10.34, Numerical Methods Applied to Chemical Engineering
Prof. William Green
Lecture 1: Using MatLab to Evaluate and Plot Expressions

In this course, you will learn various numerical methods to solve equations you may encounter here at the Institute. Before you make a computation, know the answer ahead of time. This way, you will be prepared for a surprise should there be one. The homework will be difficult.

As far as textbooks go, Beers text is comprehensive and dense. I advise you to read "Numerical Recipes" first, then Beers text, and then follow the references in either book for additional information.

First we will solve systems of equations:

$$\underline{F}(\underline{x}) = \underline{0}$$

A special case is when $\underline{F}$ is linear in $\underline{x}$: a non-iterative, exact solution is possible. The equation to solve is:

$$\underline{\underline{M}} * \underline{x} - \underline{b} = \underline{0}.$$

When we encounter nonlinear systems, we will attempt to linearize them. Then we will examine eigenvalues and eigenvector bases. We will apply the eigenvector bases to differential equations. Other topics include singular value decomposition, ordinary differential equations, differential algebraic equations, optimization, boundary value problems involving partial differential equations. We will study models and data including Bayesian analysis. The remainder of the course, we will study some special topics, which include stochastic differential equations, turbulence, fourier transforms, and calculus of variations.

MatLab® Introduction

Please read the first 37 pages of the MatLab tutorial available on-line. The pages below refer to the MatLab tutorial. I illustrated how to

- Make a matrix (pp. 10-12)
- Make a vector
- Transpose a vector (Hermitian)
- Multiply a matrix and vector. I warn you that often times errors occur with matrices and vectors that have incorrect dimensions.
- Create a function with one output and several inputs (Note every function must contain your name, date, and what it does) (pp. 42-49)
- Add comments into your function
- Specify scalar multiplication and exponentiation with .* and .^. With many chemical engineering calculations you will want to use scalar computations as oppose to matrix computations.
- Create an x-y plot (pp. 23-27)
- Distinguish between global variable and local variables to a function
- Create a function with several outputs and several inputs
- Save a function file as its [function name].m
- Invoke help.

Some of the commands used include

```
M=[1 2.5; 3 4];
V=[300, 400, 500, 750]
V'
;
:
function
.*
.^
whos
log10
plot
help
```
linspace(300, 1000)

Example of Matlab program

Save as: rate.m

```
Function k = rate(T, A, n, Ea)
% returns the rate constant corresponding to input temperature
% by Bill Green 9/6/06
% input: T in Kelvin
% output: k in 1/second
R = 8.314;
Ea = 1000.*Ea;
k = A.*T.^n.*exp(-Ea./R.*T);
```

Note: Use period before *, /, ^, etc to do scalar math
Note: Use semi-colon at end of each line to prevent program from printing all values

Plot(X,Y) creates graph of line X vs. Y
Plot(X,Y, 'o') creates graph of points X vs. Y

'linspace' creates vector of increment space N: vector = linspace (X, Y, N)