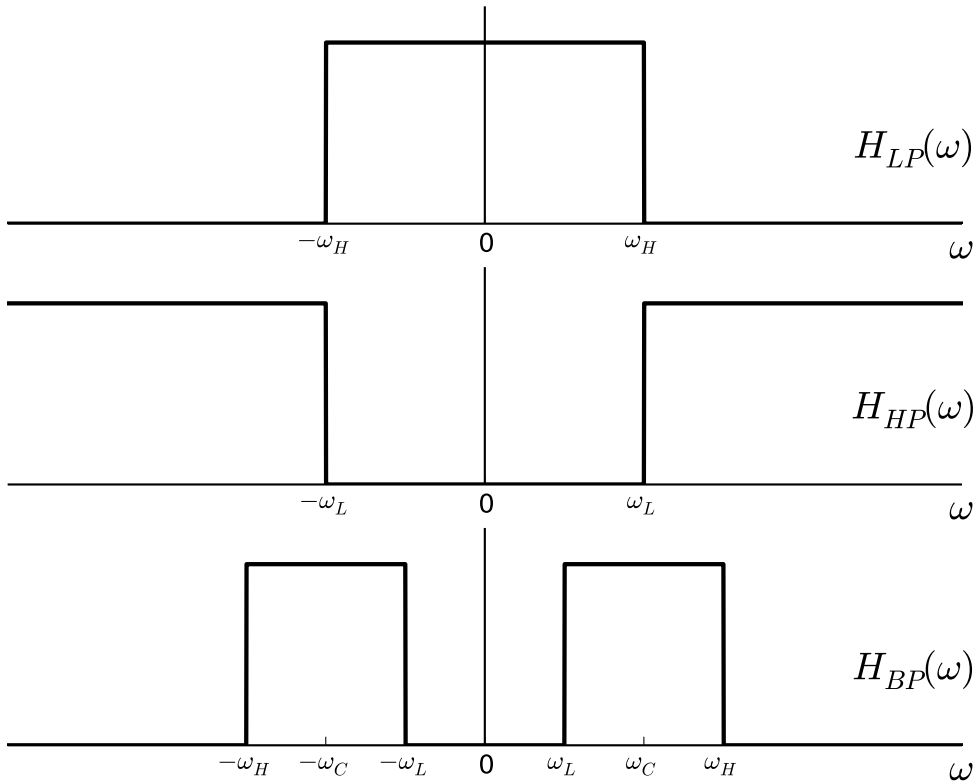
EBU5375 Signals and Systems: Filtering and sampling in Matlab

Dr Jesús Requena Carrión

Queen Mary
University of London

Background: Ideal filters

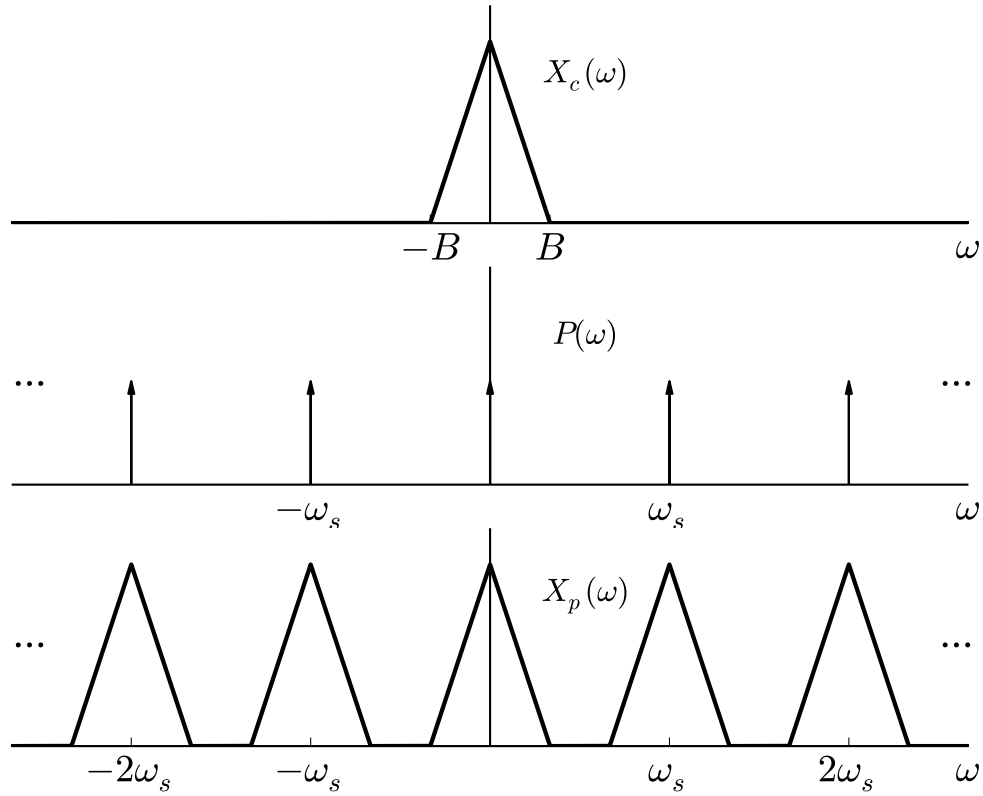
We have learnt three types of filters: lowpass, highpass and bandpass filters. We represent them in the following figure by the frequency responses $H_{LP}(\omega)$, $H_{HP}(\omega)$ and $H_{BP}(\omega)$:



Today, we will use Matlab to filter an audio signal.

Background: Sampling

We have also learnt that the Fourier transform of a sampled signal $X_p(\omega)$ consists of replicas of the Fourier transform of the original signal $X_c(\omega)$.



Today, we will use Matlab to explore sampling.

Objectives of the lab

In this lab, we will explore the signal processing methods of filtering and interpolation.

We will listen to different versions of a familiar audio signal and look at its spectrum (i.e. its frequency domain representation). Specifically, we will analyse the effects of:

1. Using the wrong sampling frequency during reproduction.
2. Lowpass filtering.
3. Highpass filtering.
4. Sampling.

Step 1: The original audio signal

The following fragment of Matlab code loads the file NewYork.mat, plots the audio signal in the time and frequency domain and reproduces it.

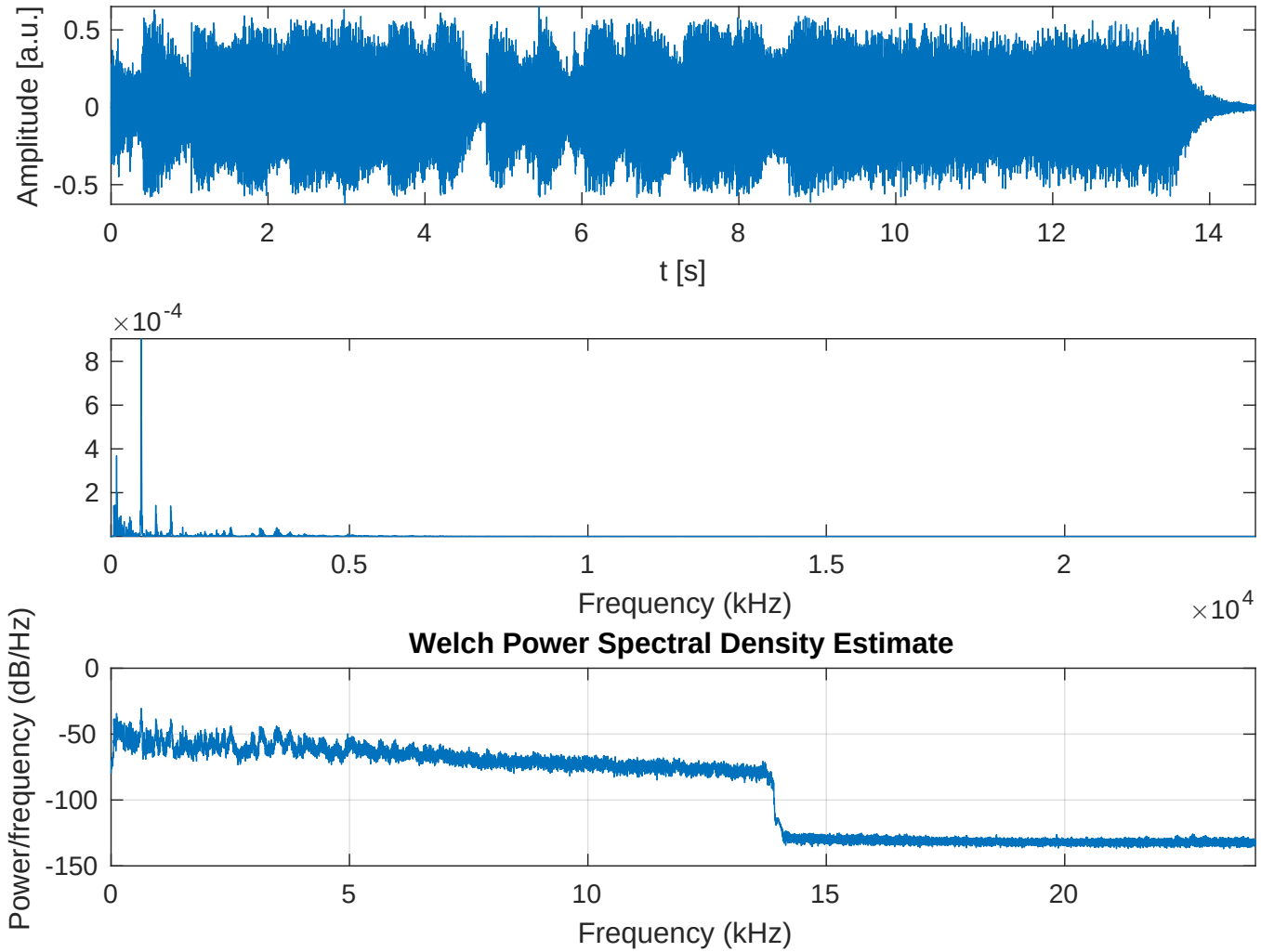
```
load NewYork % Loads the audio signal x
sound(x,Fs,BITS) % Reproduces audio signal

ts=1/Fs; % Fs: Sampling frequency, ts: sampling period
t=[0:ts:(length(x)-1)*ts]; % Time axis
subplot(3,1,1)
plot(t,x) % Plot signal in the time domain
ylabel('Amplitude [a.u.]')
xlabel('t [s]')
axis tight

subplot(3,1,2)
[Pxx,F]=pwelch(x,[],[],[],Fs,'onesided');
plot(F,Pxx) % Plots spectrum
xlabel('Frequency (kHz)')
axis tight

subplot(3,1,3)
pwelch(x,[],[],[],Fs); % Plots spectrum, dB
```

Step 1: The original audio signal



Step 1: The original audio signal

Your job now is to:

- Identify the sampling frequency, the number of samples of the audio signal, the duration of the audio and the number of bits per sample.
- Reproduce the audio signal and plot its spectrum substituting `Fs` by `4*Fs`. What do you observe?
- Reproduce the audio signal and plot its spectrum substituting `Fs` by `Fs/2`. What do you observe?

Step 2: Lowpass filtering the audio signal

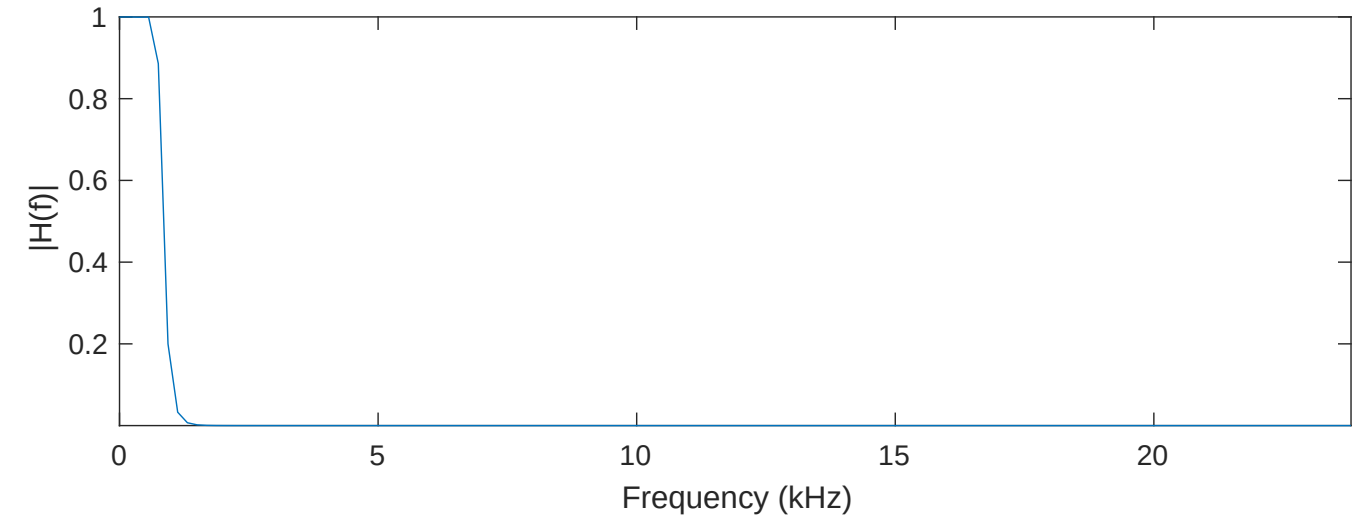
The following fragment of Matlab code creates a lowpass filter and filters the audios signal.

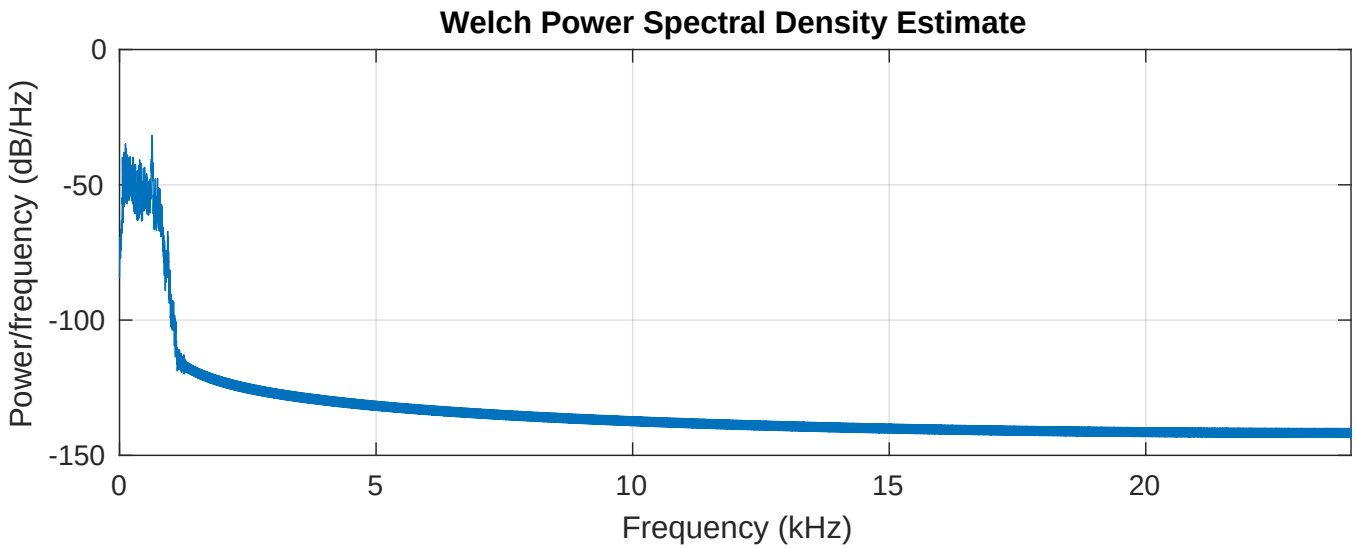
```
figure
[b,a]=butter(10,0.8/24,'low'); % Defines lowpass filter
[H,W] = freqz(b,a,128,Fs);
subplot(2,1,1)
plot(W/1000,abs(H)) % Plots lowpass filter frequency response
ylabel('|H(f)|')
xlabel('Frequency (kHz)')
axis tight
x_fpb = filtfilt(b,a,x); % Lowpass filters audio signal
subplot(2,1,2)
pwelch(x_fpb,[],[],[],Fs); % Plots spectrum of filtered signal
sound(x_fpb,Fs,BITS)
```

Your job now is to execute the above code and:

- Analyse the frequency response of the lowpass filter.
- Analyse the audio signal in the frequency domain and listen to it.

Step 2: Lowpass filtering the audio signal





Step 3: Highpass filtering the audio signal

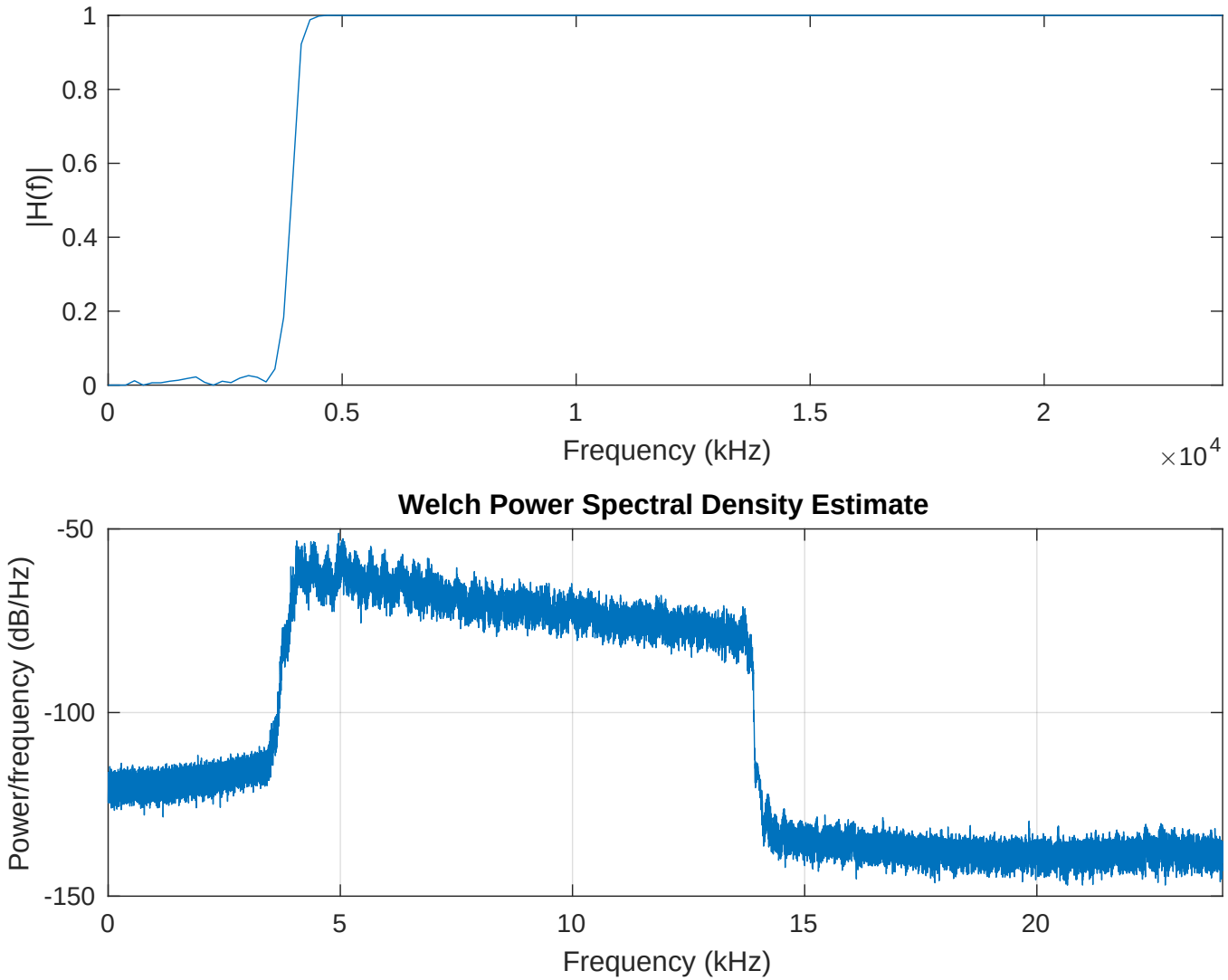
The following fragment of Matlab code creates a lowpass filter and filters the audios signal.

```
figure
[b,a]=butter(25,4/24,'high');% Defines highpass filter
[H,W] = freqz(b,a,128,Fs);
subplot(2,1,1)
plot(W,abs(H))% Plots highpass filter frequency response
ylabel('|H(f)|')
xlabel('Frequency (kHz)')
axis tight
x_fpa = filtfilt(b,a,x); % Highpass filters audio signal
subplot(2,1,2)
pwelch(x_fpa,[],[],[],Fs); % Plots spectrum of filtered signal
sound(x_fpa,Fs,BITS)
```

Your job now is to execute the above code and:

- Analyse the frequency response of the lowpass filter.
- Analyse the audio signal in the frequency domain and listen to it.

Step 3: Highpass filtering the audio signal



Step 4: Sampling the audio signal

The following lines of code sample the original audio signal.

```
[F,Pxx]=pwelch(x,[],[],[],Fs); % Plots spectrum of x
subplot(2,1,1)
plot(Pxx,F) % Plot signal in the time domain
ylabel('Original')
xlabel('Frequency (kHz)')
axis tight
x_s=zeros(size(x)); % Samples original signal x
x_s(1:6:end)=x(1:6:end);
subplot(2,1,2)
[F,Pxx]=pwelch(x_s,[],[],[],Fs); % Plots spectrum of x_s
plot(Pxx,F)
ylabel('Sampled')
xlabel('Frequency (kHz)')
axis tight
sound(x_s,Fs,BITS) % Reproduces audio signal
```

Your job now is to

- Compare the spectra of both original and sampled signals.
- Analyse the audio characteristics of the sampled signals.

Step 4: Sampling the audio signal