بسم الله الرحمن الرحیم

بازشناسی الگو

درس ۱۱

تولید ویژگی

Feature Generation

کاظم فولادی قلعه
دانشکده مهندسی، پردیس فارابی
دانشگاه تهران

http://courses.fouladi.ir/pr

تولید ویژگی

۱

مقدمه

OPTIMAL FEATURE GENERATION

- In general, feature generation is a problem-dependent task. However, there are a few general directions common in a number of applications. We focus on three such alternatives.
  - LDA
  - PCA
  - ICA

تولید ویژگی

۱

تحلیل تفکیک خطی (LDA)

❖Linear Discriminant Analysis (LDA)

➢ **Optimized features based on Scatter matrices**
(Fisher's linear discrimination).

- **The goal:** Given an original set of m measurements $\underline{x} \in \mathbb{R}^m$, compute $\underline{y} \in \mathbb{R}^\ell$, by the linear transformation, A,

$$\underline{y} = A^T \underline{x}$$

so that the J_3 scattering matrix criterion involving S_w, S_b is maximized. A^T is an $\ell \times m$ matrix.

- The basic steps in the proof:
  - $J_3 = \text{trace}(S_w^{-1} S_m)$
  - $S_{yw} = A^T S_{xw} A, \quad S_{yb} = A^T S_{xb} A,$
  - $J_3(A) = \text{trace}\{(A^T S_{xw} A)^{-1} (A^T S_{xb} A)\}$
  - Compute A so that $J_3(A)$ is maximum.

- The solution:

  - Let B be the matrix that diagonalizes simultaneously matrices S_{yw}, S_{yb}, i.e:

$$B^T S_{yw} B = I, \; B^T S_{yb} B = D$$

  where B, is a $\ell \times \ell$ matrix and D, a $\ell \times \ell$ diagonal matrix.

– Let $C = AB$ an $m \times \ell$ matrix. If A maximizes $J_3(A)$ then

$$\left(S_{xw}^{-1} S_{xb}\right) C = CD$$

The above is an eigenvalue-eigenvector problem. For an M-class problem, $S_{xw}^{-1} S_{xb}$ is of rank $M-1$.

- If $\ell = M-1$, choose C to consist of the $M-1$ eigenvectors, corresponding to the non-zero eigenvalues.

$$\underline{y} = C^T \underline{x}$$

The above guarantees maximum J_3 value.
In this case: $J_{3,x} = J_{3,y}$.

- For a two-class problem, this results to the well known Fisher's linear discriminant

$$\underline{y} = \left(\underline{\mu}_1 - \underline{\mu}_2\right) S_{xw}^{-1} \underline{x}$$

For Gaussian classes, this is the optimal Bayesian classifier, with a difference of a threshold value .

- If $\ell < M-1$, choose the ℓ eigenvectors corresponding to the ℓ largest eigenvectors.
- In this case, $J_{3,y} < J_{3,x}$, that is there is loss of information.

– Geometric interpretation. The vector $\underline{y}$ is the projection of $\underline{x}$ onto the subspace spanned by the eigenvectors of $S_{xw}^{-1}S_{xb}$.

تولید ویژگی

۳

تحلیل مؤلفه‌های اصلی (PCA)

❖ Principal Components Analysis (PCA)

(The Karhunen – Loève transform):

➢ **The goal:** Given an original set of m measurements $\underline{x} \in \mathbb{R}^m$ compute $\underline{y} \in \mathbb{R}^\ell$

$$\underline{y} = A^T \underline{x}$$

for an orthogonal A, so that the elements of $\underline{y}$ are optimally mutually uncorrelated.

That is

$$E\left[y(i)y(j)\right] = 0, i \neq j.$$

➢ Sketch of the proof:

$$R_y = E\left[\underline{y}\underline{y}^T\right] = E\left[A^T \underline{x}\underline{x}^T A\right] = A^T R_x A.$$

- If A is chosen so that its columns $\underline{a}_i$ are the orthogonal eigenvectors of R_x, then

$$R_y = A^T R_x A = \Lambda$$

where Λ is diagonal with elements the respective eigenvalues λ_i.

- Observe that this is a sufficient condition but not necessary. It **imposes** a specific orthogonal structure on A.

➢ Properties of the solution

- Mean Square Error approximation.

Due to the orthogonality of A:

$$\underline{x} = \sum_{i=0}^{m} y(i)\underline{a}_i, \quad y(i) = \underline{a}_i^T \underline{x}$$

– Define

$$\underline{\hat{x}} = \sum_{i=0}^{\ell-1} y(i)\underline{a}_i$$

– The Karhunen-Loève transform minimizes the square error:

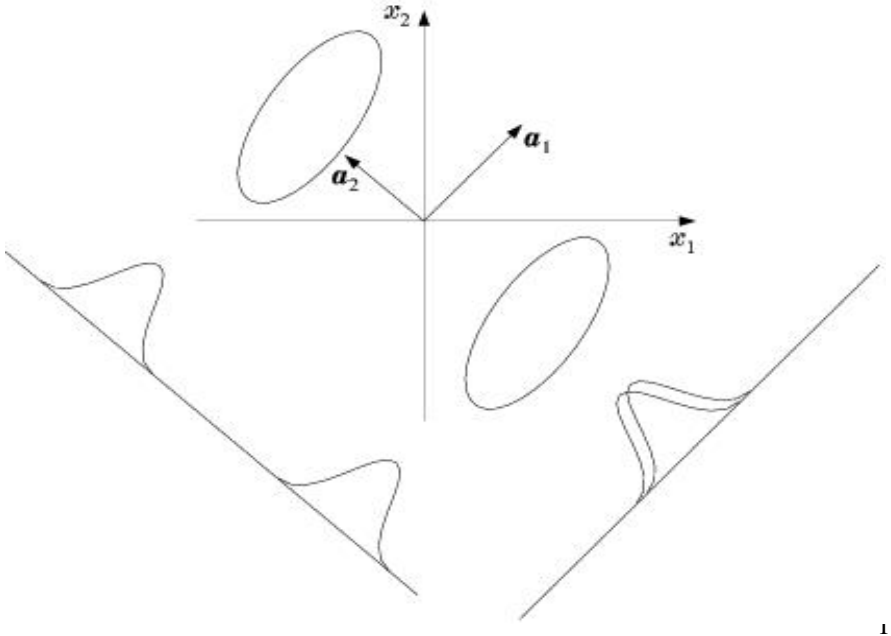
$$E\left[\left\|\underline{x} - \underline{\hat{x}}\right\|^2\right] = E\left[\left\|\sum_{i=\ell}^{m} y(i)\underline{a}_i\right\|^2\right]$$

– The error is:

$$E\left[\left\|\underline{x} - \underline{\hat{x}}\right\|^2\right] = \sum_{i=\ell}^{m} \lambda_i$$

It can be also shown that this is the minimum mean square error compared to **any** other representation of x by an ℓ-dimensional vector.

– In other words, $\hat{\underline{x}}$ is the projection of $\underline{x}$ into the subspace spanned by the principal ℓ eigenvectors. However, for Pattern Recognition this is not the always the best solution.



- Total variance: It is easily seen that

$$\sigma^2_{y(i)} = E\left[y^2(i) \right] = \lambda_i$$

Thus Karhunen-Loève transform makes the total variance maximum.

- Assuming $\underline{y}$ to be a zero mean multivariate Gaussian, then the K-L transform maximizes the entropy:

$$H_y = -E\left[\ln P_{\underline{y}}(\underline{y}) \right].$$

of the resulting $\underline{y}$ process.

➢ Subspace Classification. Following the idea of projecting in a subspace, the subspace classification classifies an unknown $\underline{x}$ to the class whose subspace is closer to $\underline{x}$.

The following steps are in order:

- For each class, estimate the autocorrelation matrix R_i, and compute the m largest eigenvalues. Form A_i, by using respective eigenvectors as columns.

- Classify $\underline{x}$ to the class ω_j, for which the norm of the subspace projection is maximum

$$\left\| A_i^T \underline{x} \right\| > \left\| A_j^T \underline{x} \right\| \quad \forall i \neq j$$

According to Pythagoras theorem, this corresponds to the subspace to which $\underline{x}$ is closer.

تولید ویژگی

۴

تحلیل مؤلفه‌های مستقل (ICA)

❖ Independent Component Analysis (ICA)

In contrast to PCA, where the goal was to produce uncorrelated features, the goal in ICA is to produce statistically independent features. This is a much stronger requirement, involving higher to second order statistics. In this way, one may overcome the problems of PCA, as exposed before.

➢ **The goal:** Given $\underline{x}$, compute $\underline{y} \in \mathbb{R}^{\ell}$

$$\underline{y} = W\underline{x}$$

so that the components of $\underline{y}$ are statistically independent.

In order the problem to have a solution, the following assumptions must be valid:

- Assume that $\underline{x}$ is indeed generated by a linear combination of independent components

$$\underline{x} = \Phi \underline{y}$$

  Φ is known as the mixing matrix and W as the demixing matrix.

- Φ must be invertible or of full column rank.
- Identifiability condition: All independent components, $y(i)$, must be non-Gaussian. Thus, in contrast to PCA that can always be performed, ICA is meaningful for non-Gaussian variables.
- Under the above assumptions, $y(i)$'s can be uniquely estimated, within a scalar factor.

➢ Common's method: Given $\underline{x}$, and under the previously stated assumptions, the following steps are adopted:

- Step 1: Perform PCA on $\underline{x}$:

$$\underline{y} = A^T \underline{x}$$

- Step 2: Compute a unitary matrix, $\hat{A}$, so that the fourth order cross-cummulants of the transform vector

$$\underline{y} = \hat{A}^T \hat{y}$$

unitary: $\hat{A}^* \hat{A} = \hat{A}\hat{A}^* = I$

are zero. This is equivalent to searching for an $\hat{A}$ that makes the squares of the auto-cummulants maximum,

$$\max_{\hat{A}\hat{A}^T = I} \Psi(\hat{A}) = \sum \kappa_4 \left(y(i) \right)^2$$

where, $\kappa_4 \left(\cdot \right)$ is the 4th order auto-cumulant.

Cummulants:

$$\kappa_1(y(i)) = E[y(i)] = 0$$

$$\kappa_2(y(i)y(j)) = E[y(i)y(j)]$$

$$\kappa_3(y(i)y(j)y(k)) = E[y(i)y(j)y(k)]$$

and the fourth-order cumulants are given by

$$\kappa_4(y(i)y(j)y(k)y(r)) = E[y(i)y(j)y(k)y(r)] - E[y(i)y(j)]E[y(k)y(r)] - E[y(i)y(k)]E[y(j)y(r)] - E[y(i)y(r)]E[y(j)y(k)]$$

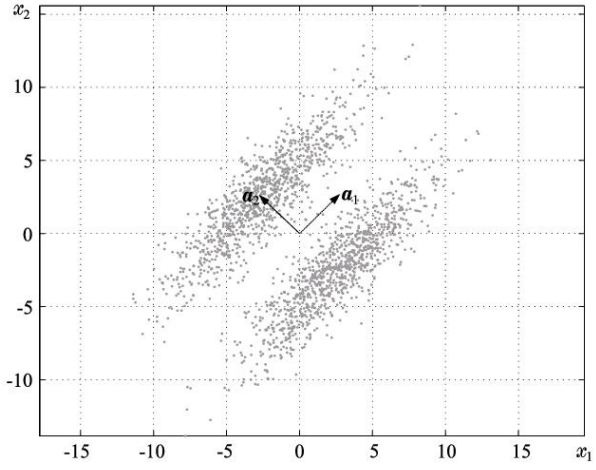
- Step 3: $W = \left(A\hat{A}\right)^T$

➢ A hierarchy of components: which ℓ to use?

In PCA one chooses the principal ones.

In ICA one can choose the ones with the least resemblance to the Gaussian pdf.

➢ Example:



The principal component is $\underline{\alpha}_1$, thus according to PCA one chooses as y the projection of $\underline{x}$ into $\underline{\alpha}_2$. According to ICA, one chooses as y the projection on $\underline{\alpha}_1$. This is the least Gaussian. Indeed:

$$\kappa_4(y_1) = -1.7$$
$$\kappa_4(y_2) = 0.1$$

Observe that across $\underline{\alpha}_2$, the statistics is bimodal. That is, no resemblance to Gaussian.

تولید ویژگی

۵

تجزیه به مقادیر تکین (SVD)

Singular Value Decomposition

A is $m \times n$

A = (orthogonal) (diagonal) (orthogonal)

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

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} = \begin{bmatrix} & & \\ & U & \\ & & \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} & & \\ & V^T & \\ & & \end{bmatrix}$$

$$A = U\Sigma V^T \rightarrow A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$$

An Application of the SVD in Image Processing

Suppose a satellite takes a picture,
and wants to send it to earth.
The picture may contain 1000 by 1000 “pixels”
(little squares each with a definite color.)
We can code the colors,
in a range between black and white,
and send back 1,000,000 numbers

picture

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

matrix

An Application of the SVD in Image Processing

It is better to find the essential information in the 1000 by 1000 matrix, and send only that.

Suppose we know the SVD.
The key is in the singular values.

Typically, some are significant and others are extremely small.

If we keep 60 and throw away 940, then we send only the corresponding 60 columns of U, and V.

The other 940 columns are multiplied by small singular values that are being ignored. In fact, we can do the matrix multiplication as columns times rows:

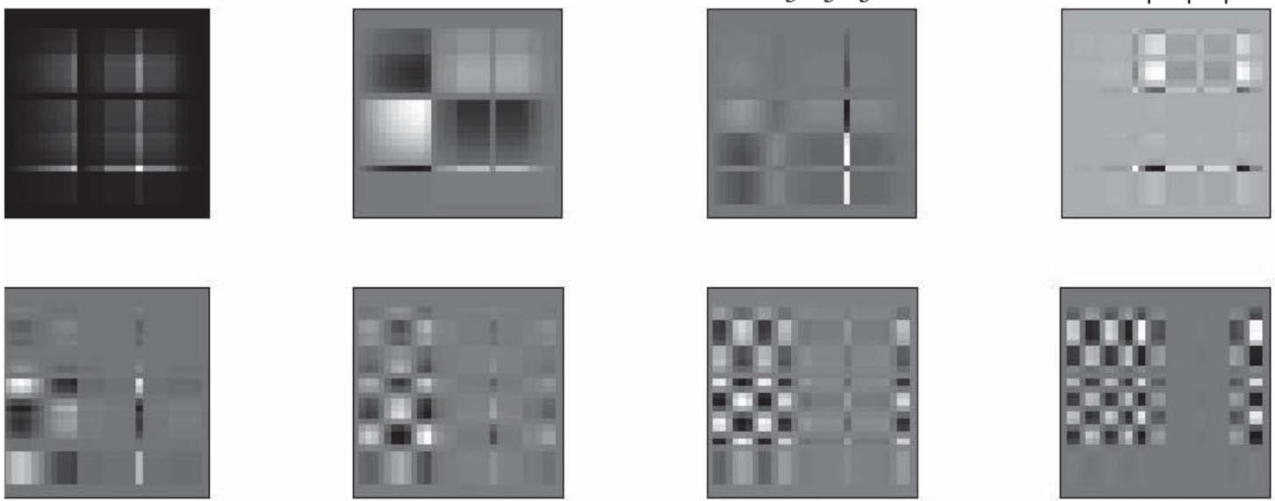
$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$$

If only 60 terms are kept, we send 60×2000 numbers instead of a million.

An Application of the SVD in Image Processing

The SVD of a 32-times-32 digital image A is computed:

$\sigma_1 u_1 v_1^T$ $\sigma_2 u_2 v_2^T$ $\sigma_3 u_3 v_3^T$ $\sigma_4 u_4 v_4^T$



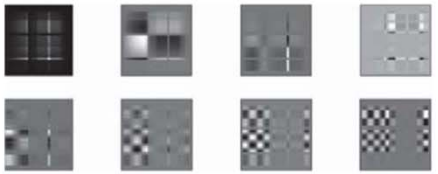
$\sigma_5 u_5 v_5^T$ $\sigma_6 u_6 v_6^T$ $\sigma_7 u_7 v_7^T$ $\sigma_8 u_8 v_8^T$

An Application of the SVD in Image Processing

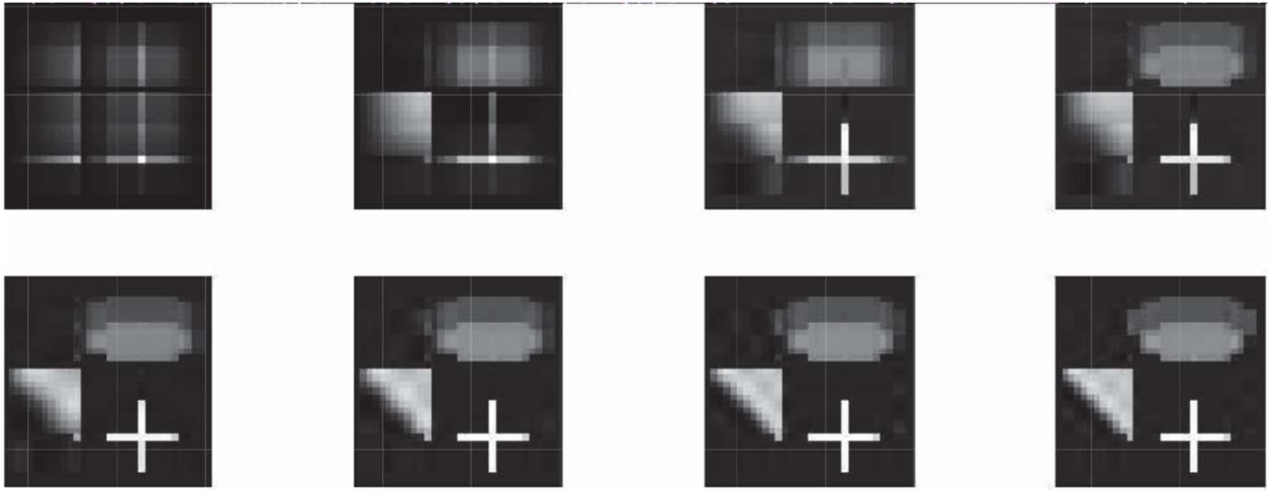
$$A_s = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_s u_s v_s^T$$

$$s \le r$$

A_1



A_2 A_3 A_4



A_5 A_6 A_7 A_8

How to compute SVD (by hand)

Eigenvalue Decomposition

If A is real n-by-n matrix, then

$$A = S \Sigma S^{-1}$$

$S = [S_1, S_2, \cdots, S_m]$ eigenvectors

$\Sigma = diag(\lambda_1, \lambda_2, \cdots, \lambda_n)$

If A is symmetric

$$A = Q \Sigma Q^T \qquad Q^T Q = I$$

Example:

$$\begin{matrix} A & & Q & & \Sigma & & Q^T \end{matrix}$$

$$\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

How to compute SVD (by hand)

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

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

$$AA^T = (U\Sigma V^T)(V\Sigma^T U^T)$$

$$AA^T = U\Sigma\Sigma^T U^T \qquad AA^T \text{ is symmetric}$$

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

$$\sigma(A) = \sqrt{\lambda(AA^T)}$$

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

$A^T A$ is symmetric

$$\sigma(A) = \sqrt{\lambda(A^T A)}$$

Theorem: The nonzero singular values of A are the square roots of the nonzero eigenvalues of A*A or AA*. (These matrices have the same nonzero eigenvalues)

How to compute SVD (by hand)

$$A = U\Sigma V^T \rightarrow AA^T = (U\Sigma V^T)(V\Sigma^T U^T) \rightarrow AA^T = U\Sigma\Sigma^T U^T$$

$$A^T = V\Sigma^T U^T \rightarrow A^T A = (V\Sigma^T U^T)(U\Sigma V^T) \rightarrow A^T A = V\Sigma^T\Sigma V^T$$

$$\sigma(A) = \sqrt{\lambda(A^T A)} \qquad \sigma(A) = \sqrt{\lambda(AA^T)}$$

Example:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \rightarrow W = A^T A = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \rightarrow \det(W - \lambda I) = \begin{vmatrix} 2-\lambda & 2 \\ 2 & 2-\lambda \end{vmatrix} = 0$$

$$\lambda_1 = 4,\ \lambda_2 = 0 \rightarrow \sigma_1 = 2,\ \sigma_2 = 0 \rightarrow v_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}, v_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix} \rightarrow u_1 = \frac{1}{\sigma_1} A v_1 = \frac{1}{2\sqrt{2}} A \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

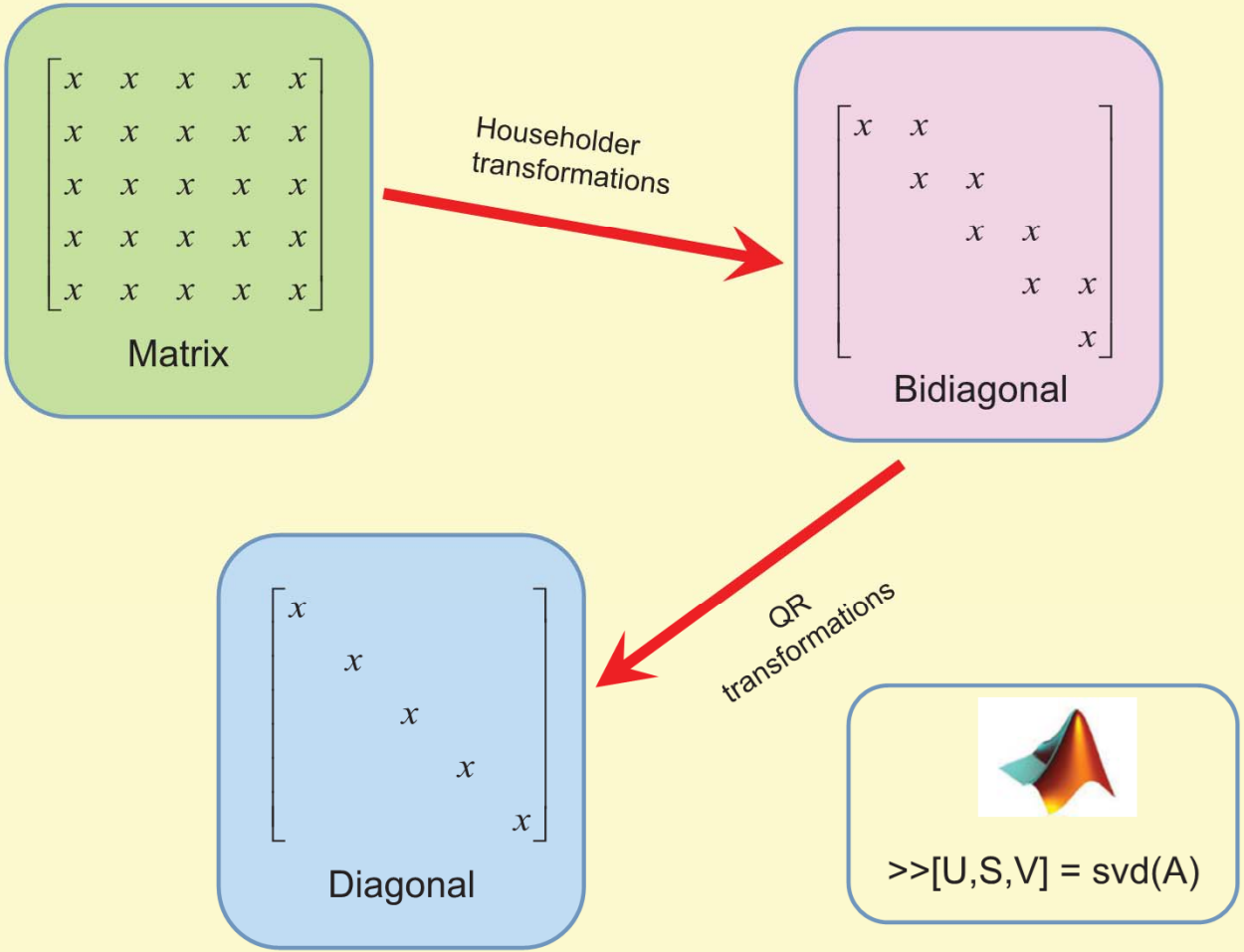
$$u_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix}$$

u_2, u_3 orthonormal basis for $\text{Null}(AA^T)$ $\quad AA^T = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow u_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

Range(A) Rank(A) Null(A)

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

How to compute SVD (Algorithm)



Singular Value Decomposition

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} = \begin{bmatrix} & & \\ & U & \\ & & \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} & & \\ & V^T & \\ & & \end{bmatrix}$$

If A is a square matrix then

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$

$$\|A\|_2 = \sigma_1$$

If A is a square matrix then

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$

$$\|A\|_F^2 = \sigma_1^2 + \sigma_2^2 + \cdots + \sigma_r^2$$

Example:

$$A = \begin{bmatrix} .96 & 1.72 \\ 2.28 & .96 \end{bmatrix} = U\Sigma V^T = \begin{bmatrix} .6 & -.8 \\ .8 & .6 \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} .8 & .6 \\ .6 & -.8 \end{bmatrix}^T.$$

$$\|A\|_2 = 3 \qquad \|A\|_F = \sqrt{10}$$

Singular Value Decomposition

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} = \begin{bmatrix} & & \\ & U & \\ & & \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} & & \\ & V^T & \\ & & \end{bmatrix}$$

If A is a square symmetric matrix then the singular values of A are the absolute values of the eigenvalues of A.

$$A = Q \Sigma Q^T = Q |\Sigma| \operatorname{sign}(\Sigma) Q^T$$

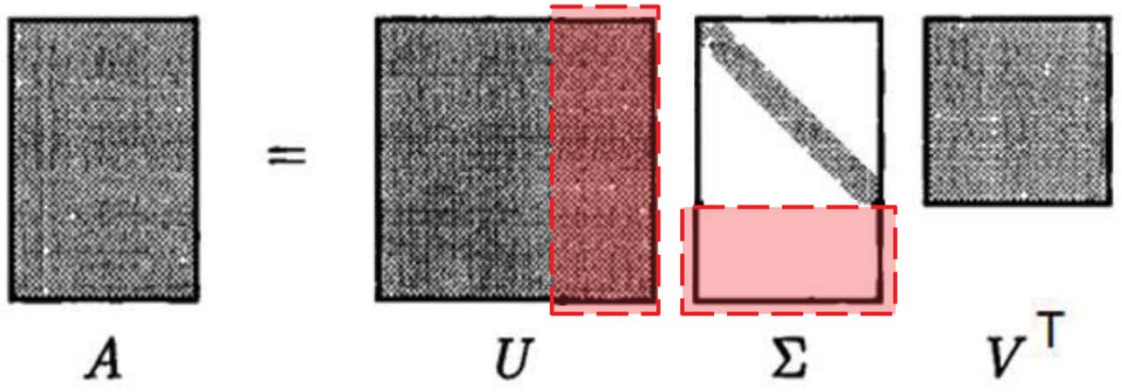
If A is a square matrix then

$$|\det(A)| = \prod_{i=1}^{n} \sigma_i$$

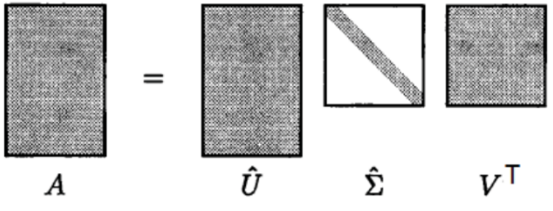
SVD and Eigenvalue Decomposition

SVD $A = U \Sigma V^T$	Eigen Decomp $A = S \Sigma S^{-1}$
Uses two different bases U, V	Uses just one (eigenvectors)
Uses orthonormal bases	Generally is not orthogonal
All matrices (even rectangular)	Not all matrices (even square) (only diagonalizable)

Reduced SVD



$$A = \hat{U} \quad \hat{\Sigma} \quad V^T$$



Reduced SVD

Singular Value Decomposition

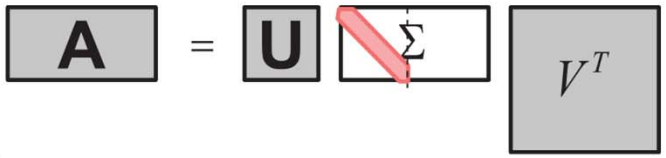
Example : Luke Olson\.illinois

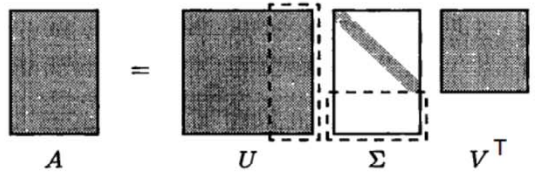
svd_test.m

iguana.jpg

```
1  clear;
2
3  I = im2double(imread('iguana.jpg'));
4
5  [U,S,V]=svd(I);
6
7  J = zeros(size(I));
8
9  figure(2);clf;
10 imshow(I)
11
12 figure(1);clf;
13 for j=1:nnz(S)
14     Itmp = S(j,j)*U(:,j)*V(:,j).';
15     J = J + Itmp;
16
17     figure(1);
18     imshow(J)
19     title(['using vector k = ' num2str(j) ', and \sigma = ' num2str(S(j,j))]);
20     pause;
21 end
22
```

Singular Value Decomposition





Theorem: (Singular Value Decomposition) SVD

If A is real m-by-n matrix, then there exist orthogonal matrices

$$U = [u_1, u_2, \cdots, u_m] \in R^{m \times m} \qquad V = [v_1, v_2, \cdots, v_n] \in R^{n \times n}$$

such that

$$A = U \Sigma V^T \qquad \Sigma = diag(\sigma_1, \cdots, \sigma_p)$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$ $\qquad p = \min(m,n)$

Proof:

Singular Value Decomposition

First approximation to A is

$$A_1 = \sigma_1 u_1 v_1^T$$

second approximation to A is

$$A_2 = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T$$

$$\cdots$$

$$A_\mu = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_\mu u_\mu v_\mu^T$$

$$\cdots$$

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$$

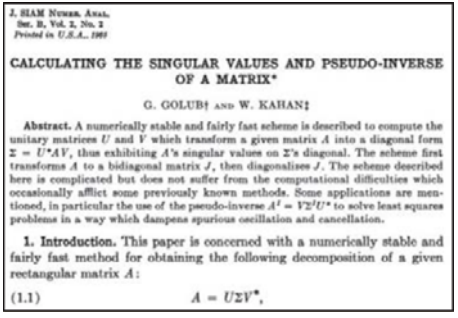
Theroem : For any μ with $0 \le \mu \le r$, the matrix A_μ also satisfies

$$\left\| A - A_\mu \right\|_F = \inf_{\substack{B \in R^{m \times n} \\ rank(B) \le \mu}} \left\| A - B \right\|_F = \sqrt{\sigma_{\mu+1}^2 + \cdots + \sigma_r^2}$$

Singular Value Decomposition

Trefethen (Textbook author):

- The SVD was discovered independently by Beltrami(1873) and Jordan(1874) and again by Sylvester(1889).
- The SVD did not become widely known in applied mathematics until the late 1960s, when Golub and others showed that it could be computed effectively.



J. SIAM NUMER. ANAL.
Ser. B, Vol. 2, No. 2
Printed in U.S.A., 1965

CALCULATING THE SINGULAR VALUES AND PSEUDO-INVERSE OF A MATRIX*

G. GOLUB† AND W. KAHAN‡

Abstract. A numerically stable and fairly fast scheme is described to compute the unitary matrices U and V which transform a given matrix A into a diagonal form $\Sigma = U^*AV$, thus exhibiting A's singular values on Σ's diagonal. The scheme first transforms A to a bidiagonal matrix J, then diagonalizes J. The scheme described here is complicated but does not suffer from the computational difficulties which occasionally afflict some previously known methods. Some applications are mentioned, in particular the use of the pseudo-inverse $A^\dagger = V\Sigma^\dagger U^*$ to solve least squares problems in a way which dampens spurious oscillation and cancellation.

1. Introduction. This paper is concerned with a numerically stable and fairly fast method for obtaining the following decomposition of a given rectangular matrix A:

(1.1) $$A = U\Sigma V^*,$$

Cleve Moler (invented MATLAB, co-founded MathWorks):

Gene Golub has done more than anyone to make the singular value decomposition one of the most powerful and widely used tools in modern matrix computation.

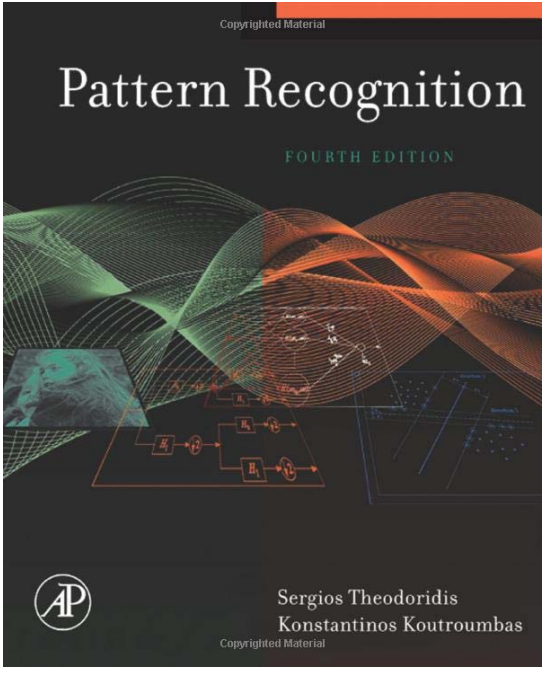
In later years he drove a car with the license plate:

تولید ویژگی

۶

منابع

منبع اصلی



S. Theodoridis, K. Koutroumbas,
Pattern Recognition,
Fourth Edition, Academic Press, 2009.

Chapter 6

CHAPTER

Feature Generation I: Data Transformation and Dimensionality Reduction

6

6.1 INTRODUCTION

Feature generation is of paramount importance in any pattern recognition task. Given a set of measurements, the goal is to discover compact and informative representations of the obtained data. A similar process is also taking place in the human perception apparatus. Our mental representation of the world is based on a relatively small number of perceptually relevant features. These are generated after processing a large amount of sensory data, such as the intensity and the color of the pixels of the images sensed by our eyes, and the power spectra of the sound signals sensed by our ears.

The basic approach followed in this chapter is to transform a given set of measurements to a new set of features. If the transform is suitably chosen, transform domain features can exhibit high *information packing* properties compared with the original input samples. This means that most of the classification-related information is "squeezed" in a relatively small number of features, leading to a reduction of the necessary feature space dimension. Sometimes we refer to such processing tasks as *dimensionality reduction* techniques.

The basic reasoning behind transform-based features is that an appropriately chosen transform can exploit and remove information redundancies, which usually exist in the set of samples obtained by the measuring devices. Let us take for example an image resulting from a measuring device, for example, X-rays or a camera. The pixels (i.e., the input samples) at the various positions in the image have a large degree of correlation, due to the internal morphological consistencies of real-world images that distinguish them from noise. Thus, if one uses the pixels as features, there will be a large degree of redundant information. Alternatively, if one obtains the Fourier transform, for example, of a typical real-world image, it turns out that most of the energy lies in the low-frequency components, due to the high correlation between the pixels' gray levels. Hence, using the Fourier coefficients as features seems a reasonable choice, because the low-energy, high-frequency coefficients can be neglected, with little loss of information. In this chapter we will

323