Lecture 6

Table of Contents

Power Spectral Density (PSD)

Since we cannot have a mathematical model of random signals, so instead we have a statistical model.

The top half of figure 13.12 shows us how to compute the autocorrelation of the signal coming out, if we have the autocorrelation of the input signal.
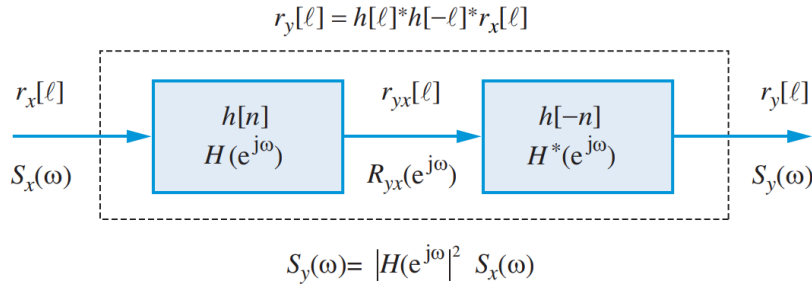
$$r_y[\ell] = h[\ell]^*h[-\ell]^*r_x[\ell]$$

$$S_y(\omega) = \left| H(e^{j\omega}) \right|^2 S_x(\omega)$$

Figure 13.12 The ACRS and the PSD of the output process can be thought of as "filtered" by an LTI system with impulse response $r_{hh}[n] = h[n] * h[-n]$. Therefore, although LTI systems process individual sequences, they have the same effect on all sequences with the same mean and ACRS.

The bottom half of the figure has to do with the power spectral density.

Definition of PSD

Power spectral density is used to characterise stationary random processes in the frequency domain. Power Spectral Density is defined as follows:

$$S_{xx}(\omega) = R_{xx}(e^{j\omega}) = \sum_{\ell=-\infty}^{\infty} r_{xx}(\ell)e^{-j\ell\omega}$$

Essentially, Power Spectral Density is the discrete-time Fourier Transform of the auto-correlation sequence $r_{xx}(\ell)$.

Notice that phases disappear when we compute the spectrum.

SIDE NOTE: If we have a signal with some oscillation, the oscilation is observed in the autocorrelation. Taking the Fourier Transform of the same autocorrelation, we will observe that there is energy at that particular frequency.

We can go the opposite direction. Given the PSD, we can compute the auto-correlation sequence by the inverse Fourier Transform as follows:

$$r_{xx}(\ell) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_{xx}(\omega)e^{j\omega\ell} \, d\omega$$

Quiz: Compare the PSD of two sinusoids

2

Consider the following two signals

$$x_1(k) = A_1 \cos(\omega k + \phi) + A_2 \cos(2\omega k + \phi)$$
$$x_2(k) = A_1 \cos(\omega k + \phi) + A_2 \cos(2\omega k + \phi + \Delta\theta)$$

Where ϕ is uniformly distributed on $[0 : 2\pi]$ and $\Delta\theta$ is a fixed number.

Do the two signals have the same power density spectrum?

 A: Yes
 B: No
 C: Not enough information given

The answer is A; the two signals have the power density spectrum.

In this quiz, we need to compare the PSD of the two signals; $x_1(k)$ and $x_2(k)$

The only difference between the two signals is that $x_2(k)$ is a fixed phase shift of $\Delta\theta$

In order to compare compute the PSD, we need to compute the autocorrelation of the signal.

Last week (see problem ADSI Problem 4.4.3) we computed the autocorrelation of a generic real sinusoid $x(n) = A\cos(\omega n + \phi)$, which was:

$$r_{xx}(\ell) = \frac{A^2}{2}\cos(\omega\ell)$$

From this result, we see that the phase does not matter. Therefore, the autocorrelation of $x_1(k)$ and $x_2(k)$ are the same.

This means that the corresponding PSDs are also the same.

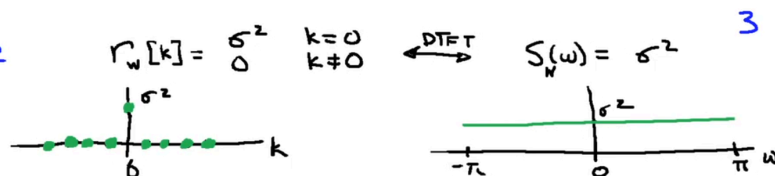**Lesson: when you compute the PSD, we only know the amount of energy at each given frequency. But we have no clue about the phase, which is essentially lost.**

Computing PSD of Example Signals

White Noise

The power spectral density is constant which means that we have the same power across all frequencies. This signal is called white noise because the power is equally distributed across the entire spectrum.

3

Random Sinusoid

2) Random Sinusoid

$s[n] = A\cos(\omega_o n + \phi)$

ϕ: uniform $[0, 2\pi]$

A: Gaussian, $E\{A\} = 0, E\{A^2\} = \sigma_A^2$

$r_s[k] = \frac{\sigma_A^2}{2}\cos(\omega_o k) \xleftrightarrow{\text{DTFT}} S_s(\omega) = \frac{\sigma_A^2}{4}\delta(\omega + \omega_o) + \frac{\sigma_A^2}{4}\delta(\omega - \omega_o)$
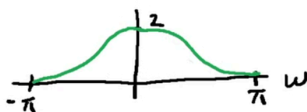

The power spectral density shows that the power is concentrated at $\pm\omega_0$. The area under these concentrations is $\frac{1}{4}\sigma A^2$.

Coloured Noise

3) Colored Noise

$r_c[k] = \begin{cases} 1 & k = 0 \\ \frac{1}{2} & k = \pm 1 \\ 0 & \text{otherwise} \end{cases} \xleftrightarrow{\text{DTFT}} S_c(\omega) = 1 + \cos(\omega)$

Average output of a filter

Suppose we want to know the average output of the filter i.e., $E[y^2(n)]$

The average output of a filter is actually just the value of the autocorrelation at lag 0. How? We can see this if look at the expression from a different perspective:

$$E[y^2(n)] = E[y(n)y(n-0)] = r_y(0)$$

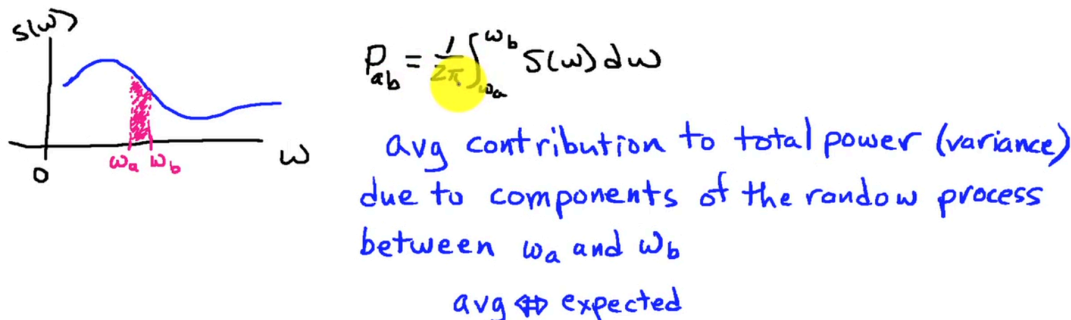Using the inverse definition of the PSD, we can compute $r_y(0)$

$$r_y(0) = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(\omega)e^{j\omega \cdot 0}\,d\omega$$

$$r_y(0) = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(\omega)\,d\omega$$

If we integrate the PSD from $-\pi$ to π, the resulting quantity is equal to the power in the time-domain. This means that **energy in the time-domain is equal to energy in the frequency domain**.

4

Another perspective:

Let P_{ab} be a normalised integration of the PSD from the interval ω_a to ω_b. The quantity P_{ab} is the expected (or average) contribution to the total power (or variance) that is due to the components of the random process between ω_a and ω_b. In other words, the area under the curve between ω_a and ω_b is the power that that portion of the spectrum is expected to contribute to the random process. It tells us how power is distrubuted in a frequency spectrum.

$$P_{ab} = \frac{1}{2\pi} \int_{\omega_a}^{\omega_b} S(\omega)\, d\omega$$

avg contribution to total power (variance) due to components of the random process between ω_a and ω_b

avg ⟷ expected

Example: Power Spectrum Analyser

Suppose we want to know the power spectrum of some input signal $x[n]$. We can pass that signal to an ideal bandpass filter with very narrow band and compute the power at the output. We define the bandpass filter to have the unit response in the vecinity of ω_c and the band $\Delta\omega$ is very small. Our bandpass filter rejects frequencies outside $\Delta\omega$ and pass any frequency inside this band.
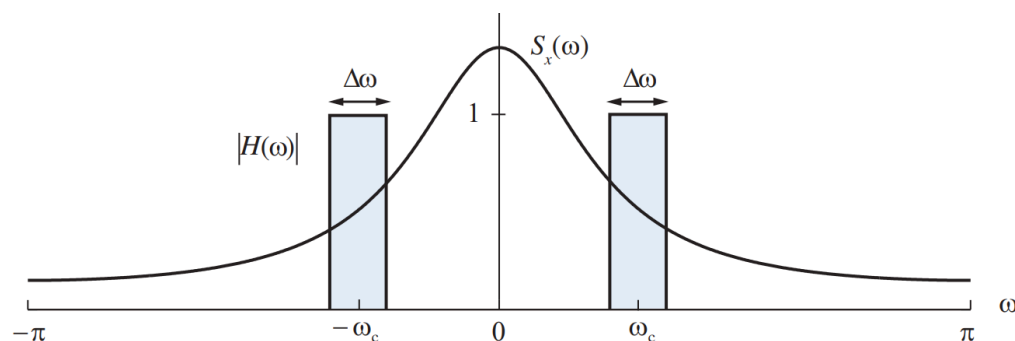

Figure 13.13 Physical interpretation of power spectrum density as power at the output of a narrowband LTI system.

Now, the question is how to do we compute the power spectrum of our ideal narrow bandpass filter when the signal is random?

The answer is to compute the the average output of the filter.

$$E[y^2(n)] = 2\frac{1}{2\pi}\int_{\omega_c-\Delta\omega/2}^{\omega_c+\Delta\omega/2} S_{xx}(\omega)\,d\omega \approx \frac{1}{\pi}S_{xx}(\omega_c)\Delta\omega$$

Since the spectrum is symmetic around zero, we get power contribution from both sides of zero. This is called a double-sided spectrum. Therefore, we have $2\frac{1}{2\pi}$ in the equation above.

5

Quiz: How do we know if a system in the real-world is linear and time-invariant?

In this class, we always make the assumptions that our systems are LTI system, but how do we actually know if a real-world system is LTI.

How do we measure if a system is linear and time-invariant?

We can measure if a system is linear by checking the two linearity properties:

1) $T(u + v) = T(u) + T(v)$

- Feed the system with the sum of two signals $x_1(n) + x_2(n)$ and observe the output $y(n)$.
- Feed the system with $x_1(n)$ to record the output $y_1(n)$.
- Feed the system with $x_2(n)$ to record the output $y_2(n)$.
- Now, the output $y(n)$ should be the same as $y_1(n) + y_2(n)$

2) $T(A\,v) = A\,T(v)$

- Feed the system with a signal with certain amplitude A e.g. $A = 42$. Record the output $y(n)$
- Feed the system with the same signal but where the amplitude is 1. Record the output $z(n)$
- The two outputs should have following relations: $y(n) = 42 \cdot z(n)$

We can measure if a system is time-invariant as follows:

- Feed the system with a known signal $x(n)$ and measure the output $y(n)$
- Feed the system with the same signal but shifted $x(n - k)$ the output should also be shifted $y(n - k)$

A more elegant approach is to compute the coherence function:

$$|\gamma_{yx}(\omega)|^2 = \frac{|S_{yx}(\omega)|^2}{S_{yy}(\omega) S_{xx}(\omega)} \qquad \text{with } 0 \leq |\gamma_{yx}(\omega)|^2 \leq 1$$

For a linear system, the coherence function should be 1 or very close to it:

$$|\gamma_{yx}(\omega)|^2 = 1$$

Before talking about the coherence function, we need to describe the cross power spectrum density.

Cross Power Spectrum Density

Cross power spectrum density $S_{yx}(\omega)$ is the Fourier Transform of the cross-correlation:

$$S_{yx}(\omega) = \sum_{\ell=-\infty}^{\infty} r_{yx}(\ell)e^{-j\omega\ell}$$

The cross power spectrum compares two signals and computes how much the two signals oscillate in phase. If we get a large value at a given frequency ω_0, then this means the two signals are oscilliting together at ω_0. If $S_{yx}(\omega_0)$ is small then the two signal are not in phase at ω_0.

Why do we care? The cross power spectrum density allows us to define a coherence function.

Coherence Function

The (magnitude squared) coherence function allows us to determine in an elegant way when a system is linear.

Formally, the (magnitude squared) coherence function is defined:

$$|\gamma_{yx}(\omega)|^2 = \frac{|S_{yx}(\omega)|^2}{S_{yy}(\omega)S_{xx}(\omega)}$$
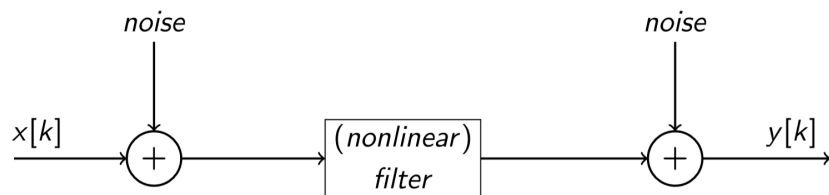
where $0 \leq |\gamma_{yx}(\omega)|^2 \leq 1$ i.e., the function yields a number between zero and one.

The coherence function of a perfect linear system is equal to 1.

However, in the real-world we don't always get a coherence of 1.

There are three primal factors that causes the coherence function to fall below 1.

Suppose we want to determine whether a system or a filter is linear:

- Factor 1: After we measure $x(n)$, some unaccounted noise creeps into the signal before going into the system.
- Factor 2: Some unaccounted noise creeps into the output signal before we measure it.
- Factor 3: The filter contains some nonlinear components.

Important lesson: The coherence function is a tool that we can use to measure whether any given system is actually linear. Since the coherence function measures how close a system is linear, we don't know whether a low coherence is due to noise or a non-linear filter.

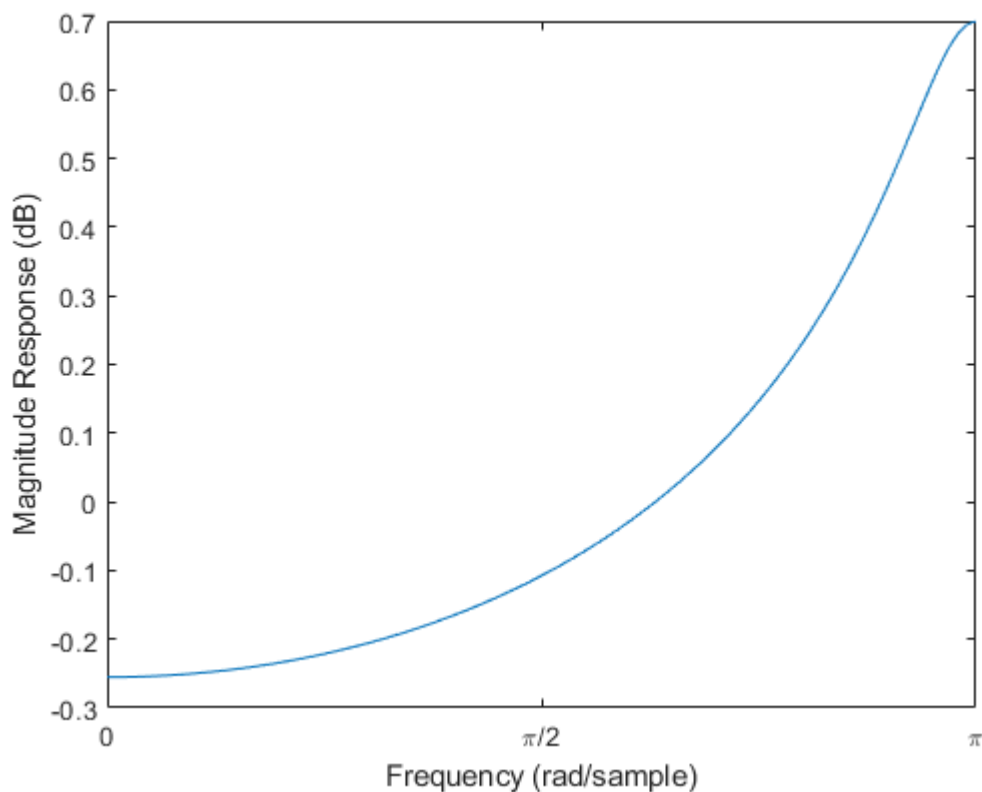## Coherence Estimates in MATLAB

Let us try it in MATLAB:

High-Pass IIR filter

```
N = 2000;
n = 1800:N;

% Create an input signal of random numbers
x = randn(N, 1);

% Define a high pass IIR filter that attenuates low
% frequencies and amplifies high frequencies
b = 1;
a = [1, 0.8];

y = filter(b, a, x);
plot(n, x(n), n, y(n));
legend('Input signal x[n]', 'Output signal y[n]')
```
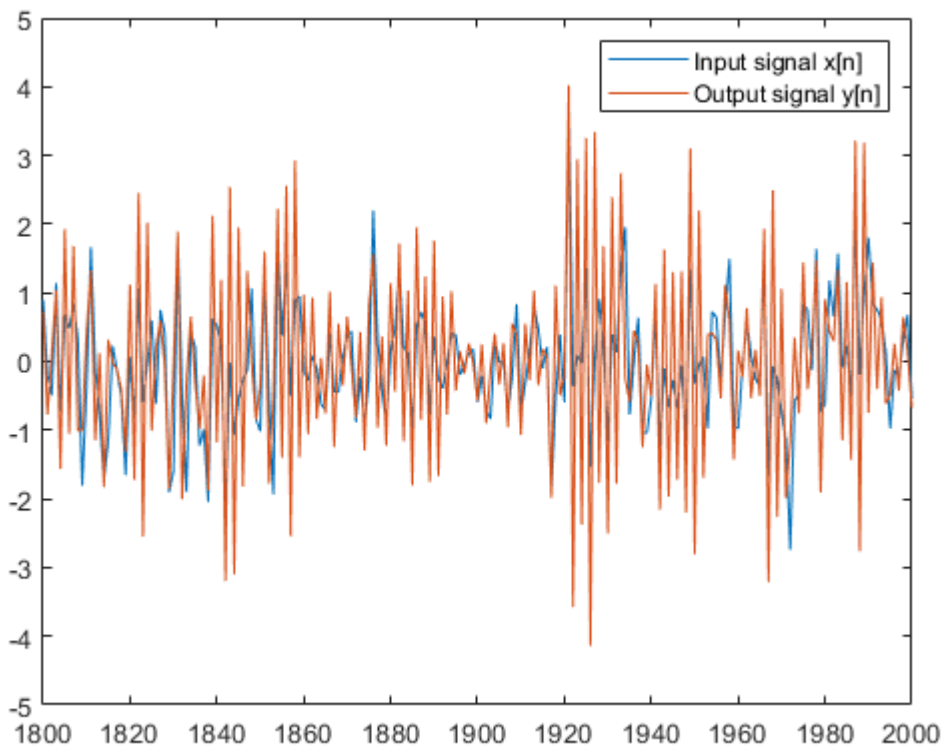
We can plot the magnitude response to see which frequencies the filter is attenuating and which frequency components it is amplifying:

```
[H, w] = freqz(b, a, 'whole');
plot(w, log10(abs(H)));
set(gca,'XTick',0:pi/2:2*pi)
set(gca,'XTickLabel',{'0','\pi/2','\pi','3\pi/2','2\pi'})
xlabel('Frequency (rad/sample)')
ylabel('Magnitude Response (dB)')
xlim([0, pi]);
```



```
% Compute and plot the coherence estimate
mscohere(y, x);
ylim([0, 1.05]);
```

Coherence Estimate via Welch
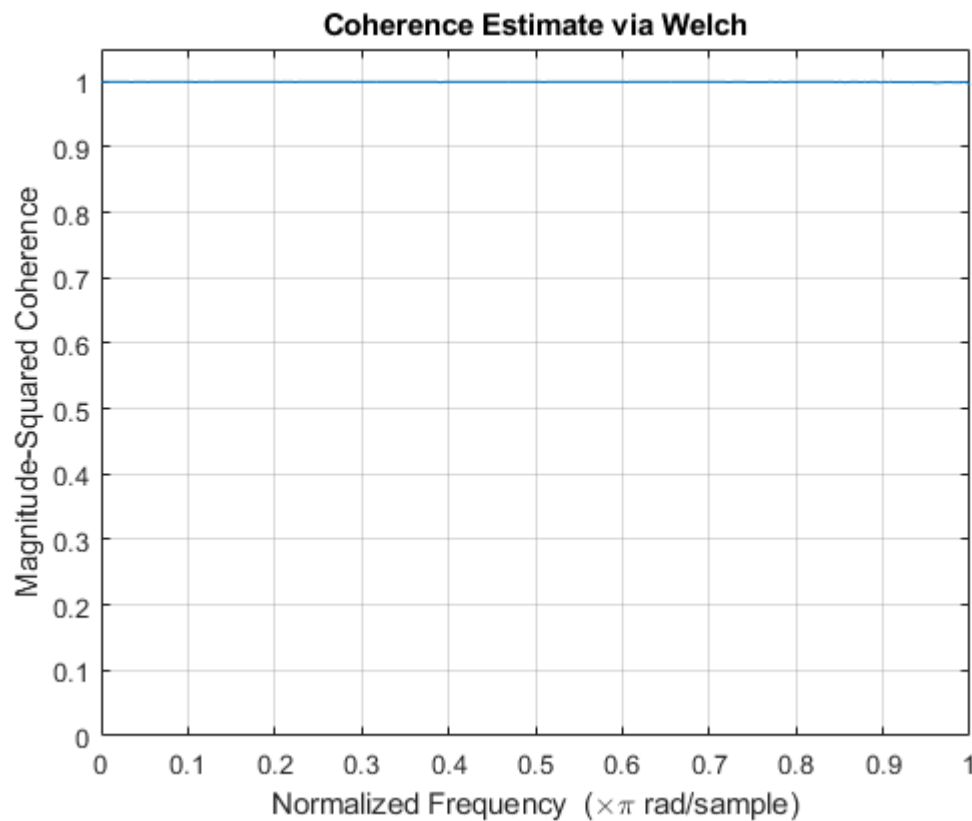
Notice that the estimate coherence function is 1. This is not a surprise because we are sending the input signal through a LTI system.

High-pass filter with noise at the output

```
% Define a high pass IIR filter that attenuates low
% frequencies and amplifies high frequencies
noise = 0.5;
y = filter(b, a, x);
y = y + noise*randn(N, 1);

% Compute and plot the coherence estimate
mscohere(y, x);
ylim([0, 1.05]);
```

Coherence Estimate via Welch
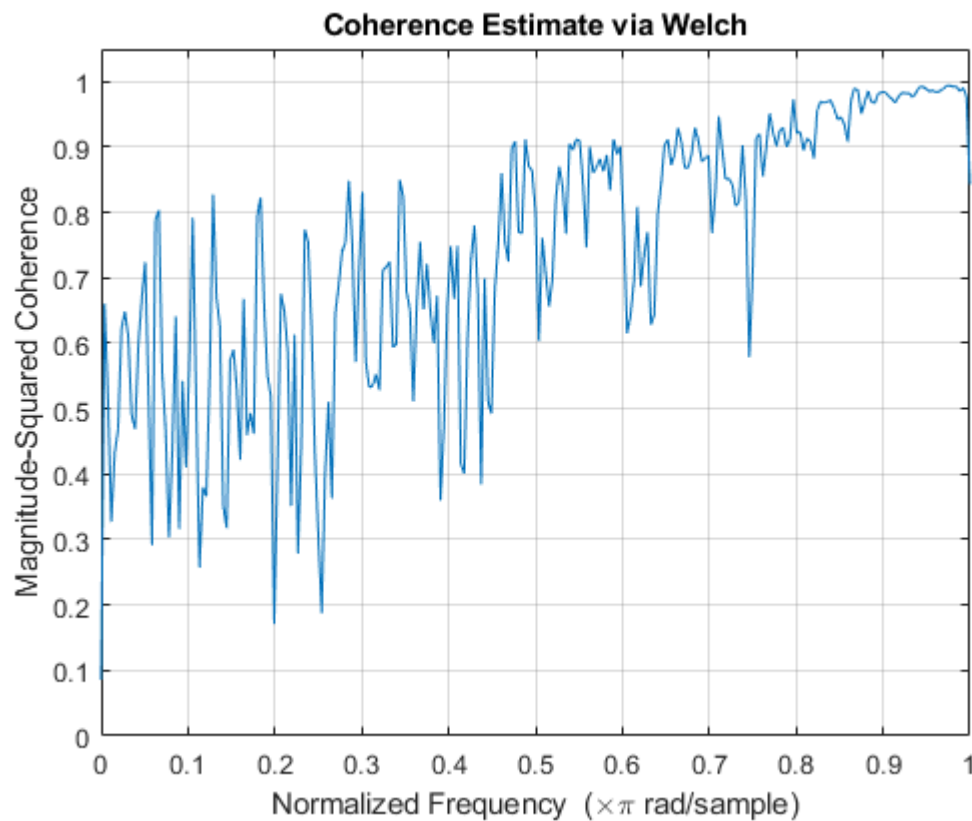
We observe that the coherence falls below 1 although we have a LTI filter because we have added a lot of noise to the output signal. Notice that there are low coherence at the low frequencies and high coherence at the high frequencies. This is because the high-pass filter attenuates low frequencies, and amplifies high frequencies.

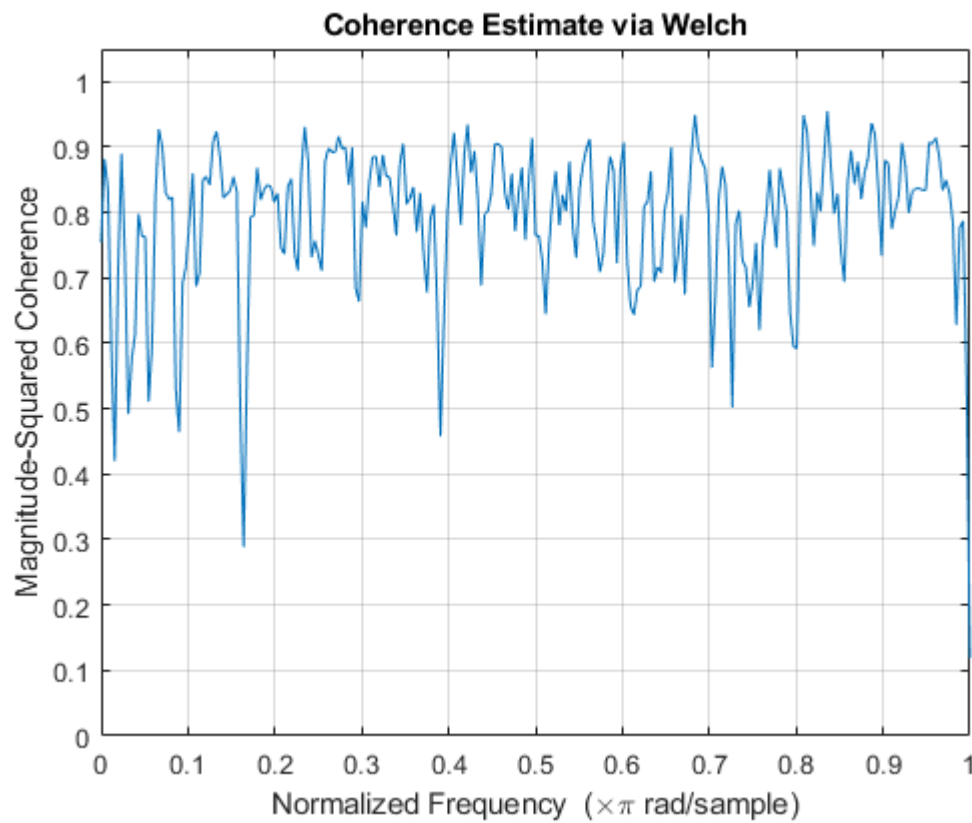If the noise is reduced, we get close to 1:

```
noise = 0.2;
y = filter(b, a, x);
y = y + noise*randn(N, 1);

% Compute and plot the coherence estimate
mscohere(y, x);
ylim([0, 1.05]);
```

Coherence Estimate via Welch
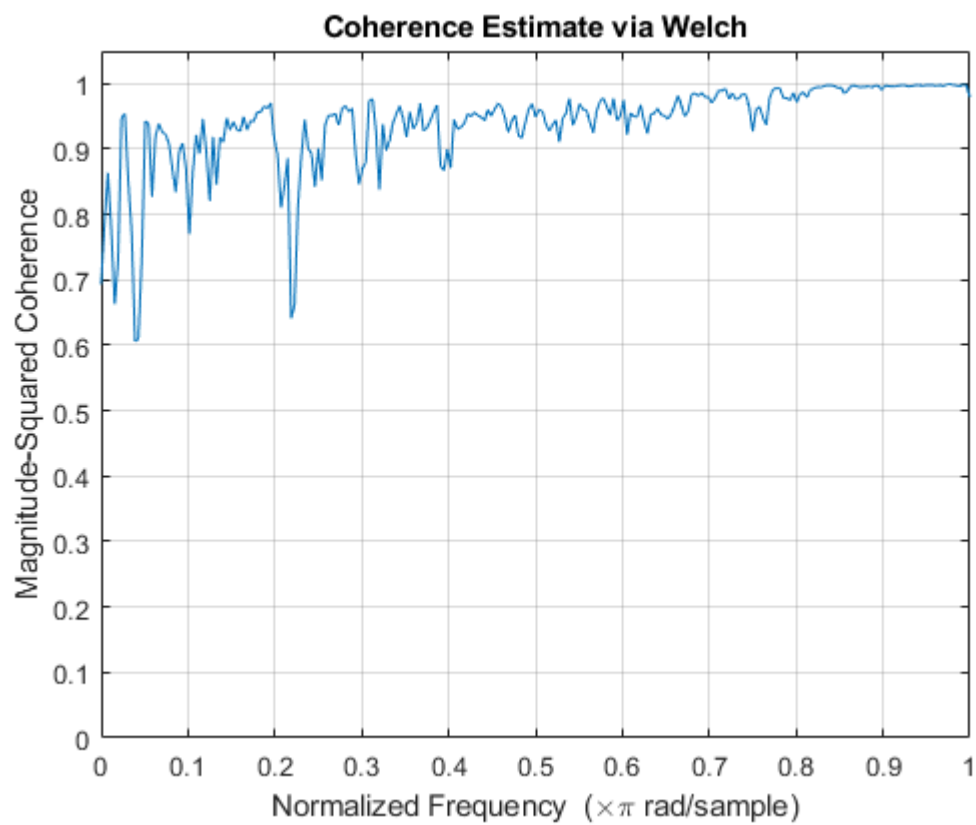
High-pass filter with noise at the input

```
noise = 0.5;
y = filter(b, a, x + noise*randn(N, 1));
mscohere(y, x);
ylim([0, 1.05]);
```
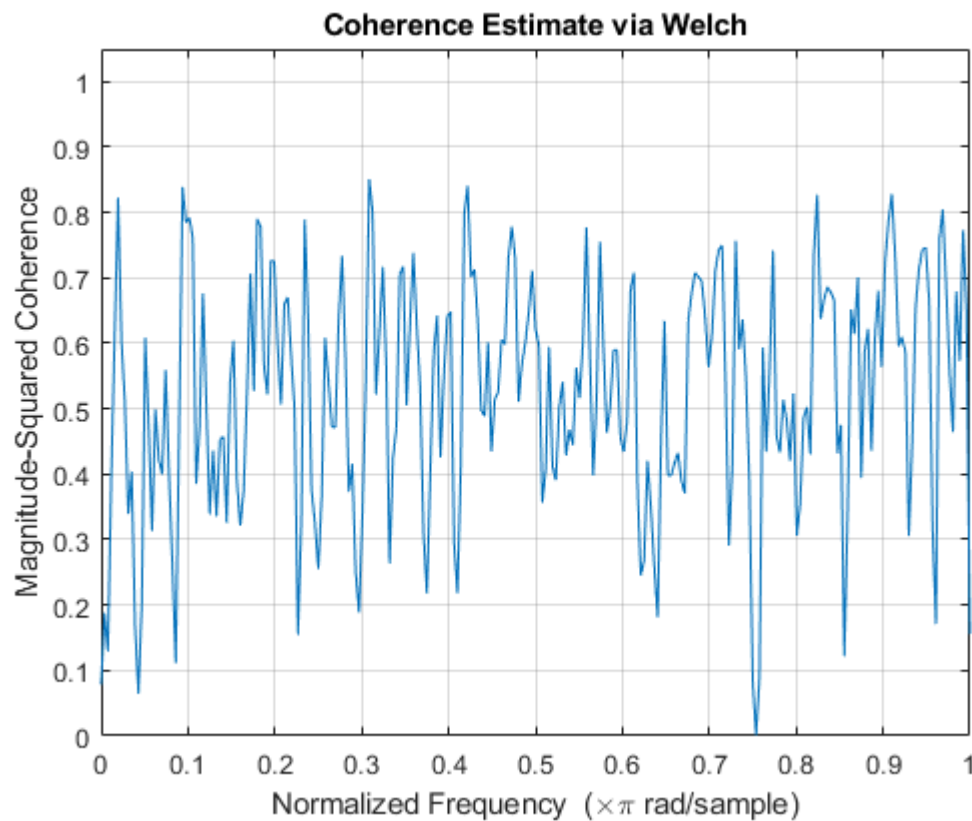
Coherence Estimate via Welch

The additional noise is also being filtered so we have equal amount.

Nonlinear filter

```matlab
% Take each sample and cube it
y = x.^3;
plot(n, x(n), n, y(n));
legend('Input signal x[n]', 'Output signal y[n]')
```

```
mscohere(y, x);
ylim([0, 1.05]);
```

Coherence Estimate via Welch
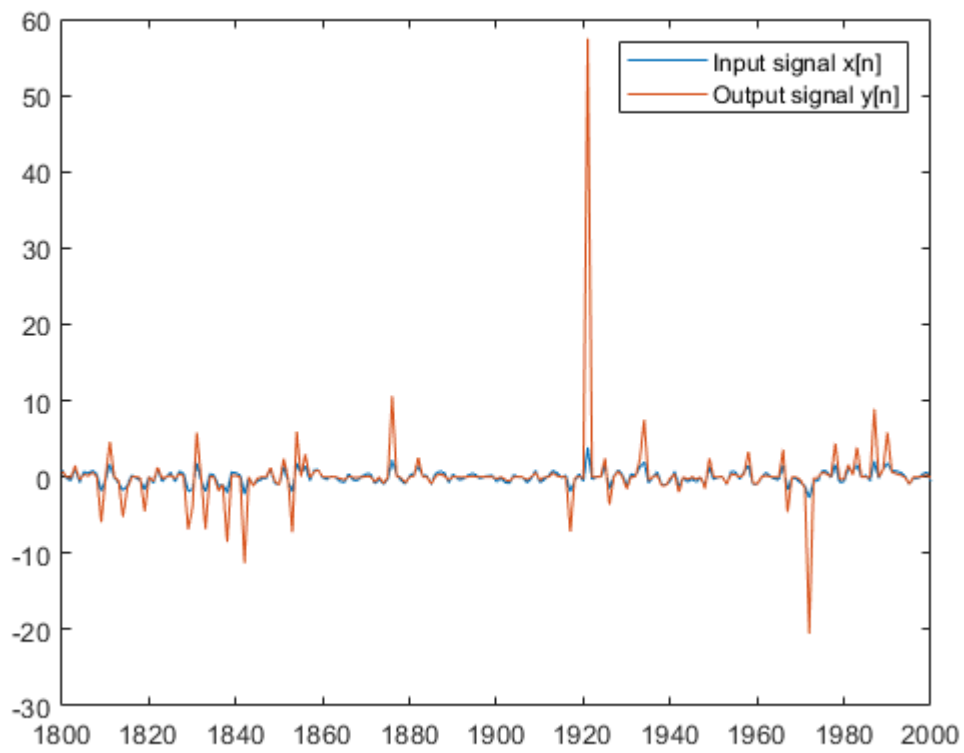
The coherence is all over the place.

Lesson: Since the coherence function measures how close a system is linear, we don't know whether a low coherence is due to noise or a non-linear filter.

In practice, the coherence function is close to 1. The difference $1 - \mathrm{mscohere}(y, x)$ tells us how well we can remove noise.

Nonlinear filter via clipping

```
% Non-linear filter via clipping
y = filter(b, a, x);
for k=1:N
    if y(k)>1
        y(k)=1;
    end
    if y(k)<-1
        y(k)=-1;
    end
end
plot(n, x(n), n, y(n));
legend('Input signal x[n]', 'Output signal y[n]')
```

```
mscohere(y, x);
ylim([0, 1.05]);
```

Coherence Estimate via Welch
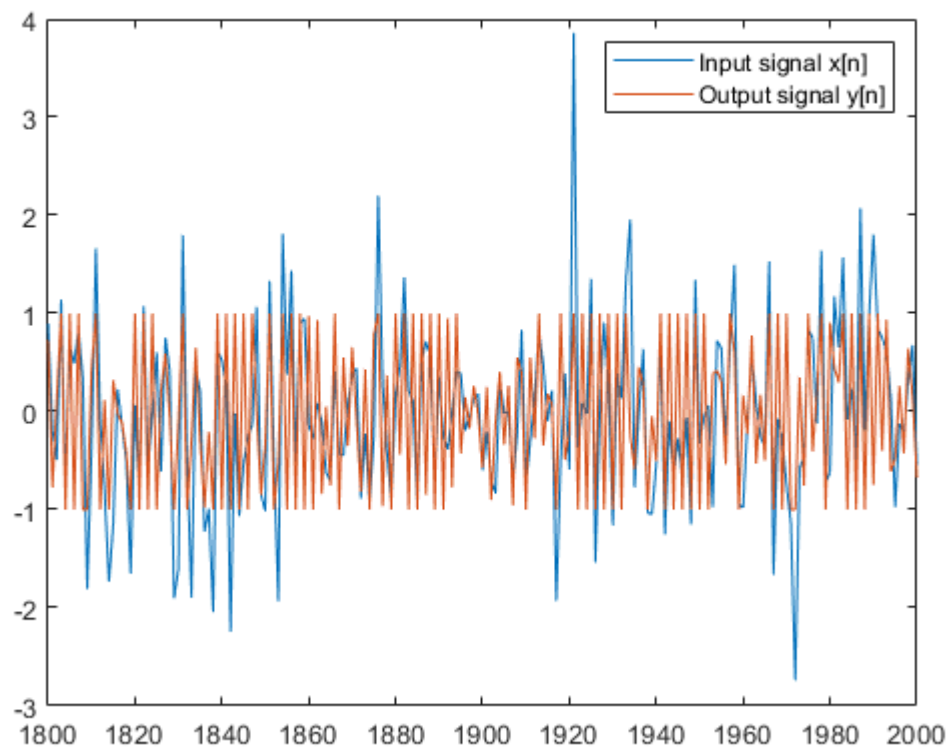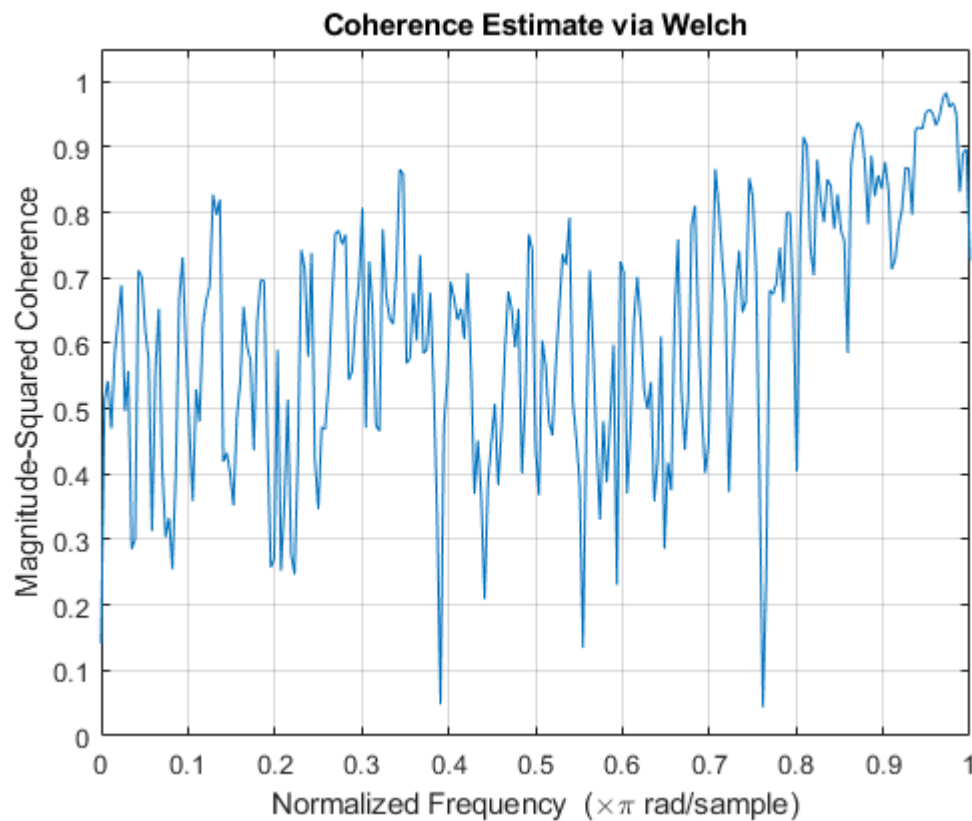
We see more coherence in the high frequencies than the low frequencies. Research why?
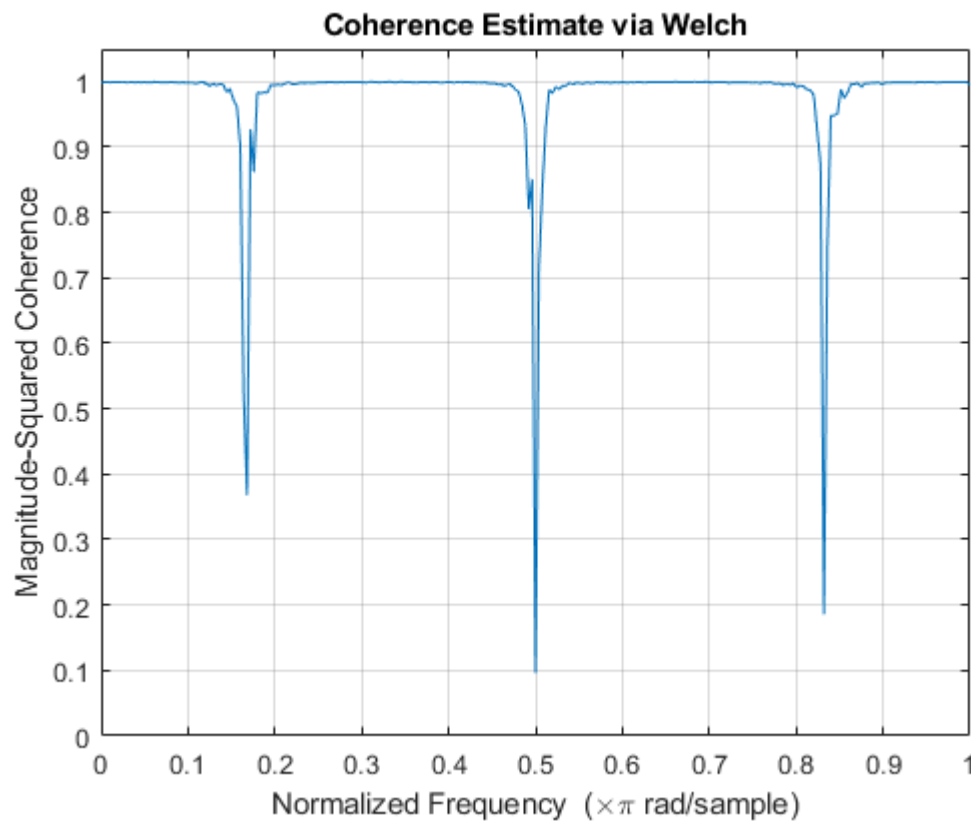
Comb Filter

Let us defined a comb filter which has the difference equation:

$$y[n] = b_0 x[n] + b_M x[n - M]$$

```
% Define a comb filter
b = [1 0 0 0 0 0 1];
a = 1;
y = filter(b, a, x);
plot(n, x(n), n, y(n));
legend('Input signal x[n]', 'Output signal y[n]')
```

```
% Compute and plot the coherence estimate
mscohere(y, x);
ylim([0, 1.05]);
```
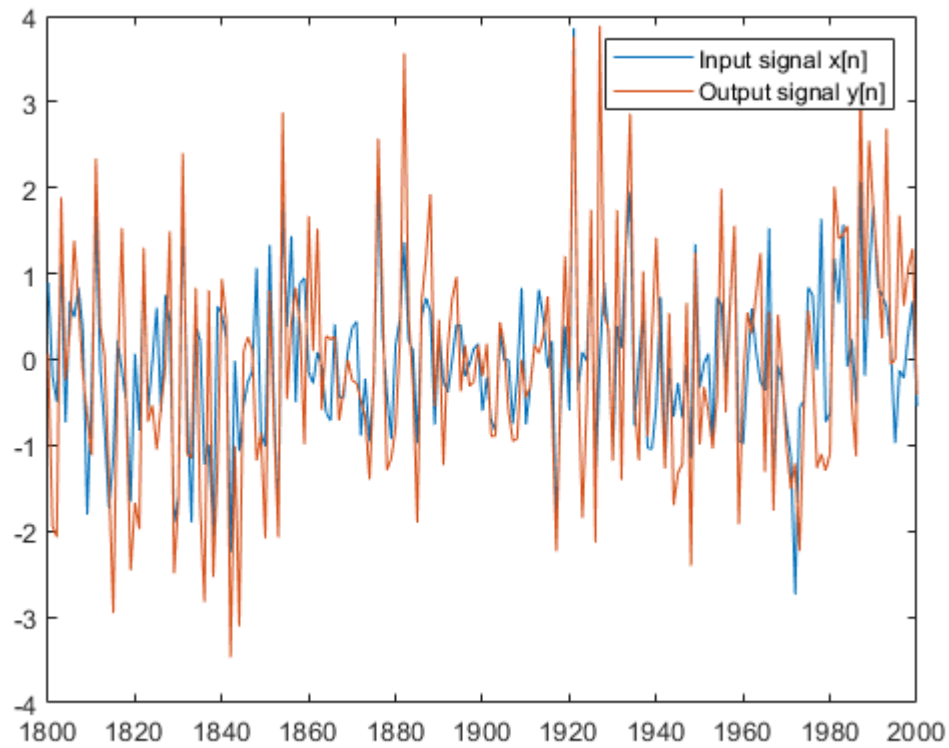
Coherence Estimate via Welch

Notice that the coherence function dips at certain frequencies. This is because the comb filter removes certain frequency components of the input signal. Thefore, there is no coherence between the input signal and the output signal at those specific frequencies since the output signal does not contain these filtered frequencies.

The filter has 6 zeros on the unit circle, which means that it is removing 6/2=3 frequency components:

```
zplane(b, a)
```
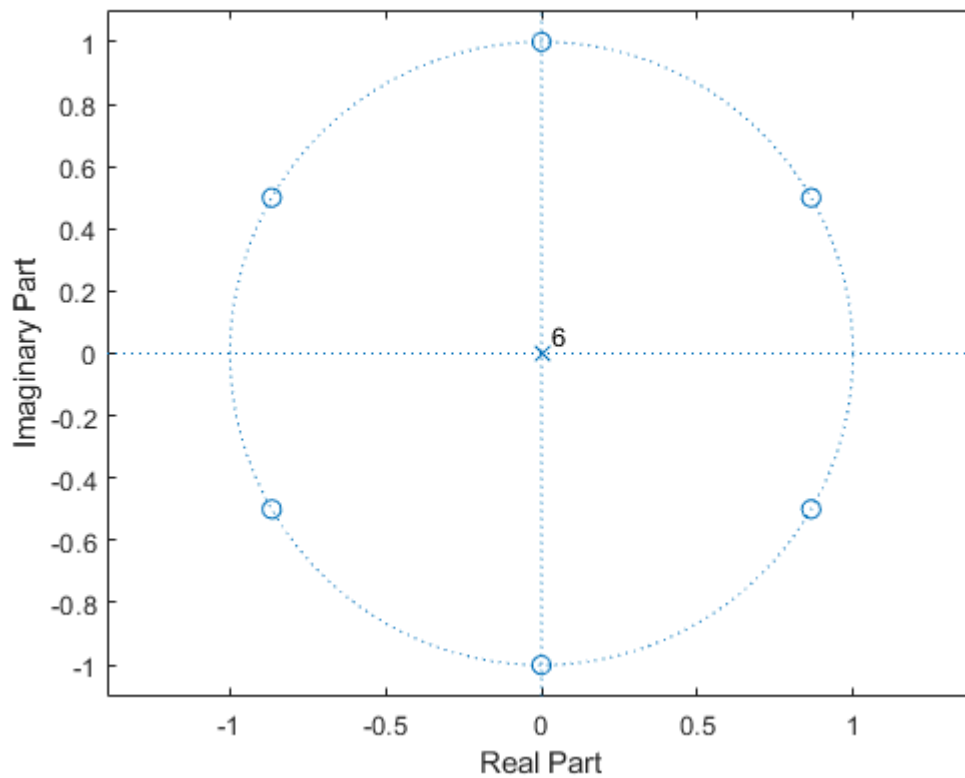
```
% Plot the magnitude response to see which frequency
% components are being attenuated
[H, w] = freqz(b, a, 'whole');
plot(w, log10(abs(H)));
set(gca,'XTick',0:pi/2:2*pi)
set(gca,'XTickLabel',{'0','\pi/2','\pi','3\pi/2','2\pi'})
xlabel('Frequency (rad/sample)')
ylabel('Magnitude Response (dB)')
xlim([0, pi]);
```
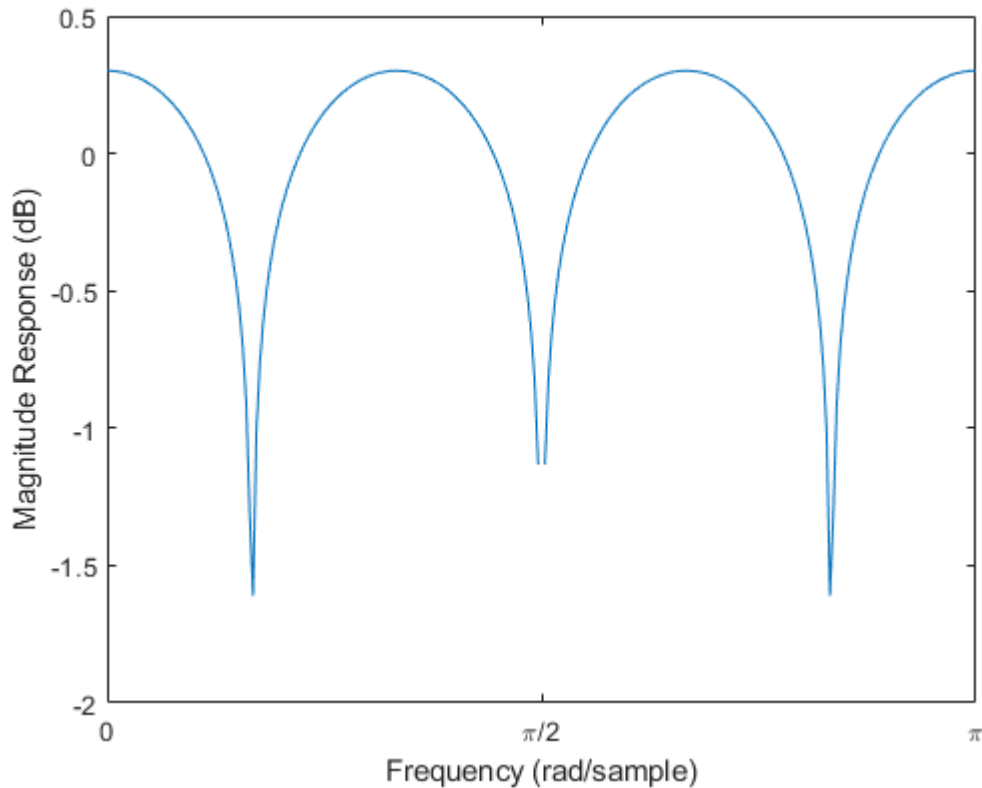
Properties of White Noise

White noise is important for random signal modelling.

INSIGHT: we can use white noise to find the impulse response of a system because the autocorrelation of white noise is the same the delta signal.

Suppose we have a random process that produces perfect random noise.

Let $w(n)$ be a random signal from this process.

The **expected value** of the signal is zero because there are no patterns in white noise:

$$E[w(n)] = 0$$

The **autocorrelation of white noise** generates one peak at $\ell = 0$ because that is the only time when there is any correlation of the signal. One peak at $\ell = 0$ can be modelled by delta signal:

$$r_{ww}(\ell) = E[w(n)w(n - \ell)] = \sigma_w^2 \delta(\ell)$$

21

where σ_w^2 is the variance of the signal which can be computed as follow:

$$\sigma_w^2 = E[w^2(n)] - E[w(n)]^2$$

$$\sigma_w^2 = E[w^2(n)] - 0 \qquad \text{(by definition } E[w(n)] = 0)$$

$$\sigma_w^2 = E[w^2(n)]$$

The **power spectral density of white noise** is:

$$S_w(\omega) = \sum_{\ell=-\infty}^{\infty} r_{ww}(\ell)e^{-j\omega\ell}$$
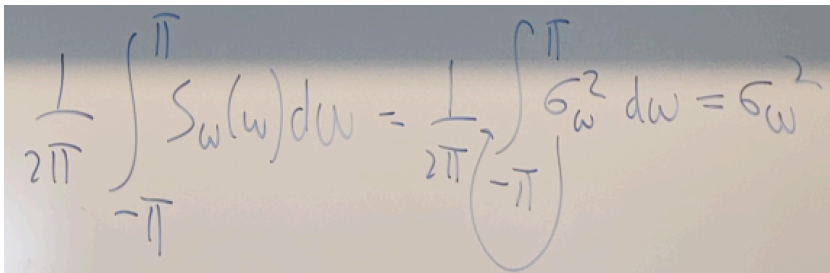
$$S_w(\omega) = \sum_{\ell=-\infty}^{\infty} \sigma_w^2\delta(\ell)e^{-j\omega\ell}$$

All terms where $\ell \neq 0$ become zero so we left with one term when $\ell = 0$

$$S_w(\omega) = \sigma_w^2\delta(0)e^{-j\omega\cdot 0}$$

$$S_w(\omega) = \sigma_w^2 \cdot 1 \cdot 1$$

$$S_w(\omega) = \sigma_w^2$$

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} S_w(\omega)d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sigma_w^2 \, d\omega = \sigma_w^2$$

Quiz: white noise cubed

We are used to white Gaussian noise

$$x[n] \sim WGN(0, \sigma_x^2) \quad \text{with} \quad r_{xx}(l) = \sigma_x^2 \delta(l)$$

What happens if we play with white Gaussian noise

```
x=randn(10000,1);
y=x.^3;
[ryy,lags]=xcorr(y,20,'biased');
stem(lags,ryy)
```

Is $y[n]$ also white noise?

A: Yes

B: No

So the question is this: suppose we have a white noise signal $x(n)$. If we take each element of this signal and cube it, would the resulting signal also be white noise?

The answer is yes! The output is also white noise because we are not introducing any correlation between different samples.

Let us prove it in MATLAB.

If we plot the autocorrelation of white noise, we should observe a signal similar to delta signal with amplitude of σ_x^2 because:

if $x(n) \sim WGN\left(0, \sigma_x^2\right)$ then $r_{xx}(\ell) = \sigma_x^2 \delta(\ell)$

First, we plot $r_{xx}(\ell)$

```
N = 10000;
n = [1:N];
x = wgn(N,1,0);

[r_xx, lags] = xcorr(x, 20, 'biased');
stem(lags, r_xx);
```
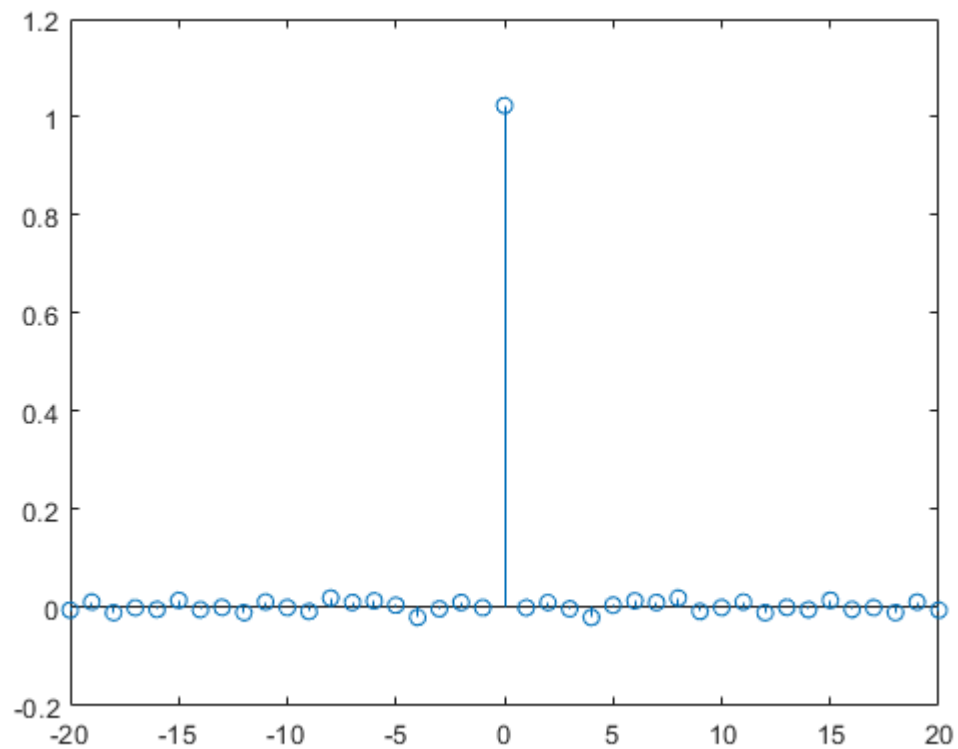
This looks like what have expected.

Now, let us plot the autocorrelation of $y(n) = x^3(n)$

```
y = x.^3;
[r_yy, lags] = xcorr(y, 20, 'biased');
stem(lags, r_yy);
```
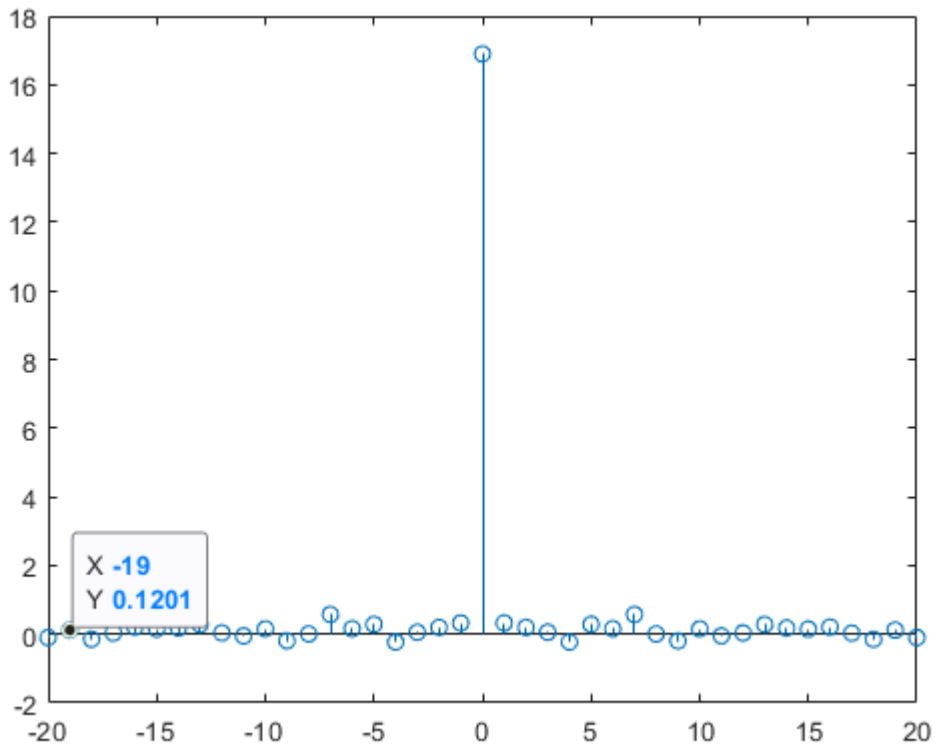
This proves that the answer to the quiz is Yes!

INSIGHT: Noise does not necessarily have to be Gaussian in order to be white. White noise can have any probability density function. Noise is said to be white when there is no correlation between samples.

```
hist(x, 50)
```
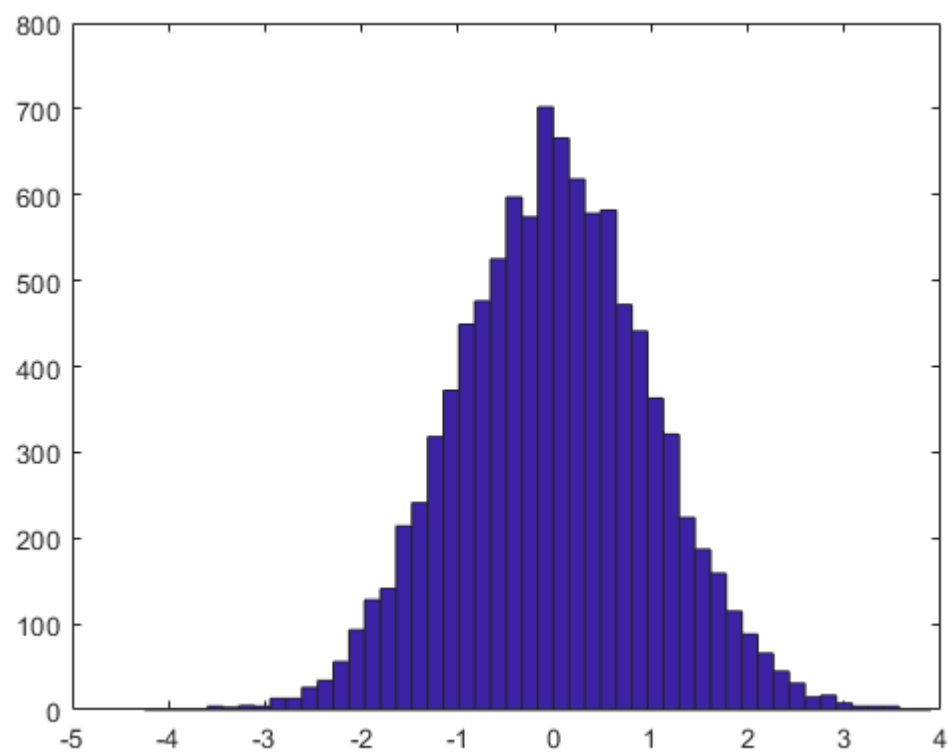
```
hist(y, 50)
```
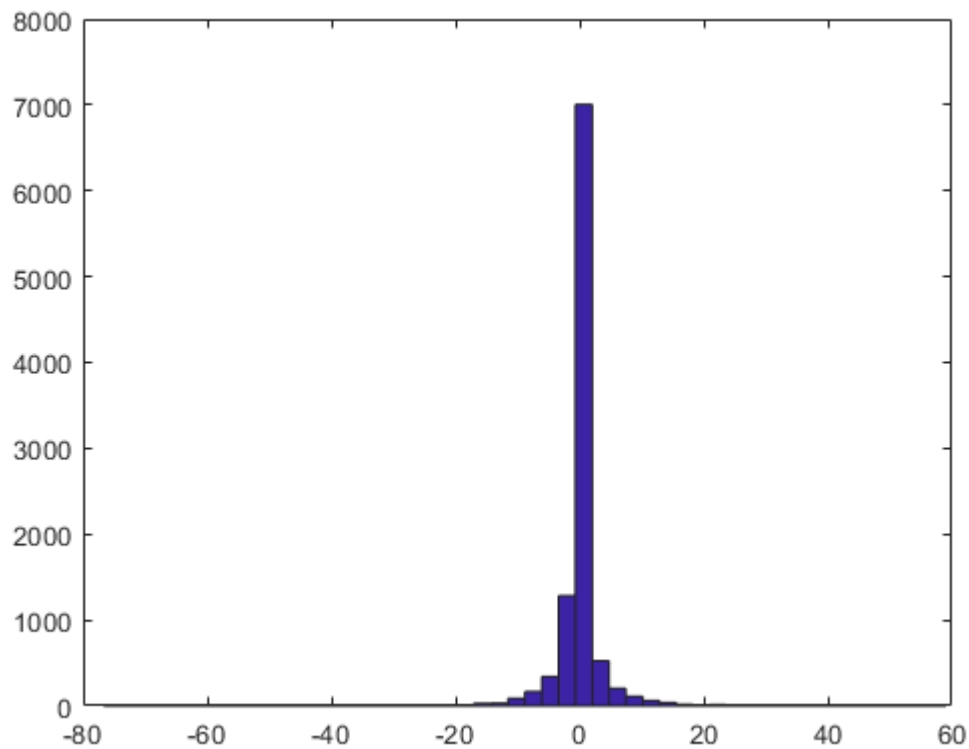
Quiz: decide whether a sequence is white noise

Assume we draw samples $\{x(n)\}$ from the following probability density function
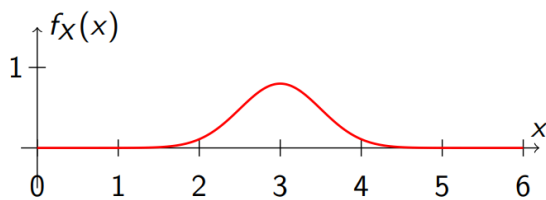
Is the sequence $\{x(n)\}$ white noise?

A: Yes

B: No

The samples are drawn from a Gaussian probability density function, but the mean value is non-zero, i.e. $x(n) = x_0 + x_1(n)$, where $E[x_1(n)] = 0$. Hence

$$
\begin{aligned}
r_{xx}(l) &= E[x(n)x(n-l)] \\
&= E[(x_0 + x_1(n))(x_0 + x_1(n-l))] \\
&= x_0^2 + \sigma_{x_1}^2 \delta(l)
\end{aligned}
$$

27

Signal Modelling

Signal modelling is about gaining some understanding of certain signals so we can model them. We do not need to have a full suite of samples but we want to understand from a statistical perspective.

White noise a building blocks that we can use.

Signal modelling is about taking some unknown signal $y(n)$ and coming up with an LTI system that generates $y(n)$ given some white noise input.

All the correlation that was between samples before are stored in the filter coefficients. This becomes a model of our signal. If we know the filter, we know the signal. The samples of the signal will not be the same but the signal will have the same statistical properties.

We cannot generate the exact same $y(n)$ using different white noise inputs. Instead we generate signals with the **same statistical properties**.

Regular processes

A regular process is just a minimum-phase LTI system given by the difference equation:

$$y[n] = \sum_{k=0}^{\infty} h[k]x[n-k] \qquad\qquad (13.129)$$

This is just a FIR filter with infinitely many coefficients.

The filter $h[n]$ is known as a **synthesis** or **colouring** filter.
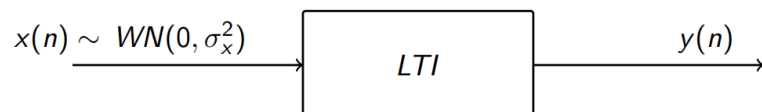
Since the LTI system is minimum-phase, it is causal, stable and therefore also invertible.

$$x[n] = \sum_{k=0}^{\infty} h_{\text{inv}}[k]y[n-k] \qquad\qquad (13.130)$$

This system is known as an **analysis** or **whitening** filter.

If feed this minimum-phase LTI system a white noise signal, we can compute the autocorrelation and PSD of the resulting process:

The autocorrelation of the input signal (white noise) is:

$$r_{xx}(\ell) = \sigma_x^2 \delta(\ell), \quad \text{where } \sigma_x^2 = E[x^2(n)] \text{ is the variance}$$

The power spectral density of the input signal is:

$$S_{xx}(\omega) = \sigma_x^2$$

The autocorrelation of the output signal is:

$$r_{yy}(\ell) = \sigma_x^2 \, r_{hh}(\ell)$$

The power spectral density of the output signal is power transfer function of the LTI system:

$$S_{yy}(\omega) = \sigma_x^2 \left| H(e^{j\omega}) \right|^2$$

Finding the synthesis filter $h[m]$ from the autocorrelation or PSD is known as **spectral factorization**. This is essentially what signal modelling is about; taking some unknown signal $y[n]$ and coming up with a minimum-phase LTI system that generates $y(n)$ given some white noise input.

$$x(n) \sim WN(0, \sigma_x^2) \longrightarrow \boxed{\quad LTI \quad} \xrightarrow{\; y(n) \;}$$
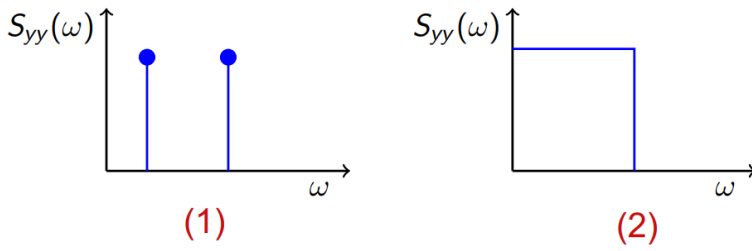
Paley-Wiener condition

A process is regular if its PSD satisfies the Paley–Wiener condition:

$$\int_{-\pi}^{\pi} |\ln S_{yy}(\omega)| d\omega < \infty. \qquad (13.131)$$

There are two kind of signals that cannot be modelled by an LTI system driven by white noise:

1. signal with line spectra e.g. sinusoids $\cos(\omega)$
2. bandwidth limited spectra
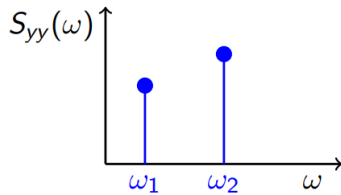
(1)   (2)

We cannot model these types of signals because it does not fulfill the Paley-Wiener condition:

$$\int_{-\pi}^{\pi} |\ln S_{yy}(\omega)| d\omega < \infty$$

Modelling signals with line spectra

Signals with line spectra are easy to model.

Suppose we want to model following signal:

We can just use a **harmonic process** to model the signal:

$$x[n] = \sum_{k=1}^{p} A_k \cos(\omega_k n + \phi_k)$$

where

- $\omega_k \neq 0$
- p is a constant denoting the number of frequency components
- $A_1, \cdots, A_p$ are constants denoting amplitudes
- $\omega_1, \cdots, \omega_p$ are constants denoting frequencies
- $\phi_1, \cdots, \phi_p$ are pairwise independent random variables uniformly distributed in the interval $(0, 2\pi)$

This model has $3p$ parameters A_k, ω_k and ϕ_k that we need to estimate. We can find these parameters using the least-squares.

The harmonic process is wide-sense stationary with mean value zero:

$$E[x(n)] = 0$$

The autocorrelation of the harmonic process is:

$$r_{xx}(\ell) = \frac{1}{2}\sum_{k=1}^{p} A_k^2 \cos(\omega_k \ell)$$

Take the Fourier Transform, we get the power spectral density:

$$S_{xx}(\omega) = \sum_{k=1}^{p} 2\pi \frac{A_k^2}{4}\left[\delta(\omega - \omega_k) + \delta(\omega + \omega_k)\right]$$

Modelling signals with bandwidth limited spectra

If we want to compute a filter with the brick-wall shape (2), we need an infinitely long non-causal filter. However, we said that we only wanted minimum-phase filters i.e. causal and stable filter. Therefore, this is not going to work.

Wold decomposition

The Wold decomposition says: *any stationary discrete-time process can be expressed as the sum of two **uncorrelated** processes, one regular process and one harmonic process.*

Now, let us go back to the regular process and introduce the ARMA(p, q) model.

ARMA(p, q) model

An ARMA(p,q) process is defined by the difference equation:

$$y[n] = \underbrace{-\sum_{k=1}^{p} a_k y[n-k]}_{\substack{\text{feedback} \\ \text{autoregressive}}} + \underbrace{\sum_{k=0}^{q} b_k x[n-k]}_{\substack{\text{feed forward} \\ \text{moving average}}}, \qquad (13.132)$$

where $x[n] \sim \mathrm{WN}(0, \sigma_x^2)$

This is just a standard IIR filter.

31

Finding ARMA(p,q) Model Parameters

How can find the model coefficients $\{a_k, b_k\}$ given an output signal $y(n)$?

The autocorrelation can be computed numerically in MATLAB given an output signal.

We can also derive an analytical expression of the autocorrelation of the output signal:

$$r_{yy}[\ell] = -\sum_{k=1}^{p} a_k r_{yy}[\ell - k] + \sigma_x^2 \sum_{k=0}^{q} b_k h[k - \ell], \quad \text{all } \ell. \qquad (13.137)$$

However, we have an issue with this expression. The impulse response $h[n]$ is actually some function of the coefficients $\{a_k, b_k\}$. The issue is that this function is a non-linear. So solving the analytical expression becomes troublesome.

However, if we get rid of the second term by setting $q = 0$, the relation becomes linear and we have an AR(p) process:

$$r_{yy}[\ell] = -\sum_{k=1}^{p} a_k r_{yy}[\ell - k] + \sigma_x^2 b_0 h[-\ell]. \quad \text{all } \ell \qquad (13.138)$$

The impulse response of an all-pole system satisfies the difference equation

$$h[n] = -\sum_{k=1}^{p} a_k h[n - k] + b_0 \delta[n], \quad n \geq 0 \qquad (13.139)$$

and $h[n] = 0$ for $n < 0$. From (13.139) we obtain $h[0] = b_0$ and without loss of generality we choose $b_0 = 1$

Quiz: why can we choose to set $b_0 = 1$?

Well, we are sending a WN signal into an LTI system. There is a certain about of power in the white noise. The b_0 is a constant used to amplify the power. It is difficult to distinguish whether the power originates from the power in white noise or the power in the gain. The power appears as one quantity. In this case, we fix one of the filter coefficients and we say that any power that we observe is in the white noise.

$x(n) \sim WN(0, \sigma_x^2)$ → [LTI] → $y(n)$

We only consider positive lags $\ell > 0$ because $h(\ell) = 0$ for $\ell < 0$ i.e., the impulse response does not exists of negative times because it is a causal system.

$$r_{yy}[\ell] = -\sum_{k=1}^{p} a_k r_{yy}[\ell - k], \quad \ell > 0 \qquad (13.140)$$

The ARMA(p, q) model consists of two submodels. Normally, we do not use the full ARMA model but only one of the submodels.

AR(q) models

AR(p) model is given by:

$$y(n) = -\sum_{k=1}^{p} \left[a_k\, y(n-k) \right] + b_0 x(n)$$

The autocorrelation of AR(q) model was derived in Eq. 13.141 as:

$$r_{yy}(\ell) = -\sum_{k=1}^{p} a_k r_{yy}(\ell - k), \quad \ell > 0$$

This is useful because we can use it to find the coefficients $\{a_k\}$ of an AR(q) model. Given an output signal $y(n)$, we can compute the autocorrelation $r_{yy}(\ell)$ numerically in MATLAB. This means that the expression in Eq. 13.141 becomes a set of p linear equations.

For example, if we want to model second-order AR model, $p = 2$ we get two equations with two unknowns:

$$r_{yy}(1) = -a_1 r_{yy}(0) - a_2 r_{yy}(-1)$$

$$r_{yy}(2) = -a_1 r_{yy}(1) - a_2 r_{yy}(0)$$

We can write it into matrix form:

$$\begin{bmatrix} r_{yy}(0) & r_{yy}(-1) \\ r_{yy}(1) & r_{yy}(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = - \begin{bmatrix} r_{yy}(1) \\ r_{yy}(2) \end{bmatrix}$$
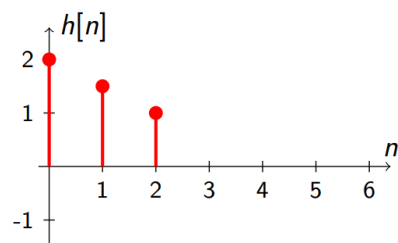
MA(q) models

MA(q) model is given by:

$$y(n) = \sum_{k=0}^{q} b_k\, x(n-k)$$

This is the ARMA(p, q) model where the feedback part is excluded i.e., all values of a_k is set to zero.

Quiz: determine kind of model based on the inpulse response

The impulse response for a specific ARMA(p,q) model is shown below.

What kind of model is shown?

A: ARMA(1,0)=AR(1)

B: ARMA(2,0)=AR(2)

C: ARMA(0,1)=MA(1)

D: ARMA(1,1)

E: There are two correct answers among A-D

F: A-D are all wrong

The correct answer is F.

The autoregressive part of the ARMA(p,q) model has infinite impulse response. What we see in the figure is finite impulse response with three elements. Therefore, it must be an MA(q) model.

Since the figure shows three filter coefficients, the system must be an MA(2) process.

If the figure indicated infinite filter coefficients, we could not have determined the order of the model. In this case, it would be either an autoregressive model or a high-order MA(q) model.

Quiz: is the comb filter an MA(1) proces?

One special type of filter, *the comb filter*, is given by

$$y(n) = x(n) - x(n - D)$$

where D is a positive integer. Is this an MA(1) process?

A: Yes

B: No

The answer is no!