

Chapter 2

Basic Concepts for Matrix Computations

2.1 Exercises

Exercise 2.1 Find the eigenvalues and spectral radius of matrices:

1. $A = \begin{bmatrix} 2 & 1 \\ -1 & 1 \end{bmatrix}$

Answer:

$$\lambda^2 - 3\lambda + 3 = 0$$

$$\lambda_1 = \frac{3}{2} + i\frac{\sqrt{3}}{2} \quad \lambda_2 = \frac{3}{2} - i\frac{\sqrt{3}}{2}$$

$$\rho(A) = \max_{\lambda \in \Lambda(A)} (|\lambda|) = 3$$

2. $A = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$

Answer:

$$\lambda^2 - 3\lambda + 1 = 0$$

$$\lambda_1 = \frac{3+\sqrt{5}}{2} \quad \lambda_2 = \frac{3-\sqrt{5}}{2}$$

$$\rho(A) = \max_{\lambda \in \Lambda(A)} (|\lambda|) = \frac{3+\sqrt{5}}{2}$$

3. $A = \begin{bmatrix} 0 & 2 & 1 \\ 1 & 2 & 1 \\ -1 & 1 & 2 \end{bmatrix}$

Answer:

$$\lambda^3 - 4\lambda^2 + 2\lambda + 3 = (\lambda - 3)(\lambda^2 - \lambda - 1) = 0$$

$$\lambda_1 = -0.618 \quad \lambda_2 = 3 \quad \lambda_3 = 1.618$$

$$\rho(A) = \max_{\lambda \in \Lambda(A)} (|\lambda|) = 3$$

4. $A = \begin{bmatrix} 2 & 2 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$

Answer:

$$\lambda^3 - 6\lambda^2 + 8\lambda - 3 = (\lambda - 1)(\lambda^2 - 5\lambda + 3) = 0$$

$$\lambda_1 = 0.6972 \quad \lambda_2 = 4.3028 \quad \lambda_3 = 1$$

$$\rho(A) = \max_{\lambda \in \Lambda(A)} (|\lambda|) = 4.3028$$

Exercise 2.2 Let u and v be 2 **nonzero** (column) vectors in \mathbb{R}^n . Let $A = I - uv^T$ where I is the n by n identity matrix.

1. Prove that the set of eigenvalues of A reduces to at most 2 numbers:

$$\Lambda(A) = \{\lambda_1, \lambda_2\},$$

where $\lambda_1 = 1$, $\lambda_2 = 1 - v^T u$. Which property should the vectors u and v satisfy in order for A to be singular (not invertible)?

Answer: Let p be an eigenvector of A . Then $Ap = \lambda p$, i.e., $p - uv^T p = \lambda p$. Hence:

$$(1 - \lambda)p = uv^T p = (v^T p)u.$$

Therefore, we have two options:

- 1) $\lambda_1 = 1$ and $v^T p = 0$, i.e. $E_1 = \{span\{v\}\}^\perp$ is the eigenspace of dimension $n - 1$ corresponding to the eigenvalue λ_1 , or
- 2) $\lambda_2 \neq 1$, hence $E_2 = \{u\}$ is the eigenspace of dimension 1, co-linear to u , corresponding to $\lambda_2 = 1 - v^T u \neq 1$, i.e., u and v are not orthogonal. In case u and v are orthogonal, then 1 is the unique eigenvalue of A . It has an algebraic multiplicity of $n - 1$ and an arithmetic multiplicity of n .

2. Show that when A is invertible, its inverse $A^{-1} = I + \alpha uv^T$. Find α .

Answer: A is invertible provided $1 - v^T u \neq 0$. In that case:

$$(I + \alpha uv^T)(I - uv^T) = I + (\alpha - 1)uv^T - \alpha(v^T u)uv^T = I + ((1 - v^T u)\alpha - 1)uv^T.$$

$$\text{Hence } \alpha = \frac{1}{1 - v^T u}.$$

Exercise 2.3 Let $v \in \mathbb{R}^n$, $v \neq 0$ and $a \in \mathbb{R}$, $a \neq 0$. Let:

$$A = I - a vv^T.$$

1. Show that A is symmetric.

Answer: $A^T = I - a(vv^T)^T = I - a vv^T = A$.

2. Show that $\Lambda(A)$, the spectrum of A , consists of 2 distinct eigenvalues. Find the corresponding eigenvectors to each of these 2 eigenvalues.

Answer: As per the previous exercise with $u = av$ then $\lambda_1 = 1$ with $E_1 = \{span\{v\}\}^\perp$ and $\lambda_2 = 1 - a||v||^2$ with $E_2 = span\{v\}$.

3. Find the relation between a and $||v||$ that makes A spd.

Answer: For that purpose one needs $1 - a||v||^2 > 0$, i.e., $\frac{1}{||v||^2} > a$.

Exercise 2.4 Let $F_k = [f_1, f_2, \dots, f_k]$ a subset of orthonormal vectors in \mathbb{R}^n ($F_k^T F_k = I_k$). Prove the identity:

$$I - F_k F_k^T = (I - f_k f_k^T) \dots (I - f_2 f_2^T) (I - f_1 f_1^T).$$

Answer: The proof is done by inductions For $k = 1$, the identity is trivial. Assume the identity true for $k - 1$, i.e.

$$I - F_{k-1} F_{k-1}^T = (I - f_{k-1} f_{k-1}^T) \dots (I - f_2 f_2^T) (I - f_1 f_1^T),$$

Using the external product formula for matrix multiplications, one has:

$$I - F_{k-1} F_{k-1}^T = I - (f_1 f_1^T + f_2 f_2^T + \dots + f_{k-1} f_{k-1}^T)$$

Hence:

$$I - (f_1 f_1^T + f_2 f_2^T + \dots + f_{k-1} f_{k-1}^T) = (I - f_{k-1} f_{k-1}^T) \dots (I - f_2 f_2^T) (I - f_1 f_1^T).$$

Multiplying both sides by $(I - f_k f_k^T)$, one has:

$$\begin{aligned} I - f_k f_k^T - (I - f_k f_k^T)(f_1 f_1^T + f_2 f_2^T + \dots + f_{k-1} f_{k-1}^T) = \dots \\ \dots (I - f_k f_k^T)(I - f_{k-1} f_{k-1}^T) \dots (I - f_2 f_2^T)(I - f_1 f_1^T), \end{aligned}$$

i.e.,:

$$I - (f_1 f_1^T + f_2 f_2^T + \dots + f_{k-1} f_{k-1}^T + f_k f_k^T) = (I - f_k f_k^T) \dots (I - f_2 f_2^T) (I - f_1 f_1^T)$$

and therefore: $I - F_K F_K^T = (I - f_k f_k^T) \dots (I - f_2 f_2^T) (I - f_1 f_1^T)$

Exercise 2.5 Prove, for $U \in \mathbb{C}^{n \times n}$ unitary, that $|\det(U)| = 1$.

Answer: Since $UU^* = I$, then $\det(UU^*) = 1$ and $\det(U)\det(U^*) = 1$. As $\det(U^*) = \overline{\det(U)}$, then $|\det(U)|^2 = 1$ and $|\det(U)| = 1$.

Exercise 2.6 Prove, for $Q \in \mathbb{R}^{n \times n}$ orthogonal, that $\det(Q) = \pm 1$.

Answer: since $QQ^T = I$, then $\det(QQ^T) = 1$ and $\det(Q)\det(Q^T) = 1$. As $\det(Q^T) = \det(Q)$, then $|\det(Q)|^2 = 1$ and therefore: $|\det(Q)| = \pm 1$.

Exercise 2.7 Prove, for $A \in \mathbb{C}^{n \times n}$ hermitian, that $\Lambda(A) \subset \mathbb{R}$.

Answer: We use Schur's decomposition to write:

$$AU = UT \Leftrightarrow T = U^*AU,$$

where U is unitary and T is upper triangular. Hence as A is hermitian, $A = A^*$ and:

$$T^* = U^*A^*U = U^*AU = T.$$

Therefore $T = D = \text{diag}(\{\lambda_1, \dots, \lambda_n\})$ is a real diagonal matrix and if we set:

$$U = [U_1, U_2, \dots, U_n],$$

then: $\forall j = 1, \dots, n$, $AU_j = \lambda_j U_j$, i.e., the j th column of U is the eigenvector of A with $\lambda_j \in \mathbb{R}$ being the associated eigenvalue. Hence, $\Lambda(A) \subset \mathbb{R}$.

Exercise 2.8 Show, directly and through using Schur's decomposition, that for any n by n matrix, $p_A(\lambda)$ is a polynomial of degree exactly n .

Answer: Since $A = UTU^*$ with $UU^* = I$, then:

$$A - \lambda I = U(T - \lambda I)U^* \Rightarrow \det(A - \lambda I) = \det(T - \lambda I).$$

As $T - \lambda I$ is upper triangular, then:

$$p_A(\lambda) = (T_{11} - \lambda)(T_{22} - \lambda) \dots (T_{nn} - \lambda) = (-\lambda)^n + \dots$$

Exercise 2.9 Show if $A \in \mathbb{R}^{n \times n}$ is spd, then $\Lambda(A) \subset \mathbb{R}_+$.

Answer: The inequality $x^T A x > 0$ holds for any $x \neq 0$. If x , $\|x\| = 1$ is an eigenvector corresponding to an eigenvalue λ , then $x^A x = \lambda x^T x = \lambda > 0$.

Exercise 2.10 Show if $A \in \mathbb{R}^{n \times n}$ is spsd, then $\Lambda(A) \subset \mathbb{R}_+ \cup \{0\}$.

Answer: The inequality $x^T A x \geq 0$ holds for any $x \neq 0$. If $x, \|x\| = 1$ is an eigenvector corresponding to an eigenvalue λ , then $x^T A x = \lambda x^T x = \lambda \geq 0$.

Exercise 2.11 Prove that for $A \in \mathbb{R}^{n \times n}$:

$$A \text{ p.d.} \iff \frac{1}{2}(A + A^T) \text{ spd}.$$

Answer: The inequality $x^T A x = x^T A^T x > 0$ holds for any $x \neq 0$. Hence in this case:

$$x^T A x + x^T A^T x = 2x^T \left(\frac{1}{2}(A + A^T) \right) x > 0,$$

i.e., $\frac{1}{2}(A + A^T)$ is spd.

Exercise 2.12 Let $A \in \mathbb{R}^{m \times n}$ be a general rectangular matrix ($m < n$, $m = n$ or $m > n$). Prove that the matrices $AA^T \in \mathbb{R}^{m, m}$ and $A^T A \in \mathbb{R}^{n \times n}$ are both s.p.s.d with $\text{rank}(AA^T) = \text{rank}(A^T A) = \text{rank}(A)$.

Answer: Consider $AA^T : \mathbb{R}^m \rightarrow \mathbb{R}^m$. For $x \in \mathbb{R}^m$, $x^T AA^T x = \|A^T x\|^2 \geq 0$. $\text{rank}(AA^T) = \dim(\text{Range}(AA^T))$ and $\text{rank}(A) = \dim(\text{Range}(A))$. Let $y \in \text{Range}(AA^T)$. There exists $x \in \mathbb{R}^m$ such that $y = AA^T x = A(A^T x)$. Hence $y \in \text{Range}(A)$.

Let now $y \in \text{Range}(A)$. There exists $z \in \mathbb{R}^n$, such that $y = Az$. On the other hand:

$$\mathbb{R}^n = \text{Null}(A) \oplus (\text{Null}(A))^\perp = \text{Null}(A) \oplus \text{Range}(A^\perp).$$

Hence $z = z_1 + z_2$ with $z_1 \in \text{Null}(A)$ and $z_2 \in \text{Range}(A^\perp)$. Therefore:

$$y = Az = Az_2 = AA^T x, \quad x \in \mathbb{R}^m \Rightarrow y \in \text{Range}(AA^T).$$

This implies:

$$\text{Range}(AA^T) = \text{Range}(A)$$

Exercise 2.13 Reformulate Theorem 2.7 to have it apply for any complex matrix in $\mathbb{C}^{m \times n}$.

Answer:

Given a matrix $A \in \mathbb{C}^{m \times n}$ with $r = \text{rank}(A)$ and $p = \min\{m, n\}$, the set of p singular values of A , $\Sigma(A)$, is obtained by taking the *square roots* of the r positive eigenvalues of $\Lambda(AA^*)$ (or $\Lambda(A^*A)$) and completing these by $p - r$ zeros, i.e.,

$$\Sigma(A) = \underbrace{\{\sigma_1 \sigma_2 \dots, \sigma_r\}}_r \underbrace{\{0 \dots 0\}}_{p-r}$$

Exercise 2.14 Show that for $p = \infty$, $\|A\|_\infty = \max_i \sum_j |a_{ij}|$.

Answer:

$$\|A\|_\infty = \max_{\|x\|_\infty=1} \|Ax\|_\infty.$$

Let $y = Ax$, $\|y\|_\infty = \max_i |y_i| = \max_i \left| \sum_{j=1}^n a_{ij} x_j \right| \leq \max_i \sum_{j=1}^n |a_{ij}| \cdot |x_j|$.
 Since $|x_j| \leq \|x\|_\infty = 1$, then $\|Ax\|_\infty \leq \max_i \sum_{j=1}^n |a_{ij}|$. Hence:

$$\|A\|_\infty \leq \max_i \sum_{j=1}^n |a_{ij}|.$$

Let now $i_0 \in \{1, 2, \dots, n\}$ such that $\sum_{j=1}^n |a_{i_0 j}| = \max_i \sum_{j=1}^n |a_{ij}|$. Consider then the vector x_0 such that: $x_{0,j} = \text{sign } a_{i_0,j}$. Then $\|x_0\|_\infty = 1$ and:

$$(Ax_0)_{i_0} = \sum_{j=1}^n |a_{i_0,j}|.$$

Therefore:

$$|(Ax_0)_{i_0}| = \sum_{j=1}^n |a_{i_0,j}| \leq \|A\|_\infty \leq \sum_{j=1}^n |a_{i_0,j}|.$$

Exercise 2.15 Give the proof of Proposition 2.10.

Answer:

If Q is an orthogonal matrix then $\|Qx\|^2 = x^T Q^T Q x = x^T x$ and therefore for $\forall x \neq 0$:

$$\frac{\|Qx\|}{\|x\|} = 1.$$

The same argument works also for a hermitian matrix. **Exercise 2.16** Show that the matrix Frobenius norm $\|A\|_F$ is not subordinate.

Answer: $\|A\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2$. For $A = I$, one has $\|I\|_F^2 = n > 1$ for $n \geq 2$.

2.2 Computer Exercises

Computer Exercise 2.1: Writing the classical Gram-Schmidt program to any set of n vectors in \mathbb{R}^m or \mathbb{C}^m , $n \leq m$. Consider the MATLAB Algorithm 2.2 that takes for its input a set of n column vectors in the form of a non-zero $m \times n$ matrix V and outputs a set of $r \leq n$ orthonormal column vectors in the form of an $m \times r$ matrix F ; a tolerance TOL is used to break the process whenever $r < n$ is reached and the set of column vectors of V are linearly dependent. The output parameter $k \leq n$ would give the order k at which the process would break (in such case $k < n$).

Algorithm 2.2: Classical Gram-Schmidt Orthogonalization Process

```
function [F,r]=GramSchmidtClassical(V,tol)
[m,n]=size(V);
F=zeros(m,n);
r=n;
IV=1:n;
for k=1:n
    if k==1
        z=1;
        w=V(:,k);
    else
        % Project V(:,k) on the span{F(:,1),...,F(:,z-1)}
        e=V(:,k);
        Fl=F(:,1:z-1);
        vc=(e'*Fl)'; % 2n(k-1) flops
        v=Fl*vc;%2n*(k-1)
        w=e-v;%n
    end
    a=norm(w);%2n flops
    if a>tol
        F(:,z)=(1/a)*w;%n flops
        IV(z)=k;z=z+1;
    else
        r=r-1;
    end
end
F=F(:,1:r);
end
```

1. Test the program on a set of matrices obtained using the built-in matrices generators, `magic`, `rand`, `pascal`.

Answer:

```
tol=0.5*10^(-8);A=magic(5); [F r]=GramSchmidtClassical(A,tol)
```

```
F =
```

```
    0.5234    0.5058   -0.6735    0.1215    0.0441
    0.7081   -0.6966    0.0177   -0.0815    0.0800
    0.1231    0.1367    0.3558    0.6307    0.6646
    0.3079    0.1911    0.4122    0.4247   -0.7200
    0.3387    0.4514    0.4996   -0.6328    0.1774
```

```
r =5
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
A=rand(6); [F r]=GramSchmidtClassical(A,tol)
```

```
F =
```

```
    0.4918   -0.2143    0.6130   -0.5675   -0.0852   -0.0839
    0.5468   -0.0638   -0.0596    0.5237    0.0363   -0.6463
    0.0767    0.6389    0.1919   -0.0732    0.7366   -0.0339
    0.5514    0.2397   -0.6743   -0.3550   -0.1153    0.2108
    0.3817   -0.2094    0.1833    0.4751    0.1746    0.7215
    0.0589    0.6637    0.3090    0.2157   -0.6364    0.0949
```

```
r =6
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
>> A=pascal(5); [F r]=GramSchmidtClassical(A,tol)
```

```
F =
```

```
    0.4472   -0.6325    0.5345   -0.3162    0.1195
    0.4472   -0.3162   -0.2673    0.6325   -0.4781
    0.4472         0       -0.5345         0       0.7171
    0.4472    0.3162   -0.2673   -0.6325   -0.4781
    0.4472    0.6325    0.5345    0.3162    0.1195
```

```
r =5
```

2. Test whether Algorithm 2.2 does (or does not) give orthonormal vectors $F(:, 1:r)$ for the cases run in 1.

Answer:

1. For the magic square matrix:

```
B=F'*F
```

```
B =
```

```
    1.0000    0.0000    0.0000   -0.0000    0.0000
    0.0000    1.0000   -0.0000    0.0000   -0.0000
    0.0000   -0.0000    1.0000   -0.0000    0.0000
   -0.0000    0.0000   -0.0000    1.0000    0.0000
    0.0000   -0.0000    0.0000    0.0000    1.0000
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

2. For the random matrix:

B= F'*F

B =

```

    1.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000
    0.0000    1.0000   -0.0000    0.0000    0.0000    0.0000
   -0.0000   -0.0000    1.0000    0.0000    0.0000    0.0000
   -0.0000    0.0000    0.0000    1.0000    0.0000   -0.0000
   -0.0000    0.0000    0.0000    0.0000    1.0000   -0.0000
    0.0000    0.0000    0.0000   -0.0000   -0.0000    1.0000

```

%%%

3. For the Pascal matrix:

B= F'*F

B =

```

    1.0000    0.0000    0.0000   -0.0000    0.0000
    0.0000    1.0000   -0.0000    0.0000   -0.0000
    0.0000   -0.0000    1.0000   -0.0000    0.0000
   -0.0000    0.0000   -0.0000    1.0000    0.0000
    0.0000   -0.0000    0.0000    0.0000    1.0000

```

3. Analyze the number of flops as a function of m and n .

Answer: The number of flops is:

$$\sum_{k=1}^n 4n(k-1) + 4n = \sum_{k=1}^n 4nk = 2n^2(n+1) = O(2n^3).$$

4. Consider the case when the input matrices are obtained by the function `rand(n)` for $n = 2^k$, $k = 5, \dots, 10$. Check the execution time $T(n)$ using `tic toc` and the number of flops $f(n)$.

Answer:

Algorithm is:

`A=rand(2^k);tic ; [F r]=GramSchmidtClassical(A,tol); toc`

k	n	T(n)	f(n)	T(n)/f(n)
5	32	2.154E-03	63616	3.386E-08
6	64	1.300E-01	516352	2.517E-07
7	128	1.690E-02	4162048	4.060E-09
8	256	2.249E-01	33424384	6.728E-09
9	512	8.103E-01	267913216	3.024E-09
10	1024	5.271E+00	2145390592	2.457E-09

5. Extend Algorithm 2.2 to the case when the matrix V is complex, thus producing a unitary matrix F .

Answer: The algorithm does not change even if the matrix is complex because $\mathbf{x}^*\mathbf{y}$ includes the complex case of an inner product.

6. Modify this algorithm so as to use identity (2.8) obtaining therefore the Modified Gram Schmidt algorithm. Run this algorithm on the cases considered in 1 and test whether one obtains (or does not obtain) orthonormal vectors $F(:, 1:r)$.

Answer:

```
function [ F,r ] = ModifiedGramSchmidt( V,tol )
[m,n]=size(V);F=zeros(m,n);
r=n;IV=1:n;
for k=1:n
    if k==1
        z=1;w=V(:,k);
    else
        % Project V(:,k) on the span{F(:,1),...,F(:,z-1)}
        e=V(:,k); w=e;
        for l=1:k
            f=F(:,l);w=w-(f'*w)*f;
        end
        a=norm(w);%2n flops
        if a>tol
            F(:,z)=(1/a)*w;%n flops
            IV(z)=k;z=z+1;
        else
            r=r-1;
        end
    end
end
F=F(:,1:r);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Testing:
tol=0.5*10^(-8)
1. A=magic(5); [F,r]=ModifiedGramSchmidt(A,tol)
F =
    0.5234    0.5058   -0.6735    0.1215    0.0441
    0.7081   -0.6966    0.0177   -0.0815    0.0800
    0.1231    0.1367    0.3558    0.6307    0.6646
    0.3079    0.1911    0.4122    0.4247   -0.7200
    0.3387    0.4514    0.4996   -0.6328    0.1774
r =5
```

Computer Exercise 2.2: Algorithm 2.3 is a simulation of the proof of Theorem 2.4. It uses the built-in MATLAB `eigs` function and also the Classical Gram Schmidt Algorithm 2.2. Test Algorithm 2.3 for the cases of matrices generated by the built-in functions `rand`, `magic`, `pascal`.

Algorithm 2.3

```
function [F,T] = myschur(A)
% Input: a square n by n matrix
% Output: An upper triangular matrix T
%         A unitary matrix F
% The procedure uses the Matlab function eigs to seek
% a square matrix the largest eigenvalue of
n = length(A); d=zeros(n,1); % d is used to store the eigenvalues of A
T=zeros(n); k=1;
while k<=n-1
    [v,lambda] = eigs(A,1,'lm'); d(k)=lambda; E = [v eye(n-k+1)];
    F1 = GramSchmidtClassical(E,0.5*10^(-5));
    A1 = F1'*A*F1; T(k:n,k:n)=A1;
    % Updating the matrix F
    if k==1
        F=F1;
    else
        F(:,k:n)=F(:,k:n)*F1; T(1:k-1,k:n)=T(1:k-1,k:n)*F1;
    end
    % Update matrix A
    A=A1(2:n-k+1,2:n-k+1); k=k+1;
end
```

Answer:

```
A=rand(4); [F,T] = myschur(A)
F =
    -0.4001         -0.8989    -0.0264 - 0.1053i    0.1416 + 0.0000i
    -0.7340         0.2007     0.2004 + 0.3748i   -0.4838 - 0.0792i
    -0.2968         0.2459     0.4887 - 0.3387i    0.5621 - 0.4267i
    -0.4616         0.3019     0.6098 - 0.2869i    0.2853 + 0.4002i
T =
    1.9021         0.0721    -0.1655 + 0.2306i   -0.3516 + 0.1661i
    0.0000         0.7188     0.0487 + 0.0901i   -0.1163 - 0.0194i
    0.0000         0.0000    -0.0596 - 0.3019i    0.1953 - 0.1125i
    0.0000         0.0000     0.0000 + 0.0000i   -0.0596 + 0.3019i
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
A=magic(5); [F,T] = myschur(A)
```

```
F =
```

```
-0.4001          -0.8989          -0.0264 - 0.1053i    0.1416 + 0.0000i
-0.7340          0.2007          0.2004 + 0.3748i   -0.4838 - 0.0792i
-0.2968          0.2459          0.4887 - 0.3387i    0.5621 - 0.4267i
-0.4616          0.3019          -0.6098 - 0.2869i    0.2853 + 0.4002i
```

```
T =
```

```
1.9021          0.0721          -0.1655 + 0.2306i   -0.3516 + 0.1661i
0.0000          0.7188          0.0487 + 0.0901i   -0.1163 - 0.0194i
0.0000          0.0000          -0.0596 - 0.3019i    0.1953 - 0.1125i
0.0000          0.0000          0.0000 - 0.0000i    -0.0596 + 0.3019i
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
A=pascal(4); [F,T] = myschur(A)
```

```
F =
```

```
-0.4001          -0.8989          -0.0264 - 0.1053i    0.1416 + 0.0000i
-0.7340          0.2007          0.2004 + 0.3748i   -0.4838 - 0.0792i
-0.2968          0.2459          0.4887 - 0.3387i    0.5621 - 0.4267i
-0.4616          0.3019          -0.6098 - 0.2869i    0.2853 + 0.4002i
```

```
T =
```

```
1.9021          0.0721          -0.1655 + 0.2306i   -0.3516 + 0.1661i
-0.0000          0.7188          0.0487 + 0.0901i   -0.1163 - 0.0194i
-0.0000          -0.0000          -0.0596 - 0.3019i    0.1953 - 0.1125i
0.0000          0.0000          0 - 0.0000i          -0.0596 + 0.3019i
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Computer Exercise 2.3: In the following commands:

```
[U,S,V]=svd(A);
[Q,D]=eig(A*A');
[Q1,D1]=eig(A'*A);
```

Find the relationships between the matrices Q, Q1, D, D1, U, S, V. Justify your answer.

Answer:

```
A=(fix(10*rand(5,4)))
```

```
A =
```

```
8      8      6      6
9      9      7      1
0      0      5      2
5      9      8      1
0      0      8      0
```

```
[U,S,V]=svd(A)
```

U =

-0.5616	-0.2662	0.7210	-0.0516	0.3020
-0.5910	-0.2092	-0.3805	0.6246	-0.2685
-0.1355	0.4792	0.3903	-0.0617	-0.7719
-0.5297	0.1040	-0.4262	-0.7260	0.0000
-0.1907	0.8030	-0.0379	0.2764	0.4908

S =

24.1871	0	0	0
0	8.1028	0	0
0	0	4.5574	0
0	0	0	2.1351
0	0	0	0

V =

-0.5152	-0.4310	0.0467	0.7393
-0.6028	-0.3797	-0.3274	-0.6207
-0.5766	0.8134	-0.0216	0.0738
-0.1969	-0.0918	0.9435	-0.2503

[Q,D]=eig(A*A')

Q =

-0.3020	0.0516	0.7210	-0.2662	0.5616
0.2685	-0.6246	-0.3805	-0.2092	0.5910
0.7719	0.0617	0.3903	0.4792	0.1355
-0.0000	0.7260	-0.4262	0.1040	0.5297
-0.4908	-0.2764	-0.0379	0.8030	0.1907

D =

0.0000	0	0	0	0
0	4.5587	0	0	0
0	0	20.7702	0	0
0	0	0	65.6556	0
0	0	0	0	585.0155

[Q1,D1]=eig(A'*A)

Q1 =

0.7393	-0.0467	0.4310	0.5152
-0.6207	0.3274	0.3797	0.6028
0.0738	0.0216	-0.8134	0.5766
-0.2503	-0.9435	0.0918	0.1969

D1 =

4.5587	0	0	0
0	20.7702	0	0
0	0	65.6556	0
0	0	0	585.0155

One checks first that the diagonal elements of S are the square roots of the positive diagonal elements of D . Furthermore, one has $AV = US$ and $AA^TQ =$

QD . Letting $D_r = D(1:4, 1:4)$ and $Q_r = Q(:, 1:4)$ then:

$$AA^T Q_r = Q_r D_r \iff AA^T Q_r D_r^{-1/2} = Q_r D_r^{1/2}.$$

Letting $V_r = A^T Q_r D_r^{-1/2}$, then $AV_r = Q_r D_r^{1/2}$. One checks that:

$$V_r = V(:, 4:-1:1) \text{ and } Q_r = U(:, 4:-1:1)$$

Computer Exercise 2.4:

1. Implement the proof of Theorem 2.7 by completing MATLAB Algorithm 2.4 that takes as input a rectangular matrix A and outputs two orthogonal matrices $U \in \mathbb{R}^{m \times m}$, and $V \in \mathbb{R}^{n \times n}$ and a diagonal matrix $S \in \mathbb{R}^{m \times n}$ consisting of the singular values of A , such that $AV = US$. The procedure uses MATLAB `schur(X, 'real')` and the already written Algorithm 2.2
2. Test Algorithm ?? by generating a rectangular matrix using the function `rand` and compare the results with those obtained using MATLAB `svd`.

Answer 1:

```
function [ U S V ] = mysvd( A )
[m n]=size(A);
[U T]=schur(A*A','real'); %AA'U=UT
S=zeros(m,n);r=rank(T);%r <= m
s=sqrt(diag(T)); %s=sort(s,'descend')
for i=1:r
    S(i,i)=s(i);
end
Ur=U(:,1:r);Sr=S(1:r,1:r);Vr=(A')*Ur*(inv(Sr));
V=zeros(n,n+r);
for k=1:r
    V(:,k)=Vr(:,k);
end
I=eye(n);
for k=r+1:n+r
    V(:,k)=I(:,k-r);
end
V=GramSchmidtClassical(V,0.5*10^(-6));
end
```

Test for the case of one random matrix:

```
A=rand(4,5);
[ U S V ] = mysvd( A );
```

```

U =
    0.4627    0.4479    0.3299    0.6902
   -0.6081    0.3893   -0.5511    0.4184
    0.2607   -0.7166   -0.4209    0.4914
   -0.5901   -0.3665    0.6406    0.3273

S =
    0.3001         0         0         0         0
         0    0.6581         0         0         0
         0         0    0.7045         0         0
         0         0         0    2.3228         0

V =
   -0.1958    0.1349    0.5616    0.3194    0.7253
    0.8585    0.0927    0.0370    0.5019   -0.0351
   -0.2935   -0.2780    0.4857    0.4747   -0.6126
   -0.3672    0.5008   -0.5341    0.5728   -0.0310
   -0.0612   -0.8031   -0.4026    0.3044    0.3106

```

For the MATLAB svd:

```

[ U S V ] = svd( A )
U =
    0.6902    0.3299   -0.4479    0.4627
    0.4184   -0.5511   -0.3893   -0.6081
    0.4914   -0.4209    0.7166    0.2607
    0.3273    0.6406    0.3665   -0.5901

S =
    2.3228         0         0         0         0
         0    0.7045         0         0         0
         0         0    0.6581         0         0
         0         0         0    0.3001         0

V =
    0.3194    0.5616   -0.1349   -0.1958   -0.7253
    0.5019    0.0370   -0.0927    0.8585    0.0351
    0.4747    0.4857    0.2780   -0.2935    0.6126
    0.5728   -0.5341   -0.5008   -0.3672    0.0310
    0.3044   -0.4026    0.8031   -0.0612   -0.3106

```

Therefore, we see clearly that `mysvd` gives another singular value decomposition, the singular values being obtained in a reverse order than those yielded by MATLAB `svd`.