ChE 400: Applied Chemical Engineering Calculations
Tutorial 3: Linear and Nonlinear Algebraic Eqs. with Excel and Matlab

Gerardine G. Botte

This handout contains information and examples on:

* How to calculate inverse, determinant and perform basic matrix operations in Matlab
* How to solve LAE and Non LAE using the command "solver" in Excel
* How to linearize systems of nonlinear algebraic equations using Newton Raphson method.
* Symbolic differentiation in Matlab
* Solve nonlinear systems of algebraic equations using both excel and Matlab

It is important that you review Tutorial IX of ChE-101: Solving Non Linear Algebraic Equations (see your ChE-101 binder, web: http://webche.ent.ohiou.edu/WEB-che101/T9-Non-LAEs.pdf)

1. Matrix Operations. The commands for the following Matrix operations in Matlab are:
1.1 determinant: det(A) where A is the matrix
1.2 inverse: inv(A) where A is the matrix
1.3 product of two matrices: A*B

For instance, given the following matrices:

$$[A]=\begin{bmatrix}2 & 1 & 3\\ 4 & 0 & 1\\ 2 & 1 & 1\end{bmatrix} \qquad \{B\}=\begin{Bmatrix}2\\ 1\\ 0\end{Bmatrix}$$

use Matlab to:

1.1 Calculate the determinant and the inverse matrix of [A].
1.2 Evaluate [C] and [E] defined as: [C] = [A] {B} and [E] = $[A]^{-1}$ {B}.

2. More exercises. Use the inverse and the product functions in excel to calculate the solution of the following systems of algebraic equations (look at Tutorial 1 to remember how to perform matrix operations in Excel):

3.1
$$\begin{aligned}&3x_1+18x_2+9x_3=18\\ &2x_1+3x_2+3x_3=117\\ &4x_1+x_2+2x_3=283\\ &\text{sol}:\{X^T\}=\{72,-13,4\}\end{aligned}$$

3.2
$$\begin{aligned}&x_1+9x_2+18x_3+x_4=18\\ &2x_1+3x_2+3x_3-x_4=117\\ &4x_1+x_2+2x_3=283\\ &x_2-3x_3=115\\ &\text{sol}:\{X^T\}=\{74.18,37.76,-25.75,67.41\}\end{aligned}$$

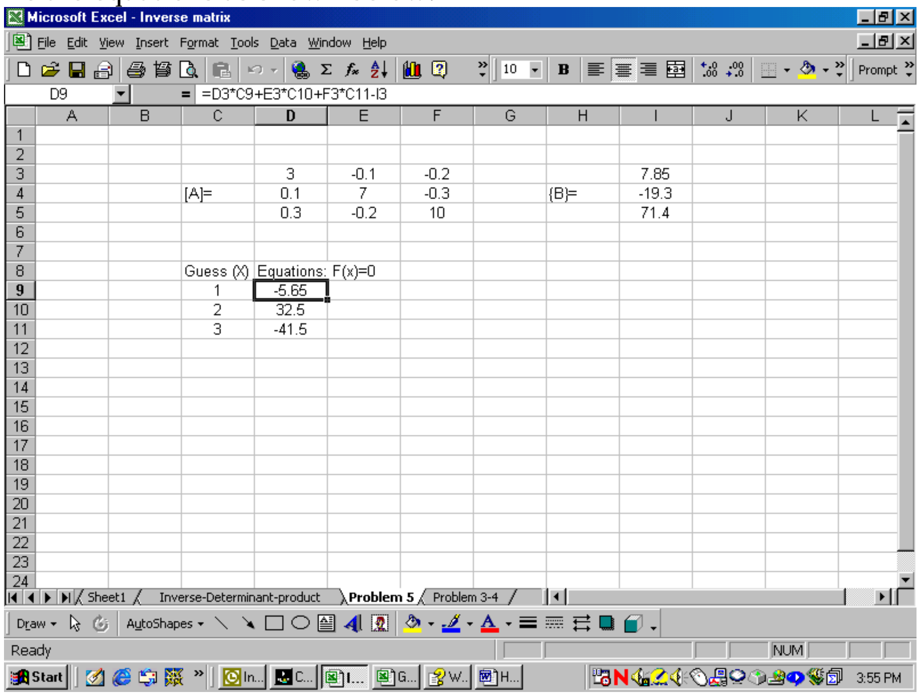
Compare your solutions by performing matrix operations in Matlab

3. Use of the command "Solver." The command "Solver" in Excel can also be used to solve a system of algebraic (linear or non-linear) equations.

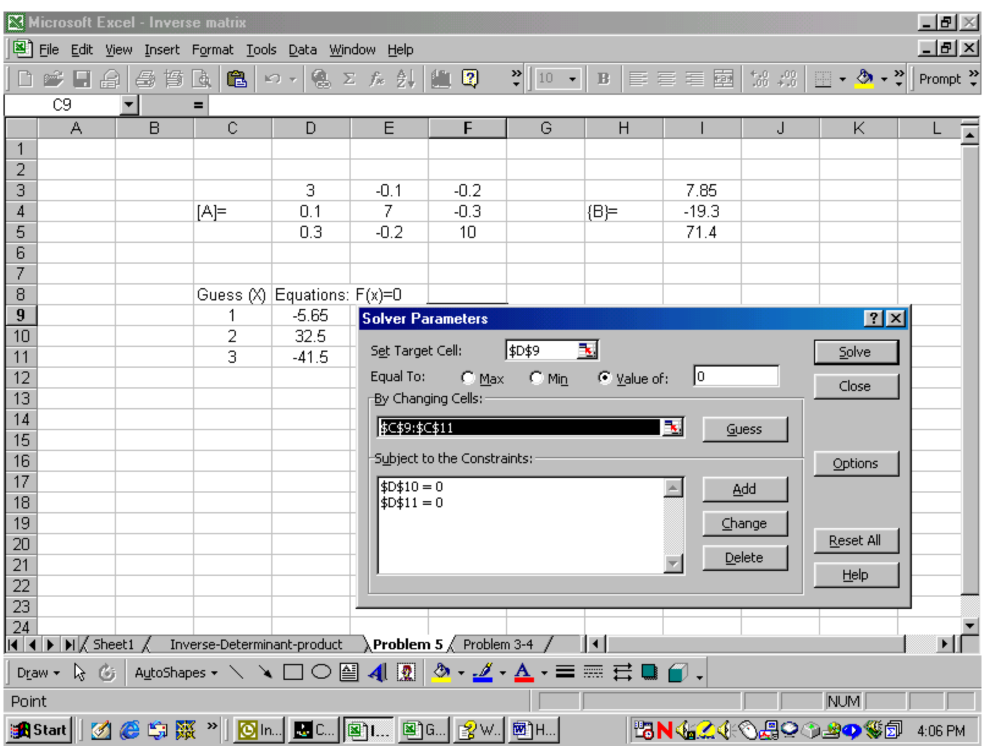
3.1 The first step is to define the equations (F(x)=0) and the guesses. For instance, suppose that we want to solve the following system of LAE using solver:

$$3x_1 - 0.1x_2 - 0.2x_3 = 7.85$$
$$0.1x_1 + 7x_2 - 0.3x_3 = -19.3$$
$$0.3x_1 - 0.2x_2 + 10x_3 = 71.4$$

we need to define the equations as shown below:



3.2 Go to the "Tools" menu and use the command "Solver". When using the "Solver" command only one cell is allow to set as the "target value". However, other constraints can be added to the problem in order to solve a system of equations as shown below:



Notice that the target cell is D9 that will be set to "0" by changing the values of the guesses (X values). However, the constraints that D10 and D11 have to be also "0" have been added to the calculation. This allows solving for the system of equations. The solution is obtained when clicking on "Solve". The results are given below:

	3	-0.1	-0.2		7.85
[A]=	0.1	7	-0.3	{B}=	-19.3
	0.3	-0.2	10		71.4

Guess (X)	Equations: F(x)=0
3	0
-2.5	0
7	0

4. <u>System of Nonlinear algebraic Equations using "Solver".</u> Use Solver to solve for the following system of nonlinear algebraic equations:

$$x_1^2 + x_1 x_2 = 10$$
$$x_2 + 3x_1 x_2^2 = 57$$

Solution:

System: x1^2+x1x2-10=0
x2+3x1x2^2-57=0

Guesses (X)	Equations (F(X)=0)
2.000000001	4.71E-09
2.999999998	-3.3E-08

5. <u>Newton Raphson's method of linearization:</u> As explained in class the two steps necessary to solve a system of nonlinear algebraic equations are:

1. Linearize using Newton Raphson's method
2. Solve the system of linear algebraic equations using any method of our choice.

Example: Use Newton Raphson's method and Cramer's rule to solve the system of non-linear algebraic equations shown in class:

$$x_1^2 + x_1 x_2 = 10$$
$$x_2 + 3x_1 x_2^2 = 57$$

Hint: Use Matlab to calculate the Jacobians (as explained below). Stop your calculation for abs(Error) = 0.001%. Remember that the problem needs to be linearized according to:

$$[J]\{X_{i+1}\} = -\{F_i\} + [J]\{X_i\} \text{ where } [J] = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}.$$

Before showing the solution of the problem we will discuss how to calculate symbolic derivatives using Matlab.

Symbolic derivatives using Matlab:

There are several ways to find information on using Symbolic Math Toolbox functions. One of them is to use online Help, which contains tutorials and reference information for all the functions.

You can also use MATLAB's command line help system. Generally, you can obtain help on MATLAB functions simply by typing:

help function

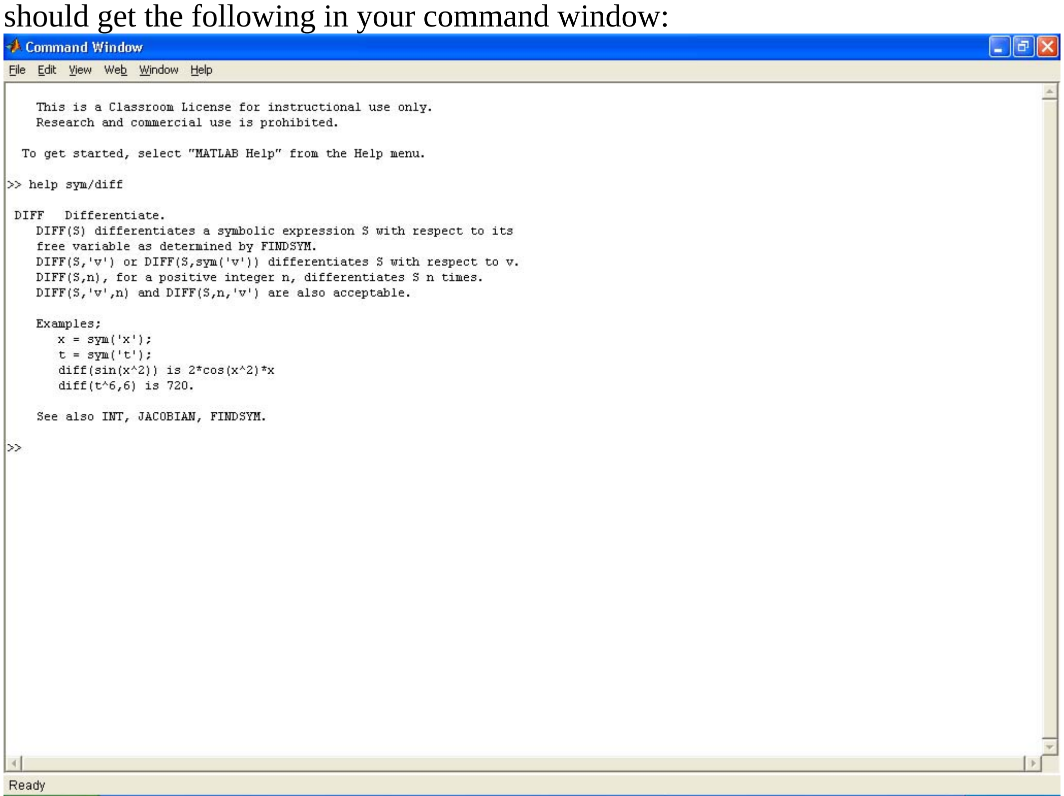
where function is the name of the MATLAB function for which you need help. This is not sufficient, however, for some Symbolic Math Toolbox functions. The reason? The Symbolic Math Toolbox "overloads" many of MATLAB's numeric functions. That is, it provides symbolic-specific implementations of the functions, using the same function name. To obtain help for the symbolic version of an overloaded function, type:

help sym/function

where function is the overloaded function's name. For example, to obtain help on the symbolic version of the overloaded function, diff, type

help sym/diff

you should get the following in your command window:



As an example let us calculate the derivative of the function sin(a*x). The first step is to create a symbolic expression using the command:

syms a x

(if you don't define the variables as symbolic it won't work).

*f = sin(a*x)*

Then diff(f) differentiates f with respect to its symbolic variable (in this case x), as determined by:

*ans = cos(a*x)*a*

To differentiate with respect to the variable a, type

diff(f,a)

which returns:

*ans =cos(a*x)*x*

To calculate the second derivatives with respect to x and a, respectively, type

diff(f,2) or diff(f,x,2)

which returns

*ans =-sin(a*x)*a^2*

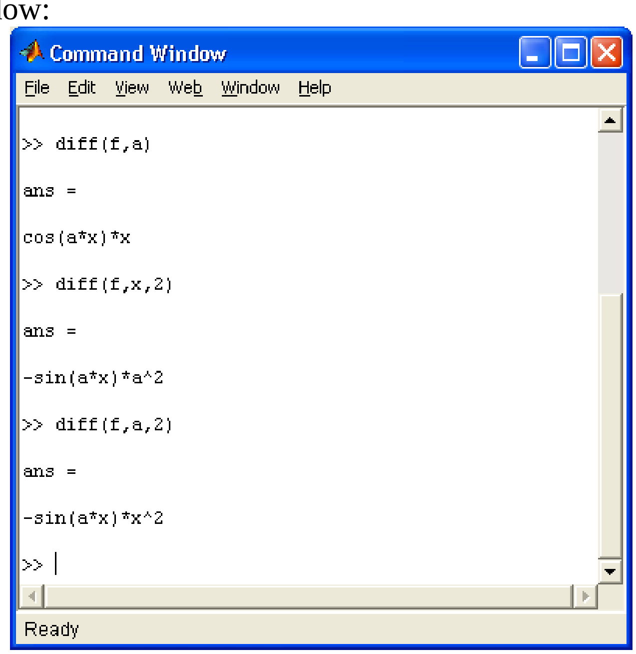
and

diff(f,a,2)

which returns

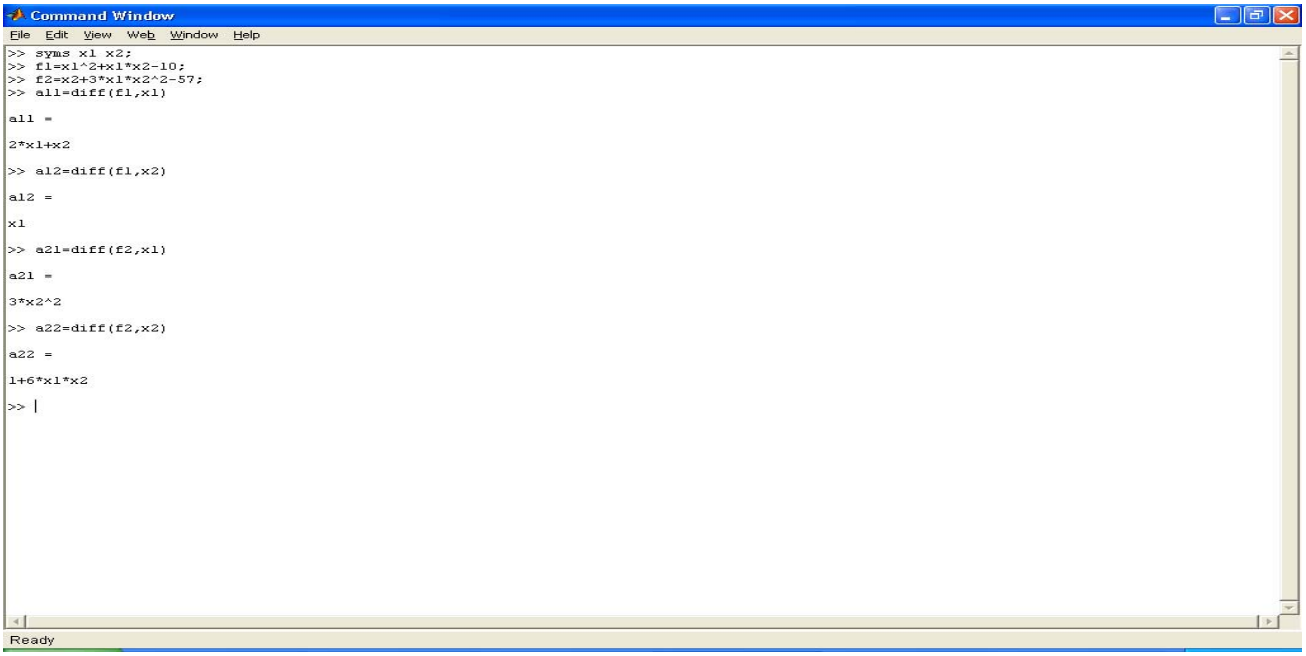
*ans = -sin(a*x)*x^2*

Try all the above operations given above. Compare your results with the summary given in the following command window:



<u>Solution to Problem 5</u>:

The solution of the problem is given below. The nomenclature is the same used in class: $a_{11} = \frac{\partial f_1}{x_1}\Big|_i$, $a_{12} = \frac{\partial f_1}{x_2}\Big|_i$, $a_{21} = \frac{\partial f_2}{x_1}\Big|_i$, $a_{22} = \frac{\partial f_2}{x_2}\Big|_i$, $b_1 = -f_{1,i} + a_{11}x_{1,i} + a_{12}x_{2,1}$, and $b_2 = -f_{2,i} + a_{21}x_{1,i} + a_{22}x_{2,1}$. The Jacobians were calculated in Matlab as shown:



```
>> syms x1 x2;
>> f1=x1^2+x1*x2-10;
>> f2=x2+3*x1*x2^2-57;
>> a11=diff(f1,x1)

a11 =

2*x1+x2

>> a12=diff(f1,x2)

a12 =

x1

>> a21=diff(f2,x1)

a21 =

3*x2^2

>> a22=diff(f2,x2)

a22 =

1+6*x1*x2

>> |
```

Using X1=1 and X2=9 as initial guesses the following answers were obtained:

Iteration	$X_{1,I}$	X2,I	a11	a12	a21	a22	f1,I	f2,I	b1	b2	$X_{1,I+1}$	X2,I+1	abs(Error1)%	abs(Error2)%
1	1.0000	9.0000	11.0000	1.0000	243.0000	55.0000	0.0000	195.0000	20.0000	543.0000	1.5387	3.0746	35.009	192.722
2	1.5387	3.0746	6.1519	1.5387	28.3592	29.3847	-2.9017	-10.2898	17.0983	144.2712	2.0450	2.9361	24.758	4.716
3	2.0450	2.9361	7.0261	2.0450	25.8625	37.0259	0.1862	-1.1756	20.1862	162.7766	2.0001	2.9992	2.243	2.104
4	2.0001	2.9992	6.9994	2.0001	26.9860	36.9926	-0.0008	-0.0260	19.9992	164.9496	2.0000	3.0000	0.005	0.026
5	**2.0000**	**3.0000**	**7.0000**	**2.0000**	**27.0000**	**37.0000**	**0.0000**	**0.0000**	**20.0000**	**165.0000**	**2.0000**	**3.0000**	**0.000**	**0.000**

Finally, $\{X^T\} = \{2, 3\}$

6. <u>Use of "fsolve"</u>. Solve the system of non-linear algebraic equations given in Problem 1 using "fsolve."

<u>Solution</u>: The first step is to draw the flowchart for your algorithm or sketch your procedure before you write the code

Write the system of nonlinear algebraic equations function:

C:\MATLAB6p1\work\func1.m

```
function F=func1(x)
%This function is used by the main program in H3.m as described in H3 problem 2.
%Call func1(x) where x is the vector with the dependent variables
%Developed by GGB on 10/14/02
%---------------------------------------------------------------------------------
F(1)=x(1)^2+x(1)*x(2)-10;
F(2)=x(2)+3*x(1)*x(2)^2-57;
```

Write your main code according to your flowchart:

C:\MATLAB6p1\work\H3.m

```
%Solution for Problem 2 Handout 3 (H3)
%This program solves a system of nonlinear algebraic equations using fsolve.
%The input data is the guesses for the dependent variables x(1) and x(2).
%It calls for the function file fun1(x) which contains the set of eqs. to
%solve simultaneously.
%Developed by G.G. Botte on 10/14/02.
%----------------------------------------------------------------------
clear
clc
%----------------------------------------------------------------------------------------
% Initial guesses: it makes sense to use the initial concentration to reactor 1 as the initial gu
%----------------------------------------------------------------------------------------
xo(1)=1;
xo(2)=9;
%----------------------------------------------------------------------------------------
% Solution of the equations
%----------------------------------------------------------------------------------------
options = optimset('Largescale','off','Display','off','TolX',1e-5);
x = fsolve('func1',xo,options);
%----------------------------------------------------------------------------------------
%     Printing the results
%----------------------------------------------------------------------------------------
fprintf(1,'RESULTS: Dependent variablex x1 and x2\n');     %Results
for i=1:2
fprintf(1,'x1( %1.0f \',i);
fprintf(1,') = %8.4f \n',x(i));
end
```

After running the program you should get the following answers: