CS 4220: Final Exam and Grades

Take Home Final Averages based on 15 points per problem: P1 = 13, P2 = 12, P3 = 12. I spent some time getting faulty codes to work. The test sets were randomized and I always ran them several times to make sure I wasn't seeing fluke results.

Homework Average Normalized so each of the 5 assignments worth 20 points: H = 79.7
Prelim I Average: P = 75
Take Home Final Average Normalized to 100 points: T = 80.4
In-Class Final Average: F = 78
Median Total Score = .40*H + .15*P + .20*T + .25* F = 80

Exam Pickup possible after May 20. Send Email for my schedule

1. (15 points)

(a) Consider the nonsingular systems $Ax = b$ and $(A + E)y = b + f$ and assume that

$$\frac{\| E \|}{\| A \|} \approx \epsilon \qquad \frac{\| f \|}{\| b \|} \approx \epsilon$$

What can you say about $\| y - x \| / \| x \|$?

Answer (5 points)

$$\frac{\| y - x \|}{\| x \|} \approx \epsilon\kappa(A)$$

where $\kappa(A) = \| A \| \| A^1 \|$. -4 if no condition number factor

(b) Consider the following fragment designed to compute $v = F^{-1}G^{-1}b$

```
u = G\b;
v = F\u;
```

What can you say about the computed version of v? A proof is not required. However, you must justify any approximations that are part of your argument.

Answer (5 points)

Let $\tilde{v}$ be the computed version of v. We know that

$$\frac{\| \tilde{v} - v \|}{\| v \|} \approx \epsilon\kappa(F)$$

where ϵ is the relative error in the rhs. But that relative error is about unit roundoff times the condition of G. Thus,

$$\frac{\| \tilde{v} - v \|}{\| v \|} \approx \text{unit round off } \kappa(F)\kappa G$$

-2 for unit round off $(\kappa(F) + \kappa G)$

(c) "If A is nonsingular and E is small enough, then $A + E$ is nonsingular." Using the SVD, make this statement more precise.

Answer (5 points)

If $\| A \|_2 < \sigma_{min}$, the smallest singular value of A, then $A + E$ will be nonsingular.

2. (15 points)

Complete the following function so that it performs as specified using a minimum number of flops.

```
   function B = Prod(A,alfa,X,Y)
 % A is n-by-n, alfa is a scalar, and X and Y are n-by-p.
 % B = A*(I + alfa*X*Y')



   [n,p] = size(X)

   if p <= n
      B = A + (alfa*A*X)*Y'   % 10 points
   else
      B = A + (alfa*A)*(X*Y')  % 5 points
   end
```

3. (15 points)

If $C \in \mathbb{R}^{n\times n}$ is nonsingular and $u, v \in \mathbb{R}^n$, then the Sherman-Morrison formula gives a recipe for the inverse of $(C + uv^T)$ if this matrix is nonsingular. In particular,

$$(C + uv^T)^{-1} = C^{-1} + \alpha w z^T$$

where $w = C^{-1}u$, $z^T = v^T C^{-1}$, and $\alpha = -1/(1 + v^T C^{-1}u)$.

By making effective use of the SVD, complete the following function so that it performs as specified:

```
   function [x,z] = DoubleSolve(A,f,g,b)
 % A is a nonsingular n-by-n matrix.
 % f, g, and b are column n-vectors.
 % x and z are column n-vectors with the property that Ax = b and
 % (A + f*g')z = b. (Assume the latter system is nonsingular.)

   [U,S,V] = svd(A);

   x = V*((U'*b)./diag(S))        % -10 if you leave parens of and have mat-mat multiplies

   ftilde = V*((U'*f)./diag(S));

   alfa = -1/(1+ g'*ftilde);

   y = x + (alfa*(g'*x))*ftilde
```

4. (10 points)

(a) Suppose f and all its derivatives are continuous everywhere. "If x_0 is close enough to a root x_*, then with a starting value of x_*, Newton's method will converge quadratically to x_*." What does this mean? And what properties of f in the vicinity of x_* determine the criteria for being "close enough"?

Answer (5 points)

The radius of convergence depends on $|f''|$ (the smaller the better–2pts) and $|f'|$ (the larger the better –1 pt). Quadratic convergence means that the error is ultimately squared each step (2pts).

(b) The secant method has a local convergence rate that is less than that for Newton's method. Give as many reasons as you can why this doesn't make much difference in practice.

Answer (5 points)

Most of the time in an iteration is spent just getting close enough. During this phase local convergence properties are irrelevant (2pts). Secant does not require f' evaluations (2 points). $r = 2$ convergence might save you one step over $r = 1.6$ convergence given machine precision. (1 point)

5. (15 points)

Assume that $A \in \mathbb{R}^{m\times n}$ has rank $r < n$ and that $F \in \mathbb{R}^{n\times n}$ is nonsingular. Show how QR-with-column pivoting can be used to produce a minimizer of $\| AFx - b \|_2$ where $b \in \mathbb{R}^m$ is given. For full credit, your method must not explicitly form the matrix AF.

Answer
First solve min $\| Az - b \|$ using $AP = QR$. This involves choosing the largest r so that $R(r{:}m, r{:}n)$ is "small". The minimizing z is then given by

$$z = P \left[\begin{array}{c} R(1{:}r, 1{:}r)\backslash\tilde{b}(1{:}r) \\ 0 \end{array} \right]$$

where $\tilde{b} = Q^T b$. (10 points). Then solve $Fx = z$, say via LU. (5 points)

6. (15 points)

Consider the finite-difference Newton method for finding a zero of $F{:}R^n \to R^n$.

(a) Explain the interplay between "calculus errors" and rounding errors associated with the computation of the k-th column of the finite-difference Jacobian?

The calculation for column k is

$$\frac{F(x_c + h * e_k) - F(x_c)}{h}$$

The calulus error is $O(h)$ while the roundoff error is $O(u/h)$ where u is the unit roundoff. Minimizing $(h + u/h)$ is roughly the goal. $h_{opt} \approx \sqrt{u}$. 5 points

(b) Suppose $n = 6$ and that for all x the component functions in

$$F(x) = [F_1(x)\;\; F_2(x)\;\; F_3(x)\;\; F_4(x)\;\; F_5(x)\;\; F_6(x)\;]^T$$

have these properties:

F_1 depends only on $x_2,\ x_5$
F_2 depends only on $x_1,\ x_6$
F_3 depends only on $x_3,\ x_4$
F_4 depends only on $x_5,\ x_6$
F_5 depends only on $x_1,\ x_4$
F_6 depends only on $x_2,\ x_3,\ x_6$

How many F-evaluations would be required to compute the finite difference Jacobian? Explain.

Answer The Jacobian looks like this:

$$J = \begin{bmatrix} 0 & x & 0 & 0 & x & 0 \\ x & 0 & 0 & 0 & 0 & x \\ 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & x & x \\ x & 0 & 0 & x & 0 & 0 \\ 0 & x & x & 0 & 0 & x \end{bmatrix}$$

If

$$\frac{F(x + he_1 + he_3 + he_5) - F(x)}{h} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

then

$$J(:,1) \approx \begin{bmatrix} 0 \\ b \\ 0 \\ 0 \\ e \\ 0 \end{bmatrix} J(:,3) \approx \begin{bmatrix} 0 \\ 0 \\ c \\ 0 \\ 0 \\ f \end{bmatrix} \qquad J(:,5) \approx \begin{bmatrix} a \\ 0 \\ 0 \\ d \\ 0 \\ 0 \end{bmatrix}$$

If

$$\frac{F(x + he_2 + he_4) - F(x)}{h} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

then

$$J(:,2) \approx \begin{bmatrix} a \\ 0 \\ 0 \\ 0 \\ 0 \\ f \end{bmatrix} \qquad J(:,4) \approx \begin{bmatrix} 0 \\ 0 \\ c \\ 0 \\ e \\ 0 \end{bmatrix}$$

If

$$\frac{F(x + he_6) - F(x)}{h} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

then

$$J(:,6) \approx \begin{bmatrix} 0 \\ b \\ 0 \\ d \\ 0 \\ f \end{bmatrix}$$

So four f-evals required

7. (20 points)

(a) Assume that $A \in \mathbb{R}^{n\times n}$ is symmetric and that we wish to solve the linear system

$$(A^2 - \mu I)x = b$$

for many different values of μ, all of which are positive. What factorization would you use to do this? Discuss the overall amount of work.

Answer If $A = QDQ^T$ is the Schur decomposition $O(n^3)$, then

$$A^2 - \mu I \;=\; (QDQ^T)^2 - \mu I \;=\; Q(D^2 - \mu I)Q^T.$$

It follows that $(A^2 - \mu I)x = b$ transforms to

$$(D^2 - \mu I)\tilde{x} = \tilde{b}$$

where $\tilde{b} = Q^T b$ and $\tilde{x} = Q^T x$. It follows that $x = Q\left(D^2 - \mu I\right)^{-1}\tilde{b}$ can be computed in $O(n^2)$ flops.

(b) Suppose the Hessian for a function $f{:}R^n \rightarrow R$ is always tridiagonal. We wish to use Newton's method to find a zero of the gradient of f. What factorization would you use to solve the linear system associated with each step? Would the linear equation solving cost $O(n)$, $O(n^2)$, or $O(n^3)$? Explain.

Answer (10 points) $PA = LU$ costs $O(n)$ for tridiagonal A. Tridiagonal Cholesky is also $O(n)$ but you cannot assume that the Hessian is positive definite unless you are near a local minima. (-4 points)

Some MATLAB Functions

LU Factorization

```
[L,U,P] = LU(X) returns unit lower triangular matrix L, upper
triangular matrix U, and permutation matrix P so that
P*X = L*U.
```

Cholesky Factorization

```
R = CHOL(X)  returns an upper triangular R so that R'*R = X
where X is symmetric and positive definite.
```

QR Factorization

```
[Q,R,E] = QR(A) produces unitary Q, upper triangular R and a
permutation matrix E so that A*E = Q*R. The column permutation E is
chosen so that ABS(DIAG(R)) is decreasing.
```

Singular Value Decomposition

```
[U,S,V] = SVD(X) produces a diagonal matrix S, of the same
dimension as X and with nonnegative diagonal elements in
decreasing order, and unitary matrices U and V so that
X = U*S*V'.
```

Schur Decomposition

```
[U,D] = SCHUR(X) produces a diagonal matrix D and
an orthogonal matrix U so that X = U*D*U' assuming that X
is real and symmetric.
```