AMTH 118 - Numerical Methods - Spring 2022
Project - Due 3 June 2022

CAREFULLY READ THIS PROJECT DESCRIPTION FROM BEGINNING TO END !

Define

$$f(x) := 1 + \sum_{j=1}^{n} \frac{b_j}{d_j - x} ,$$

where $b_j > 0$ for all j and $d_1 < d_2 < \cdots < d_{n-1} < d_n$. This function has n poles at each d_j, and also n roots, one in each interval (d_j, d_{j+1}) for $j = 1, \ldots, n-1$, and one in $(d_n, +\infty)$. In addition, $d_i = 0$ for some given index i with $2 \leq i \leq n-1$. The equation $f(x) = 0$ is known as a *secular equation*. Solving such an equation lies at the heart of the "Divide and Conquer" method (Cuppen, J.J.M., "A divide and conquer method for the symmetric tridiagonal eigenproblem", Numerische Mathematik, 36 (1981), 177–195) to compute the eigenvalues of a tridiagonal symmetric matrix, the last and crucial step in the computation of the eigenvalues of a general symmetric matrix. It is one of the fastest known methods to accomplish this.

Your goal is to compute the root of f in the interval $(d_i, d_{i+1}) = (0, d_{i+1})$ with a method that is simple, fast, and accurate. Methods such as the Bisection or Regula Falsi methods are too slow, and faster methods such as the Secant or Newton methods can easily diverge. There are several options to deal with these problems. Here, we use rational approximations to construct a root-finding method, as explained below.

Write the function f as $f = \varphi + \psi$, with

$$\varphi(x) := 1 + \sum_{j=1}^{i} \frac{b_j}{d_j - x} \quad \text{and} \quad \psi(x) := \sum_{j=i+1}^{n} \frac{b_j}{d_j - x} ,$$

and construct an iterative numerical method to compute the unique root of f on the interval $(0, d_{i+1})$ by approximating f by $g + h$, where g approximates φ up to first order and h approximates ψ up to first order at a given iterate, with

$$g(x) := \alpha + \frac{\beta}{x} \quad \text{and} \quad h(x) := \frac{\gamma}{\delta - x} .$$

In other words, at a given iterate z, the following conditions must be satisfied:

$$\begin{cases} \varphi(z) = g(z) \\ \varphi'(z) = g'(z) \end{cases} \quad \text{and} \quad \begin{cases} \psi(z) = h(z) \\ \psi'(z) = h'(z) . \end{cases}$$

This will determine the parameters α, β, γ, and δ, which, in turn, define the functions g and h. The next iterate is, of course, the root of this approximation. This method also requires a suitable starting point in your interval that lies *to the right of the root*. Find one, it is worth 15%. It can be shown that convergence from such a starting point will be monotonic and quadratic. *You do not have to prove this!* If you can't find such a starting point, use the midpoint of the interval. The stopping criterion for your method should be

$$\left| \frac{z_{k+1} - z_k}{z_{k+1}} \right| \leq tol \ ,$$

where tol is a predefined tolerance.

<u>TASKS TO BE CARRIED OUT:</u>

(1) Write a single Matlab function called *seceq.m* that implements the above method to compute the root of f in the interval $(0, d_{i+1})$. It should be self-contained, i.e., it should not contain any other user-defined functions, and it should have the following *input* arguments:

- argument 1 : the number n
- argument 2 : the index i
- argument 3 : the vector b
- argument 4 : the vector d
- argument 5 : the tolerance tol.

It should have the following *output* arguments:

- argument 1 : the root in the interval $(0, d_{i+1})$.
- argument 2 : the number of iterations (including the initial point) it took to compute the root.

(2) Include a pdf file in your submission, detailing your mathematical calculations to obtain the coefficients, and explaining how you found the initial point if found one. Please use the same notation as we have used here. This may be (neatly!) handwritten. Scan and submit together with the rest of your work.

SUBMISSION INSTRUCTIONS AND NOTES:

- In future releases, Matlab will no longer support certain commands that currently generate warnings. This is a technical matter that is of no concern to us. Do not worry about it. Warnings can be turned off with the command "warning('off','all')".
- Upload your function *seceq.m* file to Camino. Do not send it via email. Also, if applicable, submit your notes for the starting point as a scanned .pdf file. Write your name at the top of your notes. Check carefully before submitting as you can submit *only once*. No late submissions will be accepted.
- Your function should be a standard Matlab function, i.e., it should not print out any intermediate results or anything else not asked for, and it should not solicit user input (an immediate and logical consequence of the very fact that the function already has input arguments). Make sure that the input and output arguments are exactly as stipulated.
- Using the Matlab "roots" function is not allowed.
- Include your name at the top of your function file in a comment.
- Your code needs to be properly indented and commented.
- Feel free to look up Matlab functions, but don't forget that copying other people's work is considered plagiarism and is therefore (obviously) not allowed.
- The use of toolbox commands is not allowed. When in doubt, type "which *command*" and it will tell you if it is a toolbox command. (Try "which roots" and "which sym" and compare the subfolders that contain the command: if the subfolder immediately after "toolbox" is "matlab", it is not a toolbox command. If anything else appears immediately after "toolbox", it is a toolbox command.)
- Make sure that your code does not generate Matlab errors, as this will result in a zero score.
- *You may not discuss any aspect of this project with anyone. This project is an individual effort!*
- *Do not wait until the last moment to work on your project!*
- *Please follow these instructions to the letter!*

Due 3 June 2022 at 23:00.