Applied Numerical Computing for Scientists and Engineers

Computational Assignment 2

Electronic submission via Bitbucket and course website.
Assignment weight: 5%.

The purpose of this assignment is to give you practice with programming in MATLAB while developing best practices for scientific computing. Exercise 1 uses the files you generate in Exercise 2.

1. Git (5%)

Use your bitbucket repository named FirstnameLastnameApplNumComp (change firstname and lastname to your own name) for version control with the .m and .html files associated with Exercise 2. Create a subfolder called "CA 2". Work on Exercise 2 of this assignment in the "CA 2" subfolder. There should be at least one commit of each required file and at least three total commits for this assignment with comments that briefly explain states of progress on the assignment, e.g., "outline of MATLAB function for Exercise 2". The last commit you wish to submit for a grade should have the commit -m message "assignment 2 submission". Everything between the assignment 1 submission commit (or the start of the repository if you didn't complete the git portion of assignment 1) and the assignment 2 submission commit will be evaluated.

For the course website Computational Assignment 2 submission, use the text box to enter the web address for the commit that corresponds to the submission. For example, `https://bitbucket.org/ashleefv/ashleefordversyptapplnumcomp/commits/d8390344f1b0ef0faed7db84c01ffce95c2f0423`. This is simply to have a clear time stamp for your submission. **Submissions and/or final**

commits after the deadline submitted on the due date will receive maximum of half credit for the assignment.

2. MATLAB (95%)

This Exercise has two parts: writing a .m file in MATLAB and generating a .html file through the publish feature.

2.1. .m file (75%)

Write a MATLAB function titled `system_of_ODEs.m` to define a system of ordinary differential equations

$$\frac{dC_A}{dt} = -k_1 C_A - k_2 C_A \tag{1}$$

$$\frac{dC_B}{dt} = k_1 C_A - k_3 - k_4 C_B \tag{2}$$

which describe the mass concentrations of species A and B subject to chemical reactions. In (1) and (2), the units are hours for t and mg L^{-1} for C_A and C_B. Note: `system_of_ODEs.m` is one function that includes both (1) and (2).

The desired function header (first line) is

```
1 function output = system_of_ODEs(t,C,k1,k2,k3,k4)
```

where `t` is a single time value; `C` is a 1×2 matrix (row vector) of concentrations; `k1`, `k2`, `k3`, and `k4` are parameter values; and `output` is a 2×1 matrix (column vector) of differential equations evaluated at time `t`. However, to make the function more flexible, the function should accept variable arguments using `varargin`. So the required function header is

```
1 function output = system_of_ODEs(varargin)
```

Use the `if ... else ... end` flow control structure to take the input `varargin` and translate it to the desired quantities `(t,C,k1,k2,k3,k4)` for three different cases while minimizing code duplication (if you feel the

need to copy & paste any parameter or variable values, there is a better way). If all are specified in the function call, connect elements of `varargin` to the proper variable and parameter names. If only `t` and C are supplied, default values for the rate constants k_1, k_2, k_3 and k_4 should be specified as listed in Table 1. If t and C are not supplied, their default values are $t = 0$ h, $C_A = 6.25$ mg L^{-1}, and $C_B = 0$ mg L^{-1}.

Table 1: Default rate constant values to be used if no parameter values specified.

Rate Constant	Default Value	Units
k_1	0.15	h^{-1}
k_2	0.6	h^{-1}
k_3	0.1	mg L^{-1} h^{-1}
k_4	0.2	h^{-1}

2.2. .html file (20%)

Use the publish feature in MATLAB to generate an html file and to evaluate the function at the default values. Note: you should **NOT** solve the differential equations, just evaluate the derivatives with the default values of the variables and parameters. Your code and the html output should demonstrate the following coding best practices:

- Write thorough comments and documentation
- Define the purpose of your function,the author, and all input and output parameters or variables at the top of the code
- Define the problem you are solving using comments (with the publish feature you can and **must** include LaTeX equations for grading purposes)
- Use descriptive variable names
- Indent code and use whitespace
- Avoid duplication: "Define once, and reuse often"

- Specify the units for physical quantities
- Suppress MATLAB printing to the command window with use of semicolon
- Specify default values while allowing for parameter values to be passed to the function by a user calling the function