

**Engin. 100: Music Signal Processing
Lab #3[2]: Applying Signal Spectra**

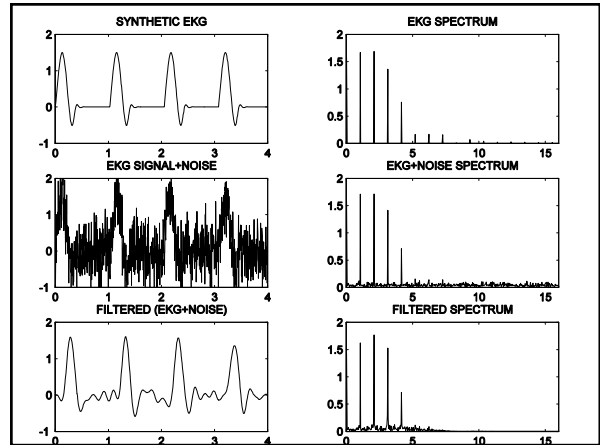
- Filtering noisy signals to remove noise
- Filtering signals to remove interference
- Spectrogram: Time-varying spectrum

Spectral filtering of noisy signals

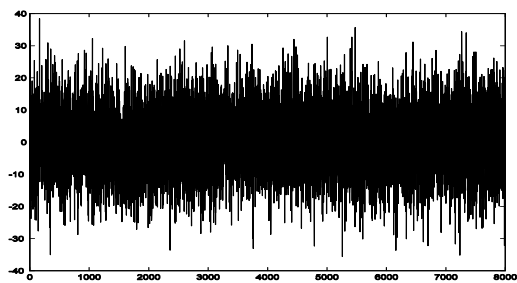
- **GIVEN:** Observe: $y=x+n$ =desired signal+noise.
- **GOAL:** Filter out as much noise n as possible.
- **KNOW:** Signal bandlimited to $f=(K-1)S/N$ Hertz.
- **IDEA:** Eliminate frequencies above F Hertz, since they must be noise. Do this using “fft” as follows:
- $y=x+n; fy=fft(y); fy(K:N+2-K)=0; z=real(iff(y));$
- z is *low-pass filtered*, hence most noise eliminated.
- **BUT:** Can't eliminate noise below F Hertz, since those noise components overlap signal components.

Low-pass filtering a noisy EKG signal

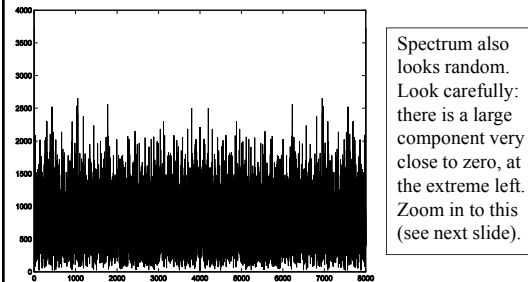
- EKG=Electrocardiogram (heart) electric signal.
- Periodic (we hope!) with period about 1 second (note 60 beats per minute=1 beat per second).
- Components at 1,2,3...Hertz up to about 7 Hertz.
- **So:** We eliminate all frequencies above 7 Hertz.
- **Result:** Most noise gone; not low frequencies.
- See next slide for signals and their spectra.



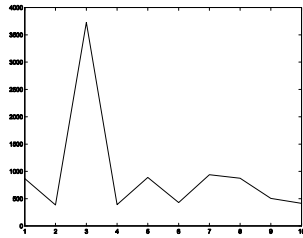
**Noisy Signal (from Lab Lecture #1)
What is the signal buried in noise?
How do we recover the signal?**



**Spectrum of noisy signal: abs(fft(X))
Huh? But look at the vertical scale**

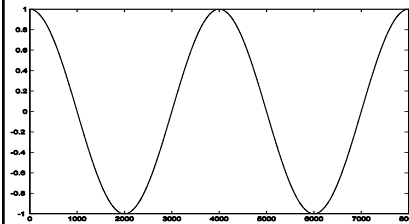


Spectrum of noisy signal: Zoom in.
Now it's clear what the signal is!



There is a large component (peak) at index K=3. This is a sinusoid at the frequency $f=(K-1)S/N=(3-1)8000/8000=4$ Hertz (is right). Next, we filter it.

Filtering Noisy Signal: Set all of the components to zero except K=3,7999



```

Matlab code used
T=[0:7999]/8000;
X=cos(4*pi*T);
N=randn(1,8000);
Y=X+10*N;
F=fft(Y);
F(4:7998)=0;
F([1 2 8000])=0;
Z=real(iff(F));
    
```

NOTE: Need to keep both K=3 and its mirror index K=8000+2-3. Mirror index of K is N+2-K due to Matlab indexing starting at 1. If counting from k=0 to N-1, so $f=kS/N$, the mirror index of k is N-k.

Engin. 100: Music Signal Processing
Lab #3[2]: Applying Signal Spectra

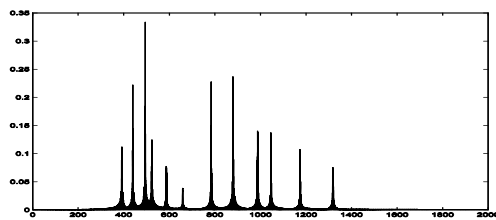
- Filtering noisy signals to remove noise
- Filtering signals to remove interference
- Spectrogram: Time-varying spectrum

Removing Interference from Signals

- Noise: Unknown, unpredictable, but usually dominated by high frequency components.
- Interference: Known signal; want to eliminate. Easy if we have a record of the interference, or if we can generate it. But usually we have neither.
- Listen to this.
- How can we get rid of this *awful* interference?

Removing Interference from Signals

Spectrum of the signal. How can we tell signal from interference? We need something better: a time-varying spectrum of this signal.



```

>>FZ=abs(fft(Z)); FZ=FZ(1:20000)*2/78000; F=[0:19999]*8192/78000;
plot(F,FZ) where Z is the signal. This doesn't help very much, does it?
    
```

Engin. 100: Music Signal Processing
Lab #3[2]: Applying Signal Spectra

- Filtering noisy signals to remove noise
- Filtering signals to remove interference
- Spectrogram: Time-varying spectrum

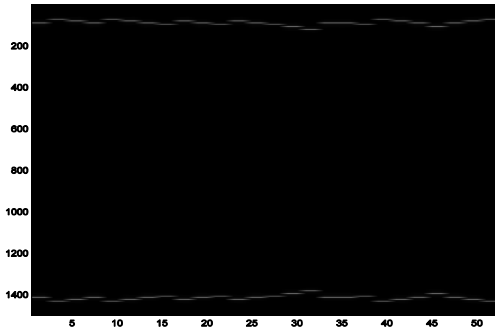
Time-varying spectral analysis

- **IDEA**: Segment (chop up) signal in time.
- **THEN**: Apply “fft” to each signal segment.
- **HOW**: `imagesc(abs(fft(reshape(X',N,L),N)))`
- **WHERE**: $L = \# \text{segments}$ and $N = \text{length}(X)/L$;
- **WHAT**: Computes “fft” of each of L segments.
- **SHOW**: Display freq. vertical, time horizontal.
- **WHY**: Gives a spectrum that varies with time.

Example: “The Victors” spectrogram

- $X = \text{Tonal “The Victors”}$; sampled at 8192 Hertz.
- Has 26 notes of length $3000/8192$ seconds each.
- $\text{Length}(X) = 78000 = 26(3000)$. So $L = 26$, $N = 3000$.
- `>>imagesc(abs(fft(reshape(X',3000,26))))`,
`colormap(gray)`. This is shown on the next slide.
- This is called the “**spectrogram**” of X .
- You can *see* that the signal is a single sinusoid whose frequency *jumps* every $3000/8192$ seconds.
- **Much** more information than just the spectrum!

Example: “The Victors: spectrogram



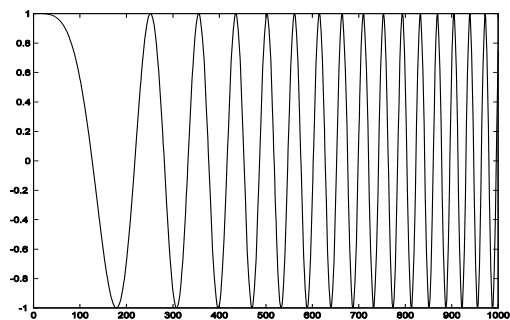
Example: “The Victors: spectrogram

- **Frequency** actually displayed from top down, but due to “fft” mirror also displayed from down up.
- **Time** displayed as increasing from left to right.
- Vertical **slices** are the spectra at different times.
- Horizontal **slices** are the presence/absence of a specific frequency as time varies.
- **Brightness** indicates strength at that time-freq.

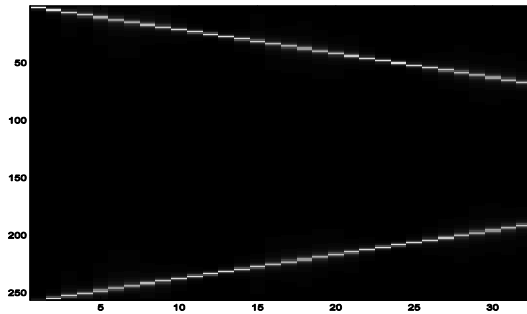
Example: chirp signal

- **Chirp**: $x(t) = \cos(2\pi Ft^2)$: birds, dolphins.
- **Frequency** increases linearly with time.
- **Instantaneous frequency** $= 2Ft$ (not Ft) Hertz.
- `>>X=cos([0:8191].^2/10000);plot(X(1:1000))`
- `>>imagesc(abs(fft(reshape(X',256,32))))`
- `>>colormap(gray)` shown on next 2 slides.

Chirp signal: time waveform



Chirp signal: spectrogram

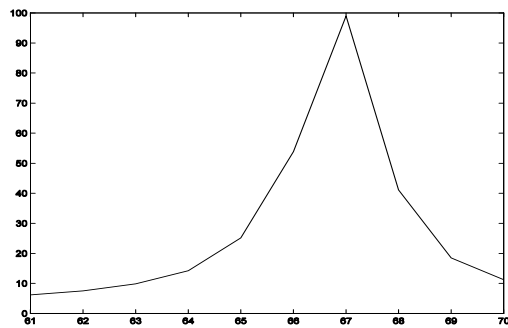


Chirp: Instantaneous frequency

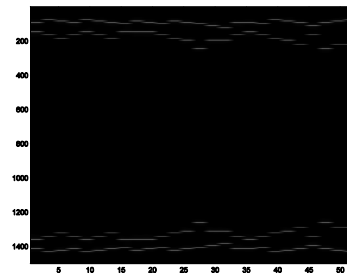
$\gg X = \cos([0:8191].^2/10000)$ means this:

- $x(t) = \cos(t^2)$ sampled at $t = n/100$; duration = 81.9
- The “Instantaneous frequency” = $(2t)/(2\pi)$ Hertz.
- Increases from 0 to $2(81.9)/(2\pi) = 26.08$ Hertz.
- Interpret spectrogram: $F=100$; $N=256$; $T=2.56$
- Freq. in final window: $(67-1)100/256 = 25.8$ Hz. This is average of freqs in final time window.

Vertical slice of final (32nd) time window



Removing Interference, Continued: Spectrogram of signal+interference. Now we can see what's happening.



Now you can see what you need to do to eliminate the interference. The rest is up to you in Lab.

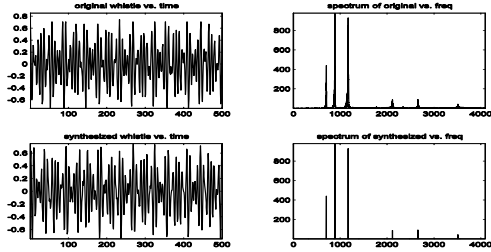
Conclusion

- Can filter out noise from a noisy signal y using: $fy = \text{fft}(y)$; $fy(K:N+2-K) = 0$; $z = \text{real}(\text{ifft}(fy))$; where the signal has no components above index K .
- Noise tends to be high-frequency, so lowpass filter
- To remove interference, time-varying spectrum, computed as spectrogram, can give a much better picture (visualization!) of what is happening.
- Spectrogram can also aid in interpreting signal.

MORE EXAMPLES OF INTERPRETING RESULTS OF MATLAB'S "fft"

Not presented in lecture unless time.

Matlab train whistle spectrum: Approximate using 6 sinusoids



Matlab's train whistle signal

- Signal duration=12880 samples=1.57 sec. Take *periodic extension* (repeats; T=1.57).
- Sampling rate=8192 Hertz. So N=ST=12880.
- `>>load train.mat;length(y)[=12880];plot(y); F=fft(y);plot(abs(F(1:12880/2)))` [1st half only]
- F(k) is component at frequency $f=(k-1)S/N$. $f=(\text{index}-1)(8192/12880)=(\text{index}-1)/1.57$ Hertz.
- $[2/N*\text{abs}(F) \text{ angle}(F)]$ gives numerical values.

Matlab train whistle spectrum

- Only about 6 significant sinusoids present.
- Closer examination of spectrum: Peaks at:

Index	1109	1394	1840	3326	4180	5521
Hertz	705	886	1170	2115	2658	3511

$$440\cos(2\pi 705t-1.70)+979\cos(2\pi 886t+2.93)+928\cos(2\pi 1170t+1.35)+88\cos(2\pi 2115t+1.41)+93\cos(2\pi 2658t+0.84)+43\cos(2\pi 3511t+3.03)$$

Interpreting results of Matlab's "fft"

- Data acquisition system samples@1024 Hertz. Results loaded into Matlab, in a vector X.
- **GOAL:** Determine spectrum of loaded data.
- Step #1: `>>length(X)` gives 3072. Means what?
- Step #1: (3072 samples)/(1024 samples/second). Duration=3 seconds. Take periodic extension.

Interpreting results of Matlab's "fft"

- Step #2: `>>F=(2/3072)*abs(fft(X,3072));`
- F=0 except indices 97,193,289,2785,2881,2977. F=7 at those indices. How do we interpret this?
- Step #2: Last 3 nonzero mirror images of 1st 3: 2977=3074-97;2881=3074-193;2785=3074-289.
- Three sinusoidal components at frequencies determined by table on next slide.

Interpreting results of Matlab's "fft"

K=Matlab indices of nonzero values of F	97	193	289
Hertz=(K-1)1024/3072=(K-1)/T (N=FT)	32	64	96

$$\text{Data}(t)=7\cos(2\pi 32t+\theta_1)+7\cos(2\pi 64t+\theta_2)+7\cos(2\pi 96t+\theta_3).$$

Phase θ_1 : `>>angle(F(97))`. θ_2 : `>>angle(F(193))`. θ_3 : `>>angle(F(289))`.

Harmonic frequencies: Data(t) periodic with period=1/32 second.
Don't confuse this with periodic extension of Data(t) (3 seconds).