

CAAM 353: Computational Numerical Analysis
Take-Home Exam - April 24, 2003
Instructor: *Dr. Kartik Krishnan*

INSTRUCTIONS

1. TIME LIMIT: 5 hours.
2. There are 4 questions in all. Read each question carefully.
3. 100 points in all.
4. Solve each problem in detail. Define variables carefully. Always explain your answer, unless otherwise stated.
5. Write clearly, including all the steps to the final solution. If I can't read it, you won't get credit.
6. Sources: Open book, notes, handouts, homeworks, solutions to homework assignments, and other articles posted on the course webpage.
7. This has to be all your own work. Do not consult anyone else. If you have any questions, my email address is kartik@caam.rice.edu, and phone number 348-2649. I'll have my regular office hours on Monday, Wednesday, and Friday between 12 - 1pm.
8. Due in my office at DH 3018 Thursday, May 1st for all graduating students, and Tuesday, May 6th for all others. No late exams will be accepted.

On my honor I have neither given nor received any aid on this examination.

1. [25 points] Suppose you need to solve the linear system $Cz = d$, where C is a complex $n \times n$ matrix, and d and z are complex n vectors, but your linear equation solver handles only real systems. Let $C = A + iB$, and $d = b + ic$, where A , B , b , and c are real and $i = \sqrt{-1}$.

- (a) [10 points] Show that the solution $z = x + iy$ is given by the $2n \times 2n$ real linear system

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ c \end{bmatrix} \quad (1)$$

Note :- $a + ib = c + id$ if and only if $a = c$, and $b = d$.

- (b) [15 points] Is this an efficient way to solve the problem?. Why?. Base your answer on the estimated flop count, and the amount of storage if you use the LU factorization procedure to solve (1)?. How does this compare with the regular LU factorization procedure with complex multiplies?.
2. [20 points] Complexity of LU factorization
- (a) [5 points] For a series of matrices A of order n , record the flops for MATLAB's LU factorization routine, that computes $A = LU$. If the *flops* is no longer valid on your computer, evaluate the execution times, using the following sequence of MATLAB commands

```
tic
[L, U] = lu(A);
toc
```

In either case to obtain reliable results, use a fairly wide range of values of n , say $n = 100$ to $n = 1000$ in increments of 100. You may obtain more accurate timings by avergaing several runs for a given matrix size.

- (b) [10 points] Using MATLAB's *polyfit* routine (a least squares routine in this case), fit a cubic polynomial to the execution times as a function of n . If you did not use the *flops* command earlier determine the basic execution rate (a flop) for your computer by timing the multiplication of two numbers (about the same magnitude as the entires in A). Use this to convert the execution times in (a) into flops.
- (c) [5 points] After converting the polynomial fit of the execution times into flops as needed, how does the dominant term compare with the theoretically expected value of $2\frac{n^3}{3}$ (counting both additions and multiplications)?. Try to explain any discrepancy.
3. [25 points] Consider the function $f : \mathcal{R}^2 \rightarrow \mathcal{R}$ defined by

$$f(x, y) = 2x^3 - 3x^2 - 6xy(x - y - 1)$$

- (a) [5 points] Determine all of the critical points of f analytically (i.e. without using a computer).
- (b) [5 points] Classify each critical point found in part (a) as a minimum, a maximum, or a saddle point. If necessary, you can use MATLAB's *eig* command to compute the eigenvalues of the Hessian matrix.
- (c) [5 points] Verify your analysis graphically, by creating a three dimensional surface plot of f over the region

$$-2 \leq x \leq 2, \quad -2 \leq y \leq 2$$

using the MATLAB file *surfplot.m* from
<http://www.caam.rice.edu/~kartik/caam353/programs>.

- (d) [10 points] Use your steepest descent routine, and Newton's method (with line search) from Homework 5 to find the minima of both f and $-f$. Experiment with various starting points to see how well the routine gets around other types of critical points to find the minima and maxima respectively.
4. [30 points] Solve the two-point BVP

$$u'' = 10u^3 + 3u + t^2, \quad 0 < t < 1$$

with boundary conditions

$$u(0) = 0, \quad u(1) = 1$$

by the following *finite difference* method. Divide the interval $0 \leq t \leq 1$ into $n+1$ equal subintervals,

$$0 = t_0 < t_1 < \dots < t_n < t_{n+1} = 1$$

with each subinterval of length $h = \frac{1}{n+1}$. Let $y_i = u(t_i)$, $i = 1, \dots, n$, represent the approximate solution values at the n interior points.

- (a) [5 points] Obtain a system of n algebraic equations for the y_i by replacing the second derivative in the differential equation by the finite difference approximation

$$y_i''(t) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}, \quad i = 1, \dots, n$$

- (b) [15 points] Use Newton's method to solve the resulting system of nonlinear equations, for $n = 1$, and 3. In the first case you use *newt1d.m*, and in the second *newtnd.m* from the course webpage at
<http://www.caam.rice.edu/~kartik/caam353/programs>. A reasonable starting guess for the nonlinear solver is a straight line between the boundary values.
- (c) [10 points] Compare the results with MATLAB's boundary value ODE solver *bvp4c*. You will have to write this second order ODE as a system of two first order ODE's first. To see how to use *bvp4c* on a sample example check out
<http://www.mathworks.com/access/helpdesk/help/techdoc/ref/bvp4c.shtml>. You can prescribe the initial conditions in the same way as in this example. Plot the resulting solutions $y = u(t)$ against t in each of the three cases, i.e $n = 1$, $n = 3$, and MATLAB.