Midterm Examination Solutions
February 25, 2014

1. Solution of a System of Nonlinear Algebraic Equations

Consider the set of nonlinear algebraic equations

$$0 = 10x_1 - 4x_2^3 + 9$$
$$0 = \exp(x_1) - 4x_2^2 + 1$$

(a) Use the multivariate Newton-Raphson method to find the roots to this system of equations near (2,2) and (-1,1). Report the RMS error on x and the number of iterations to converge.

Solution:
I used the code nrndn.m I modified the last two lines of the input file as follows.

```
function f = funkeval(x)
n = max(size(x));
f = zeros(n,1);
f(1) = 10*x(1) - 4*x(2)^3 +9;
f(2) = exp(x(1)) - 4*x(2)^2 + 1;
```

To find the root near (2,2), at the command prompt, I typed

```
>> [x,err,f] = nrndn([2,2],1.0e-6,1)
```

This generated the following output.

```
>> [x,err,f] = nrndn([2,2],1.0e-6,1)
 iter =    1, err =  1.17e+00 f =  5.78e+00
 iter =    2, err =  3.86e-01 f =  1.28e+01
 iter =    3, err =  1.73e-01 f =  3.17e+00
 iter =    4, err =  3.05e-02 f =  4.13e-01
 iter =    5, err =  8.36e-04 f =  1.07e-02
 iter =    6, err =  6.30e-07 f =  7.99e-06

x =    2.8136    2.1017

err =   6.2953e-07

f =   7.9882e-06
```

Therefore, the root near (1,1) is (x_1, x_2) = (2.8136,2.1017). The RMS error on x is 6.2x10^{-7}. It took six iterations to converge.

To find the root near (-1,1), at the command prompt, I typed

```
>> [x,err,f] = nrndn([-1,1],1.0e-6,1)
```

This generated the following output.

```
 iter =    1, err =  2.42e-01 f =  4.00e+00
 iter =    2, err =  7.23e-02 f =  8.47e-01
 iter =    3, err =  3.15e-03 f =  3.15e-02
 iter =    4, err =  8.86e-06 f =  8.40e-05
 iter =    5, err =  1.64e-10 f =  1.82e-09

x =   -0.8133    0.6007

err =   1.6358e-10

f =   1.8203e-09
```

Therefore, the root near (-1,1) is (x_1, x_2) = (-0.8133,0.6007). The RMS error on x is $1.6\text{x}10^{-10}$. It took five iterations to converge.

2. Solution of a System of Ordinary Differential Equations

The Van der Pol oscillator is a non-conservative oscillator with non-linear damping. It evolves in time according to the second order differential equation,

$$\frac{d^2x}{dt^2} - \mu\left(1 - x^2\right)\frac{dx}{dt} + x = 0$$

where x is the position coordinate, which is a function of the time t, and μ is a scalar parameter indicating the nonlinearity and the strength of the damping.

(a) Convert this second order ODE to a system of two first order ODEs.
(b) Use the classical fourth-order Runge Kutta method to solve this ODE from $t = 0$ to $t = 10$ for $\mu = 1/2$ and subject to the initial conditions, $x(t = 0) = 1$ and $x(t = 0) = 0$. Sketch the plot and report the value of x at t = 10.

solution:

(a) Convert this second order ODE to a system of two first order ODEs.

There is a three-step procedure to converting a higher order ODE to a system of first order ODEs.

First, identify the new variables.

$$y_1 = x \qquad \text{and} \qquad y_2 = \frac{dx}{dt}$$

Second, write the new ODEs in terms of the new variables

$$\frac{dy_1}{dt} = y_2 \qquad \text{and} \qquad \frac{dy_2}{dt} = \mu\left(1 - {y_1}^2\right)y_2 - y_1$$

Third, write the initial conditions in terms of the new variables.

$y_1(t=0)=1$ and $y_2(t=0)=0$

(b) Use the classical fourth-order Runge Kutta method to solve this ODE from $t=0$ to $t=10$ for $\mu = 1/2$ and subject to the initial condition, $x(t=0)=1.0$. Sketch the plot and report the value of x at t = 10.

I used the code `rk4n.m.`

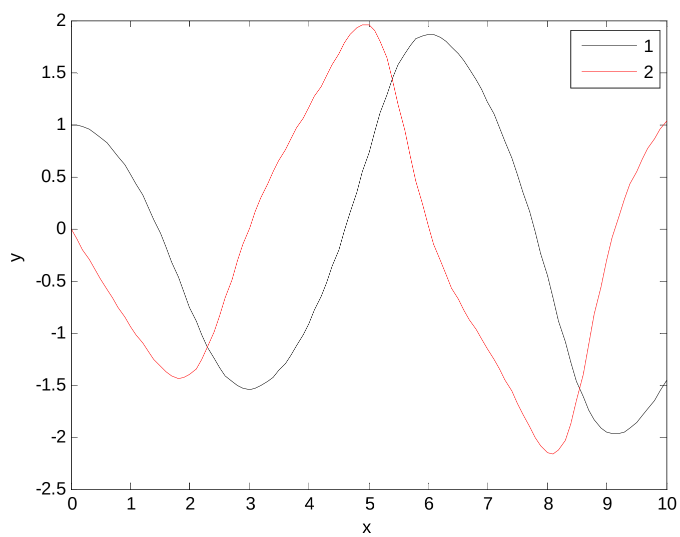
I changed the input function to

```
function dydx = funkeval(x,y);
mu = 0.5;
dydx(1) = y(2);
dydx(2) = mu*(1-y(1)^2)*y(2)-y(1);
```

At the command line prompt, I typed

>> [x,y]=rk4n(100,0,10,[1,0]);

This is the plot that was generated.



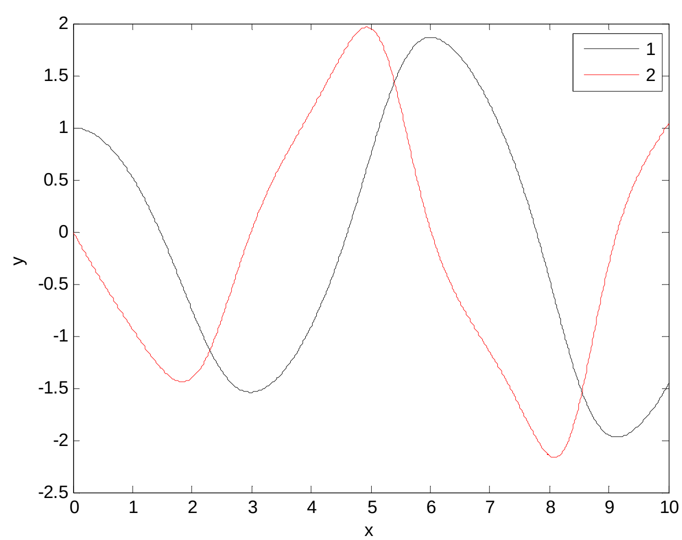
To obtain the value of $x(t=10)$, I accessed the last value of the first variable.

>> y(101,1)
ans = -1.4601

Therefore, $x(t=10) = -1.4601$

To confirm that my answer is reasonable, I reran the code with 1000 intervals.
>> [x,y]=rk4n(1000,0,10,[1,0]);

This is the plot that was generated.

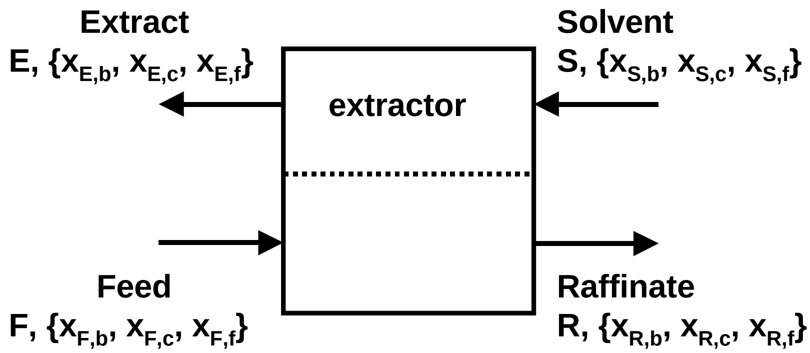


To obtain the value of $x(t=10)$, I accessed the last value of the first variable.

>> y(1001,1)
ans = -1.4601

Therefore, $x(t=10) = -1.4601$

3. Formulation and Solution of a System of Linear Algebraic Equations
Consider a liquid-liquid extractor as shown in the figure below that removes benzene from a primarily cyclohexane Feedstream using a furfural Solvent stream.



You are given all four flow rates and the compositions of the Feed (F) and Solvent (S) streams. Your task is to determine the composition of the two exiting streams, the Raffinate (R) and Extract (E).

$F = 100\, mol/hr$	$S = 150\, mol/hr$	$R = 95\, mol/hr$	$E = 155\, mol/hr$
$x_{F,b} = 0.1$	$x_{S,b} = 0.0010$	$x_{R,b} = ?$	$x_{E,b} = ?$
$x_{F,c} = 0.9$	$x_{S,c} = 0.0001$	$x_{R,c} = ?$	$x_{E,c} = ?$
$x_{F,f} = 0.0$	$x_{S,f} = 0.9989$	$x_{R,f} = ?$	$x_{E,f} = ?$

An analysis of the system yields six equations for your six unknowns.

(1) a benzene molar balance: $Rx_{R,b} + Ex_{E,b} = Fx_{F,b} + Sx_{S,b}$

(2) a cyclohexane molar balance: $Rx_{R,c} + Ex_{E,c} = Fx_{F,c} + Sx_{S,c}$

(3) raffinate mole fractions sum to unity: $x_{R,b} + x_{R,c} + x_{R,f} = 1$

(4) extract mole fractions sum to unity: $x_{E,b} + x_{E,c} + x_{E,f} = 1$

(5) benzene equilibrium constraint: $K_b = \frac{x_{E,b}}{x_{R,b}} = 20.0$

(6) c-hexane equilibrium constraint: $K_c = \frac{x_{E,c}}{x_{R,c}} = 0.01$

(a) Formulate the equations as a system of six linear algebraic equations in six unknowns.
(b) Convert the equations to matrix notation, $\underline{\underline{A}}\underline{x} = \underline{b}$. Identify $\underline{\underline{A}}$, $\underline{x}$ and $\underline{b}$.
(c) Determine and report the compositions of the Raffinate and Extract streams.

Solution:

(a) Formulate the equations as a system of six linear algebraic equations in six unknowns.

benzene mole balance:	$Rx_{R,b} + Ex_{E,b} = Fx_{F,b} + Sx_{S,b}$
cyclohexane mole balance:	$Rx_{R,c} + Ex_{E,c} = Fx_{F,c} + Sx_{S,c}$
raffinate mole fraction constraint:	$x_{R,b} + x_{R,c} + x_{R,f} = 1$
extract mole fraction constraint:	$x_{E,b} + x_{E,c} + x_{E,f} = 1$
benzene equilibrium constraint:	$x_{E,b} - x_{R,b}K_b = 0$
c-hexane equilibrium constraint:	$x_{E,c} - x_{R,c}K_c = 0$

(b) Convert the equations to matrix notation, $\underline{\underline{A}}\underline{x} = \underline{b}$. Identify $\underline{\underline{A}}$, $\underline{x}$ and $\underline{b}$.

$$\underline{x} = \begin{bmatrix} x_{R,b} \\ x_{R,c} \\ x_{R,f} \\ x_{E,b} \\ x_{E,c} \\ x_{E,f} \end{bmatrix}, \quad \underline{\underline{A}} = \begin{bmatrix} R & 0 & 0 & E & 0 & 0 \\ 0 & R & 0 & 0 & E & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ -K_b & 0 & 0 & 1 & 0 & 0 \\ 0 & -K_c & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \underline{b} = \begin{bmatrix} Fx_{F,b} + Sx_{S,b} \\ Fx_{F,c} + Sx_{S,c} \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

(c) Determine and report the compositions of the Raffinate and Extract streams.

I created the Matlab Script titled exam01_s14_p01.m and provided below.

```
clear all;

F = 100;
S = 150;
R = 95;
E = 155;
xFb = 0.1;
xFc = 0.9;
xFf = 0.0;
xSb = 0.001;
xSc = 0.0001;
xSf = 0.9989;
Kb = 20.0;
Kc = 0.010;

A = [R 0 0 E 0 0
    0 R 0 0 E 0
    1 1 1 0 0 0
    0 0 0 1 1 1
    -Kb 0 0 1 0 0
    0 -Kc 0 0 1 0];

b= [F*xFb + S*xSb
    F*xFc + S*xSc
    1
    1
    0
    0];
```

```
detA = det(A)
invA = inv(A);
x = invA*b
```

At the Matlab prompt, I typed

```
>>  exam01_s14_p01
```

This yielded the following output

```
detA =   3.0848e+05

x =
    0.0032
    0.9323
    0.0645
    0.0635
    0.0093
    0.9271
```

Therefore the unknown compositions are

x(1) = 0.0032 = $x_{R,b}$ x(2) = 0.9323 = $x_{R,c}$ x(3) = 0.0645 = $x_{R,f}$

x(4) = 0.0635 = $x_{E,b}$ x(5) = 0.0093 = $x_{E,c}$ x(6) = 0.9271= $x_{E,f}$