MATLAB - An Introduction for Mathematicians

Scott Morgan

July 19, 2016

Introduction - The MATLAB Interface

Installing Matlab

Installing Matlab on Windows/UNIX. Installing Octave as an alternative on Windows/UNIX.

Using Matlab for the first time

Opening Matlab for the first time.

[FIGURE]

Typing at the command prompt. m files and scripts.

Matlab as a calculator. help. Semi-colons and suppression. 'disp' eliminates warnings.

built in fns. 'demo'!

Complex numbers

Remarks

More than one way to write a program! And you MUST write human-readable code...

Not talking about symbolic stuff.

Basic Coding Ideas

You can assign a value to a *variable* in the following way

```
>> x = 3; %assigns the value 3 to he variable x
```

This will give the variable x the value 3, which can then be used in future calculations such as

```
>> x = 3; %assigns the value 3 to he variable x
>> 3 + x %uses the assigned value in the calculation 3+x
ans =
    6
```

These values will remain until they are overwritten or *cleared*. You can clear specific variables using the *clear* command.

```
>> x = 3;
>> 3 + x
ans =
    6
>> clear x;
```

Trying to use the variable x again will result in an error.

Unlike many other programming languages such as VB, Fortran or C, there is no need to *declare* variables to have a certain *type* in Matlab. Matlab will automatically figure out whether you are specifying an integer, a real number or a complex number. All computations in Matlab are done in double precision which means about 15 significant figures.

Variables can be named any combination of letters and numbers both lower case and upper case but must start with a letter.

You can see the variables which are currently stored by Matlab in the *workspace* - usually to the right or left of the command window. Alternatively, you can see the same information by running the command

```
>> whos
```

The interface, scripts and functions.

If there is nothing else you take from this first lesson, let the thing you learn be how to use *for* loops and *if* statements. These are central to most programs you will write during this course and usually play an important role in many other programs.

The very simple example below illustrates what these statements do.

```
for j = 1:10
    j %display the value assigned to j at each point in the loop
end

for j = 1:10
    if j == 5
        j + 3 %if j = 5, display 8
    elseif j == 6
        j + 1 %if j = 6, display 7
    else
        j %if j is not equal to 5 or 6, display j
    end
end
```

Note: Be careful using i and j in loops - Matlab may treat them as complex numbers. [ADD STUFF HERE]. Can use any loop counter name. Logicals ==, ¿= etc. Brief mention of while. While is good when you don't know how many iterations you need. Or even return.

Task: Write a script that will loop through the numbers 1 to N and print out all numbers that are a multiple of 3. Hint: you may find the following useful.

```
>> help mod
```

Task: Print Fibonacci sequence.

Task: Collatz.

Plotting. linspace. Adding annotations to plots. hold on;

Quick: Plot the graph of $y = \cos(3x)$. Experiment with changing the value of N - the number of points plotted across.

Task: Use previous Fib program to plot cvgnce to golden ratio.

Task: Use previous Collatz program to plot route to 1

Task: Prime counting.

Matlab Specific Coding Ideas

functions.

Vectors and arrays. Row and column vectors. Transpose. Hadamard products. abs.

Matrices. ones(N), zeros(N), diag. Hadamard again. det

Task: Write a script that will create an $N \times N$ identity matrix using for loops and if statements. Check with eye(N).

Matrix products. Norm - lots of other possibilities if needed, look them up!

Systems of linear equations:

mldivide, mention multiplication by inverse being slow.

Calculating eigenvalues using eigs.

Some More Advanced Ideas - with Applications in Applied Mathematics

Undergraduate problems. Approximate integration and differentiation. Newton's method and iterations. Error analysis. trapz and quad

Simple differential equations. Introduce ode23/ode45 - plotting and interpreting. cf WA book.

Simple finite differences. Error analysis and the Mesh-Peclet number. Solve simple odes using finite differences. Look at TP - CFD.

Spectral methods and solving odes using Chebyshev methods - differentiation matrices and the like.

Curve fitting - spline and ppval.

Extra Ideas

Pathological examples.

Some software development techniques. Git and version control. Commenting, portability and readability.

Task: Write unseen program. Give to friend. Do they understand it?

How does Matlab match up to other programming languages?

Other ideas - parallelisation?