Homework1_2

Programmers

Mohammad Mahdi Elyasi - 9823007

Moein Nasiri - 9823093

Clear the Workspace

```
Clear all;
close all;
clc;
```

Homework1_6

In this task we want to show what would be sampled signal and how different it is with original signal

Declaring variables:

```
fs = 50000;
t = 0:1 / fs:0.004;
f1 = 1000;
f2 = 4000;
f3 = 6000;
fs_low = 5000;
t_sampled = 0:1 / fs_low:0.004;
```

Now we declare signals need to be plootted

```
x_original = cos(2 * pi * f1 * t) + cos(2 * pi * f2 * t) + cos(2 * pi * f3 * t);
x_sampled = cos(2 * pi * f1 * t_sampled) + cos(2 * pi * f2 * t_sampled) + cos(2 * pi * f3 * t_s
```

Now we plot signals for the first part

```
figure('Name', 'Aliasing in the time domain');
plot(t, x_original, 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Signals');
grid on;
hold on;
plot(t_sampled, x_sampled, 'o', 'LineWidth', 2);
legend('Original signal', 'Sampled signal');
```

part 2

Here we want to reconstruct original signals, with a specific frequency

Now we declare variables for second part of this task

```
prc_rate = 100;
t1 = -0.004:1 / (prc_rate * fs_low):0.004;
```

```
x1 = zeros(1, (length(t1) + 1) / 2);
x1(1:prc_rate:end) = x_sampled;
```

Here we declare signals need to be plotted

```
h = sinc(fs_low * t1);
y = conv(x1, h, 'same');
```

Now we plot original signal, Sampled signal and reconstructed signal

```
figure('Name', 'Aliasing in the time domain');
plot(t, x_original, 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Signals');
grid on;
hold on;
plot(t_sampled, x_sampled, 'o', 'LineWidth', 2);
hold on;
plot(t1((length(t1) + 1) / 2:end), y, 'LineWidth', 1.5);
legend('Original signal', 'Sampled signal', 'Reconstructed signal')
```

- Based on sampled frequency we would know that frequencies which doesn't satisfy nyquist law can't be plotted. so only cosine with 1k hertz frquency can be plotted
- If we don't have ideal filter, then the reconstructed signal can't be the same as original signal

Homework1_7

Here in this task we want to show if we sample signals, what would happen to their frequency spectrum

Now we declare variables needed for plot:

```
t = -5:0.01:5;
f1 = 4;
f2 = 5;
f3 = 10;
f4 = 20;
t_1 = -5:1 / f1:5;
t_2 = -5:1 / f2:5;
t_3 = -5:1 / f3:5;
t_4 = -5:1 / f4:5;
```

Now we declare signals needed for plot

```
x_t = (sinc (5 * t)) .^ 2;
x_t1 = (sinc (5 * t_1)) .^ 2;
x_t2 = (sinc (5 * t_2)) .^ 2;
x_t3 = (sinc (5 * t_3)) .^ 2;
x_t4 = (sinc (5 * t_4)) .^ 2;

figure('Name', 'Fft of signals');
subplot(2, 1, 1)
```

```
plot(t, abs(fftshift(fft(x_t))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of oiginal signal');
grid on;

subplot(2, 1, 2)
plot(t_1, abs(fftshift(fft(x_t1))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of sampled signal1');
grid on;

figure('Name', 'Fft of signals');
subplot(2, 1, 1)
plot(t, abs(fftshift(fft(x_t))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of original signal');
grid on;

subplot(2, 1, 2)
plot(t_2, abs(fftshift(fft(x_t2))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of sampled signal2');
grid on;

figure('Name', 'Fft of signals');
subplot(2, 1, 1)
plot(t, abs(fftshift(fft(x_t))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of original signal');
grid on;

subplot(2, 1, 2)
plot(t_3, abs(fftshift(fft(x_t3))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of sampled signal3');
grid on;

figure('Name', 'Fft of signals');
subplot(2, 1, 1)
plot(t, abs(fftshift(fft(x_t))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Fft of original signal');
grid on;

subplot(2, 1, 2)
plot(t_4, abs(fftshift(fft(x_t4))), 'LineWidth', 2);
xlabel('Time (s)');
```

```
ylabel('Amplitude');
title('Fft of sampled signal4');
grid on;
```

- Based on what we saw, Signal3 and Signal4 had no problem in terms of aliasing of course based on Nyquist law, signal3 which had 2B=fs may have problems if we had sinusodial signals.
- So we prefer to have 2B > fs.
- For signal2 since Nyquist law isn't satisfied we have aliasing , and it makes sinc to be integer. and we know it is 0 if sinc becomes inteer (except in 0). so we see a DC signal.
- For signal1 Nyquist law isn't satisfied so we see sum of DC and tip of the signal.
- In conclusion, if we don't satisfy Nyquist law, then we see aliasing if frequency spectrum.

Homework1_8

In this task, we want to see aliasing in frequency spectrum

Now we declare needed variables:

```
N = 256;
t = -5:1 / N:5;
t1 = -5:1 / (2 * N):5;
t2 = -5:3 / (2 * N):5;
t3 = -5:3 / N:5;
```

Here we declare needed signals:

```
xt = sinc(2 * t);
xt1 = sinc(2 * t1);
xt2 = sinc(2 * t2);
xt3 = sinc(2 * t3);
```

And finally we plot the signals

```
figure('Name', 'Original Signal');
subplot(2, 1, 1)
plot(t, xt, 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Original signal');
grid on;

subplot(2, 1, 2)
plot(t, abs(fftshift(fft(xt))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('sampled signal');
grid on;

figure('Name', 'Original Signal');
subplot(2, 1, 1)
plot(t1, xt1, 'LineWidth', 2);
```

```
xlabel('Time (s)');
ylabel('Amplitude');
title('Original signal');
grid on;

subplot(2, 1, 2)
plot(t1, abs(fftshift(fft(xt1))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('sampled signal');
grid on;

figure('Name', 'Original Signal');
subplot(2, 1, 1)
plot(t2, xt2, 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Original signal');
grid on;

subplot(2, 1, 2)
plot(t2, abs(fftshift(fft(xt2))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('sampled signal');
grid on;

figure('Name', 'Original Signal');
subplot(2, 1, 1)
plot(t3, xt3, 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('Original signal');
grid on;

subplot(2, 1, 2)
plot(t3, abs(fftshift(fft(xt3))), 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Amplitude');
title('sampled signal');
grid on;
```

Homework1_9

Here we want to display multi-rate filter bank

Firstly we need to declare needed variables

```
fs = 2;
t_max = 256;
t = -t_max:1 / fs:t_max;
f1 = 1/16;
f2 = 5/16;
f3 = 9/16;
```

```
f4 = 13/16;
```

Here we declare signals:

```
x = cos(2 * pi * f1 * t) + cos(2 * pi * f2 * t) + cos(2 * pi * f3 * t) + cos(2 * pi * f4 * t);
FT_x = (1 / fs) * fftshift(fft(x));
f_axis = linspace(-fs / 2, fs / 2, length(FT_x));
```

in this cell we plot the declared signals

```
figure('Name', 'Fourier Transform of Continuous Time Signal');
plot(f_axis, abs(FT_x), 'LineWidth', 1.5);
xlabel('Frequency (Hz)');
ylabel('Amplitude');
title('Fourier Transform of Continuous Time Signal');
xlim([-1, 1]);
grid on;
```

In this part we want to read excells data and import it into matlab

```
analy_filter = readmatrix('filters.xls', 'Sheet', 1);
analy_filter1 = analy_filter(1, :);
analy_filter2 = analy_filter(2, :);
analy_filter3 = analy_filter(3, :);
analy_filter4 = analy_filter(4, :);
```

Here we want to declare some variables in order to plot it

```
w_axis = linspace(0, pi, 1000);
N = length(analy_filter1);
n = 0:N -1;
```

Here we declare DTFT of signal:

```
DTFT_analy_filter1 = analy_filter1 * exp(-1j * n' * w_axis);
```

Here we plot some of analytic filters:

```
figure('Name', 'Amplitude and phase 4 analytic filters');
subplot(4, 2, 1)
plot(w_axis / pi, abs(DTFT_analy_filter1), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT analy filter1');
xlim([0 1]);
grid on;

subplot(4, 2, 2)
plot(w_axis / pi, angle(DTFT_analy_filter1), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT analy filter1');
```

```
grid on;

w_axis = linspace(0, pi, 1000);
N = length(analy_filter2);
n = 0:N -1;
DTFT_analy_filter2 = analy_filter2 * exp(-1j * n' * w_axis);

subplot(4, 2, 3)
plot(w_axis / pi, abs(DTFT_analy_filter2), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT analy filter2');
xlim([0 1]);
grid on;

subplot(4, 2, 4)
plot(w_axis / pi, angle(DTFT_analy_filter2), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT analy filter2');
grid on;

w_axis = linspace(-0, pi, 1000);
N = length(analy_filter3);
n = 0:N -1;
DTFT_analy_filter3 = analy_filter3 * exp(-1j * n' * w_axis);

subplot(4, 2, 5)
plot(w_axis / pi, abs(DTFT_analy_filter3), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT analy filter3');
xlim([0 1]);
grid on;

subplot(4, 2, 6)
plot(w_axis / pi, angle(DTFT_analy_filter3), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT analy filter3');
grid on;

w_axis = linspace(0, pi, 1000);
N = length(analy_filter4);
n = 0:N -1;
DTFT_analy_filter4 = analy_filter4 * exp(-1j * n' * w_axis);

subplot(4, 2, 7)
plot(w_axis / pi, abs(DTFT_analy_filter4), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT analy filter4');
```

```
xlim([0 1]);
grid on;

subplot(4, 2, 8)
plot(w_axis / pi, angle(DTFT_analy_filter4), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT analy filter4');
grid on;
```

Now we import datas from 2ns sheet of excell and plot composition filters

```
Composition_filter = readmatrix('filters.xls', 'Sheet', 2);
Composition_filter1 = Composition_filter(1, :);
Composition_filter2 = Composition_filter(2, :);
Composition_filter3 = Composition_filter(3, :);
Composition_filter4 = Composition_filter(4, :);
```

Here we declare some variables for plot:

```
w_axis = linspace(0, pi, 1000);
N = length(Composition_filter1);
n = 0:N -1;
% Here we declare DTFT of composition filters:
DTFT_Composition_filter1 = Composition_filter1 * exp(-1j * n' * w_axis);
```

2 of processes from top will be repeated for every subplot plot of filter is written

```
figure('Name', 'Amplitude and phase 4 Composition filters');
subplot(4, 2, 1)
plot(w_axis / pi, abs(DTFT_Composition_filter1), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT Composition filter1');
xlim([0 1]);
grid on;

subplot(4, 2, 2)
plot(w_axis / pi, angle(DTFT_Composition_filter1), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT Composition filter1');
grid on;

w_axis = linspace(0, pi, 1000);
N = length(Composition_filter2);
n = 0:N -1;
DTFT_Composition_filter2 = Composition_filter2 * exp(-1j * n' * w_axis);

subplot(4, 2, 3)
plot(w_axis / pi, abs(DTFT_Composition_filter2), 'LineWidth', 1.5);
```

```
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT Composition filter2');
xlim([0 1]);
grid on;

subplot(4, 2, 4)
plot(w_axis / pi, angle(DTFT_Composition_filter2), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT Composition filter2');
grid on;

w_axis = linspace(-0, pi, 1000);
N = length(Composition_filter3);
n = 0:N -1;
DTFT_Composition_filter3 = Composition_filter3 * exp(-1j * n' * w_axis);

subplot(4, 2, 5)
plot(w_axis / pi, abs(DTFT_Composition_filter3), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT Composition filter3');
xlim([0 1]);
grid on;

subplot(4, 2, 6)
plot(w_axis / pi, angle(DTFT_Composition_filter3), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT Composition filter3');
grid on;

w_axis = linspace(0, pi, 1000);
N = length(Composition_filter4);
n = 0:N -1;
DTFT_Composition_filter4 = Composition_filter4 * exp(-1j * n' * w_axis);

subplot(4, 2, 7)
plot(w_axis / pi, abs(DTFT_Composition_filter4), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Amplitude');
title('Absolute Value of DTFT Composition filter4');
xlim([0 1]);
grid on;

subplot(4, 2, 8)
plot(w_axis / pi, angle(DTFT_Composition_filter4), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase');
xlim([0 1]);
title('Phase of DTFT Composition filter4');
```

```
grid on;
```

In this part we want to do the processing work, like down-sampling and up-sampling of datas and then plot input and output signals

```
analy_filter1_out = filter(analy_filter1, 1, x);
analy_filter2_out = filter(analy_filter2, 1, x);
analy_filter3_out = filter(analy_filter3, 1, x);
analy_filter4_out = filter(analy_filter4, 1, x);
analy_filter1_out_down = downsample(analy_filter1_out, 4);
analy_filter2_out_down = downsample(analy_filter2_out, 4);
analy_filter3_out_down = downsample(analy_filter3_out, 4);
analy_filter4_out_down = downsample(analy_filter4_out, 4);
analy_filter1_out_down_pc = 2 * analy_filter1_out_down;
analy_filter2_out_down_pc = 0 * analy_filter2_out_down;
analy_filter3_out_down_pc = analy_filter3_out_down;
analy_filter4_out_down_pc = 0.5 * analy_filter4_out_down;
analy_filter1_out_up = upsample(analy_filter1_out_down_pc, 4);
analy_filter2_out_up = upsample(analy_filter2_out_down_pc, 4);
analy_filter3_out_up = upsample(analy_filter3_out_down_pc, 4);
analy_filter4_out_up = upsample(analy_filter4_out_down_pc, 4);

Composition_filter1_out = filter(Composition_filter1, 1, analy_filter1_out_up);
Composition_filter2_out = filter(Composition_filter2, 1, analy_filter2_out_up);
Composition_filter3_out = filter(Composition_filter3, 1, analy_filter3_out_up);
Composition_filter4_out = filter(Composition_filter4, 1, analy_filter4_out_up);
out_of_filter_bank = Composition_filter1_out + Composition_filter2_out + Composition_filter3_ou
FT_out_of_filter_bank = (1 / fs) * fftshift(fft(out_of_filter_bank));
f_axis = linspace(-fs / 2, fs / 2, length(FT_out_of_filter_bank));
```

Here finally we plot input signal and output signal after passing through multi-rate filter bank

```
figure('Name', 'Fourier Transform of Continuous Time Signal');
plot(f_axis, abs(FT_out_of_filter_bank), 'LineWidth', 1.5);
xlabel('Frequency (Hz)');
ylabel('Amplitude');
title('Fourier Transform of Continuous Time Signal');
xlim([-1, 1]);

figure('Name', 'output and input of filter bank');
f_axis = linspace(-fs / 2, fs / 2, length(FT_x));
plot(f_axis, abs(FT_x), 'LineWidth', 1.5, 'Color', 'r');
xlabel('Frequency (Hz)');
ylabel('Amplitude');
title('output and input of filter bank');
xlim([-1, 1]);
grid on;
hold on; % Hold the figure
f_axis = linspace(-fs / 2, fs / 2, length(FT_out_of_filter_bank));
plot(f_axis, abs(FT_out_of_filter_bank), 'LineWidth', 1.5, 'Color', 'b');
xlim([-1, 1])
legend('input', 'output'); % Add legend to the figure
```

Thanks for your attention