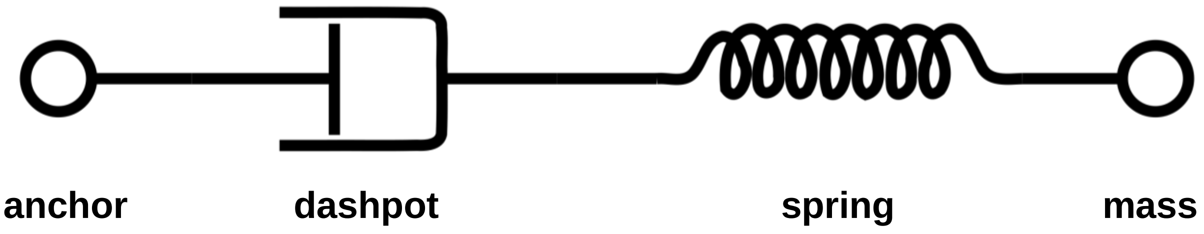
Final Exam Solutions

Administered: Tuesday, December 7, 2021
10:30 AM – 12:45 PM
40 points

Problem 1. (14 points)

Consider a one-dimensional problem in the x direction in which a mass is attached to a dashpot and spring in series, anchored to a fixed point.



The equation of motion for the particle is

$$ma = F_{dashpot} + F_{spring} \tag{1}$$

where m is the mass of the particle, a is the acceleration of the particle. $F_{dashpot}$ is the force exerted by the dashpot,

$$F_{dashpot} = -cv \tag{2}$$

where c is a constant associated with the dashpot and v is the velocity of the particle. F_{spring} is the force of the spring,

$$F_{spring} = -kx \tag{3}$$

where k is a constant associated with the dashpot and x is the position of the particle.

The velocity is the first derivative of the position and the acceleration is the second derivative.

$$v = \frac{dx}{dt} \tag{4}$$

$$a = \frac{d^2x}{dt^2} \tag{5}$$

where t is time. Substituting equations (2) through (5) into equation (1) yields

$$\frac{d^2x}{dt^2} = -\frac{c}{m}\frac{dx}{dt} - \frac{k}{m}x \tag{6}$$

Consider a system with initial position of the particle, $x_o = 1.5$ m, and initial velocity of the particle, $v_o = 0.0$ m/s. Numerical values of the system constants are $m = 1.0$ kg, $c = 0.5$ N·s/m and $k = 1.0$ N/m. Perform the following tasks.

(a) Convert the second order ODE in equation (6) to a system of first order ODEs.
(b) Is this an initial value problem or a boundary value problem?
(c) What numerical technique will you use to solve this system of first order ODEs?
(d) Plot the system of ODEs from time = 0 to time =10.0 s.
(e) Report the particle position and velocity at time =10.0 s.
(f) Numerically verify that your result in (e) is accurate.
(g) By running a longer simulation, estimate the final particle position and velocity when the system comes to rest.

Solution:

(a) Convert the second order ODE in equation (6) to a system of first order ODEs.

This conversion follows a three step process.

Step 1. Define new variables.

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

Step 2. Write ODEs for the new variables.

In this transformation, the first equation is always

$$\frac{dy_1}{dt} = y_2$$

The second equation is substituting the variables in Step 1 into the original ODE.

$$\frac{d^2x}{dt^2} = -\frac{c}{m}\frac{dx}{dt} - \frac{k}{m}x$$

or

$$\frac{dv}{dt} = -\frac{c}{m}v - \frac{k}{m}x$$

$$\frac{dy_2}{dt} = -\frac{c}{m}y_2 - \frac{k}{m}y_1$$

Step 3. Write initial conditions for the new variables.

$$y_1(t = 0) = x(t = 0) = 1.5 \qquad y_2(t = 0) = v(t = 0) = 0.0$$

(b) Is this an initial value problem or a boundary value problem?

This is an initial value problem because the values of both variables, position and velocity, are given at the same value of the independent variable, time.

(c) What numerical technique will you use to solve this system of first order ODEs?

I would use the classical fourth order Runge-Kutta method to solve this system of two first order ordinary differential equations.

(d) Plot the system of ODEs from time = 0 to time =10.0 s.
(e) Report the particle position and velocity at time =10.0 s.

To solve this problem, I first entered the system of ODEs into the input function for rk4n.m as follows

```
function dydt = funkeval(t,y);
x = y(1);
v = y(2);
m = 1.0;
c = 0.5;
k = 1.0;
% ODE #1 dxdt = v;
dydt(1) = y(2);
% ODE #2 dvdt = -c/m*v -k/m*x;
dydt(2) = -c/m*v -k/m*x;
```

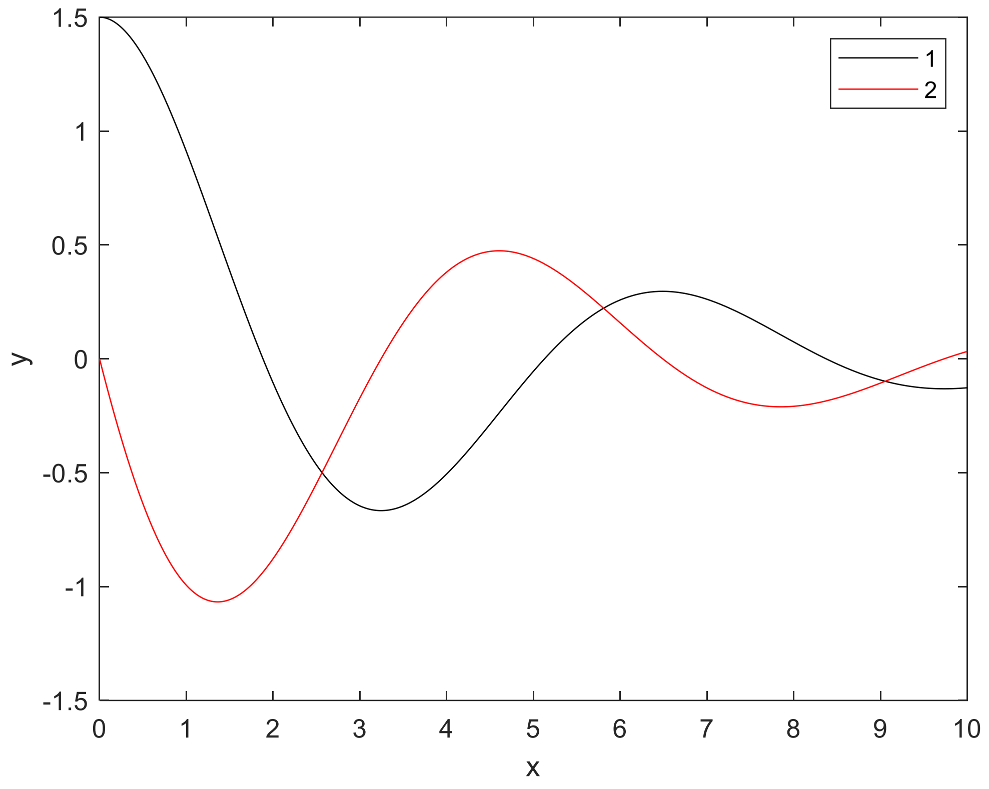
I then wrote a little script to set the parameters for Runge-Kutta as follows,

```
clear all;
format long

n = 1000;
to = 0.0; %s
tf = 10.0; %s
xo = 1.5; % m
vo = 0.0; % K
yo = [xo,vo];

[t,y]=rk4n(n,to,tf,yo);
x_f = y(n+1,1)
v_f = y(n+1,2)
```

At the command line prompt, I executed the script:

```
>> xm4p01_f21
```

This command generated the following plot and output.



and

```
x_f =  -0.127163943497666
v_f =   0.032406639188047
```

In this plot, the first curve is position and the second curve is velocity. At a time of 10 s, the position is -0.127 m and the velocity is 0.032 m/s.

(f) Numerically verify that your result in (e) is accurate.

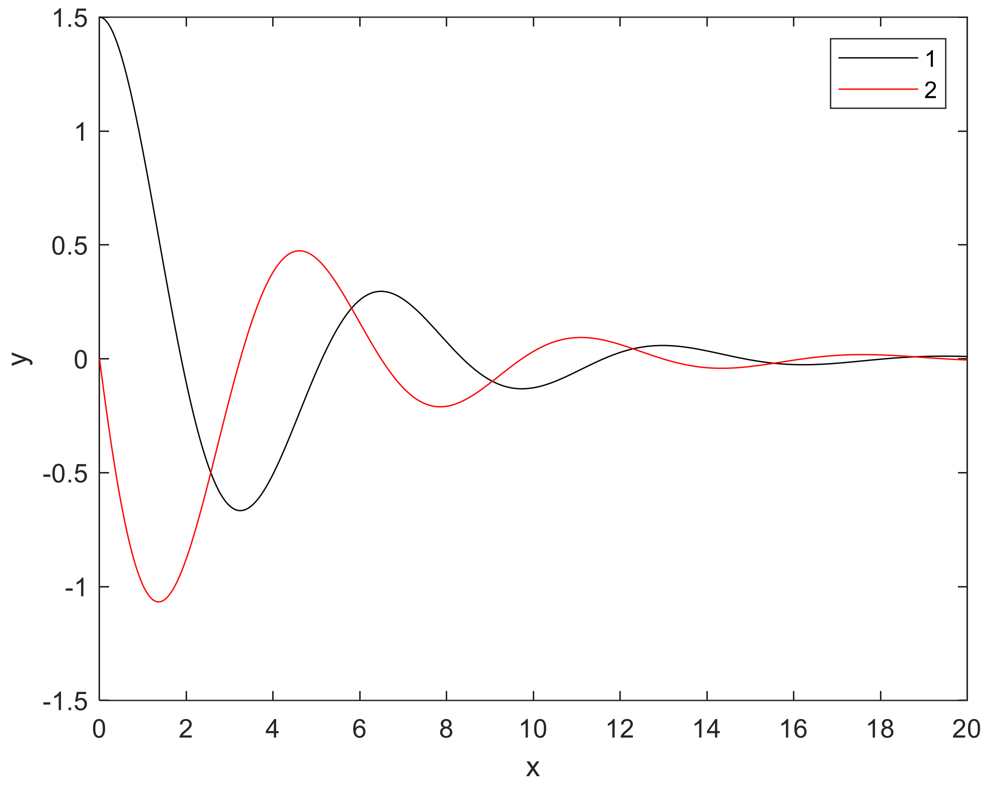
If we rerun the script with 10,000 steps rather than the original 100, we find

```
x_f =  -0.127163943396562
v_f =   0.032406639194179
```

So the results are not significantly changes as a function of step size and we can regard the solution as accurate.

(g) By running a longer simulation, estimate the final particle position and velocity when the system comes to rest.

If we run the simulation for a time of 20 s, we find



and

```
x_f =   0.010080318841609
v_f =  -0.005144544625056
```

From the plot, it is clear that the system will come to rest when neither the spring nor the dashpot exerts any force on the particle, which occurs at a position of zero and a velocity of zero.

Problem 2. (8 points)
The longest relaxation time of a polymer can be measured through an auto-correlation function (acf) of the polymer end-to-end distance.

$$acf = c \cdot exp\left(-\frac{t}{\tau}\right) \tag{1}$$

where t is time (sec), τ is relaxation time (sec) and c is a prefactor. The acf is dimensionless.

For the acf vs t data given in the file, http://utkstair.org/clausius/docs/mse301/data/xm4p02_f21.txt, perform the following tasks. In this data file, the first column is time and the second column contains the values of the acf.

(a) Identify all variables, $y = mx + b$, when equation (1) is linearized.
(b) Report the best value of τ and c.
(c) Report the standard deviations of τ and c.
(d) Report the measure of fit.

Solution

(a) Identify all variables, $y = mx + b$, when equation (1) is linearized.

$$ln(acf) = ln(c) + -\frac{t}{\tau} \tag{2}$$

$$y = ln(acf) \tag{3.y}$$

$$x = t \tag{3.x}$$

$$b = ln(c) \tag{3.b}$$

$$m = -\frac{1}{\tau} \tag{3.m}$$

(b) Report the best value of τ and c.
(c) Report the standard deviations of τ and c.
(d) Report the measure of fit.

I wrote the following script, xm4p02_f21.m, in Matlab.

```
clear all;
format long;

M = [0  1.945163992
0.1 2.176598578
0.2 2.031599186
0.3 2.01298014
...
44.7    0.092141451
44.8    0.103621804
44.9    0.093780592
45  0.108042989];

n = max(size(M));
for i = 1:1:n
    x(i) = M(i,1);
    y(i) = log(M(i,2));
end
[b,bsd,MOF] = linreg1(x, y)

slope=b(2);
intercept = b(1);
slope_sd = bsd(2);
intercept_sd = bsd(1);

tau = -1/slope
c = exp(intercept)
tau_sd = abs(slope_sd/slope*tau)
c_sd = abs(intercept_sd/intercept*c)


%
% plot model (optional)
%
yhat = zeros(n,1);
for i = 1:1:n
    yhat(i) = c*exp(-M(i,1)/tau);
end
figure(2)
plot(M(:,1),M(:,2),'ro');
hold on;
plot(M(:,1),yhat(:),'k-');
xlabel('time');
ylabel('auto correlation function');
legend('data','model');
```

At the command line prompt, I executed the script.

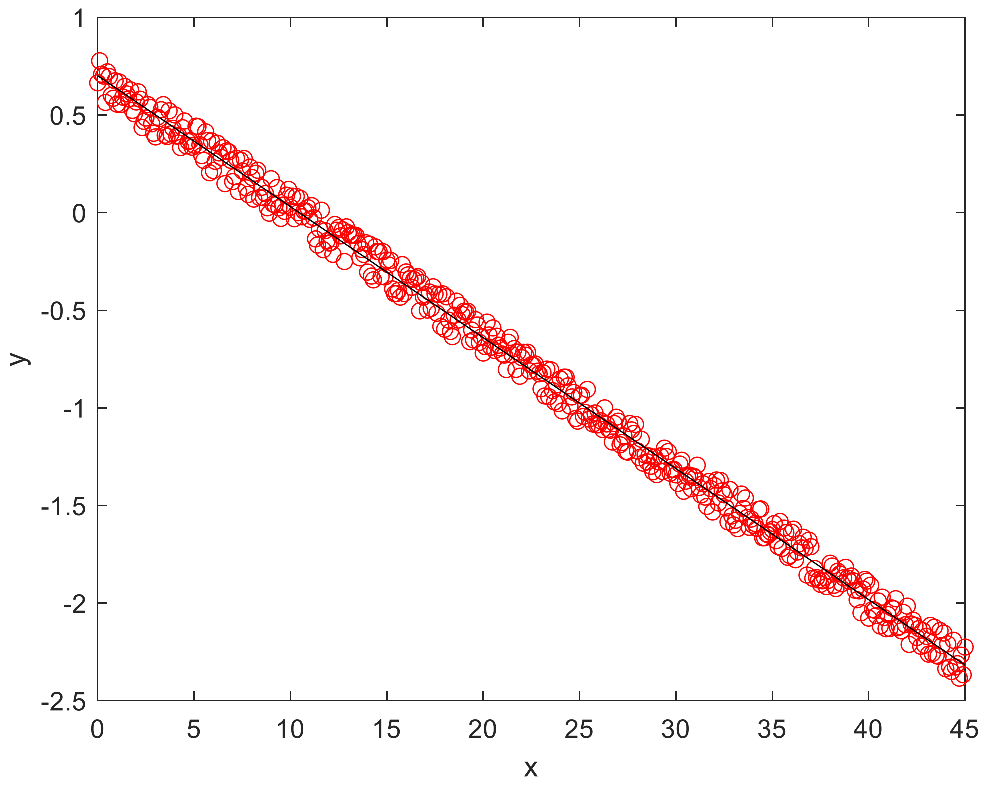
```
>> xm4p02_f21
```

This generated the following output.

```
b =
   0.701531753921776
  -0.067095076487183

bsd =
   0.005657512532778
   0.000217636883104

MOF =   0.995297994202787
```

```
tau =  14.904223265786433
c =   2.016839644688160
tau_sd =  0.048344958624072
c_sd =   0.016264831210619
```

and the plot:



Thus

(b) Report the best value of τ and c.

Based on the slope, m, and intercept, b,

$$c = exp(b) \tag{4.b}$$

$$\tau = -\frac{1}{m} \tag{4.m}$$

The mean value of these two parameters are

$\tau = 14.9$ s $\qquad c = 2.02$

(c) Report the standard deviations of τ and c.

Using rules for propagation of uncertainty,

$$\frac{\sigma_c}{c} = \left|\frac{\sigma_b}{b}\right| \tag{5.b}$$

$$\frac{\sigma_\tau}{\tau} = \left|\frac{\sigma_m}{m}\right| \tag{5.m}$$

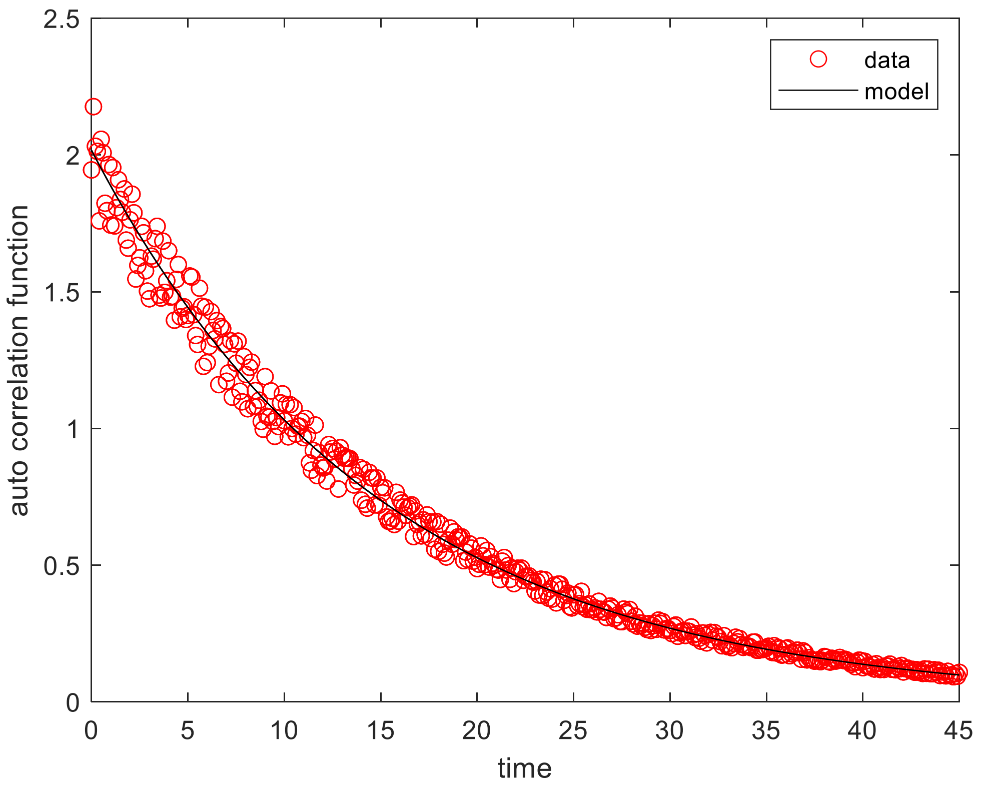
The standard deviations of these two parameters are

$\sigma_c = 0.016$ $\sigma_\tau = 0.048$ s

(d) Report the measure of fit.

MOF = 0.995

Although this is not required in the exam, we also provide a comparison of the data and model for the physical variables.



Problem 3. (8 points)

In the data of a spectroscopy experiment two peaks are measured. The first peak corresponds to the abundance of compound A and the second peak corresponds to the abundance of compound B. The signal data is given in the file, http://utkstair.org/clausius/docs/mse301/data/xm4p03_f21.txt. In this data file, the first column is x, the second column contains the values of the peak 1 signal, and the third column contains the values of the peak 2 signal. perform the following tasks.

(a) What is the appropriate numerical method to integrate these signals?
(b) Find the integral of the first peak.
(c) Find the integral of the second peak.
(d) Find the relative abundance of compound A with respect to compound B.

Solution:

(a) What is the appropriate numerical method to integrate these signals?

A method for numerical integration, such as the trapezoidal rule should be used.

(b) Find the integral of the first peak.
(c) Find the integral of the second peak.
(d) Find the relative abundance of compound A with respect to compound B.

I wrote a little script in the file xm4p03_f21.m, which contained the following text

```
clear all;
close all;

datamat = [3.00 8.1225E-10  8.2242E-32
3.01    1.1583E-09  1.0682E-31
3.02    1.2522E-09  1.4538E-31
3.03    1.5841E-09  1.8082E-31
...
11.98   1.2230E-117 2.8776E-08
11.99   5.1707E-118 2.9705E-08
12.00   2.2145E-118 2.6513E-08];

x = datamat(:,1);
peak1 = datamat(:,2);
peak2 = datamat(:,3);
n = length(x);
dx = x(2) - x(1);

% trapezoidal rule for peak one
peak1_first = peak1(1);
peak1_last = peak1(n);
peak1_mid = 0.0;
for i = 2:1:n-1
    peak1_mid = peak1_mid + peak1(i);
end
integral_peak1 = 0.5*dx*(peak1_first + peak1_last + 2.0*peak1_mid)

% trapezoidal rule for peak one
peak2_first = peak2(1);
```

```
peak2_last = peak2(n);
peak2_mid = 0.0;
for i = 2:1:n-1
    peak2_mid = peak2_mid + peak2(i);
end
integral_peak2 = 0.5*dx*(peak2_first + peak2_last + 2.0*peak2_mid)

ratio1to2 = integral_peak1/integral_peak2

% plot optional
figure(1)
plot(x,peak1,'ro');
hold on;
plot(x,peak2,'bo');
xlabel('x');
ylabel('signal');
legend('peak 1','peak 2');
```

At the command line prompt, I typed the command

```
>> xm4p03_f21
```

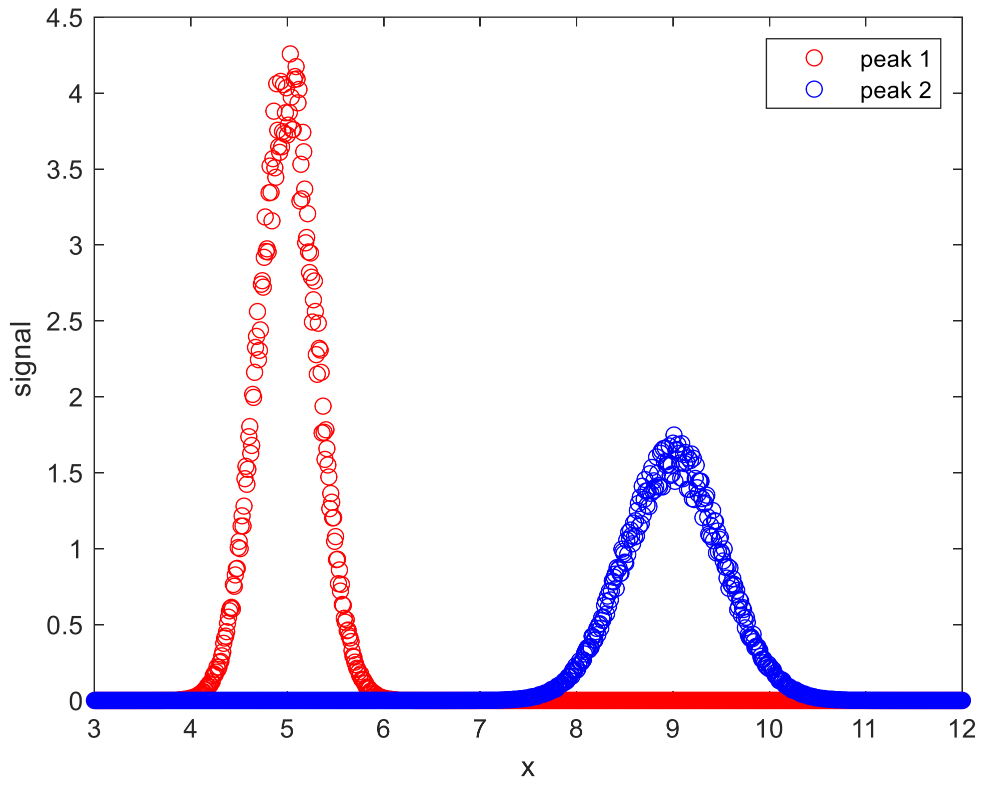
which generated teh following output:

```
integral_peak1 =    3.0067
integral_peak2 =    2.0043
ratio1to2 =    1.5002
```

Thus, the integral of Peak 1 is 3.01.
Thus, the integral of Peak 2 is 2.00.
The relative abundance is given by the ratio of the integrals and is thus 1.50.

There is 1.5 times more compound A than compound B.

This script also generated an optional plot of the data.



peak 1
peak 2
signal
x
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
3
4
5
6
7
8
9
10
11
12

Problem 4. (10 points)
Consider a mixture where the volume of the mixture, V_{mix}, is given by the sum of the pure component molar volumes, V_i, weighted by the mole fraction, x_i.

$$V_{mix} = \sum_{i=1}^{n_c} x_i V_i \tag{1}$$

where n_c is the number of components. The enthalpy of the mixture, H_{mix}, is nonideal,

$$H_{mix} = \sum_{i=1}^{n_c} \sum_{j \ge i}^{n_c} \left(2 - \delta_{ij}\right) \Omega_{ij} x_i x_j \tag{2.a}$$

where δ_{ij} is the Kronecker delta function and is 1 for $i = j$ and 0 for $i \neq j$. For a three-component mixture, this equation becomes

$$\begin{aligned} H_{mix} = \Omega_{AA} x_A x_A &+ 2\Omega_{AB} x_A x_B + 2\Omega_{AC} x_A x_C \\ &+ \Omega_{BB} x_B x_B \quad + 2\Omega_{BC} x_B x_C \\ &\qquad\qquad\qquad + \ \Omega_{CC} x_C x_C \end{aligned} \tag{2.b}$$

(This is one equation split into three lines for ease in reading.) The mixing parameters, $\Omega_{ij} = \sqrt{H_i H_j}$, where H_i and H_j are the pure component enthalpies. As a reminder, the sum of the mole fractions is unity.

$$1 = \sum_{i=1}^{n_c} x_i \tag{3}$$

The molar volumes and enthalpies of the pure components and mixtures are given below.

component	A	B	C	mixture
molar volume (liter/mol)	18	14	12	16.0
enthalpy (kJ/mol)	41	64	52	50.0

(a) Is this system of algebraic equations linear or nonlinear? (2 pts)
(b) What is the appropriate method to solve this system of algebraic equations? (2 pts)
(c) Determine the composition of this mixture. Show reasoning and method. (6 pts)

Solution:

(a) Is this system of algebraic equations linear or nonlinear? (2 pts)

This system of algebraic equations in nonlinear because of equation (2), where variables are multiplied together.

(b) What is the appropriate method to solve this system of algebraic equations? (2 pts)

The appropriate method to solve a system of nonlinear algebraic equations is the multivariate Newton-Raphson method.

(c) Determine the composition of this mixture. Show reasoning and method. (6 pts)

This set of non-linear algebraic equations can be written as follows.

$$f_1(x_A, x_B, x_C) = x_A V_A + x_B V_B + x_C V_C - V_{mix}$$
$$f_2(x_A, x_B, x_C) = x_A + x_B + x_C - 1$$
$$\begin{aligned} f_3(x_A, x_B, x_C) = \Omega_{AA}x_A x_A + 2\Omega_{AB}x_A x_B + 2\Omega_{AC}x_A x_C \\ + \ \Omega_{BB}x_B x_B + 2\Omega_{BC}x_B x_C \\ + \ \Omega_{CC}x_C x_C - H_{mix} \end{aligned}$$

I will solve this using the Newton Raphson Method with Numerical Approximations to the Derivatives, as implemented in the code `nrndn.m`.

This code requires that I input my system of nonlinear algebraic equations in the function, `funkeval.m`.

```
function f = funkeval(x)
%
%  these two lines force a column vector of length n
%
n = max(size(x));
f = zeros(n,1);
%
%  enter the functions here
%
VA = 18.0;
VB = 14.0;
VC = 12.0;
Vmix = 16.0;
HA = 41.0;
HB = 64.0;
HC = 52.0;
omega_AA = sqrt(HA*HA);
omega_AB = sqrt(HA*HB);
omega_AC = sqrt(HA*HC);
omega_BB = sqrt(HB*HB);
omega_BC = sqrt(HB*HC);
omega_CC = sqrt(HC*HC);
Hmix = 50.0;
f(1) = VA*x(1) + VB*x(2) + VC*x(3)  - Vmix;
f(2) = x(1) + x(2) + x(3) - 1.0;
f(3) = omega_AA*x(1)*x(1) + 2*omega_AB*x(1)*x(2) + 2*omega_AC*x(1)*x(3) ...
                          +   omega_BB*x(2)*x(2) + 2*omega_BC*x(2)*x(3)  ...
                                                 +   omega_CC*x(3)*x(3) - Hmix;
```

The Newton Raphson method requires an initial guess. I will use [1/3,1/3,1/3] as my initial guess. I want the tolerance to be 1.0^{-6}. I set the print flag to 1. At the command line prompt, I executed the following commands:

```
clear all;
x0 = [1.0/3.0,1.0/3.0,1.0/3.0];
tol = 1.0e-6;
iprint = 1;
```

```
[x,err,f] = nrndn(x0,tol,iprint)
```

At the command line prompt, I executed the script

```
>> xm4p04_f21
```

This command provided the following output.

```
iter =    1, err =  1.90e-01 f =  1.34e+00
 iter =    2, err =  8.44e-04 f =  1.01e-02
 iter =    3, err =  7.40e-08 f =  8.88e-07

x =
   0.541104832846414   0.376685501460758   0.082209665692828

err =     7.397931484967102e-08
```

Because the error is less than the specified tolerance, the Newton Raphson method has converged. Therefore the composition of the mixture is given by

$$\underline{x} = \begin{bmatrix} x_A \\ x_B \\ x_C \end{bmatrix} = \begin{bmatrix} 0.5411 \\ 0.3767 \\ 0.0822 \end{bmatrix}$$

This solution is not very sensitive to the initial guess. All of the following initial guesses converged to this solution.

```
x0 = [1.0/3.0; 1.0/3.0; 1.0/3.0];
x0 = [0.5; 0.25; 0.25];
x0 = [0.25; 0.5; 0.25];
x0 = [0.25; 0.25; 0.5];
```