CORRECTED Take-home Final Exam

Due 5pm on Friday 5 May 2017, in my office box (Chapman 101).

100 points total. As stated on the syllabus, this exam is 20% of your course grade.

Rules. This take-home exam must be only *your own work*. You may not talk or communicate about it with any person other than me, Ed Bueler. You *are* encouraged to ask me questions about the exam during lecture time, or during my office hours. You may use any reference book or article, print or electronic, as long as it is clearly cited. You may not seek complete online solutions.

This exam is not intended to be significantly-longer than a typical Assignment, but you should devote care to it, because it is a large fraction of the grade. Please ask me questions about it!

F1. a) (*5 pts*) Consider the advection-diffusion equation

$$u_t + bu_x = Du_{xx} + f(x) \tag{1}$$

for $u(x,t)$, defined on $0 \le x \le 1$ and $0 \le t \le t_f$, with zero Dirichlet boundary conditions $u(0,t) = 0$ and $u(1,t) = 0$. Let $h = 1/(m+1)$ and define the grid with $x_j = jh$ for $j = 0, 1, \ldots, m+1$. Applying centered differences to the space derivatives, and eliminating the boundary values (as knowns), give the details of the method-of-lines (MOL) system

$$U(t)' = A\,U(t) + F. \tag{2}$$

Note $U(t) \in \mathbb{R}^m$, $F \in \mathbb{R}^m$, and A an $m \times m$ tridiagonal matrix. Specifically, show A and F.

b) (*10 pts*) Now, for specific values $b = 10$ and $D = 1$, and $h = 0.05$ (so $m = 19$), use MATLAB's `eig()`, or similar numerical eigenvalue tool, to compute and plot the eigenvalues λ_p of A. Then, for each of the following grids,

$$h = 0.1,\ 0.05,\ 0.01,\ 0.005,\ 0.001,$$

find the largest time-step k so that RK2 is absolutely-stable when applied to the MOL system (2). (*There is no need to make a plot for every h value. You will compute eigenvalues of $A = A_h$ numerically. Report your answer as a table with h, k values.*)

c) (*5 pts*) Implement the scheme, namely the numerical approximation of (2) using RK2 time-stepping, with initial condition $u(x,0) = \sin^{10}(\pi x)$ and source function $f(x) = e^{-x}$. (*Show the code. Suggested signature:* `[x,U]=rk2ad(h,k,tf)`.)

d) (*10 pts*) Using the $h = 0.001$ grid compute the numerical approximation of

$$u(x = 0.8, t_f = 0.1) \qquad \leftarrow \textit{(CORRECTED)}$$

to five digits past the decimal point. (*Get the right answer. No points for the wrong answer.*)

(*In implementing parts* **b)** *and* **d)**, *errors may occur that are not obvious. You can get results that look reasonable and yet are wrong. One way to verify is to generate a time-dependent exact solution of a closely-related problem. Feel free to pursue this as a way to be confident in your results above, but it won't directly be worth any points.*)

e) (*5 pts*) The formula

$$\tilde{u}(x) = c_1 + c_2 e^{10x} - \frac{1}{11} e^{-x}, \qquad c_1 = \frac{e^{10} - e^{-1}}{11(e^{10} - 1)}, \quad c_2 = -\frac{1 - e^{-1}}{11(e^{10} - 1)},$$

is an exact solution to the *steady-state* of the problem you solved in part **d)**. First, confirm this by hand. Next, write a few sentences to explain how to use this steady-state knowledge to most-effectively verify the code that went into solving parts **b)** and **d)**. (*What I want here is your*

explanation/analysis of the usefulness of an exact steady-state solution for the job of verifying time-dependent numerical results. Verification will be incomplete and uncertainties will remain. Feel free to apply your ideas to a code, but doing that will not be worth any points.)

F2. **a)** (*5 pts*) Consider the nonlinear Poisson equation in 2D

$$u_{xx} + u_{yy} + \gamma\, u^4 = f(x,y) \tag{3}$$

on the unit square $(x,y) \in [0,1] \times [0,1]$, subject to zero Dirichlet boundary conditions. We can manufacture a solution because we are free to choose the right-hand side $f(x,y)$. Let

$$u(x,y) = \sin(\pi x)\sin(3\pi y). \tag{4}$$

Compute $f(x,y)$ so that (4) is an exact solution of (3).

b) (*15 pts*) Based on the MATLAB program `poisson.m`, which is online at

bueler.github.io/M615S17/matlab/poisson.m,

or a similar code for the 2D linear Poisson equation, solve (3). Use this approach:

- Use centered differences, and spacing $h_x = h_y = 1/(m+1)$ to set up a nonlinear system

$$F(U) = 0 \tag{5}$$

  which approximates (3). Here $U \in \mathbb{R}^N$ with $N = m^2$. (Recall F is the "residual.")
- Solve the algebraic equations (5) by Newton's method. Use $U = 0$ as the initial iterate.

(*Note that when you set $\gamma = 0$ your code should solve the linear Poisson problem $u_{xx} + u_{yy} = f(x,y)$ by doing one Newton step. Also, for any γ the first Newton iterate will be the solution to the linear Poisson equation.*) Stop the Newton iteration when the residual norm is reduced by 10^{-9} of the initial residual norm. I suggest your code has signature `[x,y,U]=nlp(m,gamma,f)`, or similar, and you design it so that if f is not given as an argument, the verification case from part **a)** is used and the numerical error is reported; use the $\|\cdot\|_\infty$ norm.

c) (*5 pts*) Show, for the verification case based on the exact solution from part **a)**, that with both $\gamma = 0$ and $\gamma = 10$ your code exhibits the expected $O(h^2)$ convergence.

d) (*10 pts*) Now set $f(x,y) = -10$ (constant function) and $\gamma = 3$, so that you do *not* know the exact solution. Approximate the maximum value of $u(x,y)$ by computing the maximum of the numerical solution on an $m = 100$ grid (i.e. $N = 10^4$ unknowns); show four digits past the decimal point. (*Get the right answer. No points for the wrong answer.*)

F3. (*30 pts*) For the linear ODE system

$$U(t)' = AU(t) \qquad \text{where} \quad A = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & -200 \end{bmatrix}, \qquad \leftarrow \textit{(CORRECTED)}$$

with initial value $U(0) = [1\,1\,1]^\top$, find the exact solution. For this task you *may* use Wolfram alpha or similar. Specifically, find the exact value of $U(1)$. Then implement the TR-BDF2 scheme (equation (8.6) in the textbook) for this problem. (*Note that the scheme is implicit but your implementation can assume that the ODE system is linear and autonomous and homogeneous. You will not need Newton's method.*) By using the exact solution, give a convergence graph which shows convincingly that your implementation converges at the expected rate, namely $O(k^2)$. Finally, give concrete and convincing evidence that the scheme is more efficient than RK2 if you seek only five-digit-accurate results for $U(1)$. (*You should run both TR-BDF2 and RK2 with a fixed step size. Your convergence graph will tell you the step size needed for TR-BDF2. For RK2 you will have to respect an absolute-stability restriction.*) Your grade will be determined, in substantial part, by the correctness and clarity of what you write.