CSci 5304 HW5 Due: Dec 3 2014

- This assignment is a mix of handwritten questions and Matlab coding questions. You should hand in everything on paper, including a printout of your matlab functions and scripts.
- You are encouraged to do this in pairs: hand in one copy of the entire assignment with both names on it. You should still satisfy yourself that you could answer every question on your own, since similar questions will appear on future exams.

1. Problem 7.3.2 p373: Suppose the eigenvalues of a real matrix A of largest absolute value are a complex conjugate pair, and that all the other eigenvalues are strictly smaller in absolute value. How does the power method behave when started with a real vector? Show the power method yields an approximate basis for the invariant subspace spanned by the two eigenvectors of those two largest eigenvalues.

2. What are the eigenvalues and eigenvectors of a Householder reflection $P = I - 2uu'$ and a 2×2 Givens rotation $\begin{pmatrix} c & s \\ -s & c \end{pmatrix}$?

3. For each of the following upper triangular matrices, list all the eigenvalues together with their algebraic and geometric multiplicities. Solve for all the corresponding eigenvectors, scaled so that their largest component in absolute value is a 1. (Hand calculation).

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$

4. Generate a random $n \times n$ symmetric matrix A with eigenvalues $1, \ldots, n-2, 2n-1, 2n$ using the following matlab commands:

   - `[Q,R] = qr(rand(n,n)) ;`
   - `A = Q * diag([1:n-2,2*n-1,2*n]) * Q' ;`

   Use this matrix as test case for the following algorithms. For each algorithm, show the code [fragment] and give an outline of the method (if not self-explanatory from the code fragment). find how many steps it takes to find a eigenvalue/vector pair within an accuracy of 10^{-7}. You should check the accuracy by computing the residual, not by comparing to the true answer. Use $n = 5, 10$ to test your algorithm, and then use $n = 40$ to report the number of iterations needed.

   (a) Power method starting with a random starting vector. At each iteration compute and save the Rayleigh quotient, then plot the difference between the values of these Rayleigh quotients and the final value (use `semilogy`). What is the rate of convergence? Did the power method converge to the largest eigenvalue? Can the rate of convergence give an indication of the second largest eigenvalue? Hint: since this matrix is symmetric, you might need to take the square root of the rate of convergence. Do not go over 1,000 iterations.

   (b) Rayleigh Quotient Iteration (RQI: inverse iteration with the shift at each step determined by the Rayleigh quotient with the iterate). Start with a random normalized vector. Which eigenvalue does it converge to? To force it to converge to the largest eigenvalue, apply RQI with a shift fixed at $\|A\|_\infty$ for a few initial iterations before switching to the pure Rayleigh quotient. How many initial iterations with fixed shifts do you need to make it converge to the largest eigenvalue? This number should be less than 10.

(c) Simple explicit QR algorithm (no shifts). Try symmetrizing after each iteration. Allow up to 1000 iterations.

5. Implement a simple Lanczos procedure (as given in the lecture notes, or Alg. 10.1.1* in the text or Alg. 36.1 in TB) to compute the factorization $AX = XT$ where A is a given symmetric matrix, X is a set of generated orthonormal columns, and T is a generated tridiagonal matrix. Apply the procedure to the test matrices in the previous question and use the result to obtain a Ritz approximation (§10.1.4) to the leading eigenvalue/vector pair for A. You should use the standard simple Lanczos procedure, except that you should check for loss of orthogonality by the following heuristic: check that each generated Lanczos vector is orthogonal to the starting vector (i.e., the inner product is less than the given tolerance). Carry out the Lanczos expansion until n Lanczos vectors have been generated, or until an off-diagonal element of T is less than 10^{-7} (in absolute value), or until orthogonality is lost (to the same tolerance). Use a random starting vector.

   Compute the Ritz approximation to the eigenvalue at each iteration, and plot the differences between the Ritz values and the final value as done with the Power Method above. You can also just compute the maximum eigenvalue of the leading $k \times k$ principal submatrix of the final T, for $k = 1, 2, \ldots$. To obtain the Ritz values, just apply the built-in `eig` function to the tridiagonal matrix T.

The following is required only for those working in pairs.

6. Apply the power method as well as the Lanczos procedure to the matrix A defined by the following commands using $\texttt{n} = 20$:
   - `A = diag([n+2:-1:2,3:n+2])+diag(ones(2*n,1),-1)+diag(ones(2*n,1),1);`

   Notice how $A - \lambda I$ almost has a 2-dimensional nullspace for some λ, even though no sub-diagonal is small.

7. Use your Lanczos procedure to find the leading singular value/vectors of the matrix in `lsidata` from the previous homework, by implicitly finding the leading eigepair for $A^T A$ (or AA^T if easier). Write your code so that you do not have to explicitly form $A^T A$ or even explicitly transpose the matrix A at all. You can use the `svds` function to check your answers.

*This algorithm should be rearranged so that the matrix A is accessed only once per iteration.