New York University
Courant Institute of Mathematical Sciences

Homework 8

Objective: Eigenvalue decompostion based on Givens Rotation. SVD based on the pivoted Gram-Schmidt.

1. Given n-by-n symmetric matrix A, write a Matlab function `lambda=jacobi(A,eps0)` to implement the Jacobi Algorithm and find the eigenvalues `lambda` = [$\lambda_1,\ \lambda_2, \ldots, \lambda_n$] of A.

As usual, `eps0` is a threshold to terminate the sweep iterations. If the Frobenius norm of `A-diag(A)` divided by `norm(A, 'fro')` is smaller than `eps0`, we terminate terminate the iterations.

`lambda` is a vector of size n. Note that it is not required to compute the eigenvectors in this exercise.

It is always prudent to first verify, on a 2-by-2 symmetric matrix, the basic step of Jacobi Algorithm

$$J^T \begin{bmatrix} a & d \\ d & b \end{bmatrix} J = \begin{bmatrix} * & 0 \\ 0 & * \end{bmatrix}, \quad J = \begin{bmatrix} \cos(\theta)\sin(\theta) \\ -\sin(\theta)\cos(\theta) \end{bmatrix}, \quad \theta = \texttt{atan2}(2*\texttt{d}, \texttt{b}-\texttt{a})/2. \tag{1}$$

(i) Provide Matlab script for the function

(ii) Run it on the matrix `A=hilb(12)` with eps0=1.0e-16. Provide the number of sweep required for the prescribed precision eps0. For all the sweeps, print the ratios

$$\frac{\|\texttt{A} - \texttt{diag(A)}\|_F}{\|\texttt{A}\|_F} \tag{2}$$

(iii) Show the eigenvalues with long e format. Compare them with the eigenvalues obtained via Matlab function `[q,s]=eig(A)` by computing the ratios of the corresponding pairs

$$\lambda_j / s_j, \quad j = 1:12 \tag{3}$$

2. Given m-by-n matrix A, write a Matlab function `[sigma,rank]=qralg(A,eps0)` to implement the QR Algorithm and find the singular values `sigma` = [$\sigma_1,\ \sigma_2, \ldots, \sigma_{\text{rank}}$] of A. You are required to use the pivoted Gram-Schmidt function you constructed in the previous homework.

The basic idea behind the QR Algorithm is to successively produce a sequence of matrices `A0, A1, A2, ...` with `A0=A`. More precisely, a cycle in the QR Algorithm iteration consists of

a. Compute the QR decomposition (based on the pivoted Gram-Schmidt algorithm) of A0, so that A0=Q*R

b. Compute the QR decomposition of transpose(R), so that transpose(R)=U*T is its QR factorization where U is like Q, and T is like R.

c. At this point, we have A=A0=Q*R = Q * T' * U'. Thus, we call T' our next A matrix: A1=T'. Repeat to A1 what we did to A0 by following steps a, b, c.

As in Problem 1, `eps0` plays exactly the same role to terminate the cycle iterations. Note that it is not required to compute the singular vectors in this exercise.

(i) Provide Matlab script for the function

(ii) Run it on the matrix `A=hilb1(m,n)` with m=20, n=10, and eps0=1.0e-16. Show the singular values with long e format.

(iii) Compare them with the singular values obtained via Matlab function `[u,s,v]=svd(A)` by computing the ratio of the corresponding pairs

$$\sigma_j/s_j, \quad j = 1 : rank \tag{4}$$

3. Use the Matlab function `[sigma,rank]=qralg(A,eps0)` you constructed in Problem 2 to investigate the behavior of convergence of the diagonal entries of A0, A1, A2, ... to the singular values.

(i) Construct a "random" matrix A with known singular values sigma=[0.1:0.1:1] by: [U,r]=qr(rand(15,10)); [V,r]=qr(rand(10,10)); A = U * diag(sigma) * V'

(ii) Run `[sigma,rank]=qralg(A,eps0)` with eps0=1.0e-16, and watch how each of the designated singular value is approached by the corresponding diagonal entry of A0, A1, A2, ... Pay particular attention to the rate of convergence of each diagonal entry. Use no more than 30 words to summarize your observations.