CS 322: Prelim 2 Solutions

Thursday, April 8, 2004

95-100	xxx
90-94	xxxxxxxxx
85-89	xxxxxxxxxxx
80-84	xxxxxxxxxx
75-79	xxxxxxxxxxxx
70-74	xxxxxxxxxxx
65-69	xxxxxxxxxx
60-64	xxxxx
55-59	xxx
50-54	xxxx
< 50	xx

Median ≈ 75.

Letter grade guidelines : A $\in [85, 100]$, B $\in [70, 79]$, C $\in [50, 64]$

Problem 1 (20 points)

(a) Suppose S is a cubic spline with breakpoints $a = x_1 < \cdots < x_n = b$. Why is the maximum value of $S''(x)$ on $[a, b]$ at a breakpoint?

S"(x) is piecewise linear. The max of the ith local linear function is either at x_i or x_{i+1}. So the overall max must be at a breakpoint.

Around 4 points if you if you revealed a knowledge about cubic splines, but gave a spurious argument.

(b) Simpson's rule is exact for cubic polynomials. Does it follow that the composite Simpson rule is exact for cubic splines? Explain why or why not.

It will be exact as long as no subinterval in the composite rule has a spline breakpoint strictly in between its endpoints. Stated another way, the composite rule will be exact if every subinterval associated with the rule must has breakpoints for endpoints.

5 points if you said "yes" and just assumed that the composite simpson breakpoints = spline's breakpoints.

Problem 2 (20 points)

(a) Suppose `P` is a nonsingular n-by-n matrix and `v` is a column n-vector. If the following script is executed

```
v0 = v;
v = P\v; v = P\v ; v = P\v; v = P*v; v = P*v ; v = P*v;
err = norm(v - v0,1)/norm(v0,1);
```

then the value assigned to `err` is $O(1)$. Offer a brief explanation.

If P has condition 10^d, then d significant digits are lost every time we solve a linear system involving P. With 3 linear system solves, we could have lost $3d$ digits by the time look at the last v. Apparently, $10^{4d} \approx 10^{17}$. You have to talk about the condition number (about 3 pts), the heuristic that deals with loss of significant digits (about 44 points), and the fact that we are repeatedly solving a system with P as the matrix (about 3 points).

(b) Consider the following function:

```
  function d = InsideDist(x,y,theta)
% x and y are column n vectors and theta satisfies 0 <= theta < 2pi.
% Let R be the ray that originates at (0,0) and makes an angle theta
% with the positive x-axis.
% Let P be the polygon defined by connecting
% (x(1),y(1)),...,(x(n),y(n)),(x(1),y(1)) in order.
% Assume that (0,0) is outside P, only adjacent polygon edges intersect,
% the ray does not go through any polygon vertex and is not parallel
% to any polygon edge.
% d is the length of that part of R that is inside P.
```

For a particular choice of valid inputs `x`, `y`, and `theta`, it is known that the ray does not intersect the polygon. However, when we invoke `InsideDist` with this data and carry out the computations using floating point arithmetic, we observe that the value returned in `d` is $O(1)$. Explain how this can happen using a few sentences and a picture.

Suppose the ray passes very close to a vertex associated with two nearly parallel edges. Because of ill-conditioning, when we solve the two systems for intersection points, we may conclude that the ray intersects each of these edges. And this could result in a "serious", $O(1)$ inside distance.

Or, the ray is very close to a vertex *and* one of the edges associated with that vertex is nearly parallel to the ray.

The key idea is that when you solve a system associated with nearly [parallel lines, it is very ill-conditioned. Thus, the computed t_1 and t_2 (in the language of the problem) may indicate intersection with the edge.

Note that $O(1)$ means a number that is roughly order unity. That means a condition number of around $1/eps$.

Problem 3 (20 points)

Recall that in MATLAB we can solve the least square problem of minimizing $\| Ax - b \|_2$ simply by setting `x = A\b`. Complete the following function so that it performs as specified:

```
  function u = Close(f,g,h)
% f, g and h are column m-vectors. u is the closest vector to h that is
% a linear combination of f and g. Assume that distance is measured in
% the 2-norm.


  x = [f g]\h;    % 13 points
  u = [f g]*x;    % 7 points
```

Problem 4 (20 points)

Complete the following MATLAB function so that it performs as specified.

```
  function X = Solve(A,C,B)
% A is nonsingular and n-by-n
% C is n-by-n
% B is n-by-p
% X is n-by-p such that
%
%                 B(:,k) = A*X(:,k)                 if k =  1
%                 B(:,k) = A*X(:,k) + C*X(:,k-1)    if k >= 2\
```

Make effective use of `[L,U,P] = lu(A)` which computes the factorization $PA = LU$ where P is a permutation matrix, L is lower triangular, and U is upper triangular. You may use the "\" operator to solve triangular systems.

```
[L,U,P] = lu(A)
[n,p] = size(B);
X = zeros(n,p);
X(:,1) = U\(L\(P*B(:,1)));
for k=2:p
   X(:,k) = U\(P*(L\(B(:,k) - C*X(:,k-1))));
end
```

5 points for using P correctly
8 points for just doing LU once and using l and U correctly
7 points for the correct recipe for $X(:,k)$ in terms of $X(:,k-1)$

Problem 5 (20 points)

We wish to compute

$$I = \int_a^b \int_c^d f(x,y)dxdy$$

by making effective use of `QUAD`. `QUAD(fname,alpha,beta,[],z)` returns an approximation to the integral from `alpha` to `beta` of the function named by `fname`. The argument `z` is a parameter. Thus, if

```
function y = F(x,tau)
y = sin(tau*x)
```

then `Q = QUAD('F',0,1,[],2)` assigns to `Q` an approximation of the integral $\int_0^1 \sin(2x)dx$. If no parameters are involved, then `Q = QUAD(fname,alpha,beta)` returns the approximation. (Default tolerances prevail in this problem.)

(a) (15 points) How would you use `QUAD` to estimate the integral I above? Assume that `a`, `b`, `c` , and `d` are available along with a function `MyF(x,y)` that returns the value of f at a specified point. Specify completely any integrand functions that you must write to solve the problem.

```
function tau = F1(y)
tau =  QUAD('MyF',c,d,[],y)     % 12 points

I = QUAD('F1',a,b)    % 3 points
```

-10 points for doing outer integral as a simple sum of inner integral evaluations, i.e., for not making effective use of `QUAD`.

(b) (5 points) Assume that $f(-x,y) = -f(x,y)$ and that $f(x,-y) = -f(x,y)$ in part (a). How could you make your solution in part `(a)` more efficient given that $c < 0 < d$, $-c \square d$, $a < 0 < b$, and $-a \square b$.

```
function tau = F1(x,y)
tau =  QUAD('MyF',d+c,d,[],y)

I = QUAD('F1',b+a,b)
```

-4 points if you acted as if $f(-x,y) = f(x,y)$ and $f(x,-y) = f(x,y)$.