Lecture 08: Class-Chosen Topics

NENS 230: Analysis Techniques in Neuroscience

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Cell Arrays v. Structs

- Similarities
  - Both are generic arrays
  - Both are iterable over elements
  - Both support arbitrary number of dimensions
  - Easy to convert between the two (`cell2struct, struct2cell`)
  - Anything you can do with a cell array you can also do with a struct.
- Differences
  - Cell arrays are like tables w/o headers.
  - Structs are like tables with headers.

Cell Arrays v. Structs

- Use cell arrays when you want to access elements via numerical indices.
- Use when you want to access elements via field names.

Choosing one or the other is usually a point of style, convenience, or clarity.

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

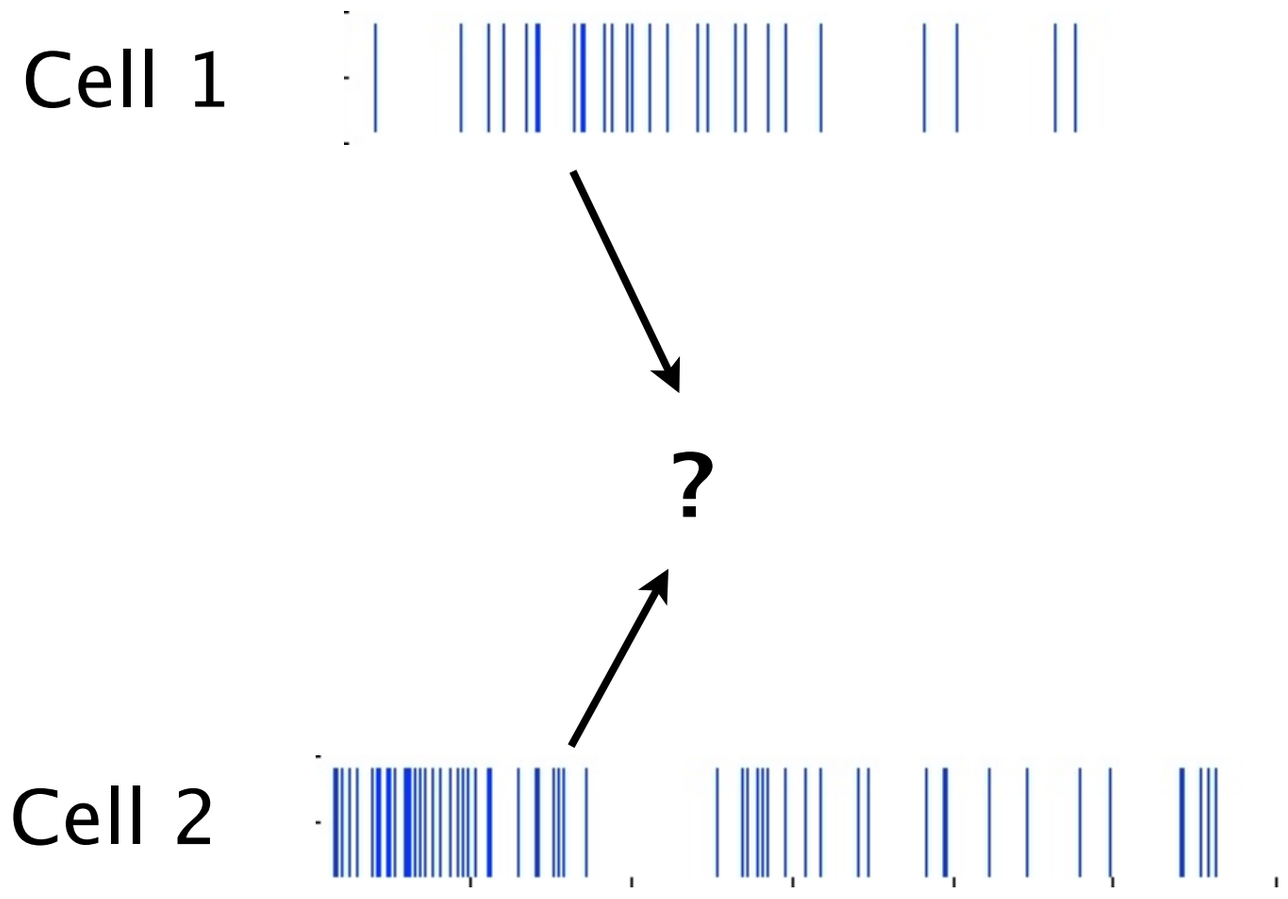
Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Spike Train Analysis

- Inter-Spike Interval Histogram
- Cross-Correlogram
- Autocorrelogram

How can we assess relationships between spike trains?



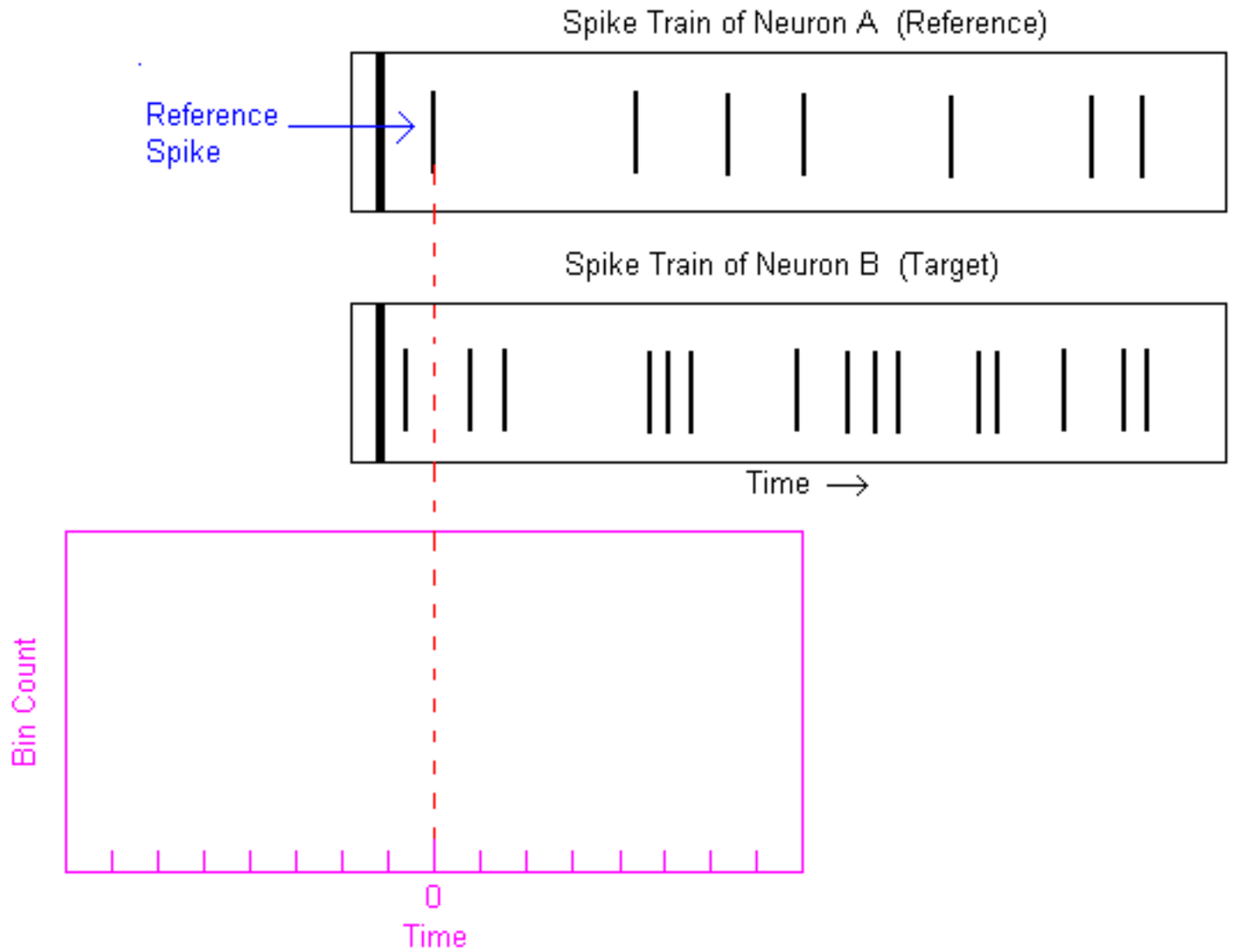
Multiple Spike Train Analysis

Methods Available

- Cross-Correlation analysis (Cross Correlogram)
- Information Theoretic Methods
- Granger Causality
- Generalized Linear Models
- Graphical Models

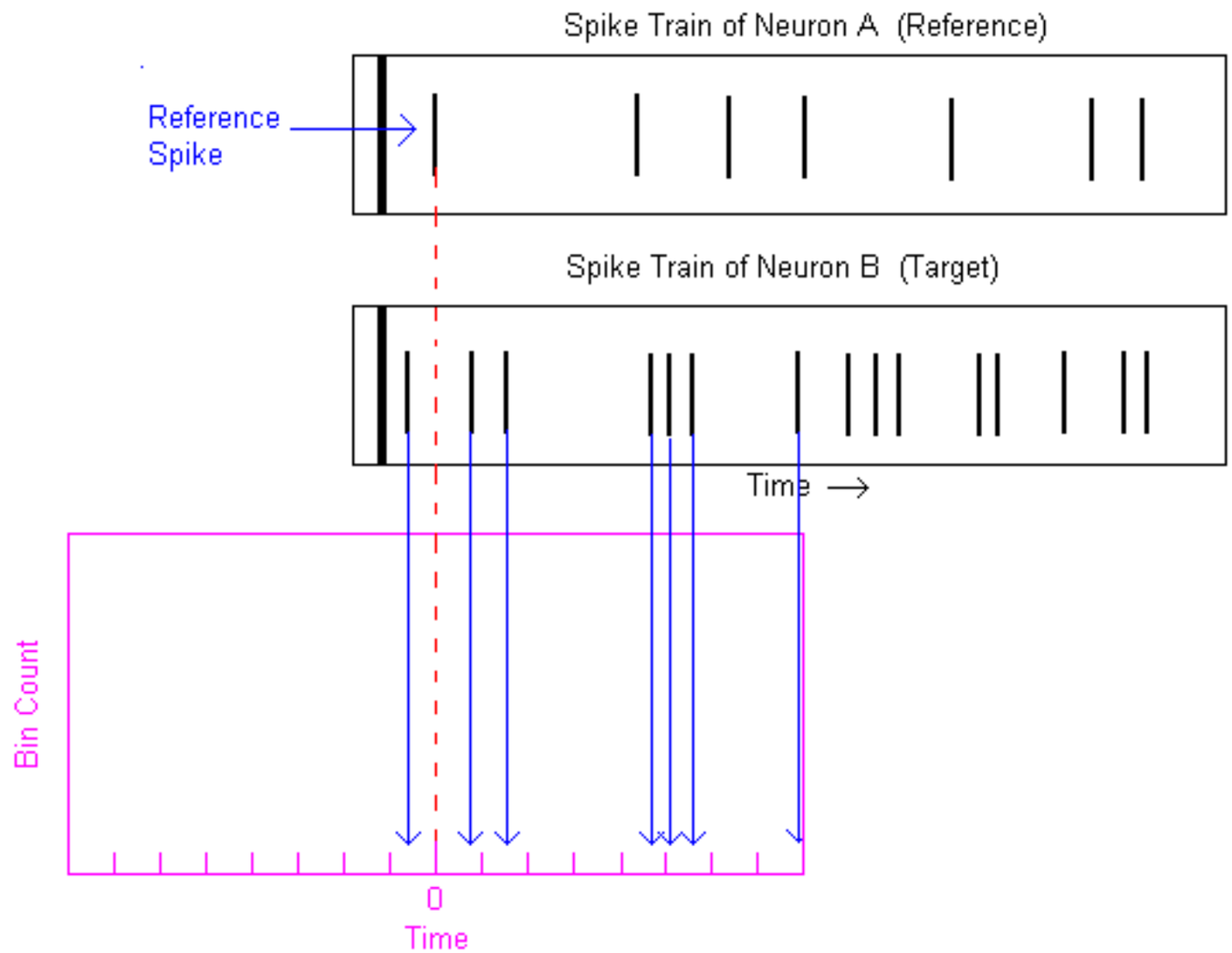
Reference: Statistical Signal Processing for Neuroscience and Technology

Cross Correlogram



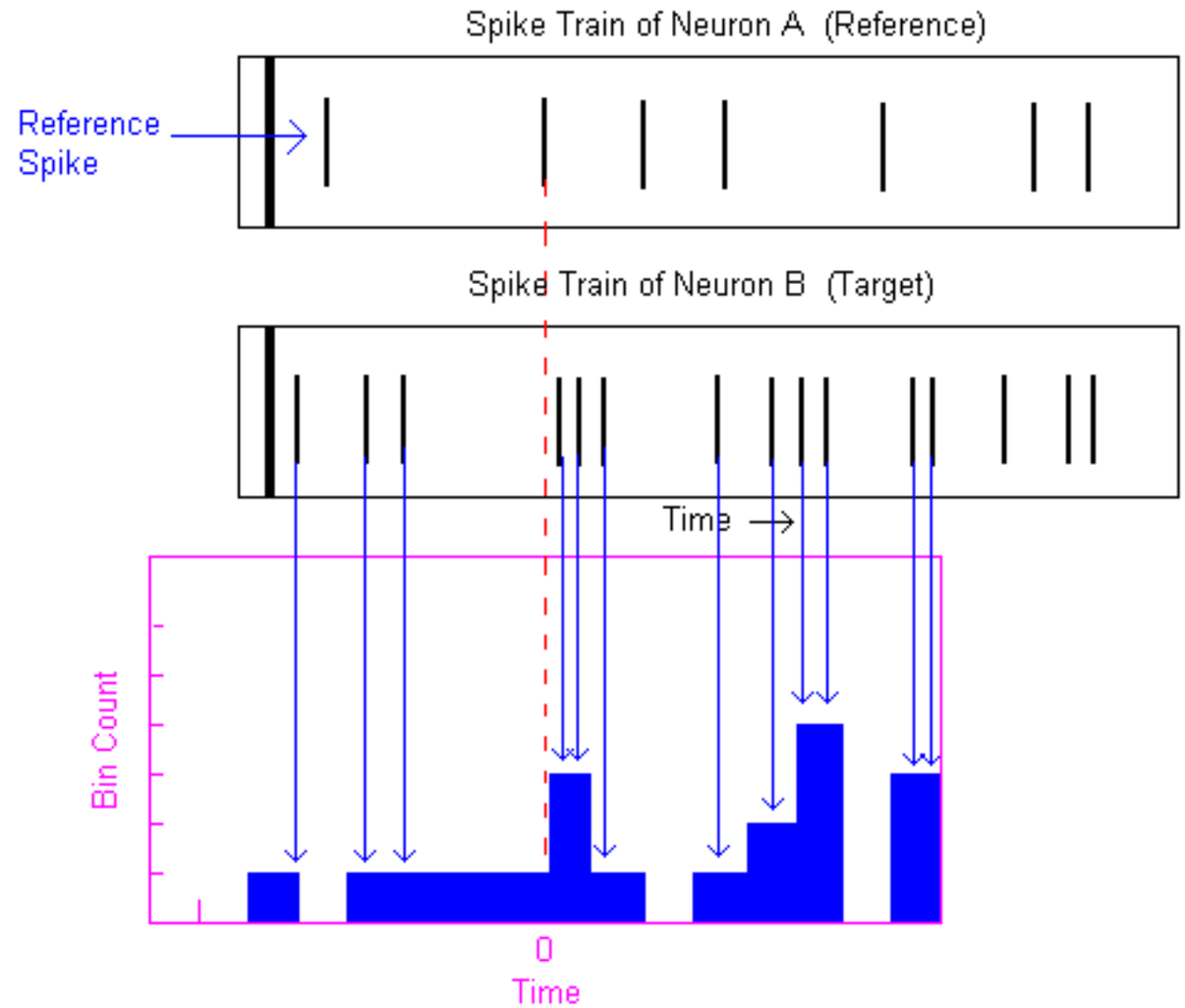
http://donoghue.neuro.brown.edu/library/CCH3.htm

Cross Correlogram



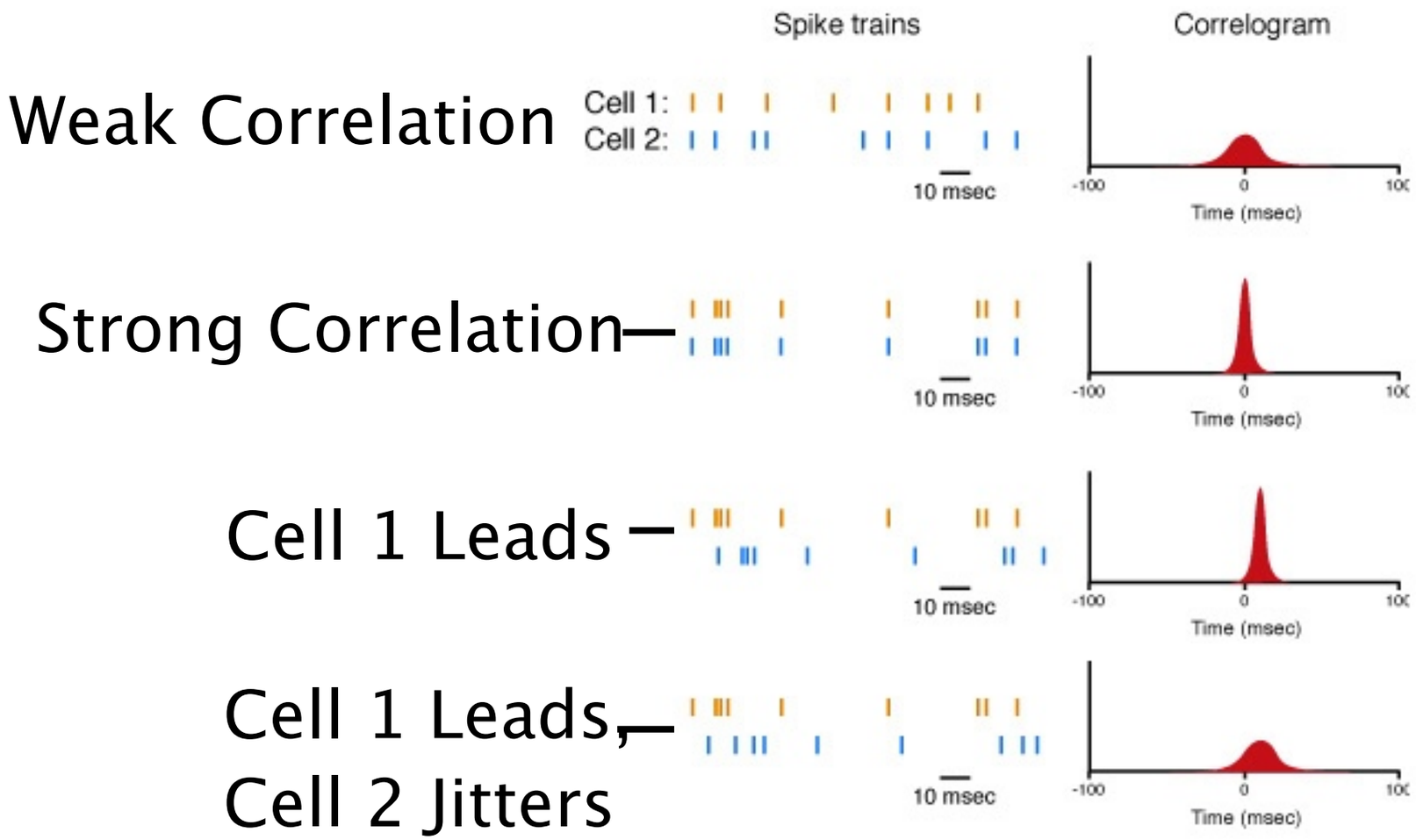
http://donoghue.neuro.brown.edu/library/CCH3.htm

Cross Correlogram



http://donoghue.neuro.brown.edu/library/CCH3.htm

How does spike timing affect cross-correlogram?



Source: http://www.psy.vanderbilt.edu/faculty/roeaw/edgeinduction/Fig-W3B.htm

Example

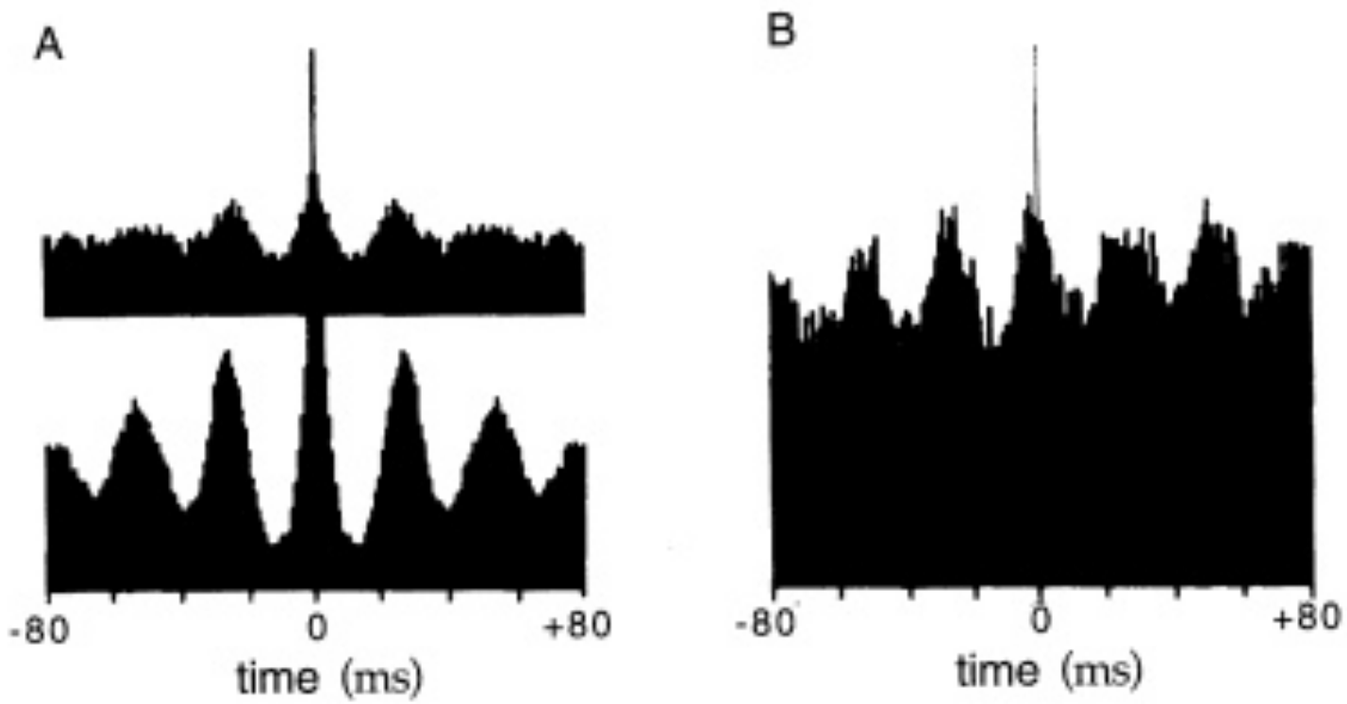


Figure 1.12 Autocorrelation and cross-correlation histograms for neurons in the primary visual cortex of a cat. (A) Autocorrelation histograms for neurons recorded in the right (upper) and left (lower) hemispheres show a periodic pattern indicating oscillations at about 40 Hz. The lower diagram indicates stronger oscillations in the left hemisphere. (B) The cross-correlation histogram for these two neurons shows that their oscillations are synchronized with little time delay. (Adapted from Engel et al., 1991.)

Source: Theoretical Neuroscience

Limitations

1. Analysis Bin Size is fixed
   - neuron firing is assumed to be stationary.
   - STDP can change temporal length for neuron excitability
2. CC only considers pairs of neurons, not populations
3. CC as measure of statistical dependence is only valid under certain conditions
4. Analysis limited to short windows – for longer windows it's impossible to rule out driving inputs

See: Statistical Signal Processing for Neuroscience and Neurotechnology

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

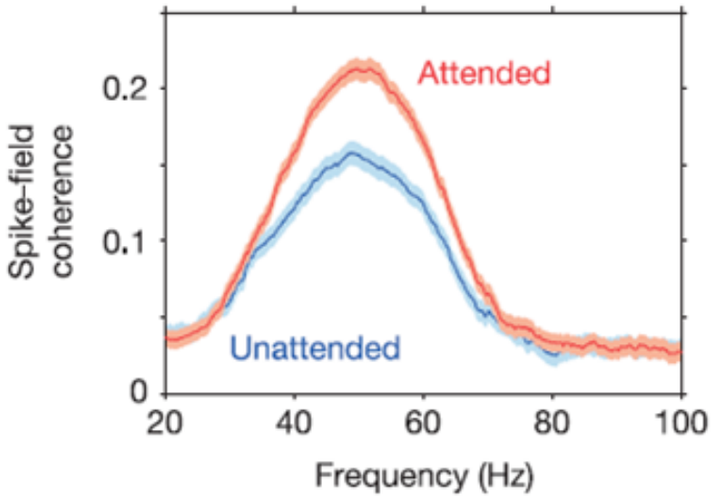
Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

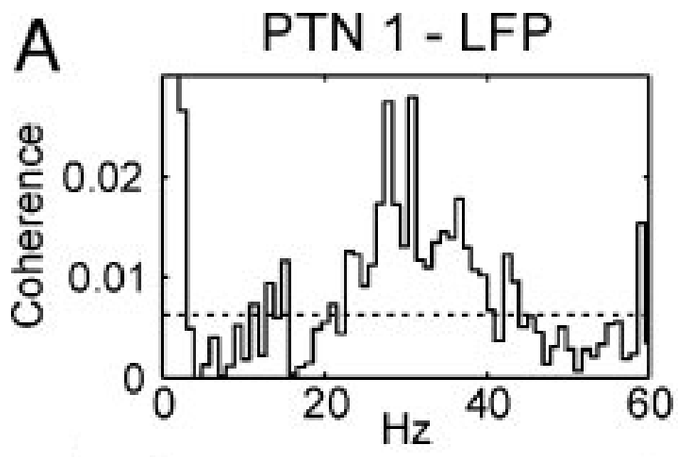
Spike-Field Coherence

- One way of looking at interactions between different brain regions is by looking at **spike-field coherence**
  - (Applies equally to looking at interactions between cells and their neighbors)
- Coherence between two signals: fraction of the variance at a given frequency in one signal that can be explained by the other signal
- Spike-field coherence provides an estimate of the proportion of the cell's output (spiking) that is synchronized with the local field potential



In visual cortex of monkey

From Chronux Manual, originally from Pascal Friehs



Macaque Pyramidal Tract Neurons during holding task (Baker, *J. Neurophys, 2003)*

- More background:

"Multiple neural spike train data analysis: state-of-the-art and future challenges", Brown, Kass, Mitra, *Nature Neuroscience*, 2004

"Synchronization in monkey motor cortex during a precision grip task. II. effect of oscillatory activity on corticospinal output", Baker, Pinches, Lemon, *Journal of Neurophysiology*, 2003

"Assessing Neuronal Coherence with Single-Unit, Multi-Unit, and Local Field Potentials", Zeitler, Fries, Gielen, *Neural Computation*, 2006

Spike-Field Coherence

Chronux (www.chronux.org) is a useful open-source MATLAB package for analyzing neural data

To use Chronux, download it and put on the MATLAB path

We will use the `coherencycpt` function to compute coherence between a continuous signal (LFP) and a point-process (spike train)

Chronux uses multitaper spectral decomposition; think of it as like Fourier Transform, but better at estimating the frequency content of a signal by smoothing across nearby frequencies

Optimally balances variance/bias of the estimate given a specified taper **T*W** product

T: how many seconds of data to use; e.g. .5 seconds

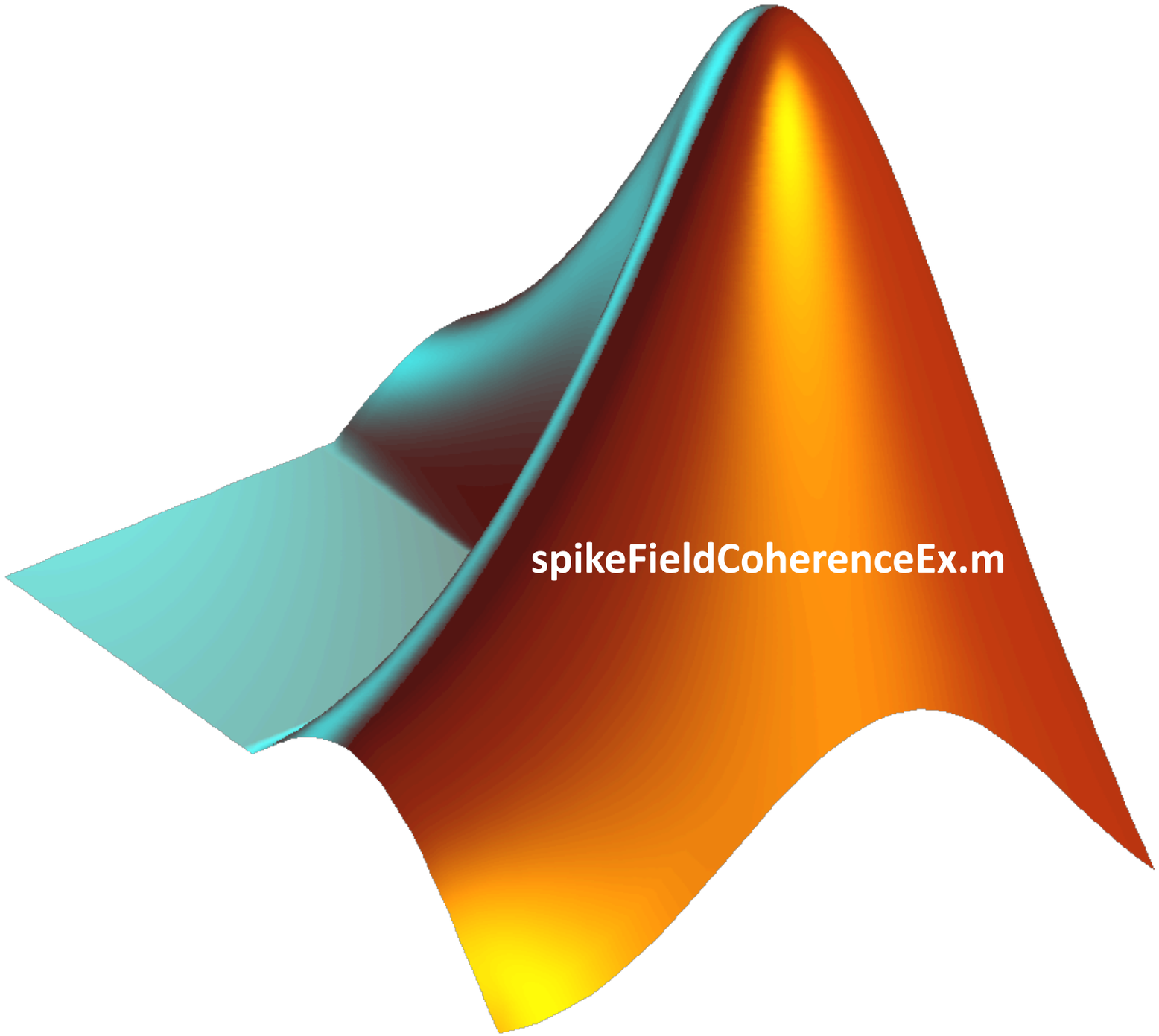
W: frequency resolution, e.g 5 Hz

The higher TW, the more accurate the estimate. But generally we want low T and W, hence a trade off
If we want a sliding window of T=0.5 seconds, more accurate estimate by setting W = 10 Hz (course resolution)

If we need a more precise frequency resolution and set W = 1 Hz, we'd have TW = 0.5
To have same accuracy as in T = 0.5 seconds, W = 10 Hz, , we'd need T = 5 seconds (thus losing temporal resolution)

T and W can be set as parameter when using `coherencycpt`

Demo: spike-field coherence



Spike-Field Coherence Limitations

Real neural data shows much lower coherences; 0.05 is often both significant and reflects considerable influence by the LFP signal on the neuron's output

Higher-rate neurons will show higher coherence with a continuous signal given the same actual shared input

Other things being equal, recording from more neurons will increase coherence

Coherence decreases with LFP frequency even if the true signal relationship is the same

Non-linearity of spiking invalidates strict interpretation of coherence as variance explained

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

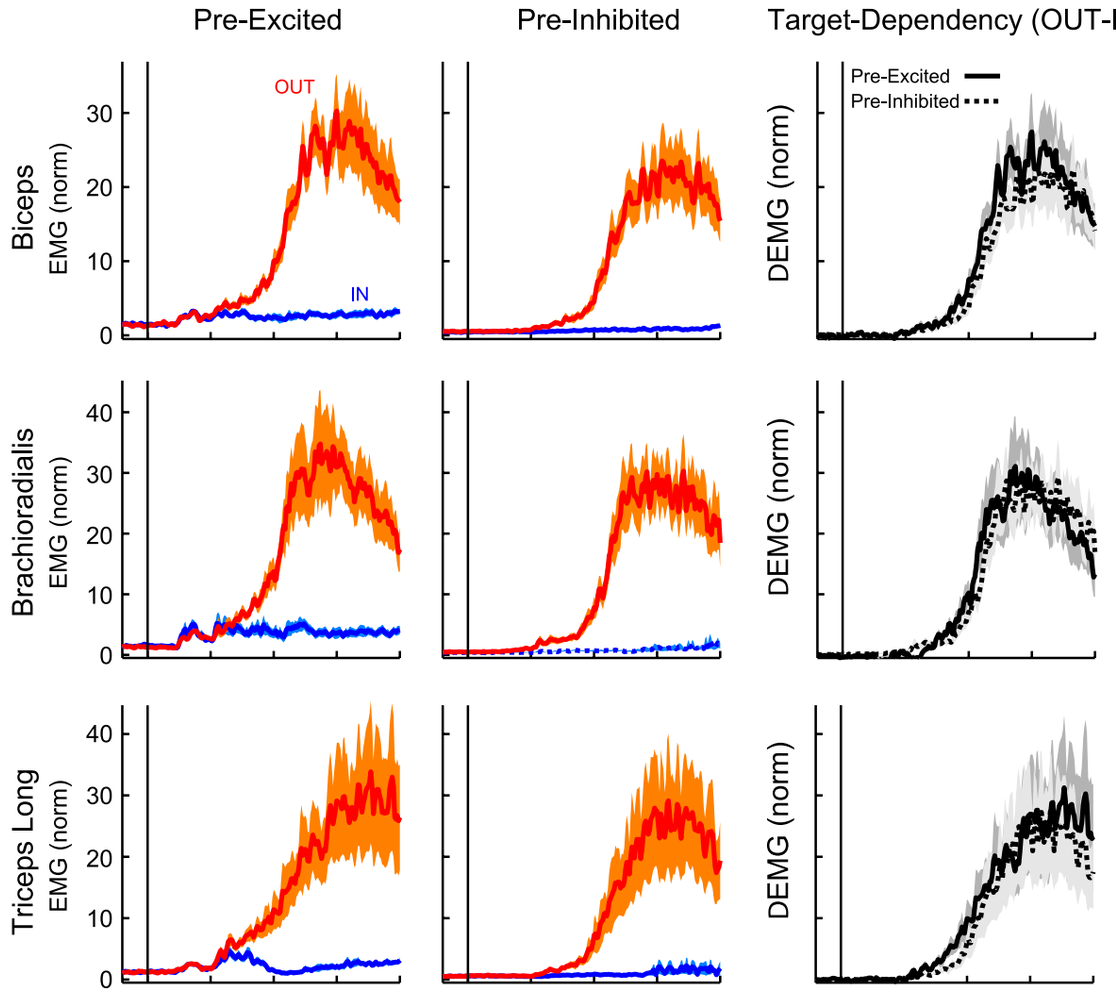
Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Shaded Variance Clouds

Often want to plot data with a shaded cloud surrounding it to represent uncertainty or range



(Pruszynski et al, J. Neurophysiology, 2011)

Fig. 5. Population muscle activation: *experiment 1*. Each row of panels presents the average muscle activation across subjects aligned on perturbation onset (black vertical line). Panels in the *left* and *middle* columns represent activation patterns when the muscle was preexcited and preinhibited, respectively. Panels in the *right* column plot the mean target-dependent activity calculated by subtracting the evoked response for the 2 target locations (OUT − IN) for both the excitatory (solid) and inhibitory (dashed) background loads. Note that target dependency is remarkably similar regardless of initial background load.

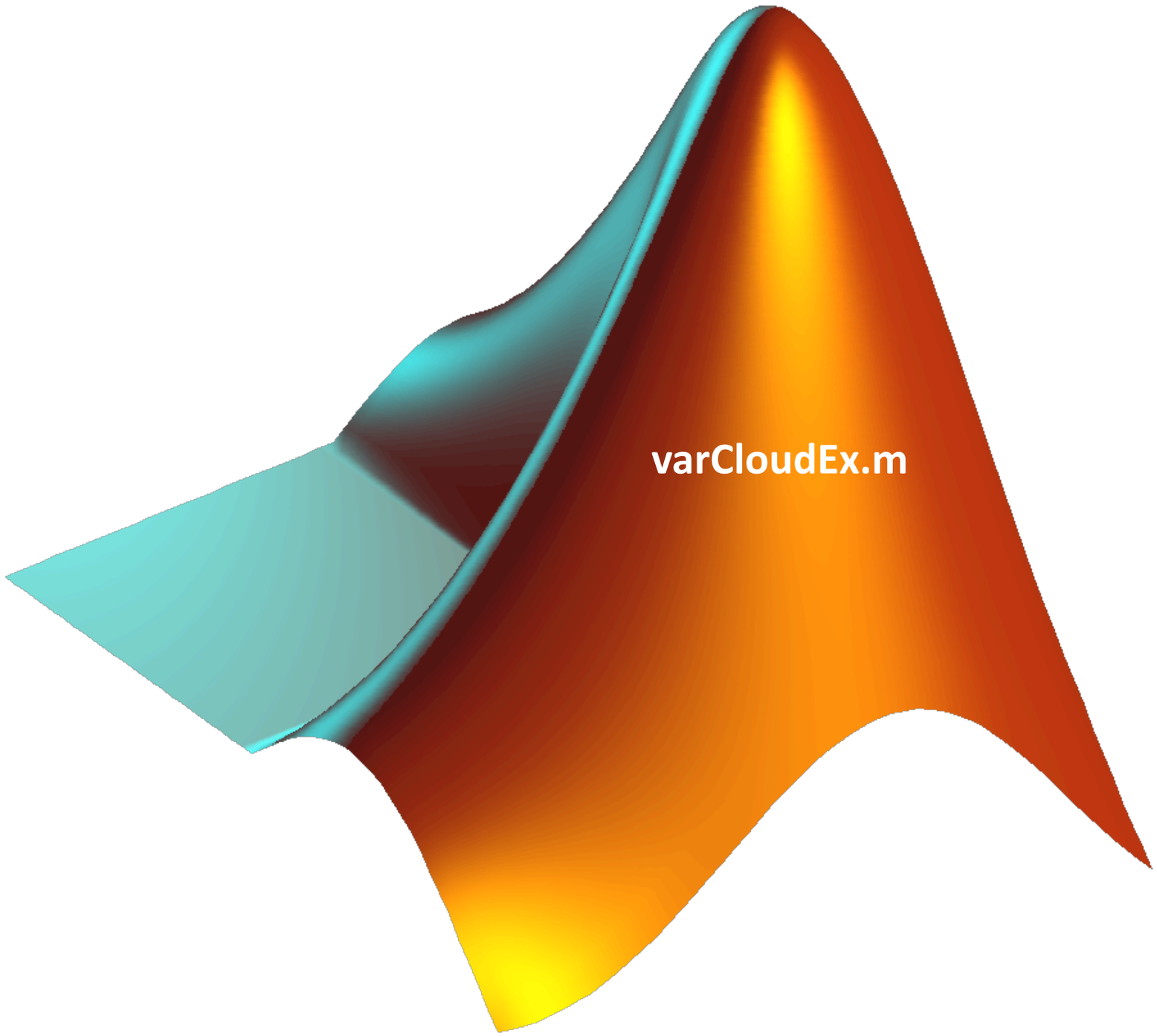
Do this using **patch( )**, with vertices at **mean(i) + std(i)** and **mean(i) - std(i)** for all plotted points i

Use a clockwise ordering for the vertices

fliplr or **flipud** are useful

FaceAlpha property of the patch should be less than 1

Demo: shaded variance clouds



Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

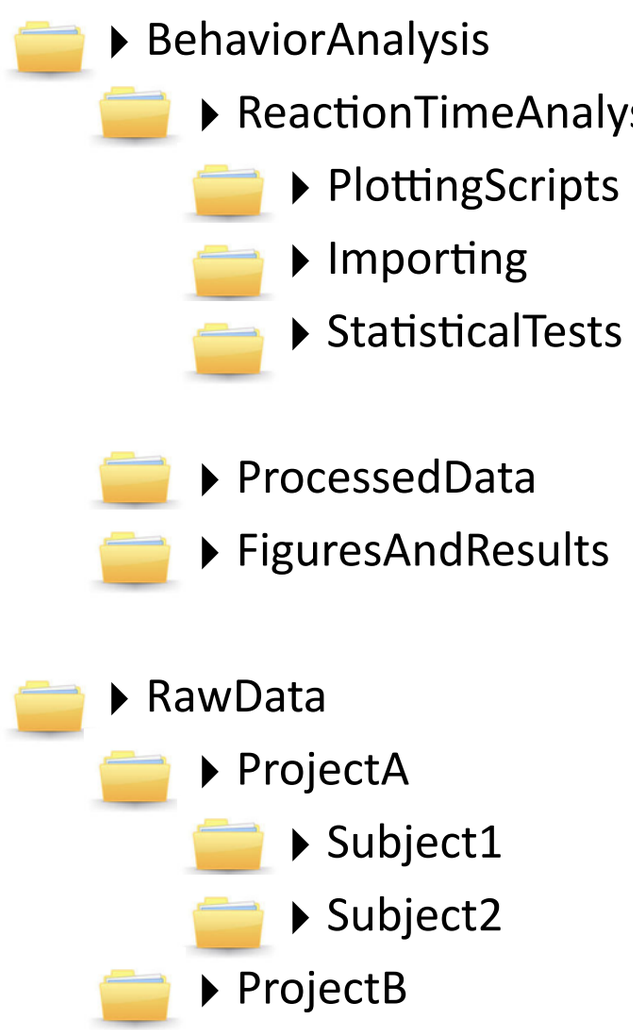
Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Organizing Your MATLAB Directories

One good organization strategy is separate top-level directories for **Raw Data** and **Analyse**s

Within a project, separate directories for code, processed data, and deliverables (figures, movies, results)



Code

Output

Raw Data (never modified!!)

Organizing data for later MATLAB processing

Generally speaking, organize records in a way that is easy on you when generating them

You can “do the work” later when reading your data in MATLAB

Excel is fine, as are various database softwares, or simple text files

Some rules of thumb which will make importing easier:

- Put format-identifying ID on first line/cell
  - E.g. start with header `1`. If a month later you change the number of columns, so now use header of `2`
  - Import function should first look at this header and call appropriate import function

  e.g. `importMyDataType1.m`, `importMyDataType2.m`
- Parsing through what you import is easier when there’s same type of data in a given row or column
- Avoid using a single spreadsheet as a canvas with lots of different things scattered throughout
  - E.g. Make a new worksheet for a new experiment
- Excel 1997-2003 file format is better supported on non-Windows MATLAB
- Can also export from Excel (and other software) in .csv format, which MATLAB reads with **csvread**

Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Simulation

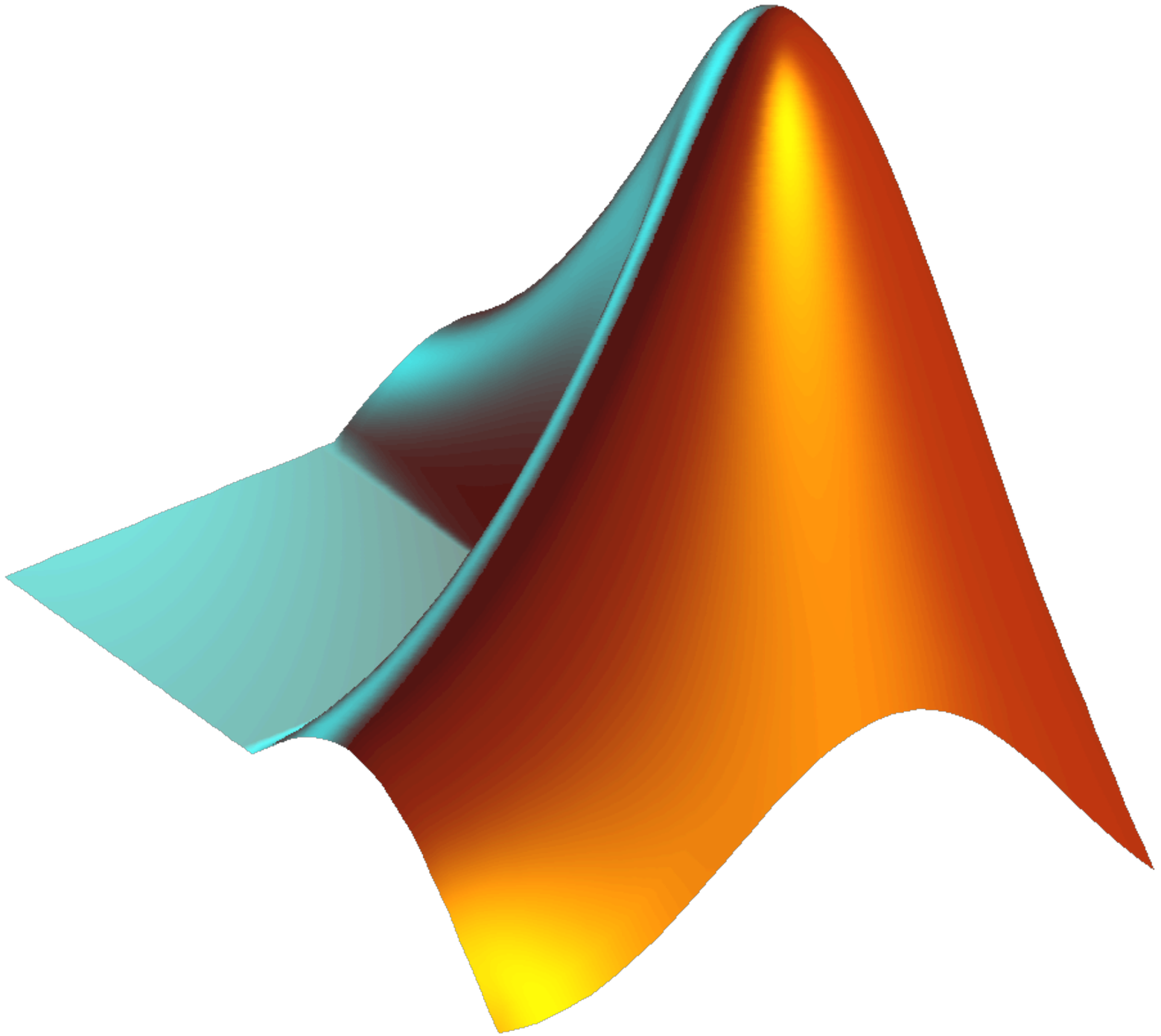
The goal is to simulate a leaky integrate and fire neuron:

$$\tau \dot{V} = -(V - V_{rest}) + R_m * I_{inj}$$

The method for doing this is called Euler integration:

$$V(t+1) = V(t) + \dot{V} * dt$$

Demo: I&F neuron simulation



Lecture 08 Outline

Questions?

Structures Versus Cell Arrays

Spike-Spike Interactions

Spike-Field Interaction

Plotting Transparent Confidence Regions

Organizing Your Work and Data

Modeling a Leaky Integrate-and-Fire Neuron

Curve fitting more types of functions

Curve fitting

Fitting a curve in MATLAB uses two basic tools:

`fittype()`

- Allows you to describe the form of the function you're trying to fit to your data
- Specify what the independent variables are
- Specify what the parameters to fit are

`fit()`

- Actually does the iterative fitting
- Specify upper and lower bounds for each parameter
- Specify a starting point for each parameter

Fit types

Linear relationship:

$$y = a*x + b$$

Use polyfit or regress rather than iterative methods

Linear regression

```
[p s] = polyfit(x, y, degree);
```

Use degree == 1 to fit a line, degree == 2 for a quadratic function. Can use polyval or polyconf to generate confidence intervals for where new points might lie.

```
[coeffs ci residuals residualIntervals stats] =
    regress(y, XWithOnes);
```

Can handle several predictors as columns in X. Make sure that X has a column of all ones in order to include a constant offset term. Tries to fit the model:

```
y(i) = coeffs(1)*x(i,1) + coeffs(2)*x(i,2) + ...
```

Make the last column all ones

Fit types

Sigmoidal relationship (binding curves):

```
y = mn + (mx-mn) / (1+ (x/e50)^(-hillExp))
```

Logistic relationship:

```
y = 1 / (1+exp(-ax-b))
```

Fit types

Exponential relationship:

$$y = a*\exp(k*x)$$

With no offset, you can see an exponential relationship if it appears linear on a logarithmic scale for the dependent variable.

$$\log(y) = k*x + \log(a)$$

Fit types

Exponential relationship:

$$y = a*exp(k*x) + b$$

With an offset, you can use an iterative fitting procedure.

```
fitModel = fittype('a * exp(k*x) + b',
                   'independent', 'x')
[fitResults gof] = fit(x, y,fitModel);
```

Bounds and initial guesses

While not strictly necessary, it's often a good idea to specify upper and lower bounds and an initial guess for each parameter.

```
fit(xData, yData, fitModel, ...
    'Lower', [minParam1 minParam2 ...], ...
    'Upper', [maxParam1 maxParam2 ...], ...
    'StartPoint', [initParam1 initParam2 ...]);
```

Fit results object

Results are stored in the fitResults object:

```
fitResults.a
fitResults.b
fitResults.k
```

You can evaluate the fit directly using the fitResultsObject:

```
xFit = 0:0.01:5;
yFit = fitResults(xFit);
```

Goodness of fit

The fields of the `goodnessOfFit` structure contain useful information:

Field	Value
`sse`	Sum of squares due to error
`rsquare`	Coefficient of determination
`adjrsquare`	Degree-of-freedom adjusted coefficient of determination
`stdError`	Root mean squared error (standard error)
`dfe`	Degrees of freedom

Confidence intervals

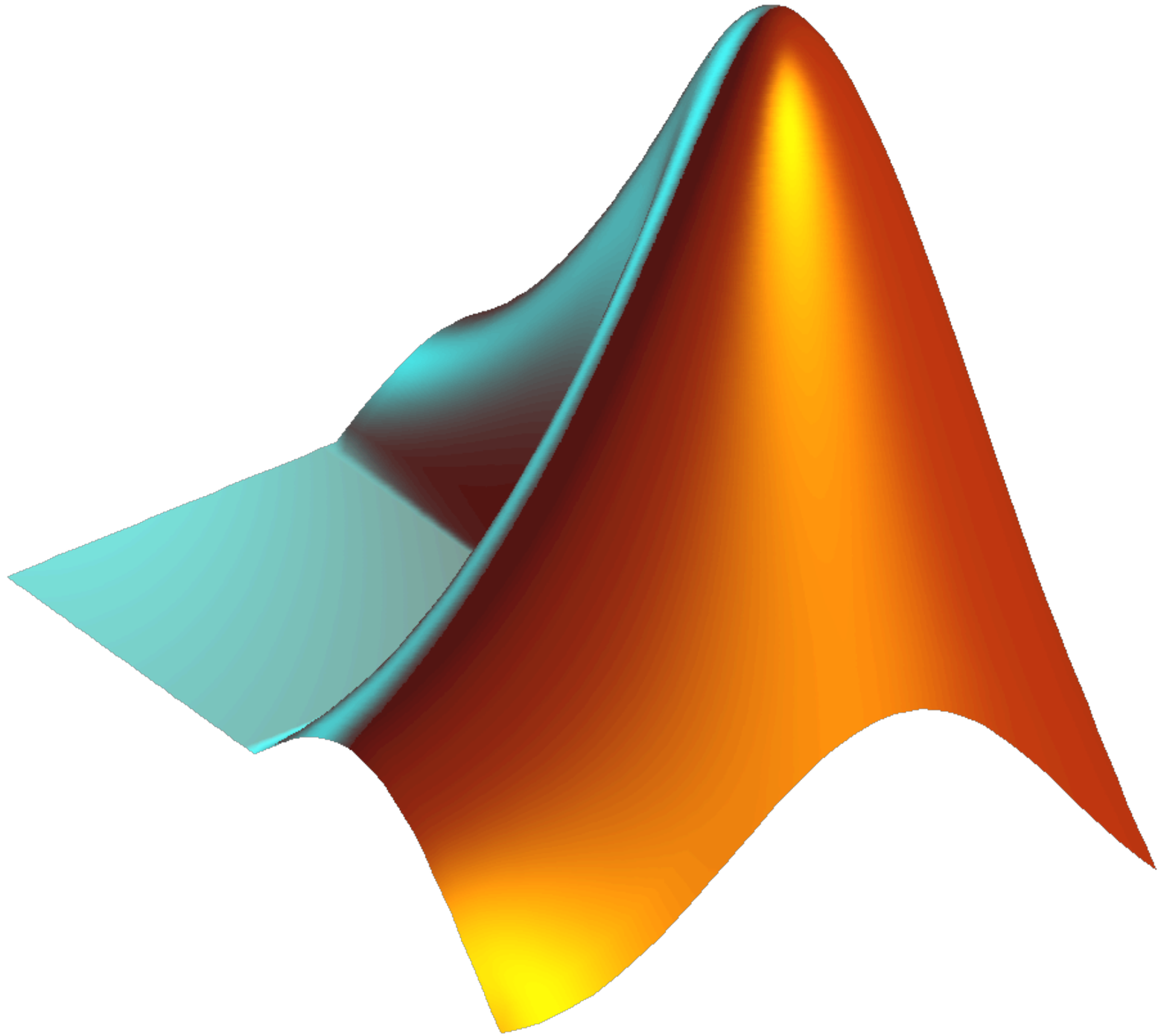
You can use `confint` to get confidence intervals around each fitted parameter:

```
paramsCI = confint(fitResults);
```

The confidence interval for parameter iParam (in alphabetical order) is given by:

```
paramsCI(:, iParam)
```

Demo: `cftool`



Lecture 08 Review

Key Concepts

Variance clouds can be made using patch and FaceAlpha < 1

When organizing your work, separate raw data, code, and processed data and results

Importing from excel or plain text files is easier if you keep the format consistent

Use a f**ile ID** in the header of your data file to specify how it should be imported

Coherence between two signals is a measure of correlation as a function of frequency

Chronux is a useful package for frequency-domain and coherence analysis of neural data

Functions

`fliplr`

`flipud`

`coherencycpt` (Chronux)