Lab #3b -- More Programming with Matlab

Overview

This lab will focus on providing additional programming experiences with Matlab -- with some common **input/output operations**, the difference between using the **for … end** and the **while … end** looping structures, and the evaluation of **discrete equations**. As in the last lab, there will be no instructor-led problems today. Instead, most of the time will be spent on giving individual help, as needed, as you tackle the programming exercises given below -- so don't hesitate to ask the lab instructor any questions you may have…

Upon completion of this lab you should have sufficient background to address a full range of programming tasks, and be well on your way to becoming a competent problem-solver using Matlab…

Three Programming Tasks:

Your job today is to do your best to solve the following three programming problems. Be sure to follow any special instructions that are given, since the goal here is to illustrate several different features and specific programming tasks. Also be sure to complete these tasks outside of class if you run out of time during the lab period.

Problem 1 -- van der Waal's Equation of State

There are several different correlations that are used to relate the various properties of a gas -- these are referred to as the equations of state for the gas. One particular form, van der Waal's equation, is given by

$$P = \frac{RT}{\upsilon - b} - \frac{a}{\upsilon^2}$$

where R is the universal gas constant (0.08206 liters-atm/mole-K), T and P are the absolute temperature (K) and pressure (atm), respectively, and υ is the molar volume (liters/mole). The empirical constants, a and b, depend on the particular gas, and some values for a few common gases are given in the following table.

Gas	**a (L^2-atm/mole^2)**	**b (L/mole)**
He	0.0341	0.0237
H_2	0.244	0.0266
O_2	1.36	0.0318
Cl_2	6.49	0.0562
CO_2	3.59	0.0427

a. Write a user-defined function file that determines the gas pressure given the temperature, molar volume, and the specific gas of interest (these are all **scalar input parameters**). The position number within the table can be used to identify the gas (for example, O_2 is gas #3). Test your function for chlorine for T = 293 K and υ = 10 L/mole.

b. Now write a script file that allows the user to select the gas of interest from a pre-defined list of available gases (you should make use of the ***menu*** command here) and request user input for the values of T (in C) and υ (in L/mole) (use the ***input*** command here). The program should then call the function generated in Part a to evaluate and edit the gas pressure for the conditions given. Run the program for different gases (from the five gases listed in the above table) for a temperature of 20 C and molar volume 10 L/mole to make sure things are working as designed (you might want to check your numerical results with a hand calculation or two…).

Problem 2 -- Using Looping Structures within Matlab

a. Write and test a short Matlab code to evaluate the following series for N = 25 terms:

$$S=\sum_{k=1}^{N}(-1)^{k+1}\frac{1}{2k-1}$$

b. Write and test a short Matlab code to evaluate the following infinite series, where **tol = 0.001** represents the limiting condition for stopping the infinite series to some finite number of terms -- that is, when the magnitude of the relative contribution of the next term in the series is less that **tol**, you should stop adding additional terms:

$$S=\sum_{k=1}^{\infty}(-1)^{k+1}\frac{1}{2k-1}$$

Problem 3 -- Implementation of Discrete Equations within a Matlab Function

Many engineering applications require integration of a function over some domain. Assume that data are available in discrete form over an even or uneven grid. In particular, a set of data are given as x_i,y_i pairs for i = 1, 2, ... N. These data represent an approximate relationship for y(x) over the domain $x_1 \le x \le x_N$. A simple approximation to the integral $I=\int_{x_1}^{x_N} y(x)dx$ can be written as

$$I=\sum_{i=1}^{N-1}\begin{pmatrix}\text{average value of } y\\ \text{over interval } i\end{pmatrix}\begin{pmatrix}\text{width of}\\ \text{interval } i\end{pmatrix} \qquad \text{or} \qquad I=\sum_{i=1}^{N-1}\left(\frac{y_{i+1}+y_i}{2}\right)\left(x_{i+1}-x_i\right)$$

where N-1 is the number of intervals and N is the number of data points in data vectors **x** and **y**.

a. Your job is to write a complete Matlab function file to compute the definite integral using the above simple discrete formulation. Assume that **x** and **y** are already defined and that they are passed into the function from the main program. The only variable returned to the main program is I, which contains the result of the above discrete formula. For consistency, assume that the first line of the function is given by

function I = dint(x,y)

b. Now write a main program that calls your **dint.m** function file to evaluate

$$I = \int_0^2 e^{-x} dx$$

for a discrete grid containing M evenly-spaced intervals -- use Matlab's input command to interactively input the value of M. Make sure everything is properly defined and is consistent with the **dint.m** function written in Part a. Simply print the final result to the screen. Note that you can check your code by comparing the computational result to the exact value of the integral (i.e., as M becomes large, I should approach the exact value of the integral).