CS 621: Assignment 6

Due: Friday, December 3, 2004 (In Lecture or 4130 Upson by 4pm)

Scoring for each problem is on a 0-to-3 scale (3 = complete success, 2 = overlooked a small detail, 1 = germ of the right idea, 0 = missed the point of the problem.) One point will be deducted for insufficiently commented code. Unless otherwise stated, you are expected to utilize fully MATLAB'S vectorizing capability subject to the constraint of being flop-efficient. Test drivers and related material are posted on the course website `http://www.cs.cornell.edu/courses/cs621/2004fa/`. For each problem submit output and a listing of all scripts/functions that you had to write in order to produce the output. You are allowed to discuss *background* issues with other students, but the codes you submit must be your own.

P1. (Invariant Subspace Associated with a Complex Eigenvalue)

Complete the following function so that it performs as specified

```
  function [Q,sep] = Invariant22(T)
% T is a real n-by-n upper quasi-triangular matrix.
% Assume that T11 = T(1:n-2,1:n-2) is upper triangular.
% Assume that T22 = T(n-1:n,n-1:n) has complex eigenvalues.
% Q is a real n-by-2 matrix with Q'*Q = eye(2,2) such that T*Q = Q*S where S (2-by-2)
%    has the same eigenvalues as T22.
% sep = sep(T11,T22).
```

You are not allowed to use the MATLAB function `ordschur`. Regarding the *sep* function, recall that if $T_{11} \in \mathbb{R}^{r\times r}$ and $T_{22} \in \mathbb{R}^{(n-r)\times(n-r)}$ then

$$\text{sep}(T_{11}, T_{22}) = \min_{X \in \mathbb{R}^{r\times(n-r)},\ \| X \|_F = 1} \| T_{11}X - XT_{22} \|_F$$

Show that this is the minimum singular value of a matrix $M \in \mathbb{R}^{(2n-2)\times(2n-2)}$ that is upper quasi-triangular. You may apply `svd` to this matrix to compute the required minimum singular value. Test your implementation with the script `P1`.

P2. (Hessenberg Decomposition $U^TAU = H$ with Special U)

The Hessenberg Decomposition $U^TAU = H$ is not unique. The Householder reduction computes U as a product of Householder transformations $U = P_1 \cdots P_{n-2}$ with the property that $U(:,1) = I_n(:,1)$. The MATLAB function `hess` does this. Now if P_0 is *any* orthogonal matrix and $U^T(P_0^TAP_0)U = H$ is obtained by applying `hess` to $P_0^TAP_0$, then $Q^TAQ = H$ is a Hessenberg decomposition and $Q = P_0U$ has the same first column as P_0. Complete the following function so that it performs as specified

```
  function [Q,H] = SpecialHess(A)
% A is an n-by-n real matrix.
% Q is orthogonal such that (a) Q'*A*Q = H is upper hessenberg and
% (b) ones(1,n)*Q(:,2:n) = zeros(1,n-1)
```

P3 & P4. (The Second Eigenvalue of the Google Matrix)

Suppose we have a web with N pages. Let $G \in \mathrm{I\!R}^{N \times N}$ be a matrix with the property that

$$g_{ij} = \begin{cases} 1 & \text{if there is a link to page } i \text{ on page } j \\ 0 & \text{otherwise} \end{cases}$$

Suppose $0 < p < 1$ and $e \in \mathrm{I\!R}^N$ is the vector of all ones. Define the *Google matrix* $A_p \in \mathrm{I\!R}^{N \times N}$ as follows

$$A_p(:,j) \;=\; \begin{cases} pG(:,j)/\| \, G(:,j) \, \|_1 \;+\; (1-p)e/N & \text{if } G(:,j) \neq 0 \\ e/N & \text{otherwise} \end{cases}$$

Since A_p is an unreduced Markov matrix, we can find $x \in \mathrm{I\!R}^N$ with $x_i > 0$ and $\| \, x \, \|_1 = 1$ so $A_p x = x$. Let λ_2 be the second largest eigenvalue of A_p in absolute value. Note that $|\lambda_2| \to 0$ as $p \to 0$ because A_p starts to look more and more like the rank-one matrix ee^T/N. Complete the following function so that it performs as specified:

```
  function [x,Lambda2,condLambda2] = GoogleMatrix(G,p,k)
% G is an N-by-N 0-1 matrix represented in sparse form and 0 < p < 1.
% k is a positive integer, the number of orthogonal iteration steps
% Let A = A_p be the Google matrix.
% x is a column N-vector with positive entries so that Ax = x (approximately)
% Lamda2 is the absolute value of A's second largest eigenvalue in
%    absolute value
% Lambda2 is the condition number of Lambda2, i.e., the secant of the
% angle between the left and right eigenvector associated with Lambda2.
```

The output values should be based upon k steps of "2-column" orthogonal iteration applied to both A_p and A_p^T. (You need the latter since a left eigenvector is required.) Use random starting values. Note that even though G is in sparse form, you can act as if it is conventionally stored. For example

```
x = rand(N,1);
y = G*x;
alpha = norm(G(:,j),1);
```

A function `G = MakeG(N,q)` is provided on the website which generates the G matrix (in sparse form) for a random web in which each page has up to q outlinks to random pages.

Test your implementation of GoogleMatrix on `P3and4`. It is critical that your solution be well-commented.