

University of Central Florida  
School of Electrical Engineering and Computer Science  
EGN-3420 - Engineering Analysis.  
Fall 2009 - dcm

## Homework 5 due Thursday week 13 (100 points)

Problems 1-10 (10 points each)

Solve problems: 11.2 (page 261); 11.10 (page 262); 12.6 (page 279); 12.10 (page 280); 13.3 (page 312); 13.24 (page 314); 14.2 (page 332); 15.10 (page 357); 16.3 (page 386); 16.6 (page 387);

**Extra credit problem** (10 points for boundary grade considerations)

### 1. Display an audio waveform and its spectrum

Audio signals, much like images, can undergo filtering. It is somewhat easier to understand the impact of signal processing on audio, since audio needs not be translated from a spatial to a frequency domain.

To load a wave (PCM) audio file, Matlab provides the function *wavread*. For example:

```
kiki = wavread('kiki.wav');
```

To capture the sampling frequency at which the sound was recorded, use:

```
[kiki, f] = wavread('kiki.wav');
```

otherwise the speed of playback and results of further processing is not guaranteed to be correct.

To play a wave file at sampling frequency  $f$ :

```
wavplay(kiki, f);
```

To view the waveform, plot the wave. Since audio is represented with many thousand samples per second, it may be required to plot small portions of the waveform at a time.

```
subplot(2,1,1), plot(kiki), title('Entire waveform');
```

```
smallRange = 100000:100000+floor(f/100);
```

```
subplot(2,1,2), plot(smallRange, kiki(smallRange)), title('100 milliseconds');
```

Plot the original function and use the *specgram* built-in function to see the spectrum to find a correspondence between the original waveform and the spectrum.

### 2. Implement a low pass filter

Matlab includes function *butter* for building Butterworth low-, high-, and stop-pass filters. Low-pass filters remove frequencies greater than some specified value; high-pass filters remove frequencies lower than some specified value. stop-band filters remove frequencies in a given range of values.

Frequencies values are specified in normalized terms between 0.0 and 1.0, where 1.0 corresponds to half the sampling frequency:  $f/2$ . A given frequency is thus expressed in terms of this value, for example,  $1000\text{Hz} = 1000/(f/2)$ .

Filters are described in terms of 2 vectors ( $[b, a] = [\text{numerator}, \text{denominator}]$ ).

To apply a filter to a 1-D audio waveform, Matlab provides function *filtfilt*, which takes as arguments the result  $[b, a]$  from *butter*, the waveform, and a value denoting the order (number of coefficients) of the filter.

A filter's frequency response can be plotted using function *freqz*. Magnitude values at zero dB are unaffected by the filter. Magnitude values below 0 dB are suppressed.

Design a 10th order low-pass filter to suppress frequencies higher than 400Hz