Scanned

AMSC/CMSC 460 H. M. Glaz
FINAL EXAM
Friday, December 17, 2004 1:30 – 3:30 PM

Instructions: YOU MUST start each problem on a new page. Make sure that you **show your work, including intermediate results.**

'Matlab code' means something fairly close to working code in the easier situations, but syntax rules, etc. are not important. Concentrate on making it clear that you know how to implement the algorithm in code, given enough time. If need be, use some pseudocode mixed in to make your point.

Important: It is not likely that anyone will do all the problems. There are (25 + 20 + 40 + 50 + 75 + 75 +75) 360 possible points, plus two (2) Extra Credit parts. You should assume that I will set a maximum (i.e., perfect) score of 250 points, but that the grading will be fairly rigorous (I am keeping open the possibility of moving this +/- 25 points). You may try all problems if you wish, but I advise otherwise.

(new) Honor Pledge: I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

You MUST write this out in full and sign it on the 1st page.

(1) (25 pts) Suppose that (i) you sample a function $y(t)$ at distinct times resulting in the data (t_k, y_k), $k = 1, \ldots, 8$; (ii) that you are told to model the function $y(t)$ by a cubic polynomial derived from a least squares fit to the data; and (iii) the coefficients $\mathbf{a}$ of the polynomial are to be calculated via the Matlab expression $\cancel{\mathbf{x}}$ a $= \mathbf{A}\backslash\mathbf{b}$.

(a) (23 pts) Write down the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ that make this work.

(b) (2 pts) How does Matlab obtain the answer?

(2) (20 pts) Let

$$I = \int_0^4 exp(x)dx .$$

Obtain approximate solutions for the value of I by using the compound trapezoid rule with

(a) (10 pts) 2 panels (equally spaced) ,

(b) (10 pts) 4 panels (equally spaced) .

Note: it is not necessary to evaluate exponentials or actually perform the summations. Put away your calculators for this problem.

(3) (40 pts) Let

$$s(x) = \begin{cases} a_1(x-4)^2 + a_2(x-3)^3 + a_3(x-2)^4 , & x \leq 3 \\ b_1(x-4)^2 + b_2(x-4)^3 + b_3(x-4)^4 , & 3 \leq x \leq 5 \\ c_1(x-4)^2 + c_2(x-5)^3 + c_3(x-5)^4 , & 5 \leq x . \end{cases}$$

(a) (20 pts) Determine what constraints a_i, b_i, c_i, $i = 1, 2, 3$, must satisfy for $s(x)$ to be a cubic spline. Your answer should consist of equations which define or relate these 9 parameters.

(b) (20 pts) Determine the values of the parameters so that the cubic spline interpolates the points $(x, y) = (3, 5)$, $(0, -1)$, $(6, 21)$.

(*Hint: If you know what 'cubic' means, then the determination of* a_3, b_3, c_3 *should take just seconds – nothing fancy here.*)

(4) (50 (+EC) pts) Consider the problem of finding an $x \in \mathcal{R}$ such that

$$\cos(x) = e^x - 1 .$$

(a) (50 pts) Write a Matlab program which would find an approximate solution using Newton's method. Using an error tolerance, try to insure that the error in the computed solution is no greater than 10^{-4}.

(b) (Extra Credit) Write a new Matlab program which is 'highly likely' to find several zeros (moderate EC). If you can write your code to get 'all possible (double precision) solutions' – more EC. You *must* explain your work.

(5) (75 pts) Consider using Newton's method to find a root of the equation $g(y) = 0$. Define the function $f(x) = \lim_{n\to\infty} y_n$ where $y_0, y_1, \ldots$ are the Newton iterates for the initial guess $y_0 = x$.

(a) (42 pts) Write a Matlab m-file which computes an approximation to

$$I = \int_0^1 f(x)dx$$

with an error no greater than 10^{-3}. (For full credit, the name of the function – or function handle – should be an input argument, along with the tolerance(s).) Here, you will need an m-file to compute $g(y)$ for general inputs y, but the body will be empty for part (a). You may use, e.g., the compound trapezoid rule for the quadrature (simplest).

(b) (15 pts) Now, define

$$g(y) = (y - \frac{1}{3})^2 - 1 .$$

Fill in the m-file for $g(y)$ and verify that your code will do what is required.

(c) (18 pts) What is the exact value of I for the special case defined in part (b)? Explain (briefly) your reasoning. *Note:* 'Briefly' does *not* mean zero or nothing; one or two sentences is enough.

Hint: Part (a) focuses on algorithm/program logic; the rest requires specific analysis of the special case. However, it's likely that part (a) will be easier if you work some on (b) – (c) first.

(6) (75 pts) The linear system $\mathcal{A}\mathbf{x} = \mathbf{b}$ is to be solved –

$$\mathcal{A} = \begin{pmatrix} \mathbf{U} & \mathbf{v} \\ \mathbf{u}^T & 0 \end{pmatrix}$$

where $\mathbf{U} \in \mathcal{R}^{m\times m}$ is assumed to be nonsingular and upper–triangular, and $\mathbf{u}, \mathbf{v} \in \mathcal{R}^m$ are column vectors.

(a) (2 pts) If $\mathcal{A} \in \mathcal{R}^{N\times N}$, what is N in terms of m.

(b) (23 pts) Specify the triangular factorization of $\mathcal{A}$, that is find $\mathbf{w}, z$ such that

$$\mathcal{A} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{w}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{U} & \mathbf{v} \\ \mathbf{0}^T & z \end{pmatrix}$$

(c) (20 pts) Show that $\mathcal{A}$ is nonsingular if and only if $\mathbf{u}^T\mathbf{U}^{-1}\mathbf{v} \neq 0$.

(d) (30 pts) Formulate an *efficient* algorithm to solve $\mathcal{A}\mathbf{x} = \mathbf{b}$. Specifically, your algorithm should require an operation count of only order n^2. Demonstrate this.

Hint: Sketch the matrix $\mathcal{A}$ – focusing on where the (large number of) zeroes are – and think carefully about how Gaussian elimination – **without** pivoting (don't even consider pivoting, or any roundoff issues) proceeds.

Notes: No Matlab code – just a sequence of steps to be performed; also, indicate how explicit matrix inversions will be avoided. Make sure that you state where the problem hypotheses are used – the n^2 count depends *directly* on the fact that $\mathbf{U}$ is triangular.

(7) (75 pts) Let $f(x,y) = x^2 + y^2 - \frac{1}{4}$, and define $C = f^{-1}(0)$; that is, the curve C is the *zero set* of $f(x,y)$.

(a) (25 pts) Let Ω = [-1,1] x [-1,1]. Write a Matlab code which does the following: (1) given an integer $N > 0$, a set of N angles $\theta = 0, 2\pi/N, 2*(2\pi/N), \ldots$ is formed and (2) the following approach is implemented with the objective of obtaining N points on C : using the angles above, consider the corresponding collection of rays from the origin to the boundary of Ω. Along each ray, use the bisection method to find a point on C. Finally, (3) plot the points.

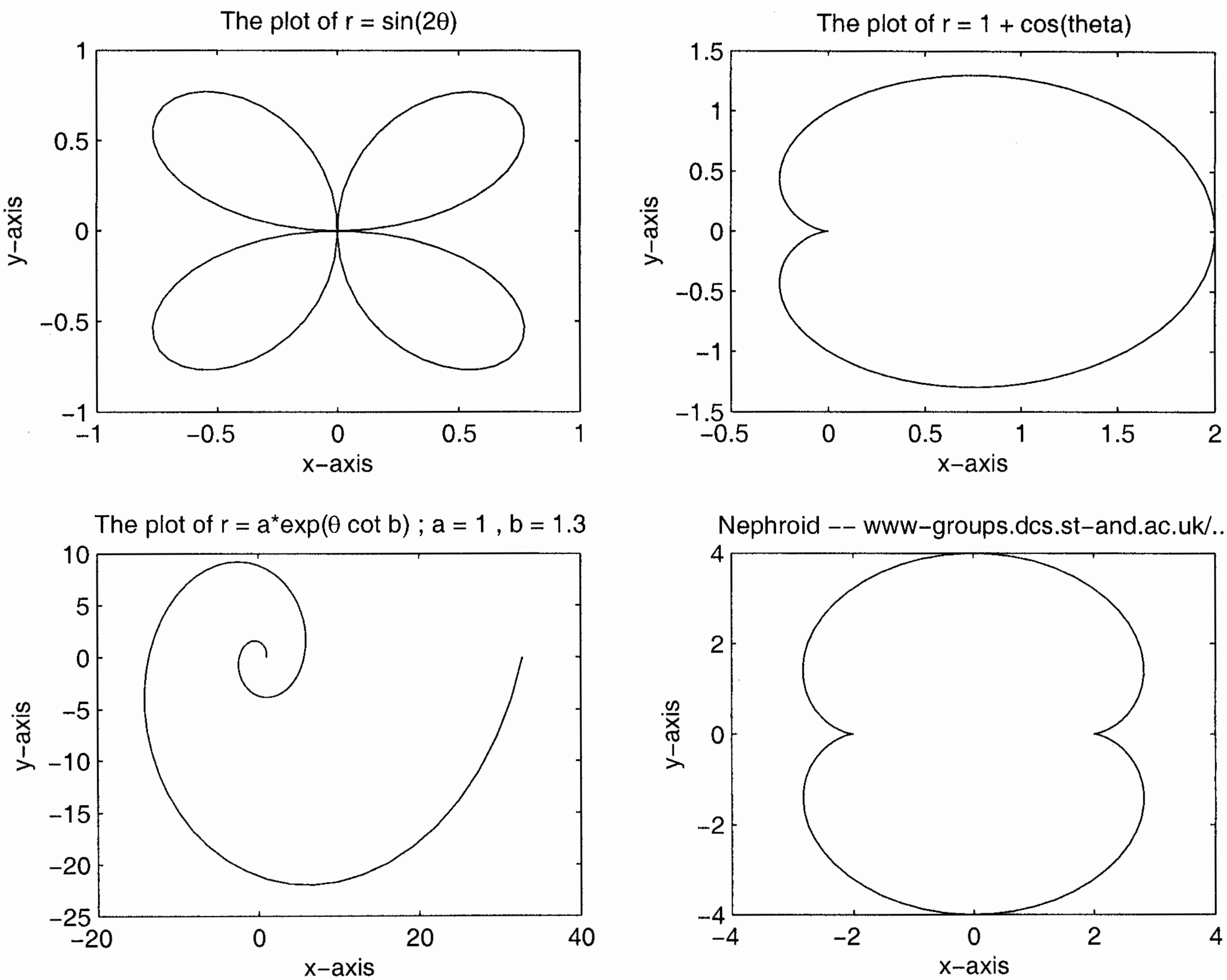
(b) (15 pts) Demonstrate that the algorithm works (the main point is that each use of the bisection method leads to a useful result, if you set it up properly).

Note: Of course, the exact solution for C is trivial to obtain. You are **not** to use the detailed solution – instead, show that the boundary of Ω is entirely outside C (and the origin is inside C) and try to write your algorithm/demonstration for more general $f(x,y)$.

(c) (10 pts) What restrictions need to be placed on $f(x,y)$ to guarantee that an approximation to C is obtained? (See part (d) below; if you work part (d), then (c) – (d) scoring is combined.)

(d) (25 pts; EC possible) Here, you are to consider the practicality of generalizing the algorithm to more complicated functions $f(x,y)$. A few examples are attached (although polar notaion is used in the titles, it is straightforward to convert the equations to rectangular coordinates – and each curve is the *zero set* for some $f(x,y)$.) *Redefine* the square Ω so that it contains the curve C in each case (equivalently, the curves can all be scaled to fit in the given Ω and retain exactly the same properties as in the figures).

Identify as many implementation issues as possible, and sketch possible generalized code. Be sure to think about what to do if bisection fails – don't let the program crash or just stop; more difficult – how do you know when a ray has multiple intersections and what to do about it. What about cusp points? You can also construct examples which illustrate new problems.


The plot of r = sin(2θ)
y-axis
x-axis
The plot of r = 1 + cos(theta)
y-axis
x-axis
The plot of r = a*exp(θ cot b) ; a = 1 , b = 1.3
y-axis
x-axis
Nephroid -- www-groups.dcs.st-and.ac.uk/...
y-axis
x-axis