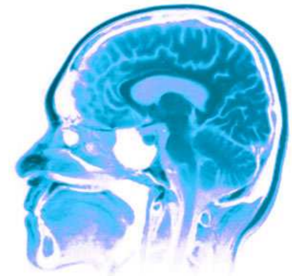
CPSC540



Linear dimensionality reduction

Nando de Freitas
September, 2011
University of British Columbia

Outline

We introduce the Singular Value Decomposition (SVD). This is a matrix factorization that has many applications, including:

- Information retrieval,
- Least-squares problems,
- Image processing,
- Dimensionality reduction.

A video can be treated as a matrix that can be decomposed

(a) Original frames (b) Low-rank $\hat{L}$ (c) Sparse $\hat{S}$

[Candes et al, 2009]

Eigenvalue decomposition

Let $\mathbf{A}$ be an $m \times m$ matrix of reals; that is $\mathbf{A} \in \mathbb{R}^{m \times m}$. If we place the eigenvalues of $\mathbf{A}$ into a diagonal matrix $\mathbf{\Lambda}$ and gather the eigenvectors into a matrix $\mathbf{Q}$, then the eigenvalue decomposition of $\mathbf{A}$ is given by

$$\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}.$$

SVD decomposition

$\mathbf{A} \in \mathbb{R}^{m \times n}$

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

$\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ is diagonal with positive entries (singular values in the diagonal).
$\mathbf{U} \in \mathbb{R}^{m \times n}$ has orthonormal columns.
$\mathbf{V} \in \mathbb{R}^{n \times n}$ has orthonormal columns and rows.
That is, $\mathbf{V}$ is an orthogonal matrix, so $\mathbf{V}^{-1} = \mathbf{V}^T$.

SVD

The equations relating the right singular values $\{\mathbf{v}_j\}$ and the left singular vectors $\{\mathbf{u}_j\}$ are

$$\mathbf{A}\mathbf{v}_j = \sigma_j \mathbf{u}_j \qquad j = 1, 2, \ldots, n$$

$$\mathbf{A}\mathbf{V} = \mathbf{U}\mathbf{\Sigma}$$

$$\mathbf{A}\begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

SVD properties

1. There is no assumption that $m \geq n$ or that $\mathbf{A}$ has full rank.

2. All diagonal elements of $\mathbf{\Sigma}$ are non-negative and in non-increasing order:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$$

   where $p = \min(m, n)$

SVD in terms of eigenvalues

Theorem 4 *The nonzero singular values of* $\mathbf{A}$ *are the (positive) square roots of the nonzero eigenvalues of* $\mathbf{A}^T\mathbf{A}$ *or* $\mathbf{A}\mathbf{A}^T$ *(these matrices have the same nonzero eigenvalues).*

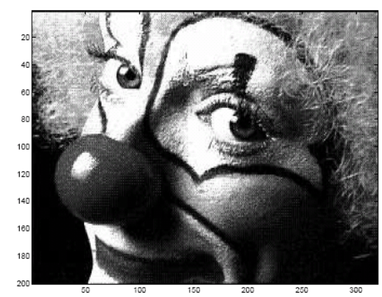
Image compression example in python

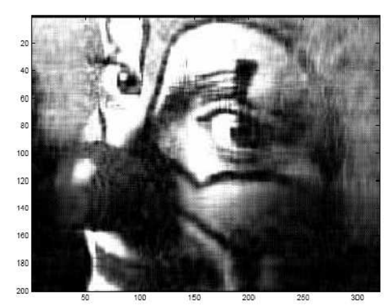
```
from scipy import *
from pylab import *

img = imread("clown.png")[:,:,0]
gray()
figure(1)
imshow(img)

m,n = img.shape
U,S,Vt = svd(img)
S = resize(S,[m,1])*eye(m,n)

k = 20
figure(2)
imshow(dot(U[:,1:k],dot(S[1:k,1:k],Vt[1:k,:])))
show()
```





The truncated SVD

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

$m = 5$

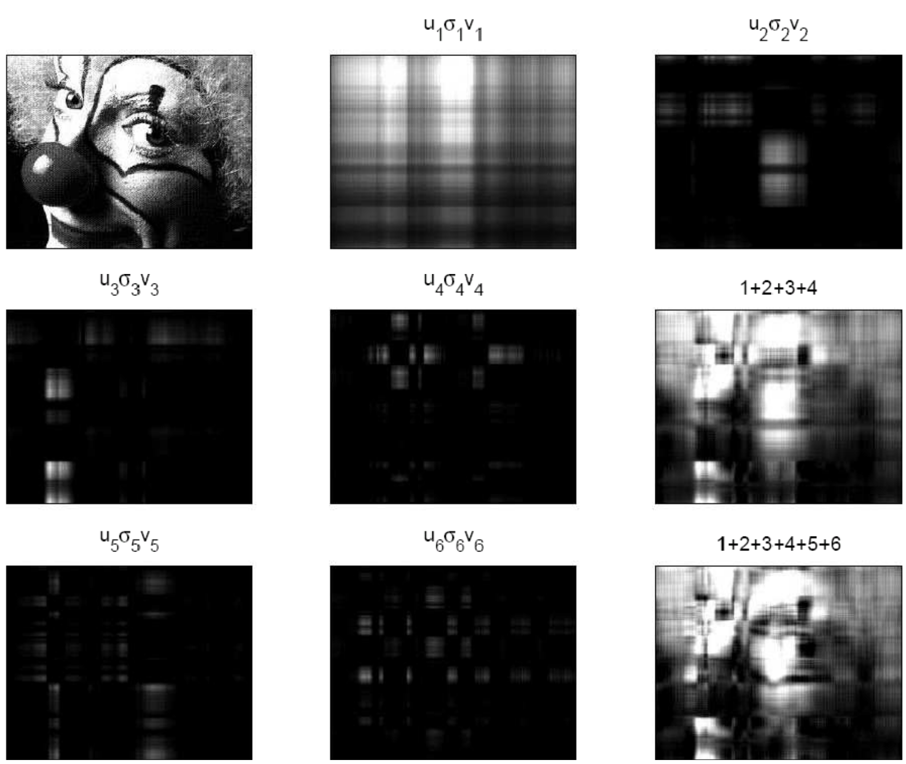
$n = 4$

$k = 2$

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

$$A = \begin{array}{cccc} u_1 & u_2 & u_3 & u_4 \\ \end{array} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_4 \end{bmatrix} \begin{array}{c} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \end{array} \begin{array}{c} v_1^T \\ v_2^T \\ v_3^T \\ v_4^T \end{array}$$

$$A_K = u_1 \sigma_1 v_1^T + u_2 \sigma_2 v_2^T \quad ; \quad K = 2$$



Smaller eigenvectors capture high frequency variations (small brush-strokes).

Image compression example

The code:

- loads a clown image into a 200 by 320 array A,
- displays the image in one figure,
- performs a singular value decomposition on A,
- displays the image obtained from a rank-20 SVD approximation of A in another figure.

The original storage requirements for A are:

The compressed representation requires:

Text retrieval: Latent semantic indexing (LSI)

The SVD can be used to cluster documents and carry out information retrieval by using concepts as opposed to exact word-matching.

This enables us to surmount the problems of synonymy (car, auto) and polysemy (money bank, river bank).

The data is available in a term-frequency (TF) matrix:

$$A = \begin{array}{c} \\ \text{word1} \\ \text{word2} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} \text{doc1} & \text{doc2} & \text{doc3} \end{array} \\ \begin{bmatrix} 1 & 3 & 2 \\ 0 & 1 & 0 \\ & & \end{bmatrix} \end{array}$$

a_{ij} = # times word i appears in document j

LSI example

$m = 5$
$n = 3$

$d_1 = \{I, \text{eat}, \text{chips}\}$

$d_2 = \{\text{computer}, \text{chips}, \text{chips}\}$

$d_3 = \{\text{intel}, \text{computer}, \text{chips}\}$

$$A = \begin{array}{c|c} \begin{matrix} d_1 & d_2 & d_3 \end{matrix} & \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} & \begin{matrix} I \\ \text{eat} \\ \text{chips} \\ \text{computer} \\ \text{intel} \end{matrix} \end{array}$$

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

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

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

$$\Sigma^{-1} U^T A = \Sigma^{-1} \Sigma V^T$$

$$\boxed{V^T = \Sigma^{-1} U^T A}$$

Truncated SVD for LSI

If we truncate the approximation to the k-largest singular values, we have

$$\mathbf{A} = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^T$$

So

$$\mathbf{V}_k^T = \mathbf{\Sigma}_k^{-1} \mathbf{U}_k^T \mathbf{A}$$

In English, $\mathbf{A}$ is projected to a lower-dimensional space spanned by the k singular vectors $\mathbf{U}_k$ (eigenvectors of $\mathbf{A}\mathbf{A}^T$).

Part I: Building the search engine

Let $k=2$

$$\begin{array}{ccc} \text{doc1} & \text{doc2} & \text{doc3} \end{array}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_3 \end{bmatrix} \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

U_k $\quad$ Σ_k $\quad$ V_k^T

$$\underset{5\times3}{A} \approx \underbrace{\underset{5\times2}{U_k}\ \underset{2\times2}{\Sigma_k}\ \underset{2\times3}{V_k^T}}_{5\times3}$$

3 docs with 5 words $\longrightarrow$ V_k^T $\quad$ 3 docs with 2 features

$$A \to \boxed{\Sigma_k^{-1} U_k^T} \to V_k^T$$

Part II: Querying the search engine

To carry out **retrieval**, a **query** $\mathbf{q} \in \mathbb{R}^n$ is first projected to the low-dimensional space:

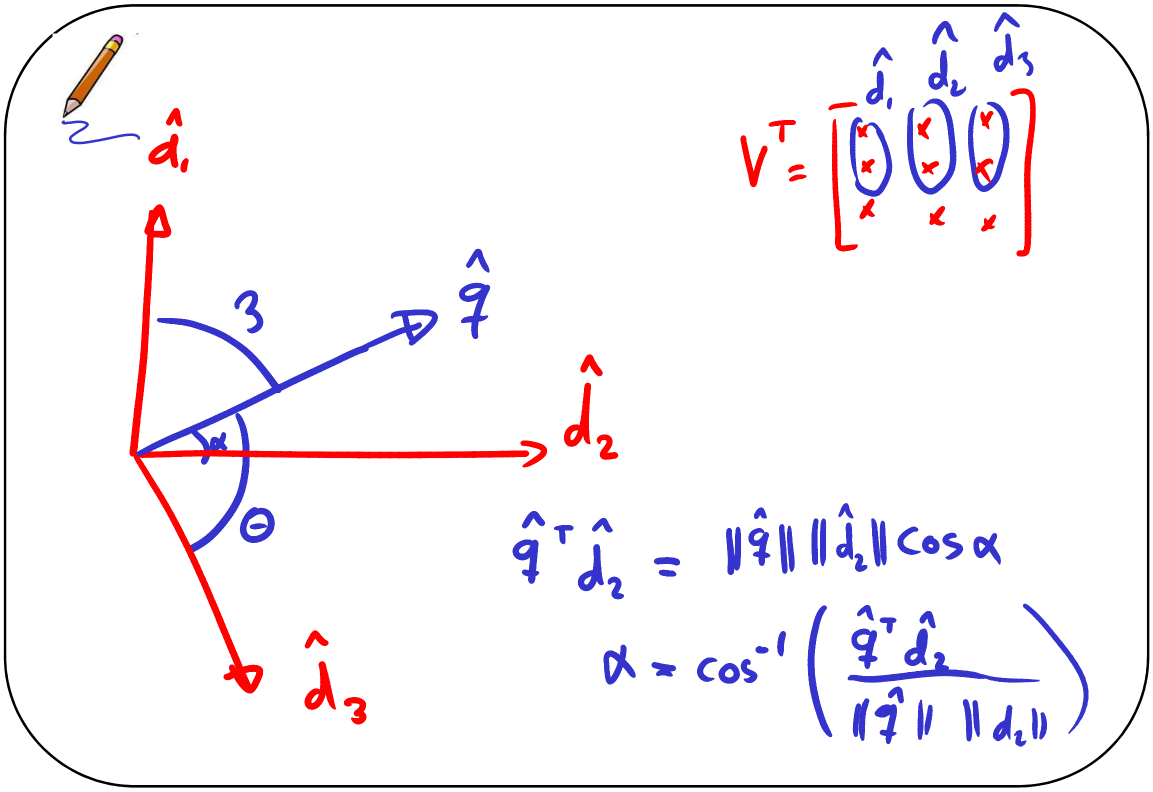
$$\underset{2\times1}{\widehat{\mathbf{q}}_k} = \underset{2\times5}{\mathbf{\Sigma}_k^{-1}\mathbf{U}_k^T}\underset{5\times1}{\mathbf{q}}$$

And then we measure the angle between $\widehat{\mathbf{q}}_k$ and the $\mathbf{v}_k$.

$q =$ "chips" $\Rightarrow$ $q = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$

$$\underset{5\times1}{q} \to \underset{2\times5}{\boxed{\Sigma_k^{-1} U_k^T}} \to \underset{2\times1}{\hat{q}_k}$$

Part II: Querying the search engine



TF-IDF

The *term frequency* of word w in document d is equal to the total number of times w appears in d, divided by the total number of words in d.
The *inverse document frequency* of word w is based on the logarithm of the inverse frequency of the word in the corpus. If the number of documents in the corpus is D and the number of documents the word appears in is D_w, then

$$idf_w = \log \frac{D}{1 + D_w}$$

and for w and d, we can compute

$$tfidf_{w,d} = tf_{w,d} idf_w$$