9.35 Sensation And Perception
Spring 2009

Pset 6

Answer Key

Contents

- Init
- Problem 1
- Problem 2
- Problem 3
- 3C
- Problem 4
- Problem 5

Init

```
clear all
close all
clc
drawnow
```

Problem 1

A) N_samples = Samples_per_sec / cycles_per_sec 40,000 Hz (2 samples/cycle)

B) See function mySpectrogram

Problem 2

A) 'Illumination' is analogous to the source of a sound, since these are both the source of light/sound energy. 'Reflectance' is just the environment that the sound interacts with - in fact, the sound will reflect off all the various surfaces it comes into contact with. Although these are very natural counterparts, they play opposite roles in terms of what is perceptually important: in vision, reflectance of an object is what is important, but in audition, it is the source of the sound (and not the environment which reflects the sound) which we care about.

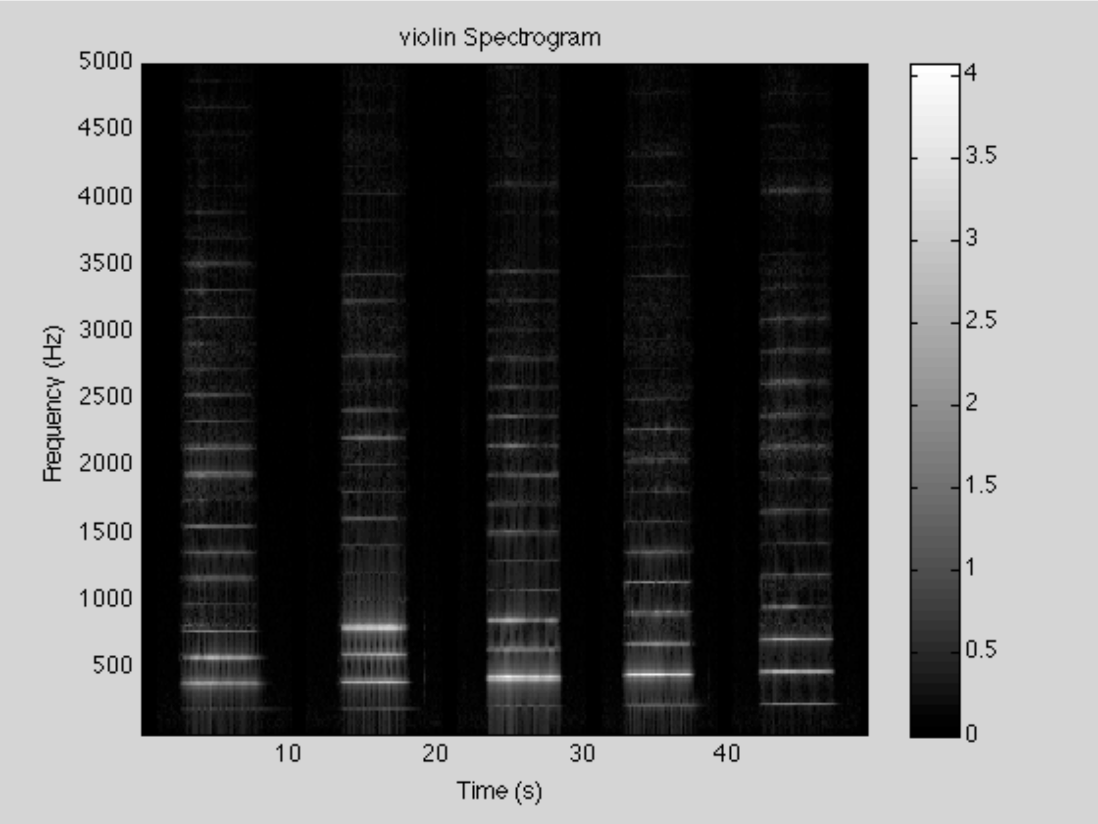
B) Our cochlea measures frequency of vibrations -- since the speed of sound varies substantially across different media, we are actually not sensitive to the wavelength of sound: only its frequency (with the exception of selectively filtering particular wavelengths given the shape/size of our head, but that is a rather small effect compared to the rest of the hearing organ).

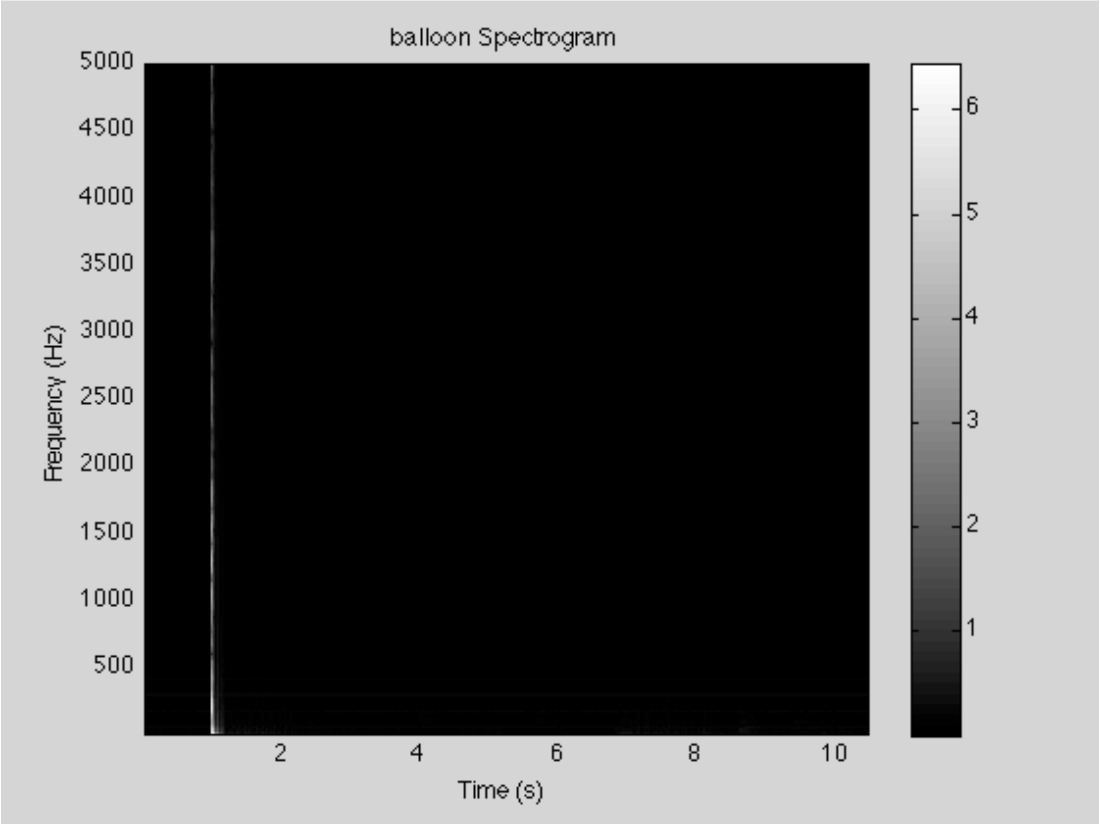
The frequency of a photon or sound wave is a fundamental propert that determines the color or pitch of our perception. Our visual system behaves quite differently from our auditory system when measuring the frequency. In vision, we have three broadly tuned sensors (L, M and S cones) which effectively triangulate the frequency to yield a color percept. Because of the broad tuning, there can be several unique frequency combinations which yield identical activation across the three types, and these combinations are called metamers. However, sound frequencies are detected in many small bands, since hair cell frequencies are tuned across the length of the cochlea, and we effectively have a continuous representation of sound frequencies. So, any frequency will only stimulate a small number of hair cells, and creating a sound 'metamer' out of different combinations of frequencies would be impossible.

Problem 3

```
sounds = {'violin', 'balloon'};
for s = 1:length(sounds)
    % Load sound
    [y Fs] = wavread([sounds{s} '.wav']);

    % Create spectrogram
    figure
    mySpectrogram(y, Fs, true);
    colormap gray
    title([sounds{s} ' Spectrogram']);
    drawnow
end
```





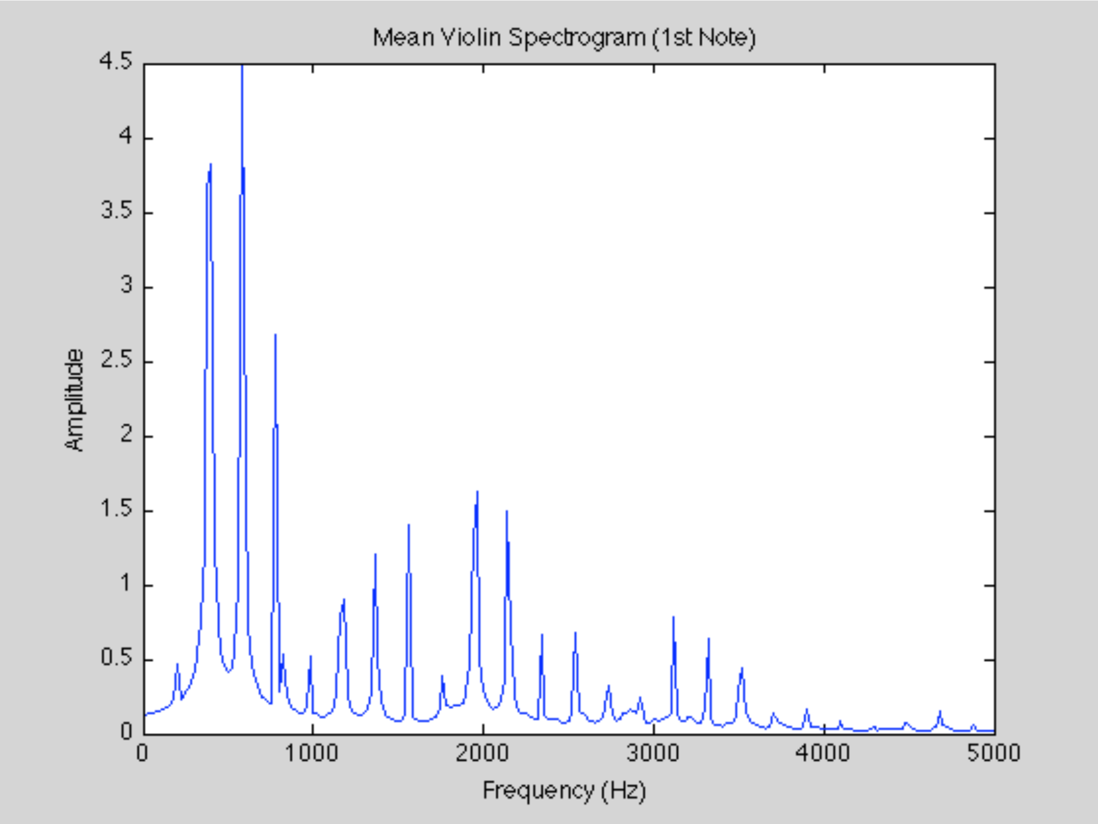
A) In the violin spectrogram, you can very clearly see the harmonic structure. The columnar structure, of course, simply shows the temporal aspects of the sound - several long, prolonged strums on the violin. The fact that each set of bars is about even in amplitude over time means that the strum is relatively steady, and the fact that it is even in pitch means that it is a single note. It is harder to see this, but the fundamental harmonic is slightly higher in pitch in each note, meaning the violin is playing up some kind of scale. The balloon spectrogram simply looks like a delta function - an fast burst of energy in one small time point, containing energy across all frequencies (though decaying as the frequencies get higher). This is also known as broad-band noise.

B) Excluding the loss of information created by clipping the spectrogram and jumping across time points, the only information lacking is the phase of each frequency component. Just like in an image where we can scramble the phase to create a cloudy image which preserves the amplitude of each component, two sounds can have identical frequency amplitudes, but the phase dictates the structure of the sound. However, auditory neurons are typically much less sensitive to phase than visual neurons, so depending on the range of frequencies, a phase scramble may or may not change the percept of the sound! That is, a scrambled image is completely unrecognizeable, but a scrambled auditory signal may contain some recognizeable elements, especially if you preserve the temporal aspects of the signal (i.e. all frequencies < 20 Hz). I encourage you to play with this in Matlab if you are interested.

3C

First, extract a single strum of the violin

```
[y Fs] = wavread('violin.wav');
figure
[spect time freq] = mySpectrogram(y, Fs);
close(gcf);
violinSpect = spect(:, time>3 & time<7);    % A single note
violinMeanSpect = mean(violinSpect, 2);
figure
plot(freq, violinMeanSpect);
xlabel('Frequency (Hz)');
ylabel('Amplitude');
title('Mean Violin Spectrogram (1st Note)')
```



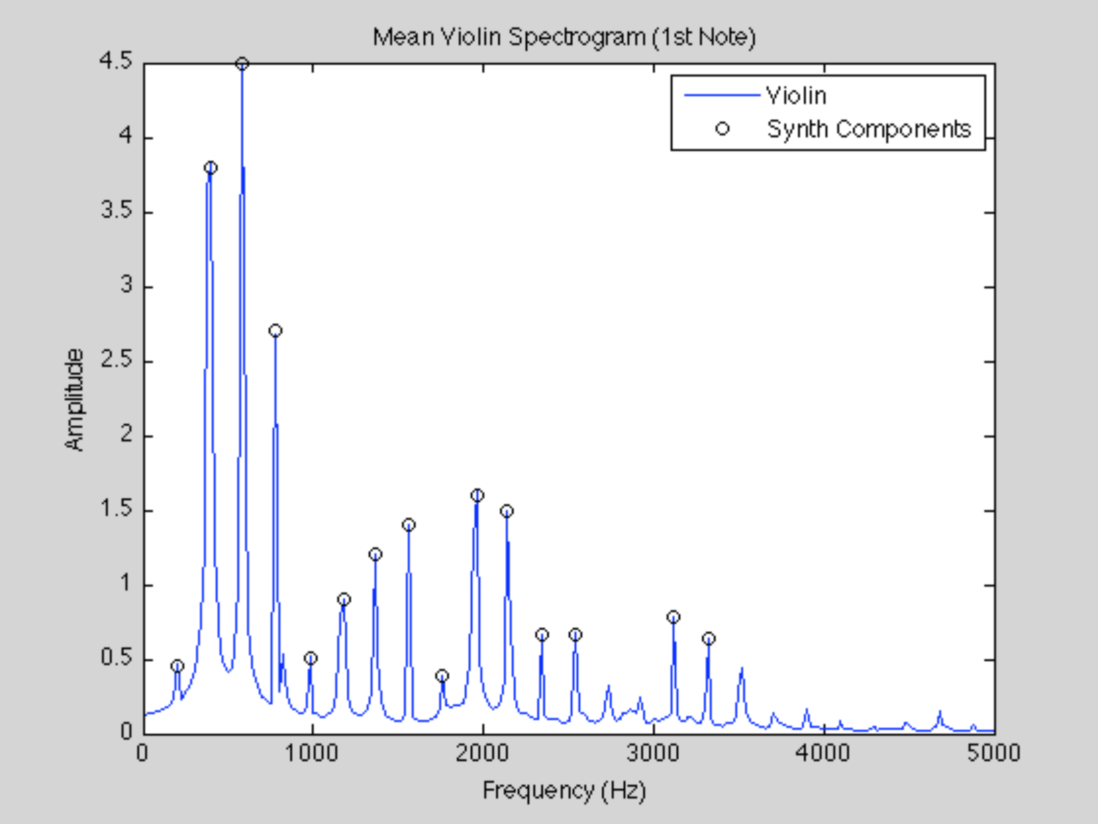
Pick the best components (done by hand here), synthesize

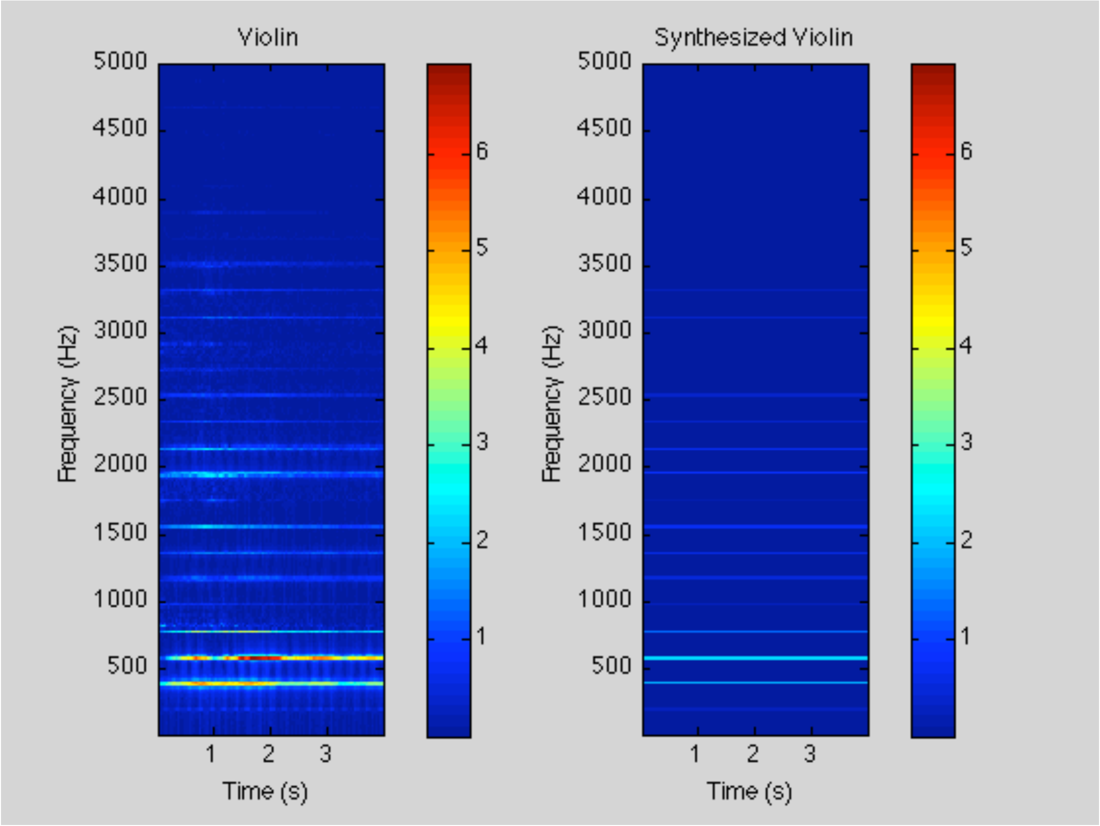
```
peak_a = [  .46 3.8 4.5 2.7 .51 .90     1.2     1.4     0.39    1.6 ...
            1.49    0.66    0.67    0.78    0.64];
peak_f = [  200 400 580 780 980 1180    1360    1560    1760    1960 ...
            2140    2340    2540    3120    3320];
hold on
scatter(peak_f, peak_a, 'ko');  % Plots on spect
hold off
legend('Violin', 'Synth Components', 'Location', 'NorthEast')

% Convert FFT mag to signal amplitude
peak_a = peak_a/round(Fs/20);

% Synthesizes sound and compares to original
```

```
violin = y(Fs*3:Fs*7);
time = 0:(1/Fs):4;
synth = peak_a * sin(2*pi*peak_f'*time);
figure
subplot(1,2,1);
mySpectrogram(violin, Fs);
title('Violin')
clim = get(gca, 'CLim');
subplot(1,2,2);
mySpectrogram(synth, Fs);
title('Synthesized Violin')
set(gca, 'CLim', clim);
```



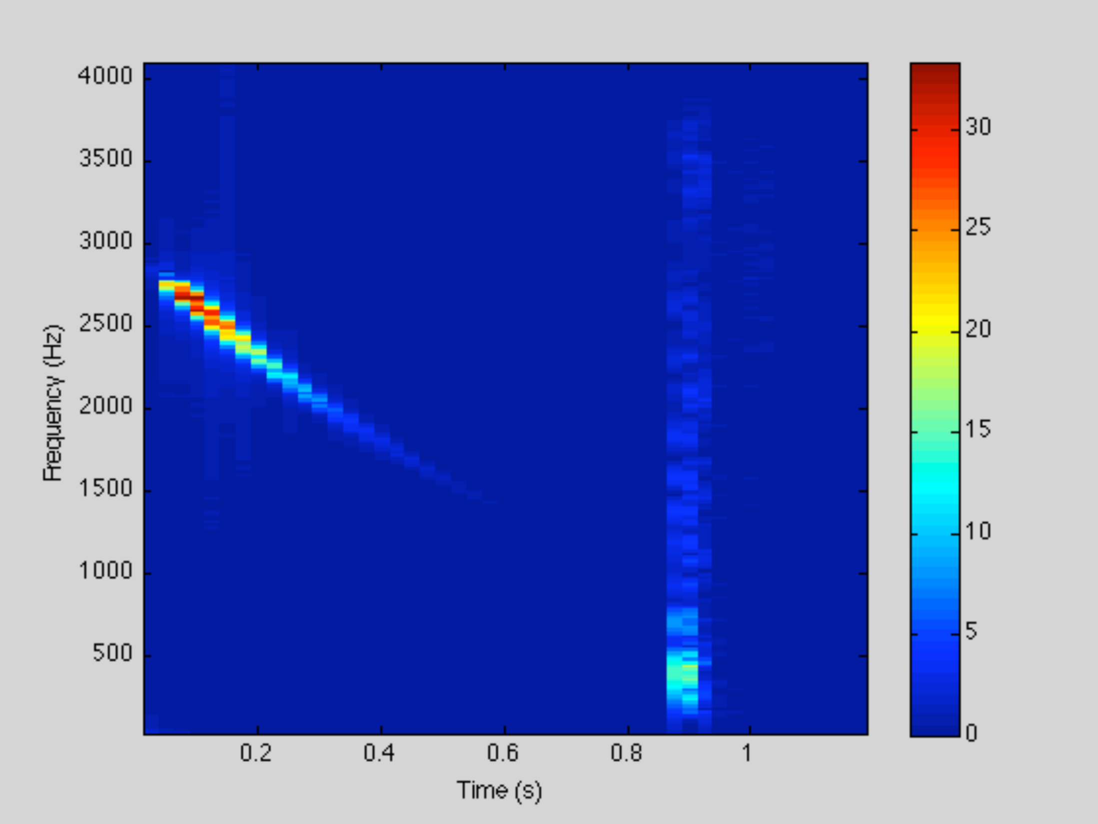


This synthesis sounds somewhat similar to the violin, but it's not very good! I needed to add about 15 components to get a decent-sounding violin. Taking components away definitely makes it worse.

Problem 4

This is a built-in Matlab sound. See if you can't guess what it sounds like! The spectrogram speaks for itself.

```
load splat
figure
mySpectrogram(y, Fs);
```

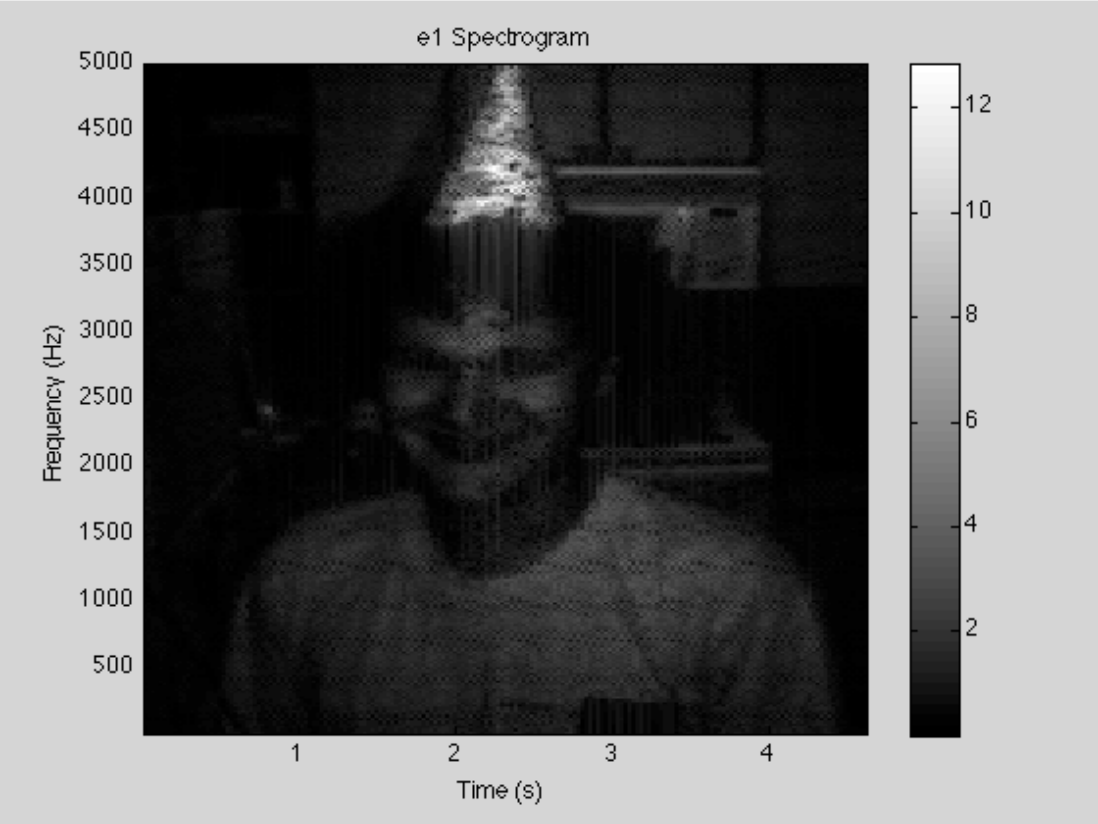


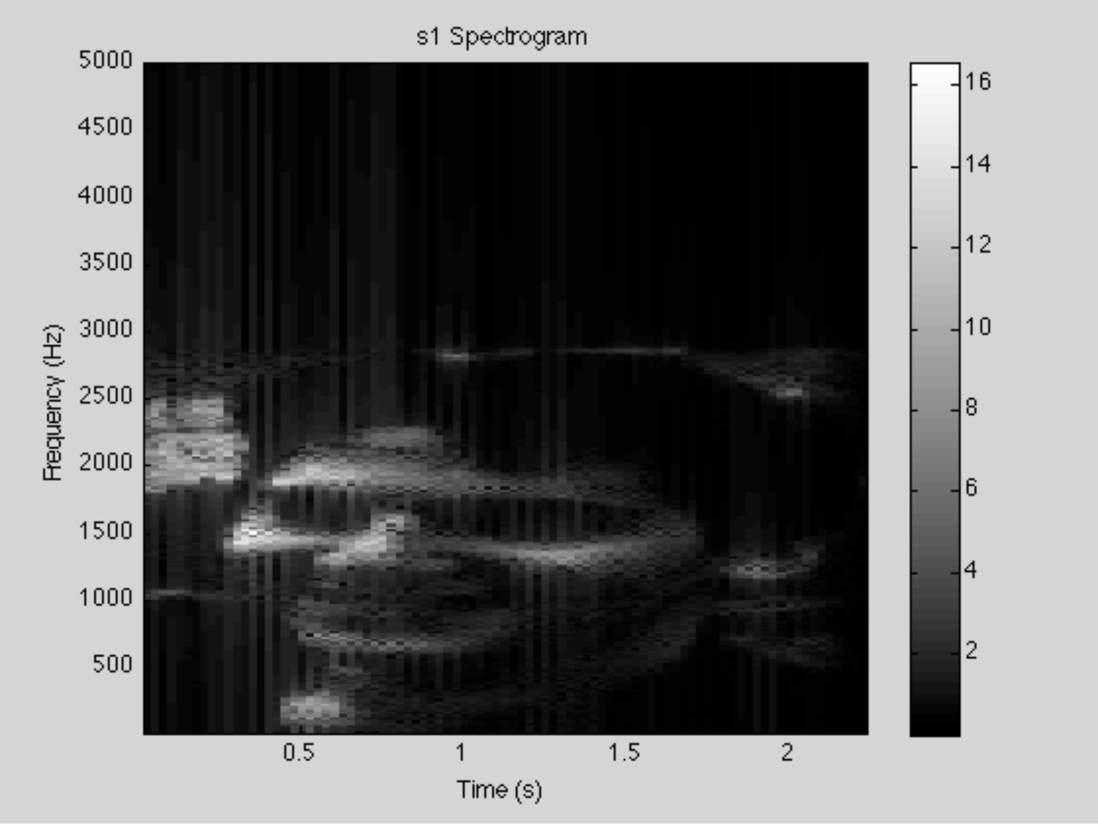
Problem 5

Both sounds certainly make for a weird auditory experience, but the spectrograms reveal just how much more handsome one of those files really is.

```
sounds = {'e1', 's1'};
for s = 1:length(sounds)
    % Load sound
    [y Fs] = wavread([sounds{s} '.wav']);

    % Create spectrogram
    figure
    mySpectrogram(y, Fs, true);
    colormap gray
    title([sounds{s} ' Spectrogram']);
    drawnow
end
```





Published with MATLAB® 7.4