**ChE 310** **Problem Set 6 (15 pts)** **Due Wed 10/16/19**

Collect all m-files in a single .zip file and upload the .zip file to the course webpage by midnight on Wednesday, October 16, 2019. Please note any collaborations in the code comments. Each student must upload their own individual copy of the work.

Keep in mind that book problems below correspond to the 3rd edition of Chapra's text; the 4th edition may have different problem numbers.

**6_1 (6 pts)** **Optimization.** These are shorter problems, so submit them all within one m-file PS6_1.m.

(A) Problem 7.14 from the text. Report the optimal insulation thickness to the command window; also report the temperature of the wire at the optimum thickness value.
(B) Problem 7.26 from the text. Have the answer reported to the command window in this format: "The max growth rate of _____ d^-1 occurs at concentration of _____ mg/L".
(C) Problem 7.30 from the text. Print the location of the peak concentration as well as the value of the peak concentration to the command window.

***Solution:***

```
%PS6_1.m
%(A) Solution to Chapra 7.14
clear; clf;
%Input parameters
q = 75; %W/m
r_w = 6e-3; %m
k = 0.17; %W/m*K
h = 12 ; %W/m^2*K
T_air = 293 ; %K

%Input function
T = @(r_i) T_air + q/(2*pi)*(1/k*log((r_w+r_i)/r_w)+1./(h*(r_w+r_i)));

%Optional, but good idea: plot to determine initial guess
fplot(T,[0 .1])

%We have a few tools available to us... let's use fminbnd since bounds
are
%clear from the plot.

disp('Part A')
[r_i_min T_min] = fminbnd(T,0.005,0.015)

%(B) Solution to Chapra 7.26
clear; figure();
%Input function
g = @(c) 2*c./(4+0.8*c+c.^2+0.2*c.^3);
%Optional, but good idea: plot to determine initial guess
```

```
fplot(g,[0 5])

%Let's try fminsearch this time.
f = @(c) -g(c);
[c_max f_min] = fminsearch(f,1.5);
g_max = -f_min;
disp('Part B')
fprintf('The max growth rate of %4.3g d^-1 occurs at concentration of
%4.3g mg/L.\n',g_max,c_max)

%(C) Solution to Chapra 7.30
clear; figure()
%Input function
c = @(x,y) 7.9 + .13*x + .21*y - .05*x.^2 - .016*y.^2 - .007*x.*y;

%Optional, but again we can plot over the suggested range.
x = linspace(-10,10); y = linspace(0,20);
[X,Y] = meshgrid(x,y);
C = c(X,Y);
surfc(X,Y,C);
xlabel('x')
ylabel('y')

%The surface appears to behave well, so a good guess in that range
should
%converge to the desired result with fminsearch.
c_mod = @(V) -c(V(1),V(2));
[V_min c_mod_min] = fminsearch(c_mod,[1 6]);

disp('Part C')
x_max = V_min(1)
y_max = V_min(2)
c_max = -c_mod_min
```

**Outputs:**

```
>> PS6_1

Part A

r_i_min =

    0.0082

T_min =

  423.5398

Part B
```

```
The max growth rate of 0.37 d^-1 occurs at concentration of 1.57
mg/L.

Part C

x_max =

    0.8537

y_max =

    6.3758

c_max =

    8.6249
```

**6_2 (3 pts)** **Mass balance/linear systems**. Solve in script PS6_2.m.

Solve problem 8.9 from the text. Include your mass balance derivations either as comments in the m-file or as an image upload (include the problem number in the image name). Output your vector of concentrations to the command line.

***Solution***:

```
%PS6_2.m
%Solution to Chapra 8.9
clear
%Input given flow rates
Q01 = 6 ; Q03 = 7 ; Q12 = 4 ;
Q15 = 5 ; Q23 = 2 ; Q24 = 1 ;
Q25 = 1 ; Q31 = 3 ; Q34 = 6 ;
Q44 = 9 ; Q54 = 2 ; Q55 = 4 ;

%Input the known inlet concentrations
c01 = 20 ; c03 = 50 ;

%Write mass balance equations on each reactor
%Rearrange to put them in the form:
%__ c1 + __c2 + __c3 + __c4 + __c5 = __
%Reactor 1
%Q01c01 + Q31c3 = Q15c1 + Q12c1
%(Q15+Q12)c1 - Q31c3 = Q01c01
%Reactor 2
%Q12c1 = Q25c2 + Q24c2 + Q23c2
%-Q12c1 + (Q25+Q24+Q23)c2 = 0
%Reactor 3
%Q03c03 + Q23c2 = Q31c3 + Q34c3
%-Q23c2 + (Q31+Q34)c3 = Q03c03
%Reactor 4
%Q24c2 + Q34c3 + Q54c5 = Q44c4
```

```
%-Q24c2 -Q34c3 + Q44c4 - Q54c5 = 0
%Reactor 5
%Q15c1 + Q25c2 = Q54c5 + Q55c5
%-Q15c1 - Q25c2 + (Q54+Q55)c5 = 0


%Convert to matrix notation and solve
A = [(Q15+Q12) 0 -Q31 0 0;
     -Q12 (Q25+Q24+Q23) 0 0 0;
     0 -Q23 (Q31+Q34) 0 0;
     0 -Q24 -Q34 Q44 -Q54 ;
     -Q15 -Q25 0 0 (Q54+Q55)] ;

b = [Q01*c01 ; 0 ; Q03*c03 ; 0 ; 0] ;

c = A\b
```

***Output:***

```
>> PS6_2


c =

   28.4000

   28.4000

   45.2000

   39.6000

   28.4000
```

**6_3 (4 pts)** **Mass balance/linear systems.** Solve in script PS6_3.m.

A counter-current staged liquid-liquid extraction process is depicted below. In such systems, a stream containing a weight fraction $y_{in}$ of a chemical enters from the left at a mass flow rate of $F_1$ (this is known as the *raffinate*). Simultaneously, a solvent carrying a weight fraction $x_{in}$ of the same chemical enters from the right at a flow rate of $F_2$ (this is known as the *extract*). Thus, at steady state for stage $i$, a mass balance can be written as:

$$F_1 y_{i-1} + F_2 x_{i+1} = F_1 y_i + F_2 x_i \qquad \text{(ps6.3.1)}$$

At each stage, an equilibrium is assumed to be established between $y_i$ and $x_i$, such that

$$K = \frac{x_i}{y_i} \qquad \text{(ps6.3.2)}$$

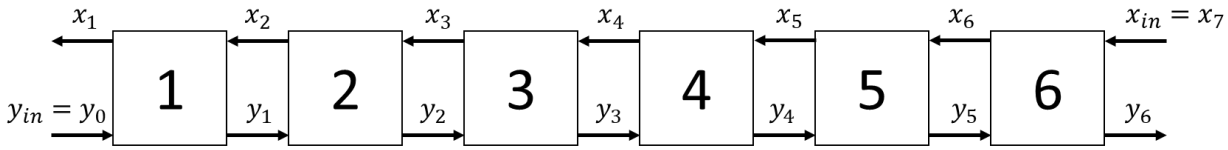
where $K$ is the distribution coefficient. Equation ps6.3.2 can be solved for $x_i$, and the result substituted into Equation ps6.3.1 and rearranged to yield:

$$y_{i-1} - \left(1 + \frac{F_2}{F_1}K\right) y_i + \left(\frac{F_2}{F_1}K\right) y_{i+1} = 0 \qquad \text{(ps6.3.3)}$$

Known parameters include: $F_1 = 200\,\frac{kg}{h}$, $y_{in} = 0.2$, $F_2 = 300\,\frac{kg}{h}$, $x_{in} = 0$, and $K = 5$.

Set up the matrix system for determining the value of $y_i$ at each stage. Note that equations (ps6.3.1) and (ps6.3.2) should be used to set up the mass balance for the last stage. Solve this problem using (i) 'tridiag', (ii) 'gausspivot', and (iii) '\' algorithms. Which is most efficient? Output the time taken for each to the command window. (Computation time can be quantified using the 'tic' and 'toc' commands.)

Then, calculate the corresponding value of $x_i$ at each stage. Create a single plot of $\log(x_i)$ and $\log(y_i)$ vs. stage number; label your plot (axes, legend, title) accordingly.




***Solution***:

```
%PS6_4.m

%Input known constants
F1 = 200 ; %kg/h
F2 = 300 ; %kg/h
y0 = 0.2 ;
x7 = 0 ;
K = 5 ;

%Write the mass balances.
%The first 5 are reasonably straightforward from ps6.4.3:
%y0 - (1+F2/F1 K)y1 + (F2/F1 K)y2 = 0
%...
%y4 - (1+F2/F1 K)y5 + (F2/F1 K)y6 = 0

%The last stage doesn't quite fit into the same formula,
%so use ps6.4.1 and manually rearrange using terms we know:
%F1*y5 + F2*x7 = F1*y6 + F2*x6
%x7 = 0 (from problem statement)
%F1*y5 - F1*y6 + F2*x6 = 0
%We know K = x6/y6, so x6 = K*y6
%F1*y5 - F1*y6 + F2*K*y6 = 0
%Divide by F1, so it has the same form as other equations
%y5 - (1+F2/F1 K)y6 = 0

%We now know our matrix A, which has 6 columns corresponding
```

```
%to the coefficients on the yi terms (i = 1:6)
%We use F2/F1 * K often - I'll call that c (for constant)
c = F2/F1 * K;
A = [-(1+c) c 0 0 0 0 ;
     1 -(1+c) c 0 0 0 ;
     0 1 -(1+c) c 0 0 ;
     0 0 1 -(1+c) c 0 ;
     0 0 0 1 -(1+c) c ;
     0 0 0 0 1 -(1+c)];
%If you noticed a pattern, this could be set up a bit faster:
% -(1+c) on the diagonals
A = -(1+c)*eye(6);
% 1 directly below the diagonal
A(2:end,1:end-1) = A(2:end,1:end-1) + diag(ones(5,1)) ;
% c directly above the diagonal
A(1:end-1,2:end) = A(1:end-1,2:end) + c*diag(ones(5,1));

%The b vector is pretty straightforward from our linear equations
b = [-y0 ; 0; 0; 0; 0; 0];

tic
y = chapra_tridiag([0 1 1 1 1 1],-(1+c)*ones(1,6),[c c c c c 0],b);
time_tridiag = toc

tic
y = chapra_GaussPivot(A,b);
time_GaussPivot = toc

tic
y = A\b;
time_leftdiv = toc

x = K*y ;

plot(1:6,log10(x),'bo--')
hold on
plot(1:6,log10(y),'ro--')
xlabel('stage')
xticks(1:6)
ylabel('log10 Mass Fraction')
legend('x_i','y_i')
```

*Output*:

```
>> PS6_4

time_tridiag =

   1.2820e-04
```

```
time_GaussPivot =

   5.1170e-04


time_leftdiv =

   9.0400e-05
```

**6_4 (2 pts) Interpolation/Optimization.** Solve in script PS6_4.m.

Density functional theory (DFT) calculations, such as those used in Dr. Roling's research group, can be used to elucidate properties of elementary chemical reactions. One popular method for determining activation energies ($E_a$) is the nudged elastic band (NEB) method, which determines the "best" pathway between two stable states. Sometimes the calculation goes a bit off track, and we need to quickly interpolate to approximate the true $E_a$.

We collect the following data for a simple reaction using our NEB/DFT code:

| Normalized Reaction Coordinate ($x$, arbitrary units) | 0.0 | 1.2 | 2.8 | 4.1 | 5.8 | 7.7 | 7.9 | 8.4 | 9.1 |
|---|---|---|---|---|---|---|---|---|---|
| Energy ($E$, eV) | 0.00 | 0.02 | 0.10 | 0.25 | 0.59 | 0.83 | 0.72 | 0.38 | 0.27 |

(1) Plot the energy as a function of reaction coordinate. As always, ensure your plot is properly labeled.

(2) Fit the data with a cubic spline, and plot your spline over the entire data range. Keep in mind that the endpoints are stable thermodynamic states, so $\frac{dE}{dx} = 0$ at these coordinates.

y = spline( [0 y 0])

(3) Use the spline function in combination with MATLAB's built-in optimization tools (i.e. fminsearch) to determine the $E_a$ predicted by your spline. Print $E_a$ along with its corresponding reaction coordinate to the command line. (Note: since the initial energy is zero, the activation energy is simply the maximum value along the reaction coordinate.) Confirm for yourself that the location of the minimum is correct by plotting this as a new data point on your plot.

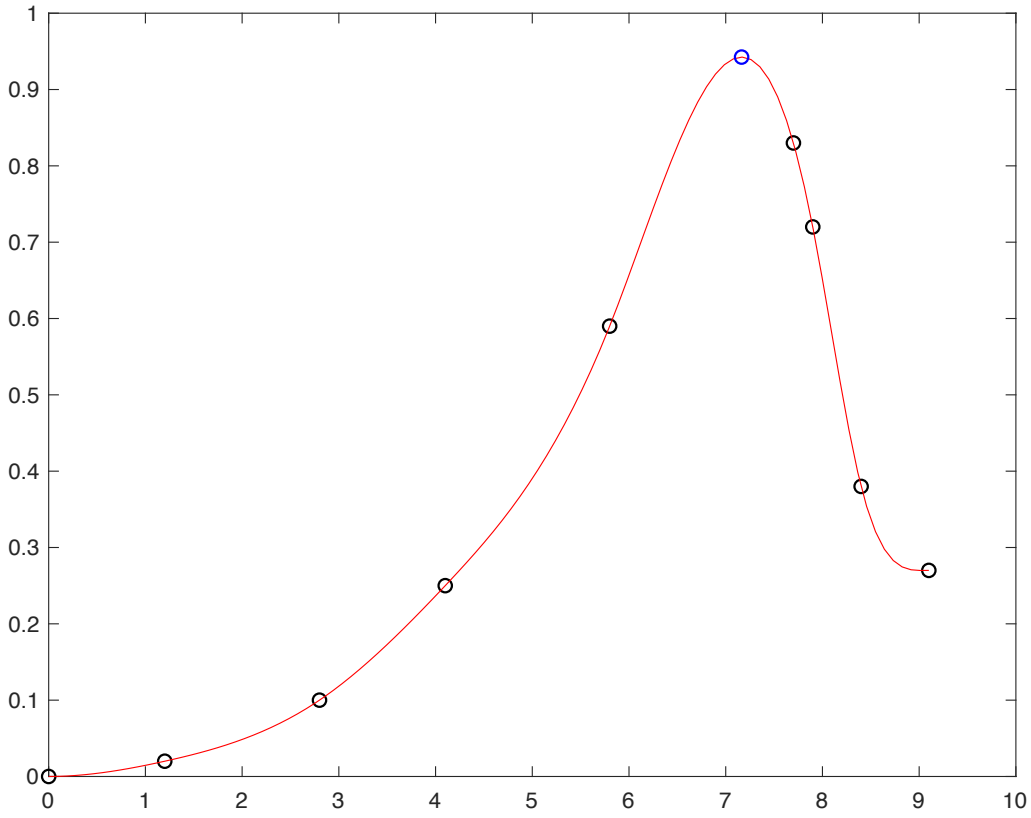
***Solution***:

```
%PS6_xx solution
%Coded by LTR on 10/6/19
clf ; clear ;
```

```
%Input data
x = [0 1.2 2.8 4.1 5.8 7.7 7.9 8.4 9.1];
E = [0 0.02 0.1 0.25 0.59 0.83 0.72 0.38 0.27];

%Plot data as points
plot(x,E,'ko')

%Cubic spline for plotting
%Note clamped form, since we know endpoint derivatives
x_spline = linspace(0,9.1);
E_spline = spline(x,[0 E 0],x_spline);
hold on
plot(x_spline,E_spline,'r-')

%Create function handle for minimization
%Must negate the function, since we're trying to get a
%maximum using fminsearch
f = @(xx) -spline(x,[0 E 0],xx);
%Use fminsearch, with a reasonable guess.
[xval, Ea_flipped] = fminsearch(f,7.9);
xval
Ea = -Ea_flipped
plot(xval,Ea,'bo')
```

Output:

```
>> PS6_4


xval =


    7.1641




Ea =


    0.9425
```


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7
8
9
10