ChE 323 Mass Transfer
Spring 2005

Homework # 2
Due Friday April 15

TA's: Pengfei Chen (Problem #1)
Manohar Murthi (Problem #2)

- Finish reading Chapter 2.
- Start reading Chapter 3.

1. (15 points) Problem 2.10

2. (45 points) Consider example 2.1.1 of your text. In this case, a liquid is present at the bottom of a tube. A stream of dry air runs across the mouth of the tube, which is 0.238 m from the surface of the liquid. The vapour pressure of the liquid is 68.4 kPa, while the total pressure is 99.94 kPa. The binary diffusion coefficient of the liquid species in air is $1.991*10^{-5}$ m^2/s. The temperature is 328.5 K and the gas phase behaves ideally.
   (a) Calculate and plot the concentration profile as a function of the position in the tube using the boundary value solver bvp4c in MATLAB
   (b) Include the analytical concentration profile on the same plot as (a).
   (c) Solve for the flux of the liquid species analytically. How does this compare to the flux found by MATLAB?
   (d) Compare the numerical and analytical solutions at 3 different locations in the tube.
   (e) Change the function used to guess the initial solution. Try several different functions and report the results. Do the same for different initial guesses of the flux.
   (f) Change the number of mesh points used and report how it affects the solution and the time taken to reach the solution.

Submit your MATLAB code with the above results.

<u>Notes on solving boundary value problems using MATLAB</u>
The function bvp4c is a code to solve boundary value problems numerically. It takes as input a function which describes the differential equation (e.g. hw2ode in the function call below), one which returns the boundary conditions (hw2bc), and a solution structure (solinit) which contains the initial guess for the solution. The function bvp4c may be called as follows:

```
solution = bvp4c(@hw2ode,@hw2bc,solinit)
```

The initial guess solinit must be a structure with fields x and y. x contains the ordered nodes of the initial mesh. Boundary conditions are imposed at a = solinit.x(1) and b = solinit.x(end). y contains the initial guess for the solution with solinit.y(:,i) a guess for the solution at the node solinit.x(i). The structure can have any name, but the fields must be named x and y. It can also contain a vector that provides an initial guess for unknown parameters. You can form solinit with the helper function bvpinit. See the reference page on bvpinit for further details.