Introduction to Matlab

Patrice Koehl

http://www.cs.ucdavis.edu/~koehl/

koehl@cs.ucdavis.edu

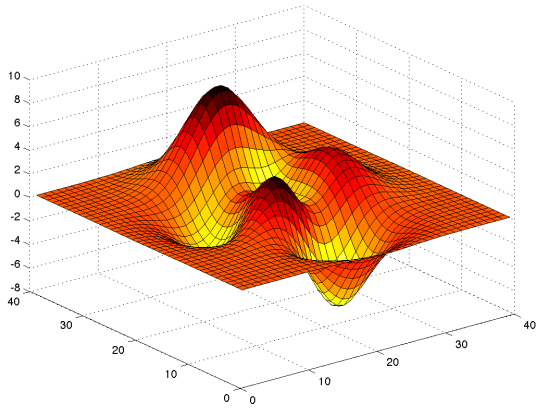
What is MATLAB?

- A high-performance language for technical computing (Mathworks, 1998)
- The name is derived from MATrix Laboratory
- Typical uses of MATLAB
  - Mathematical computations
  - Algorithmic development
  - Model prototyping
  - Data analysis and exploration of data (visualization)
  - Scientific and engineering graphics for presentation

Why Matlab?

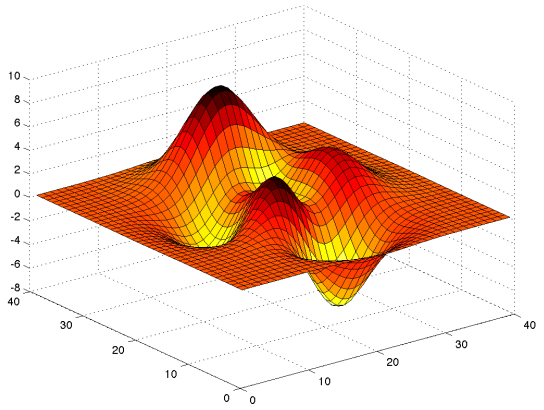
- Because it simplifies the analysis of mathematical models
- It frees you from coding in high-level languages (saves a lot of time - with some computational speed penalties)
- Provides an extensible programming/ visualization environment
- Provides professional looking graphs

Matlab



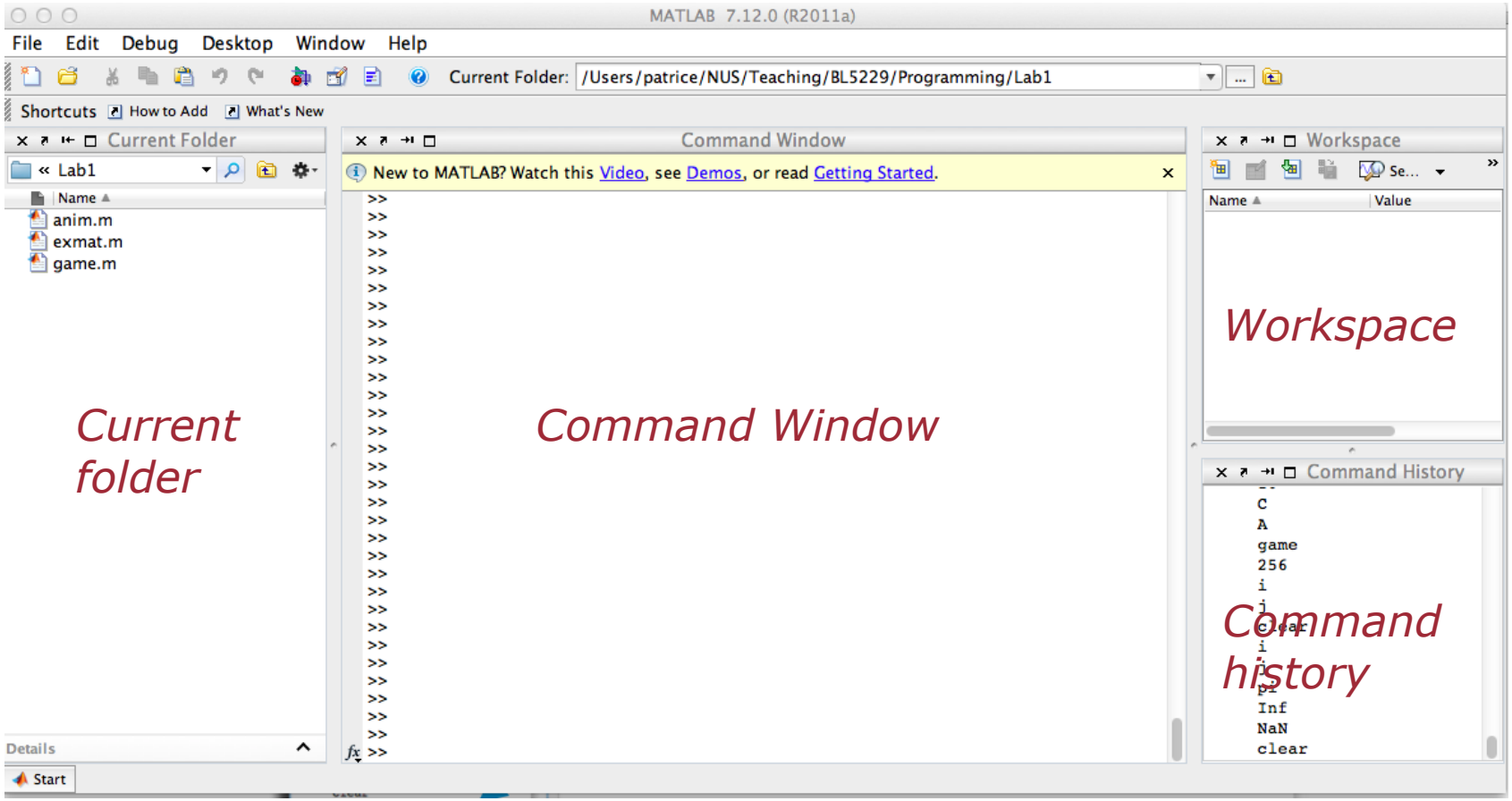
- The Matlab Environment
- Variables; operations on variables
- Programming
- Visualization

Matlab

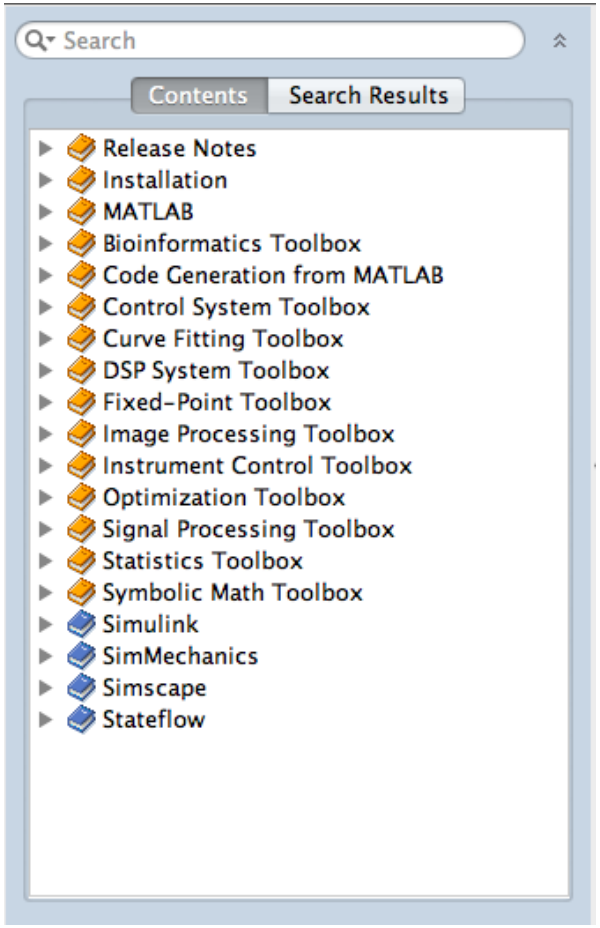


- The Matlab Environment
- Variables; operations on variables
- Programming
- Visualization

The Matlab Environment



Help in Matlab



Help Browser

-> Product Help

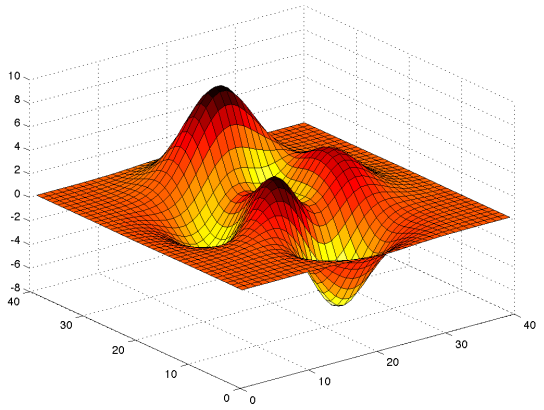
Command line:

>> help <command>

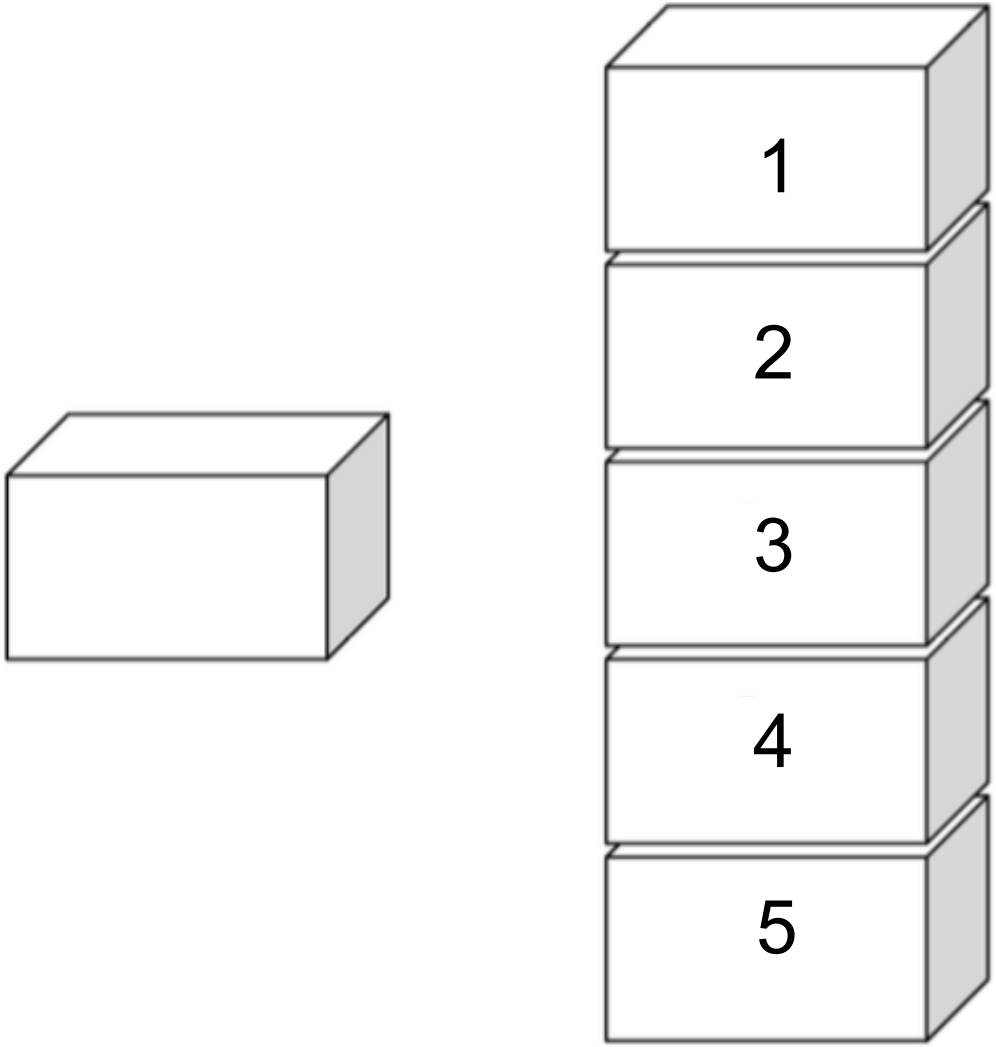
Example:

>> help sqrt

Matlab



- The Matlab Environment
- Variables; operations on variables
- Programming
- Visualization



Scalar variable:
One storage box

Array:
"chest of drawers"

Variables in Matlab

- Begin with an alphabetic character: a
- Case sensitive: a, A
- No data typing: a=10; a='OK'; a=2.5
- Default output variable: ans
- Built-in constants: **pi i j Inf**
- **clear** removes variables
- **who** lists variables
- **whos** list variables and gives size
- Special characters : **[] () {} ; % : = ... @**

Vectors in Matlab

- Row vectors

```
>> R1 = [1 6 3 8 5]
>> R2 = [1 : 5]
>> R3 = [-pi : pi/3 : pi]
```

- Column vectors

```
>> C1 = [1; 2; 3; 4; 5]
 >> C2 = R2'
```

Matrices in Matlab

- **Creating a matrix**

```
>> A = [1 2.5 5 0; 1 1.3 pi 4]
>> A =  [R1; R2]
>> A = zeros(10,5)
>> A = ones(10,5)
>> A = eye(10)
```

- **Accessing elements**

```
>> A(1,1)
>> A(1:2, 2:4)
>> A(:,2)
```

Matrix Operations

- **Operators + and –**

```
>> X = [1 2 3]
>> Y = [4 5 6]
>> A = X + Y
   A=   5  7   9
```

- **Operators *, /, and ^**

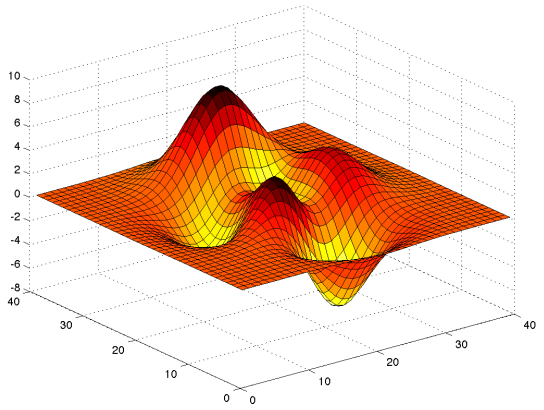
```
>> Ainv = A^-1 Matrix math is default!
```

Element wise operations

Operators .*, ./, and .^

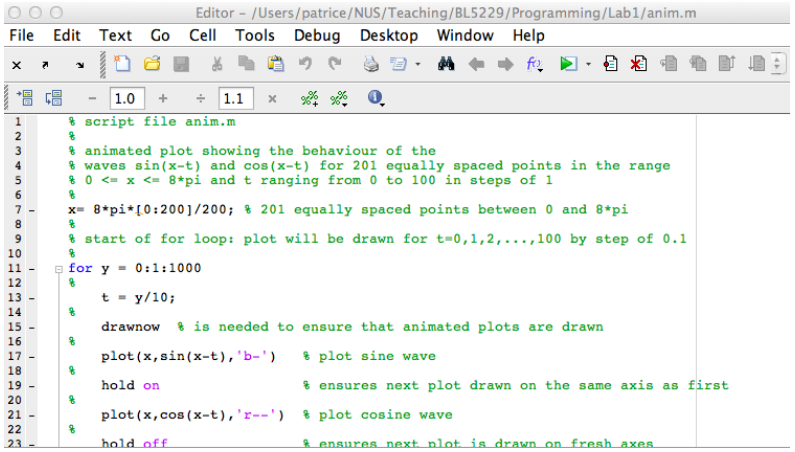
```
>> Z = [2 3 4]'

>> B = [Z.^2 Z Z.^0]

B=    4   2   1
      9   3   1
     16   4   1
```

Matlab



- The Matlab Environment
- Variables; operations on variables
- Programming
- Visualization

M-file programming



➢Script M-Files

- Automate a series of steps.
- Share workspace with other scripts and the command line interface.

➢Function M-Files

- Extend the MATLAB language.
- Can accept input arguments and return output arguments.
- Store variables in internal workspace.

M-file programming

- **Always one script M-File**
- **Uses built-in and user-defined functions**
- **Created in MATLAB Editor**

```
>> edit model.m
```

- **Run from Command Line Window**

```
>> model
```

Example of script

Example: model.m

```
% Define input
T = [ 0 : 0.01 : 30]

% Compute model
Y = exp(-T);

% Plot model
plot (T, Y);
```

Example of function

Example: amodel.m

```
function Y = amodel(t, A, B, a, w, p)
% H1 line: AMODEL computes step response.
% Help text: appears when you type
% "help amodel" in command line window.

% Comment: function body is below.
Y = A * exp(-b.*t).*cos(w.*t + p) + B;
```

Input / Output

- Get input from command window:

```
>> num = input('What is the altitude :')
>> str = input('Enter name of the planet','s')
```

- Display output in command window:

String

```
>> disp('The answer is:')
```

String + number:

```
>> disp(['The value of x is: ' num2str(x)])
```

Operators

- Arithmetic: `x+y;` `A*B;` `X.*Y;` etc.
- Logical
  - Element-wise AND: `a & b`
  - Element-wise OR: `a | b`
- Relational

  `a == 5; a >= b; b ~= 6;`
- Operator precedence

  `()` `{}` `[]` -> Arithmetic -> Relational -> Logical

Program flow control: For

Simple program that sums the squares of all the elements of a matrix A:

```
N = 10;
M = 20;

A = rand(10,20)

Sum = 0;
for i = 1:N
    for j = 1:M
        Sum = Sum + A(i,j)^2;
    end
end
```

Note that this can be done in one line:

```
Sum2 = sum(sum(A.*A));
```

Program flow control: if

Simple program that compares two numbers a and b: set j to 1 if a>b, -1 if a<b, and 0 if a = b:

```
if a > b
    j = 1;
else if a < b
    j = -1;
else
    j = 0;
end
```

Program flow control: switch

Simple program that reads in an integer number, checks if it is -1, 0, 1, or another number

N = input('Enter an integer number: ')

switch N

case -1

disp('negative one')

case 0

disp('zero')

case 1

disp('positive one')

otherwise

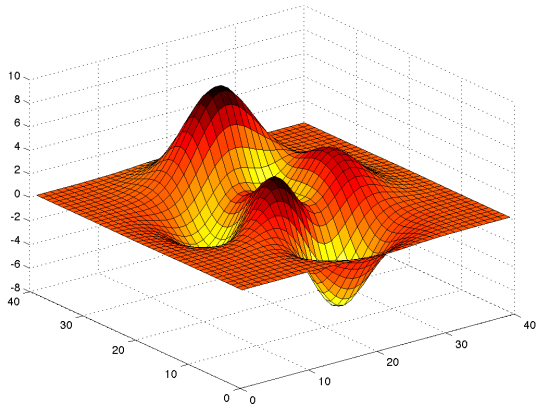
disp('other value')

Other useful commands

- **Workspace**
  - >> clear
  - >> who
  - >> whos
  - >> close

- **File operations**
  - >> ls
  - >> dir
  - >> cd
  - >> pwd
  - >> mkdir

Matlab



- The Matlab Environment
- Variables; operations on variables
- Programming
- Visualization

- Linear plots

```
>> plot (X, Y)
```

  Plotting commands open the Figure editor.

- Multiple datasets on a plot

```
>> plot(xcurve, ycurve)
>> hold on
>> plot(Xpoints, Ypoints)
>> hold off
```

- Subplots on a figure

```
>> subplot(1, 2, 1)
>> plot(time, velocity)
>> subplot(1, 2, 2)
>> plot(time, acceleration)
```

- 2D linear plots: `plot`

```
>> plot (X, Y, 'r-')
```

Colors: `b, r, g, y, m, c, k, w`

Markers: `o, *, ., +, x, d`

Line styles: `-, --, -., :`

- Annotating graphs

```
>> plot (X, Y, 'ro')
>> legend ('Points')
>> title ('Coordinates')
>> xlabel ('X')
>> ylabel ('Y')
```

- Plot Edit mode: icon  in Figure Editor

References

Violeta Ivanova, MIT

http://web.mit.edu/acmath/matlab/IAP2007/

Experiment with Matlab (Steve Moler):

http://www.mathworks.com/moler/exm/chapters.html

Matlab: Getting started

https://www.mathworks.com/help/matlab/getting-started-with-matlab.html