HW7

Wednesday, April 3, 2024 5:13 PM

1. **[Q1]** (10 pts) **(a)** A is Hermitian if $A^H = A$, and skew-Hermitian if $A^H = -A$. Show that the eigenvalues of a Hermitian matrix (e.g., real symmetric) are real, and those of a skew-Hermitian (e.g., real skew-symmetric) are purely imaginary. In both cases, show that the eigenvectors associated with distinct eigenvalues are orthogonal.

(b) For a block upper triangular $F = \begin{bmatrix} F_{11} & F_{12} & \cdots & F_{1n} \\ & F_{22} & \cdots & F_{2n} \\ & & \ddots & \vdots \\ & & & F_{nn} \end{bmatrix}$, show that $\Lambda(F) = \bigcup_{k=1}^{n} \Lambda(F_{kk})$, where $\Lambda(\cdot)$ denotes the spectrum (all eigenvalues) of a matrix.

ai) Let (λ, v) and (μ, w) be distinct eigenpairs.

$$\lambda\langle v,v\rangle = \langle \lambda v, v\rangle = \langle Av, v\rangle = \overline{\langle v, Av\rangle} = \langle v, A^H v\rangle$$
$$= \langle v, Av\rangle$$
$$= \langle v, \lambda v\rangle$$
$$= \bar{\lambda}\langle v, v\rangle$$

$\Rightarrow \lambda = \bar{\lambda} \quad \therefore \lambda \in \mathbb{R}$

$$\lambda\langle v,w\rangle = \langle Av, w\rangle = \langle v, Aw\rangle = \mu\langle v,w\rangle$$

$\Rightarrow (\lambda - \mu)\langle v,w\rangle = 0 \Rightarrow v \perp w$

aii) Let (λ, v) and $\langle \mu, w\rangle$ be distinct eigenpairs

$$\lambda\langle v,v\rangle = \langle \lambda v, v\rangle = \langle Av, v\rangle = \overline{\langle v, Av\rangle} = \langle v, A^H v\rangle$$
$$= -\langle v, Av\rangle$$
$$= -\langle v, \lambda v\rangle$$
$$= -\bar{\lambda}\langle v, v\rangle$$

$\Rightarrow \lambda = -\bar{\lambda} \quad \therefore \lambda \in \mathbb{C}$

$$\lambda\langle v,w\rangle = \langle Av, w\rangle = -\langle v, Aw\rangle = -\mu\langle v,w\rangle$$

$\Rightarrow (\lambda + \mu)\langle v,w\rangle = 0 \Rightarrow v \perp w$

b) Let $I = \begin{bmatrix} I_1 & & 0 \\ & I_2 & \\ & & \ddots \end{bmatrix} \ni \dim I_k = \dim F_{kk}$.

$$I = \begin{bmatrix} I_1 & & & \\ & I_2 & & 0 \\ & & \ddots & \\ 0 & & & I_n \end{bmatrix} \ni \dim I_k = \dim F_{kk}.$$

$\Lambda(F) = \det(F - \lambda I)$ and $\Lambda(F_{kk}) = \det(F_{kk} - \lambda I_k)$.
Since F is block upper triangular,

$$\det(F - \lambda I) = \prod_{k=1}^{n} \det(F_{kk} - \lambda I_k).$$

$\Rightarrow \Lambda(F) = \bigcup_{k=1}^{n} \Lambda(F_{kk})$

2. **[Q2]** (10 pts) **(a)** Given a complex Schur form $U^H A U = T$, where T is upper triangular, show that the first k columns of U, $u_1, u_2, \ldots, u_k$, span an invariant subspace of A. That is, $A\,\mathrm{span}\{u_1, \ldots, u_k\} \equiv \mathrm{span}\{Au_1, \ldots, Au_k\} \subset \mathrm{span}\{u_1, \ldots, u_k\}$.

(b) Let $U \in \mathbb{R}^{n\times p}$ $(n > p)$ contains basis vectors of an invariant subspace of A, such that $AU = UM$ for some $M \in \mathbb{R}^{p\times p}$. Show that the eigenvalues of M are also eigenvalues of A. If, in addition, $W \in \mathbb{R}^{n\times m}$ $(n > m > p)$ has orthonormal columns, and $\mathrm{col}(U) \subset \mathrm{col}(W)$, show that the eigenvalues of M are eigenvalues of $W^T A W$.

a) Let $x = \sum_{i=1}^{k} c_i u_i$. $\quad Ax \in \mathrm{span}\{Au_1, \ldots, Au_k\}$

$$\begin{aligned} Ax &= \sum_{i=1}^{k} c_i A u_i \\ &= \sum_{i=1}^{k} c_i \sum_{\ell=1}^{n} a_{i\ell} u_{\ell i} \\ &= \sum_{i=1}^{k} \sum_{\ell=1}^{n} (c_i a_{i\ell}) u_{\ell i} \in \mathrm{span}\{u_1, \ldots, u_k\} \end{aligned}$$

since $c_i a_{i\ell} \in \mathbb{C}$

b) Let (λ, v) be an eigenpair of M

$$\begin{aligned} AUv &= UMv \\ &= \lambda U v \end{aligned}$$
$\Rightarrow \lambda \in \Lambda(A)$

$$\begin{aligned} Mv &= \lambda v \\ UMv &= \lambda U v \\ AUv &= \lambda U v \\ W^T A U v &= \lambda W^T U v \\ W^T A W U v &= \lambda W^T W U v = \lambda U v \end{aligned}$$

Thus $\Lambda(M) = \Lambda(W^T A W)$

3. **[Q3]** (15 pts) **(a)** Describe a procedure to post-process the Q and R factors of Givens or Householder QR, such that the R factor has all non-negative diagonal entries.

(b) Verify numerically that the Simultaneous Iteration is equivalent to the unshifted QR iteration. To this end, first construct an upper Hessenberg H_0 as follows

```
rng('default'); H0 = triu(randn(7,7),-1);
```

Implement the Simultaneous Iteration and the QR iteration, described in Trefethen's book, Chapter 28. Feel free to use MATLAB's `qr`, followed by the post-processing in part (a), and the '`*`' operation directly to form $H^{(k)} = R^{(k)}Q^{(k)}$ (that is, no need to use the Givens rotations to perform the QR iteration as usually supposed to).

Compare the projection matrices $H_{SI}^{(k)}$ in (28.10) for simultaneous iteration and $H_{QR}^{(k)}$ in (28.13) in the QR iteration. Find the relative difference $\frac{\|H_{SI}^{(k)}-H_{QR}^{(k)}\|_F}{\|H_{SI}^{(k)}\|_F}$ for $k = 3$, 30, 300 and 3000, and the relative difference in the eigenvalues of $H_{SI}^{(k)}$ and $H_{QR}^{(k)}$ at these steps? What if the post-processing is not used, and in this case, do $H_{SI}^{(k)}$ and $H_{QR}^{(k)}$ have numerically the same eigenvalues?

(c) Find the eigenvalues of H_0, then use the theory we learned from class to estimate the rate of convergence of $H_{QR}^{(k)}$ toward the quasi-upper triangular T of the real Schur form. About how many iterations are needed to achieve $\|H_{QR}^{(k)} - T\|_F/\|T\|_F \approx \epsilon_{mach}$?

a)
```
for i=1:size(R)
  if r_ii < 0
    R(i,:)=-R(i,:)
    Q(:,i)=-Q(:,i)
  end
end
```

This comes from analysing the multiplication of Q and R to cancel out the negative introduced.

b) See code Comparation is done with the norm.

If PP is not done eigvalues are the same, but $H_{SI}^{(k)}$ and $H_{QR}^{(k)}$ are not.

c) Convergence rate $\rho \approx 0.8255$

$\rho^k < \varepsilon_m$

$k \log\rho < \log \varepsilon_m$

$k \approx \left\lceil \frac{-16}{\log(0.8255)} \right\rceil = 193$

4. **[Q4]** (10 pts) (Trefethen's book Prob. 28.2, but for the nonsymmetric case).

(a) Explore the nonzero structure of the Q factor of the QR factorization of an upper Hessenberg matrix, and verify that RQ is also upper Hessenberg. For clarity, you may give an illustration for a 5×5 upper Hessenberg.

(b) The computation of $H^{(k)} = R^{(k)}Q^{(k)}$, if done naively (by direct evaluation of the matrix-matrix multiplication), would need $\mathcal{O}(n^3)$ operations. Fortunately, $H^{(k)}$ can be computed only in $\mathcal{O}(n^2)$ operations. Explain, by Givens rotations, how this is achieved. Make sure that you do see the difference in cost.

a) $A = QR$ $\quad A_{ij} = \sum_{k=1}^{j} Q_{ik} R_{kj}$ $\quad$ Since R is upper triangular

Since $A_{i+2,j} = 0$ $\quad 0 = \sum_{k=1}^{j} Q_{i+2,k} R_{kj}$

$\Rightarrow Q_{i+2,k} = 0 \ \forall k$

Thus Q is upper Hessenberg

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 \end{bmatrix} \begin{bmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & * \end{bmatrix}$$

$$\begin{bmatrix} \times \\ \times \\ 0 \\ 0 \\ 0 \end{bmatrix} = * q_1 \qquad \begin{bmatrix} \times \\ \times \\ \times \\ 0 \\ 0 \end{bmatrix} = * q_1 + * q_2 \qquad \begin{bmatrix} \times \\ \times \\ \times \\ \times \\ 0 \end{bmatrix} = * q_1 + * q_2 + * q_3$$

$$\Rightarrow q_1 = \begin{bmatrix} \Delta \\ \Delta \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad q_2 = \begin{bmatrix} \Delta \\ \Delta \\ \Delta \\ 0 \\ 0 \end{bmatrix} \qquad q_3 = \begin{bmatrix} \Delta \\ \Delta \\ \Delta \\ \Delta \\ 0 \end{bmatrix}$$

Thus Q is upper Hessenberg

b) Since only $H_{i+1,i}$ is non-zero below the diagonal, a Givens rotation with 6 Flops will reduce, at each step k, to R. The reduced QR then would be the final R and the product of the $n-1$ Givens rotations.

```
format long
rng('default');
H0 = triu(randn(7,7),-1);

flag = true;
m = zeros(8,2);
[l,n]=normf(H0,3,flag);
m(1,:)=[l,n];
[l,n]=normf(H0,30,flag);
m(2,:)=[l,n];
[l,n]=normf(H0,300,flag);
m(3,:)=[l,n];
[l,n]=normf(H0,3000,flag);
m(4,:)=[l,n];

flag = false;
[l,n]=normf(H0,3,flag);
m(5,:)=[l,n];
[l,n]=normf(H0,30,flag);
m(6,:)=[l,n];
[l,n]=normf(H0,300,flag);
m(7,:)=[l,n];
[l,n]=normf(H0,3000,flag);
m(8,:)=[l,n];

m

[V,d]=eig(H0,'vector');
d=sort(unique(abs(d)),'descend');

l=zeros(size(d)-1);

for i = 1:size(d)-1
    l(i)=d(i+1)/d(i);
end
max(l)

function [l,n] = normf(A,k,flag)
    H0=SI(A,k,flag);
    H1=UQR(A,k,flag);
    l=norm(H0-H1,'fro')/norm(H0,'fro');

    [V0,d0]=eig(H0,'vector');
    [V1,d1]=eig(H1,'vector');
    d0=sort(d0);
    d1=sort(d1);
    n=norm(d0-d1)/norm(d0);
end

function H = SI(A,k,flag)
    Q=eye(size(A));
    for i=1:k
```

```
        Z=A*Q;
        [Q,R]=qr(Z);
        if flag == true
            [Q,R]=PP(Q,R);
        end
    end
    H=Q'*A*Q;
end

function H = UQR(A,k,flag)
    H=A;
    for i=1:k
        [Q,R]=qr(H);
        if flag == true
            [Q,R]=PP(Q,R);
        end
        H=R*Q;
    end
end

function [Q,R]=PP(Q, R)
    for i=1:size(R)
        if R(i,i)<0
            R(i,:)=-R(i,:);
            Q(:,i)=-Q(:,i);
        end
    end
end
```

```
m =

   0.000000000000001   0.000000000000001
   0.000000000000019   0.000000000000001
   0.000000000000577   0.000000000000002
   0.000000000014188   0.000000000000004
   1.069773291942645   0.000000000000001
   0.747451250140599   0.000000000000001
   0.944538621532716   0.000000000000002
   1.016723559782635   0.000000000000004


ans =

   0.825454625087606
```

Published with MATLAB® R2023b