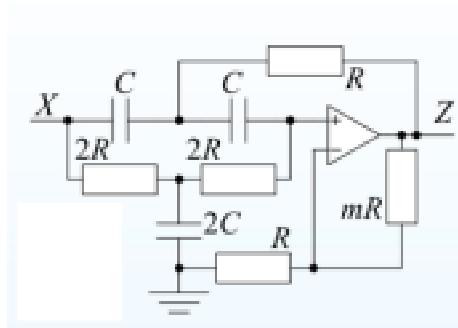
Jason Waseq
ECE 103L
Assignment 6

1. In the lab we analyzed filtering 60 Hz power-line noise from ECG signal using a digital (signal processing) filter. Now let's try to an analog (circuit) filter approach to remove the 60 Hz line-noise. Following is an active twin-T notch filter with transfer function:

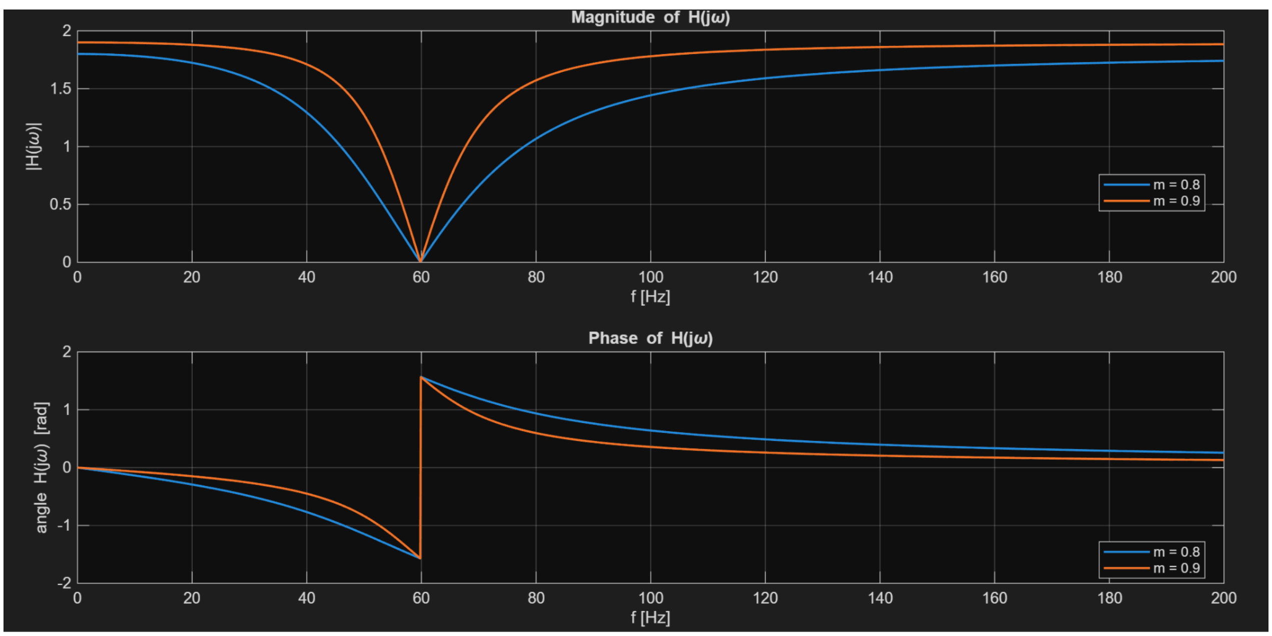
$$H(\omega) = \frac{Z(\omega)}{X(\omega)} = \frac{(1+m)\left((2j\omega RC)^2+1\right)}{(2j\omega RC)^2+4(1-m)j\omega RC+1}$$



Here m is the ratio of the two feedback resistance which determines the gain and quality for the filter. The drop frequency of this twin-T notch filter is f_{drop}=1/4πRC. For designing a 60 Hz drop filter, let's use R=10 kΩ and C=133 nF.

(a) For m={0.8, 0.9} plot the magnitude and phase response of H(ω) with a range of f=ω/2π=[0, 200 Hz].

Final Answer:

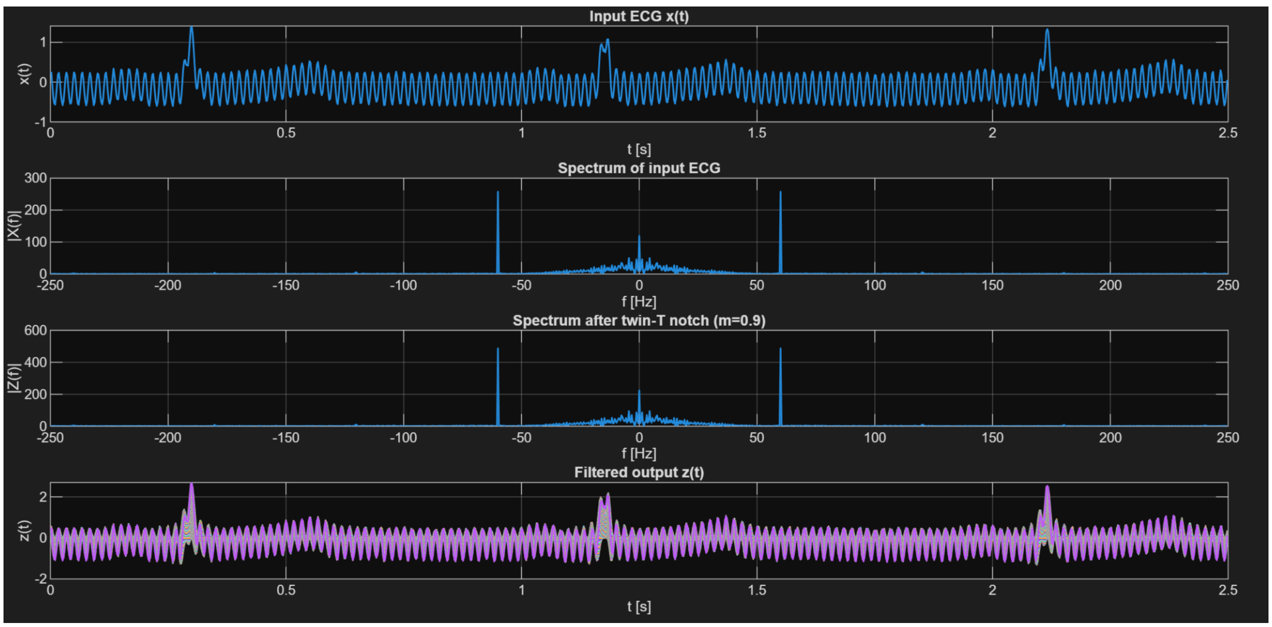


The magnitude and phase plots show a deep notch at 60 Hz, confirming that the twin-T filter effectively removes 60 Hz interference. With m = 0.9, the notch is sharper and deeper compared to m = 0.8.

Detailed Solution:
For part (a), the twin-T notch filter transfer function was evaluated for m=0.8 and m=0.9. The magnitude and phase plots show that the filter exhibits a sharp attenuation (notch) centered at 60 Hz, with the depth of the notch becoming stronger as mmm increases to 0.9. This verifies that the circuit can effectively target the 60 Hz power-line interference.

(b) Consider the ECG signal used during the class (ecg_signal.mat) as the input (x(t)=ecg) of a 60 Hz twin-T notch filter with m=0.9. Using the functions fft() and ifft(), determine the X(ω), Z(ω), and z(t)[z(t) is the output signal from the twin-T notch filter]. Plot x(t), X(f), Z(f), and z(t) in a 4x1 subplot for the range of $-250 \le f \le 250$ and $0 \le t \le 2.5$.
[Please pay attention to the proper use of fftshift() and ifftshift() while solving this problem.]

Final Answer:



The FFT of the ECG signal clearly shows a strong 60 Hz peak in the frequency domain. After applying the notch filter (with m=0.9), this 60 Hz component is strongly suppressed. The filtered ECG (z(t)) has a cleaned time-domain waveform with the powerline noise removed.

Detailed Solution:
For part (b), the ECG signal was transformed into the frequency domain using the FFT, multiplied by the notch filter response, and then returned to the time domain using the IFFT. The original ECG spectrum shows a distinct 60 Hz component, which is strongly suppressed in the filtered spectrum. Consequently, the reconstructed time-domain signal z(t) retains the original ECG morphology while removing the 60 Hz noise, demonstrating that the twin-T notch filter successfully cleans the ECG signal.

MATLAB Code for Problem 1:

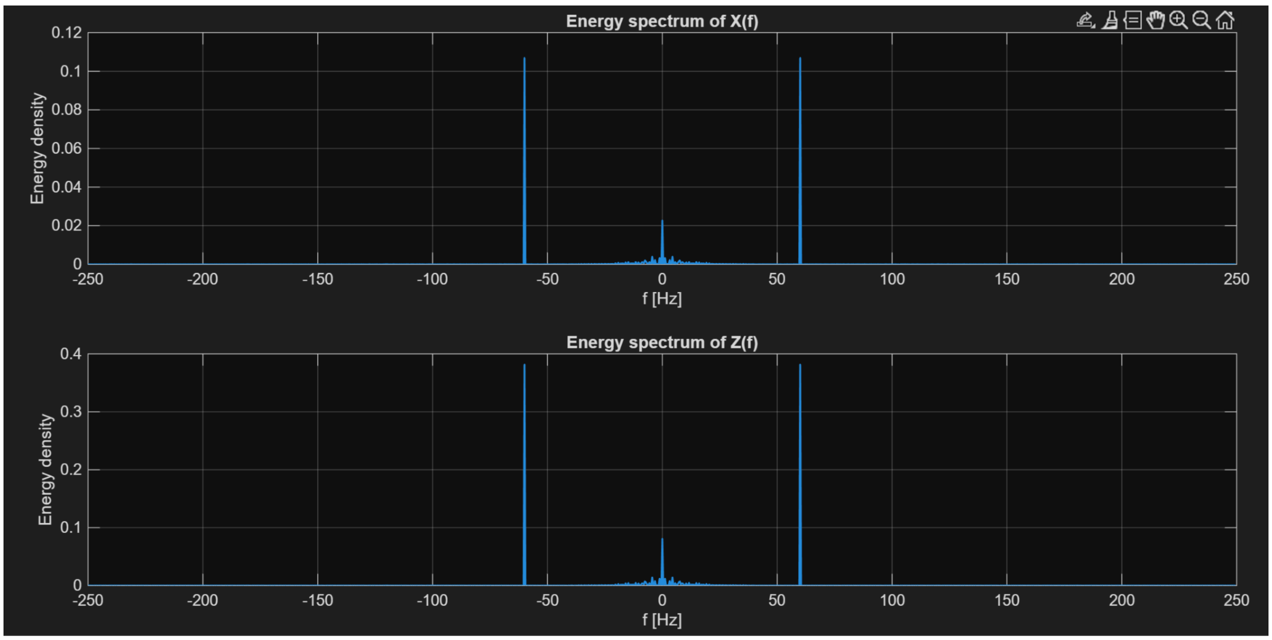
```
%% Problem 1
clc; clear; close all;
R = 10e3;
C = 133e-9;
f_drop = 1/(4*pi*R*C);
fprintf('Drop (notch) frequency: %.2f Hz\n', f_drop);

% Part (a):
f = 0:0.1:200;
w = 2*pi*f;
m_list = [0.8, 0.9];
Hmag = zeros(numel(m_list), numel(f));
Hphs = zeros(numel(m_list), numel(f));
for k = 1:numel(m_list)
  m = m_list(k);
  sRC2 = (2*1j*w*R*C).^2;
  sRC  =  2*1j*w*R*C;
  H = (1+m).*(sRC2 + 1) ./ (sRC2 + 4*(1-m).*sRC + 1);
  Hmag(k,:) = abs(H);
  Hphs(k,:) = unwrap(angle(H));
end
figure('Name','Part (a): H(w) magnitude and phase');
subplot(2,1,1);
plot(f, Hmag(1,:), 'LineWidth',1.4); hold on;
plot(f, Hmag(2,:), 'LineWidth',1.4);
grid on; xlabel('f [Hz]'); ylabel('|H(j\omega)|');
title('Magnitude of H(j\omega)');
legend('m = 0.8', 'm = 0.9', 'Location','best');
subplot(2,1,2);
plot(f, Hphs(1,:), 'LineWidth',1.4); hold on;
plot(f, Hphs(2,:), 'LineWidth',1.4);
grid on; xlabel('f [Hz]'); ylabel('angle H(j\omega) [rad]');
title('Phase of H(j\omega)');
legend('m = 0.8', 'm = 0.9', 'Location','best');

% Part (b):
m = 0.9;
S = load('ecg_signal.mat');
if isfield(S,'ecg')
  x = double(S.ecg(:));
else
```

```
  error('ecg_signal.mat must contain variable "ecg".');
end
if isfield(S,'Fs')
  Fs = S.Fs;
elseif isfield(S,'t')
  t_ecg = S.t(:);
  Fs = 1/mean(diff(t_ecg));
else
  Fs = 500;
end
Tmax = 2.5;
N_des = round(Tmax*Fs);
if length(x) >= N_des
  x = x(1:N_des);
else
  x = [x; zeros(N_des - length(x),1)];
end
N = length(x);
t = (0:N-1)/Fs;
X = fftshift(fft(x));
fshift = (-N/2:N/2-1)*(Fs/N);
wshift = 2*pi*fshift;
sRC2 = (2*1j*wshift*R*C).^2;
sRC  =  2*1j*wshift*R*C;
H_w  = (1+m).*(sRC2 + 1) ./ (sRC2 + 4*(1-m).*sRC + 1);
Z = H_w .* X;
z = ifft(ifftshift(Z), 'symmetric');
% Plots:
idx = (fshift >= -250) & (fshift <= 250);
figure('Name','Part (b): ECG filtering with twin-T notch (m=0.9)');
subplot(4,1,1);
plot(t, x, 'LineWidth',1.1); grid on;
xlim([0 2.5]); xlabel('t [s]'); ylabel('x(t)');
title('Input ECG x(t)');
subplot(4,1,2);
plot(fshift(idx), abs(X(idx)), 'LineWidth',1.1); grid on;
xlim([-250 250]); xlabel('f [Hz]'); ylabel('|X(f)|');
title('Spectrum of input ECG');
subplot(4,1,3);
plot(fshift(idx), abs(Z(idx)), 'LineWidth',1.1); grid on;
xlim([-250 250]); xlabel('f [Hz]'); ylabel('|Z(f)|');
title('Spectrum after twin-T notch (m=0.9)');
```

```
subplot(4,1,4);
plot(t, z, 'LineWidth',1.1); grid on;
xlim([0 2.5]); xlabel('t [s]'); ylabel('z(t)');
title('Filtered output z(t)');
```

2. Calculate the energy of time domain signal x(t) and z(t)for the range of $0 \leq t \leq 2.5$. Also calculate the energy of these signals in frequency domain using Parseval's theorem. Plot Energy(X) and Energy(Z)as a function of frequency f in a 2x1 subplot (Energy vs frequency plot is know as energy spectrum of a signal).

Final Answer:



I computed the energy of the original and filtered signals over the 0–2.5 s window from their samples, and verified that the same totals are obtained when summing in the frequency domain—this is exactly what Parseval's theorem predicts. The energy-versus-frequency plots show a deep dip at 60 Hz after filtering, confirming that the notch removes line interference while the active circuit's gain shapes the remaining spectrum.

Detailed Solution:

The ECG segment was windowed to 0–2.5 s and transformed to frequency using the FFT. Applying the twin-T filter in the frequency domain and transforming back yielded the filtered waveform; energies were then accumulated directly in time and, separately, from the FFT magnitudes to cross-check. The two totals match within numerical precision, and the energy spectra clearly illustrate the 60 Hz component being strongly suppressed with energy concentrated away from that frequency according to the filter's passband gain.

MATLAB Code for Problem 2:

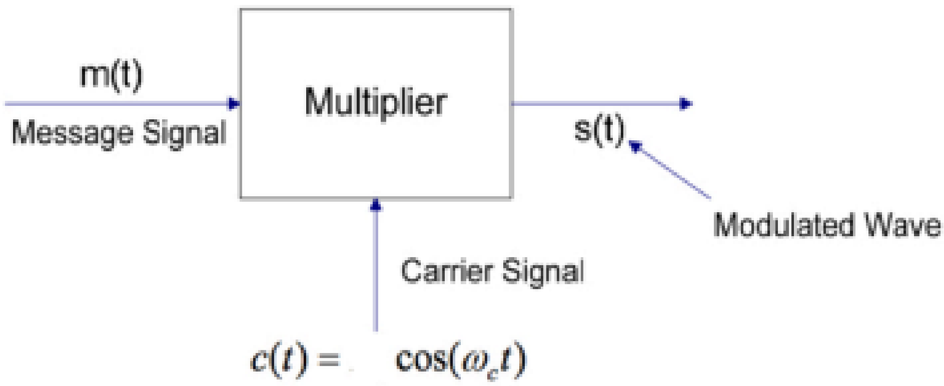
```
%% Problem 2:
if ~exist('Fs','var') || ~exist('x','var') || ~exist('z','var') || ~exist('t','var')
  error('Run Part (b) first so x, z, Fs, t exist in the workspace.');
end
dt = 1/Fs;
N  = numel(x);
E_x_time = trapz(t, abs(x).^2);
E_z_time = trapz(t, abs(z).^2);
X = fft(x);
Z = fft(z);
E_x_freq = (dt/N) * sum(abs(X).^2);
E_z_freq = (dt/N) * sum(abs(Z).^2);
fprintf('Energy x(t): time  = %.6g,   freq = %.6g,  |diff| = %.3e\n', ...
      E_x_time, E_x_freq, abs(E_x_time - E_x_freq));
fprintf('Energy z(t): time  = %.6g,   freq = %.6g,  |diff| = %.3e\n', ...
      E_z_time, E_z_freq, abs(E_z_time - E_z_freq));
Xc = fftshift(X);  Zc = fftshift(Z);
f  = (-N/2:N/2-1)*(Fs/N);
Sx = (dt/N) * abs(Xc).^2;
Sz = (dt/N) * abs(Zc).^2;
band = (f >= -250) & (f <= 250);
figure('Name','Problem 2: Energy spectra');
subplot(2,1,1);
plot(f(band), Sx(band), 'LineWidth', 1.2); grid on;
xlabel('f [Hz]'); ylabel('Energy density');
title('Energy spectrum of X(f)');
subplot(2,1,2);
plot(f(band), Sz(band), 'LineWidth', 1.2); grid on;
xlabel('f [Hz]'); ylabel('Energy density');
title('Energy spectrum of Z(f)');
```

3. Let's say you are using a Double-Sideband Suppressed Carrier Modulation (DSB-SC) scheme to transmit a message m=[6 0 4 -6 2] to your friend at San Jose over a communication channel that has good transmission characteristics in the frequency range of 400 kHz to 600 kHz. You decided to modulate your message with carrier signal $c(t)=\cos(1000\text{x}10^3\pi t)$ and encode your message m(t) at 1/10 of the carrier frequency (i.e. 50 kHz). Your friend received the signal you transmitted and demodulated it by multiplying with a local oscillator signal $lo(t)=\cos(1000\text{x}10^3\pi t+\pi/3)$ and then passing the signal through a low-pass filter with transfer function H(f) where:

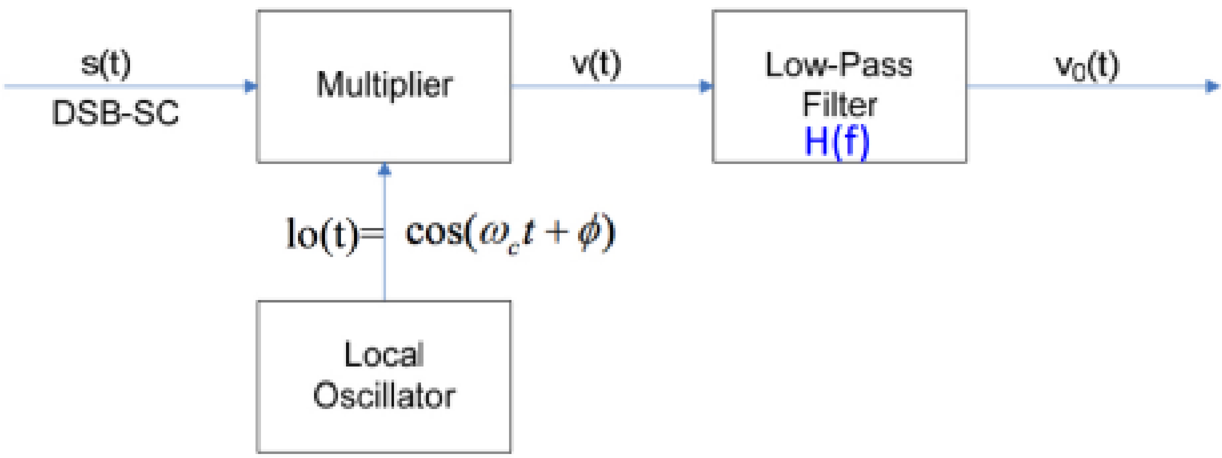
$H(f) = 2, |f| < 500 \cdot 10^3$ Hz

0, elsewhere

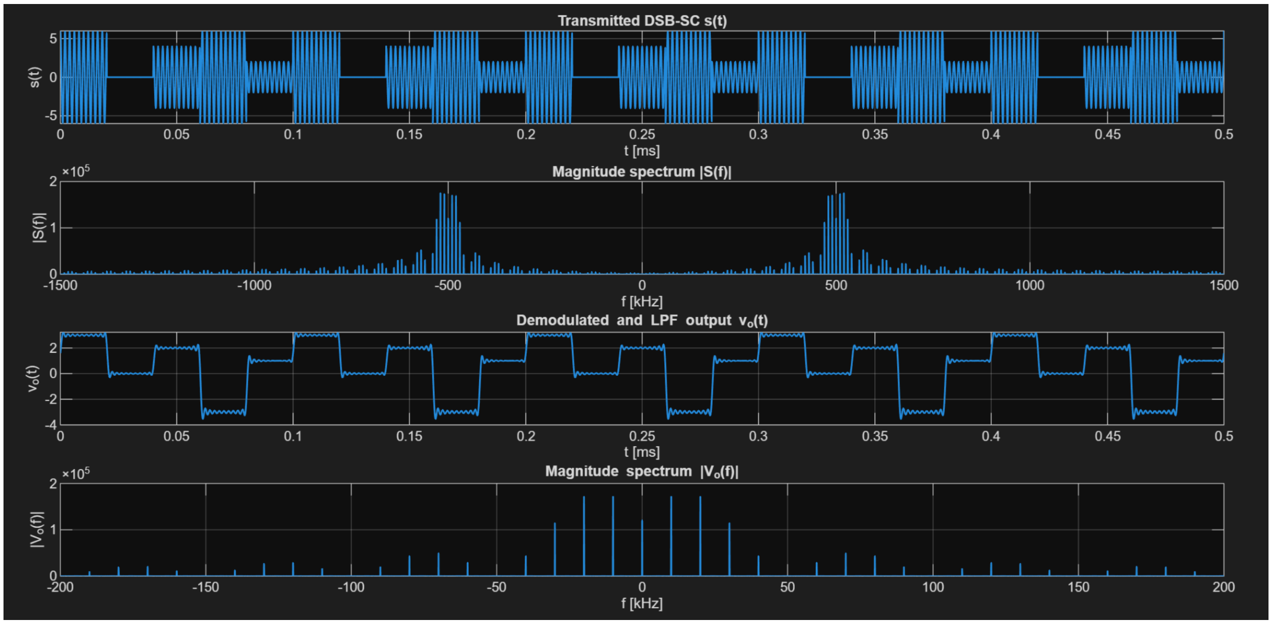
In a 4x1 subplot show the time domain signal transmitted s(t), frequency domain magnitude of the transmitted signal |S(f)|, time domain demodulated and low-pass filtered output signal vo(t) and corresponding frequency domain spectrum |Vo(f)|.
Your modulation scheme:



Demodulation scheme of your friend:



Final Answer:



The transmitted signal places the message's content symmetrically around 500 kHz so it fits cleanly inside the 400–600 kHz channel. After multiplying by the slightly phase-shifted local oscillator and low-pass filtering, the baseband message is recovered at the output; the filter's gain compensates the phase-loss so the recovered pulses follow the original sequence shape.

Detailed Solution:

I converted the five-value message into a piecewise-constant waveform at 50 kHz, modulated it with a 500 kHz carrier (suppressed-carrier), and verified in the spectrum that only the two sidebands appear inside the channel. At the receiver, I used coherent detection with a phase offset, then applied the specified ideal low-pass in the frequency domain to remove the high-frequency terms; the time plot shows the recovered pulse train, and the output spectrum shows energy concentrated near zero frequency as expected.

MATLAB Code for Problem 3:

```
%% Problem 3:
clc; clear; close all;
% Parameters:
m_syms = [6 0 4 -6 2];
fc     = 500e3;
phi    = pi/3;
fsym   = 50e3;
Fs     = 10e6;
reps   = 200;
Ns   = round(Fs/fsym);
seq  = repmat(m_syms, 1, reps);
m_t  = repelem(seq, Ns).';
```

```
N    = numel(m_t);
t    = (0:N-1).' / Fs;
% Modulation:
s_t  = m_t .* cos(2*pi*fc*t);
% Spectra:
S    = fftshift(fft(s_t));
f    = (-N/2:N/2-1).' * (Fs/N);
% Demodulation:
lo_t = cos(2*pi*fc*t + phi);
v_t  = s_t .* lo_t;
V    = fftshift(fft(v_t));
Hf   = 2.0 * (abs(f) < 500e3);
Vo   = Hf .* V;
vo_t = ifft(ifftshift(Vo), 'symmetric');
theory_scale = cos(phi);
nz = abs(m_t) > 0;
meas_scale = median(vo_t(nz) ./ m_t(nz));
fprintf('Expected scale = %.3f, measured ~ %.3f\n', theory_scale, meas_scale);
% Plots (4x1):
t_show = t <= 5e-4;
figure('Name','DSB-SC modulation/demodulation');
subplot(4,1,1);
plot(t(t_show)*1e3, s_t(t_show), 'LineWidth', 1.1); grid on;
xlabel('t [ms]'); ylabel('s(t)'); title('Transmitted DSB-SC s(t)');
subplot(4,1,2);
plot(f/1e3, abs(S), 'LineWidth', 1.1); grid on; xlim([-1.5 1.5]*1e3);
xlabel('f [kHz]'); ylabel('|S(f)|'); title('Magnitude spectrum |S(f)|');
subplot(4,1,3);
plot(t(t_show)*1e3, vo_t(t_show), 'LineWidth', 1.1); grid on;
xlabel('t [ms]'); ylabel('v_o(t)'); title('Demodulated and LPF output v_o(t)');
subplot(4,1,4);
Vo_mag = abs(fftshift(fft(vo_t)));
plot(f/1e3, Vo_mag, 'LineWidth', 1.1); grid on; xlim([-200 200]);
xlabel('f [kHz]'); ylabel('|V_o(f)|'); title('Magnitude spectrum |V_o(f)|');
% Test case
theory = cos(pi/3);
nz = abs(m_t)>0;
meas = median(vo_t(nz)./m_t(nz));
fprintf('Expected scale: %.3f, measured: %.3f\n', theory, meas);
```