Lecture 09: Previews of Further MATLAB Power

NENS 230: Analysis Techniques in Neuroscience

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

Real-Time Simulink for experiment control

Graphical user interfaces and GUIDE

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

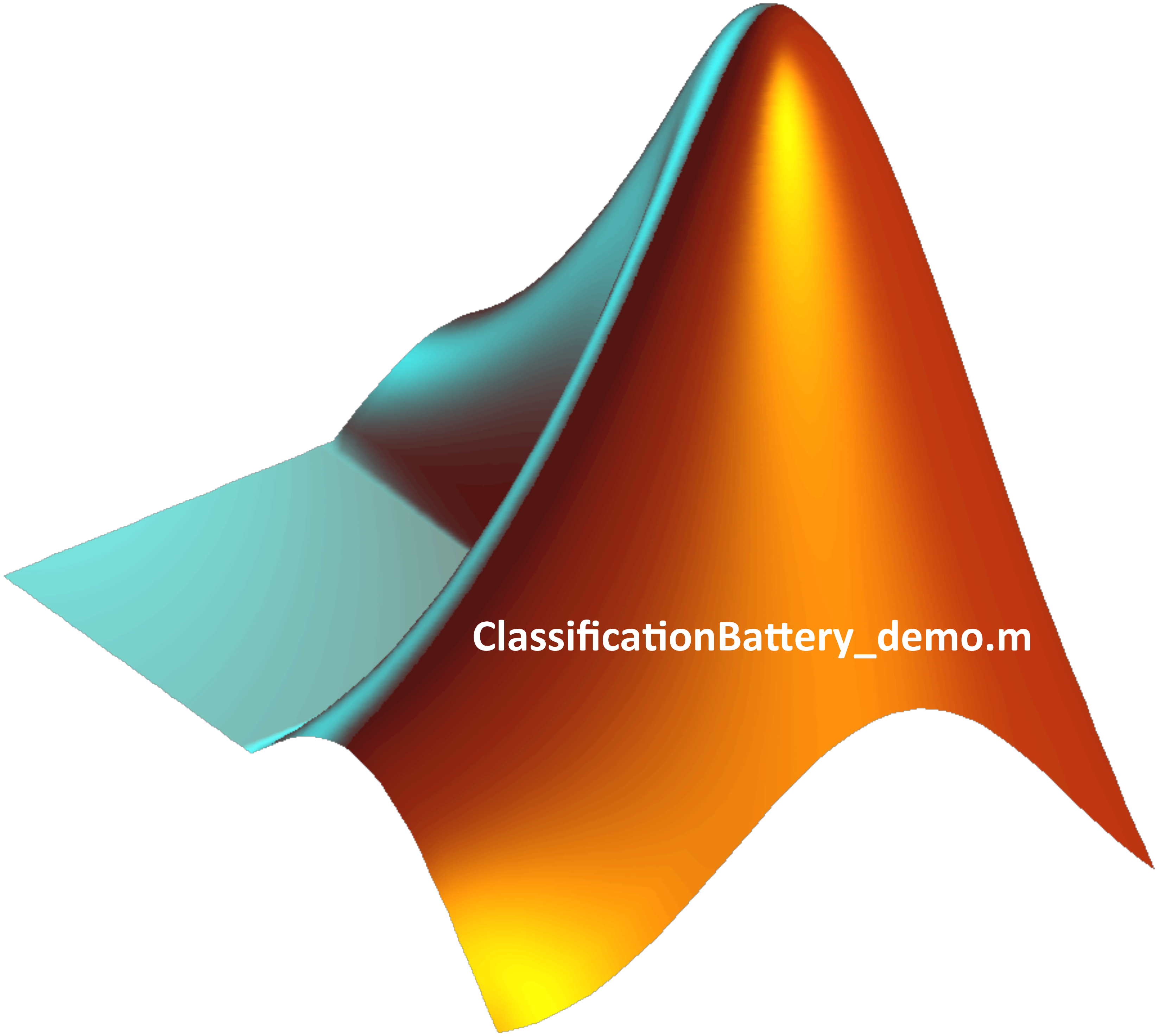
Real-Time Simulink for experiment control

Graphical user interfaces and GUIDE

PsychToolbox

- Free, Open-Source MATLAB package found at psychtoolbox.org (current version is PTB-3)
- Widely used to designed for **creating and presenting audiovisual stimuli during experiments**
- Allows you to use all of MATLAB, plus some functions for precise control of what gets displayed to the monitor
- Use it correctly and you have control of what goes up on the screen with every monitor refresh
- To use, download it and put onto your path
- Most PsychToolbox functionality used via the Screen( ) command
    - `Screen( ‘FillRect’ , coordinates, color )`
    - `Screen( ‘DrawLine’ , window, color, x),y0, x1, y1, width )`
    - `Screen( ‘DrawTexture’ , window, textureID )`
    - `Screen( ‘Flip’ , screenID, inHowManySeconds)`

Demo: PsychToolbox



ClassificationBattery_demo.m

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

Real-Time Simulink for experiment control

Graphical user interfaces and GUIDE

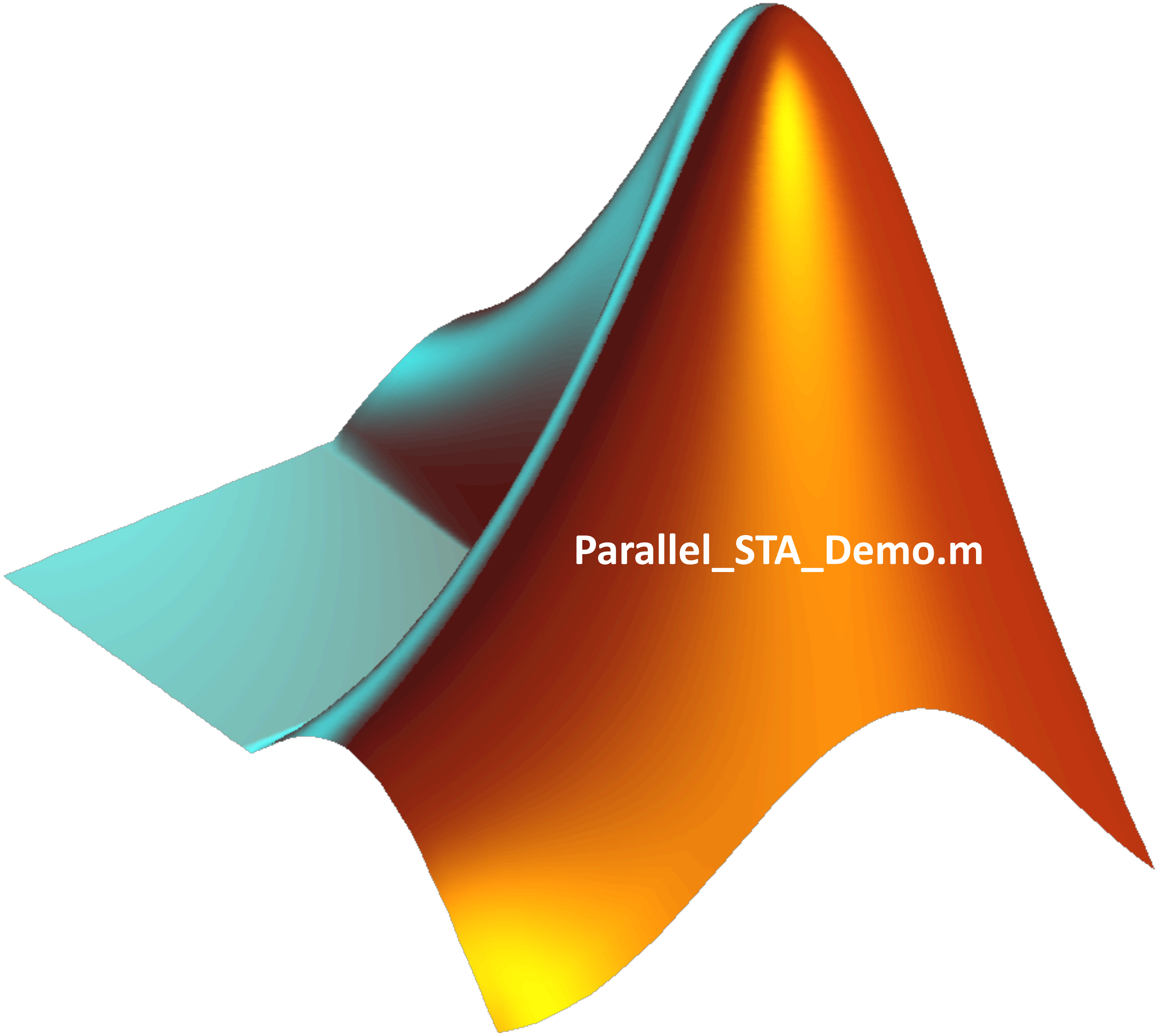
Parallelization

- Using the **Parallel Computing Toolbox** you can run code across multiple processors (cores)
    - Multiple cores of one machine (e.g. quad-core desktop processor)
    - Even more cores across multiple machines all under control of one MATLAB instance

- Easy to use: `matlabpool( numCores )` and then replace **`for` loops** with **`parfor` loops**

- Some restrictions about what can go inside a parfor loop:
    - Cannot have interactions between variables manipulated by each loop iteration
    - Best to parallelize at the outermost level because there is communication overhead

Demo: Parfor



Parallel_STA_Demo.m

Even Faster Performance

- Parallel Computing Toolbox also allows you to:
  - Distribute an array (e.g. matrix) over multiple computers (gets around memory limitations)
  - Use graphics card to do matrix operations even faster (GPUs contain hundreds of small, specialized processors)

- Compile pieces of code in the **C** language into **mex functions** that MATLAB m-code calls

- Compile MATLAB code into stand-alone programs for distribution (need MATLAB Compiler)

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

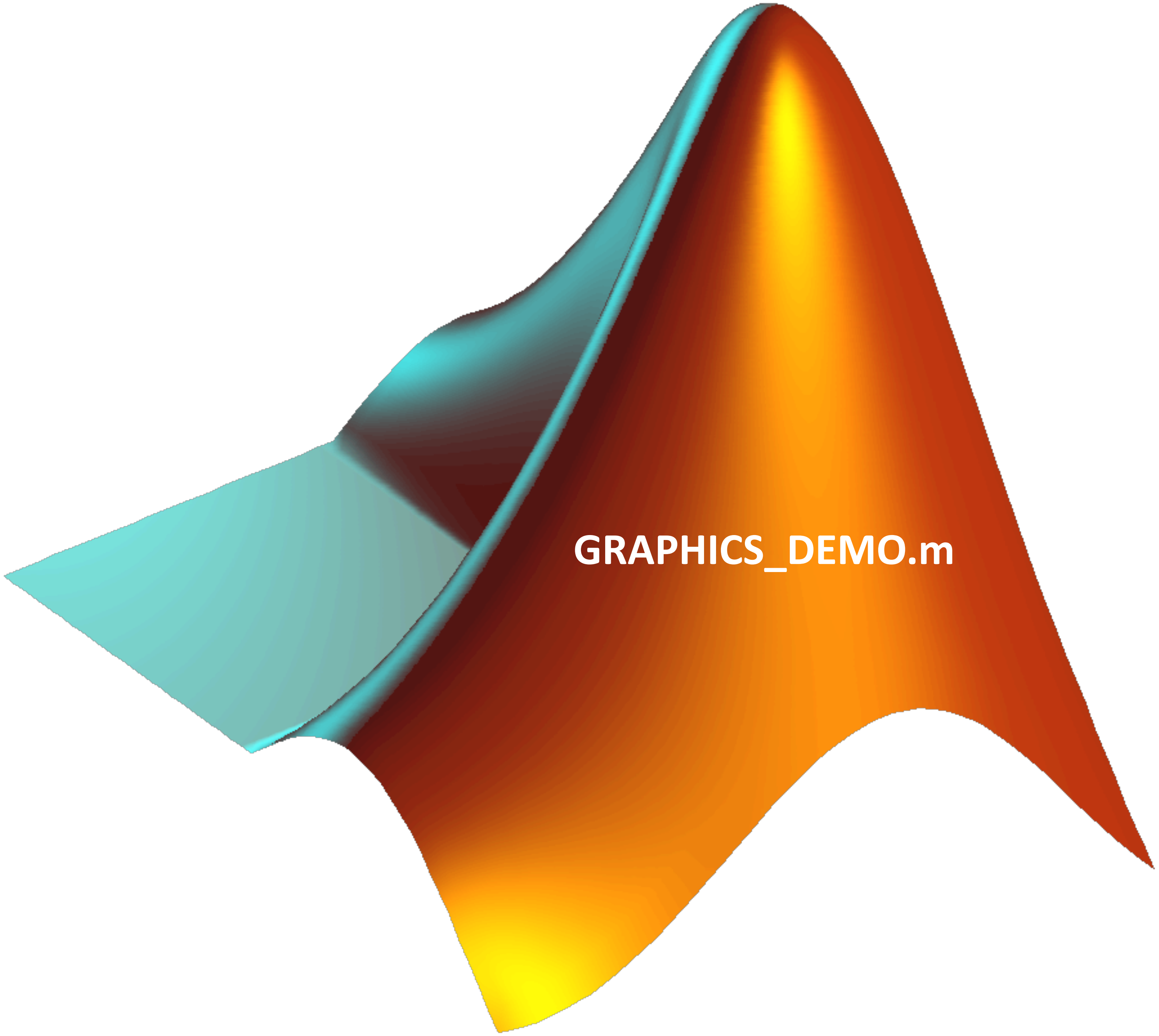
Real-Time Simulink for experiment control

Graphical user interfaces and GUIDE

3D Graphics Game Engine

- MATLAB can take advantage of the OpenGL graphics library to do things like lighting, shading, 3D polygons
- This example demos a simple “game” (used for neural prosthetics experiments) written in MATLAB

Demo: 3D Graphics Game Engine



GRAPHICS_DEMO.m

3D Graphics Game Engine

- MATLAB can take advantage of the OpenGL graphics library to do things like lighting, shading, 3D polygons
- This example demos a simple “game” (used for neural prosthetics experiments) written in MATLAB
- This program uses **Object-Oriented MATLAB**:
    - A different way of programming where you define **classes** which contain **properties** (data) and **methods** (functions that they can do)
    - A principled way of organizing code and doing abstraction
    - Has an **event-listener** framework which lets functions be executed whenever a defined event happens anywhere else
    - Uses **timer objects** to execute certain functions at specific times

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

Real-Time Simulink for experiment control

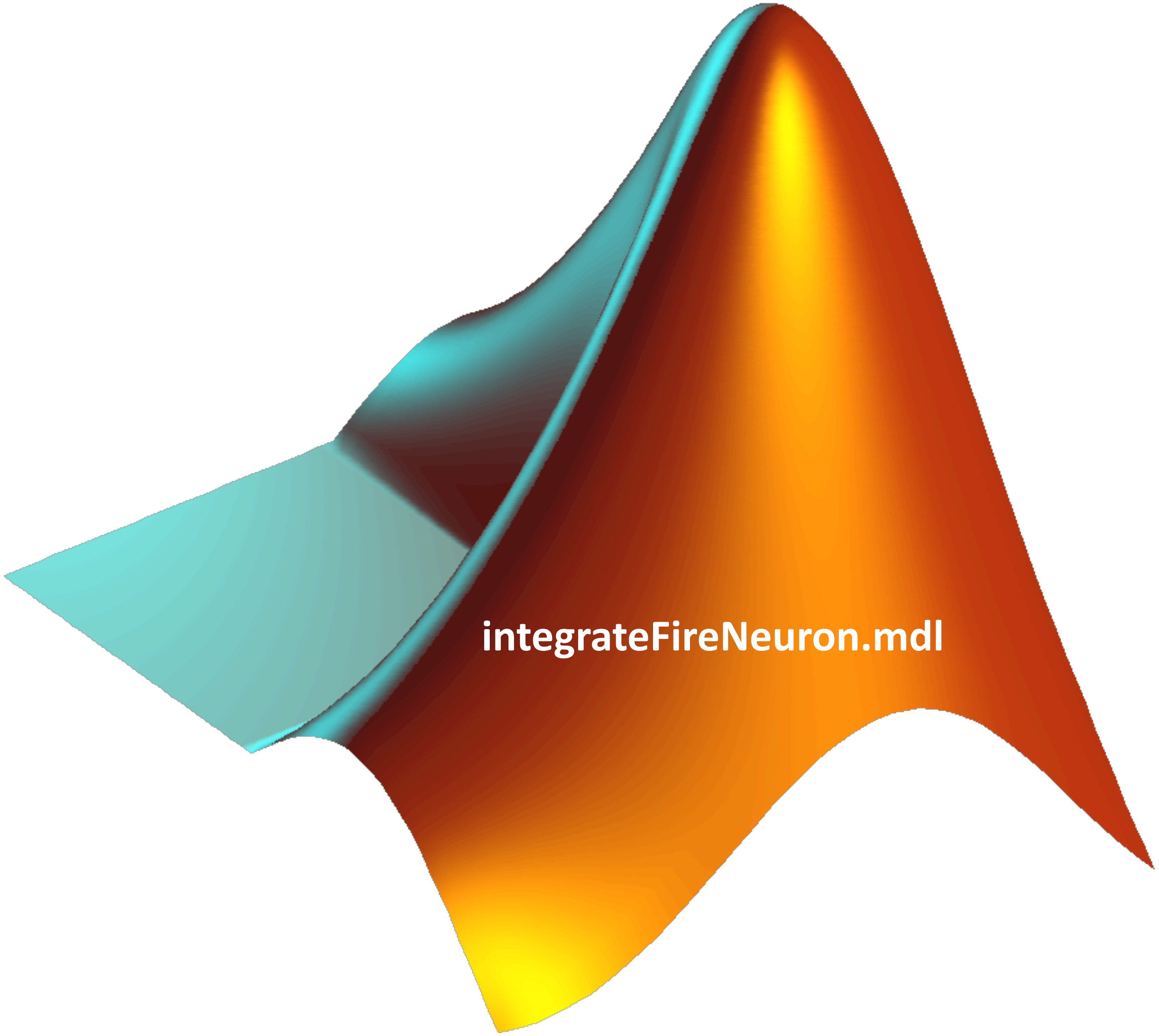
Graphical user interfaces and GUIDE

Simulink

- Simulink is a graphical programming language included with MATLAB
- The typical use is for simulating dynamical systems

- You create models containing blocks, which represent operations, and connect them with lines (signals)
- A block can also be MATLAB code (so you can create any operation you need)
- When the model is run, Simulink takes care of optimally solving the differential equations to implement the logic you've drawn out

Demo: Simulink for Modeling



integrateFireNeuron.mdl

Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

Real-Time Simulink for experiment control

Graphical user interfaces and GUIDE

Simulink for Experiment Control

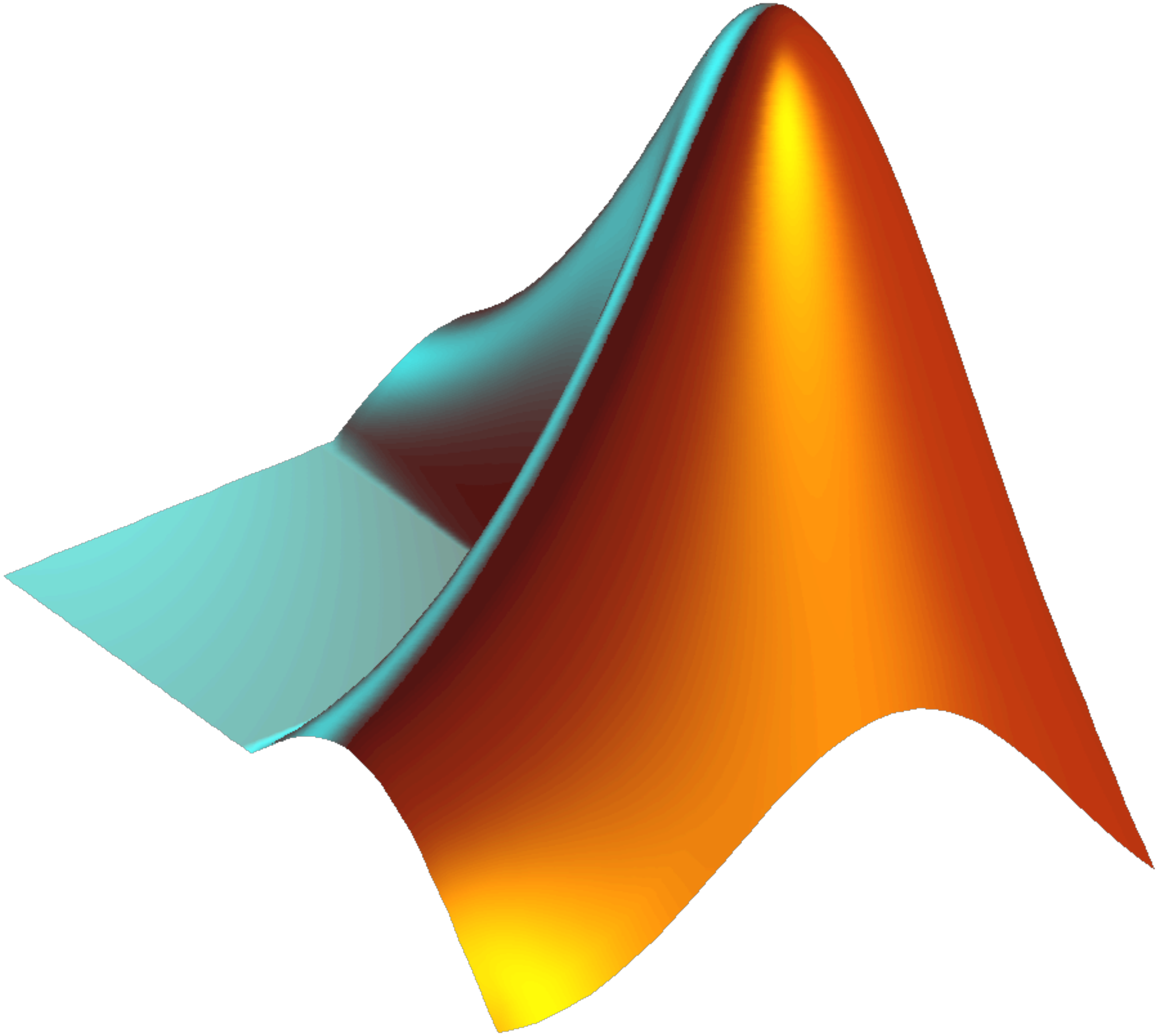
Simulink can also be used to control and record data from various experimental equipment.

- xPC Target is a set of tools for running Simulink models in guaranteed real-time on a dedicated PC.
- Stateflow allows you to express experimental flows as a state diagram

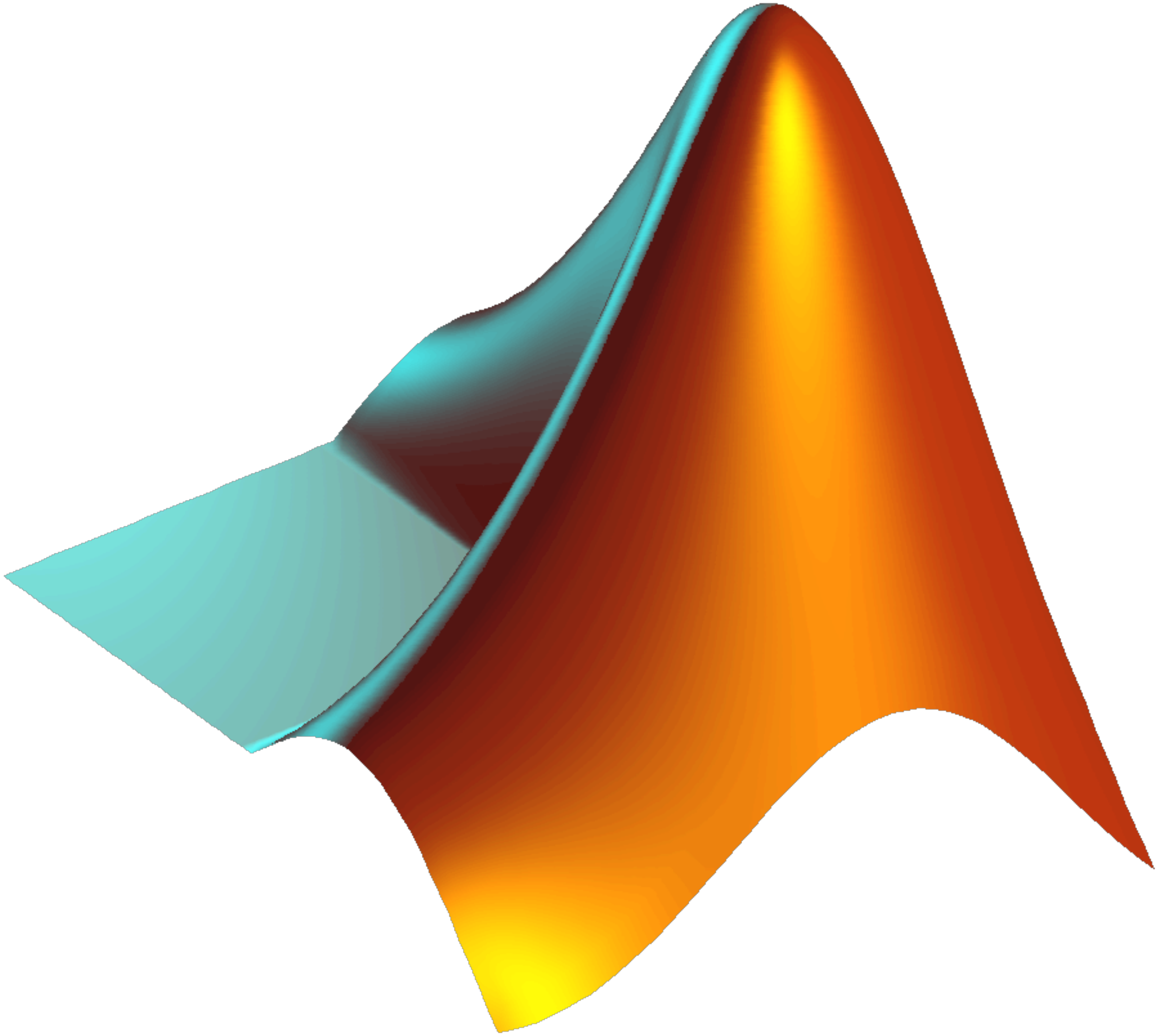
Simulink has blocks which facilitate interaction with various pieces of hardware

- Network interface cards
- Serial and parallel cable I/O
- National instruments data acquisition (NI-DAQ) card

Demo: Simulink Laser Driver



Demo: Stateflow diagram



Lecture 09 Outline

PsychToolbox

Faster Performance with Parallel Computing and Compilation

Graphics: 3D Game Engine Example

Simulink

Real-Time Simulink for experiment control

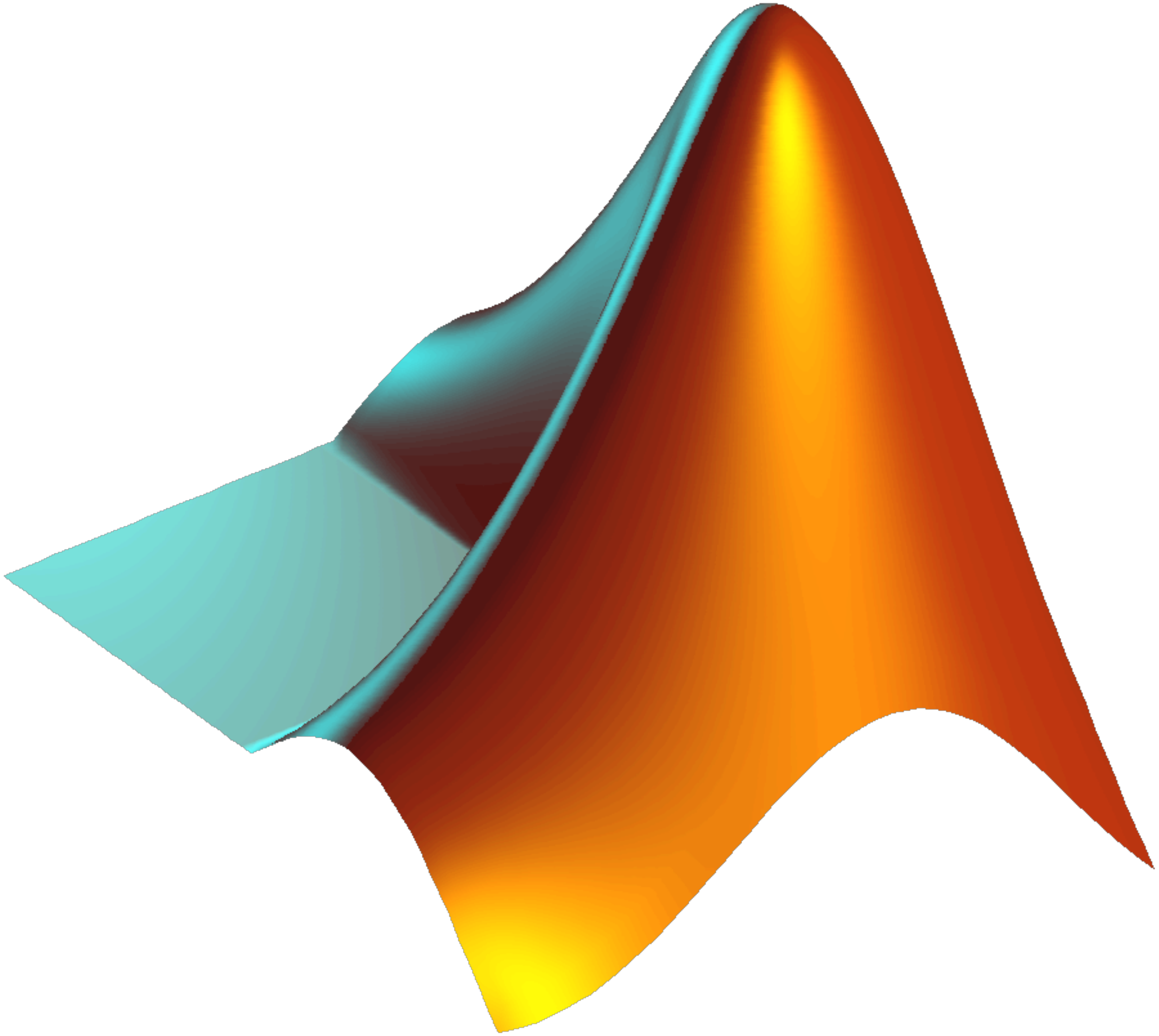
Graphical user interfaces and GUIDE

Graphical User Interfaces

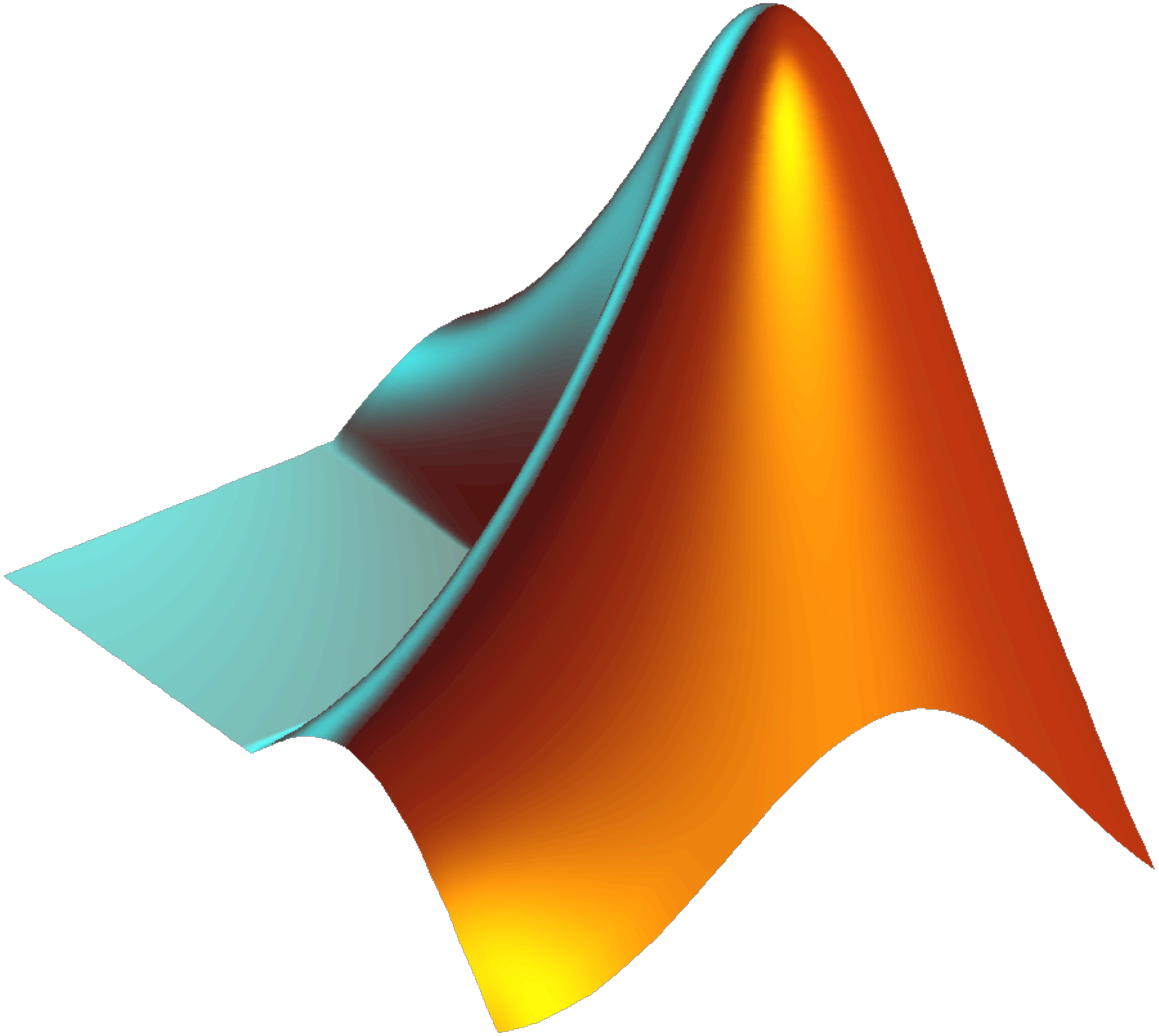
MATLAB provides tools for interactive GUIs

- Interaction is mediated through the use of **callback functions** and `uicontrol` objects
- Data storage is facilitated by the `guidata` function
- **GUIDE** is a tool for drawing controls on the figure visually and generating starter callback functions

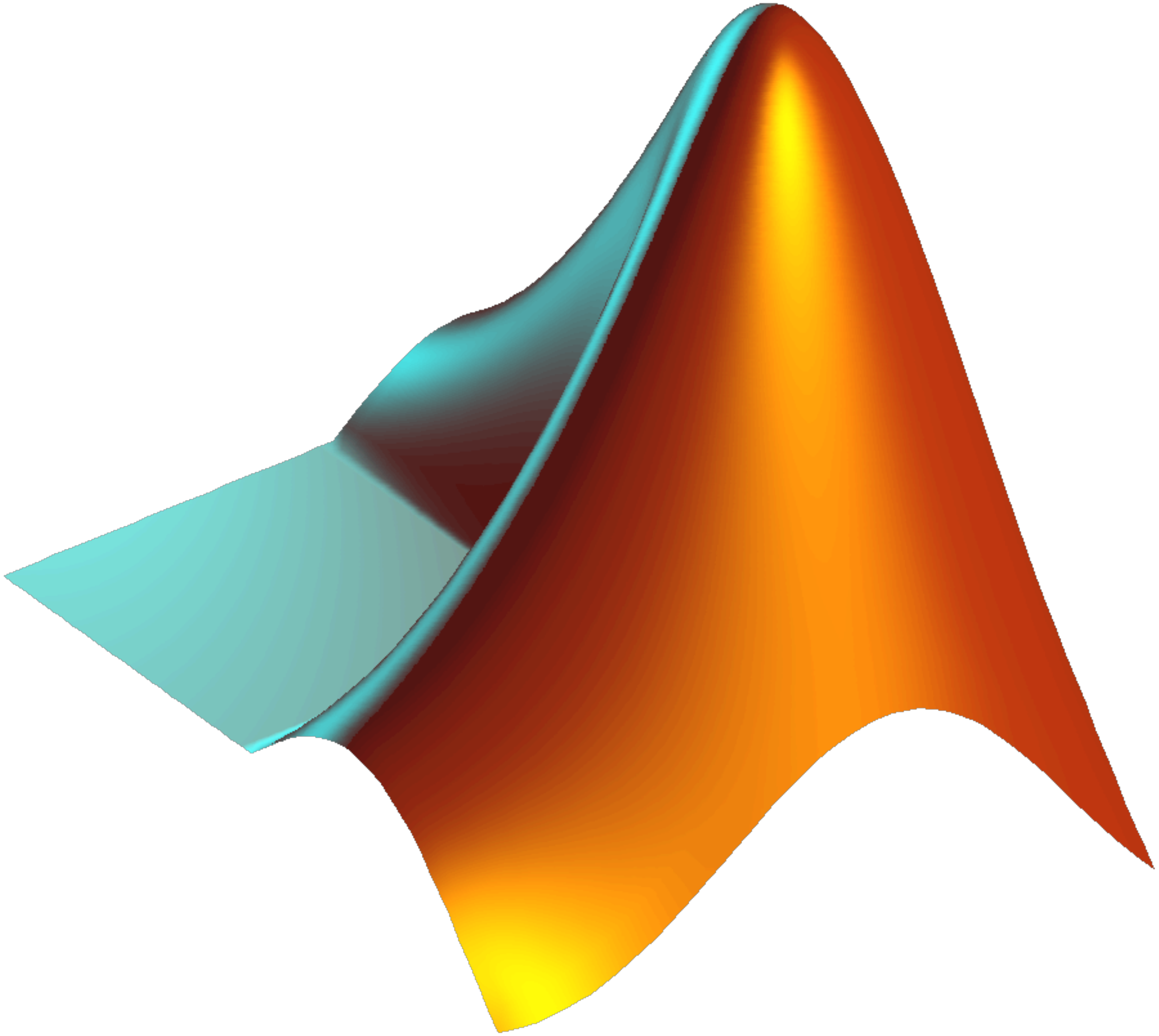
Demo: Drag and Drop



Demo: Spike sorting tool



Demo: Spike sorting tool



NENS 230 Topics Covered

Representing and Indexing Data: Variables, Vectors, Matrices, Multidimensional Matrices, Cell Arrays, Structures

Manipulating Variables with Concatenation, Excision, Comparisons

Scripts, Functions, and Scope

Controlling Execution Flow with Loops and Conditional Statements

Reading, Manipulating, and Generating Strings

Importing data from text, Excel, generic csv data, and proprietary files (e.g. abf)

Making figures, axes, and graphics objects; controlling them through object handles and properties

Movies, multi-axis figures, 3D figures, manipulating images

Statistics review, and MATLAB implementation of common statistics and hypothesis tests

ANOVA, Principal Component Analysis, regression, and curve fitting

Good coding style and writing flexible functions

Measuring and improving computation speed and memory usage

Spike-spike interactions (ISI Histogram, Cross/Autocorrelogram) and Spike-Field Coherence

Simulating an integrate-and-fire neuron

NENS 230

Thank You!