Math 477/577 — Computer Assignment 6, due Nov.28, 2006

1. This five-part problem asks you to put together a MATLAB program that finds all the eigenvalues of a real symmetric matrix, using only elementary building blocks. It is not necessary to achieve optimal constant factors by exploiting symmetry or zero structure optimally. It is possible to solve the whole problem by a program about fifty lines long.

   (a) Write a function `T = tridiag(A)` that reduces a real symmetric $m \times m$ matrix to tridiagonal form by orthogonal similarity transformations. Your program should use only elementary MATLAB operations – not the built-in function `hess`, for example. Your output matrix `T` should be symmetric and tridiagonal up to rounding errors. If you like, add a line that forces `T` at the end to be exactly symmetric and tridiagonal. For an example, apply your program to `A = hilb(4)`.

   (b) Write a function `Tnew = qralg(T)` that runs the unshifted QR algorithm on a real tridiagonal matrix `T`. For the QR factorization at each step, use programs `[W,R] = house(A)` and `Q = formQ(W)` of Computer Assignment 3 if available, or MATLAB's command `qr`, or, for greater efficiency, a new code based on 2×2 Householder reflections rather than $m \times m$ operations. Again, you may wish to enforce symmetry and tridiagonality at each step. Your program should stop and return the current tridiagonal matrix `T` as `Tnew` when the $m, m-1$ element satisfies $|t_{m,m-1}| < 10^{-12}$ (hardly an industrial strength convergence criterion!). Again, apply your program to `A = hilb(4)`.

   (c) Write a driver program which (i) calls `tridiag`, (ii) calls `qralg` to get one eigenvalue, (iii) calls `qralg` with a smaller matrix to get another eigenvalue, and so on until all of the eigenvalues of A are determined. Set things up so that the values $|t_{m,m-1}|$ at every QR iteration are stored in a vector and so that at the end, your program generates a semilogy plot of these values as a function of the number of QR factorizations. (Here m will step from `length(A)` to `length(A)-1` and so on down to 3 and finally 2 as the deflation proceeds, and the plot will be correspondingly sawtoothed.) Run your program for `A = hilb(4)`. The output should be a set of eigenvalues and a "sawtooth plot".

   (d) Modify `qralg` so that it uses the Wilkinson shift at each step. Turn in the new sawtooth plot for the same example.

   (e) Rerun your program for the matrix `A = diag(15:-1:1) + ones(15,15)` and generate two sawtooth plots corresponding to shift and no shift. Discuss the rates of convergence observed here and for the earlier matrix. Is the convergence linear, superlinear, quadratic, cubic ...? is it meaningful to speak of a certain "number of QR iterations per eigenvalue"?

2. Let A be the $m \times m$ upper-triangular matrix with 0.1 on the mail diagonal and 1 everywhere above the diagonal. Write a program to compute the smallest singular value of A in two ways: by calling MATLAB's standard `svd` command, and by forming A^*A and computing the square root of its smallest eigenvalue. Run your program for $1 \leq m \leq 30$ and plot the results as two curves on a log scale. Do the results conform to our general discussion of these algorithms?