Homework 13

Table of Contents

[✔] ADSI Problem 7.1

A tremendous amount of functionality is found in MATLABs toolboxes and Hilbert transforms are no exception. Type doc hilbert in MATLAB and read the documentation.

```
doc hilbert
```

[✔] ADSI Problem 7.2: Hilbert transform of a cosine signal

A continuous time signal is given by $x(t) = 4 + 3\cos(\omega t)$*. What is the Hilbert transform of this signal?*

Signal $u(t)$	Hilbert transform[fn 1] $H(u)(t)$
$\sin(\omega t)$ [fn 2]	$\operatorname{sgn}(\omega)\sin\left(\omega t - \frac{\pi}{2}\right) = -\operatorname{sgn}(\omega)\cos(\omega t)$
$\cos(\omega t)$ [fn 2]	$\operatorname{sgn}(\omega)\cos\left(\omega t - \frac{\pi}{2}\right) = \operatorname{sgn}(\omega)\sin(\omega t)$
$e^{i\omega t}$	$\operatorname{sgn}(\omega)e^{i\left(\omega t - \frac{\pi}{2}\right)} = -i \cdot \operatorname{sgn}(\omega)e^{i\omega t}$
$\frac{1}{t^2+1}$	$\frac{t}{t^2+1}$
e^{-t^2}	$2\pi^{-1/2}F(t)$ (see Dawson function)
Sinc function $\frac{\sin(t)}{t}$	$\frac{1-\cos(t)}{t}$
Rectangular function $\operatorname{rect}(t) = \Pi(t) = \begin{cases} 0, & \text{if } \lvert t\rvert > \frac{1}{2} \\ \frac{1}{2}, & \text{if } \lvert t\rvert = \frac{1}{2} \\ 1, & \text{if } \lvert t\rvert < \frac{1}{2}. \end{cases}$	$\frac{1}{\pi}\ln\left\lvert\frac{t+\frac{1}{2}}{t-\frac{1}{2}}\right\rvert$
Dirac delta function $\delta(x) = \begin{cases} +\infty, & x = 0 \\ 0, & x \neq 0 \end{cases}$	$\frac{1}{\pi t}$
Characteristic Function $\chi_{[a,b]}(t)$	$\frac{1}{\pi}\ln\left\lvert\frac{t-a}{t-b}\right\rvert$

Real signal	Hilbert transform
$a_1 g_1(t) + a_2 g_2(t);\ a_1, a_2 \in \mathbb{C}$	$a_1 \hat{g}_1(t) + a_2 \hat{g}_2(t)$
$h(t - t_0)$	$\hat{h}(t - t_0)$
$h(at);\ a \neq 0$	$\operatorname{sgn}(a)\hat{h}(at)$
$\frac{\mathrm{d}}{\mathrm{d}t}h(t)$	$\frac{\mathrm{d}}{\mathrm{d}t}\hat{h}(t)$
$\delta(t)$	$\frac{1}{\pi t}$
$e^{\mathrm{j}t}$	$-\mathrm{j}e^{\mathrm{j}t}$
$e^{-\mathrm{j}t}$	$\mathrm{j}e^{-\mathrm{j}t}$
$\cos(t)$	$\sin(t)$
$\operatorname{rect}(t)$	$\frac{1}{\pi}\ln\lvert(2t+1)/(2t-1)\rvert$
$\operatorname{sinc}(t)$	$\frac{\pi t}{2}\operatorname{sinc}^2(t/2) = \sin(\pi t/2)\operatorname{sinc}(t/2)$
$1/(1+t^2)$	$t/(1+t^2)$

$$H[4 + 3\cos(\omega t)] = H[4] + 3H[\cos(\omega t)]$$

$$H[4 + 3\cos(\omega t)] = 0 + 3\operatorname{sgn}(\omega)\sin(\omega t)$$

[✔] ADSI Problem 7.3: One-sided spectrum of analytical signals

Consider the signal

$$x(n) = e^{-0.001(n-255)^2}\cos(1.8n) \quad 0 \leq n \leq 511.$$

Calculate the analytical signal in MATLAB and check that the spectrum is one-sided as expected.

Recall that we start with a real-valued signal $x_r(t)$ and we want we need to represent the signal as a complex sinusoid:

$$x_c(t) = x_r(t) + j\,x_i(t)$$

$x_c(t)$ is known as an ***analytic signal*** because it has no negative-frequency spectral components.

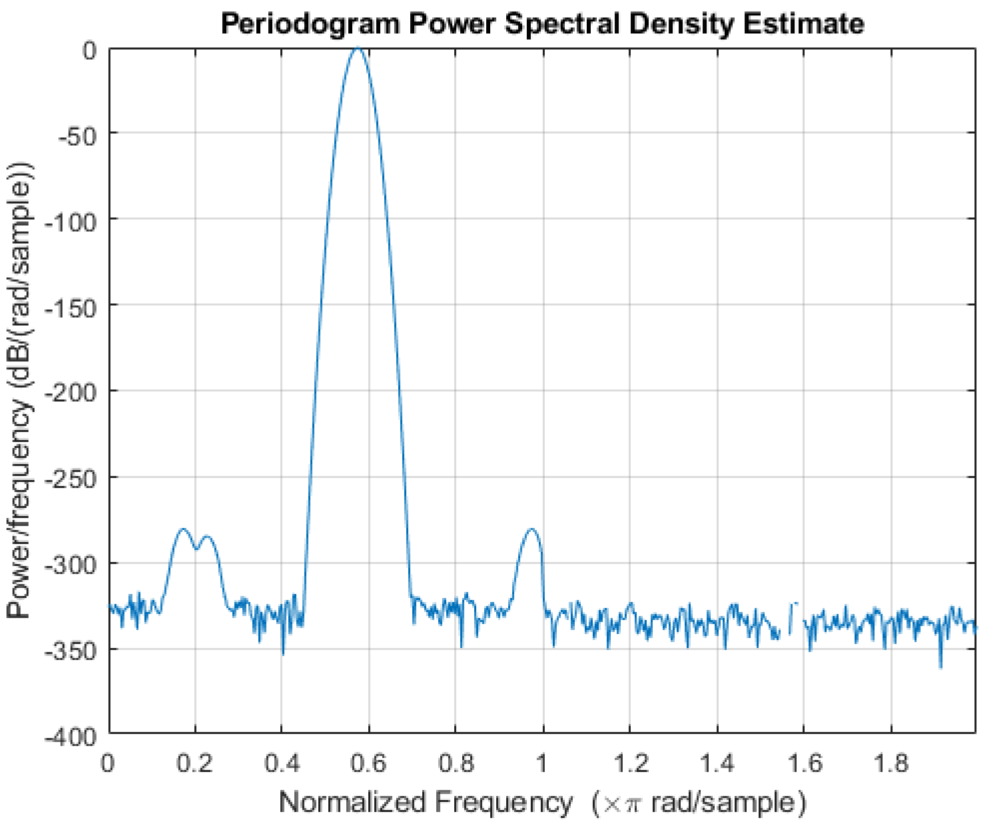
To show that the spectrum is one-sided, we have to show that the frequency response $X_c(e^{j\omega})$ is zero over the negative frequency range.

We can do this by estimating the power spectrum density (PSD) of the analytical signal using the periodogram.

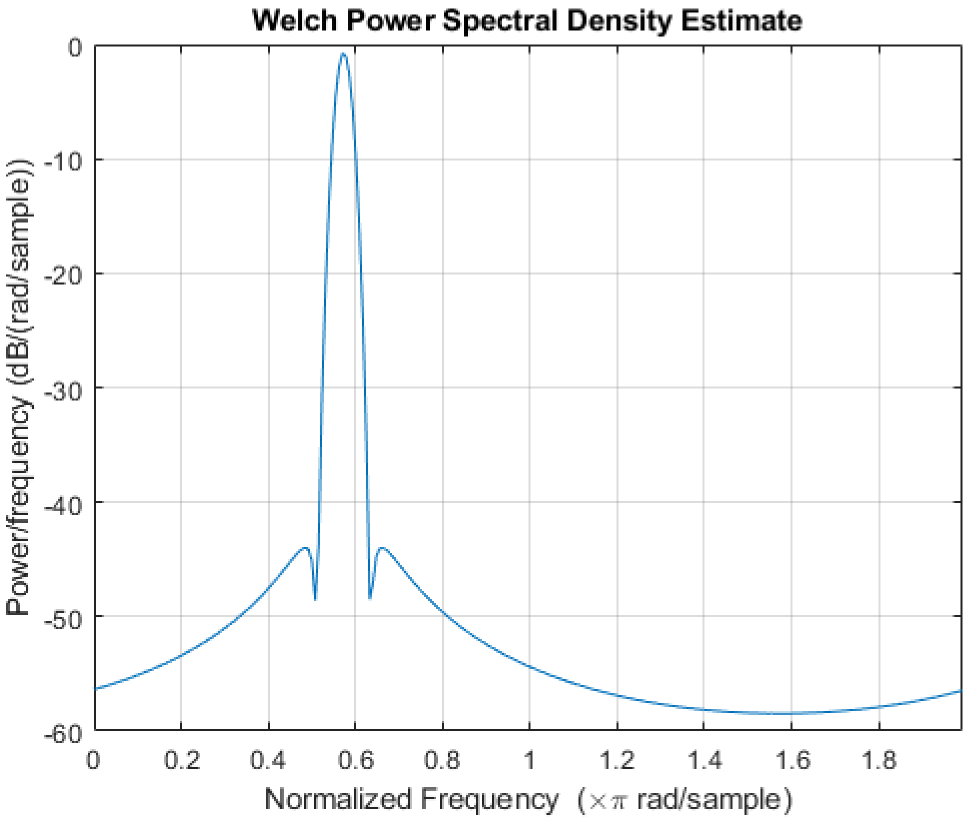
In the MATLAB plot, the normalised **positive frequency range** is [0, 1] whereas the **negative frequency range** is between 1 and 2. Notice there is no power spikes in the negative frequency range, only noise.

```
n = (0:511)';
x = exp(-0.001.*(n-225).^2).*cos(1.8*n);
```

```
% Compute the analytic signal of the real signal x(n)
y = hilbert(x);

% Compute and plot the PSD estimate
periodogram(y)
```




```
% PSD estimate using the Welch method
pwelch(y)
```



[✔] ADSI Problem 7.4: Swept sine wave

In (audio) measurements an exponentially swept sine wave is often used to measure transfer functions and nonlinear distortion, see e.g. Novák et al. "Nonlinear System Identification Using Exponential Swept-Sine Signal" IEEE Trans. Instrum. Measure. **59**, 2220-2229, (2010) or "Synchronized Swept-Sine: Theory, Application and Implementation". J. Audio Eng. Soc. **63**, 787-798, (2015). In the continuous time domain an exponential swept sine wave is given by

$$s(t) = \sin\left(2\pi f_1 L \left[\exp\left(\frac{t}{L}\right) - 1\right]\right)$$

where

$$L = \frac{T}{\ln\left(\frac{f_2}{f_1}\right)}$$

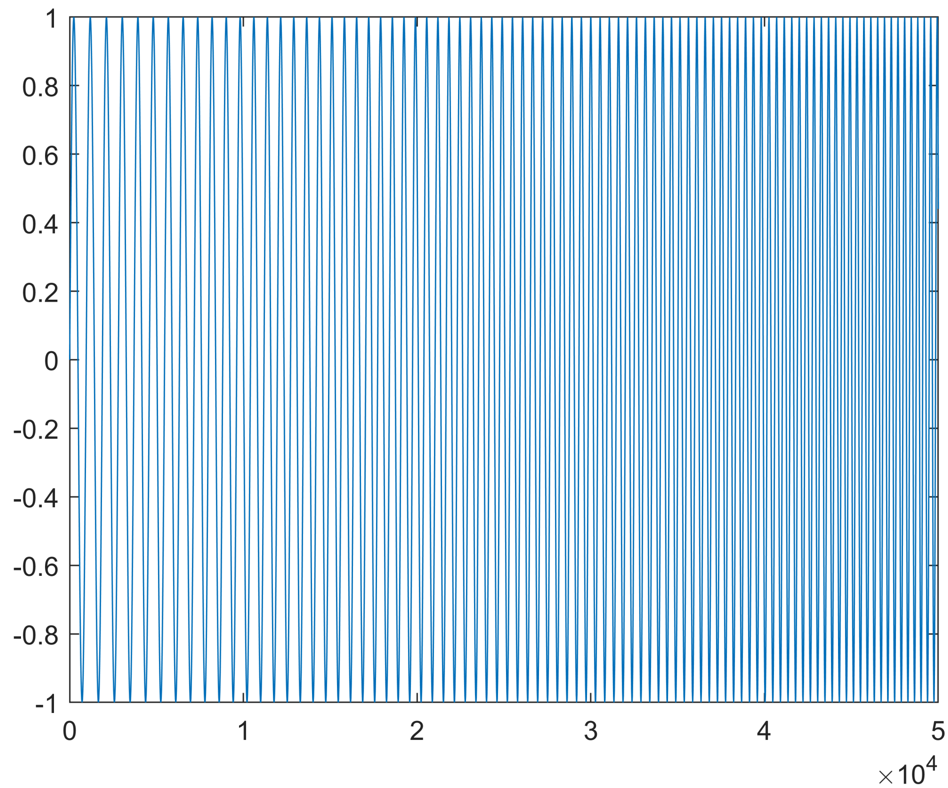
T is the duration of the sweep, f_1 and f_2 are the start and stopping frequencies of the sweep.

```
clear variables;
```

[✔] 1. Design swept sine signal

1. Design a 5 second long swept sine sampled at 48 kHz. The sine should sweep from 50 Hz to 5000 Hz. Play the sound on your pc and check that is performs as expected.

```
Ts = 48000;  % Sampling rate
f1 = 50;     % Starting 1/frequency
f2 = 5000;   % Stopping 1/frequency
T = 5*Ts;    % Duration of the sweep
L = T / log(f2/f1);

n = 0:T-1;
s = sin(2*pi * f1 * L * (exp(n/L) - 1) / Ts);

% Write to file
% audiowrite('swept_sine.wav', s, Ts);

% Play sound
soundsc(s, Ts);

% Plot it
plot(n, s)
xlim([0, 50000]);
```

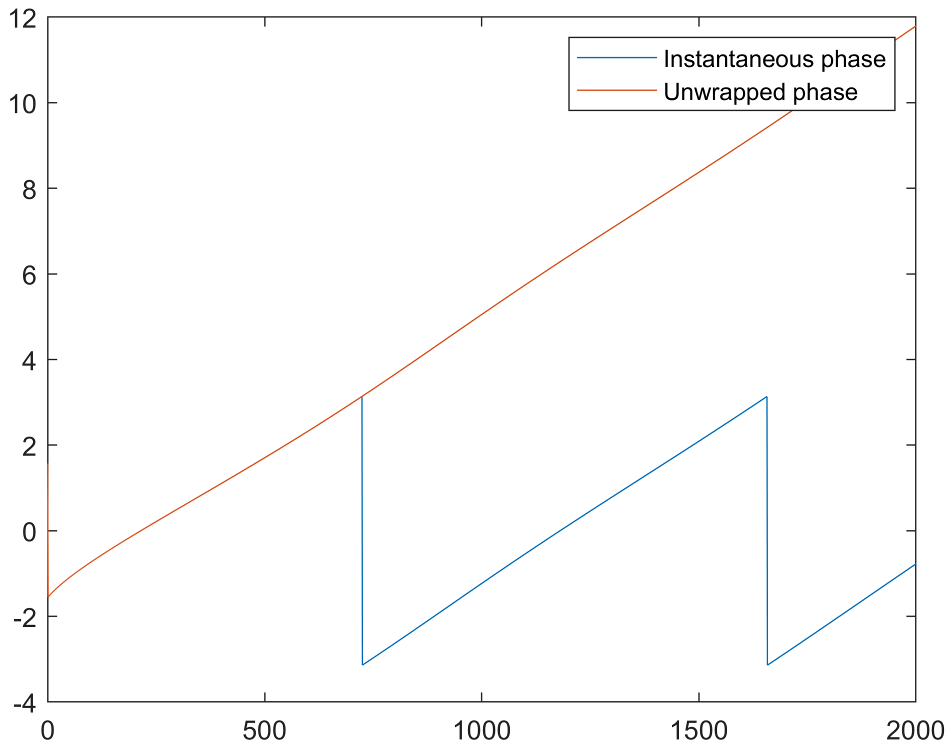


[✓] 2. Compute instantaneous frequency

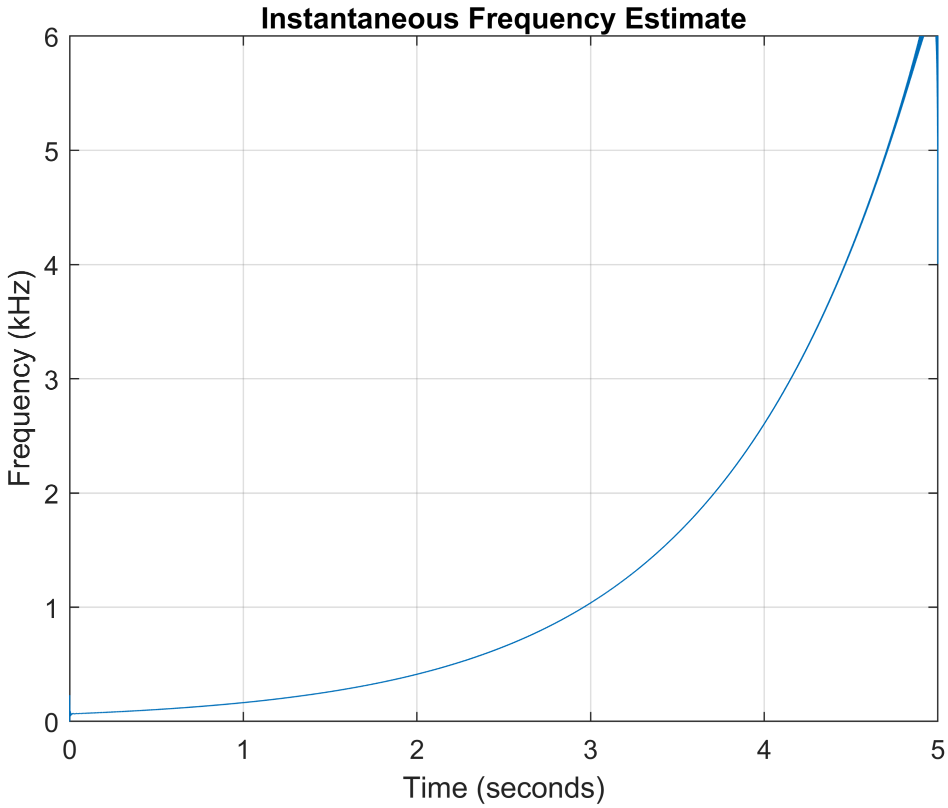
2. Use equation 9.6 $F(t) = d/dt \tan^{-1} x_i(t)/x_r(t)$ to calculate the instantaneous frequency of the swept sine signal and compare with the real frequency.

$$F(t) = \frac{d}{dt}\,\phi(t) = \frac{d}{dt}\tan^{-1}\left(\frac{x_i(t)}{x_r(t)}\right). \qquad (9\text{–}6)$$

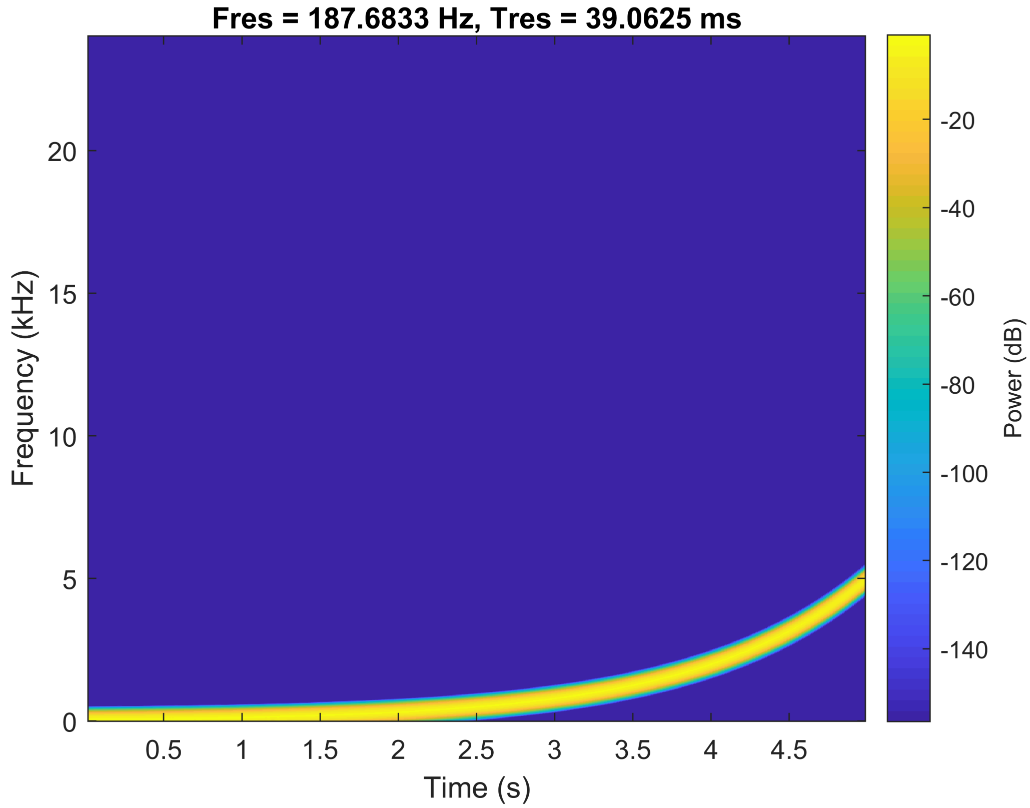
```
s_c = hilbert(s);  % Compute the analytic signal

% The instantaneous phase can be computed using MATLAB's angle function
inst_phase = angle(s_c);
plot(n, inst_phase, n, unwrap(inst_phase))
legend('Instantaneous phase', 'Unwrapped phase')
xlim([0, 2000])
```



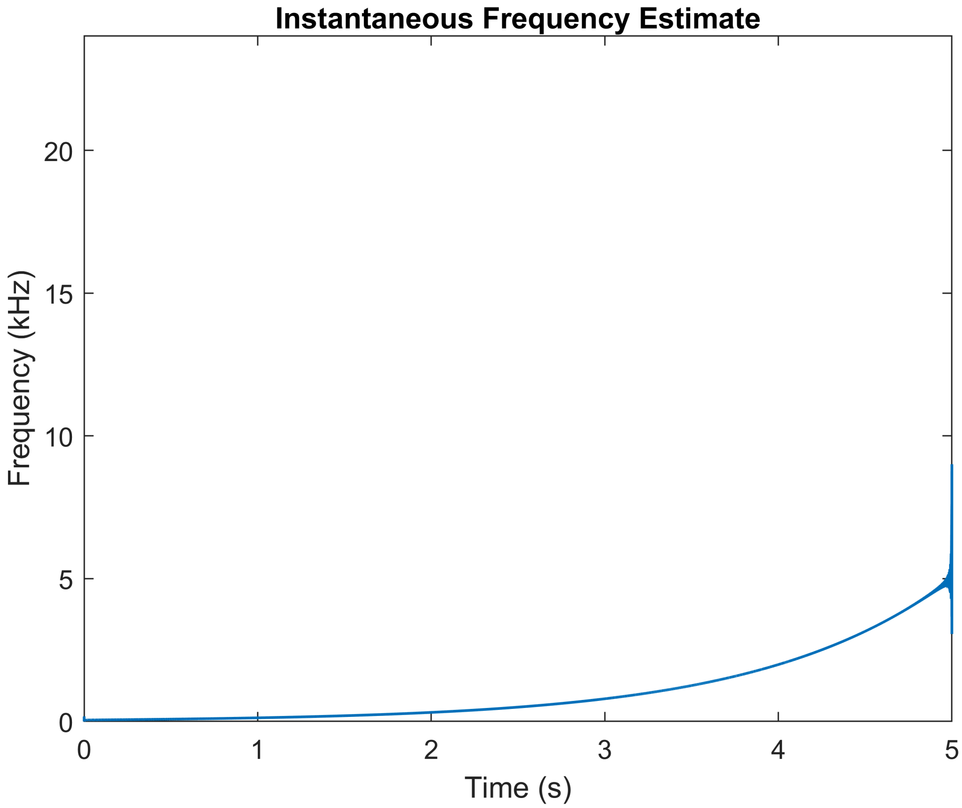

```
% Approximate the derivative of the unwrapped instantaneous phase
F = diff(unwrap(inst_phase));

% Plot it
plot(n(2:end)/Ts, F*10)
ylim([0, 6])
grid on
ylabel('Frequency (kHz)')
xlabel('Time (seconds)')
title('Instantaneous Frequency Estimate')
```




```
% Estimate the spectrogram of the waveform using the pspectrum function.
pspectrum(s, Ts, 'spectrogram')
```




```
% The instfreq function computes and displays the instantaneous frequency in one step.
instfreq(s, Ts, 'Method', 'hilbert')
```



[?] ADSI Problem 7.5: Envelope Detection

The textbook states that the Hilbert transform can be used to find the envelope of a signal. The validity of this statement is investigated in this problem.

Consider an exponentially decaying oscillating signal given by

$$x(n) = 4e^{-0.01n}\cos(\pi n)$$

```
clear variables;
clf
```

[?] 1. Plot the signal and envelope on the same graph

Plot this signal and use equation 9.4 to plot the envelope on the same graph. Compare the result with the expected outcome.

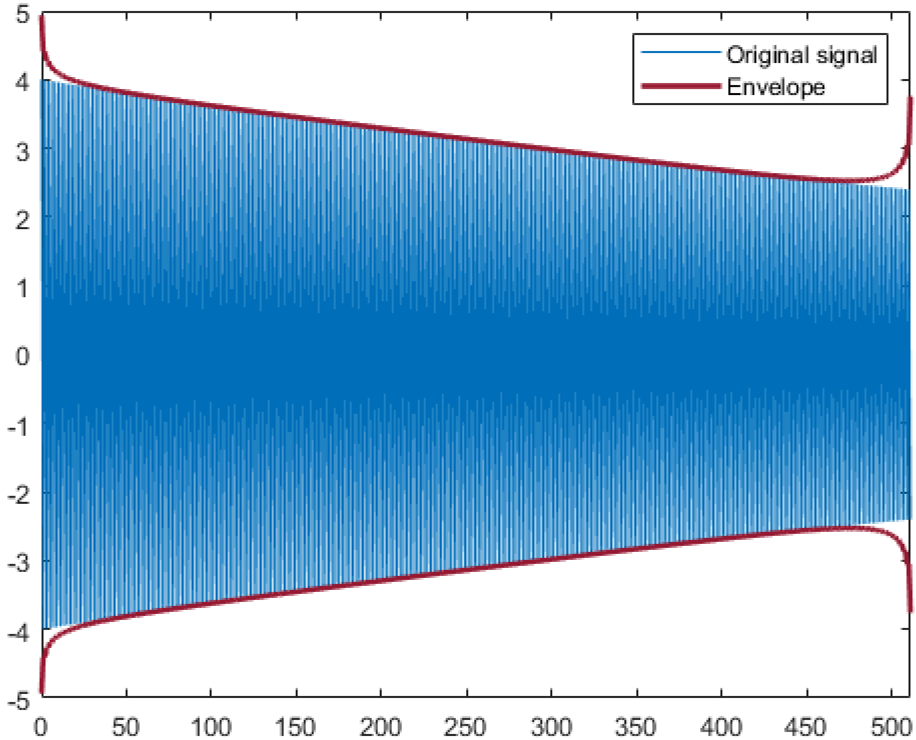
The books states: If a real sinewave $x_r(t)$ is amplitude modulated so its envelope contains information, then we can measure the instantaneous envelope $E(t)$ values from an analytic version of the signal using the Equation 9-4:

$$E(t) = |x_c(t)| = \sqrt{x_r(t)^2 + x_i(t)^2}. \tag{9-4}$$

Equation 9-4 states that the envelope of the signal is equal to magnitude of $x_c(t)$.

In MATLAB the complex magnitude of a signal can be computed with the abs function.

```
n = (0:511);
x_r = 4*exp(-0.001.*n).*cos(pi*n);

% Compute the analytic signal
x_c = hilbert(x_r);

% Find the envelope by computing the magnitude of the signal
x_envelope = abs(x_c);

% Plot the real signal
plot(n, x_r)
hold on

% Plot the envelope
plot(n, [-1;1]*x_envelope, 'Color', [0.6 0.1 0.2],'Linewidth',2)
hold off

legend('Original signal', 'Envelope')
xlim([0, max(n)])
```



[?] I expected the envelope to be smooth over the whole range. However, it seems that envelope around the edges are off slight. Why is that?

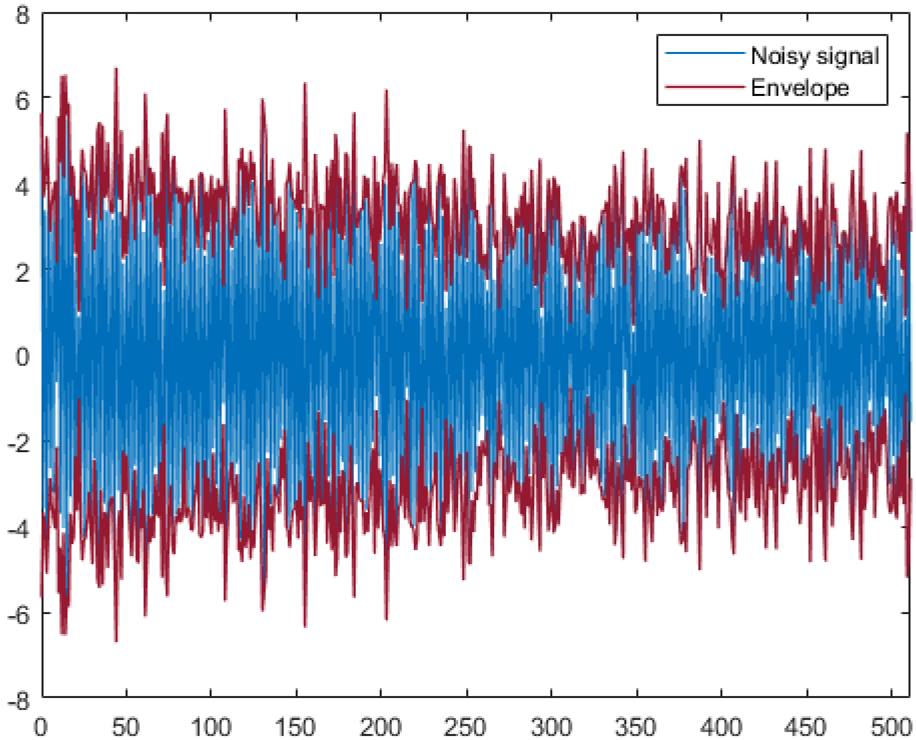
[✔] 2. Describe how noise affects Hilbert transform's ability to recover the envelope

In real life noise is always present in measurements.

2. Add white Gaussian noise WGN $\sim (0,1)$ to the signal and repeat. Describe the influence of the noise in the ability of the Hilbert transform to recover the envelope.

```
N = 512;
n = (0:N-1);
w = randn(N, 1)';
x_r = 4*exp(-0.001.*n).*cos(pi*n) + w;

x_c = hilbert(x_r);
x_envelope = abs(x_c);

plot(n, x_r)
hold on
plot(n, [-1;1]*x_envelope, 'Color', [0.6 0.1 0.2],'Linewidth',1)
hold off
legend('Noisy signal', 'Envelope')
```

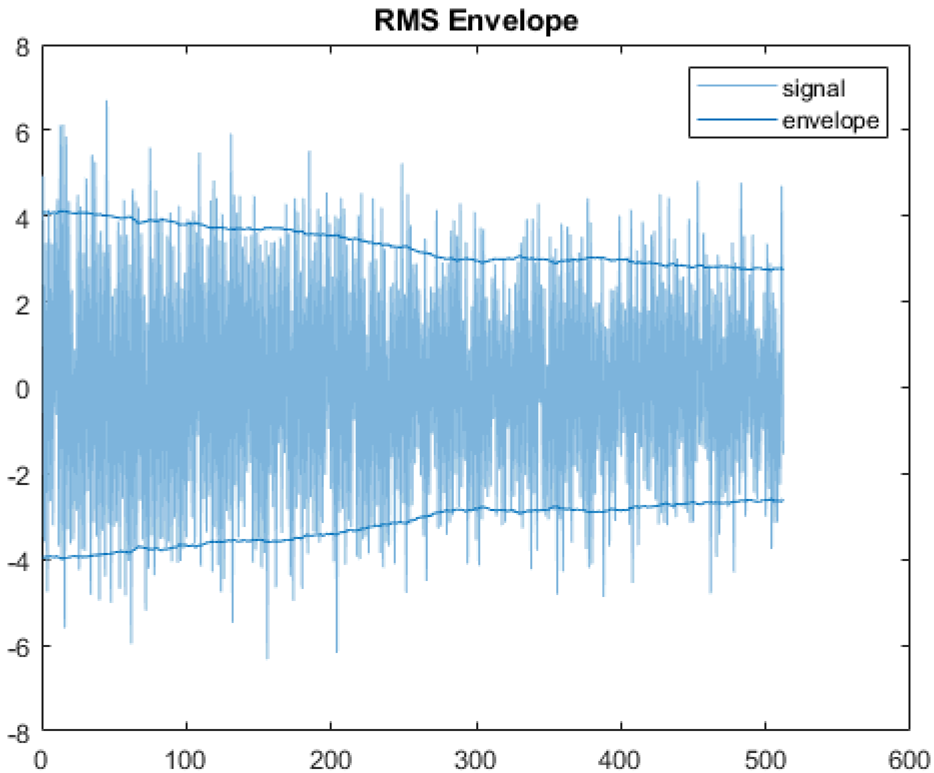
```
xlim([0, max(n)])
```



Hilbert transform is sensitive to noise. We observe spurious peaks in the envelope due to noise.

We can generate a moving Root-Mean-Square (RMS) average envelopes by using a sliding window of size W. Using a large window size smooths out the envelope.

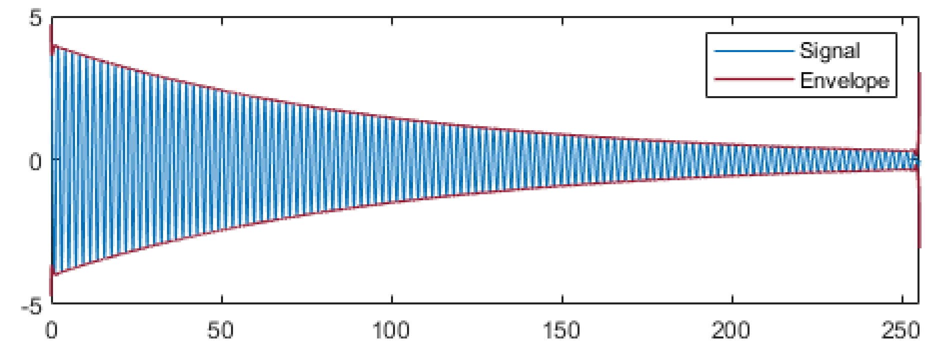
```
envelope(x_r, 100, 'rms');
```

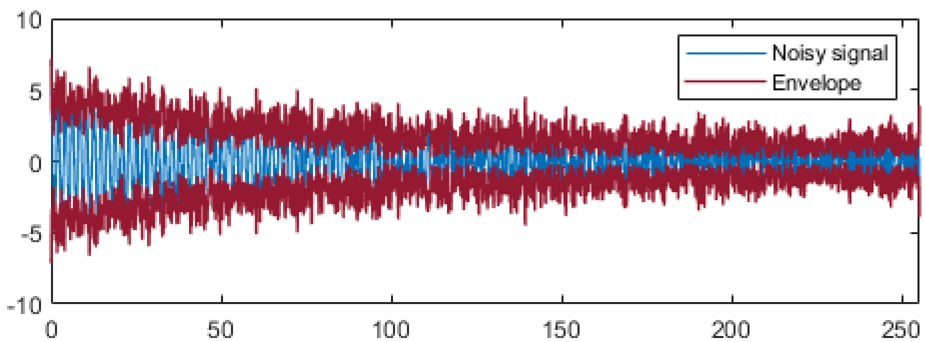


[✔] 3. Try with other two signals

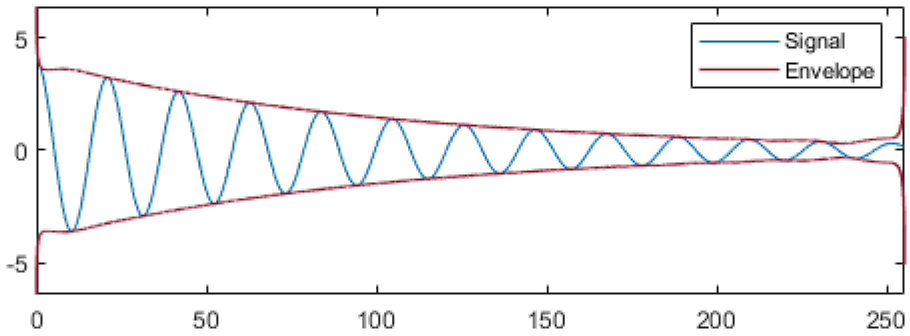
3. Repeat the above two questions for the signals $x_1(n) = 4e^{-0.01n}\cos(3n)$ and $x_2(n) = 4e^{-0.01n}\cos(0.3n)$.

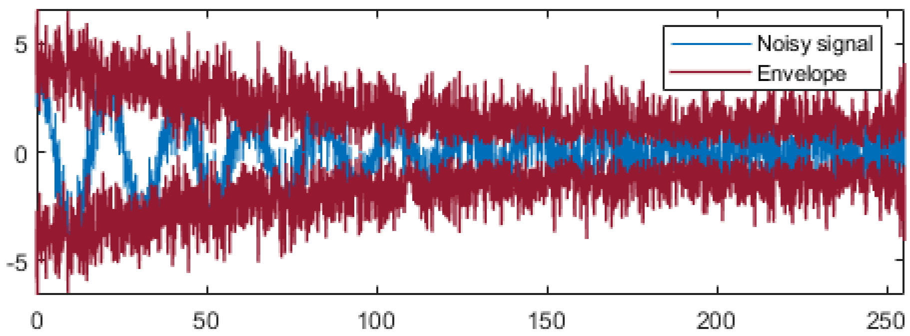
```
clear variables;

N = 256;
n = 0:0.1:N-1;
x = 4*exp(-0.01.*n).*cos(3*n);
plot_signal_and_envelope(n, x)
```






```
x = 4*exp(-0.01.*n).*cos(0.3*n);
plot_signal_and_envelope(n, x)
```





[»] ADSI Problem 7.8: Amplitude modulated signal

Consider an amplitude modulated signal given by

$$x(t) = \big(1 + 0.2\cos(2\pi 37t)\big)\cos(2\pi * 20000t)$$

```
clear variables;
```

[✔] 1. Plot 1 second of the signal and confirm that it is an amplitude modulated signal with a carrier frequency of 20 kHz.

The discrete-time signal $x[n]$ of continuous-time signal $x(t)$ can be represented as:

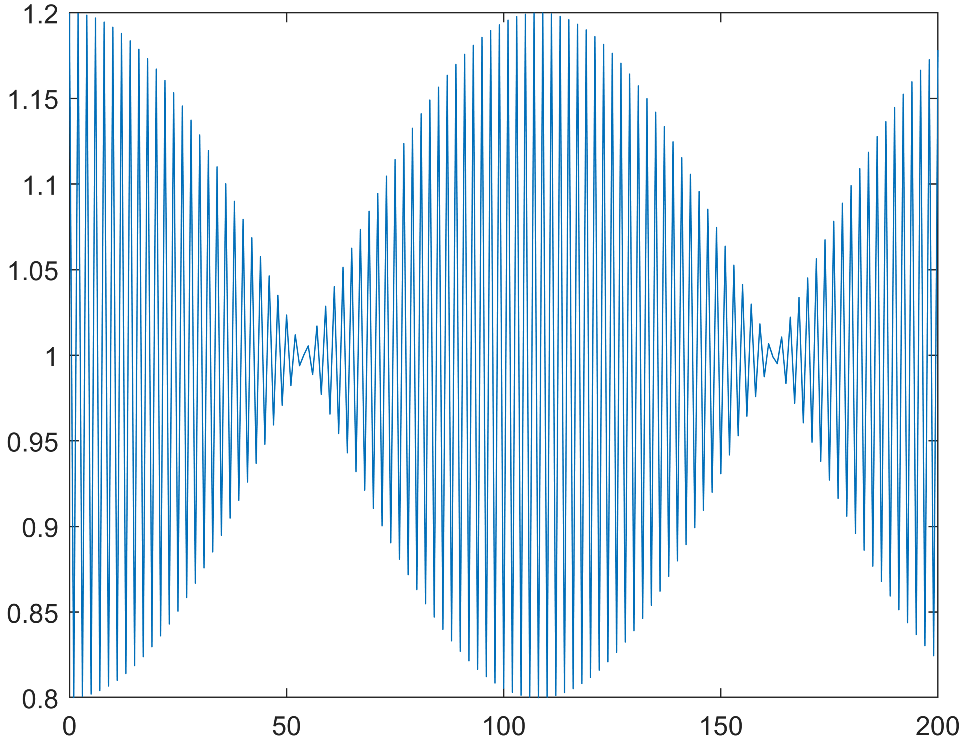
$$x[n] = s(n \cdot T_s)$$

where

- n is an integer denoting the sample index

- T_s is the sampling period

```
Ts = 8000; % Sampling rate
n = 0:200;
x = 1 + 0.2.*cos(2*pi*37*n/Ts).*cos(2*pi*20000*n/Ts);
plot(n, x)
```



[»] 2. Use the procedure from Figure 9.8 in the note to frequency shift the signal to 15 kHz. Plot the spectrum of the signal at each step in the procedure.

Functions

```
function plot_signal_and_envelope(n, x)
w = randn(numel(n), 1)';
x_noisy = x + w;

x_c = hilbert(x);
```

```
x_envelope = abs(x_c);

x_noisy_c = hilbert(x_noisy);
x_noisy_envelope = abs(x_noisy_c);

clf
figure
ax1 = subplot(2,1,1);
plot(n, x)
hold on
plot(n, [-1;1]*x_envelope, 'Color', [0.6 0.1 0.2],'Linewidth',1)
hold off
legend('Signal', 'Envelope')
xlim([0, max(n)])

ax2 = subplot(2,1,2);
plot(n, x_noisy)
hold on
plot(n, [-1;1]*x_noisy_envelope, 'Color', [0.6 0.1 0.2],'Linewidth',1)
hold off
legend('Noisy signal', 'Envelope')
xlim([0, max(n)])
end
```