MATLAB Tutorial

Gregory Burlet
Department of Computing Science

About Me

- BSc Specialization in Computing Science
  University of Alberta

- MA Music Technology
  Music Information Retrieval
  McGill University

- Currently:
  MSc Computing Science
  Statistical Machine Learning
  University of Alberta

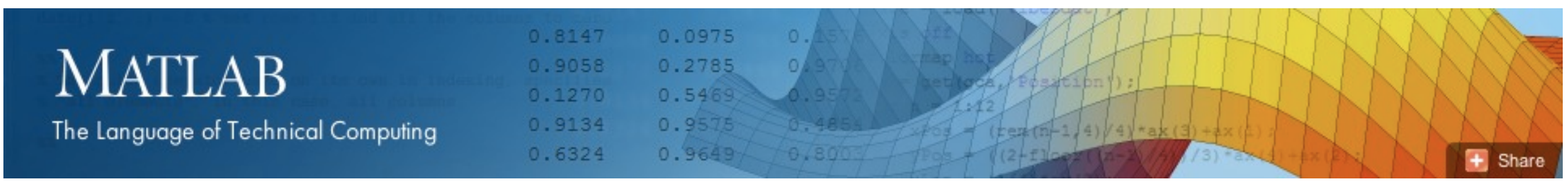
Research Interests:

Optimization, polyphonic transcription, speech recognition, anything to do with time-series data

MATLAB

—Matrix Laboratory—

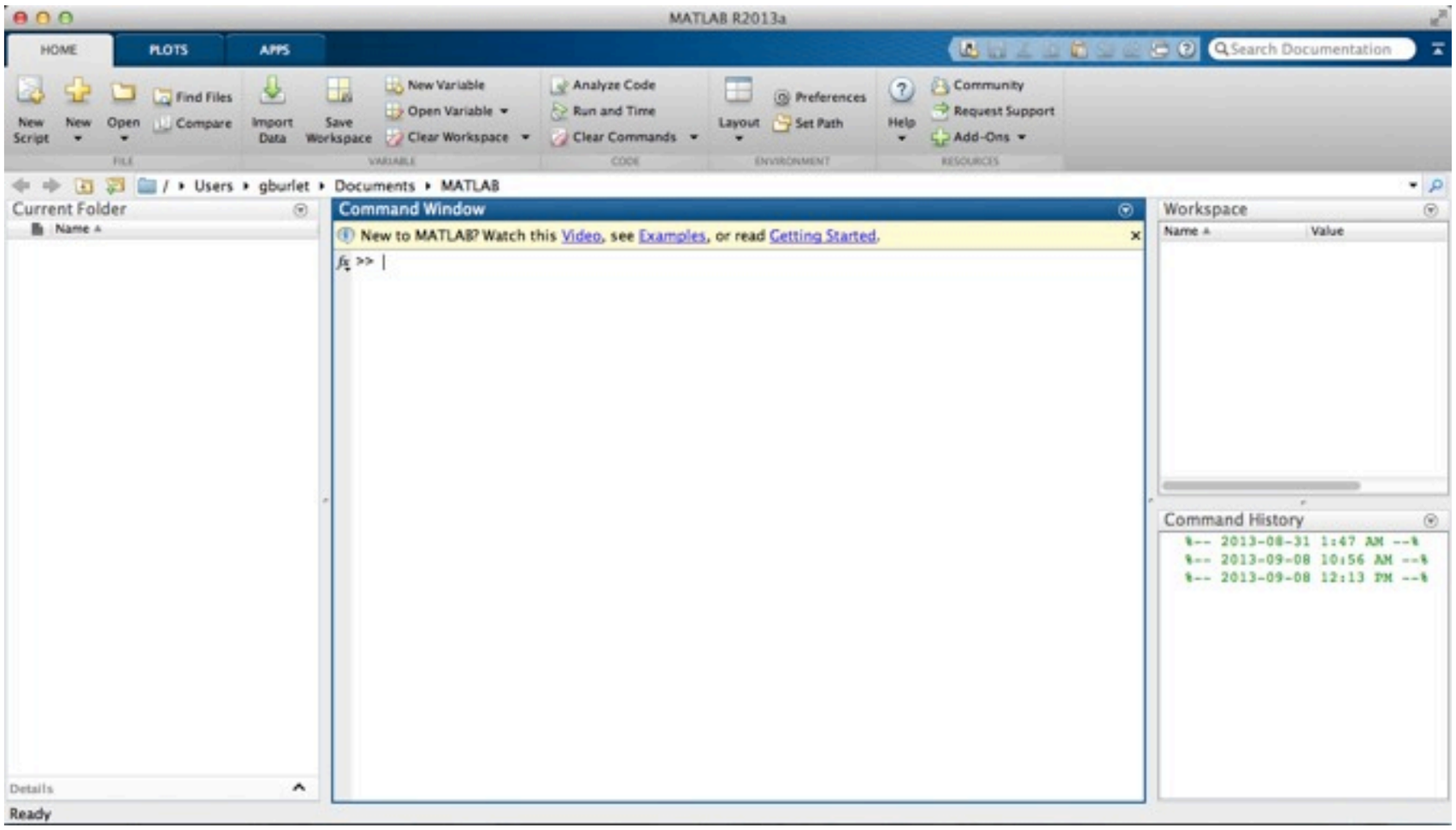
- Programming Language for numerical computing and visualization
- Ideal for matrix computations



www.mathworks.com/products/matlab

Using MATLAB

- Graphical User Interface



- Command line, type `matlab` to quit, type `exit`

>>

Matrices

- 2 x 3 matrix
  >> [1 2 3; 4 5 6]

  ans = 1 2 3
  4 5 6

- 2 x 4 matrix of zeros
  >> zeros(2,4)

  ans = 0 0 0 0
  0 0 0 0

- 2 x 2 matrix of ones
  >> ones(2)
  ans = 1 1
  1 1

- 2 x 2 identity matrix
  >> eye(2)
  ans = 1 0
  0 1

Matrices

- 2 x 3 random (uniformly distributed) matrix

```
>> rand(2,3)

ans = 0.8147    0.1270    0.6324
      0.9058    0.9134    0.0975
```

- 2 x 3 random (normally distributed) matrix

```
>> randn(2,3)
ans = -0.4336    3.5784   -1.3499
       0.3426    2.7694    3.0349
```

Concatenation

- Concatenate matrices horizontally:

```
>> a = [eye(2), ones(2,3)]
ans = 1    0    1    1    1
      0    1    1    1    1
```

- Concatenate matrices vertically:

```
>> b = [zeros(2); eye(1,2)]
ans = 0    0
      0    0
      1    0
```

- Warning: make sure dimensions work with each other!

Variables

- Built-in constants and variables:
  pi, eps (2.2204e-16), inf, i, j, NaN

- case sensitive variable & function names

  >> a = 2;

  >> b = ‘foo’;

  >> c = rand(a);

- List all variables: who

Indexing

- Indices start at one, not zero!

- Access particular element:

```
>> a = reshape(1:12, [3,4])
a =  1   4   7  10
     2   5   8  11
     3   6   9  12
>> a(2,2) = 5
>> a(1,:) = 1   4   7   10
>> a(:,3) = 7
            8
            9
```

Indexing

```
>> a(:, 1:2:4)
ans =  1   7
       2   8
       3   9
```

- Can also generate sequences with this syntax:

```
>> b = [1:3:10]
b = 1   4   7   10
```

Deleting Elements

```
>> b = [1:4]
b = 1   2   3   4
```

```
>> b(2) = []
b = 1   3   4
```

Matrix Operations

- Transpose:

```
>> a = [1:3]
>> a = 1   2   3
>> a'

ans = 1
      2
      3
```

Matrix Operations

- Addition:

```
>> a = eye(3);
>> b = ones(3);
>> a + b

ans = 2   1   1
      1   2   1
      1   1   2
```

- Subtraction works the same way

Matrix Operations

- Multiplication:

```
>> a = [1:3];
>> b = [2:4];
>> a * b'
```

```
ans = 20
```

- Element-wise multiplication:

```
>> a = [1:3];
>> b = [2:4];
>> a .* b
```

```
ans = 2   6   12
```

Can also do ./ for element-wise division

Matrix Operations

- Exponentiate:

```
>> a = [1:3];
>> b = ones(1,3) .* 2;
>> b .^ a
```

```
ans = 2    4    8
```

Saving & Loading

- Save matrix to data file:
  >> a = rand(3,4);
  >> save a
- matrix a is saved to file a.mat
- Load matrix a to data file:
  >> load('filepath')

Functions

- Frequently used functions:
  >> det(X): determinant of X
  >> eig(X): eigenvalues and vectors
  >> inv(X): inverse of X
  >> svd(X): singular value decomposition
  >> norm(X): norm of matrix
  >> find(X > n): indices of elements > n
  >> sqrt(X): square root
  >> sin(X): sinusoidal function
  >> abs(X): absolute value

Functions

- and so many more ...

- Don’t know which function to use or which functions are available?

help <function name>
doc <function name>

or doc and browse for functions

If else

- Boolean operators: ==, ~=
- Conditional statements:

```
>> a = 1;
>> if a > 0
>>    display('foo')
>> else
>>    display('bar')
>> end
```

Loops

- for loop:

```
>> a = 0;
>> for i = 1:3
>>    a = a + i;
>> end
```

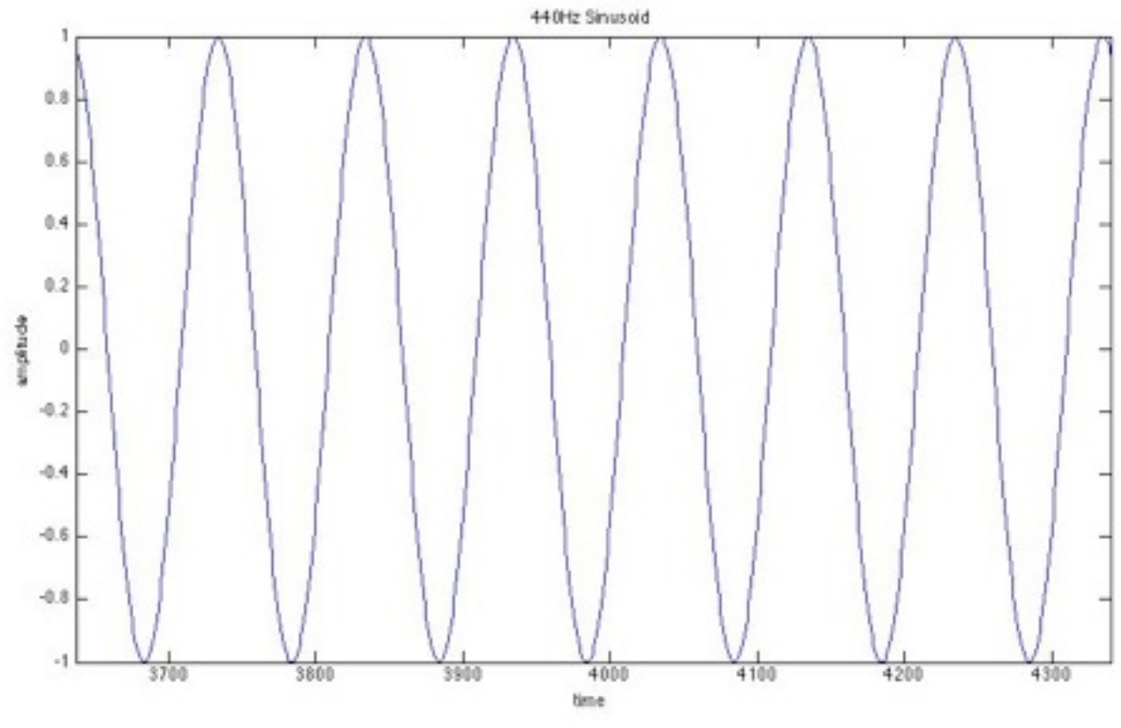
- while loop:

```
>> a = 3;
>> while a ~= 0
>>     a = a - 1
>> end
```

Plotting

- Plotting a sinusoid:

```
>> Fs = 44100
>> t = (0:Fs)*(1/Fs)
>> x = Fs*t
>> y = sin(2*pi*440*t)      % 440 Hz sinusoid
>> plot(x, y)
>> title('440 Hz Sinusoid')
>> xlabel('time')
>> ylabel('amplitude')
```



Scripts

- Add commands to a .m file
- From the command line, type the file name and the script will be run.
- Comments are made using the % character
- Function files declared using

```
function [y1,...,yN] = funcName(x1,...,xM)
end
```

Practical Example

—Least Squares—

- Find the least squares solution of the linear system:

  $x_1 - x_2 = 4$

  $3x_1 + 2x_2 = 1$

  $-2x_1 + 4x_2 = 3$

- Matrix representation of the linear system Ax=b,I such that $x = (A^TA)^{-1}A^Tb$:

```
>> A = [1 -1; 3 2; -2 4];
>> b = [4 1 3]';
>> x = (A'*A)\A'*b
x = 0.1789, 0.5018
```

H. Anton and R. Busby. 2003. *Contemporary Linear Algebra*, Section 7.8, Example 3, p. 396.