CS 621: Final Exam Solutions

Tuesday, December 14, 2004

Final exam scores:

90-100	xxx
80-89	xxxxxx
70-79	xxx
60-69	xx
50-59	xxxx
40-49	xxxx
< 40	xx

Approximate centroids for various final exam grades:

A+	95
A	85
A-	75
B+	65
B	55
B-	50
C	40

1. (10 points) Assume the availability of the following function:

```
  function [c,s] = Rotate(x)
% x is a column 2-vector. c and s satisfy c^2 + s^2 = 1 with the property
% that if y = [c s ; -s  c]'*x, then y(2) = 0.
```

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

```
  function [c,s] = GeneralRotate(x,y)
% x and y are nonzero column 2-vectors. c and s satisfy c^2 + s^2 = 1
% with the property that [c s ; -s  c]*x is a scalar multiple of y.
```

You must make effective use of `Rotate`. For full credit, do not make use of any built-in MATLAB functions.

Solution:

```
[c1,s1] = Rotate(x)
[c2,s2] = Rotate(y)

%  It follows that [c1  s1; -s1  c1]'*x  is a multiple of [c2  s2;  -s2  c2]'*y.
%  I.e., ([c1  s1; -s1  s1]*[c2  s2; -s2 c2])'*x  is a multiple of y.
%  I.e., [ c  s ; -s  c]'*x  is a multiple of y  where

c =  c1*c2 + s1*s2;
s = -c1*s2 + s1*c2;
```

Another derivation:

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \alpha \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

which means that

$$\frac{cx_1 - sx_2}{y_1} = = \frac{sx_1 + cx_2}{y_2}.$$

Thus, $s(x_1y_1 + x_2y_2) + c(y_1x_2 - x_1y_2) = 0$. Thus,

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} x_1y_1 + x_2y_2 \\ y_1x_2 - x_1y_2 \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

Thus, `[c,s] = Rotate(x(1)*y(1)+x(2)*y(2),y(1)*x(2)-x(1)*y(2))`.

3 or 4 points for codes that don't work if there is some sensible sine-cosine manipulation.

2. (15 points) Complete the following function so that it performs as specified.

```
  function x = LowHessSolve(H,b)
% H is an n-by-n nonsingular lower Hessenberg matrix, i.e.,
% H(i,j) = 0 whenever j > i+1.
% b is a column n-vector
% x is a column n-vector that solves Hx = b.
```

To receive full credit, you must make effective use of of the MATLAB `lu` function:

```
  function [L,U,P] = LU(X)
% X is an n-by-n matrix
% Returns unit lower triangular matrix L, upper
% triangular matrix U, and permutation matrix P so that
% P*X = L*U.
```

You may use the \ operator to solve triangular systems. In assessing the efficiency of your solution, you may ignore flops associated with operations of the form P-times-vector or P^T-times-vector where P is a permutation matrix.

Solution:

```
[L,U,P] = lu(H');
% PH' = LU means H*P' = U'*L'  so H = U'*L'*P
% To solve U'*(L'*(Px)) = b we
z = U'\b;   %U'z = b
y = L'\z;   %U'*L'y = b
x = P'*y    % U'*L'*P*x = Hx = b
```

Note that `[L,U,P] = lu(H)` can be $O(n^3)$ because of pivoting. Indeed, suppose in the very firat step we have to swap rows 1 and n:

$$\tilde{H} = P_1 H = \begin{bmatrix} x & x & x & x & x \\ x & x & x & 0 & 0 \\ x & x & x & x & 0 \\ x & x & x & x & x \\ x & x & 0 & 0 & 0 \end{bmatrix}$$

The zeroing of $\tilde{H}(2{:}n, 1)$ will result in a full $\tilde{H}(2{:}n, 2{:}n)$.

7 points if you correctly worked with `[L,U,P] = lu(H)` and another 2 points if you said U is upper bidiagonal because it shows some thinking about the Hessenberg structure. (U actually isn't upper triangular if you do pivoting.)

3. (20 points) Complete the following MATLAB function so that it performs as specified.

```
  function [Q,T] = FinishRealSchur(A,Q1,T11)
% A is a real n-by-n
% Q1 is n-by-r with Q1'*Q1 = eye(r,r).
% T11 is r-by-r and upper triangular.
% Assume that A*Q1 = Q1*T11.
%
% Q is real orthogonal and T = Q'*A*Q is upper quasi-triangular.
% In addition, Q(:,1:r) = Q1, and T(1:r,1:r) = T11.
```

The only MATLAB factorization functions that you can use in this problem are

```
QR     Orthogonal-triangular decomposition.
   [Q,R] = QR(A) produces an upper triangular matrix R of the same
   dimension as A and a unitary matrix Q so that A = Q*R.

SCHUR  Schur decomposition.
   [U,T] = SCHUR(X) produces a quasitriangular Schur matrix T and
   a unitary matrix U so that X = U*T*U' and U'*U = EYE(SIZE(U)).
   X must be square.
```

Solution:

```
[n,r]= = size(Q1);
[Q,R] = qr(Q1);    % Note, that Q1 = Q*R implies R = diag(plus/minus ones)
Q2 = Q(:,r+1:n);   % Q2'*Q1 = 0
Q(:,1:r) = Q1;     % Takes care of the pluis/minus problem

% Since Q = [Q1 Q2], it follows that
%
%      Q'*A*Q  = [Q1'*A*Q1  Q1'*A*Q2 ; Q2'*A*Q2  Q2'*A*Q2] = [T11  S12 ; S21 S22] = T0
%
% and that S21 = Q2'*A*Q1 = Q2'*Q1*T11 = 0. Thus, we have to compute the real Schur
% form of  T0 = [T11  S12  ; 0 S22].

  [U,T22] = schur(Q2'*A*Q2);
  Q(:,r+1:n) = Q(:,r+1:n)*U;
  T = [T11  Q1'*A*Q(:,r+1:n) ; zeros(n-r,r) T22];
```

Up to 10 points for 2-by-2 block portrayals of the problem that show something about what is going on. Note that computing the QR factorization of A doesn't help. And you cannot just compute the Schur decomp of $A(r+1{:}n, r+1{:}n)$.

4. (15 points) Complete the following function so that it performs as specified.

```
  function X = InvPageRanks(G,b,rho)
% b is a column N-vector.
% rho is a column m-vector with 0 < rho(1) < ... < rho(m) = 1
% G is an N-by-N matrix with the property that if p satisfies
% 0 < p <= 1 then A(p) = p*G + (1-p)*e*e'/N is nonsingular where e = ones(N,1).
%
% X is an N-by-m matrix with the property that
%
%                    A(rho(j))*X(:,j) = b      j=1:m
```

You may use the \ operator to solve all linear syystems. Assume that operations of the fiorm `G\z` where $z \in \mathbb{R}^N$ require $O(N^2)$ flops. In the derivation of your method, you will want to exploit the Sherman-Morrison formula

$$(I_N + uv^T)^{-1} \;=\; I_N - \alpha uv^T \qquad \alpha = 1/(1+v^Tu) \qquad u,v \in \mathbb{R}^N$$

Solution:

Apply the inverse of pG to both sides of $(pG + (1-p)ee^T/N)y = b$ and observe that

$$\left(I \;+\; \frac{1-p}{pN} we^T\right) y = \frac{1}{p} z$$

where $Gz = b$ and $Gw = e$. Using the Sherman-Morrison formula with $u = (1-p)w/(pN)$ and $v = e$, we see that

$$\begin{aligned}
y &= \frac{1}{p}\left(I + uv^T\right)^{-1} z \\
&= \frac{1}{p}\left(I - \alpha uv^T\right) z \qquad \alpha = 1/(1+u^Tv) = 1/(1+(1-p)e^Tw/(pN)) \\
&= \frac{1}{p}\left(z - \alpha(v^Tz)u\right) \\
&= \frac{1}{p}\left(z - \frac{(1-p)e^Tz}{pN+(1-p)e^Tw} w\right)
\end{aligned}$$

so

```
N = length(b); m = length(rho); e = ones(N,1);
z = G\b; gamma = e'*z;
w = G\e; beta =  e'*w;
X = zeros(N,m);
for j=1:m
   p = rho(j);
   X(:,j) = (z - ((1-p)*gamma/(p*N+(1-p)*beta))*w)/p;
end
```

-5 if you do not get the linear system solves outside the loop. Up to 7 points for no code but sensible manipulations with the Sherman Morrison idea.

5. (10 points) If $A \in \mathbb{R}^{n\times n}$ is symmetric, tridiagonal, and positive definite then it has an LDL^T factorization, e.g.,

$$A = \begin{bmatrix} a_1 & c_2 & 0 & 0 \\ c_2 & a_2 & c_3 & 0 \\ 0 & c_3 & a_3 & c_4 \\ 0 & 0 & c_4 & a_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ f_2 & 1 & 0 & 0 \\ 0 & f_3 & 1 & 0 \\ 0 & 0 & f_4 & 1 \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 & 0 \\ 0 & d_2 & 0 & 0 \\ 0 & 0 & d_3 & 0 \\ 0 & 0 & 0 & d_4 \end{bmatrix} \begin{bmatrix} 1 & f_2 & 0 & 0 \\ 0 & 1 & f_3 & 0 \\ 0 & 0 & 1 & f_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Comparing coefficients we find

$$\begin{aligned} a_1 &= d_1 \\ a_k &= d_k + d_{k-1} f_k^2 \qquad k = 2{:}n \\ c_k &= f_k d_{k-1} \qquad k = 2{:}n \end{aligned}$$

(a) Complete the following function so that it performs as specified

```
function  [d,f] = LDLT(a,c)
% a and c are column n-vectors with the property that if A is the symmetric tridiagonal
% matrix with a = diag(A) and c(2:n) = diag(A,-1), then A is positive definite.
% d and f are column n-vectors so that if D = diag(d) and L is unit lower bidiagonal with
% f(2:n) = diag(A,-1), then A = L*D*L'.
```

Solution:

```
n = length(a)
d(1) = a(1);
for k=2:n
   f(k) = c(k)/d(k-1);
   d(k) = a(k) - c(k)*f(k);
end
```

-1 if you have `d(k) = a(k) - d(k-1)*f(k)*f(k)` as this is significant in an $O(n)$ setting. **(b)** Suppose $e_1 = I_n(:,1)$. Assume that $A + \alpha e_1 e_1^T$ is positive definite and that $A + \alpha e_1 e_1^T = L(\alpha)D(\alpha)L(\alpha)^T$ is its LDL^T factorization. Suppose $D_{kk}(\alpha)$ is the (k,k) entry of $D(\alpha)$. How would you compute the value at zero of the derivative of $D_{nn}(\alpha)$. You may assume that A's LDL^T factorization is available. A script is not required–just be precise. Hint. Differentiate $A + \alpha e_1 e_1^T = L(\alpha)D(\alpha)L(\alpha)^T$ and manipulate.

Solution:

Differentiating $A + \alpha e_1 e_1^T = L(\alpha)D(\alpha)L(\alpha)^T$ gives

$$e_1 e_1^T = \dot{L}DL^T + L\dot{D}L^T + LD\dot{L}^T$$

and so if $Lv = e_1$ we have $vv^T = L^{-1}\dot{L}D + \dot{D} + D\dot{L}^T L^{-T}$. Note that $\dot{L}$ has a zero diagonal. By comparing diagonal entries it follows that v_n^2 is the derivative of of $D_{nn}(\alpha)$ at $\alpha = 0$.

3 points for the $e_1 e_1^T = \cdots$ equation and noting that $\dot{L}$ has a zero diagonal. 2 points for the rest.

6. (15 points) (a) Explain why the eigenvalues of a symmetric matrix are well-conditioned but that the eigenvectors may not be.

3 points for saying that the eigenvalue condition is 1/u'*v where u and v are unit left and right eigenvectors. This is unity in the symmetric case–the smallest possible value.

2 points for saying that the separation of eigenvalues affects eigenvector sensitivity. I nparticular, if λ is a distinct wigenvalue with a "neighbor" that is δ away, then the eigenvector invariant subspace moves ϵ/δ when $O(\epsilon)$ perturbations are made.

(b) The condition of the full rank least squares problem $\min \| Ax - b \|_2$ depends on $\kappa_2(A)^2$. Does that mean that the method of normal equations $A^TAx = A^Tb$ always produces solutions that are as accurate as those obtained when the QR factorization method is used?

With "hats" denoting computed quantities and ϵ the unit roundoff, we have

Normal equations:

$$\frac{\| \hat{x}_{LS} - x_{LS} \|}{\| x_{LS} \|} \approx \epsilon \kappa_2(A)^2$$

Via QR:

$$\frac{\| \hat{x}_{LS} - x_{LS} \|}{\| x_{LS} \|} \approx \epsilon \left(\kappa_2(A) + \rho_{LS} \kappa_2(A)^2 \right) \qquad \rho_{LS} = \| Ax_{LS} - b \|$$

So QR can be noticably more accurate in "small residual" problems.

Up to 3 points for vague κ vs κ^2 remarks

(c) Gaussian elimination with aprtial pivoting is applied to a 2-by-2 linear system $Ax = b$ and the computed result is given by

$$\hat{x} = \begin{bmatrix} 1.234567890123456 \\ 0.000012345678901 \end{bmatrix}$$

It is known that $\kappa_2(A) \approx 10^{10}$ and that the unit roundoff is about 10^{-17}. Underline the digits in $\hat{x}$ that are probably correct. Explain.

Since $\frac{\| \hat{x} - x \|}{\| x \|} \approx \epsilon \kappa_2(A) \approx 10^{-7}$ it follows that $\hat{x}$ has about 7 correct digits. This means $|\hat{x}_i - x_i| / \| x \| \approx 10^{-7}$. Since $x_1 \approx \| x \|$ we get 7 gitits in $\hat{x}_1$. But since $x_2 \approx 10^{-5}$ we do worse in the second component.

-2 for no explanation. -1 for too much underlining in $\hat{x}_2$

$$\text{reliable part of } \hat{x} \approx \begin{bmatrix} 1.234567 \\ 0.000012 \end{bmatrix}$$

7. (15 points) Complete the following function so that it performs as specified.

```
  function  X = MatrixLS(A,B,F,G)
% A, B, F, and G are m-by-n matrices with m > n.
% Assume that rank(A) = rank(B) = n.
% X is n-by-n and minimizes norm(A*X*B' - F*G','fro')
```

For full credit, the only MATLAB factorization function that you are allowed to use in this problem is the "economy-size" SVD:

```
[U,S,V] = SVD(X,0) produces the "economy size"
decomposition. If X is m-by-n with m > n, then only the
first n columns of U are computed and S is n-by-n.
```

Solution:

```
[m,n] = size(A);
[UA,SA,VA] = svd(A,0);
[UB,SB,VB] = svd(B,0);
F = UA'*F;
G = UB'*G;
X = F*G';
for j=1:n
   X(:,j) = X(:,j)/SB(j,j);
   X(j,:) = X(j,:)/SA(j,j);
end
X = VA*X*VB';
```

```
-1 for things like X = UA'*F*G'*UB because it is O(m^2)).

Up to seven points for relevant transformation
of problem but with no solving.
```