CH2013
Computational Programming and Simulations Lab
Aug-Dec 2021

Problem Sheet #3

September 1 2021

In this problem sheet, we learn about structure arrays. They are helpful for sure in large codes with multiple functions. We also learn that MATLAB has some cool tools for solving non-linear equations in an efficient manner, and examine ways and means of harnessing various options and exitflags we have at our disposal.

1. a) Define a function "myfun" which takes x as an input and calculates f(x), f'(x) and f''(x) within it. The $f(x) = x^3 - 10x^2 + 33x - 36$.
   b) In the script file, create a vector X that spans from 0 to 5 in increments of 0.1. Find the values of f, f' and f'' for each element of X and put them in all in a matrix F. The first column of F should contain the function values, the second column the first derivative, and the last column the second derivative.
   c) Plot f(x),f'(x) and f''(x) vs x in the same plot.

2. The Newton's Method hinges on using the equation $x_{new} = x_{old} - \frac{f(x_{old})}{f'(x_{old})}$ to solve for the root(s) of $f(x) = 0$. Of course for any given function f(x), you have to derive the expression for f'(x) and then include it in your code. Use the function "myfun" created above to calculate *f(x)* and *f'(x)* at every iteration. **In this question, we will use a while loop. The logic of the while loop is**

   while condition/expression
   statements
   end

   This will enable to execute the 'statements' block as long as the 'condition' is true. One of the statements has to be incrementing loop index value (i.e. i=i+1).

   In this case, the 'condition' is that **% relative error is > TolX OR abs(f(x)) > TolF.** When the 'condition' becomes false, we will exit the loop.

   After exiting the loop, save the final value of x in xfinal1 and the final value of **f** in ffinal1 and error in error1. Also keep the iteration number (or while loop index) in iter2.

   Use TolX = 10^{-6} and TolF = 10^{-12} and initial guess of x=2. *(At initial guess, please set the while loop counter at 1.)*

   Now change the initial guess to x=100. Find xfinal2, ffinal2, error2 and iter2 to correspond to this initial guess.

3. Structure arrays offer a compact notation for sharing data across various parts of the MATLAB code. Look up the syntax, how to set up the values, and how to access each element of the array, carefully.

   a) Define a structure array called "**reactiondata**" to set up values of the pre-exponentials and activation energies for two reactions – in the script file. The pre-exponentials should be in a (vector) field called '**k0**' and the activation energies in a (vector) field called '**Eact**'. The values are 0.001 and 0.34 for the pre-exponentials, and 12000 and 5600 for the activation energies, respectively.

b) The reaction rates can be calculated as

$$ri = k_{0i}\exp\left(-\frac{E_{act,i}}{RT}\right)C_1C_2$$

(same expression for both the reactions, only the values of k0 and Eact are different; R = 8.314 in consistent units)
C_1 and C_2 are two concentrations; T is the temperature

c) Write a function 'reactionrate1' which has C_1, C_2, T, and the structure array reactiondata as inputs and the outputs are r1 and r2.

d) Call the function 'reactionrate1' to get the values of reaction rate 1 & 2 (call it rate1 and rate2), for the given data - C_1 and C_2 are can be assumed be to 0.1 and 0.01 respectively. T varies between 300 & 1000 (consistent units; 100 units intervals). Plot rate1 and rate2 vs. T (use semi log y).

4. a) Define a structure array called 'data' to set up the coefficients of a polynomial $f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$; the values of a_i are -36,33,-10 & 1, respectively ***(sound familiar?)***

b) Create a function 'myfun1' which takes x and data as inputs and gives f(x) as the output.

c) Use ***fzero*** to solve the f(x) above. Use myfun1 to get f. Use an initial guess of 0; set up the options so that the iterations are displayed, make sure that the function tolerance is changed to 1.e-6 and X tolerance to 1.e-2, and report the answers in xfinal3, ffinal3, iter3. *(I am purposely making these tolerance values high so that we get some not very accurate results. After you submit, you can play around with the tolerances to get better solutions; also note that we have a "parametrized" function now so the command for fzero is slightly different. Help documentation has this at bottom).*

d) Now try to use fzero to solve the same problem, but give the initial guess as x0=[1 2]. What happens? *(no submission – but spend time to understand the issue & try to tackle it)*

e) Read up the documentation for ***fsolve*** - we will use it in the next class, but do make some notes about various aspects of it. We will use it to solve F(X)=0 (vector equation) and not for any other application, right now.

--END--