CS 4220: Final Exam (Solutions)

Spring 2010

Median = 74

Problem 1. (15 points)

Consider the following matrix

$$C = \begin{bmatrix} A & uu^T \\ vv^T & -A^T \end{bmatrix} \qquad A \in \mathbb{R}^{n\times n},\ u \in \mathbb{R}^n,\ v \in \mathbb{R}^n.$$

How would you efficiently compute $B = C^2$? Answer the question by completing the following MATLAB function:

```
  function  B = HamSqr(A,u,v)
% A is an n-by-n matrix and u and v are column n-vectors.
% B is the square of the matrix C = [ A  u*u' ; v*v'  -A']
```

Solution

First, work out the product...

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A & uu^T \\ vv^T & -A^T \end{bmatrix}\begin{bmatrix} A & uu^T \\ vv^T & -A^T \end{bmatrix} = \begin{bmatrix} A^2 + uu^Tvv^T & Auu^T - uu^TA^T \\ vv^TA - A^Tvv^T & vv^Tuu^T + A^{2T} \end{bmatrix}$$

Then make some observations and put parentheses in the right places....

$$C_{11} = A^2 + (u^Tv)uv^T = C_{22}^T$$

$$C_{12} = (Au)u^T - u * (Au)^T \qquad C_{21} = v(A^Tv)^T - (A^Tv)v^T$$

Note that $(Av)v^T$ is $O(n^2)$ while $A * (vv^T)$ is $O(n^3)$. So...

```
y = A*u; z = A'*v; C11 = A*A + (u'*v)*u*v';
C = [C11 y*u'-u*y'; v*z'-z*v' C11'];
```

5 points for the diagonal blocks, 5 points for the corner blocks. -8 points for correct but very inefficient, e.g., things like

```
T1 = u*u'; T2 = v*v'; C11 = A*A + T1*T2; C12 = A*T1 - T1*A'
```

Problem 2. (10 points)

Improve this implementation and explain your changes.

```
function xStar = Bisection(f,a,b)
% f is a continuous function that changes sign on the interval [a,b].
% Assume a > 0.
% xStar is a nearest floating point number to a zero of f.

while b-a > eps
   mid = (b+a)/2;
   if f(a)*f(m) < 0
      b = m;
   else
      a = m;
   end
end
xStar = (b+a)/2;
```

```
function xStar = Bisection(f,a,b)
% f is a continuous function that changes sign on the interval [a,b].
% Assume a > 0.
% xStar is a nearest floating point number to a zero of f.

fa = f(a); fb = f(b);
while b-a > eps*abs(a)                % relative error        5 points
   m = (b+a)/2; fm = f(m);            % Efficient f-eval strategy   5 points
   if sign(fa) ~= sign(fm)            % avoids underflow/overflow
      b = m; fb = fm;
   else
      a = m; fa = fm;
   end
end
xStar = (b+a)/2;
```

Problem 3. (10 points)

If λ is a distinct, real eigenvalue of $A \in \mathbb{R}^{n\times n}$, then its condition number is given by

$$\text{cond}(\lambda) \;=\; \frac{1}{|x^T y|}$$

where x and y are unit 2-norm vectors that satisfy $Ax = \lambda x$ and $A^T y = \lambda y$. Complete the following function so that it performs as specified:

```
 function kappa = EigCond(T,k)
% T is a real n-by-n upper triangular matrix with distinct diagonal entries.
% k satisfies 1 < k < n.
% kappa is the condition of the eigenvalue T(k,k).
```

You may use \. (You might want to display what is going on if $n = 5$ and $k = 3$.) How could `EigCond` be used to estimate the accuracy of an eigenvalue that is computed via the MATLAB function `schur`?

If

$$T \;=\; \begin{bmatrix} T_{11} & v & T_{13} \\ 0 & t_{kk} & w^T \\ 0 & 0 & T_{33} \end{bmatrix}$$

then $Tx = t_{kk}x$ leads to

$$\begin{bmatrix} T_{11} - t_{kk}I_{k-1} & v & T_{13} \\ 0 & 0 & w^T \\ 0 & 0 & T_{33} - t_{kk}I_{n-k} \end{bmatrix} \begin{bmatrix} x(1{:}k-1) \\ -1 \\ 0 \end{bmatrix} = 0$$

while $T^T y = t_{kk}y$ leads to

$$\begin{bmatrix} T_{11}^T - t_{kk}I_{k-1} & 0 & 0 \\ v^T & 0 & 0 \\ T_{13}^T & w & T_{33}^T - t_{kk}I_{n-k} \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ y(k+1{:}n) \end{bmatrix} = 0$$

```
x = [(T(1:k-1,1:k-1)-T(k,k)*eye(k-1,k-1))\T(1:k-1,k) ; -1; zeros(n-k,1)];   %6 points
x = x/norm(x);

y = [zeros(k-1,1);  -1 ; (T(k+1:n,k+1:n) - T(k,k)*eye(n-k,n-k))'\T(1:k-1,k)'];  %6 points
y = y/norm(y);

kappa = 1/abs(x'*y);
```

3 points: Error in schur-computed eigenvalue $\approx$ cond(λ) $\cdot$ eps

Problem 4. (15 points) One way of fitting the data $(t_1, f_1), \ldots, (t_m, f_m)$ with a polynomial $p(x) = a_1 + a_2x + \cdots + a_nx^{n-1}$ is to minimize

$$\phi(a) = \sum_{i=1}^{m}(p(t_i) - f_i)^2 + \mu^2 \sum_{i=1}^{m} p''(t_i)^2$$

where μ is a scalar. The larger the value of μ the less nonlinear will be the optimum fitting polynomial. Complete the following MATLAB function so that it performs as specified.

```
  function a = LScubic(t,f,mu)
% t and f are column m-vectors, m>=4, and mu > 0.
% a is a column 4-vector with the property that if
%
%            p(x) = a(1) + a(2)*x + a(3)*x^2 + a(4)*x^3
%
% then
%      phi(a) = (p(t(1)) - f(1))^2 + ... + (p(t(m)) - f(m))^2 + ...
%                                 mu^2(p''(t(1))^2 + ... + p''(t(m))^2))
%
% is minimized.
```

You may use \ Hint. Set up a $2m$-by-4 linear least squares problem. You might want to display the $m = 5$ case to see what is going on.

Solution

For $m = 5$ want to minimize the 2-norm of the vector

$$r = \begin{bmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 1 & t_2 & t_2^2 & t_2^3 \\ 1 & t_3 & t_3^2 & t_3^3 \\ 1 & t_4 & t_4^2 & t_4^3 \\ 1 & t_5 & t_5^2 & t_5^3 \\ 0 & 0 & 2\mu & 6t_1\mu \\ 0 & 0 & 2\mu & 6t_2\mu \\ 0 & 0 & 2\mu & 6t_3\mu \\ 0 & 0 & 2\mu & 6t_4\mu \\ 0 & 0 & 2\mu & 6t_5\mu \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} - \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

so in general we do this:

```
m = length(f);
C = [ones(m,1) t t.^2 t.^3 ; zeros(m,1) zeros(m,1) 2*mu*ones(m,1) 6*mu*t];
g = [f ; zeros(m,1)];
a = C\g;
```

5points for top half of C. 5 points for bottom part of C. 5 points for rhs vector.

Problem 5. (15 points) Suppose (x_1, y_1) and (x_2, y_2) are distinct and that $g(x, y)$ is a continuous function. We want to use `fzero` to find t_* such that if

$$\begin{aligned} x_* &= x_1 + t_*(x_2 - x_1) \\ y_* &= y_1 + t_*(y_2 - y_1) \end{aligned}$$

then

$$g(x_*, y_*) = (g(x_1, y_1) + g(x_2, y_2))/2$$

Complete the following function so that it accomplishes this task.

```
 function [xStar,yStar] = AvePoint(g,x1,y1,x2,y2)
% g is a continuous function of two variables defined everywhere.
% (xStar,yStar) is on the line segment that connects (x1,y1) and (x2,y2)
% with the property that g(xStar,yStar) = (g(x1,y1)+g(x2,y2))/2.



% Compute the average once and for all...              % 3 points
ave = (g(x1,y1) + g(x2,y2))/2

tStar = fzero(@(t) MyF(t,g,ave,x1,y1,x2,y2),[0 1])      % 5 points for initial guess

xStar = x1+tstar*(x2-x1)
yStar = y1+tstar*(y2-y1)                                % 7 points for the rest

function alfa = MyF(t,g,ave,x1,y1,x2,y2)
alfa = g(x1+t*(x2-x1),y1+t*(y2-y1))-ave;
```

Problem 6. (15 points)

(a) Why is the Quasi-Newton framework appealing when solving systems of nonlinear equations?

No Jacobian eval.s. No $O(n^3)$ linear equation solve overhead becuase of updating QR

(b) For what problem is the power method of interest?

Computing the dominant eigenvalue and its eigenvector.

(c) Briefly explain how the singular value and QR-with-column-pivoting decompositions can be used to estimate matrix rank in the presence of round off error.

SVD: The largest r so $\sigma_r \geq tol$.

QR with colpivoting: largest r so $|r_{rr}| \geq tol$.

In both cases tol should be something like $eps * \| A \|_1$

Problem 7. (15 points)

Complete the following function making effective use of the MATLAB function `lu`. You may use \ to solve triangular systems.

```
 function X = MatSolve(A,B)
% A is n-by-n and nonsingular.
% B is n-by-n.
% X is n-by-n and satisfies A*X*A = B



[L,U,P] = lu(A);
% Solve AY = B i.e. L(UX)=PB
Y = U\(L\(P*B));                                  % 8 points

% Solve XA = Y, i.e., A'*X' = Y'
%              i.e., (P'LU)'*X' = Y'
%              i.e., (U'(L'(PX'))) = Y'
X = (P'*(L'\(U'\Y')))';                           % 7 points
                                                %-5 points if you forget about P
```