上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Lecture 13

Eigenvalues and Singular Values

Ye Ding (丁烨)

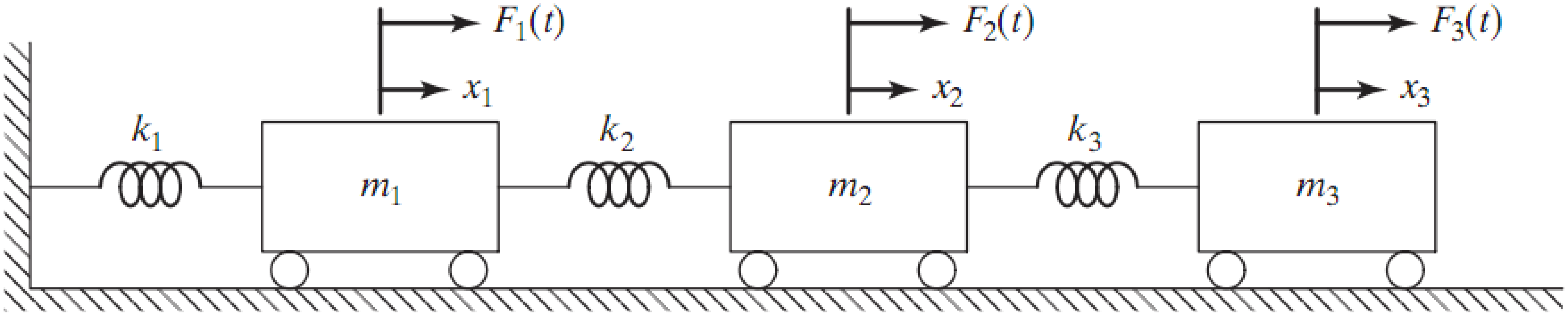
Email: y.ding@sjtu.edu.cn

School of Mechanical Engineering

Shanghai Jiao Tong University

Eigenvalues and Singular Values

Motivation: Example from Vibration Theory



The equations of motion of the system:

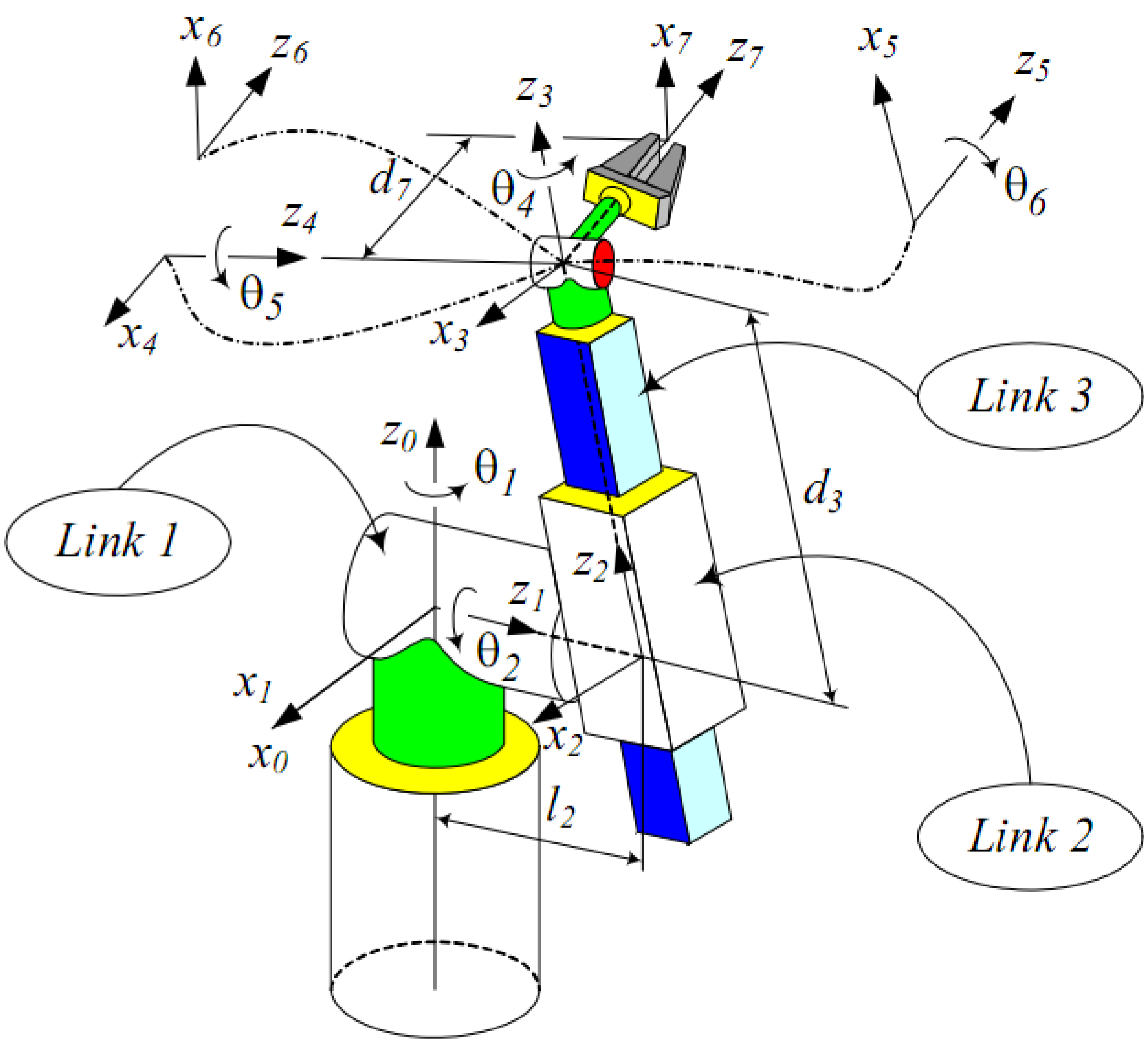
$$[m]\ddot{\bar{x}} + [k]\bar{x} = \bar{F}$$

$$[m] = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}, \quad [k] = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}, \quad \bar{F} = \begin{Bmatrix} F_1(t) \\ F_2(t) \\ F_3(t) \end{Bmatrix}$$

- **How to find the natural frequencies and mode shapes?**

Eigenvalues and Singular Values

- **Motivation: Example from Robotics**



微分运动学:

$$v_e = J(q)\dot{q}$$

$$J = U\Sigma V^T$$

Eigenvalues and Singular Values

- **References for Eigenvalues and Singular Values**

[1] Timothy Sauer, Numerical analysis (2nd ed.), Pearson Education, 2012. Chapter 12

[2] Cleve Moler, Numerical Computing with MATLAB, Society for Industrial and Applied Mathematics, 2004. Chapter 10

[3] Gene H. Golub, Charles F. Van Loan, Matrix Computations (4th ed.), Johns Hopkins University Press, 2013. Chapter 7

[4] 李庆扬等，数值分析（第5版），清华大学出版社，2008.　第8章

Eigenvalues and Singular Values

1. **Review of Eigenvalues and Eigenvectors**
2. **Approximating Eigenvalues**
   - **Power Iteration Methods**
   - **QR Algorithm**
3. **Singular Value Decomposition**
   - **Applications of the SVD**

Review of Linear Algebra

Basic Theory

Let A be an $m \times m$ matrix and x a nonzero m-dimensional real or complex vector. If

$$Ax = \lambda x \qquad (1)$$

for some real or complex number λ, then λ is called an eigenvalue of A and x is the corresponding eigenvector.

- **Eigenvalues are the roots λ of the characteristic polynomial $\det(A - \lambda I)$.**
- **If λ is an eigenvalue of A, then any nonzero vector in the nullspace of $A - \lambda I$ is an eigenvector corresponding to λ.**

Review of Linear Algebra

Basic Theory

The determinant in det(*A*- λI) can be written in the form

$$\begin{vmatrix} a_{11}-\lambda & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22}-\lambda & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm}-\lambda \end{vmatrix} = 0 \qquad (2)$$

We can derive the eigenvalues λ according to Eq. (2). Then, each root λ can be substituted into *Ax* = λ*x* to obtain the eigenvector *x*.

We now state the following definitions and theorems necessary for the study of eigenvalues.

Review of Linear Algebra

- **Definition** **The $m \times m$ matrices A_1 and A_2 are similar, denoted $A_1 \sim A_2$, if there exists an invertible $m \times m$ matrix S such that**

$$A_1 = SA_2S^{-1}$$

- **Similar matrices have identical eigenvalues, because their characteristic polynomials are identical.**
- **Assume that A is a symmetric $m \times m$ matrix with real entries. Then the eigenvalues are real numbers, and the set of unit eigenvectors of A is an orthonormal set $\{w_1,\ldots, w_m\}$ that forms a basis of $\mathbb{R}^m$.**

Review of Linear Algebra

- **Example 1. Find the eigenpairs for the matrix**

$$A=\begin{bmatrix}2 & -3 & 6\\ 0 & 3 & -4\\ 0 & 2 & -3\end{bmatrix}$$

The characteristic equation det(A- λI) = 0 is

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

The roots of the equation are the three eigenvalues λ_1 =1, λ_2 =2, and λ_3 =-1.

To find the eigenvector x_1 corresponding to λ_1, we substitute λ_1 =1 into $Ax = \lambda x$ to get the system of equations

$$\begin{aligned}x_1-3x_2+6x_3&=0\\ 2x_2-4x_3&=0\\ 2x_2-4x_3&=0\end{aligned}$$

Review of Linear Algebra

We can obtain the first eigenvector $x_1=(0, 2, 1)^T$. Hence, the first eigenpair is

$$\lambda_1=1 \text{ and } x_1=(0, 2, 1)^T$$

Similarly, we can derive the second eigenvector $x_2=(1, 0, 0)^T$, and the second eigenpair is

$$\lambda_2=2 \text{ and } x_2=(1, 0, 0)^T$$

The third eigenpair is

$$\lambda_3=-1 \text{ and } x_3=(-1, 1, 1)^T$$

Disadvantage: when the dimension m is large it is difficult to determine the zeros of $\det(A - \lambda I)=0$ and also to find the nonzero solution of the homogeneous linear system $(A - \lambda I)x=0$.

Solution: Power Iteration Method.

Eigenvalues and Singular Values

1. Review of Eigenvalues and Eigenvectors
2. Approximating Eigenvalues
   - Power Iteration Methods
   - QR Algorithm
3. Singular Value Decomposition
   - Applications of the SVD

Approximating Eigenvalues

The Power Iteration Method

```
function [lam,u]=PowerIteration(A,x,k)
% Input: matrix A, initial (nonzero) vector x, number of steps k
% Output: dominant eigenvalue lam, eigenvector u
for j = 1:k
    u = x / norm(x);  % normalize vector
    x = A * u;        % power step
    lam = u' * x;     % Rayleigh quotient
end
u = x / norm(x);
```

Approximating Eigenvalues

The Power Iteration Method: Example 2

The matrix

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

has an eigenvalue of 4 with eigenvector $[1,1]^T$, and an eigenvalue that is smaller in magnitude, −1, with associated eigenvector $[-3,2]^T$.

Question: let x0 = [-5;5]; k = 10;

Result of [lam,u] = PowerIteration(A,x0,k) ?

Approximating Eigenvalues

The Power Iteration Method: Example 2

The matrix

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

let x_0 = [-5, 5]^T

$$x_1 = Ax_0 = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} -5 \\ 5 \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \end{bmatrix}$$

$$x_2 = A^2x_0 = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 10 \\ 0 \end{bmatrix} = \begin{bmatrix} 10 \\ 20 \end{bmatrix}$$

$$x_3 = A^3x_0 = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 10 \\ 20 \end{bmatrix} = \begin{bmatrix} 70 \\ 60 \end{bmatrix}$$

$$x_4 = A^4x_0 = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 70 \\ 60 \end{bmatrix} = \begin{bmatrix} 250 \\ 260 \end{bmatrix} = 260\begin{bmatrix} \frac{25}{26} \\ 1 \end{bmatrix}$$

Approximating Eigenvalues

The Power Iteration Method: Example 2

By expressing x_0 as a linear combination of the eigenvectors:

$$x_0 = \begin{bmatrix} -5 \\ 5 \end{bmatrix} = 1\begin{bmatrix} 1 \\ 1 \end{bmatrix} + 2\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

Reviewing the calculation in this light:

$$x_1 = Ax_0 = 4\begin{bmatrix} 1 \\ 1 \end{bmatrix} - 2\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$x_2 = A^2x_0 = 4^2\begin{bmatrix} 1 \\ 1 \end{bmatrix} + 2\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$x_3 = A^3x_0 = 4^3\begin{bmatrix} 1 \\ 1 \end{bmatrix} - 2\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$x_4 = A^4x_0 = 4^4\begin{bmatrix} 1 \\ 1 \end{bmatrix} + 2\begin{bmatrix} -3 \\ 2 \end{bmatrix} = 256\begin{bmatrix} 1 \\ 1 \end{bmatrix} + 2\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

Approximating Eigenvalues

- **The Power Iteration Method**

Let A be an $m \times m$ matrix. A dominant eigenvalue of A is an eigenvalue λ whose magnitude is greater than all other eigenvalues of A. If it exists, an eigenvector associated to λ is called a dominant eigenvector.

In Example 2, the matrix $$A = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}$$

has the dominant eigenvalue of 4 with eigenvector $[1,1]^T$.

Approximating Eigenvalues

The Power Iteration Method

Next question: as the steps deliver improved approximate eigenvectors, how do we find approximate eigenvalues?

Solution: least squares

Consider the eigenvalue equation

$$x\lambda = Ax$$

The normal equations say that the least squares answer is the solution of

$$x^T x\lambda = x^T Ax \quad \Rightarrow \quad \lambda = \frac{x^T Ax}{x^T x}$$

Rayleigh quotient

Approximating Eigenvalues

- **The Power Iteration Method: Pseudo-codes**

Power Iteration

Given initial vector x_0.

for $j = 1, 2, 3, \ldots$

$$u_{j-1} = x_{j-1}/||x_{j-1}||_2$$
$$x_j = Au_{j-1}$$
$$\lambda_j = u_{j-1}^T A u_{j-1}$$

end

$$u_j = x_j/||x_j||_2$$

Approximating Eigenvalues

- The Power Iteration Method: theory

THEOREM12.2 of Ref.[1]

Let A be an $m \times m$ matrix with real eigenvalues $\lambda_1,\ldots,\lambda_m$ satisfying $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_m|$. Assume that the eigenvectors of A span R^m. For almost every initial vector, Power Iteration converges linearly to an eigenvector associated to λ_1 with convergence rate constant $S = |\lambda_2/\lambda_1|$

Approximating Eigenvalues

Disadvantage of the Power Iteration Method

- **Power Iteration is limited to locating the eigenvalue of largest magnitude (absolute value).**
- **Its rate of convergence is slow when the dominance ratio**

$$S = \left| \frac{\lambda_2}{\lambda_1} \right|$$

of the eigenvalues with the two largest magnitude is close to one.

Approximating Eigenvalues

The Inverse Power Iteration Method

Let the eigenvalues of the $m \times m$ matrix A be denoted by λ_1, λ_2, ...,λ_m.

(a) The eigenvalues of the inverse matrix A^{-1} are λ_1^{-1}, λ_2^{-1},...,λ_m^{-1}, assuming that the inverse exists.

The eigenvectors are the same as those of A.

(b) The eigenvalues of the shifted matrix $A - sI$ are $\lambda_1 - s$, $\lambda_2 - s$,..., $\lambda_m - s$ and the eigenvectors are the same as those of A.

Approximating Eigenvalues

- **The Inverse Power Iteration Method: Pseudo-codes**

Inverse Power Iteration

Given initial vector x_0 and shift s

for $j = 1, 2, 3, \ldots$

$u_{j-1} = x_{j-1}/||x_{j-1}||_2$

Solve $(A - sI)x_j = u_{j-1}$

$\lambda_j = u_{j-1}^T x_j$

end

$u_j = x_j/||x_j||_2$

Approximating Eigenvalues

The Inverse Power Iteration Method: Example 2

The matrix

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

has an eigenvalue of 4 with eigenvector $[1,1]^T$, and an eigenvalue that is smaller in magnitude, −1, with associated eigenvector $[-3,2]^T$.

```
x0 = [-5;5]; s = 0; k = 10;
[lam,u] = InversePowerIteration(A,x0,s,k)
```

How about the result of s = 3?

上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

Eigenvalues and Singular Values

1. Review of Eigenvalues and Eigenvectors
2. Approximating Eigenvalues
   - Power Iteration Methods
   - QR Algorithm
     - Simultaneous Iteration
     - Real Schur form
     - Upper Hessenberg form
3. Singular Value Decomposition
   - Applications of the SVD

Approximating Eigenvalues

QR Algorithm: Motivation

How to develop methods for finding all eigenvalues at once?

We begin with a method that works for symmetric matrices.

- **their eigenvalues are real**
- **their unit eigenvectors form an orthonormal basis of R^m**

Key idea: applying Power Iteration with *m* vectors in parallel, keeping the vectors orthogonal to one another.

Key point: re-orthogonalization

Approximating Eigenvalues

QR Algorithm: Simultaneous Iteration

If the elementary basis vectors are used as initial vectors, then the first step of Power Iteration followed by re-orthogonalization using QR is

$$\left[A\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix} \middle| A\begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix} \middle| \cdots \middle| A\begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix} \right] = \left[\bar{q}_1^1 | \cdots | \bar{q}_m^1\right] \begin{bmatrix} r_{11}^1 & r_{12}^1 & \cdots & r_{1m}^1 \\ & r_{22}^1 & & \vdots \\ & & \ddots & \vdots \\ & & & r_{mm}^1 \end{bmatrix}$$

The $\bar{q}_i^1, i = 1, \ldots, m$ are the new orthogonal set of unit vectors in the Power Iteration process.

Approximating Eigenvalues

- **QR Algorithm: Simultaneous Iteration**

Next, we repeat the step:

$$
\begin{aligned}
A\overline{Q}_1 &= \left[A\overline{q}_1^1 | A\overline{q}_2^1 | \cdots | A\overline{q}_m^1\right] \\
&= \left[\overline{q}_1^2 | \overline{q}_2^2 | \cdots | \overline{q}_m^2\right] \begin{bmatrix} r_{11}^2 & r_{12}^2 & \cdots & r_{1m}^2 \\ & r_{22}^2 & & \vdots \\ & & \ddots & \vdots \\ & & & r_{mm}^2 \end{bmatrix} \\
&= \overline{Q}_2 R_2.
\end{aligned}
$$

Approximating Eigenvalues

- **QR Algorithm: Simultaneous Iteration**

Normalized Simultaneous Iteration (NSI)

for a symmetric matrix

$$
\begin{array}{ll}
\text{Set} & \overline{Q}_0 = I \\
\textbf{for} & j = 1, 2, 3, \ldots \\
 & A\overline{Q}_j = \overline{Q}_{j+1} R_{j+1} \\
\textbf{end} &
\end{array}
$$

At the *j*-th step, the columns of $\bar{Q}_j$ are approximations to the eigenvectors of *A*, and the diagonal elements

$$r_{11}^{j}, r_{22}^{j}, \ldots r_{mm}^{j}$$

are approximations to the eigenvalues.

Approximating Eigenvalues

Unshifted QR Algorithm:

- **NSI**

$$\overline{Q}_0 = I$$

$$A\overline{Q}_0 = \overline{Q}_1 R_1$$

$$A\overline{Q}_1 = \overline{Q}_2 R_2$$

$$A\overline{Q}_2 = \overline{Q}_3 R_3$$

- **Unshifted QR algorithm**

$$Q_0 = I$$

$$A_0 \equiv AQ_0 = Q_1 R_1'$$

$$A_1 \equiv R_1' Q_1 = Q_2 R_2'$$

$$A_2 \equiv R_2' Q_2 = Q_3 R_3'$$

$$Q_1 = \overline{Q}_1 \text{ and } R_1 = R_1'$$

$$\overline{Q}_2 = Q_1 Q_2 \text{ and } R_2 = R_2'$$

$$\overline{Q}_2 R_2 = A\overline{Q}_1 = Q_1 R_1' \overline{Q}_1 = Q_1 R_1' Q_1 = Q_1 Q_2 R_2'$$

$$\overline{Q}_j = Q_1 \cdots Q_j \text{ and } R_j = R_j'$$

Approximating Eigenvalues

Unshifted QR Algorithm:

- **NSI**

$$\overline{Q}_0 = I$$

$$A\overline{Q}_0 = \overline{Q}_1 R_1$$
$$A\overline{Q}_1 = \overline{Q}_2 R_2$$
$$A\overline{Q}_2 = \overline{Q}_3 R_3$$
$$\vdots$$

- **Unshifted QR algorithm**

$$Q_0 = I$$

$$A_0 \equiv A Q_0 = Q_1 R_1'$$
$$A_1 \equiv R_1' Q_1 = Q_2 R_2'$$
$$A_2 \equiv R_2' Q_2 = Q_3 R_3'$$
$$\vdots$$

Remark: the unshifted QR algorithm does the same calculations as Normalized Simultaneous Iteration.

$$\overline{Q}_j = Q_1 \cdots Q_j \text{ and } R_j = R_j'$$

Approximating Eigenvalues

- **Unshifted QR Algorithm:**

```
function [lam,Qbar]=UnshiftedQR(A,k)
% Computes eigenvalues/vectors of symmetric matrix
% Input: matrix A, number of steps k
% Output: eigenvalues lam and eigenvector matrix Qbar
[m,n]=size(A);
Q=eye(m,m);
Qbar=Q;
R=A;
for j=1:k
    [Q,R] = qr(R*Q); % QR factorization
    Qbar = Qbar*Q; % accumulate Q's
end
lam = diag(R*Q); % diagonal converges to eigenvalues
```

Approximating Eigenvalues

- **Unshifted QR Algorithm: Example 3**

For the matrix $A = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$, we use the algorithm:

A1 =

3.6000	0.8602	0.0000
0.8602	3.1297	-0.8974
0	-0.8974	2.2703

A2 =

3.9635	0.6931	-0.0000
0.6931	3.2223	0.5724
0	0.5724	1.8142

A3 =

4.1767	0.5354	0.0000
0.5354	3.1727	-0.3093
0	-0.3093	1.6506

A15 =

4.4142	0.0061	0.0000
0.0061	3.0000	-0.0001
0	-0.0001	1.5858

上海交通大学 SHANGHAI JIAO TONG UNIVERSITY

Approximating Eigenvalues

Unshifted QR Algorithm: Theory

THEOREM12.4 of Ref.[1]

Assume that A is a symmetric m $\times$ m matrix with eigenvalues λ_i satisfying $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_m|$.

The unshifted QR algorithm converges linearly to the eigenvectors and eigenvalues of A.

As $j \to \infty$, A_j converges to a diagonal matrix containing the eigenvalues on the main diagonal and $\overline{Q}_j = Q_1 \cdots Q_j$ converges to an orthogonal matrix whose columns are the eigenvectors.

Approximating Eigenvalues

- **Unshifted QR Algorithm:**

The eigenvalues of the symmetric matrix

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

are *1* and *−1* , both of magnitude 1.

The unshifted QR algorithm fails!

Approximating Eigenvalues

- **QR Algorithm: Real Schur form**

The eigenvalues of the nonsymmetric matrix

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

are *i* and −*i* , both of complex magnitude 1.

The unshifted QR algorithm fails!

Approximating Eigenvalues

QR Algorithm: Real Schur form

A matrix T has **real Schur form** if it is upper triangular, except possibly for 2 $\times$ 2 blocks on the main diagonal.

For example, a matrix of the form (**quasi-upper-triangular**)

$$\begin{bmatrix} x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & x & x & x \\ & & & & x \end{bmatrix}$$

has real Schur form.

Approximating Eigenvalues

- **QR Algorithm: a theorem on Real Schur form**

THEOREM12.6 of Ref. [1]

Let A be a square matrix with real entries. Then there exists an orthogonal matrix Q and a matrix T in real Schur form such that

$$A = Q^T T Q$$

- **First we will install the Inverse Power Iteration idea with shifts and add the idea of deflation to develop the shifted QR algorithm.**
- **Then we will develop an improved version that allows for complex eigenvalues.**

Approximating Eigenvalues

- **Shifted QR Algorithm: Real Schur form**

The idea of the shifted version: each step consists of applying the shift, completing a QR factorization, and then taking the shift back, i.e.,

$$A_0 - sI = Q_1 R_1$$
$$A_1 = R_1 Q_1 + sI$$

Noting that

$$\begin{aligned} A_1 - sI &= R_1 Q_1 \\ &= Q_1^T (A_0 - sI) Q_1 \\ &= Q_1^T A_0 Q_1 - sI \end{aligned}$$

implies that A_1 is similar to A_0 and so has the same eigenvalues.

Approximating Eigenvalues

- **Shifted QR Algorithm: Real Schur form**

How to choose the shift *s*?

Key point: use the concept of deflation for eigenvalue calculations.

We will choose the shift to be the bottom right entry of the matrix A_k . This will cause the iteration, as it converges to real Schur form, to move the bottom row to a row of zeros, except for the bottom right entry. After this entry has converged to an eigenvalue, we deflate the matrix by eliminating the last row and column.

Then we proceed to find the rest of the eigenvalues.

Approximating Eigenvalues

- **Shifted QR Algorithm: Preliminary Version**

```
function lam = ShiftedQR0(A)
% Input: matrix A
% Output: eigenvalues lam
tol=1e-14; m=size(A,1); lam=zeros(m,1); n=m;
while n>1
    while max(abs(A(n,1:n-1)))>tol
        mu=A(n,n);                    % define shift mu
        [q,r]=qr(A-mu*eye(n));
        A=r*q+mu*eye(n);
    end
    lam(n)=A(n,n);                    % declare eigenvalue
    n=n-1;                            % decrement n
    A=A(1:n,1:n);                     % deflate
end
lam(1)=A(1,1);                        % 1x1 matrix remains
```

Approximating Eigenvalues

- **Shifted QR Algorithm: Real Schur form**

How to allow for the calculation of complex eigenvalues?

Key point: we must allow for the existence of 2×2 blocks on the diagonal of the real Schur form.

The improved version of the shifted QR algorithm tries to iterate the matrix to a 1×1 diagonal block in the bottom right corner;

if it fails (after a user-specified number of tries), it declares a 2×2 block, finds the pair of eigenvalues, and then deflates by 2.

Approximating Eigenvalues

Shifted QR Algorithm: General Version

```
function lam = ShiftedQR(A)
tol=1e-14; kounttol=500; m=size(A,1); lam=zeros(m,1);n=m;
while n>1
    kount=0;
    while max(abs(A(n,1:n-1)))>tol & kount<kounttol
        kount=kount+1;    % keep track of number of qr's
        mu=A(n,n);         % shift is mu
        [q,r]=qr(A-mu*eye(n));
        A=r*q+mu*eye(n);
    end
    if kount<kounttol      % have isolated 1x1 block
        lam(n)=A(n,n);      % declare eigenvalue
        n=n-1;  A=A(1:n,1:n);    % deflate by 1
    else                     % have isolated 2x2 block
        disc=(A(n-1,n-1)-A(n,n))^2+4*A(n,n-1)*A(n-1,n);
        lam(n)=(A(n-1,n-1)+A(n,n)+sqrt(disc))/2;
        lam(n-1)=(A(n-1,n-1)+A(n,n)-sqrt(disc))/2;
        n=n-2;  A=A(1:n,1:n);    % deflate by 2
    end
end
if n>0;  lam(1)=A(1,1);  end % only a 1x1 block remains
```

Approximating Eigenvalues

- **QR Algorithm: Upper Hessenberg form**

The eigenvalues of the matrix

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

with a repeated complex eigenvalue.

The shifted QR algorithm fails!

Approximating Eigenvalues

QR Algorithm: Upper Hessenberg form

The $m \times n$ matrix A is in upper Hessenberg form if $a_{ij} = 0$ for $i > j + 1$.

A matrix of the form

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & & x & x \end{bmatrix}$$

is upper Hessenberg.

Let A be a square matrix. There exists an orthogonal matrix Q such that $A = QBQ^T$ and B is in upper Hessenberg form.

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Approximating Eigenvalues

- **QR Algorithm: Upper Hessenberg form**

Key idea: using Householder reflectors on the left and right of the matrix

$$H_1 A = \left[\begin{array}{c:cccc} 1 & 0 & 0 & 0 & 0 \\ \hdashline 0 & & & & \\ 0 & & & & \\ 0 & & \hat{H}_1 & & \\ 0 & & & & \end{array}\right] \left[\begin{array}{c:cccc} x & x & x & x & x \\ \hdashline x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{array}\right] = \left[\begin{array}{c:cccc} x & x & x & x & x \\ \hdashline x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{array}\right]$$

$\hat{H}_1$ **: the Householder reflector that moves x to**

$$(\pm||x||, 0, \ldots, 0)$$

we should choose the sign as −sign(x_1)

Approximating Eigenvalues

- **QR Algorithm: Upper Hessenberg form**

Recall Householder reflector: $H_1^{-1} = H_1^T = H_1$

$$H_1 A H_1 = \left[\begin{array}{c|cccc} x & x & x & x & x \\ \hline x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{array}\right] \left[\begin{array}{c|cccc} 1 & 0 & 0 & 0 & 0 \\ \hline 0 & & & & \\ 0 & & & & \\ 0 & & \hat{H}_1 & & \\ 0 & & & & \end{array}\right] = \left[\begin{array}{c|cccc} x & x & x & x & x \\ \hline x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{array}\right]$$

Repeat the previous step:

$$H_2(H_1 A H_1) = \left[\begin{array}{cc|ccc} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & & & \\ 0 & 0 & & \hat{H}_2 & \\ 0 & 0 & & & \end{array}\right] \left[\begin{array}{cc|ccc} x & x & x & x & x \\ x & x & x & x & x \\ \hline 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{array}\right] = \left[\begin{array}{cc|ccc} x & x & x & x & x \\ \hline x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \end{array}\right]$$

Approximating Eigenvalues

- **QR Algorithm: Upper Hessenberg form**

If n = 5, we obtain the 5 × 5 matrix:

$$H_3 H_2 H_1 A H_1^T H_2^T H_3^T$$
$$= H_3 H_2 H_1 A (H_3 H_2 H_1)^T = QAQ^T$$

In general, for an $n \times n$ matrix A, $n - 2$ Householder steps are needed to put A into upper Hessenberg form.

Approximating Eigenvalues

QR Algorithm: Upper Hessenberg form

For the matrix

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

the similar matrix with upper Hessenberg form $A' = QAQ^{\mathrm{T}}$

$$A' = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Approximating Eigenvalues

- **QR Algorithm: Upper Hessenberg form**

A complete method for finding all eigenvalues of an arbitrary square matrix A:

(1) The matrix is first put into upper Hessenberg form with the use of a similarity transformation.

(2) Then the shifted QR algorithm is applied.

***Note*: The MATLAB *eig* command provides accurate eigenvalues based on this progression of calculations**

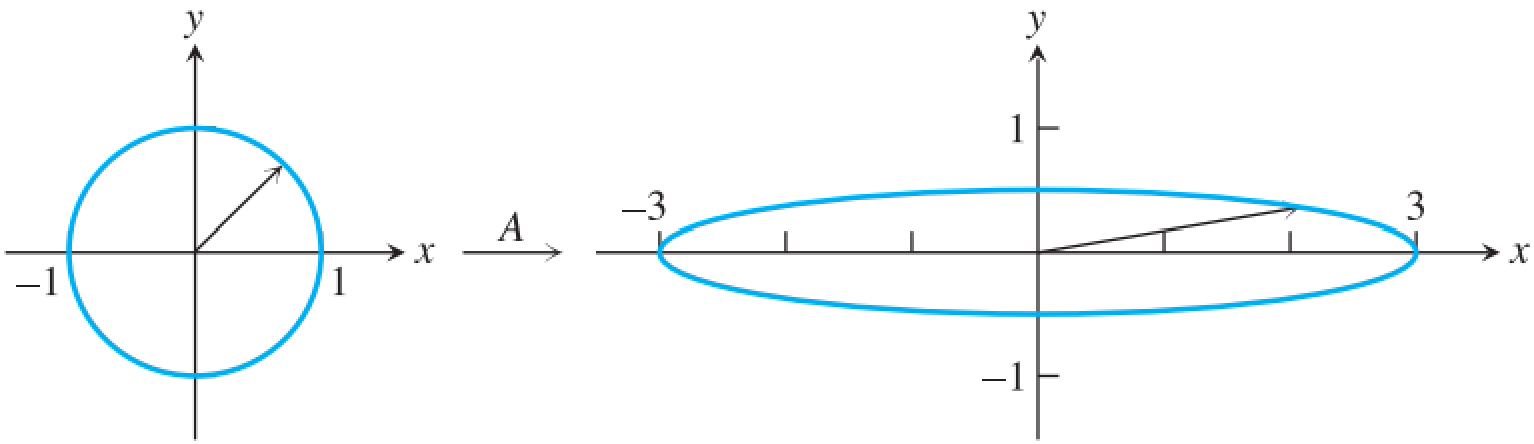
Eigenvalues and Singular Values

1. Review of Eigenvalues and Eigenvectors
2. Approximating Eigenvalues
   - Power Iteration Methods
   - QR Algorithm
3. Singular Value Decomposition
   - Applications of the SVD

Singular Value Decomposition

- **Geometry of the SVD**

For the matrix $A = \begin{bmatrix} 3 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$



The image of the unit circle under a 2 × 2 matrix.

Singular Value Decomposition

Geometry of the SVD

For every m × n matrix *A*, there are orthonormal sets $\{u_1,\ldots,u_m\}$ and $\{v_1,\ldots,v_n\}$, together with nonnegative numbers $s_1 \geq \cdots \geq s_n \geq 0$, satisfying

$$\begin{aligned} Av_1 &= s_1 u_1 \\ Av_2 &= s_2 u_2 \\ &\vdots \\ Av_n &= s_n u_n \end{aligned}$$

- **v_i : the right singular vectors of the matrix A**
- **u_i : the left singular vectors of A**
- **s_i : the singular values of A**

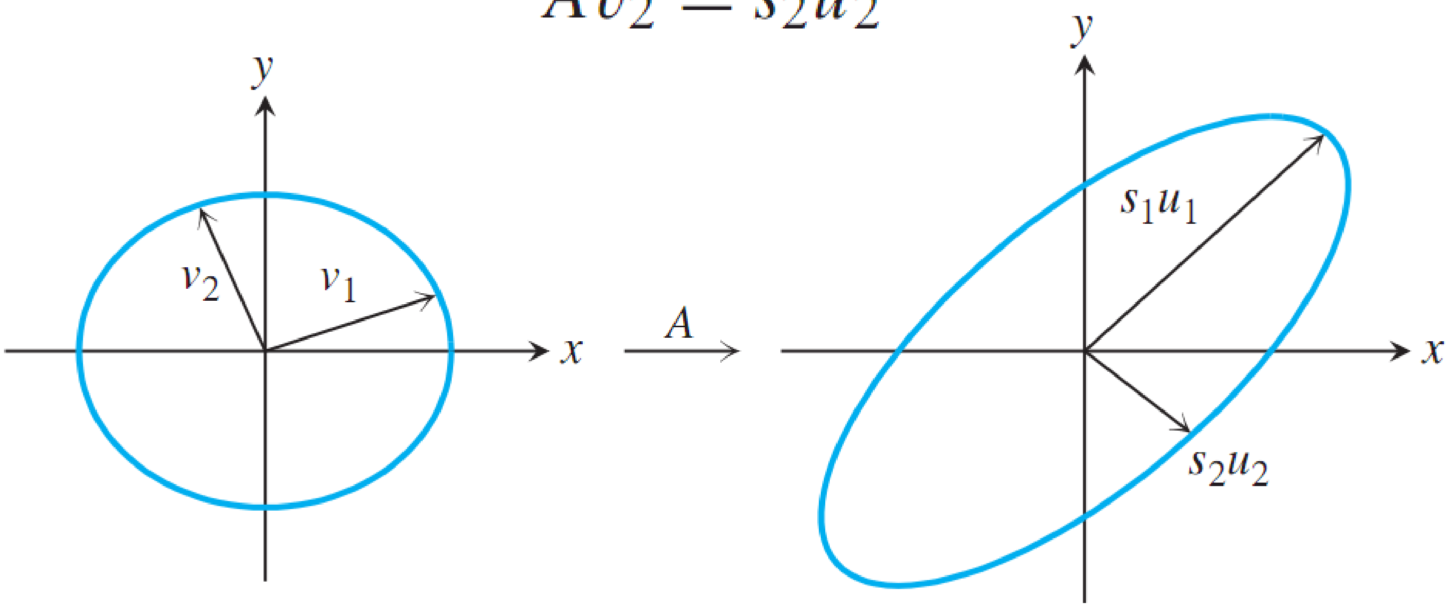
Singular Value Decomposition

Geometry of the SVD

This fact immediately explains why a 2 × 2 matrix maps the unit circle into an ellipse.

$$Av_1 = s_1 u_1$$

$$Av_2 = s_2 u_2$$



The image of the unit circle under a 2 × 2 matrix.

Singular Value Decomposition

- **Geometry of the SVD**

Form an m × m matrix U whose columns are the left singular vectors u_i , an n × n matrix V whose columns are the right singular vectors v_i , and a diagonal m × n matrix S whose diagonal entries are the singular values s_i .

Then the singular value decomposition (SVD) of the m × n matrix A is

$$A = USV^T$$

Let A be an m × n matrix. The eigenvalues of $A^T A$ are nonnegative.

Singular Value Decomposition

Finding the SVD

THEOREM12.11 of Ref. [1]

Let A be an $m \times n$ matrix where $m \geq n$. Then there exist two orthonormal bases $\{v_1,\ldots,v_n\}$ of R^n, and $\{u_1,\ldots,u_m\}$ of R^m, and real numbers $s_1 \geq \cdots \geq s_n \geq 0$ such that $Av_i = s_i u_i$ for $1 \leq i \leq n$. The columns of $V = [v_1| \ldots |v_n]$, the right singular vectors, are the set of orthonormal eigenvectors of A^TA; and the columns of $U = [u_1| \ldots |u_m]$, the left singular vectors, are the set of orthonormal eigenvectors of AA^T.

Singular Value Decomposition

Properties the SVD

- **The rank of the matrix $A = USV^T$ is the number of nonzero entries in S.**
- **If A is an n × n matrix, $|\det(A)| = s_1 \cdots s_n$.**
- **If A is an invertible m × m matrix, then $A^{-1} = VS^{-1}U^T$.**
- **The m × n matrix A can be written as the sum of rank-one matrices**

$$A = \sum_{i=1}^{r} s_i u_i v_i^T$$

Singular Value Decomposition

- **Applications of the SVD: Dimension Reduction**
- **The m × n matrix A = [a_1| ⋯ |a_n] can be written as the sum of rank-one matrices**

$$A = \sum_{i=1}^{r} s_i u_i v_i^T$$

Using the rank-*p* approximation

$$A \approx A_p = \sum_{i=1}^{p} s_i u_i v_i^T$$

Let e_j denote the *j* th elementary basis vector (all zeros except for *j* th entry 1), we can project a_j into the *p*-dimensional space:

$$a_j = Ae_j \approx A_p e_j$$

Singular Value Decomposition

- **Applications of the SVD: Example 4**

Find the singular value decomposition of the 4 × 2matrix

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

$$= USV^T = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 6 & 0 \\ 0 & 2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

```
A = [3,3; -3, -3; -1,1; 1, -1];
[U,S,V] = svd(A)
```

Singular Value Decomposition

- **Applications of the SVD: Example 5**

Find the best one-dimensional subspace fitting the data vectors [−4,−4.5], [0.8,1.9], [2.6,−0.7], [0.6,3.3].

Use the data vectors as columns of the data matrix

$$A = \begin{bmatrix} -4 & 0.8 & 2.6 & 0.6 \\ -4.5 & 1.9 & -0.7 & 3.3 \end{bmatrix}$$

and find its SVD, which is

$$USV^T = \begin{bmatrix} 0.6 & -0.8 \\ 0.8 & 0.6 \end{bmatrix} \begin{bmatrix} 5\sqrt{2} & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} -0.6\sqrt{2} & 0.2\sqrt{2} & 0.1\sqrt{2} & 0.3\sqrt{2} \\ 1/6 & 1/6 & -5/6 & 1/2 \end{bmatrix}$$

The best one-dimensional subspace is spanned by $u_1 = [0.6, 0.8]^T$

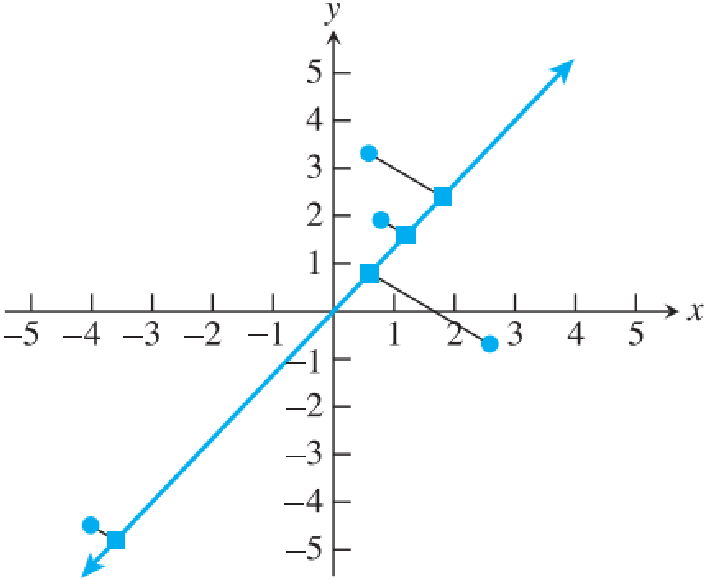
Singular Value Decomposition

Applications of the SVD: Example 5

Find the best one-dimensional subspace fitting the data vectors [−4,−4.5], [0.8,1.9], [2.6,−0.7], [0.6,3.3].

Setting $s_2 = 0$, $S_1 = \begin{bmatrix} 5\sqrt{2} & 0 \\ 0 & 0 \end{bmatrix}$

the columns of A_1 are the four projected vectors in R^1 corresponding to the four original data vectors.



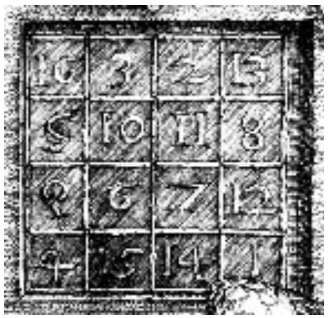
$$A_1 = \begin{bmatrix} 0.6 & -0.8 \\ 0.8 & 0.6 \end{bmatrix} \begin{bmatrix} 5\sqrt{2} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -0.6\sqrt{2} & 0.2\sqrt{2} & 0.1\sqrt{2} & 0.3\sqrt{2} \\ 1/6 & 1/6 & -5/6 & 1/2 \end{bmatrix}$$

$$= \begin{bmatrix} -3.6 & 1.2 & 0.6 & 1.8 \\ -4.8 & 1.6 & 0.8 & 2.4 \end{bmatrix}$$

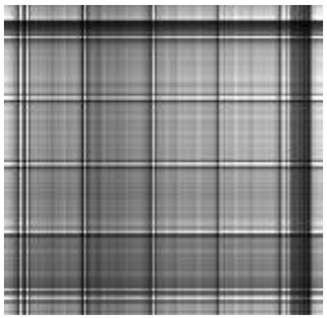
Singular Value Decomposition

- **Applications of the SVD: Example 6 Image Processing**

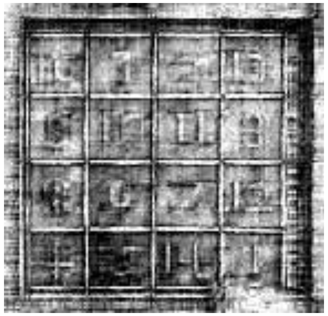
rank=359



rank=1



rank=20



rank=100

The first figure is the full rank image.

The truncation order in the second figure is rank = 1.

The truncation order in the third figure is rank = 20.

The truncation order in the fourth figure is rank = 100.

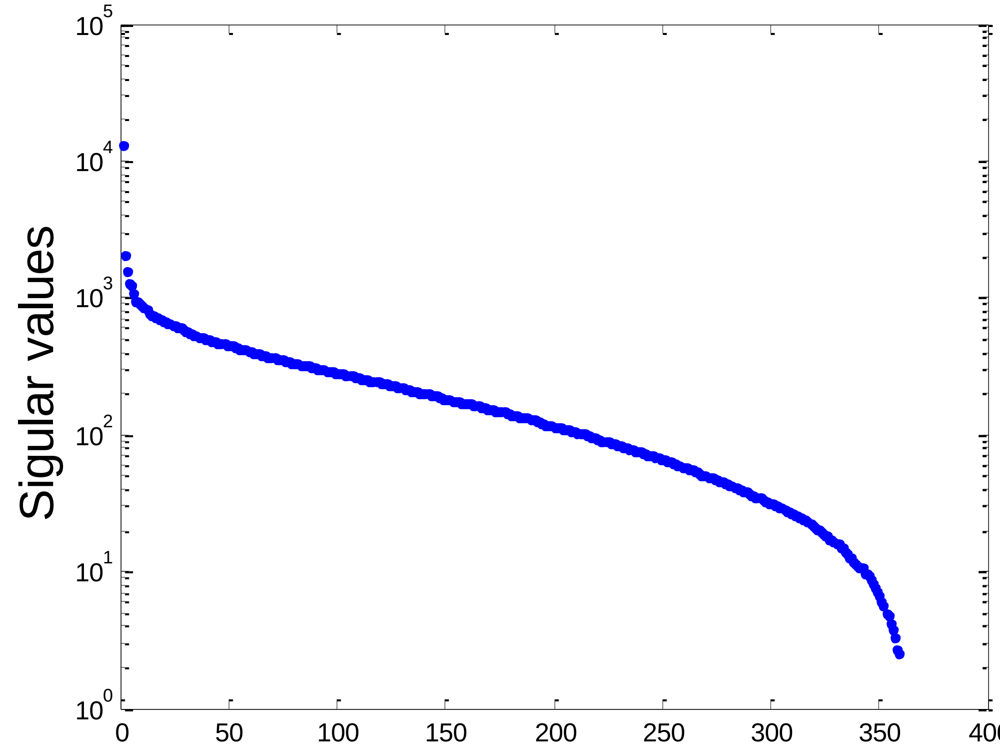
There is hardly any visible difference between the rank =100 approximation and the full rank image

Singular Value Decomposition

- **Applications of the SVD: Example 6 Image Processing**

The right figure is the logarithmic plot of the singular values of the digital image.

We can see that the singular values decrease rapidly. There is one greater than 10^4 and only six greater than 10^3 .



No. of singular values

MATLAB Built-in Functions

Functions for Eigenvalues and Singular Values

[V, D] = eig(X)

produces a diagonal matrix D of eigenvalues and a full matrix V whose columns are the corresponding eigenvectors so that X*V = V*D.

[V, D] = eig(A,B)

produces a diagonal matrix D of generalized eigenvalues and a full matrix V whose columns are the corresponding eigenvectors so that A*V = B*V*D.

d = eigs(A, k)

returns the k largest magnitude eigenvalues.

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

performs a singular value decomposition of matrix A, such that A = U*S*V'.

MATLAB Built-in Functions

Functions for Eigenvalues and Singular Values: Example 7

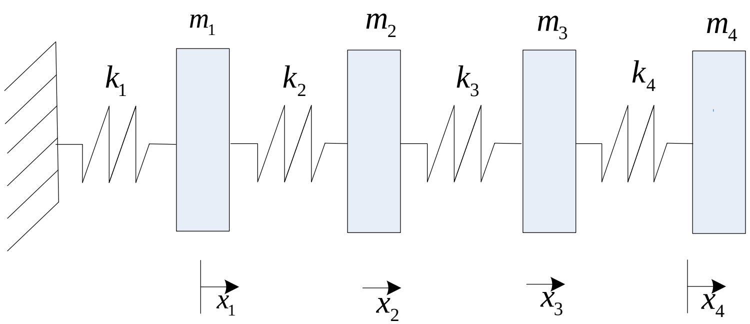
$$A = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 4 & 2 \\ 0 & 2 & 3 \end{bmatrix}$$

[V, D] = eig (A)

$$V = \begin{bmatrix} 0.3490 & 0.8944 & 0.2796 \\ -0.6252 & 0 & 0.7805 \\ 0.6981 & -0.4472 & 0.5592 \end{bmatrix} \qquad D = \begin{bmatrix} 1.2087 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5.7913 \end{bmatrix}$$

MATLAB Built-in Functions

Functions for Eigenvalues and Singular Values : Example 8



$$m_1 = m_2 = m_3 = m_4 = 1\,kg$$

$$k_1 = k_2 = k_3 = k_4 = 3.6 \times 10^4 (N/m)$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 & 0 \\ -k_2 & k_2 + k_3 & -k_3 & 0 \\ 0 & -k_3 & k_3 + k_4 & -k_4 \\ 0 & 0 & -k_4 & k_4 \end{bmatrix} \qquad M = \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ 0 & 0 & m_3 & 0 \\ 0 & 0 & 0 & m_4 \end{bmatrix}$$

MATLAB Built-in Functions

Functions for Eigenvalues and Singular Values: Example 8

[V, D] = eig (K, M)

$$V = \begin{bmatrix} -0.2280 & 0.5774 & 0.6565 & -0.4285 \\ -0.4285 & 0.5744 & -0.2280 & 0.6565 \\ -0.5744 & 0 & -0.5744 & -0.5744 \\ -0.6565 & -0.5744 & 0.4285 & 0.2280 \end{bmatrix} \quad D = \begin{bmatrix} 4340 & 0 & 0 & 0 \\ 0 & 36000 & 0 & 0 \\ 0 & 0 & 84500 & 0 \\ 0 & 0 & 0 & 127160 \end{bmatrix}$$

$$\omega = D^{1/2} = \begin{bmatrix} 65.8787 & 0 & 0 & 0 \\ 0 & 189.7367 & 0 & 0 \\ 0 & 0 & 290.6888 & 0 \\ 0 & 0 & 0 & 356.595 \end{bmatrix}$$

Eigenvalues and Singular Values

1. **Review of Eigenvalues and Eigenvectors**
2. **Approximating Eigenvalues**
   - **Power Iteration Methods**
   - **QR Algorithm**
     - **Simultaneous Iteration**
     - **Real Schur form**
     - **Upper Hessenberg form**
3. **Singular Value Decomposition**
   - **Applications of the SVD**

Thank You !