. So, here we are ready to see the signal and signal we get here, it is highly corrupted with the noise, if you look at the QRS complex because of high amplitude, it is not clear, but we look it from this point to this point that is t and the p wave we get a huge amount of noise that is

very clear ok.

So, we need to create the or design the Wiener filter to clean this actually signal.

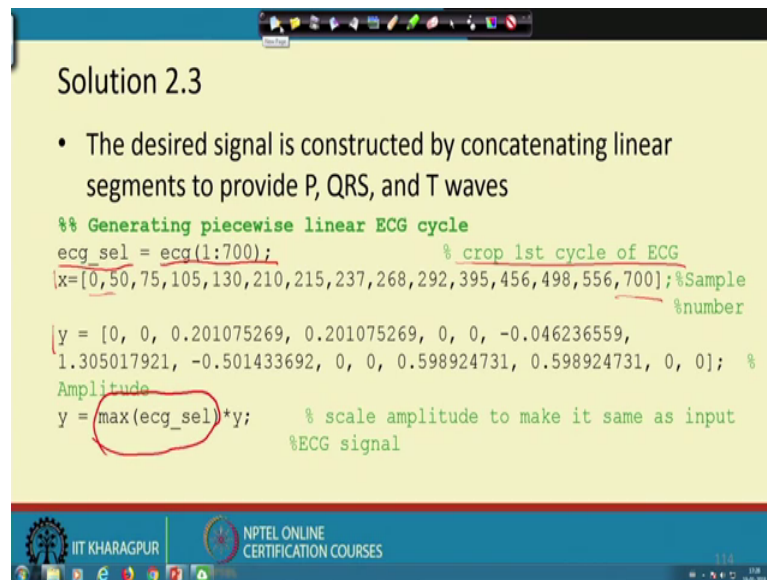
(Refer Slide Time: 10:30)



So, the first thing what we do? We take one cycle of it and first we have to create a desired actually signal that is by the linear approximation of these. So, we plot this signal several times you have seen that how to plot one cycle here the only actually addition, we can say that we have chosen only the first that 700 samples ok.

Once that is done we have done it by high estimation that we have the one cycle of the ECG next is we use a command `g input` after the plot ok, we plot the signal and we use the command `g input` which will help that if we click on the signal, then all these actually from the left side we start and click these points which can help us to build a linear model of the ECG signal ok. So, with that what we get we are able to capture actually the x and y coordinate of these the points where we have clicked ok.

(Refer Slide Time: 11:58)



**Solution 2.3**

- The desired signal is constructed by concatenating linear segments to provide P, QRS, and T waves

```
%% Generating piecewise linear ECG cycle
ecg_sel = ecg(1:700);           % crop 1st cycle of ECG
x=[0,50,75,105,130,210,215,237,268,292,395,456,498,556,700]; %Sample
                                %number
y = [0, 0, 0.201075269, 0.201075269, 0, 0, -0.046236559,
1.305017921, -0.501433692, 0, 0, 0.598924731, 0.598924731, 0, 0]; %
Amplitude
y = max(ecg_sel)*y;           % scale amplitude to make it same as input
                                %ECG signal
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, let us see that how that looks like the first thing that what we have done extra, we have shown that that we have taken that the part of the signal from 1 to 700; that is only 700 first 700 samples we have taken and we have assigned it to a variable called `ecg_underscore sel`.

Now, in these that we have the first cycle is cropped and that we have used for the display and we also get we because of use of the `g` input command that we have two vectors `x` and `y`; they are actually giving the coordinate of those points where we have actually clicked. So, we are getting the `x` in terms of that the sample instances that is varying from 0 to 700 and more or less in between 0 to 15, we can represent with a straight line ok, 50 to 75, again with a straight line, we can represent in that way, we are getting these that the abscissa that points.

Same way; we get the ordinates in the `y` axis; however, here one the difference is there that the magnitude; what is shown here that is normalised. So, we do not get actually the this is actually values to the scale. Now to convert this, the points; take it to the scale; what we need to do we need to scale it with the help of the maximum value of the that ECG signal within these segment. So, we are taking `max` into `y` ok, `max` of this segment it `y`. So, that gives us the amplitude and in that we are able to get the, that signal output properly.

(Refer Slide Time: 14:30)

```
Solution 2.3

%% Generating piecewise linear ECG cycle
linModel = [];
for i = 1:length(x)-1 % for number of piecewise segments
    if isequal(y(i+1), y(i)) % check for zero slope
        a = y(i)*ones(1,x(i+1) - x(i)); % for replicate previous values
    else
        a=y(i):(y(i+1)-y(i))/(x(i+1)-x(i)):y(i+1); % for non-zero slope
        a = a(1:end-1); % discarding last redundant point
    end
    linModel = [linModel,a];
end
t1 = (1:length(linModel))/fs; % time for plotting linear model
figure;
plot(t1, linModel); % plot piecewise linear ECG
```

Now, what we do we go next we have to create a linear model a linear approximated model of the signal. So, first we have created a actually empty array that two third bracket that creates an empty array and we have assigned the variable valuable lin model, in that way, the next point is that what we are doing we are going through all these point; that is correct collected that length of x, we are going from the beginning to the end of this last, but 1 point length minus 1.

And first check what we are doing if the two consecutive values  $i + 1$  and  $i$  at these two locations the ordinate value  $y$ ; that is if that is same; that means, at that point that that should be a horizontal actually line should be drawn ok. So, for that what we have done that we put the value  $a$  and we take the scale from the previous value  $y_i$  and we multiply it with ones  $i$  comma that range is  $x_{i+1}$  minus  $x_i$ .

Now, the reason that we use this command  $y$  that ones one comma this way is that that in between  $x_{i+1}$  to  $x_i$  in general there would be multiple samples ok; here the two points, we have clicked number of samples are there in between. So, it should be a vector. So, that many samples; we need to insert here to create a replica of the signal. So, we take it we with the help of ones we create that vector of ones and essentially that value what we get a that is also a vector of constant magnitude with magnitude  $y_i$  ok.

Next point is if these two are not equal that  $y_i$  and  $y_{i+1}$ , then we come to this else portion; what we do here we need to have a increased from  $y_i$  to  $y_{i+1}$  is a linear

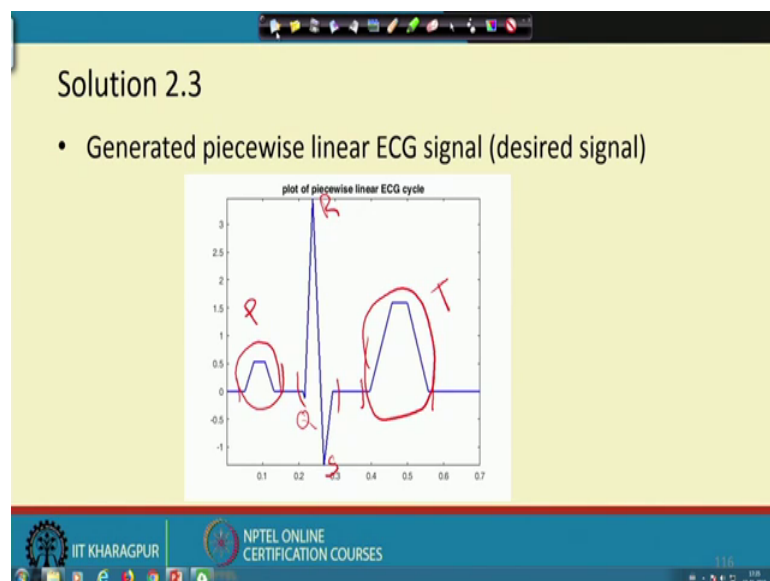


way and what should be the increment the increment is given by that it should be the total deviation  $y_{i+1} - y_i$  divided by the total duration in between  $ok$ . So, that is given by  $x_{i+1} - x_i$ .

So, it gives the increment per sample and with that rate we increase to create a ramp  $ok$ . So, with the help of that that we put that ramp in the this variable  $a$  and at the end, we put one more sample because we are going not till the end. So, that part we take care and at the end of it once it is check, once we append that with the that model linear model starting from that the actually a blank actually vector, it keeps on actually adding these as and keeps on building in that way the different segments and at the end of it it would be ready when we come out of this for loop.

So, at the end of the for loop once our linear model is ready the first thing is to look at what is the length of it, that we get using the command `length` of linear model and now, we create a ramp from one to length of this linear model and multiply it with  $1$  by  $fs$  to create the time index for plotting this linear model and for that we issue a command `figure` which creates the pen followed by the `plot` command against  $t$  one the time index we are plotting this linear model  $ok$ . So, that is the part we do and let us see how this signal looks like.

(Refer Slide Time: 19:35)



So, here we get the linearized model  $ok$ . So, we get the linearized model here the first part that these portion it is giving the P wave, here to here, we are getting the QRS

complex, this is QRS and these part is giving us the T wave and if you look at in between these portions ok. So, this part actually we are taking as 0 or isothermal line where actually we can expect the noise is rare in the input signal ok. So, these actually signal is very important in that way it not only gives us that a close approximation of the desired signal, but we also get where we expect the only the noise deposition free from the that signal component.

So, next what we do?

(Refer Slide Time: 20:54)

**Solution 2.3**

- The PSD of desired signal

```

%% PSD of desired signal
nfft=max(256,2nextpow2(length(linModel)));
[Pxx,F] = periodogram(linModel,[],nfft,fs);
figure;
plot(F,10*log10(Pxx));

```

Handwritten annotations: 10,  $N$  (circled), 2, (1024), 700

We create the PSD of the that desired signal as we have the signal here that we know the signal that lin model. Now to create the PSD the first point is to select that what how many point fft you would use and as we are using fft, it should be to the power of 2 to the power say to the power n in that form, it will come and if we take these value of n these number to the power n, it should be large enough or bigger than the that number of data points what we have. So, the minimum point what could be the value for actually this nfft, the variable which will tell that how many points fft, you would take it should be 256 and from the signal, we would like to get that what should be the power that what should be the length which should be a power of 2 as well as it should be actually bigger than the linear model.

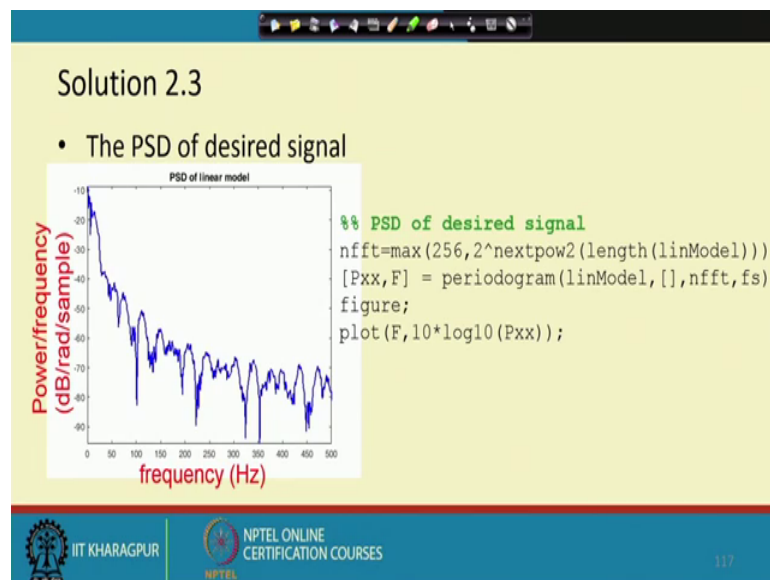
So, from that first we take the length of it we know the length of it is 700 in this case in this particular example. So, here being use this command two to the power next power 2

so, this one next power 2 will give us the power actually that the number which is bigger than that 700 the least actually that the number which is just bigger than 700 degrees 1024 ok.

So, we get that value here that 1024 and so, next power. So, we get the corresponding value that this part will give us the value 10 and 2 to the power 10, we will get 1024 here. So, 1024 and 256 the maximum value is 1024. So, we take that in the next line for creating the periodogram, the first variable of the periodogram is the input signal the second is for the window here we have chosen the default window that is the rectangular window.

Next we give that number of points for nfft that we have chosen here that should be 1024 and here the sampling frequency 1000 that is given. So, then we again create a pen with the new pen creating the that created using the command figure and then we plot it that against the frequency, we take in the dB scale that power ok. So, that is why we take log to the base 10 and multiply with 10.

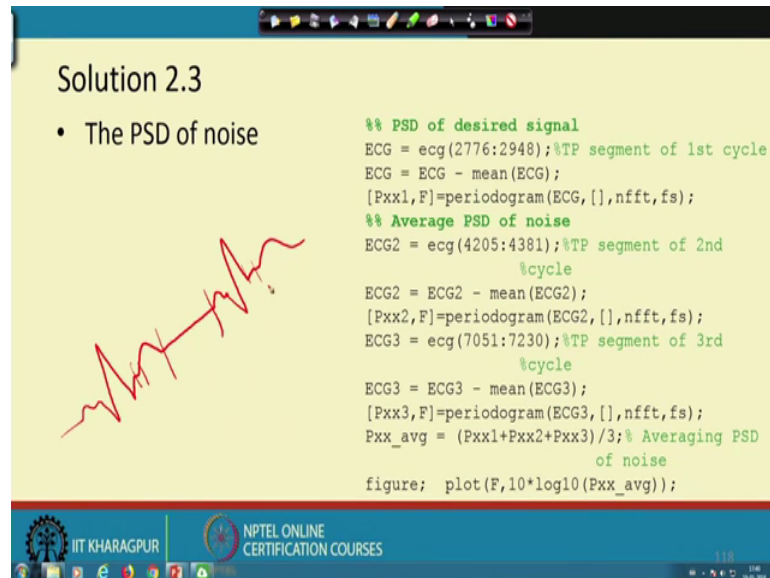
(Refer Slide Time: 24:37)



So, now let us see how the plot looks like. So, here the plot is given in terms of the frequency axis and other side it is in the dB part radian per sample, it is given what we get at the low frequency the signal strength is high and beyond the point, it is very quickly diminishing within after 50, if we look at that reduction is more than 40 dB ok. So, that is the nature of the signal.

And next we look at how we can create the, that noise energy the noise energy already we have told that how we can get that if we look at typical that ECG signal.

(Refer Slide Time: 25:11)



**Solution 2.3**

- The PSD of noise

```
%% PSD of desired signal
ECG = ecg(2776:2948); %TP segment of 1st cycle
ECG = ECG - mean(ECG);
[Pxx1,F]=periodogram(ECG,[],nfft,fs);
%% Average PSD of noise
ECG2 = ecg(4205:4381); %TP segment of 2nd
    %cycle
ECG2 = ECG2 - mean(ECG2);
[Pxx2,F]=periodogram(ECG2,[],nfft,fs);
ECG3 = ecg(7051:7230); %TP segment of 3rd
    %cycle
ECG3 = ECG3 - mean(ECG3);
[Pxx3,F]=periodogram(ECG3,[],nfft,fs);
Pxx_avg = (Pxx1+Pxx2+Pxx3)/3; % Averaging PSD
    % of noise
figure; plot(F,10*log10(Pxx_avg));
```

The slide also features a red hand-drawn ECG waveform on the left side. At the bottom, there are logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

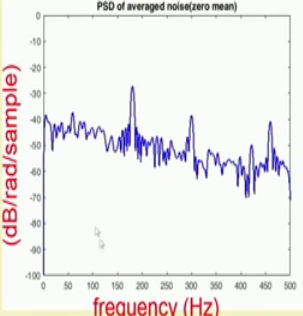
So, this is the segment where the isothermal line is the biggest here we can get it, then we can get it a little here. So, all these segments, we can get and we already know the periodogram is not a very good actually estimate and because that noise what is white that can give actually lot of fluctuations also in the PSD.

So, we are going for that the average PSD for the noise we are taking couple of segments for that that couple of segments we are taking from multiple cycles and at the end of it here what we have taken we have taken three such segments and we have calculated that  $P_x 1$ , then  $P_x 2$  and  $P_x 3$  and we are taking the average of these three and the averaged one we plot it here in the new pen.

(Refer Slide Time: 26:42)

Solution 2.3

- The PSD of noise



```
%% PSD of desired signal
ECG = ecg(2776:2948); %TP segment of 1st cycle
ECG = ECG - mean(ECG);
[Pxx1,F]=periodogram(ECG,[],nfft,fs);
%% Average PSD of noise
ECG2 = ecg(4205:4381); %TP segment of 2nd
    %cycle
ECG2 = ECG2 - mean(ECG2);
[Pxx2,F]=periodogram(ECG2,[],nfft,fs);
ECG3 = ecg(7051:7230); %TP segment of 3rd
    %cycle
ECG3 = ECG3 - mean(ECG3);
[Pxx3,F]=periodogram(ECG3,[],nfft,fs);
Pxx_avg = (Pxx1+Pxx2+Pxx3)/3; % Averaging PSD
    % of noise
figure; plot(F,10*log10(Pxx_avg));
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And what we get the frequency is more or less flat much flat compared to the that signal ok. So, that is the PSD of the noise.

(Refer Slide Time: 27:00)

Solution 2.3

- Wiener filtering

```
%% transfer function of Wiener filter
W = zeros(1,length(F));
for i = 1:length(F)
    W(i) = 1/(1+(Pxx_avg(i)/Pxx(i)));
end
%W in time domain
Y = ifftshift(abs(ifft(W,200)));
output = conv(ecg,Y);
```

$-\pi$  to  $\pi$   
0 to  $2\pi$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, we have to design the Wiener filter. So, for that what we have done that first we create the loop that the length of the that Wiener filter in the frequency domain, we take that f samples of 0s first and then we populate using that equation given there, but at the end of it we do not take the full length because we know that beyond certain point the signal length is 0 ok, it would be very low.

And if we try to take something beyond first 200 values experimentally, we found this number, then the signal strength is very small and as we are using this equation, then what will happen as  $x$  is becoming very small we get very large values and that part is not really useful.

So, we take the `ifft` of that and as the that filter should be real one, we take the absolute value because we are dealing with a real signal now in this way when we are using `fft` and `ifft` that we get the output from minus  $\pi$  to  $\pi$ . So, there is a phase shift we need to make it actually 0 to  $2\pi$  in that form we need to take. So, we need to use this command `ifftshift` and then we get the actually the filter output in the time domain.

Now, once we have the filter Wiener filter we convert with that impulse response with the input ECG signal and we get the output here and now let us see that how.

(Refer Slide Time: 29:16)

**Solution 2.3**

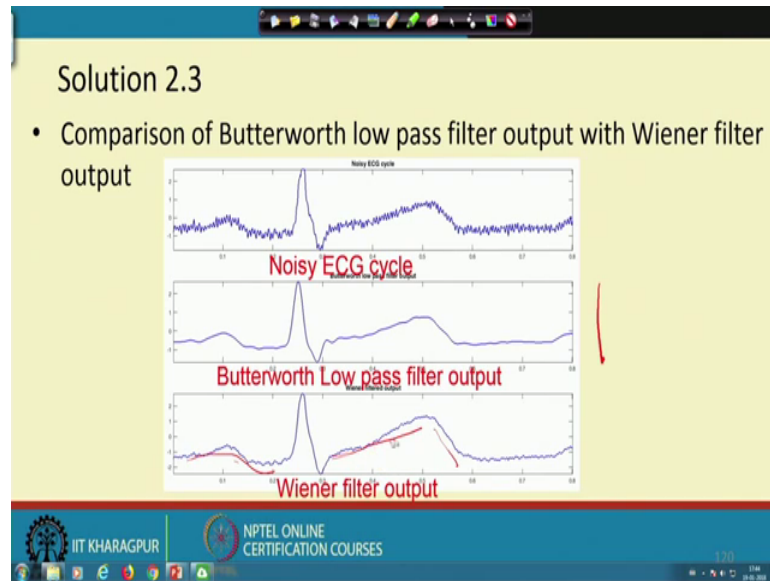
- Wiener filtering

```
%% transfer function of Wiener filter
W = zeros(1,length(F));
for i = 1:length(F)
    W(i) = 1/(1+(Pxx_avg(i)/Pxx(i)));
end
%W in time domain
Y = ifftshift(abs(ifft(W,200)));
output = conv(ecg,Y);
```

The output looks like here we get that output looks very similar to the input ECG signal it is that way pretty impressive that we got a good output.

Now, the only thing that is remaining that is we have to compare these signal.

(Refer Slide Time: 29:39)



With the two other cases; one is the that Butterworth filter given here and next we go for the synchronous average compared to the Butterworth filter, we see that though they are very similar that there is some undulations there which are actually shown in the p and the that t wave ok.

So, though this signal that output has more we can say detail there are some part of the noise is actually left behind in the output of the Wiener filter next when we compare with the synchronous averaging we see the output of the synchronous averaging means much more smooth out of all these cases synchronous averaging is even better than the Butterworth filter.

And then we get compare to that the Wiener filter has noise. So, that is the thing we observe.

(Refer Slide Time: 30:50)

**Solution 2.3**

**Observations**

- The wiener filter is able to suppress the noise but not able to remove completely
- The output of synchronized averaging is more smoother than wiener filter
- The output of low pass filter is smoother but slightly distorted compared to wiener filter

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, at the end, we summarise the whole thing we get first point that Wiener filter is able to suppress the noise, but not able to remove it completely a little remenance is there and that is because that there is some overlap in the spectrum of the signal and the noise we would not like to sacrifice that part of the signal. So, we are accepting a little part of the noise.

Next is what we get that output of the synchronous average that is the smoother than the Wiener filter. In fact, it is the smoothest one synchronous average one output of the low pass filter that using the Butterworth filter is smooth, but slightly distorted compared to the Wiener filter ok. So, that is the conclusion we make for this third experiment of the tutorial two.

Thank you.