Chapter 8: Eigenvalues and Singular Values

Uri M. Ascher and Chen Greif
Department of Computer Science
The University of British Columbia
{ascher,greif}@cs.ubc.ca

Slides for the book
A First Course in Numerical Methods (published by SIAM, 2011)
`http://www.ec-securehost.com/SIAM/CS07.html`

Some slides are from the course of Dr. Peter Arbenz, ETH.

Goals of this chapter

- To find out how eigenvalues and singular values of a given matrix are computed;
- to find out how the largest (and smallest) few eigenvalues and singular values are computed;
- to see some interesting applications of eigenvalues and singular values.

Outline

- Algorithms for a few eigenvalues
- Uses of eigenvalues and eigenvectors
- Uses of SVD

Review of basic concepts

- Let $A \in \mathbb{R}^{n\times n}$. A scalar λ and a vector $\mathbf{x} \neq \mathbf{0}$ are an eigenvalue-eigenvector pair (or eigenpair) if

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

- The set $\sigma(A)$ of all eigenvalues form the spectrum of A.
- The nullspace $\mathcal{N}(\lambda I - A)$ is called eigenspace of λ.
  $\dim(\mathcal{N}(\lambda I - A)) =$ geometric $\leq$ algebraic multiplicity of λ.
- $A\mathbf{x} = \lambda\mathbf{x} \quad \Longrightarrow \quad (A + \alpha I)\mathbf{x} = (\lambda + \alpha)\mathbf{x}$
- $A\mathbf{x} = \lambda\mathbf{x} \quad \Longrightarrow \quad A^k\mathbf{x} = \lambda^k\mathbf{x}$
- Similarity transformation: Given a nonsingular matrix S, the matrix $S^{-1}AS$ has the same eigenvalues as A.
  (What about the eigenvectors?)

Review of basic concepts (cont.)

- Spectral decomposition: For a diagonalizable $n \times n$ real matrix A there are n (generally complex-valued) eigenpairs $(\lambda_j, \mathbf{x}_j)$, with $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ nonsingular.
  Then,
  $$AX = X\Lambda \quad \iff \quad A = X\Lambda X^{-1}$$
  with $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$.
- Any vector can be written as $\mathbf{y} = X\boldsymbol{\alpha} = \sum_j \mathbf{x}_j \alpha_j$ and
  $$A\mathbf{y} = \sum_{j=1}^{n} \alpha_j A\mathbf{x}_j = \sum_{j=1}^{n} \alpha_j \lambda_j \mathbf{x}_j.$$

Review of basic concepts (cont.)

- *Does every real $n \times n$ matrix have n eigenvalues?*
  We can write the eigenvalue problem as a homogeneous linear system

  $$(\lambda I - A)\mathbf{x} = \mathbf{0}.$$

  Since we want a nontrivial $\mathbf{x}$, this means that $\lambda I - A$ must be singular.
  Therefore we can find λ by forming the characteristic polynomial and finding its roots.
  That is, in principle we solve

  $$\det(\lambda I - A) = 0,$$

  which has n in general complex, not necessarily distinct roots.

Numerical example

- Example: The matrix

$$A = \begin{pmatrix} -0.9880 & 1.8000 & -0.8793 & -0.5977 & -0.7819 \\ -1.9417 & -0.5835 & -0.1846 & -0.7250 & 1.0422 \\ 0.6003 & -0.0287 & -0.5446 & -2.0667 & -0.3961 \\ 0.8222 & 1.4453 & 1.3369 & -0.6069 & 0.8043 \\ -0.4187 & -0.2939 & 1.4814 & -0.2119 & -1.2771 \end{pmatrix}$$

has eigenvalues given approximately by $\lambda_1 = -2$, $\lambda_2 = -1 + 2.5\imath$, $\lambda_3 = -1 - 2.5\imath$, $\lambda_4 = 2\imath$, and $\lambda_5 = -2\imath$.

It is known that closed form formulas for the roots of a polynomial do not generally exist if the polynomial is of degree 5 or higher. Thus we cannot expect to be able to solve the eigenvalue problem in a finite procedure.

Singular value decomposition

Let A be real $m \times n$ (rectangular in general). Then there are orthogonal matrices $U,\ V$ such that

$$A = U\Sigma V^T,$$

where

$$\Sigma = \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}, \quad S = \mathrm{diag}\{\sigma_1, \ldots, \sigma_r\},$$

with the **singular values** $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, $\sigma_{r+1} = \cdots = \sigma_n = 0$.

Connection to eigenvalues: $\sigma_i = \sqrt{\lambda_i}$, where λ_i are eigenvalues of $A^T A$.

Recall eigenvalues and singular values (Ch. 4)

- For a real, square $n \times n$ matrix A, an eigenvalue λ and corresponding eigenvector $\mathbf{x} \neq \mathbf{0}$ satisfy $A\mathbf{x} = \lambda\mathbf{x}$.
- There are n (possibly complex) eigenpairs $\lambda_1, \lambda_2, \ldots, \lambda_n$ and eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ s.t. $A\mathbf{x}_i = \lambda_i\mathbf{x}_i$. For $\Lambda = \mathrm{diag}(\lambda_i)$

$$AX = X\Lambda.$$

- If A is non-defective then the eigenvector matrix X is nonsingular:

$$X^{-1}AX = \Lambda.$$

- For a real, $m \times n$ matrix A, the singular value decomposition (SVD) is

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

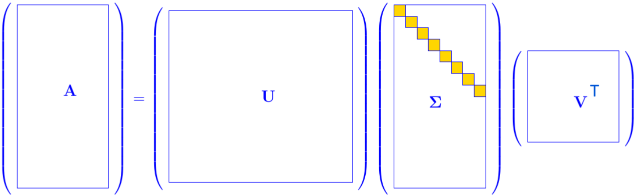
  where U $m \times m$ and V $n \times n$ are orthogonal matrices and Σ is "diagonal" consisting of zeros and singular values $\sigma_1 \geq \sigma_2, \cdots \geq \sigma_r > 0$, $r \leq \min(m,n)$.
- Note: singular values are square roots of eigenvalues of A^TA.

Singular value decomposition

Theorem (Singular value decomposition (SVD))

For any matrix $A \in \mathbb{R}^{m\times n}$, $m \geq n$, *there are unitary* $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{n\times n}$ *and a (generalized) diagonal matrix* $\Sigma = (\sigma_1, \ldots, \sigma_n)$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, $\sigma_{r+1} = \cdots = \sigma_m = 0$, *such that*

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

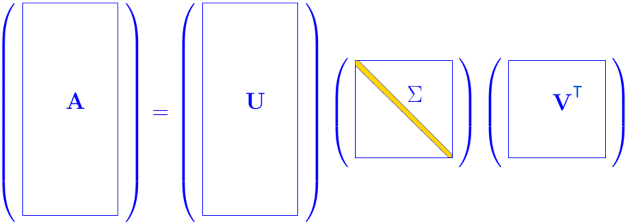


Singular value decomposition (cont.)

The decomposition

$$A = U\Sigma V^{\intercal}$$

is called singular value decomposition (SVD). There is an economical variant:



Singular value decomposition (cont.)

SVD useful to compute

- Condition number of a matrix:

$$\kappa(A) = \kappa_2(A) = \sigma_1(A)/\sigma_p(A).$$

  If $\sigma_p(A) = 0$ then the matrix is singular or rank deficient.
- Nullspace of a matrix:
  The columns of V corresponding to zero singular values span $\mathcal{N}(A)$.
- Range of a matrix:
  The columns of U corresponding to positive singular values span $\mathcal{R}(A)$.

Singular value decomposition (cont.)

MATLAB-functions for computing the SVD:

`s = svd(A)`	: computes singular values of matrix A
`[U,S,V] = svd(A)`	: computes singular value decomposition
`[U,S,V] = svd(A,0)`	: "economical" singular value decomposition for $m > n$: : $U \in \mathbb{R}^{m\times n}$, $\Sigma \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{n\times n}$
`s = svds(A,k)`	: k largest singular values of sparse A
`[U,S,V] = svds(A,k)`	: partial singular value decomposition: $U \in \mathbb{R}^{m\times k}$, $V \in \mathbb{R}^{n\times k}$, $\Sigma \in \mathbb{R}^{k,k}$ diagonal with k largest singular values of A.

MATLAB code for computing nullspace

```
function V = kerncomp(A,tol)
% computes an orthonormal basis of nullspace(A) using SVD.
% kernel selection with relative tolerance tol
if (nargin < 2), tol = eps; end
[U,S,V] = svd(A);  % Singular Value Decomposition
s = diag(S);       % Extract vector of singular values
% find singular values of relative size < tol*sigma(1)
V = V(:,find(s < tol*s(1)));  % rightmost columns of V.
```

Condition number of rectangular matrices

For a rectangular matrix $A \in \mathbb{R}^{m \times n}$, $m \geq n$, let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$ be the eigenvalues of $B = A^T A$. Define singular values $\sigma_1, \ldots, \sigma_n$ of A as

$$\sigma_i = \sqrt{\lambda_i}, \qquad i = 1, 2, \ldots, n.$$

Define condition number of the rectangular A

$$\kappa(A) = \kappa_2(A) = \frac{\sigma_1}{\sigma_n} = \sqrt{\frac{\lambda_1}{\lambda_n}}.$$

Note

$$\kappa_2(B) = \frac{\lambda_1}{\lambda_n} = \kappa_2(A)^2$$

B: symmetric and positive definite matrix Why?

Outline

- Algorithm to find a few eigenvalues: the most dominant eigenvalue
- Uses of eigenvalues and eigenvectors
- Uses of SVD

Power method

- Given $\mathbf{v}_0$, expand using eigenpairs

$$\mathbf{v}_0 = \sum_{j=1}^{n} \beta_j \mathbf{x}_j,$$

where $A\mathbf{x}_j = \lambda_j \mathbf{x}_j, \qquad j = 1, \ldots, n.$

- Then

$$A\mathbf{v}_0 = \sum_{j=1}^{n} \beta_j A\mathbf{x}_j = \sum_{j=1}^{n} (\beta_j \lambda_j)\mathbf{x}_j.$$

- Multiplying by A $k-1$ times gives

$$A^k\mathbf{v}_0 = \sum_{j=1}^{n} \beta_j \lambda_j^{k-1} A\mathbf{x}_j = \sum_{j=1}^{n} (\beta_j \lambda_j^{k})\mathbf{x}_j.$$

Power method (cont.)

Assuming:

- A non-defective
- $|\lambda_1| > |\lambda_j|, \ j = 2, \ldots, n$
- $\beta_1 \neq 0$

Obtain

$$A^k \mathbf{v}_0 \ \rightarrow \ \mathbf{x}_1.$$

Given eigenvector, obtain eigenvalue from Rayleigh quotient

$$\lambda_1 = \frac{\mathbf{x}_1^T A \mathbf{x}_1}{\mathbf{x}_1^T \mathbf{x}_1}.$$

Defective matrix: Matrices whose eigenvectors do not span R^n

Algorithm: Power method

Input matrix A and initial guess $\mathbf{v}_0$.
for $k := 1, 2, \ldots$ until termination **do**
$\quad \tilde{\mathbf{v}} = A\mathbf{v}_{k-1}$
$\quad \mathbf{v}_k = \tilde{\mathbf{v}} / \|\tilde{\mathbf{v}}\|$
$\quad \lambda_1^{(k)} = \mathbf{v}_k^T A \mathbf{v}_k$
end for

The convergence criterion may be, e.g., the angle between $\mathbf{v}_k$ and $\mathbf{v}_{k-1}$:

$$\sin \angle(\mathbf{v}_k, \mathbf{v}_{k-1}) = \|\mathbf{v}_k - \mathbf{v}_{k-1}(\mathbf{v}_{k-1}^T \mathbf{v}_k)\| = \|(I - \mathbf{v}_{k-1}\mathbf{v}_{k-1}^T)\mathbf{v}_k\|.$$

Power method algorithm and properties

- Algorithm:
  For $k = 1, 2, \ldots$ until termination

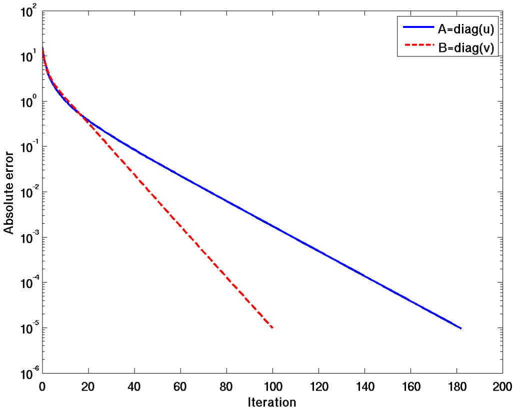
$$\tilde{\mathbf{v}} = A\mathbf{v}_{k-1}$$
$$\mathbf{v}_k = \tilde{\mathbf{v}}/\|\tilde{\mathbf{v}}\|_2$$
$$\lambda_1^{(k)} = \mathbf{v}_k^T A\mathbf{v}_k.$$

- Properties:
  - Simple, basic, can be slow.
  - Can be applied to large, sparse matrices.
  - Used as a building block for other, more robust algorithms.

If there is no single dominant eigenvalue, or eigenvectors are not linearly independent

Example

$$A = \mathrm{diag}(1, 2, \ldots, 30, 31, 32) \in \mathbb{R}^{32\times 32}$$
$$B = \mathrm{diag}(1, 2, \ldots, 30, 30, 32) \in \mathbb{R}^{32\times 32}$$



Absolute error:

$$|\lambda_1^{(k)} - \lambda_1| \equiv |\lambda_1^{(k)} - 32|.$$

$$\frac{31}{32} \approx 0.968$$

$$\frac{30}{32} \approx 0.938$$

Solving almost singular linear systems

- Let $A \in \mathbb{R}^{n \times n}$. If $\kappa(A)$ is very large, then solving $A\mathbf{x} = \mathbf{b}$ can be an ill-conditioned problem.
- With the SVD of A we get

$$\mathbf{x} = V\Sigma^{-1}U^T\mathbf{b}.$$

- If the problem is too ill-conditioned to be of use, then people often seek to regularize it. This means, replace the given problem intelligently by a nearby problem which is better conditioned.
- Using SVD this can be done by setting the singular values below a cutoff tolerance to 0, and minimizing the 2-norm of the solution to the resulting underdetermined problem.

Solving almost singular linear systems (cont.)

- We proceed as follows:
  1. Starting from n go backward until r is found such that $\frac{\sigma_1}{\sigma_r}$ is tolerable in size. This is the condition number of the problem that we actually solve.
  2. Calculate $\mathbf{z} = U^T\mathbf{b}$; in fact just the first r components of $\mathbf{z}$ are needed. In other words, if $\mathbf{u}_i = U\mathbf{e}_i$ is the ith column vector of U, then $z_i = \mathbf{u}_i^T\mathbf{b}$, $i = 1, \ldots, r$.
  3. Calculate $y_i = z_i/\sigma_i$, $i = 1, \ldots, r$, and set $y_i = 0$, $r < i \leq n$.
  4. Calculate $\mathbf{x} = V\mathbf{y}$. This really involves only the first r columns of V and the first r components of $\mathbf{y}$. In other words, if $\mathbf{v}_i$ is the ith column vector of V, then $\mathbf{x} = \sum_{i=1}^{r} y_i\mathbf{v}_i^T$

  Note: In general, the resulting $\mathbf{x}$ does not satisfy $A\mathbf{x} = \mathbf{b}$. But its the best one can do under certain circumstances, and it produces a solution x of the smallest norm for a sufficiently well-conditioned approximate problem.

Truncated SVD and data compression

- Given an $m \times n$ matrix A, the best rank-r approximation of $A = U\Sigma V^T$ is the matrix

$$A_r = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T.$$

- This is another example of truncated SVD (TSVD), so named because only the first r columns of U and V are utilized.
- It is a model reduction technique, which is a best approximation in the sense that $\|A - A_r\|_2$ is minimal over all possible rank-r matrices. The minimum residual norm is equal to σ_{r+1}.
- Note A_r uses only $r(m+n+1)$ storage locations – significantly fewer than mn if $r \ll \min(m,n)$.

Example

- The least squares problem $\min_{\mathbf{x}} \|C\mathbf{x} - \mathbf{b}\|$ is easily solved for

$$C = \begin{pmatrix} 1 & 0 & 1 \\ 2 & 3 & 5 \\ 5 & 3 & -2 \\ 3 & 5 & 4 \\ -1 & 6 & 3 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ -2 \\ 5 \\ -2 \\ 1 \end{pmatrix}.$$

  Call the solution $\hat{\mathbf{x}}$.

- Next consider $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|$, where we add to C a column that is sum of the three previous ones:

$$A = \begin{pmatrix} 1 & 0 & 1 & 2 \\ 2 & 3 & 5 & 10 \\ 5 & 3 & -2 & 6 \\ 3 & 5 & 4 & 12 \\ -1 & 6 & 3 & 8 \end{pmatrix}.$$

- Note that A has $m = 5,\ n = 4,\ r = 3$: can't reliably solve this using normal equations or even QR.
- But the truncated SVD method works: if $\mathbf{x}$ solves the problem with A, obtain $\|\mathbf{x}\| \approx \|\hat{\mathbf{x}}\|$, and $\|A\mathbf{x} - \mathbf{b}\| \approx \|C\hat{\mathbf{x}} - \mathbf{b}\|$.

Example

- The least squares problem $\min_{\mathbf{x}} \|C\mathbf{x} - \mathbf{b}\|$ is easily solved for

$$C = \begin{pmatrix} 1 & 0 & 1 \\ 2 & 3 & 5 \\ 5 & 3 & -2 \\ 3 & 5 & 4 \\ -1 & 6 & 3 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ -2 \\ 5 \\ -2 \\ 1 \end{pmatrix}.$$

  Call the solution $\hat{\mathbf{x}}$.

- Next consider $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|$, where we add to C a column that is sum of the three previous ones:

$$A = \begin{pmatrix} 1 & 0 & 1 & 2 \\ 2 & 3 & 5 & 10 \\ 5 & 3 & -2 & 6 \\ 3 & 5 & 4 & 12 \\ -1 & 6 & 3 & 8 \end{pmatrix}.$$

- Note that A has $m = 5,\ n = 4,\ r = 3$: can't reliably solve this using normal equations .
- But the truncated SVD method works: if $\mathbf{x}$ solves the problem with A, obtain $\|\mathbf{x}\| \approx \|\hat{\mathbf{x}}\|$, and $\|A\mathbf{x} - \mathbf{b}\| \approx \|C\hat{\mathbf{x}} - \mathbf{b}\|$.

Compressing image information

- Image composed of $m \times n$ pixels (greyscale, BMP format).
- Matrix $A \in \mathbb{R}^{m \times n}$, $0 \leq a_{ij} \leq 255$.
- Compression scheme using SVD: Replace A by A_k. $k = ?$
- Represent A_k by storing the first k columns of U and V.
- Instead of mn numbers only store $k(m + n)$ numbers.

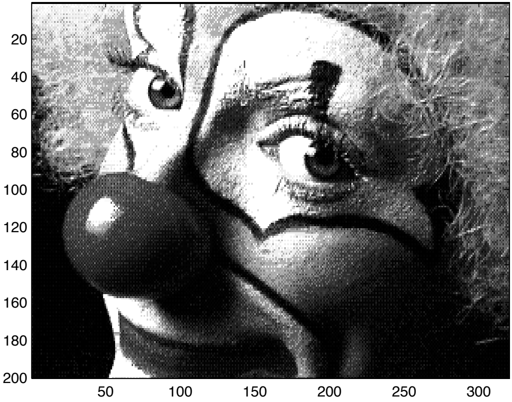
Compressing image information (cont.)

```
close all
colormap('gray')
load clown.mat;
figure(1)
image(X);

[U,S,V] = svd(X);
figure(2)
r = 20;
colormap('gray')
image(U(:,1:r)*S(1:r,1:r)*V(:,1:r)')
```

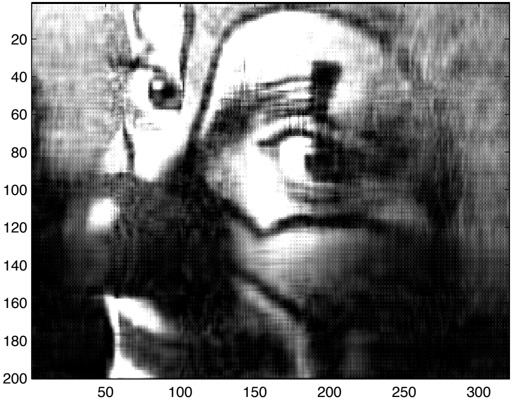
Example

Consider the following clown image, taken from MATLAB's image repository:



Compressed image of clown example

Here is what we get if we use $r = 20$.



Assessment

- The original image requires $200 \times 320 = 64000$ matrix entries to be stored; the compressed image requires merely $20 \cdot (200 + 320 + 1) \approx 10000$ storage locations.
- By storing less than 16% of the data, we get a reasonable image. It's not great, but all main the features of the clown are clearly shown.
- What are less clear in the compressed image are fine features (high frequency), such as the fine details of the clown's hair.
- Certainly, advanced techniques such as DCT (Chapter 13) and wavelet are far superior to SVD for the task of image compression. Still, our example visually shows that most information is stored already in the leading singular vectors.