Final Exam, 553.481/681, May 6, 2024

Do all **three** *of the following problems.* Show all your work. Answers without supporting work may receive no credit.

Students may discuss the exam **only** with the instructor and the teaching assistant. No discussion of the exam contents, directly or indirectly, is permitted among students or with any third parties. Any book or internet resource may be used, as long as the book or the website are cited, along with the material taken from it. However, please note that explanations must be correct and complete, whereas many online sources give only partial explanations or even contain errors. It is your responsibility to give full and accurate answers, demonstrating your own understanding.

You may use any numerical software available, unless you are specifically instructed in the problem statement to write your own code. All codes that are written by you should be turned in with the exam, either as paper printouts or preferably as a Matlab script sent by e-mail to the instructor. Numerical results without the code that produced them will receive no credit.

I attest that I have completed this exam without unauthorized assistance from any person, materials, or device:

Full name: ____

Signature: ____

(See the Johns Hopkins Handbook *Academic Ethics for Undergraduates*).

Problem 1 [20 points]. This problem studies the *Gauss-Lobatto rules* for integration, which modify usual Gaussian quadrature by including interval endpoints among the node points in order to enable composite integration.

(a) The 4-point Gauss-Lobatto rule on the symmetric interval $[-1, 1]$ is given by

$$\int_{-1}^{+1} f(x)\, dx \doteq w_1 f(-1) + w_2 f(-\xi_1) + w_2 f(\xi_1) + w_1 f(+1)$$

for specified ξ_1, w_1, w_2. The rule is exact $(= 0)$ for any odd function by the symmetry of the formula. Determine the constants ξ_1, w_1, w_2 by requiring that the formula be exact for the first three even powers, $f(x) = x^i$ with $i = 0, 2, 4$ and by then solving the resulting algebraic equations.

(b) Use the linear transformation $x \to a + (i + (x + 1)/2)h$ to transform the interval $[-1, 1]$ into $[a + ih, a + (i + 1)h]$ for integers $i = 0, ..., n - 1$ with $h = (b - a)/n$ and construct thereby the composite 4-point Gauss-Lobatto rule to calculate a general definite integral $I = \int_a^b f(x)\, dx$. Give the resulting formula and write a Matlab function code `lobattoc.m` to implement it in the format `I=lobattoc(f,a,b,n)`.

(c) The code in part (b) requires that function $f(x)$ be evaluated not only at the points $x_i = a + ih$, $i = 0, ..., n$ but also at $x_i^{\pm} = x_i + (1 \pm \xi_1)\frac{h}{2}$, $i = 0, ..., n - 1$ and is thus about as expensive as composite Simpson with $3n$ subintervals. Compare composite Gauss-Lobatto and composite Simpson for the integral

$$I = \int_0^{\pi} e^x \sin x\, dx = \frac{1}{2}(e^{\pi} + 1)$$

with $n = 10, 30, 90$ for Gauss-Lobatto and $n = 30, 90, 270$ for Simpson. Which is more accurate for the same computational cost?

Problem 2 [40 points]. (a) An inconvenient feature of the trapezoidal method as implemented in the course code `trapezoid.m` is that it uses the Newton-Raphson method to approximate the solution of the nonlinear equation $\mathbf{F}(\mathbf{y}_{n+1}) = \mathbf{0}$ for

$$\mathbf{F}(\mathbf{y}_{n+1}) = \mathbf{y}_{n+1} - \mathbf{y}_n - \frac{h}{2}[\mathbf{f}(t_n, \mathbf{y}_n) + \mathbf{f}(t_{n+1}, \mathbf{y}_{n+1})],$$

and thus requires the $d \times d$ Jacobian matrix $\mathbf{J}(t, \mathbf{y}) = \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(t, \mathbf{y})$. Write a new code `trapezoidsteff.m` which uses instead *Steffensen's method*, a quasi-Newton method employing the approximate derivative matrix

$$\frac{\partial F_i}{\partial y_j}(\mathbf{y}_{n+1}) \simeq \frac{F_i(\mathbf{y}^j_{n+1}) - F_i(\mathbf{y}^{j+1}_{n+1})}{F_j(\mathbf{y}_{n+1})}, \quad 1 \le i, j \le d$$

with $\mathbf{y}^j_{n+1} = (y_{n+1,1}, ..., y_{n+1,j-1}, y^*_{n+1,j}, ..., y^*_{n+1,d})$ and $\mathbf{y}^*_{n+1} = \mathbf{y}_{n+1} + \mathbf{F}(\mathbf{y}_{n+1})$. Show for a scalar function $F(y_{n+1})$ that this scheme reduces to Steffensen's method to find the fixed point of $G(y_{n+1}) = y_{n+1} + F(y_{n+1})$ and thus converges quadratically. Write your code `trapezoidsteff.m` to output also the average number of function evaluations per timestep and take care to minimize that number.

(b) The course code `pece2.m` implements the 2nd-order PECE method with Euler method as predictor and trapezoidal method as corrector, and with also the first timestep by Euler's method. Write a new code `pece2mid.m` instead with midpoint method as predictor $\mathbf{y}^{\text{mid}}_{n+1}$ and trapezoidal method as corrector $\mathbf{y}_{n+1}$, and with the first timestep by Heun's method $\mathbf{y}_1 = \mathbf{y}^{\text{heun}}_1$. How many iterations of the trapezoidal corrector are required to obtain $\mathbf{y}_{n+1}$ with the same leading-order truncation error as the full trapezoidal method? Explain your answer. Explain also why the function value $\mathbf{f}(t_{n+1}, \mathbf{y}^{\text{mid}}_{n+1})$ rather than $\mathbf{f}(t_{n+1}, \mathbf{y}_{n+1})$ can be saved and reused in the next timestep, without modifying the leading-order truncation error.

(c) Consider the following initial-value problem with given exact solution:

$$y' = y - y^2, \ y(0) = 1/4; \quad Y(t) = \frac{1}{1 + 3e^{-t}}.$$

Use all three of the methods midpoint, PECE2, and trapezoidal as implemented by the course code `midpoint.m` and the codes in parts (a) and (b), for numbers of steps $N_s = 30, 300, 3000$ over the interval $0 < t < 5$. For each of these three stepsizes plot together the error of the three methods and compare and discuss the performance of the three methods. Do all three appear to converge? Rate the methods in terms of efficiency, as quantified by the number of function evaluations per time step. Which methods are most accurate, under what circumstances, and why?

Hint: For the last question, consider the sign of $\frac{\partial f}{\partial y}(t, Y(t))$.

Problem 3 [20 points]. For each of the following 3-step methods

$$\begin{array}{ll} (i) & \mathbf{y}_{n+1} = \frac{1}{4}(\mathbf{y}_n + 3\mathbf{y}_{n-2}) + \frac{1}{4}h(9\dot{\mathbf{y}}_n - 2\dot{\mathbf{y}}_{n-1} + 3\dot{\mathbf{y}}_{n-2}) \\ (ii) & \mathbf{y}_{n+1} = \mathbf{y}_{n-2} + \frac{1}{4}h(9\dot{\mathbf{y}}_n + 3\dot{\mathbf{y}}_{n-2}) \\ (iii) & \mathbf{y}_{n+1} = \frac{1}{20}(27\mathbf{y}_n - 7\mathbf{y}_{n-2}) + \frac{h}{10}(6\dot{\mathbf{y}}_{n+1} - 3\dot{\mathbf{y}}_{n-2}) \end{array}$$

answer the following questions:

(a) Is the method explicit or implicit?

(b) Is the method consistent?

(c) Find the characteristic polynomial $\rho(r) = (1-h\lambda b_{-1})r^{p+1} - \sum_{j=0}^{p}(a_j + h\lambda b_j)r^{p-j}$ of the method. You do not need to find the $p+1$ roots, denoted $r_0(h\lambda), r_1(h\lambda), ..., r_p(h\lambda)$. Note for any consistent method there is a root satisfying $r_0(h\lambda) = 1 + h\lambda + O((h\lambda)^2)$.

(d) Is the method convergent?

To answer this question you may use the following fundamental result: A consistent multistep method is convergent if and only if the *root condition* is satisfied: $|r_j(0)| \leq 1$ for all $j = 0, ..., p$ and furthermore any root which satisfies $|r_j(0)| = 1$ must be simple. Note that the root condition only involves the roots $r_j(0)$ for $h\lambda = 0$, which you should be able to calculate explicitly.

(e) If the method is convergent, what is the order of convergence and the leading-order truncation error $\mathbf{T}_n(\mathbf{y})$?

(f) If the method is convergent, is it also relatively stable?

To answer (f), you might use the fact that relative stability is implied by the *strong root condition*: $|r_j(0)| < 1$ for all $j \neq 0$. If you use this condition, you must explain why it implies relative stability. If instead $|r_j(0)| = 1$ for some $j \neq 0$, then you will need to consider $r_j(h\lambda)$ for $h\lambda \neq 0$. You may do this numerically, if necessary.