Math 216 Differential Equations

Matlab - A Short Introduction

In Math 216, the principal computational tool is a software package called Matlab. Matlab (Matrix laboratory) is an interactive system designed for matrix computations: solving systems of linear equations, multiplying matrices, computing eigenvalues and eigenvectors, and much more. In this brief introduction, we will present some of the main features of Matlab, emphasizing some of the commands and functions that will be useful in Math 216. In a sense this introduction is a short version of the excellent manual by Mark S. Gockenbach, *A Practical Introduction to Matlab*. This manual is available online at

`http://www.math.mtu.edu/~msgocken/intro/intro.html`

and a postscript version (better for printing) is available from

`http://www.math.mtu.edu/~msgocken/intro/intro.ps`

We encourage the reader to experiment with the commands and to try out the examples while reading the material.

Matlab is available on most of the public computers on campus, and you may also purchase a copy of the software for your own computer if you like. On Macintosh computers, Matlab is launched either by clicking on the corresponding icon in the dock (if it is present), or by using the Finder to open the Applications folder, and then clicking on the Matlab icon in that folder. On Windows computers, Matlab can be accessed via the Start menu. Matlab is also available in a unix environment on Solaris-based machines that can be found throughout the university. If you are using a public computer and you want your Matlab work to be saved for future use, you need to set your working directory to one in your IFS space. Then whatever you save will be there the next time you log in. To exit Matlab, choose Quit from the MATLAB menu at the top of the screen. Other Matlab commands can be typed directly into the Matlab Command Window, after the Matlab prompt, >>. The command `clear` wipes out Matlab work done in the session and brings you back to the beginning of the session.

Vectors and Matrices; Variables and Functions; Getting Help

When using Matlab, one of the most useful things to remember is that every variable is a matrix. The following commands show how to assign numbers, vectors, and matrices to variables. The Matlab prompt is >>; the commands below are typed in after the prompt, and concluded with a carriage return. The response of Matlab appears on the next line.

```
>> a=5
a =
```

```
     5

>> v=[1;3;0]

v =

     1
     3
     0

>> A=[1,2,3;4,5,6;7,8,9]

A =

     1     2     3
     4     5     6
     7     8     9
```

Notice that the rows of a matrix are separated by semicolons, while the entries on any given row are separated by commas (spaces may also be used). As mentioned above, each of these three variables is regarded as a matrix; the scalar `a` is a 1×1 matrix, the vector `v` is a 3×1 matrix, and `A` is a 3×3 matrix.

Indices of variables must be positive integers. Thus, `x(0)` and `x(1.2)` are not allowed, nor is `A(0,1)` permitted. The size of a matrix can be found using the function `size(A)`, which returns the pair of numbers m, n when `A` is an $m \times n$ matrix. The length of the vector `v` is found by using the function `length(v)`.

When entering a matrix, if you want to prevent Matlab from displaying it immediately afterward (for example, if it is a very large matrix), you need to end the line with a semicolon, as the following example shows.

```
>> x=[1,2,6];
```

Matlab says nothing because you ended the line with a semicolon. However, it has nonetheless remembered your definition of the vector `x`.

Another useful thing to remember is that the complex unit $\sqrt{-1}$ is represented by either one of the built-in variables `i` or `j`. The following examples demonstrate how complex numbers are displayed in Matlab. They also show that the square root function is a built-in feature:

```
>> sqrt(-1)

ans =

        0 + 1.0000i
```

The variable `ans` contains the result of the most recent computation which can then be used as an ordinary variable in subsequent computations:

```
>> 10000-4*2*3

ans =

        9976

>> sqrt(ans)

ans =

   99.8799

>> (-100+ans)/4

ans =

   -0.0300
```

Another built-in variable that is often useful is `pi`:

```
>> pi

ans =

    3.1416
```

Only a few digits of the transcendental number π are displayed. More significantly only a finite number of digits of π are known to Matlab. This is because Matlab only deals in approximate arithmetic of real numbers. In particular, numbers smaller than a certain size cannot be represented by Matlab. This minimum size is stored in a built-in Matlab variable called `eps`:

```
>> eps

ans =

   2.2204e-16
```

Besides the square root function, other common functions are predefined. They include: