# Analyses of Rhythmic Data

Society for Neuroscience
Short Course #2: *Rhythms of the Neocortex*

Mark Kramer
Department of Mathematics and Statistics
Boston University

# Outline

- Rhythms in health, sickness, *in vivo*, and *in vitro*.
- Motivates the study of rhythms, particularly . . .
  - Quantification of rhythms in data
  - Quantification of rhythmic interactions
- An introduction . . .
- Hands on MATLAB examples
- Discussion: future issues & questions

# Get the data

- Download example data:

# Analyses of Rhythmic Data

http://makramer.info/sfn

## Society for Neuroscience Short Course #2: Supplementary Material

**Data**
Download the data set: data.mat
Load this data set in MATLAB using the *load* command.

We'll consider three examples that make use of different variables:
**Example 1** uses: v1, t1
**Example 2** uses: v2, t2
**Example 3** uses: v3a, v3b, t3

**MATLAB code**
Download an M-file that includes MATLAB code to analyze these data: analyze_data.m

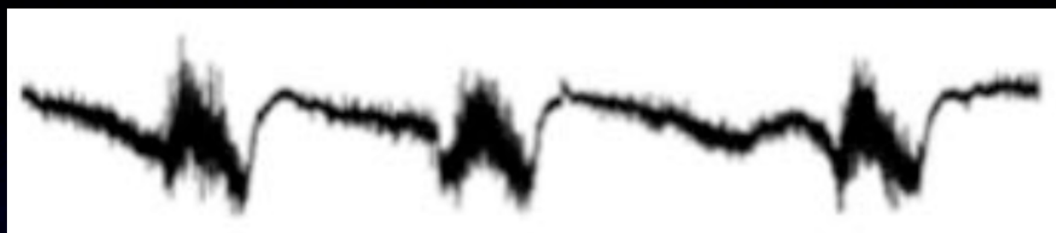
**Tutorial slides** : As a PDF.

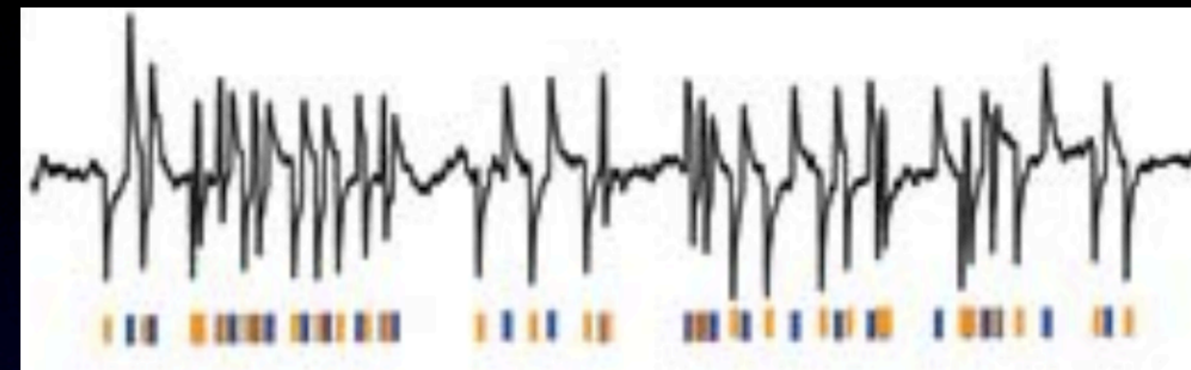
**Software Links**
Chronux
EEGLab

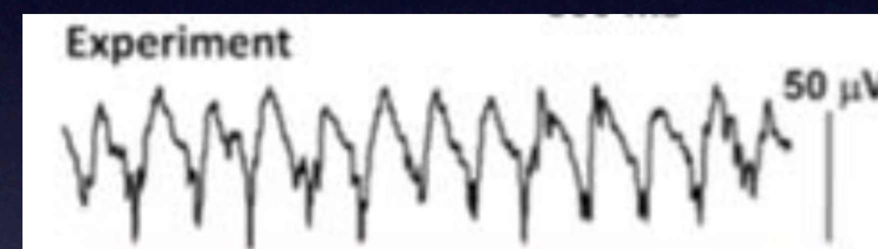
**Book Links**
Numerical Recipes in C (online)
Matlab for Neuroscientists
Signal Processing for Neuroscientists
Observed Brain Dynamics

**Contact:** Email Mark Kramer

# Motivating questions





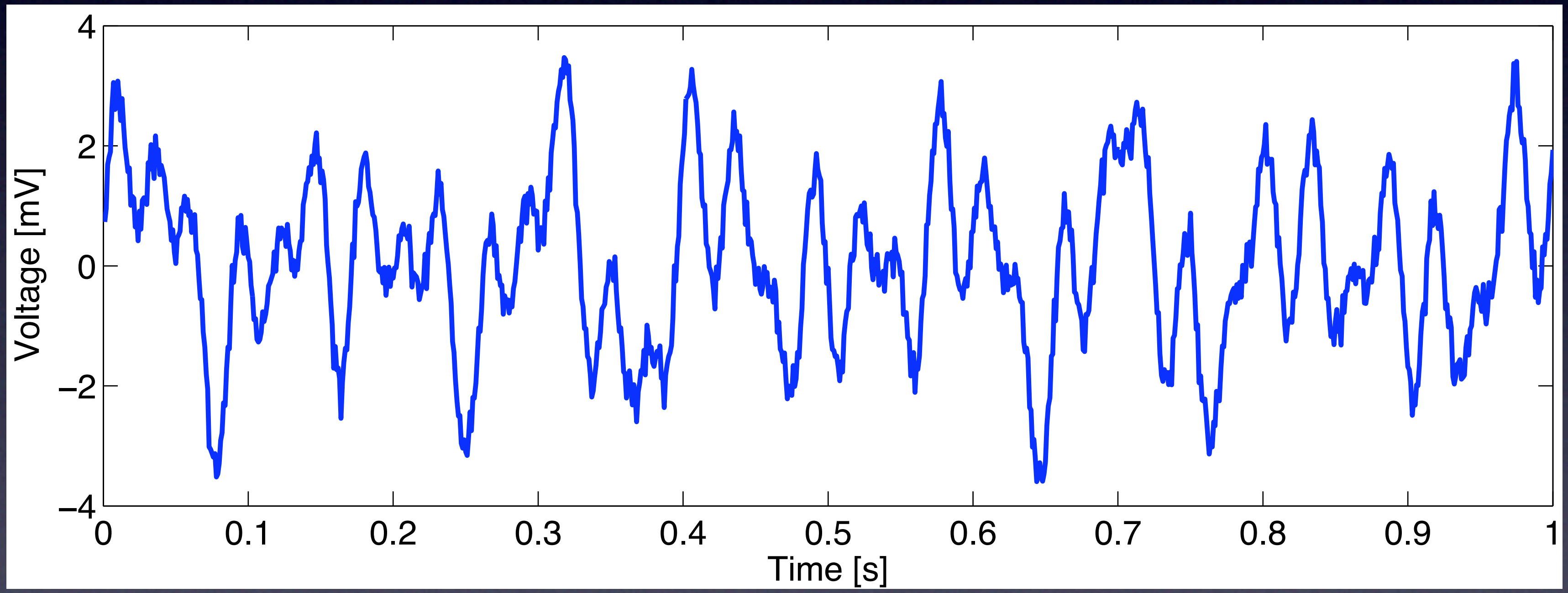




- How can we quantify rhythms in data?
  - Power spectrum
- How can we quantify coupling between rhythms?
  - Coherence

# Load data & visualize

Download example data: http://makramer.info/sfn

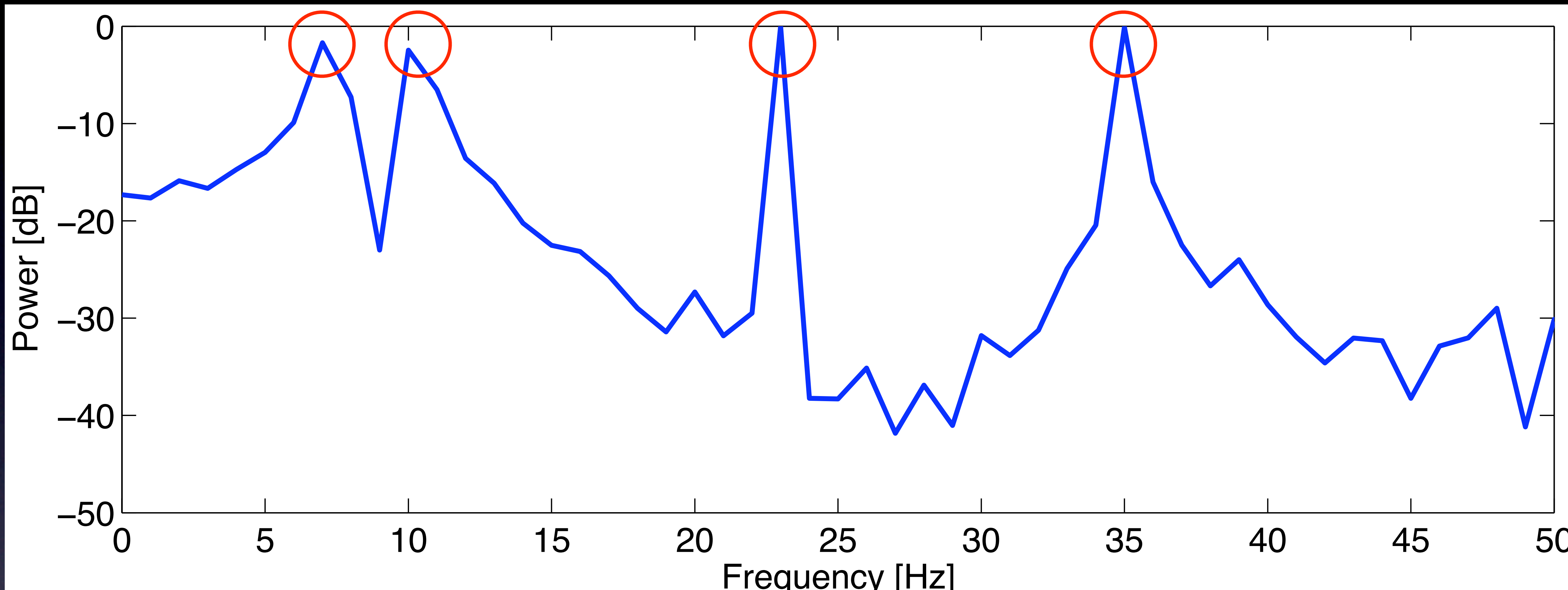
```
>> load data.mat
>> plot(t1,v1)
```




dt = 1ms

- Rhythmic
- It's complicated
- How can we simplify?

# Power spectrum




- Axes: Power [dB] vs Frequency [Hz]
- A simpler representation in frequency domain. Sum of four sinusoids at {7, 10, 23, 35} Hz
- How do we compute it?

# Formula

$$V[f] = \int_{-\infty}^{\infty} \underbrace{\boxed{v[t]}}_{\text{Data}} \underbrace{\boxed{e^{-2\pi i f t}}}_{\text{Sinusoids}} dt$$

Fourier transform

$$P[f] = \frac{|V[f]|^2}{n}$$

Power (per unit time)
n = Length of v[t]

- Idea: Write v[t] as sum of sines and cosines oscillating at different frequencies.
- Nice properties (we'll skip)
  *[MATLAB for Neuroscientists, Numerical Recipes in C]*
- In MATLAB . . .

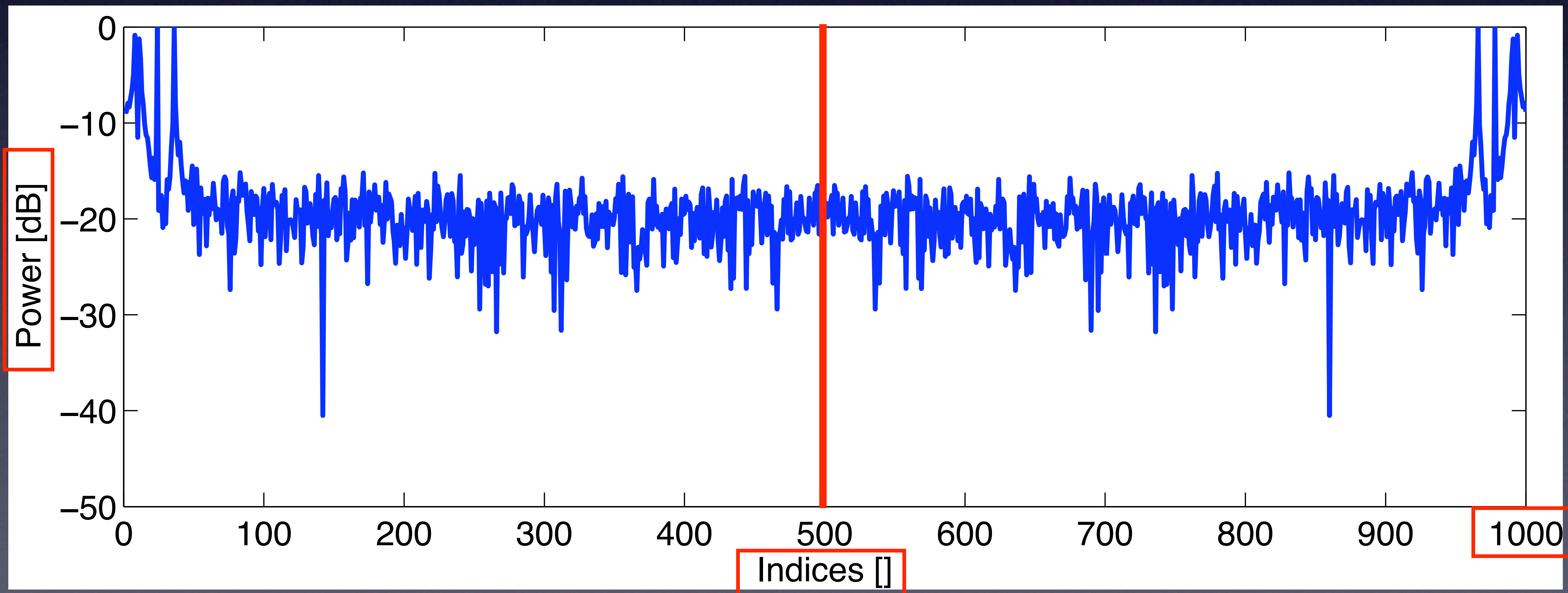
# MATLAB code

http://makramer.info/sfn

1000 data pts

```
>> pow = (abs(fft(v1)).^2)/length(v1);
>> pow = 10*log10(pow/max(pow));
>> plot(pow)
```

Clue?

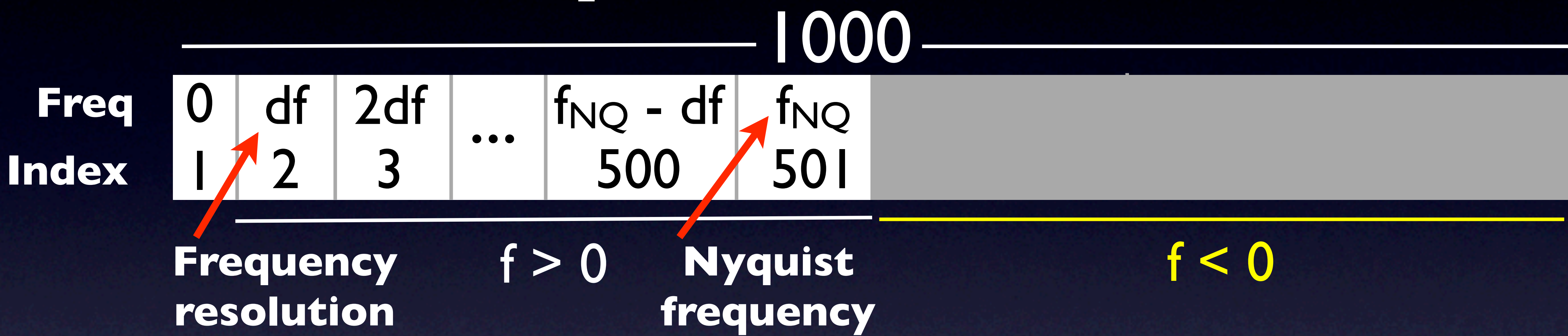



Matches length of v1

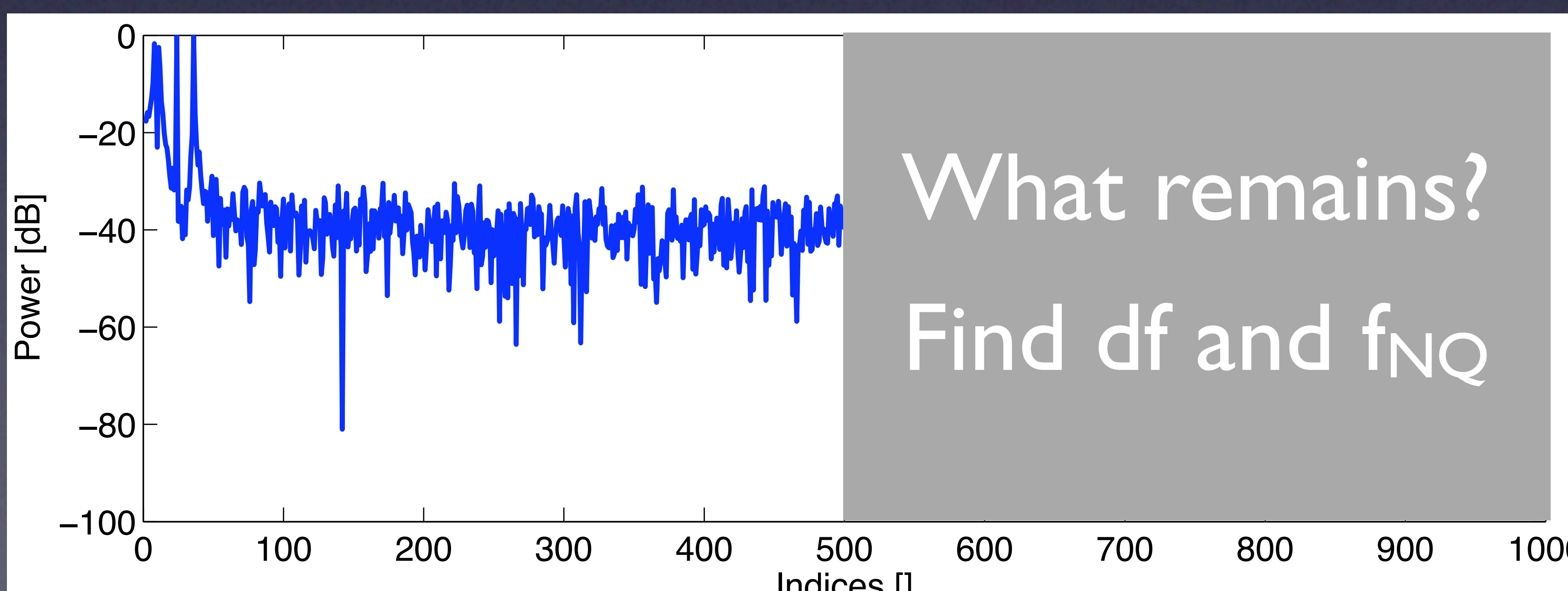
Incomplete: Must label the x-axis?

# Power spectrum x-axis

- Indices & frequencies related in a funny way …
  Examine vector `pow`:



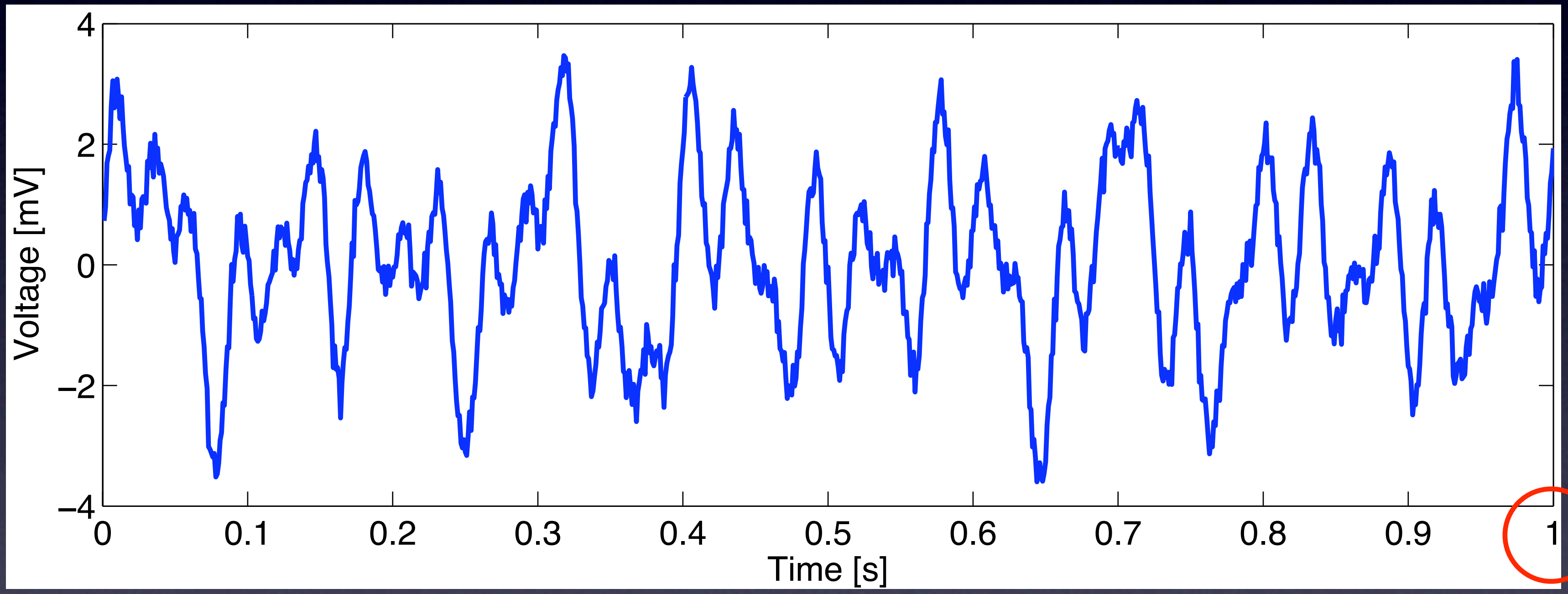

- Because data is real, f < 0 is redundant.



# Power spectrum x-axis

- What is df? $$df = \frac{1}{T}$$ where T = Total time of recording.

Ex:




T = 1 s
df = 1 Hz

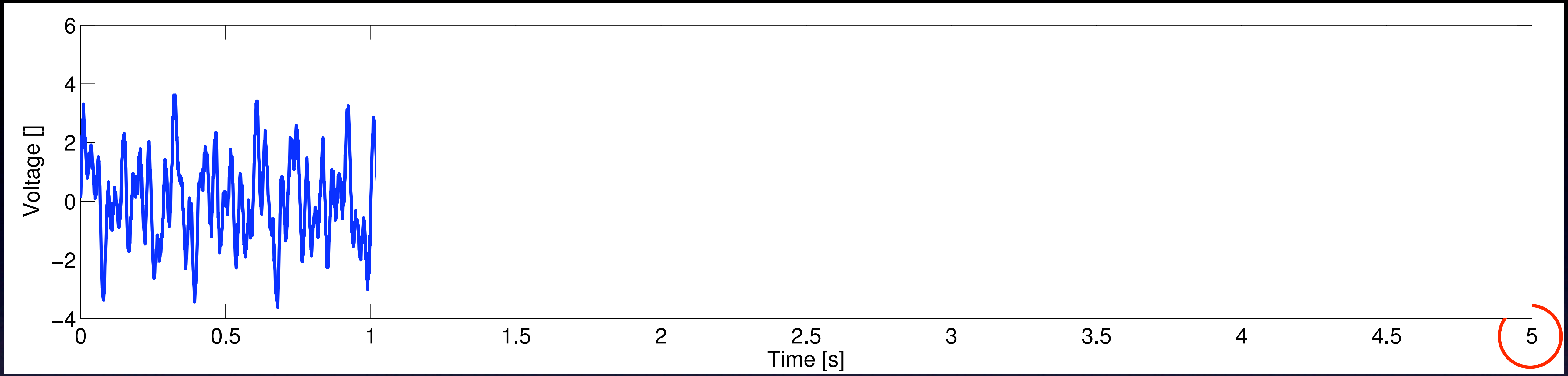
Q: How do we improve frequency resolution?

A: Increase T or record for longer time.

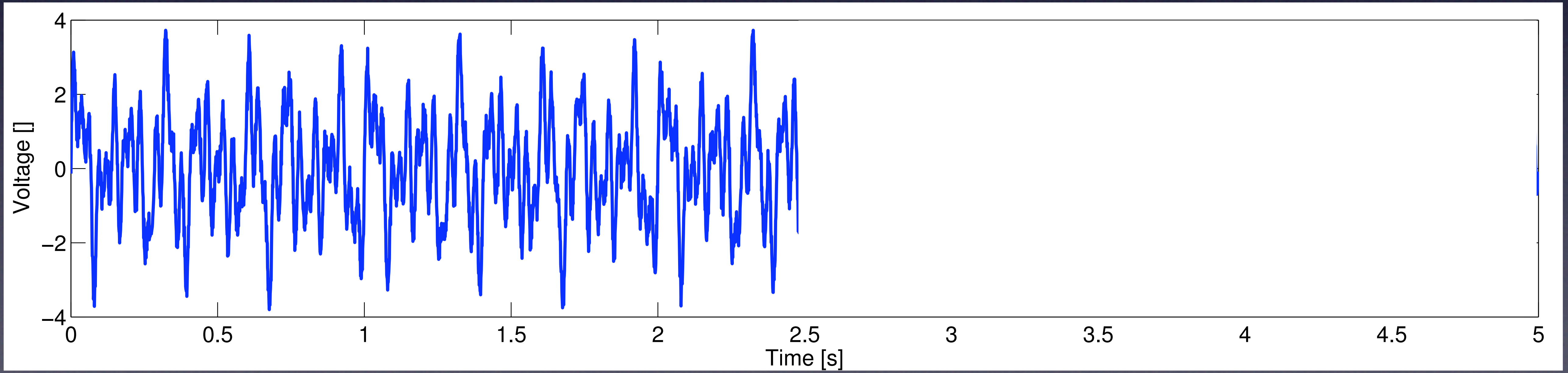
# Examples

- Demand 0.2 Hz frequency resolution.

df = 1/5s = 0.2 Hz




But, data may change during longer recordings . . .




Different spectra in 1st and 2nd half of data . . .

Balance resolution requirements with consistency in data.

# Power spectrum x-axis

- What is $f_{NQ}$?

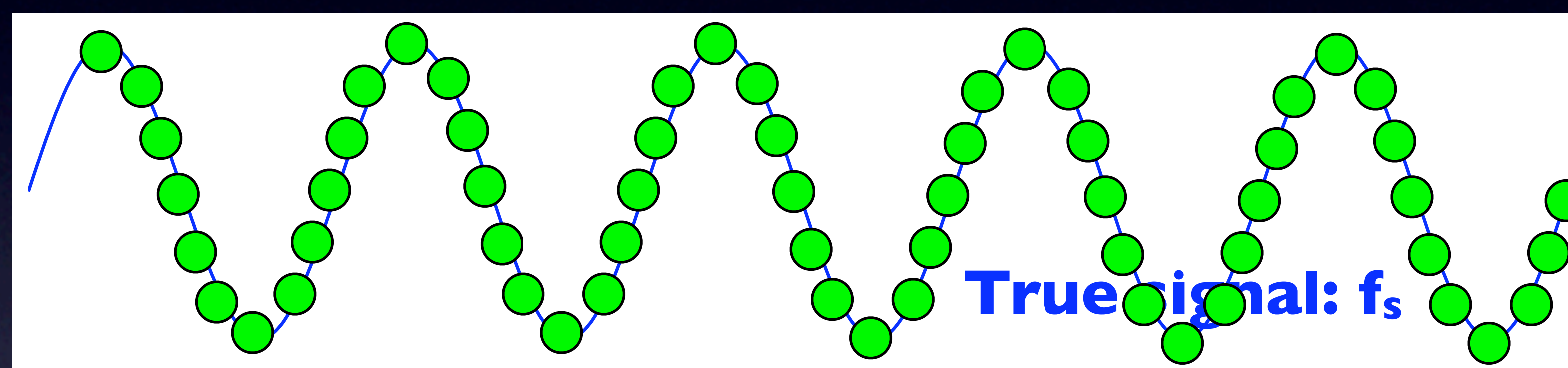
$$f_{\mathrm{NQ}} = \frac{f_0}{2}$$

The Nyquist frequency
where $f_0$ = sampling frequency.

The **highest** frequency we can observe.

Sample:

$f_0 >> 2\ f_s$

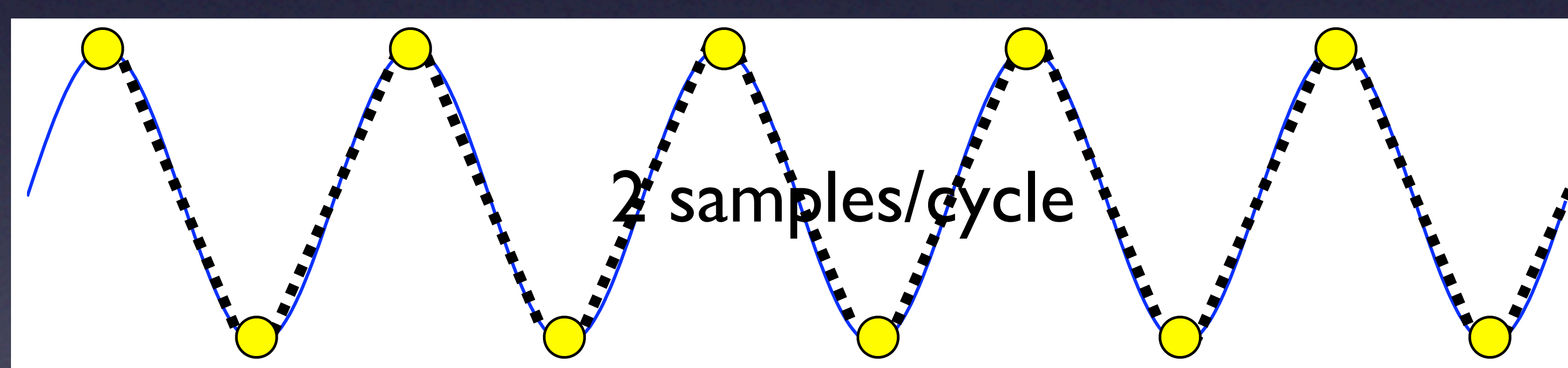



Accurate reconstruction

Too expensive!

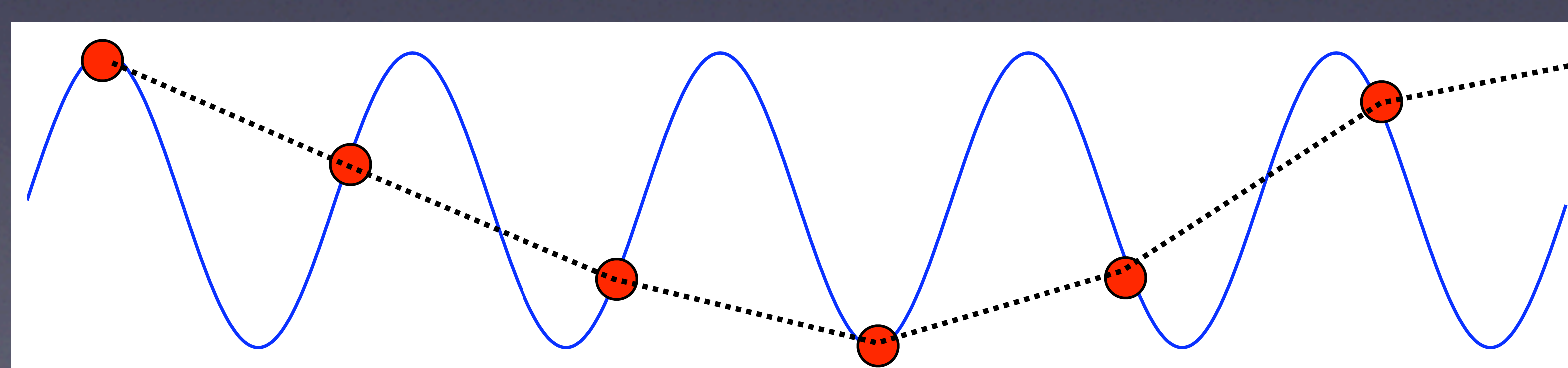
$f_0 = 2\ \boxed{f_s}$

Max freq we can observe at this sample rate!




Enough to reconstruct signal, but just barely.

$f_0 < 2\ f_s$



High frequency (in data) mapped to low frequency (**aliased**).

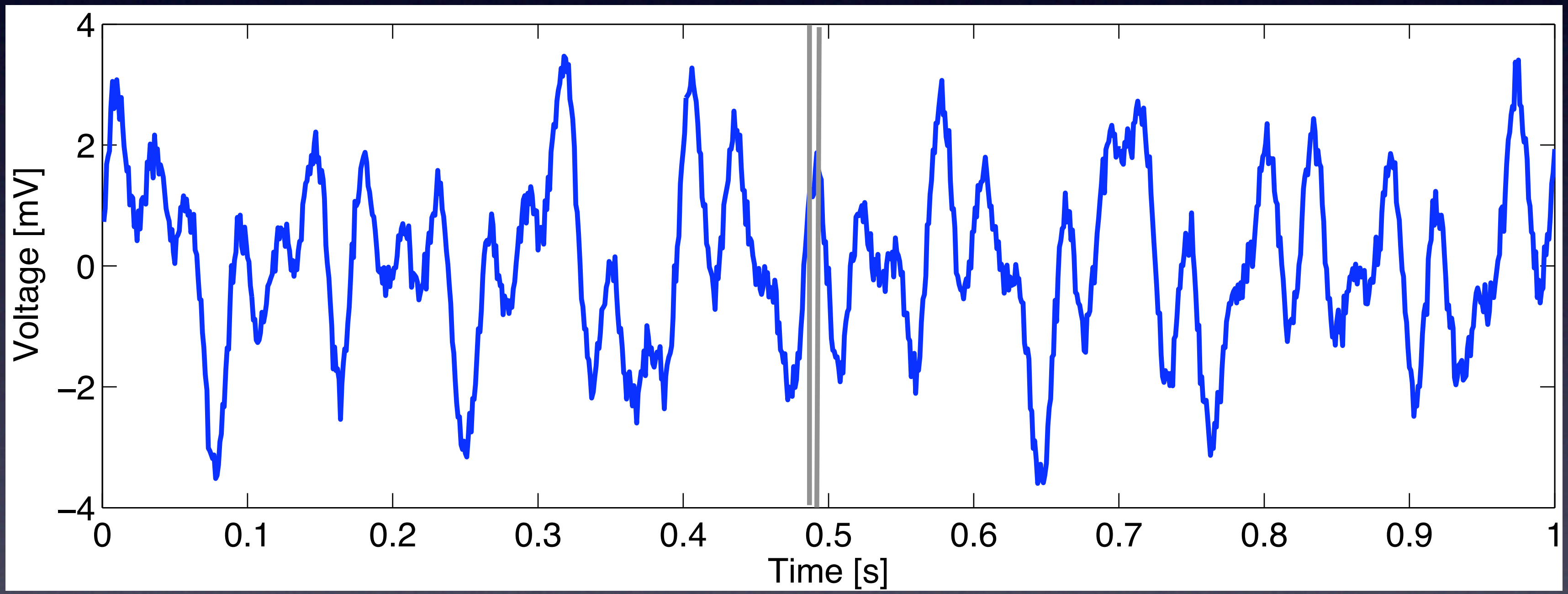
All hope lost! Indistinguishable from true low frequency signals.

# Power spectrum x-axis

Moral: Sample fast enough to capture the highest frequency "true" signal.

Sampling interval: dt = 1 ms

Ex:




Sampling frequency:

$f_0 = 1/dt$

$f_0 = 1000$ Hz

$f_{NQ} = 500$ Hz

Q: How do we increase the Nyquist frequency?

A: Increase the sampling rate $f_0$. [Hardware]

# MATLAB code

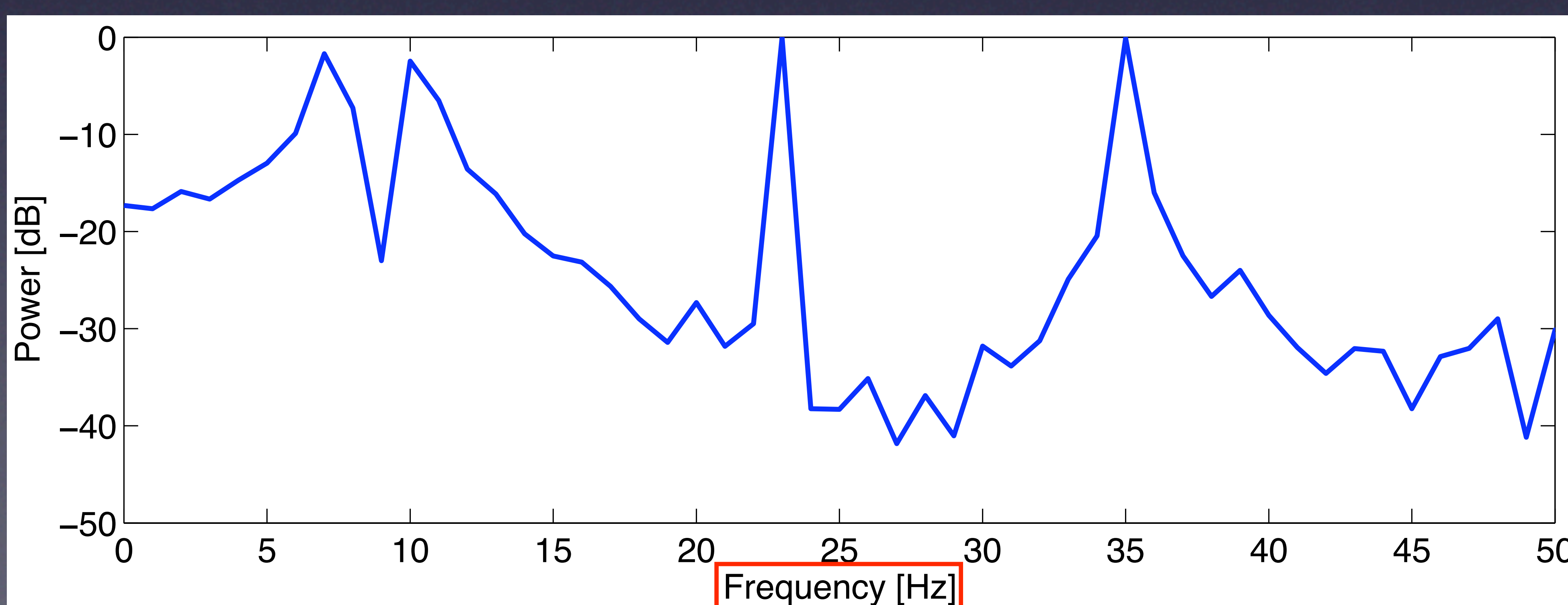
http://makramer.info/sfn

```
>> pow = (abs(fft(v1)).^2)/length(v1);
>> pow = 10*log10(pow/max(pow));
>> pow = pow(1:length(v1)/2+1);
>> df = 1/max(t1);   fNQ = 1/dt/2;
>> faxis = (0:df:fNQ);
>> plot(faxis, pow);  xlim([0 50]);
```

First half of data

Define df & $f_{NQ}$

Frequency axis



# Summary

```
>> pow = (abs(fft(v1)).^2)/length(v1);
```

**Frequency resolution** $$df = \frac{1}{T}$$

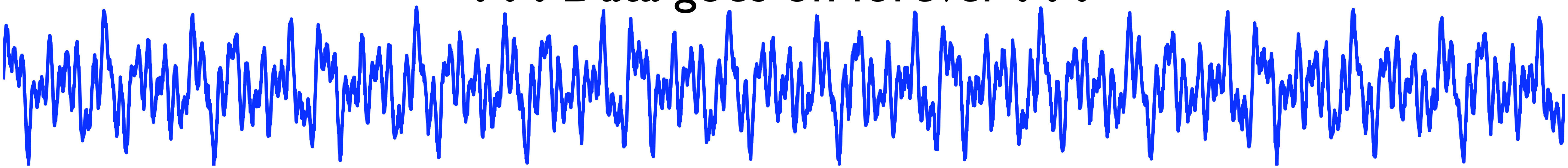
**Nyquist frequency** $$f_{\mathrm{NQ}} = \frac{f_0}{2}$$

- For finer frequency resolution: record more data.
- To observe higher frequencies: increase sampling rate.
- Built-in routines: `>> periodogram(...)`
  **Requires Signal Processing Toolbox**
- Many subtleties …

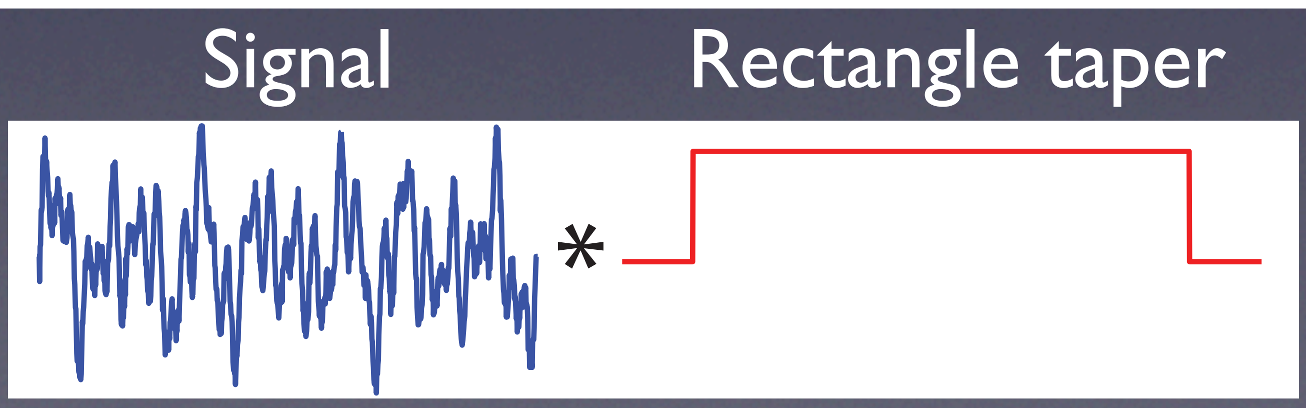
# Tapers

- Doing nothing, we make an implicit taper choice . . .

. . . Data goes on forever . . .



What we're observing:

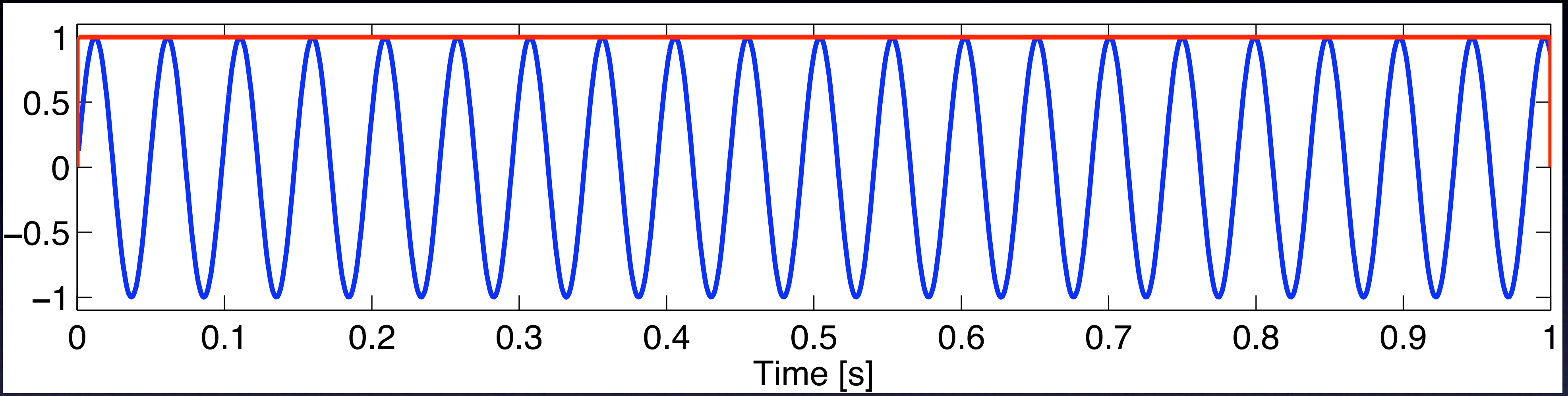


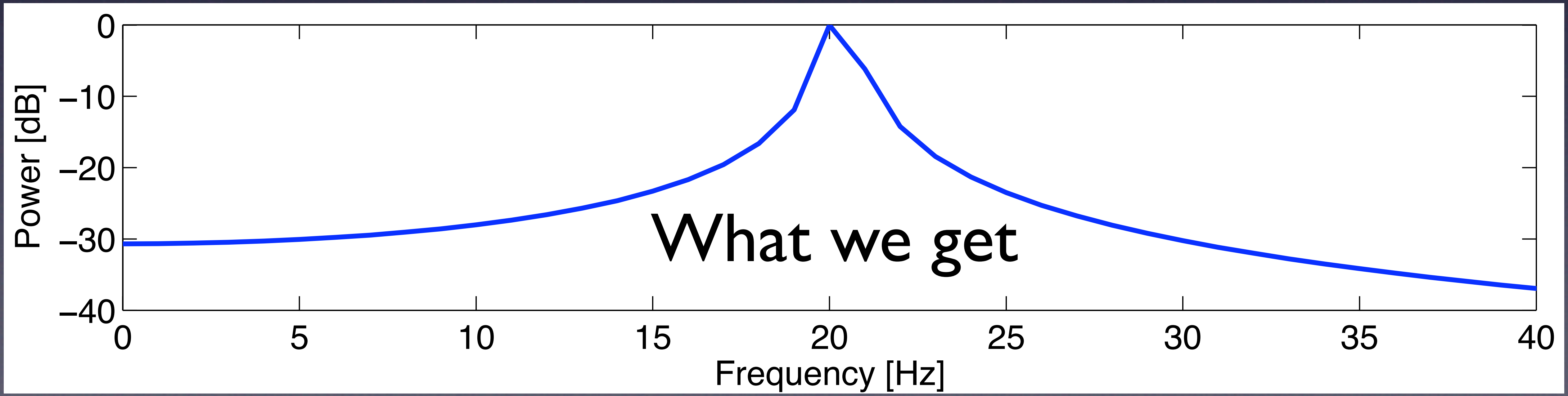
# Tapers

- The rectangle taper *blurs* the power spectrum.

Pure sinusoid

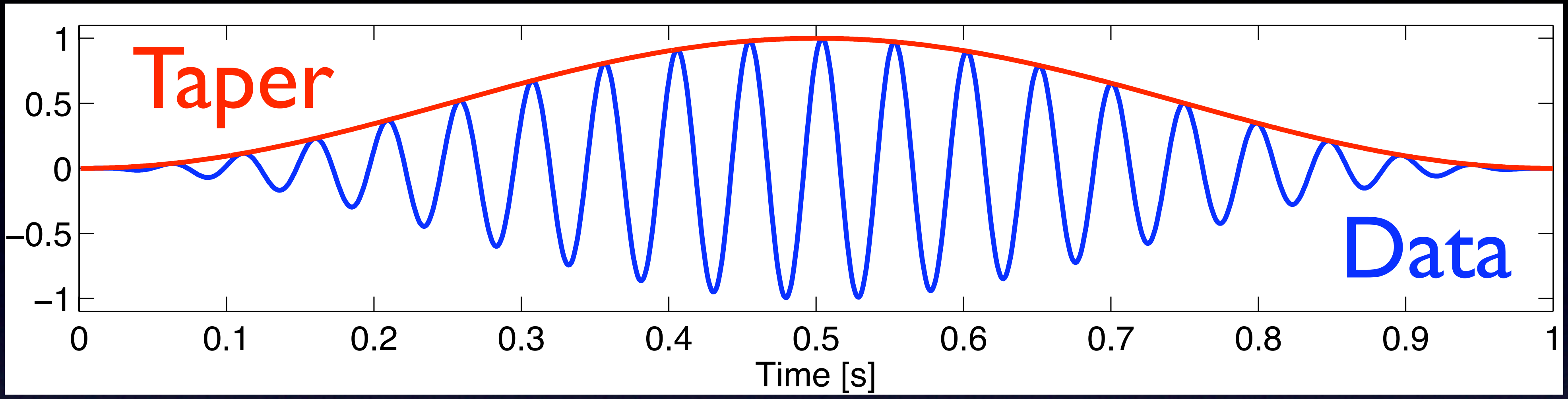
near 20 Hz



# Hann taper

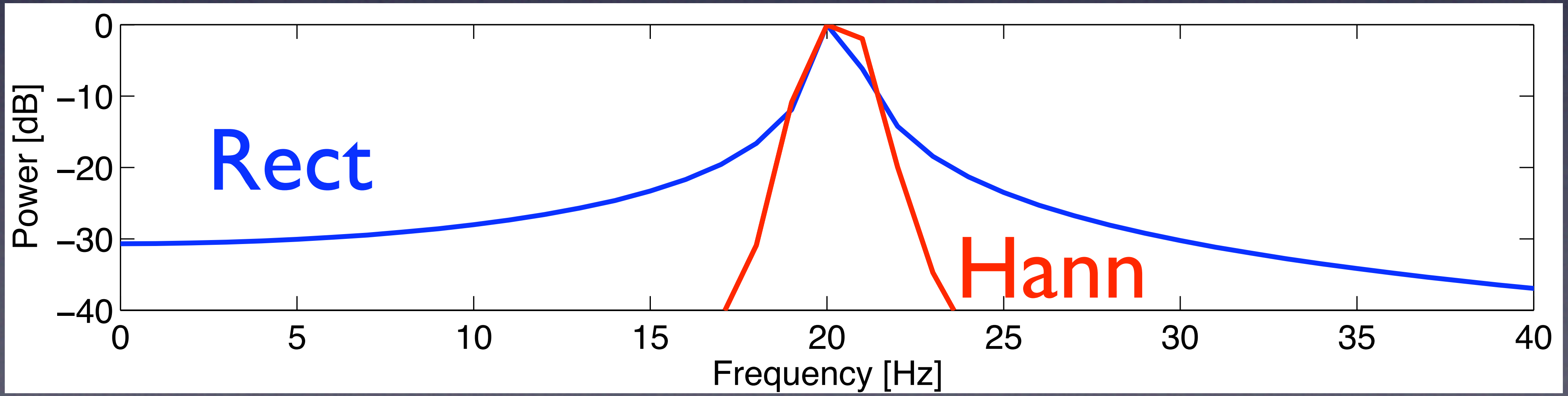
- Idea: smooth the sharp edges of rectangle taper.




```
>> st = s .* hann(length(s))' ;
```

**Requires Signal Processing Toolbox**

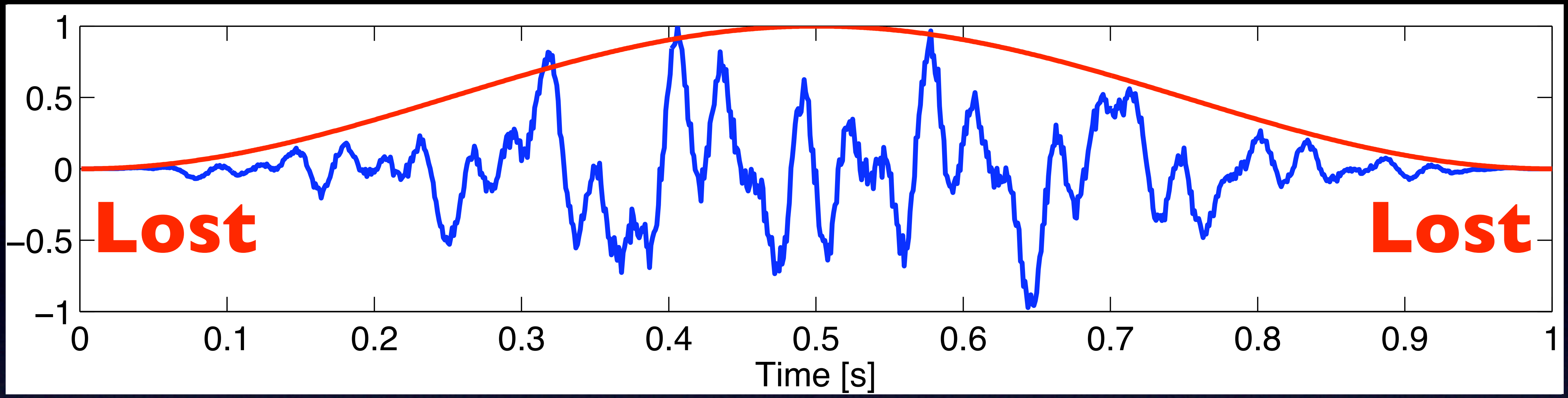
- Compute power spectrum of tapered data.

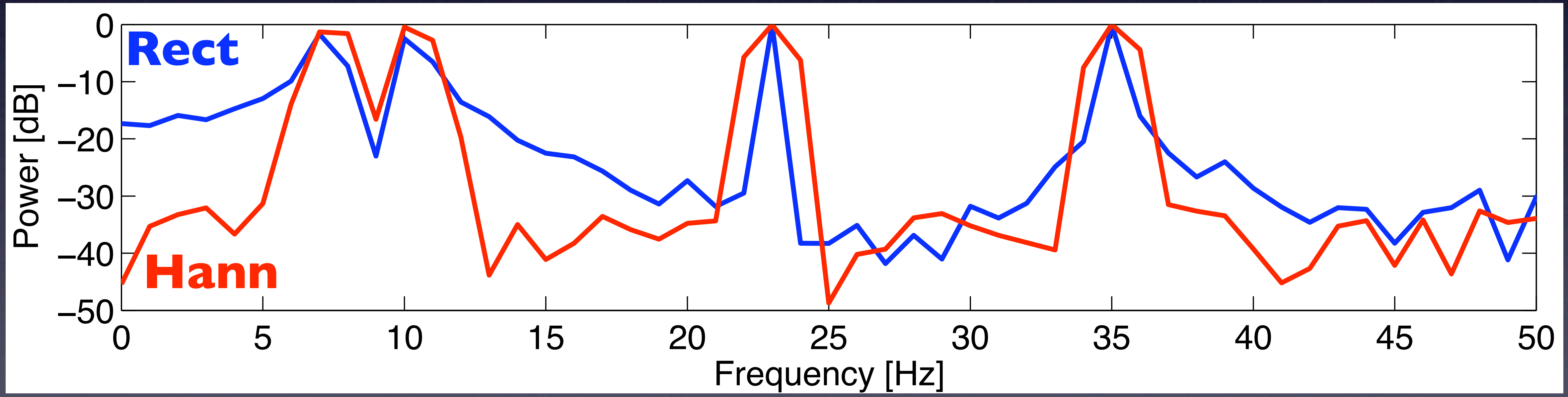



Taper reduces the “sidelobes”.

# Ex: Hann taper

v1:

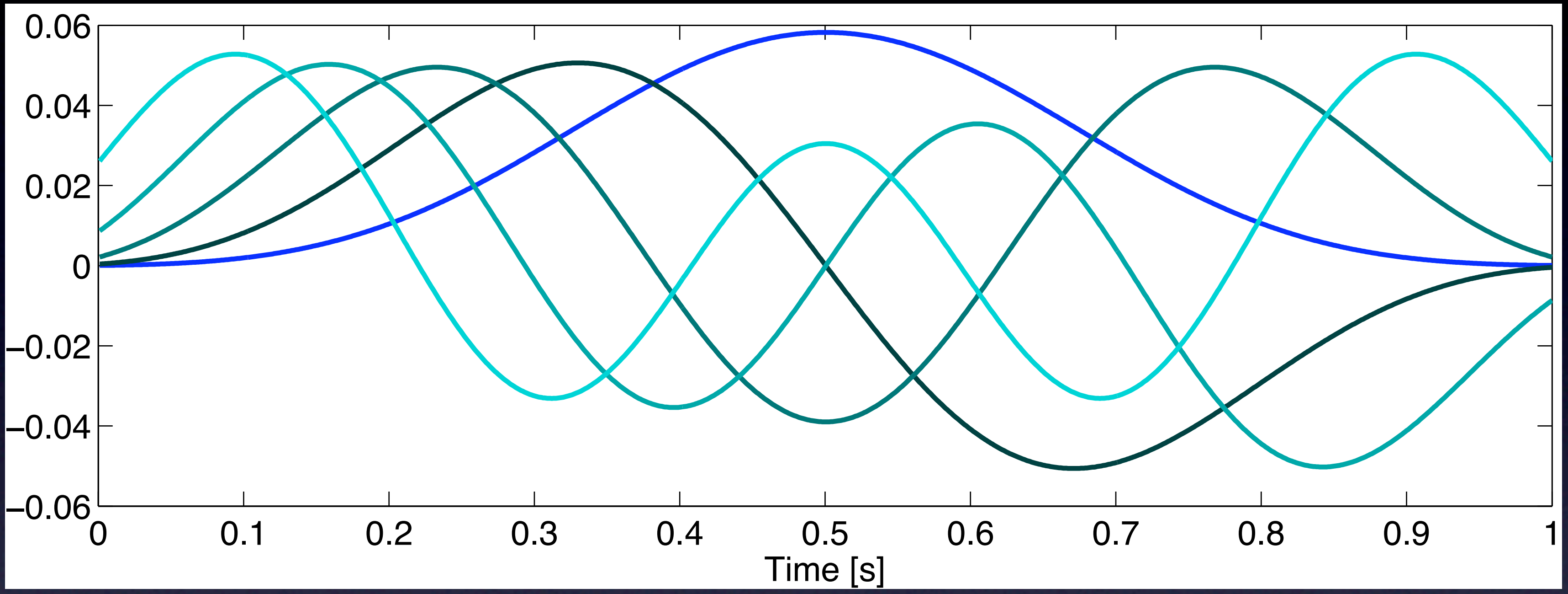



- Good: Deeper baseline
- Bad: Broader peaks & lose data at edges.

# Multi-taper Method

- Idea: Apply lots of different tapers

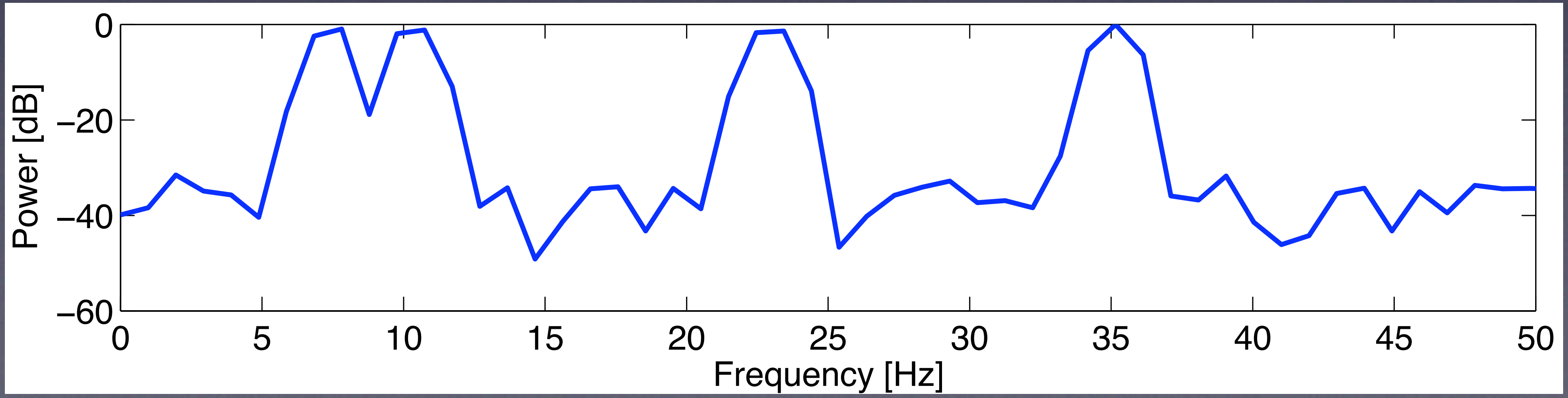



Reduce sidelobes

Keep data edges

- Chronux (www.chronux.org)

```
>>  mtmspectrumc(v1,params);
```

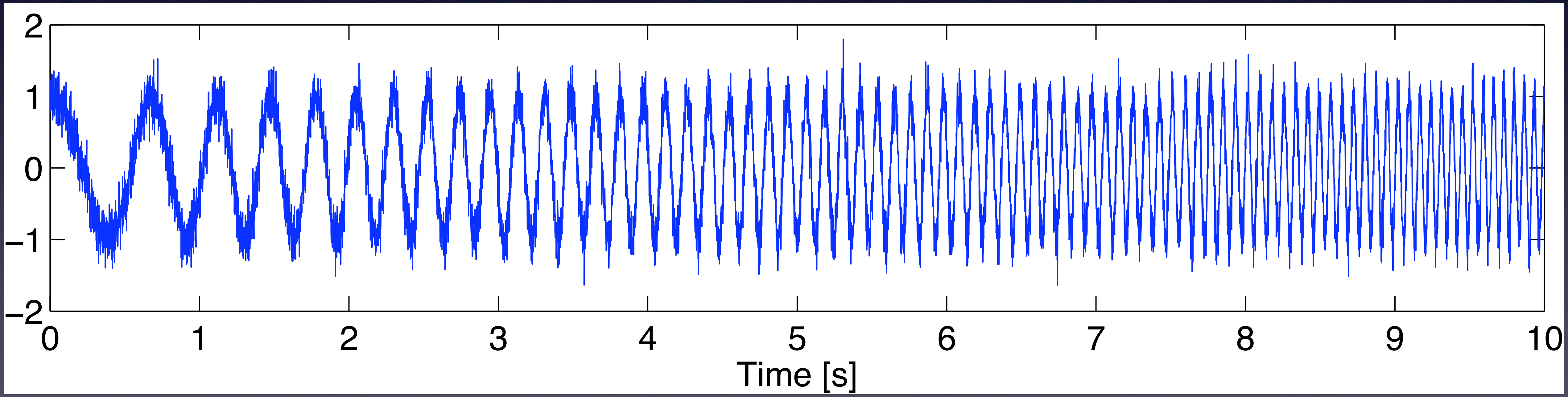


# Spectrogram

- What if signal characteristics change in time?

```
>> load data.mat
>> plot(t2,v2)
```

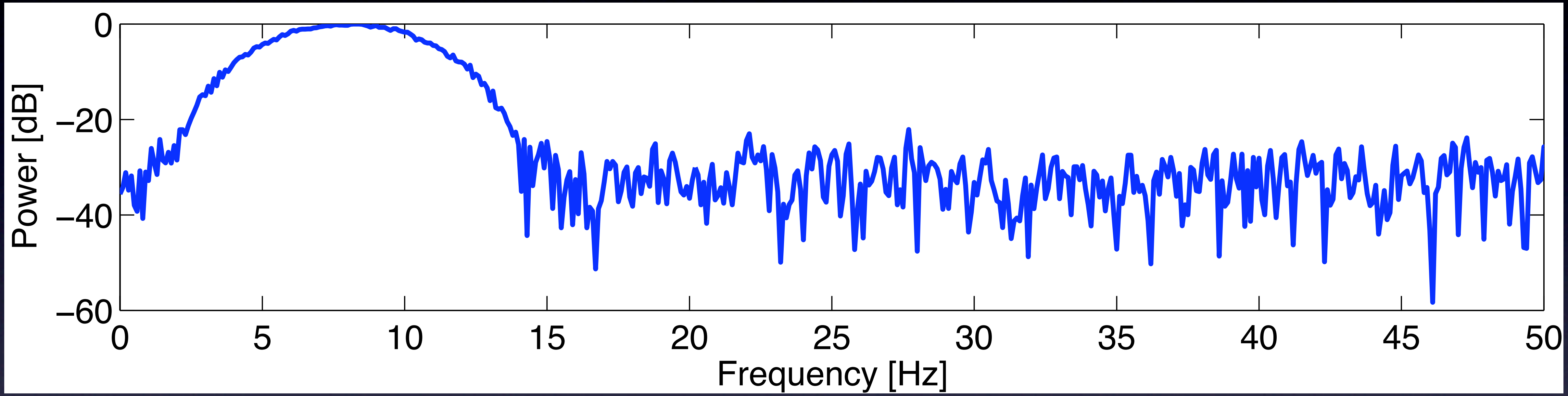
http://makramer.info/sfn



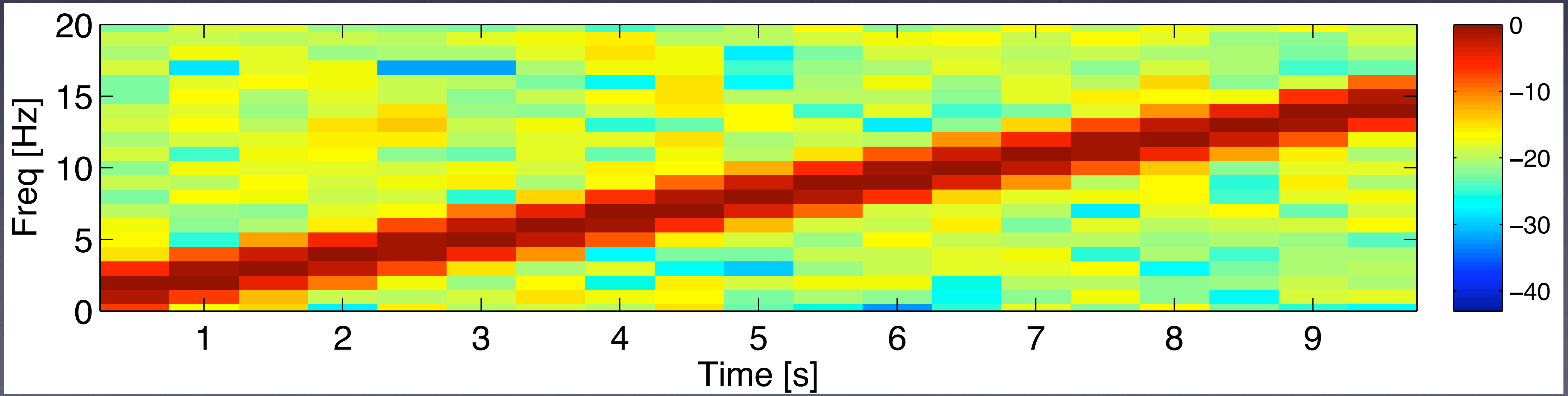

- Visual inspection
- Data characteristics change in time.

# Spectrogram

- Compute the spectrum (Hann taper) of all data

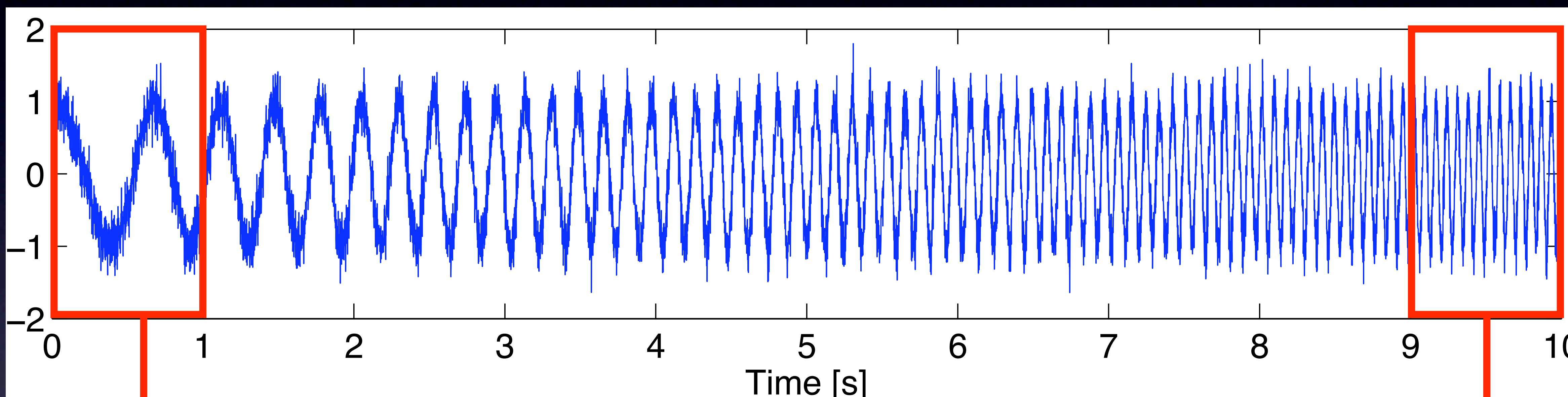


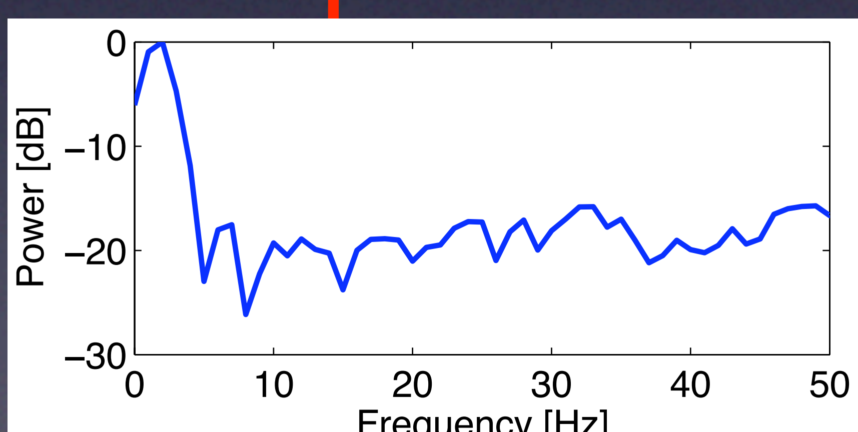

Q: Is this a good representation of the data?



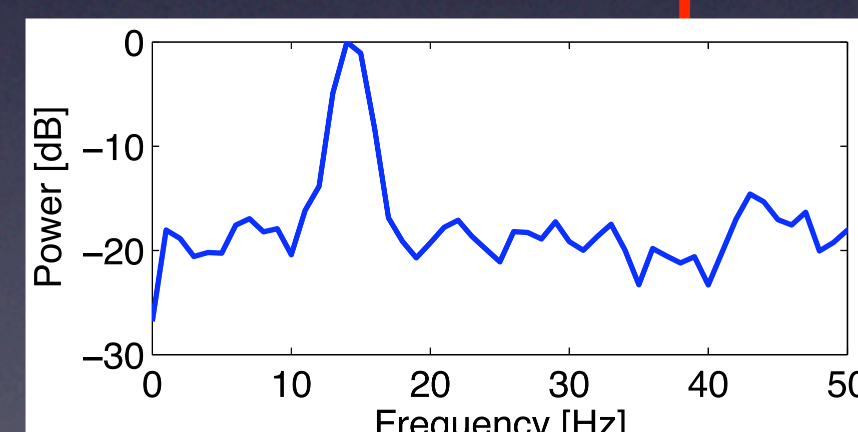
# Spectrogram

- Idea: Split up the data into windows & Compute spectrum in each.




**df = 1Hz**




Different spectra at beginning and end of signal.
Repeat for many overlapping windows . . .

# MATLAB code

http://makramer.info/sfn

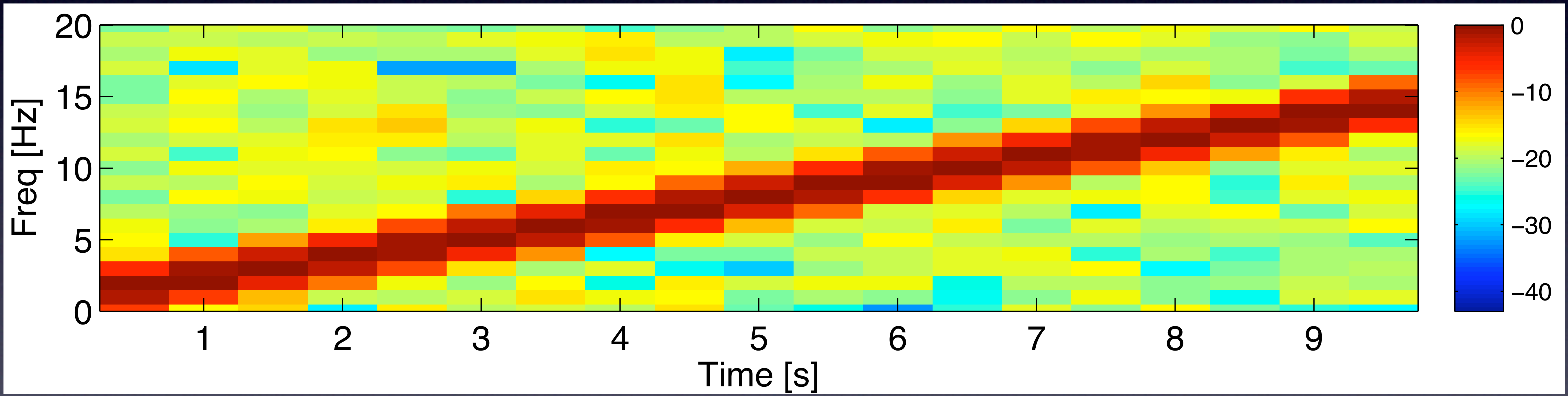
**Requires Signal Processing Toolbox**

**Window** **Padding**

```
>> [S,F,T]=spectrogram(v2,1s,0.5s,1s,1kHz)
```

**Overlap** **$f_0$**

```
>> S = abs(S);
>> imagesc(T,F,10*log10(S/max(S(:))));
```




Plot power [color] vs frequency and time

A better representation of the data?

Can compute multi-taper spectrogram! (Chronux)

# Conclusions & Refs

- We focused on power spectrum (not wavelets).
- Defined *df* and $f_{NQ}$.
- Explored tapers and spectrograms.

**References**

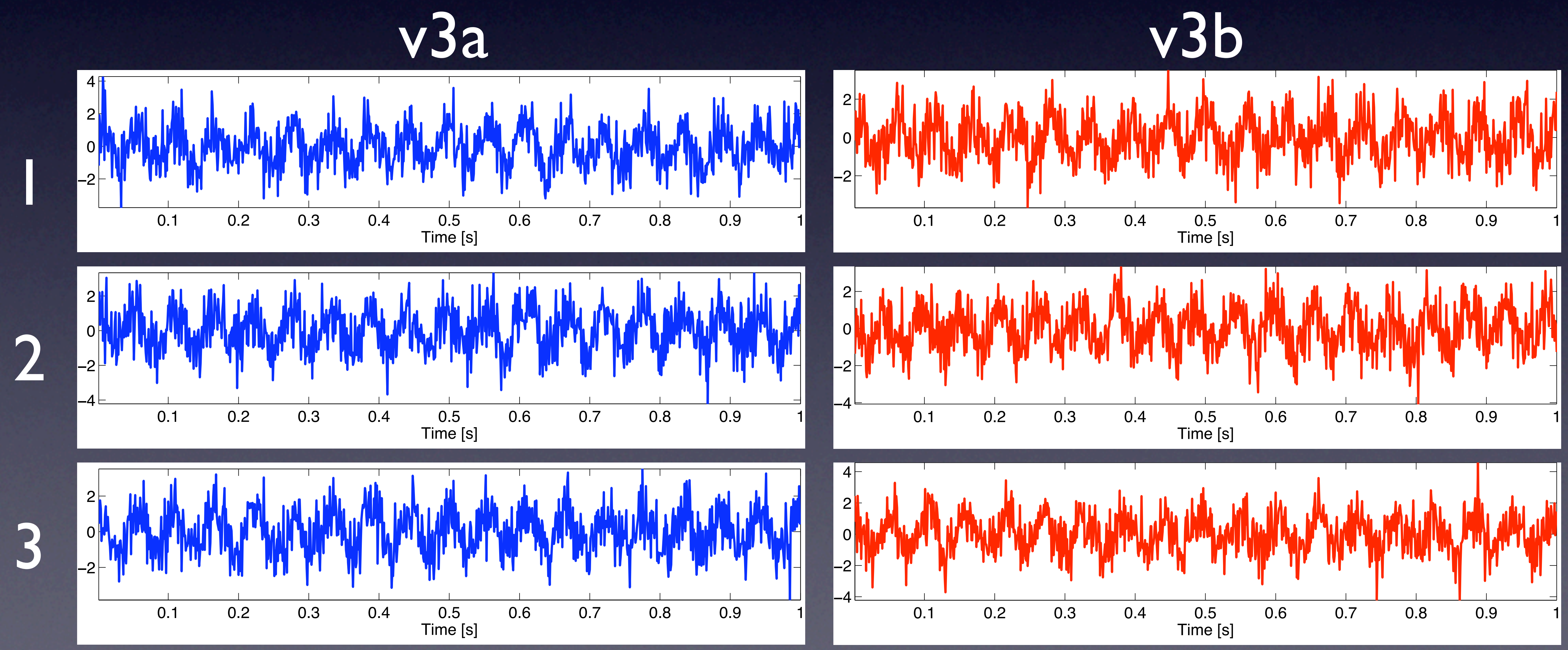

*MATLAB for Neuroscientists, Numerical Recipes in C*

*Chronux.org and Neuroinformatics Summer Course*

*EEGLab*

# Coherence

- Idea: examine phase relationship between signals.
- Requires two signals.
- Requires multiple trials for each signal.

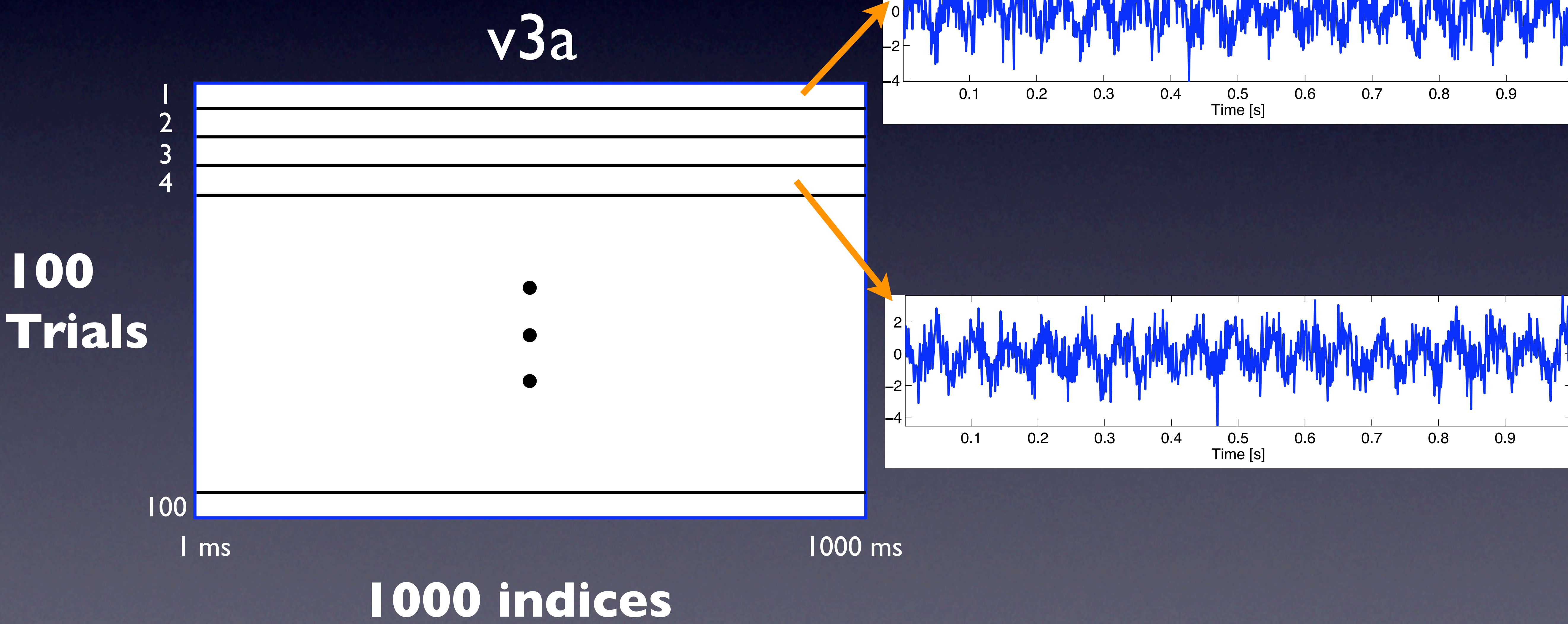


# MATLAB code

http://makramer.info/sfn

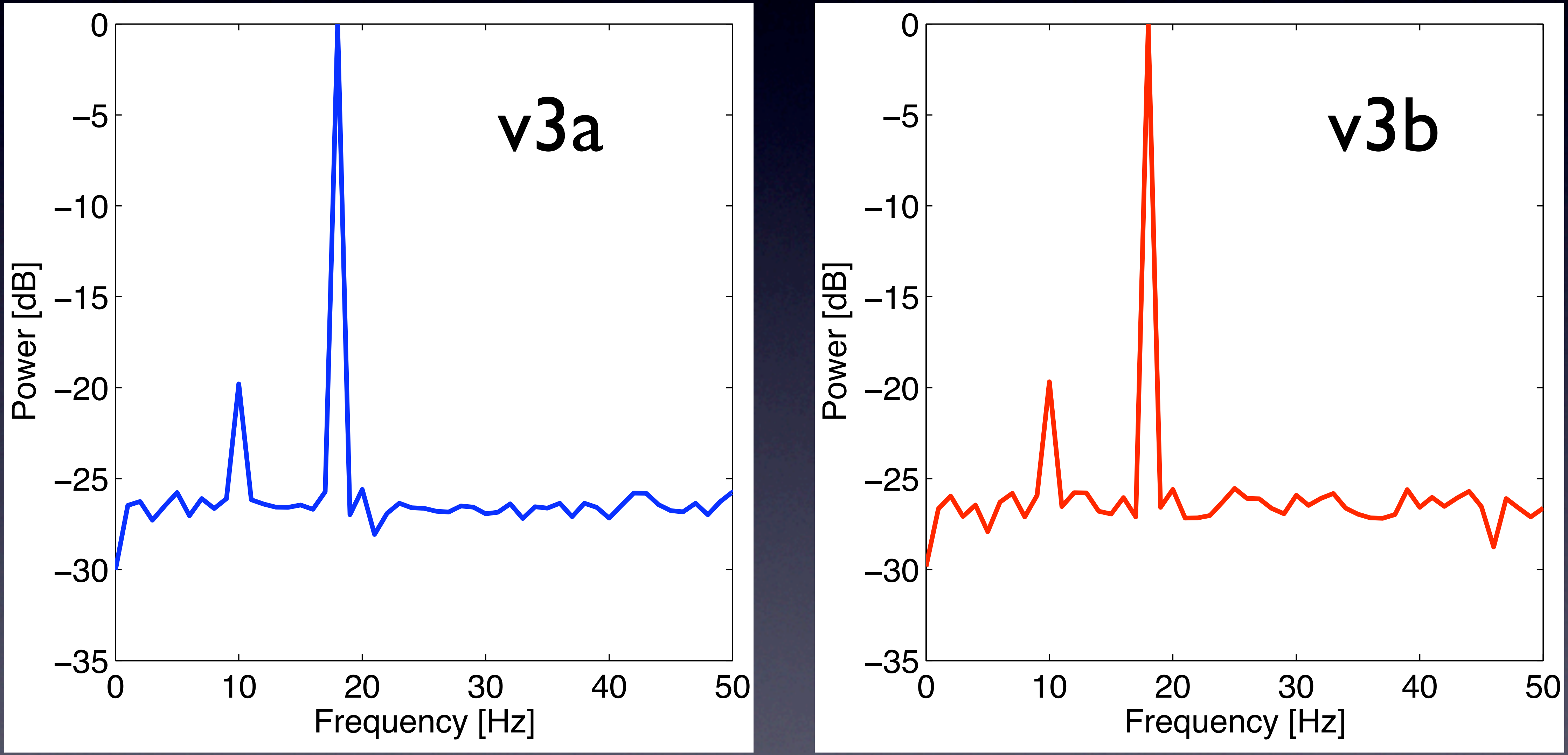
```
>> load data.mat
```

Two variables: v3a & v3b



# Power spectrum

- Compute the power spectrum for each trial,
  Then average over all trials.




**Power at 10 Hz and 18 Hz.**

Q: Are these rhythms coherent between v3a and v3b?

# Coherence

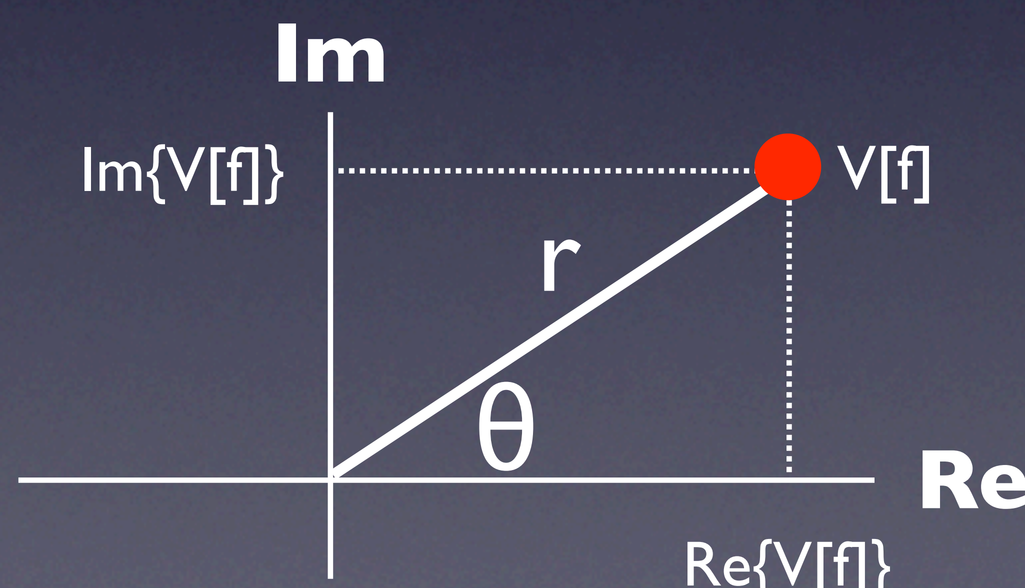
- For each trial, compute the **phase** at each frequency.

Fourier transform

complex

$$V[f] = \int_{-\infty}^{\infty} v[t] e^{-2\pi i f t} dt = (\,\mathrm{Re}\{V[f]\}, \mathrm{Im}\{V[f]\}\,)$$

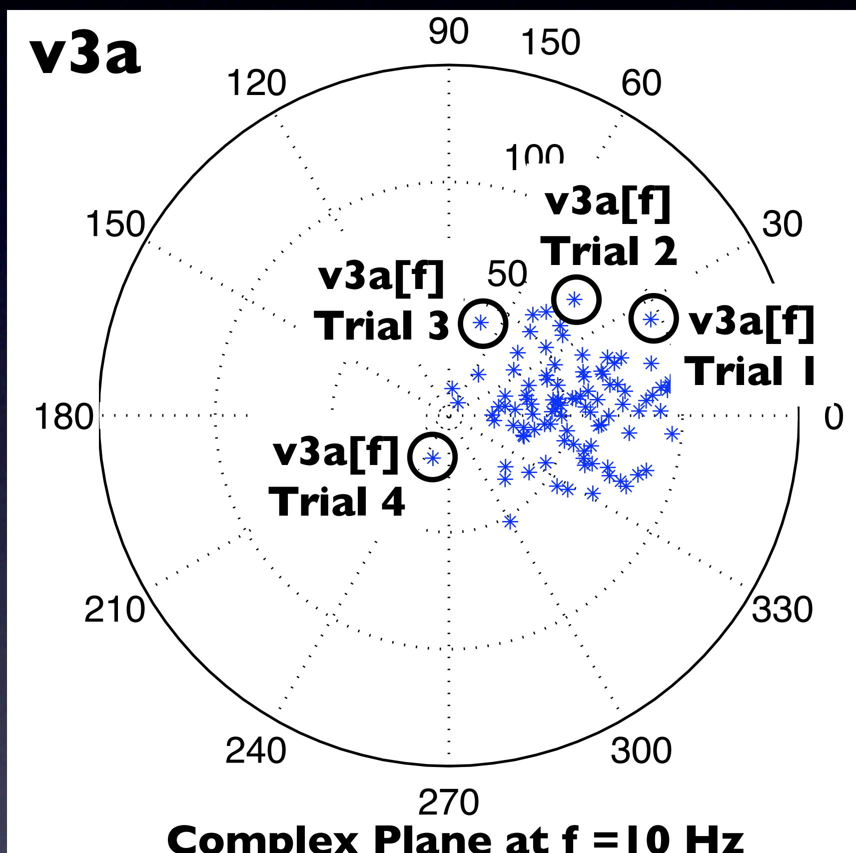
Examine this complex plane for our data . . .

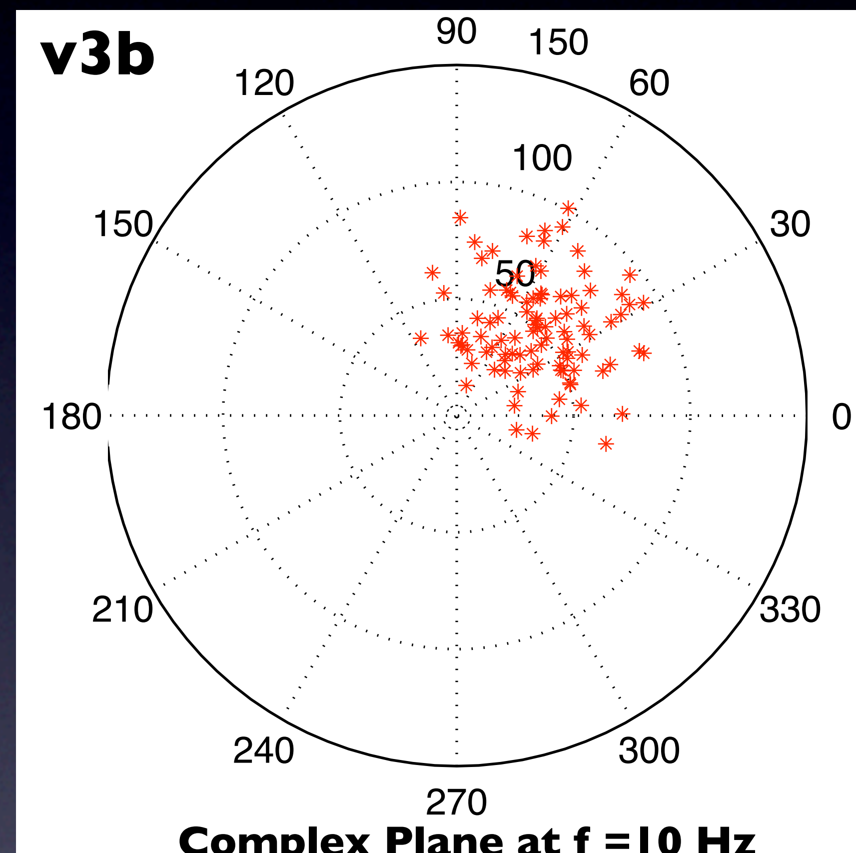


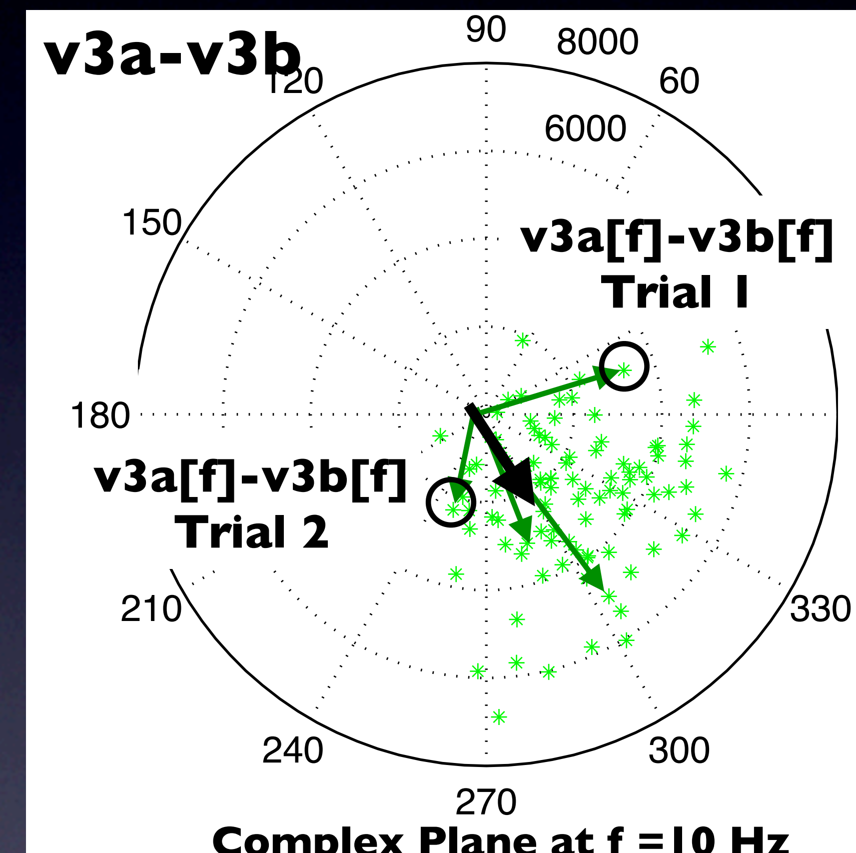

θ = phase of V at frequency f.

# Plot complex plane at 10 Hz

- For each trial, compute FT(data) & plot . . .



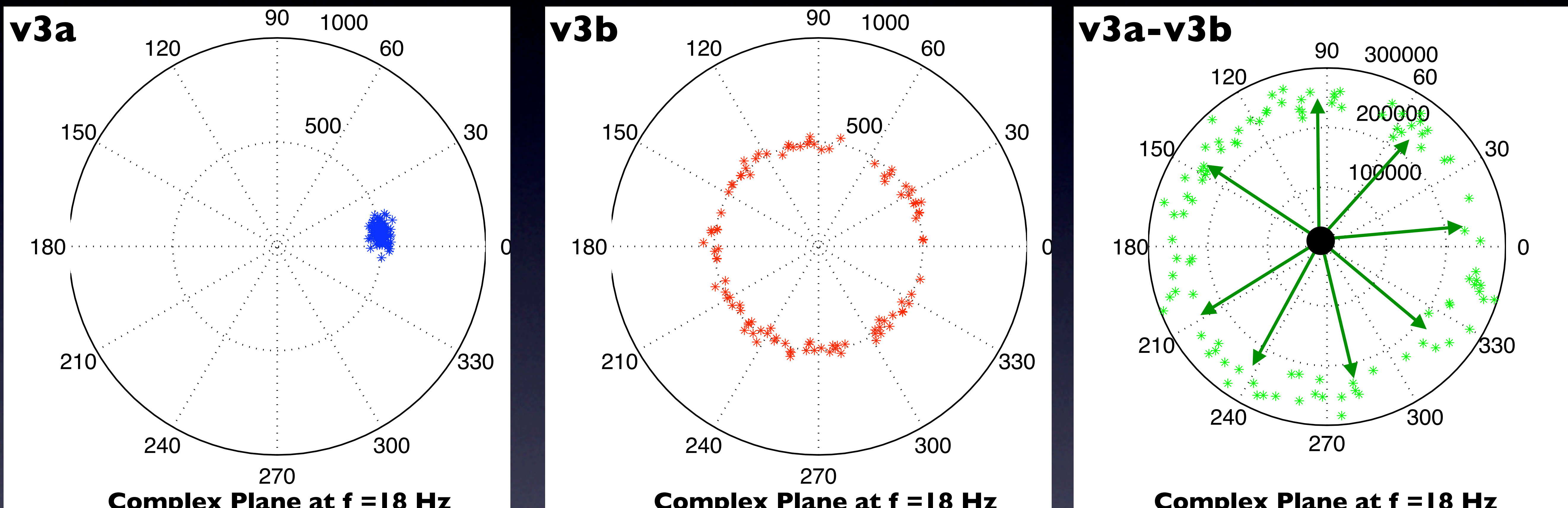

Coherent at 10 Hz? Examine their difference (trial by trial).

Summarize: Draw the vector to each complex difference.
Compute the mean vector

Phase concentration = Nonzero mean vector = **coherent**

# Plot complex plane at 18 Hz

- For each trial, compute FT(data) & plot . . .



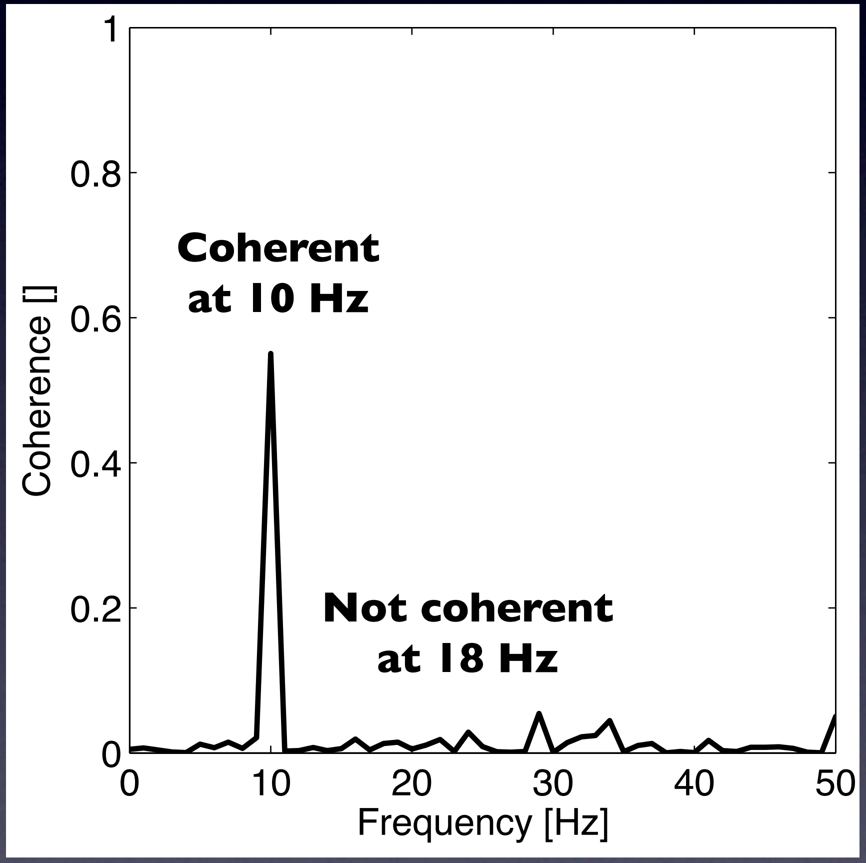

Coherent? Plot the complex difference

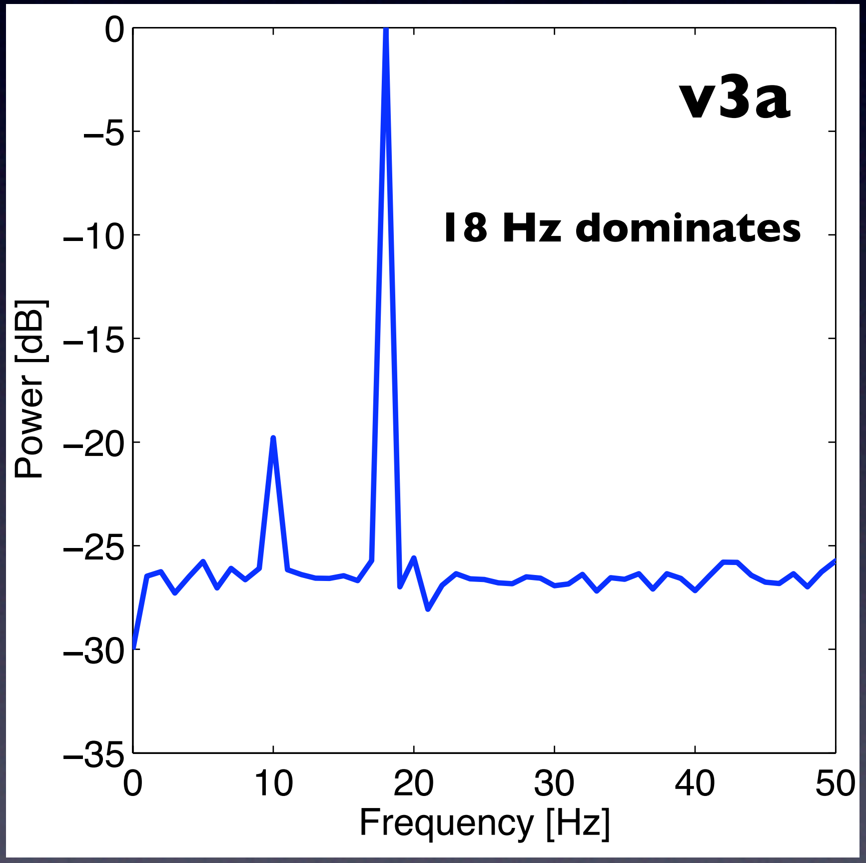
Summarize: Draw the vector to each complex difference.
Compute the mean vector

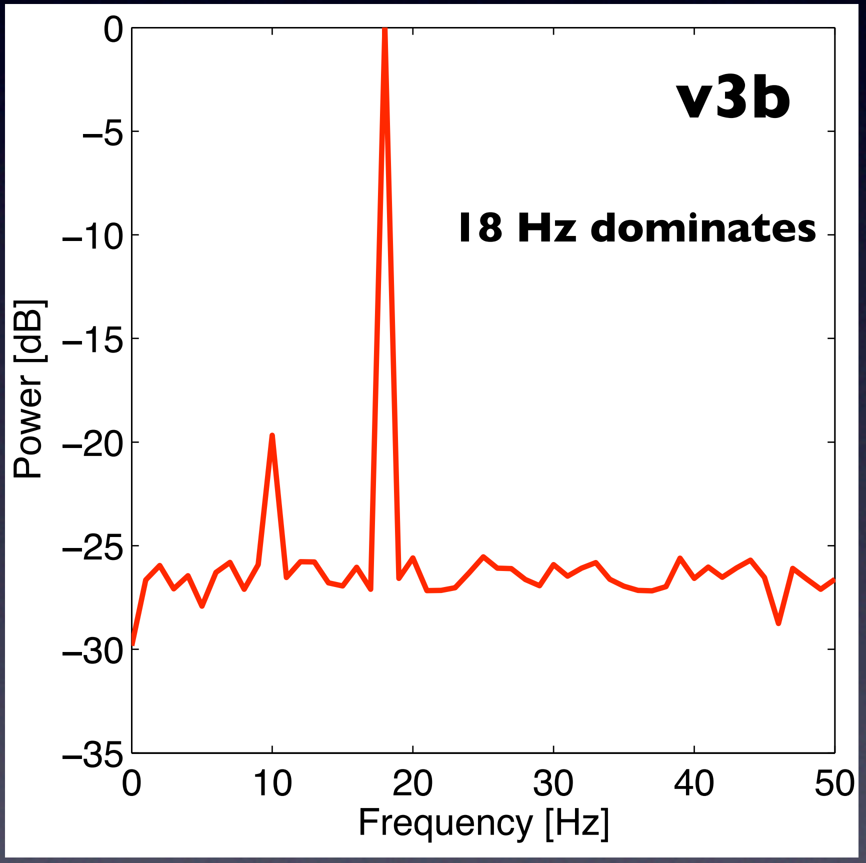
Phase dispersion = Zero mean vector = No coherence

# Coherence

- Idea: Examine the "angular concentration" of vector differences in the complex plane.




Strong power does not imply coherence.

Can compute multi-taper coherence! (Chronux)

# Conclusions & Ref

- More about coherence:
  [Nunez et al., Electroenceph Clin Neurophys, 1997]
  [Bruns, J Neurosci Methods, 2004]

- There are many coupling measures

  Cross correlation, phase consistency, Granger causality, cross-frequency coupling, . . .

  [Pereda et al., Progress in Neurobiology (2005)]

- SfN AbstractsOnline:
  137 results for “coherence”.

# Conclusions

- We examined techniques to quantify rhythms and their interactions in data.
- Many different techniques exist.
- Tutorial slides & MATLAB code available http://makramer.info/sfn
- Contact me at SfN to talk more:
  Mark --- mak@bu.edu

Thanks!

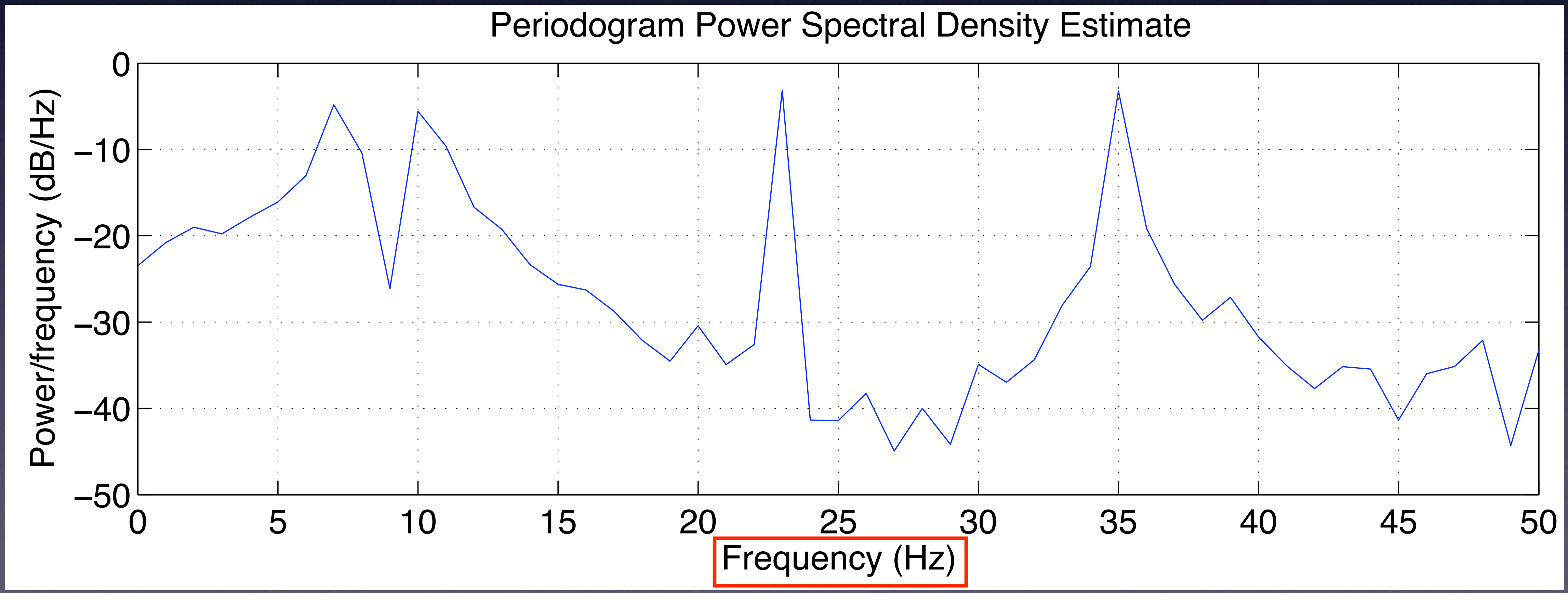
# Extra Material

# Signal Processing Toolbox

- Use built-in MATLAB routine:

**Taper** **Zero padding** **$f_0$**

```
>> periodogram(v1,[],length(v1),1000Hz);
```

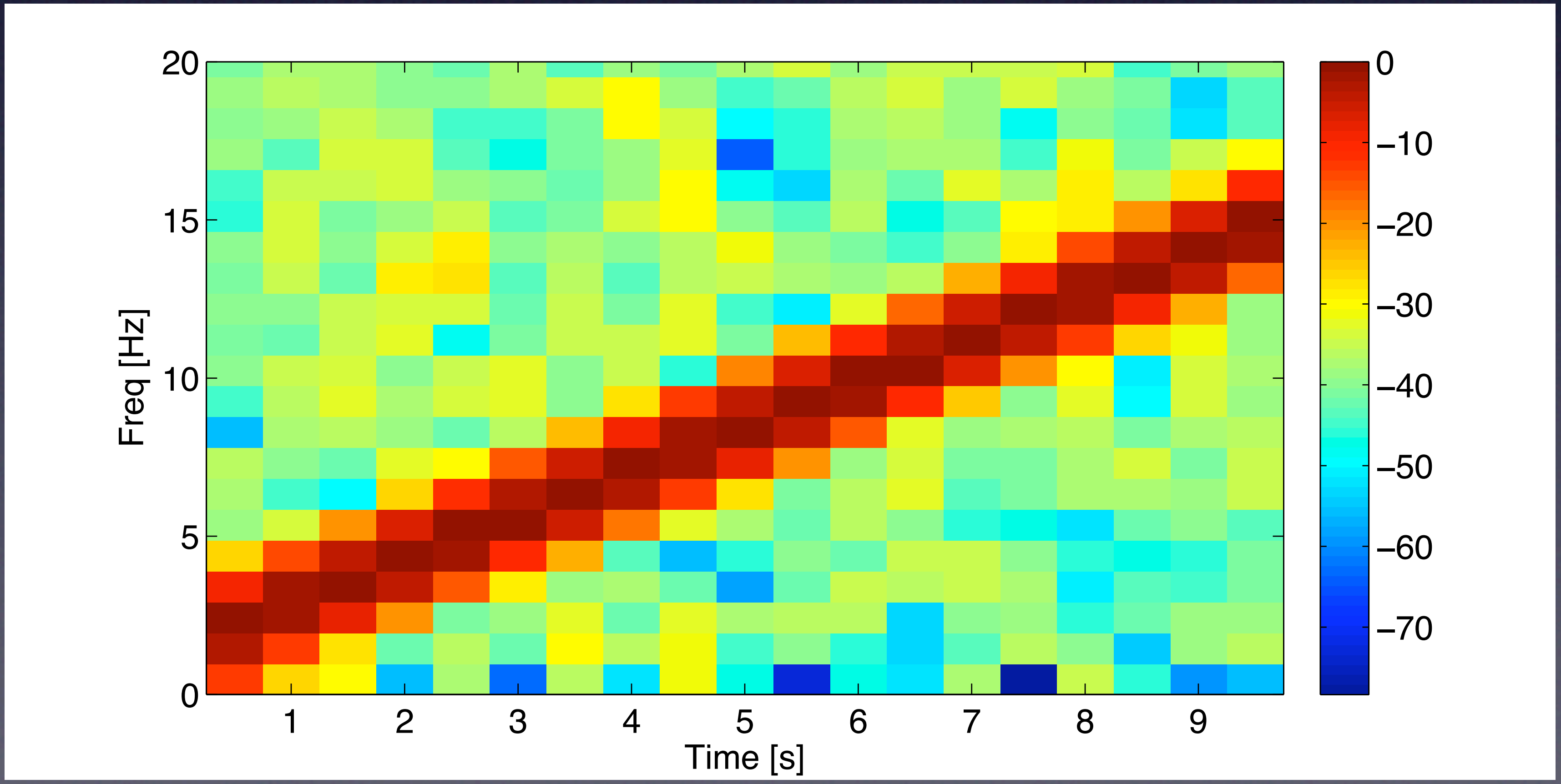



Correct axis!

# Multi-taper spectrogram

- Chronux (www.chronux.org)

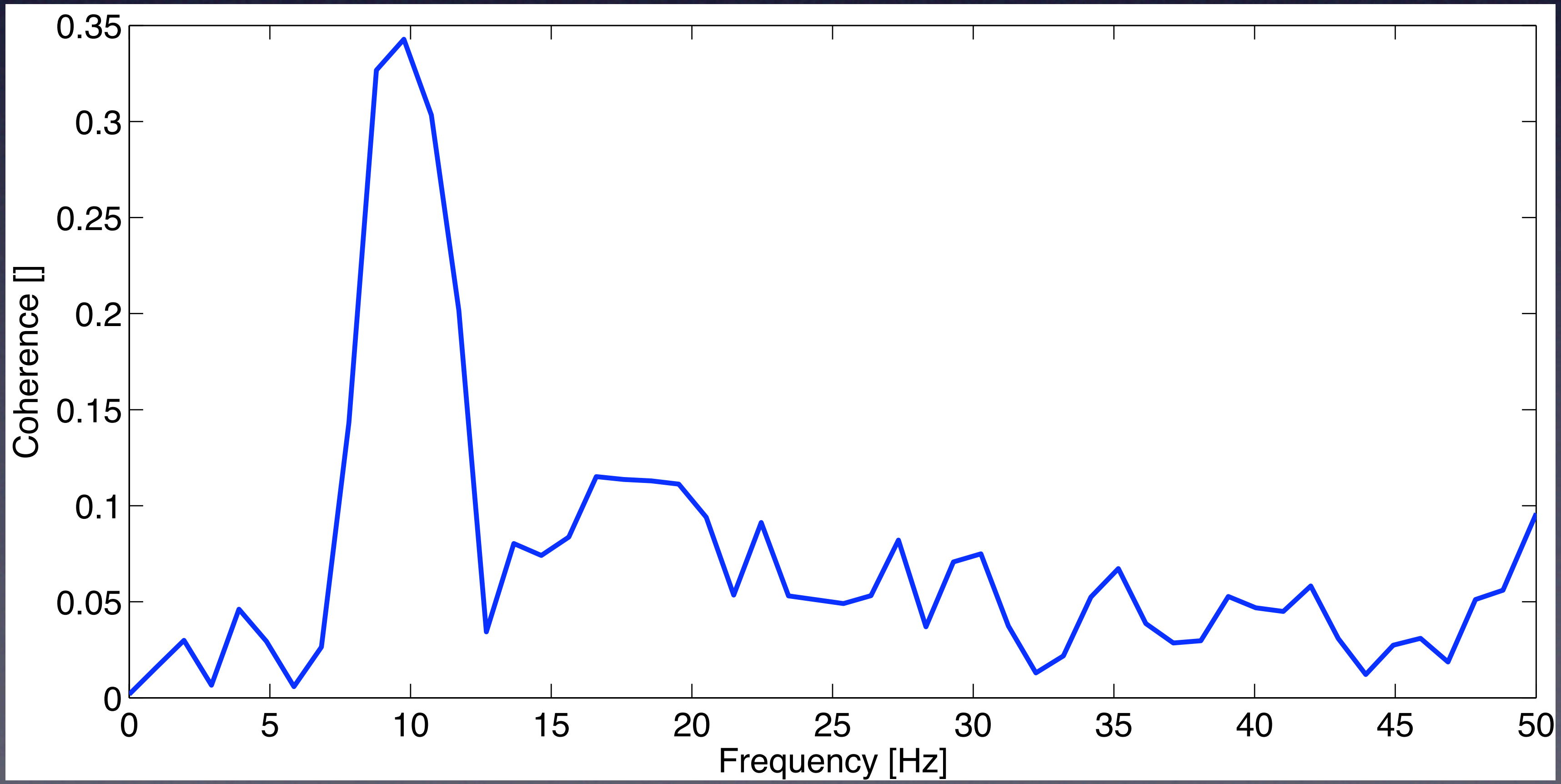
```
>> [S,T,F]=mtspectgramc(v2,[1s,0.5s],params)
>> imagesc(T,F,10*log10(S/max(S(:)))');
```



# Multi-taper coherency

- Chronux (www.chronux.org)

```
>> [C,phi,S12,S1,S2,f]=coherencyc(v3a',v3b',params)
>> plot(f,C);
```



# Coherence formalism

**cross spectrum**

$$C_{12}[f] = \frac{E\{V_1[f]V_2^*[f]\}}{E\{|V_1[f]|^2\}E\{|V_2[f]|^2\}}$$

**power spectrum**

E: Sum over trials

Note: could taper here!

http://makramer.info/sfn

```
sxy = zeros(ntrials, ttrials);
sxx = zeros(ntrials, ttrials);
syy = zeros(ntrials, ttrials);

for k=1:ntrials
    sxy(k,:) = fft(v3a(k,:)).*conj(fft(v3b(k,:)));
    sxx(k,:) = fft(v3a(k,:)).*conj(fft(v3a(k,:)));
    syy(k,:) = fft(v3b(k,:)).*conj(fft(v3b(k,:)));
end

coh = (abs(sum(sxy,1)).^2) ./ (sum(sxx,1) .* sum(syy,1));
plot(faxis, coh(1:length(coh)/2+1))
xlim([0 50]);  ylim([0 1])
xlabel('Frequency [Hz]')
ylabel('Coherence')
```

Define cross and power spectra

For each trail, compute spectra.

Sum over trials.