BIOL 708L/NACS 728B Lab #7
Fall 2011

Application of Filters

The main goal of this lab is to use frequency selective filters to clean (i.e. remove noise from) data.
For this lab, assume all data is of length $N = 24$ points. Use n from 0 to N–1, and Fourier transform frequencies f_k from 0 to (N–1)/N in steps of 1/N. For purposes of displaying graphs with continuous frequency, use $N_{graph} = 256$. Ordinarily, noisy data would just be measured data, but for the purposes of this lab we construct our noisy data in two parts: the ideal, normally inaccessible, (non-noisy) data, and the measured (noisy) data.

1) Typical "slowish" ideal data.
   a) Generate the ideal data first
   ```
   xTrueSlow = [zeros(N/8,1); sin(2*pi*[0:N/8-1]'/N*2);ones(N/8,1); ...
      cos(2*pi*[0:N/8-1]'/N*2);-0.25*sin(2*pi*[0:N/8-1]'/N*4);zeros(3*N/8,1)];
   ```
   Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this ideal data?
   b) Generate our first set of "measured" data by adding (broadband) white noise to it:
   ```
   x1 = xTrueSlow + 0.2*randn(N,1);
   ```
   Broadband noise, such as white noise, is typical of the kind of noise found in measured signals. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this "measured" data? How does it differ from the ideal case? Which will help most in cleaning this signal, a low-pass filter, a high-pass filter, or something else?
   c) Generate our second set of "measured" data by adding a narrowband noise (a fast sinusoid) to it:
   ```
   x2 = xTrueSlow + 0.5*cos(2*pi*3/8*n);
   ```
   Narrowband noise, such as 60 Hz line noise, is another kind of noise often found in measured signals. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this "measured" data? How does it differ from the ideal case? Which will help most in cleaning this signal, a low-pass filter, a high-pass filter, or something else?

2) Consider the filter with impulse response $h[n] = \frac{1}{4}\delta[n] + \frac{1}{2}\delta[n-1] + \frac{1}{4}\delta[n-2]$. This is the impulse response of the system made by taking the two-point moving average of the two-point moving average.
   a) Plot the magnitude of its frequency response using `freqz()` with N_{graph} points between 0 and ½ (Note that this is unlike the Fourier Transforms plotted above, which show f from 0 to 1):
   ```
   [H,w] = freqz(h, 1, Ngraph); f = w/(2*pi);
   figure; plot(f, abs(H)); title('|H|'); xlabel('f'); ylabel('magnitude')
   ```
   Is this filter good for cleaning `x1`? Why or why not? `x2`? Why or why not? Could it be made better for either? What is the effective delay of this filter (expressed as number of time points)?
   b) Use the Matlab function `filter()` to clean `x1` with this filter. Graph the data and the magnitude of

its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this cleaned data? How does it differ from the ideal and "measured" cases?

c) Use the Matlab function `filter()` to clean `x2` with this filter. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this cleaned data? How does it differ from the ideal and "measured" cases?

3) Consider the filter with impulse response $h[n] = 0.1609\delta[n] + 0.3391\delta[n-1] + 0.3391\delta[n-2] + 0.1609\delta[n-3]$. This is the impulse response of a filter designed using a Kaiser window (a technique we will cover later).

a) Plot the magnitude of its frequency response using `freqz` () with N_{graph} points between 0 and ½. Is this good for cleaning `x1`? Why or why not? `x2`? Why or why not? Better, or not, than in (2)? How many points is the effective delay of this filter?

b) Use the Matlab function `filter()` to clean `x1` with this filter. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. Is it any better, or not, than in (2)?

c) Use the Matlab function `filter()` to clean `x2` with this filter. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. Is it any better, or not, than in (2)?

4) Typical "fastish" data.

a) Generate the ideal data first

```
xTrueFast = [zeros(N/2,1);1;-.5;.25;zeros(N/2-3,1)];
```

Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this ideal data?

b) Generate our first set of "measured" data by adding slow drift to it:

```
x3 = xTrueFast + 0.5*cos(2*pi*n/N);
```

Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this "measured" data? How does it differ from the ideal case? Which will help most in cleaning this signal, a low-pass filter, a high-pass filter, or something else?

5) Consider the filter with impulse response $h[n] = -\frac{1}{4}\delta[n] + \frac{1}{2}\delta[n-1] - \frac{1}{4}\delta[n-2]$.

a) Plot the magnitude of its frequency response using `freqz` ()with N_{graph} points between 0 and ½. Is this good for cleaning `x3`? Why or why not? Could it be better? How long is the effective delay?

b) Use the Matlab function `filter()` to clean `x3` with this filter. Graph the data and the magnitude of its Fourier transform, as functions of n and f_k respectively. What is the frequency distribution of this cleaned data? How does it differ from the ideal and "measured" cases?

c) Can you demonstrate that the "glitch" at the beginning of the cleaned signal is purely due to initial conditions? Hint: what happens if you apply the filter backwards in time (e.g. reverse the signal before filtering and then reverse the result). What happens to the filter delay if you do this?