Eigenvalue Problems and Singular Value Decomposition

Sanzheng Qiao

Department of Computing and Software
McMaster University

August, 2012

Outline

Outline

1. Eigenvalue Problems
2. Singular Value Decomposition
3. Software Packages

Problem setting

Eigenvalue problem:

$$A\mathbf{x} = \lambda\mathbf{x},$$

λ: eigenvalue
$\mathbf{x}$: right eigenvector.
$\mathbf{y}^{\mathrm{H}}A = \lambda\mathbf{y}^{\mathrm{H}}$, $\mathbf{y}$ left eigenvector.

Canonical forms

Decomposition:

$$A = SBS^{-1}$$

where B is in a canonical (simple) form, whose eigenvalues and eigenvectors can be easily obtained.

- A and B have the same eigenvalues. (They are similar.)
- If $\mathbf{x}$ is an eigenvector of B, then $S\mathbf{x}$ is the eigenvector of A corresponding to the same eigenvalue.

Jordan canonical form

$$A = SJS^{-1}, \quad J = \operatorname{diag}(J_{n_1}(\lambda_1), \ldots, J_{n_k}(\lambda_k))$$

$$J_{n_i}(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda_i \end{bmatrix}.$$

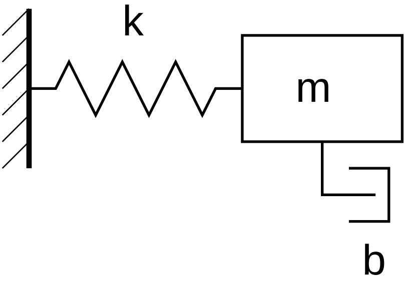
Jordan canonical form

- The algebraic multiplicity of λ_i is n_i.
- A Jordan block has one right eigenvector $[1, 0, ..., 0]^{\mathrm{T}}$ and one left eigenvector $[0, ..., 0, 1]^{\mathrm{T}}$.
- If all $n_i = 1$, then J is diagonal, A is called diagonalizable; otherwise, A is called defective.
- An n-by-n defective matrix has fewer than n eigenvectors.

Example

In practice, confronting defective matrices is a fundamental fact.

Mass-spring problem



Mass-spring problem

Newton's law $F = ma$ implies

$$m\ddot{x}(t) = -kx(t) - b\dot{x}(t).$$

Let

$$\mathbf{y}(t) = \begin{bmatrix} \dot{x}(t) \\ x(t) \end{bmatrix},$$

we transform the second order ODE into a system of the first order ODEs

$$\dot{\mathbf{y}}(t) = \begin{bmatrix} -\frac{b}{m} & -\frac{k}{m} \\ 1 & 0 \end{bmatrix} \mathbf{y}(t) =: A\mathbf{y}(t).$$

Mass-spring problem

The characteristic polynomial of A is

$$\lambda^2 + \frac{b}{m}\lambda + \frac{k}{m}$$

and the eigenvalues are

$$\lambda_\pm = \frac{-\frac{b}{m} \pm \sqrt{\left(\frac{b}{m}\right)^2 - 4\frac{k}{m}}}{2} = \frac{b}{2m}\left(-1 \pm \sqrt{1 - \frac{4km}{b^2}}\right).$$

When $4km/b^2 = 1$, critically damped, two equal eigenvalues, A is not diagonalizable.

Jordan canonical form

$$A = SJS^{-1}$$

$$A = \begin{bmatrix} -\frac{b}{m} & -\frac{k}{m} \\ 1 & 0 \end{bmatrix}, \qquad 4km = b^2,$$

$$J = \begin{bmatrix} -\frac{b}{2m} & 1 \\ 0 & -\frac{b}{2m} \end{bmatrix}, \quad S = \begin{bmatrix} -\frac{b}{2m} & 1 - \frac{b}{2m} \\ 1 & 1 \end{bmatrix}$$

Jordan canonical form

It is undesirable to compute Jordan form, because

- Jordan block is discontinuous

$$J(0) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad J(\epsilon) = \begin{bmatrix} \epsilon & 1 \\ 0 & 2\epsilon \end{bmatrix},$$

  while $J(0)$ has an eigenvalue of multiplicity two, $J(\epsilon)$ has two simple eigenvalues.

- In general, computing Jordan form is unstable, that is, there is no guarantee that $\widehat{S}\widehat{J}\widehat{S}^{-1} = A + E$ for a small E.

Schur canonical form

$$A = QTQ^{\mathrm{H}}$$

Q: unitary
T: upper triangular
The eigenvalues of A are the diagonal elements of T.

Schur canonical form

$$A = QTQ^{\mathrm{H}}$$

Q: unitary
T: upper triangular
The eigenvalues of A are the diagonal elements of T.

Real case

$$A = QTQ^{\mathrm{T}}$$

Q: orthogonal
T: quasi-upper triangular, 1-by-1 or 2-by-2 blocks on the diagonal.

Conditioning

Let λ be a simple eigenvalue of A with unit right eigenvector $\mathbf{x}$ and left eigenvector $\mathbf{y}$.
$\lambda + \epsilon$ be the corresponding eigenvalue of $A + E$, then

$$\epsilon = \frac{\mathbf{y}^{\mathrm{H}} E \mathbf{x}}{\mathbf{y}^{\mathrm{H}} \mathbf{x}} + O(\|E\|^2)$$

or

$$|\epsilon| \leq \frac{1}{|\mathbf{y}^{\mathrm{H}} \mathbf{x}|} \|E\| + O(\|E\|^2).$$

Condition number for finding a simple eigenvalue

$$1/|\mathbf{y}^{\mathrm{H}} \mathbf{x}|$$

Computing the Schur decomposition

$$A = QTQ^{\mathrm{T}}, \quad T: \text{ quasi-upper triangular}$$

Step 1: Reduce A to upper Hessenberg

$$A = Q_1 H Q_1^{\mathrm{T}}, \qquad h_{ij} = 0, \quad i > j + 1$$

Step 2: Compute the Schur decomposition of H

$$H = Q_2 T Q_2^{\mathrm{T}}$$

Introducing zeros into a vector

Householder transformation

$$H = I - 2\mathbf{u}\mathbf{u}^{\mathrm{T}} \quad \text{with } \mathbf{u}^{\mathrm{T}}\mathbf{u} = 1$$

H is symmetric and orthogonal ($H^2 = I$).
Goal: $H\mathbf{a} = \alpha\mathbf{e}_1$.

Introducing zeros into a vector

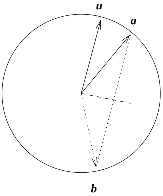
Householder transformation

$$H = I - 2\mathbf{u}\mathbf{u}^{\mathrm{T}} \quad \text{with } \mathbf{u}^{\mathrm{T}}\mathbf{u} = 1$$

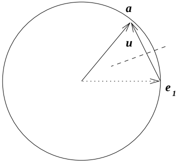
H is symmetric and orthogonal ($H^2 = I$).
Goal: $H\mathbf{a} = \alpha\mathbf{e}_1$.
Choose

$$\mathbf{u} = \mathbf{a} \pm \|\mathbf{a}\|_2\,\mathbf{e}_1$$

A geometric interpretation



(a)



(b)

Figure (a) shows the image $\mathbf{b} = (I - 2\mathbf{u}\mathbf{u}^{\mathrm{T}})\mathbf{a}$ for an arbitrary $\mathbf{u}$, in figure (b), $\mathbf{u} = \mathbf{a} - \|\mathbf{a}\|_2\, \mathbf{e}_1$.

Computing Householder transformations

Given a vector $\mathbf{x}$, it computes scalars σ, α, and vector $\mathbf{u}$ such that

$$(I - \sigma^{\dagger}\mathbf{u}\mathbf{u}^{\mathrm{T}})\mathbf{x} = -\alpha\mathbf{e}_1$$

where $\sigma^{\dagger} = 0$ if $\sigma = 0$ and σ^{-1} otherwise

- $\alpha = \mathrm{sign} x_1 \|\mathbf{x}\|_2$
- $\mathbf{u} = \mathbf{x} + \alpha\,\mathbf{e}_1$
- $\|u\|_2^2 = 2(\alpha^2 + \alpha\,x_1) = 2\alpha\,u_1$

house.m

```
function [u,sigma,alpha] = house(x)

u = x;

alpha = sign(u(1))*norm(u);
u(1) = u(1) + alpha;
sigma = alpha*u(1);
```

Reducing A to upper Hessenberg

```
n = length(A(1,:));
Q = eye(n);

for j=1:(n-2)
    [u, sigma, alpha] = myhouse(A(j+1:n, j));
    if sigma ~= 0.0
        for k = j:n
            A(j+1:n, k) = A(j+1:n, k) - ((u'*A(j+1:n, k))/sigma)*u;
        end %for k
        for i=1:n
            A(i, j+1:n) = A(i, j+1:n) - ((A(i, j+1:n)*u)/sigma)*u';
            Q(i, j+1:n) = Q(i, j+1:n) - ((Q(i, j+1:n)*u)/sigma)*u';
        end %for i
    end %if
end %for j
```

Computing eigenvalues and eigenvectors

Suppose A has distinct eigenvalues λ_i, $i = 1, ..., n$, where $|\lambda_1| > |\lambda_2| \geq ... \geq |\lambda_n|$, and $\mathbf{x}_i$ are the eigenvectors (linear independent).

An arbitrary vector $\mathbf{u}$ can be expressed as

$$\mathbf{u} = \mu_1 \mathbf{x}_1 + \mu_2 \mathbf{x}_2 + \cdots + \mu_n \mathbf{x}_n$$

If $\mu_1 \neq 0$, $A^k \mathbf{u}$ has almost the same direction as $\mathbf{x}_1$ when k and large and thus $(\lambda_i/\lambda_1)^k$ $(i > 1)$ is small.
Thus the Rayleigh quotient

$$\frac{(A^k\mathbf{u})^{\mathrm{T}} A (A^k \mathbf{u})}{(A^k\mathbf{u})^{\mathrm{T}} (A^k \mathbf{u})} \approx \lambda_1.$$

Power method

Initial $\mathbf{u}_0$; $i = 0$;
repeat

$$\mathbf{v}_{i+1} = A\mathbf{u}_i;$$
$$\mathbf{u}_{i+1} = \mathbf{v}_{i+1}/\|\mathbf{v}_{i+1}\|_2;$$
$$\tilde{\lambda}_{i+1} = \mathbf{u}_{i+1}^{\mathrm{T}} A \mathbf{u}_{i+1};$$
$$i = i + 1$$

until convergence

Power method

Initial $\mathbf{u}_0$; $i = 0$;
repeat

$$\mathbf{v}_{i+1} = A\mathbf{u}_i;$$
$$\mathbf{u}_{i+1} = \mathbf{v}_{i+1}/\|\mathbf{v}_{i+1}\|_2;$$
$$\tilde{\lambda}_{i+1} = \mathbf{u}_{i+1}^{\mathrm{T}} A\mathbf{u}_{i+1};$$
$$i = i + 1$$

until convergence

Problems

- Computes only (λ_1, x_1)
- Converges slowly when $|\lambda_1| \approx |\lambda_2|$
- Does not work when $|\lambda_1| = |\lambda_2|$

Inverse power method

Suppose that μ is an estimate of λ_k, then $(\lambda_k - \mu)^{-1}$ is the dominant eigenvalue of $(A - \mu I)^{-1}$. Applying the power method to $(A - \mu I)^{-1}$, we can compute $\mathbf{x}_k$ and λ_k.

Example.
Eigenvalues of A: -1, -0.2, 0.5, 1.5
Shift μ: -0.8
Eigenvalues of $(A - \mu I)^{-1}$: -5.0, 1.7, 0.78, 0.43

Very effective when we have a good estimate for an eigenvalue.

QR method

Goal: Generate a sequence

$$A_0 = A, A_1, ..., A_{k+1}$$

$$A_{i+1} = Q_i^{\mathrm{T}} A Q_i = \begin{bmatrix} B & \mathbf{u} \\ \mathbf{s}^{\mathrm{T}} & \mu \end{bmatrix}$$

where $\mathbf{s}$ is small and Q_i is orthogonal, i.e., $Q_i^{\mathrm{T}} = Q_i^{-1}$ (so A_{k+1} and A have the same eigenvalues).

- Since $\mathbf{s}$ is small, μ is an approximation of an eigenvalue of A_{k+1} (A);
- Deflate A_{k+1} and repeat the procedure on B when $\mathbf{s}$ is sufficiently small. The problem size is reduced by one.

QR method

What does Q_k look like?

If the last column of Q_k is a left eigenvector $\mathbf{y}$ of A, then

$$\begin{aligned} Q_k^{\mathrm{T}} A Q_k &= \begin{bmatrix} P_k^{\mathrm{T}} \\ \mathbf{y}^{\mathrm{T}} \end{bmatrix} A \,[P_k \ \mathbf{y}] \\ &= \begin{bmatrix} P_k^{\mathrm{T}} \\ \mathbf{y}^{\mathrm{T}} \end{bmatrix} [AP_k \ A\mathbf{y}] \\ &= \begin{bmatrix} B & \mathbf{u} \\ 0^{\mathrm{T}} & \lambda \end{bmatrix} \end{aligned}$$

QR method

How do we get an approximation of a left eigenvector $\mathbf{y}$ of A ($\mathbf{y}^{\mathrm{T}} A = \lambda \mathbf{y}^{\mathrm{T}}$)?

QR method

How do we get an approximation of a left eigenvector $\mathbf{y}$ of A ($\mathbf{y}^{\mathrm{T}} A = \lambda \mathbf{y}^{\mathrm{T}}$)?

One step of the inverse power method: Solve for $\mathbf{q}$ in $(A - \mu I)^{\mathrm{T}} \mathbf{q} = \mathbf{e}_n$, where μ is an estimate for an eigenvalue of A. (How? Later.)

QR method

How do we get an approximation of a left eigenvector $\mathbf{y}$ of A ($\mathbf{y}^{\mathrm{T}} A = \lambda \mathbf{y}^{\mathrm{T}}$)?

One step of the inverse power method: Solve for $\mathbf{q}$ in $(A - \mu I)^{\mathrm{T}} \mathbf{q} = \mathbf{e}_n$, where μ is an estimate for an eigenvalue of A. (How? Later.)

How do we construct an orthogonal Q whose last column is $\mathbf{q}$?

QR method

How do we get an approximation of a left eigenvector $\mathbf{y}$ of A ($\mathbf{y}^{\mathrm{T}}A = \lambda\mathbf{y}^{\mathrm{T}}$)?

One step of the inverse power method: Solve for $\mathbf{q}$ in $(A - \mu I)^{\mathrm{T}}\mathbf{q} = \mathbf{e}_n$, where μ is an estimate for an eigenvalue of A. (How? Later.)

How do we construct an orthogonal Q whose last column is $\mathbf{q}$?

If $(A - \mu I) = QR$ is the QR decomposition and $\mathbf{q}$ is the last column of Q, then

$$\mathbf{q}^{\mathrm{T}}(A - \mu I) = \mathbf{q}^{\mathrm{T}}QR = r_{n,n}\mathbf{e}_n^{\mathrm{T}}.$$

Thus, after normalizing,

$$(A - \mu I)^{\mathrm{T}}\mathbf{q} = \mathbf{e}_n.$$

QR method

How do we get an approximation of a left eigenvector $\mathbf{y}$ of A ($\mathbf{y}^{\mathrm{T}}A = \lambda\mathbf{y}^{\mathrm{T}}$)?

One step of the inverse power method: Solve for $\mathbf{q}$ in $(A - \mu I)^{\mathrm{T}}\mathbf{q} = \mathbf{e}_n$, where μ is an estimate for an eigenvalue of A. (How? Later.)

How do we construct an orthogonal Q whose last column is $\mathbf{q}$?

If $(A - \mu I) = QR$ is the QR decomposition and $\mathbf{q}$ is the last column of Q, then

$$\mathbf{q}^{\mathrm{T}}(A - \mu I) = \mathbf{q}^{\mathrm{T}}QR = r_{n,n}\mathbf{e}_n^{\mathrm{T}}.$$

Thus, after normalizing,

$$(A - \mu I)^{\mathrm{T}}\mathbf{q} = \mathbf{e}_n.$$

Q can be obtained from the QR decomposition $(A - \mu I) = QR$.

QR decomposition

QR decomposition of an upper Hessenberg matrix using the Givens rotations.

QR decomposition

QR decomposition of an upper Hessenberg matrix using the Givens rotations.

Givens rotation

$$G = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

Introducing a zero into a 2-vector:

$$G \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$$

i.e., rotate $\mathbf{x}$ onto x_1-axis.

Computing the Givens rotations

Given a vector $[a\ b]^{\mathrm{T}}$, compute $\cos\theta$ and $\sin\theta$ in the Givens rotation.

$$\cos\theta = \frac{a}{\sqrt{a^2+b^2}} \qquad \sin\theta = \frac{b}{\sqrt{a^2+b^2}}$$

Computing the Givens rotations

Given a vector $[a\ b]^{\mathrm{T}}$, compute $\cos\theta$ and $\sin\theta$ in the Givens rotation.

$$\cos\theta = \frac{a}{\sqrt{a^2+b^2}} \qquad \sin\theta = \frac{b}{\sqrt{a^2+b^2}}$$

```
function [c, s] = grotate(a, b)

if (b = 0) c = 1.0; s = 0.0; return; end;

if (abs(b) >= abs(a))
   ct = a/b;
   s = 1/sqrt(1 + ct*ct); c = s*ct;
else
   t = b/a;
   c = 1/sqrt(1 + t*t); s = c*t;
end
```

QR decomposition

Compute the QR decomposition $H = QR$ of an upper Hessenberg matrix H using the Givens rotations.

```
function [R, Q] = hqrd(H)

n = length(H(1,:));
R = H; Q = eye(n);

for j=1:n-1
    [c, s] = grotate(R(j,j), R(j+1,j));
    R(j:j+1, j:n) = [c s; -s c]*R(j:j+1, j:n);
    Q(:, j:j+1) = Q(:, j:j+1)*[c -s; s c];
end
```

QR method

But, we want

- similarity transformations of A, not $A - \mu I$;
- to carry on and improve accuracy (make $\mathbf{s}$ smaller).

QR method

But, we want

- similarity transformations of A, not $A - \mu I$;
- to carry on and improve accuracy (make **s** smaller).

$A - \mu I = QR$.

$RQ = Q^{\mathrm{T}}(A - \mu I)Q = Q^{\mathrm{T}}AQ - \mu I$.

$RQ + \mu I = Q^{\mathrm{T}}AQ$ is similar to A.

QR method

One step of QR method

```
repeat
   choose a shift mu;
   QR decomposition A - mu*I = QR;
   A = RQ + mu*I;
until convergence (A(n,1:n-1) small)
```

QR method

If A has been reduced to the upper Hessenberg form, the structure is maintained during the iteration.

H_0 is upper Hessenberg;

$H_0 - \mu I$ is upper Hessenberg;

$H_0 - \mu I = QR$, R is upper triangular and Q is upper Hessenberg;

$H_1 = RQ + \mu I$ is upper Hessenberg;

QR method

If A has been reduced to the upper Hessenberg form, the structure is maintained during the iteration.

H_0 is upper Hessenberg;

$H_0 - \mu I$ is upper Hessenberg;

$H_0 - \mu I = QR$, R is upper triangular and Q is upper Hessenberg;

$H_1 = RQ + \mu I$ is upper Hessenberg;

Implication: The QR decomposition is cheap (only eliminate the subdiagonal).

Choosing the shift

Since the last element converges to an eigenvalue, it is reasonable to choose $h_{n,n}$ as the shift. But, it doesn't always work. A more general method is to choose the eigenvalue of the trailing 2-by-2 submatrix

$$\begin{bmatrix} h_{n-1,n-1} & h_{n-1,n} \\ h_{n,n-1} & h_{n,n} \end{bmatrix}$$

that is close to $h_{n,n}$. Heuristically, it is more effective than choosing $h_{n,n}$ especially in the beginning.

Choosing the shift

Since the last element converges to an eigenvalue, it is reasonable to choose $h_{n,n}$ as the shift. But, it doesn't always work. A more general method is to choose the eigenvalue of the trailing 2-by-2 submatrix

$$\begin{bmatrix} h_{n-1,n-1} & h_{n-1,n} \\ h_{n,n-1} & h_{n,n} \end{bmatrix}$$

that is close to $h_{n,n}$. Heuristically, it is more effective than choosing $h_{n,n}$ especially in the beginning.

What if the trailing 2-by-2 submatrix has a complex conjugate pair of eigenvalues? The double shift strategy can be used to overcome the difficulty. In general, the Francis QR method using double implicit shift strategy can reduce an real Hessenberg matrix into the real Schur form.

Symmetric case

A symmetric matrix is diagonalizable: $A = Q\Lambda Q^{\mathrm{T}}$, $\Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_n)$.

QR method. This method is very efficient if only all eigenvalues are desired or all eigenvalues and eigenvectors are desired and the matrix is small ($n \leq 25$).

1. Reduce A to symmetric tridiagonal. This costs $\frac{4}{3}n^3$ or $\frac{8}{3}n^3$ if eigenvectors are also desired.
2. Apply the QR iteration to the tridiagonal. On average, it takes two QR steps per eigenvalue. Finding all eigenvalues takes $6n^2$. Finding all eigenvalues and eigenvectors requires $6n^3$.

Example

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

After tridiagonalization

$$\begin{bmatrix} 1.0000 & -5.3852 & 0 & 0 \\ -5.3852 & 5.1379 & -1.9952 & 0 \\ 0 & -1.9952 & -1.3745 & 0.2895 \\ 0 & 0 & 0.2895 & -0.7634 \end{bmatrix}$$

Example

	μ	β_1	β_2	β_3
1	-0.6480	3.8161	0.2222	-0.0494
2	-0.5859	1.2271	0.0385	10^{-5}
3	-0.5858	0.3615	0.0070	converge
4	-1.0990	0.0821	10^{-10}	
5	-1.0990	0.0186	converge	

Outline

Introduction

$$A = U\Sigma V^{\mathrm{T}}$$

A: m-by-n real matrix ($m \geq n$)
U: m-by-m orthogonal
V: n-by-n orthogonal
Σ: diagonal, $\mathrm{diag}(\sigma_i)$,
$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$
Singular values: σ_i
Left singular vectors: columns of U
Right singular vectors : columns of V

Introduction

SVD reveals many important properties of a matrix A. For example,

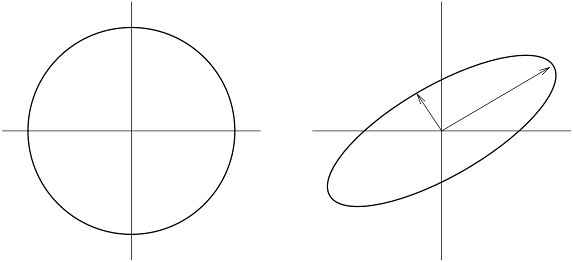
- The number of nonzero singular values is the rank of A. Suppose $\sigma_k > 0$ and $\sigma_{k+1} = 0$, then $\mathrm{rank}(A) = k$. If $k < n$, the columns of A are linearly dependent. (A is rank deficient.)
- If $\sigma_n > 0$ (A is of full rank),

$$\mathrm{cond}(A) = \frac{\sigma_1}{\sigma_n}$$

A geometric interpretation

Transformation A: $\mathbf{x} \rightarrow A\mathbf{x}$

$$\sigma_1 \geq \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|} \geq \sigma_n$$



Application: Linear least-squares problem

$$\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2^2$$

also called linear regression problem in statistics.

SVD: $A = U\Sigma V^{\mathrm{T}}$

$$\|A\mathbf{x} - \mathbf{b}\|_2^2 = \|\Sigma\mathbf{z} - \mathbf{d}\|_2^2$$

where

$$\mathbf{d} = U^{\mathrm{T}}\mathbf{b} \quad \mathbf{z} = V^{\mathrm{T}}\mathbf{x}$$

Application: Linear least-squares problem

Solution

$$\begin{aligned} z_j &= \frac{d_j}{\sigma_j} \qquad \text{if } \sigma_j \neq 0 \\ z_j &= \text{anything} \quad \text{if } \sigma_j = 0 \end{aligned}$$

Usually, we set

$$z_j = 0 \qquad \text{if } \sigma_j = 0$$

for minimum norm solution.

This allows us to solve the linear least-squares problems with singular A.

Application: Principal component analysis

Suppose that A is a data matrix. For example, each column contains samples of a variable. It is frequently standardized by subtracting the means of the columns and dividing by their standard deviations.

If A is standardized, then $A^{\mathrm{T}}A$ is the correlation matrix.

If the variables are strongly correlated, there are few components, fewer than the number of variables, can predict all the variables.

Application: Principal component analysis

In terms of SVD, let

$$A = U\Sigma V^{\mathrm{T}}$$

be the SVD of A. If A is m-by-n ($m \geq n$), we partition $U = [\mathbf{u}_1, ..., \mathbf{u}_m]$ and $V = [\mathbf{v}_1, ..., \mathbf{v}_n]$. Then we can write

$$A = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^{\mathrm{T}} + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^{\mathrm{T}}.$$

If the variables are strongly correlated, there are few singular values that are significantly larger than the others.

Application: Principal component analysis

In other words, we can find an r such that $\sigma_r \gg \sigma_{r+1}$. We can use the rank r matrix

$$A_r = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^{\mathrm{T}} + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^{\mathrm{T}}$$

to approximate A. In fact, A_r is the closest (in Frobenius norm) rank r approximation to A. Usually, we can find $r \ll n$.

It is also called low-rank approximation.

Principal component analysis is used in a wide range of fields. In image processing, for example, A is a 2D image.

Computing SVD

Note that

- the columns of U are the eigenvectors of AA^{T} (symmetric and positive semi-definite);
- the columns of V are eigenvectors of $A^{\mathrm{T}}A$.

The algorithm is parallel to the QR method for symmetric eigenvalue decomposition.

We work on A instead of $A^{\mathrm{T}}A$.

Computing the SVD

1. Bidiagonalize A using Householder transformations ($A \rightarrow B$ is upper bidiagonal and $B^{\mathrm{T}}B$ tridiagonal);
2. Implicit QR iteration
    1. Find a Givens rotation G_1 from the first column of $B^{\mathrm{T}}B - \mu I$;
    2. Apply G_1 to B;
    3. Apply a sequence of rotations, left and right, to restore the bidiagonal structure of B;

The shift μ is obtained by calculating the eigenvalues of the 2-by-2 trailing submatrix of $B^{\mathrm{T}}B$.

Outline

Software packages

- **NETLIB** LAPACK: sgees (Schur form), sgeev (eigenvalues and eigenvectors), ssyev (symmetric and dense eigenproblems), sstev (symmetric and tridiagonal eigenproblems), sgesvd (SVD), sbdsqr (small and dense SVD)
- **IMSL** evcrg, evcsf, lsvrr
- **MATLAB/Octave** schur, eig, svd
- **NAG** f02agf, f02abf, f02wef

Summary

- Eigenvalue decomposition, Jordan and Schur canonical forms
- Condition number for eigenvalue
- Householder transformation, Givens rotation
- QR method
- Singular value decomposition
- Linear least-squares problem
- Low-rank approximation

References

[1] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide. SIAM, Philadelphia, 2000.

[2] James W. Demmel. Applied Numerical Linear Algebra. SIAM, Philadelphia, 1997.
Ch 4, Ch 5

[3] G. H. Golub and C. F. Van Loan. Matrix Computations, 3rd Ed. The Johns Hopkins University Press, Baltimore, MD, 1996.
Ch 7, Ch 8