Linear Algebra Methods for Data Mining

Saara Hyvönen, Saara.Hyvonen@cs.helsinki.fi

Spring 2007

The Singular Value Decomposition (SVD)

Matrix decompositions

- We wish to **decompose** the matrix $\mathbf{A}$ by writing it as a product of two or more matrices:

$$\mathbf{A}_{m \times n} = \mathbf{B}_{m \times k} \mathbf{C}_{k \times n}, \quad \mathbf{A}_{m \times n} = \mathbf{B}_{m \times k} \mathbf{C}_{k \times r} \mathbf{D}_{r \times n}$$

- This is done in such a way that the right side of the equation yields some useful information or insight to the nature of the data matrix $\mathbf{A}$.

- Or is in other ways useful for solving the problem at hand.

Example (SVD):

customer \ day	Wed	Thu	Fri	Sat	Sun
ABC Inc.	1	1	1	0	0
CDE Co.	2	2	2	0	0
FGH Ltd.	1	1	1	0	0
NOP Inc.	5	5	5	0	0
Smith	0	0	0	2	2
Brown	0	0	0	3	3
Johnson	0	0	0	1	1

$$
\begin{pmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{pmatrix}
=
\begin{pmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{pmatrix}
\times
\begin{pmatrix}
9.64 & 0 \\
0 & 5.29
\end{pmatrix}
\times
$$

$$
\begin{pmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{pmatrix}
$$

The Singular Value Decomposition

- Any $m \times n$ matrix $\mathbf{A}$, with $m \geq n$, can be factorized

$$\mathbf{A} = \mathbf{U} \begin{pmatrix} \mathbf{\Sigma} \\ \mathbf{0} \end{pmatrix} \mathbf{V}^T,$$

where $\mathbf{U} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are orthogonal, and $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ is diagonal:

$$\mathbf{\Sigma} = \mathsf{diag}(\sigma_1, \sigma_2, ..., \sigma_n), \quad \sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0.$$

- "Skinny version": $\mathbf{A} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{V}^T$, $\mathbf{U}_1 \in \mathbb{R}^{m \times n}$.

Singular values and singular vectors

- The diagonal elements σ_j of $\mathbf{\Sigma}$ are the **singular values** of the matrix $\mathbf{A}$.

- The columns of $\mathbf{U}$ and $\mathbf{V}$ are the **left singular vectors** and **right singular vectors** respectively.

- Equivalent form of SVD:

$$\mathbf{A}\mathbf{v}_j = \sigma_j \mathbf{u}_j,$$
$$\mathbf{A}^T \mathbf{u}_i = \sigma_j \mathbf{v}_j.$$

Outer product form:

Start from skinny version of SVD:

$$\mathbf{A} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{V}^T = (\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_n) \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_n^T \end{pmatrix}$$

$$= (\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_n) \begin{pmatrix} \sigma_1 \mathbf{v}_1^T \\ \sigma_2 \mathbf{v}_2^T \\ \vdots \\ \sigma_n \mathbf{v}_n^T \end{pmatrix} = \sum_{i=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \ldots$$

to get the **outer product form**

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

This is a sum of rank one matrices!!
(Each term $\sigma_i \mathbf{u}_i \mathbf{v}_i^T$ in the sum is a rank one matrix).

Example

```
A=      1     1
        1     2
        1     3
        1     4
[U,S,V]=svd(A)

U=   -0.2195    0.8073    0.0236    0.5472
     -0.3833    0.3912   -0.4393   -0.7120
     -0.5472   -0.0249    0.8079   -0.2176
     -0.7110   -0.4410   -0.3921    0.3824

S=    5.7794         0                  V=   -0.3220    0.9467
           0    0.7738                       -0.9467   -0.3220
           0         0
           0         0
```

```
[U,S,V]=svd(A,0)

U=   -0.2195    0.8073
     -0.3833    0.3912
     -0.5472   -0.0249
     -0.7110   -0.4410

S=    5.7794         0
           0    0.7738

V=   -0.3220    0.9467
     -0.9467   -0.3220
```

Rank deficient case

```
%A is not of full rank
%i.e. the columns of A are not linearly independent
A(:,3)=A(:,1)+A(:,2)*0.5

A=     1.0000    1.0000    1.5000
       1.0000    2.0000    2.0000
       1.0000    3.0000    2.5000
       1.0000    4.0000    3.0000

[U,S,V]=svd(A,0)

U=    -0.2612   -0.7948    0.0985
      -0.4032   -0.3708    0.2703
      -0.5451    0.0533   -0.8360
      -0.6871    0.4774    0.4672
```

```
S=    7.3944         0         0
           0    0.9072         0
           0         0    0.0000

V=   -0.2565   -0.6998    0.6667
     -0.7372    0.5877    0.3333
     -0.6251   -0.4060   -0.6667
```

Matrix approximation

Theorem. Let $\mathbf{U}_k = (\mathbf{u}_1\ \mathbf{u}_2\ ...\ \mathbf{u}_k)$, $\mathbf{V}_k = (\mathbf{v}_1\ \mathbf{v}_2\ ...\ \mathbf{v}_k)$ and $\mathbf{\Sigma}_k = \text{diag}(\sigma_1, \sigma_2, ..., \sigma_k)$, and define

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

Then

$$\min_{\text{rank}(\mathbf{B}) \leq k} \|\mathbf{A} - \mathbf{B}\|_2 = \|\mathbf{A} - \mathbf{A}_k\|_2 = \sigma_{k+1}.$$

What does this mean?

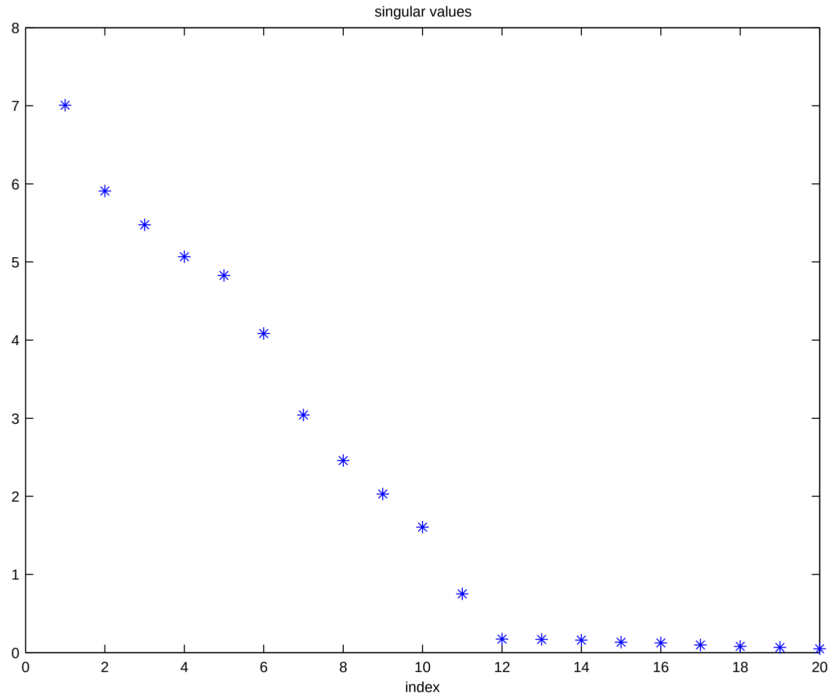
- It means, that the best approximation of rank k for the matrix $\mathbf{A}$ is

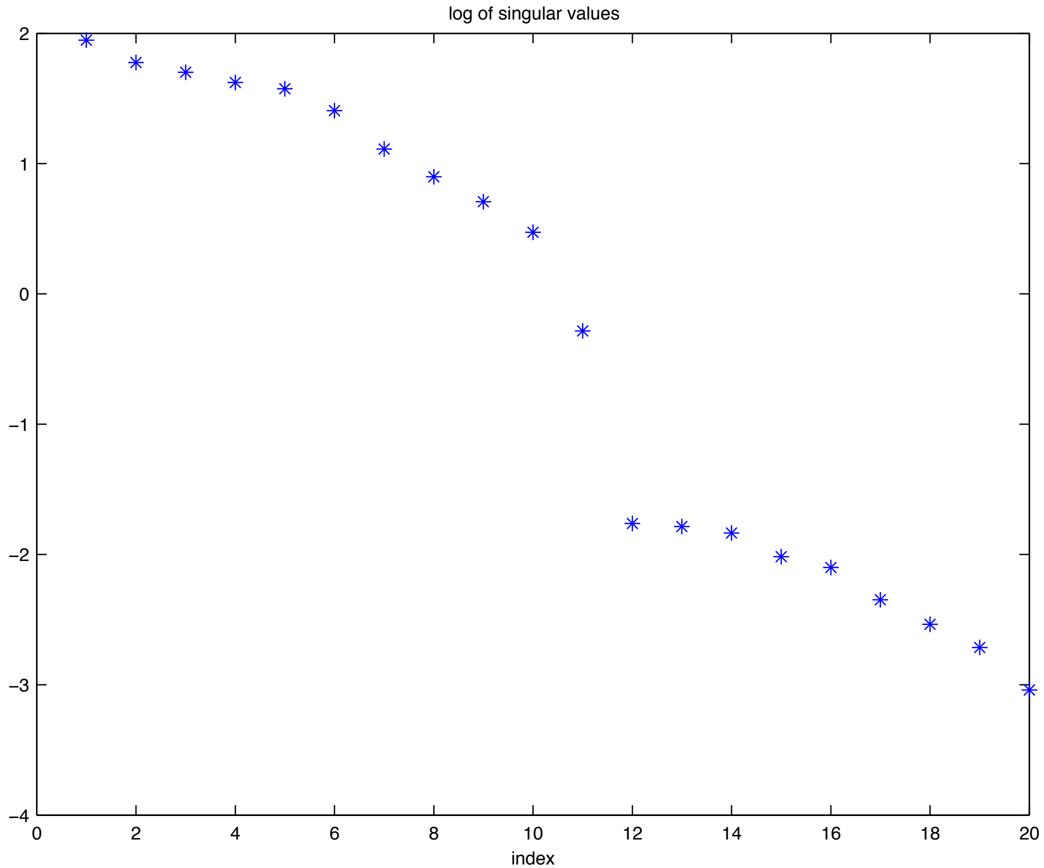
$$\mathbf{A}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^T.$$

- Useful for
  - compression
  - noise reduction
- It also means, that we can estimate the "correct" rank by looking at the singular values...

Example: noise reduction

Assume $\mathbf{A}$ is a low rank matrix plus noise: $\mathbf{A} = \mathbf{A}_k + \mathbf{N}$. Then typically singular values of $\mathbf{A}$ have the following behaviour:




log of singular values
2
1
0
−1
−2
−3
−4
0
2
4
6
8
10
12
14
16
18
20
index

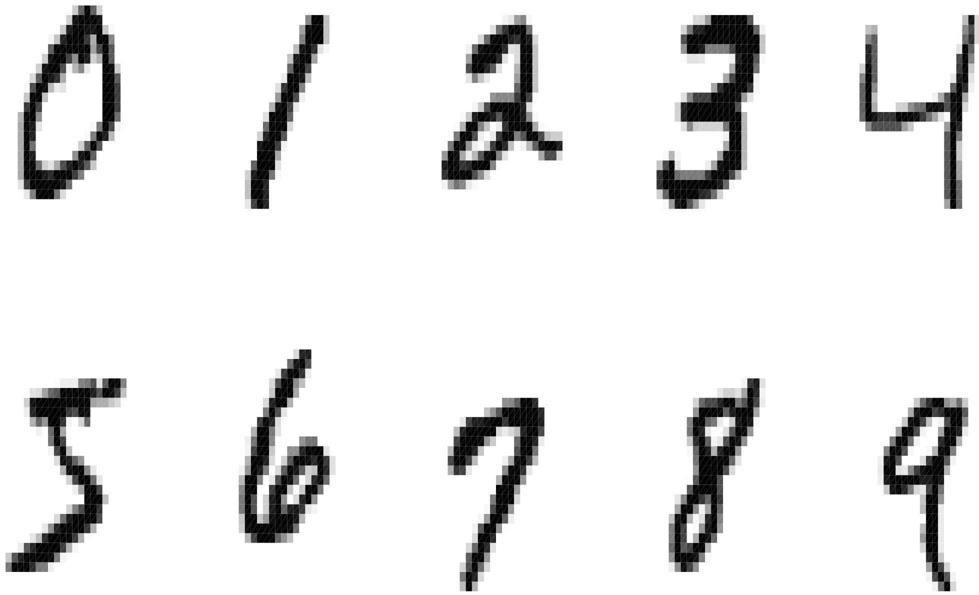
SVD is useful for:

- compression
- noise reduction
- finding "concepts" or "topics" (text mining/LSI)
- data exploration and visualizing data (e.g. spatial data/PCA)
- classification (of e.g. handwritten digits)

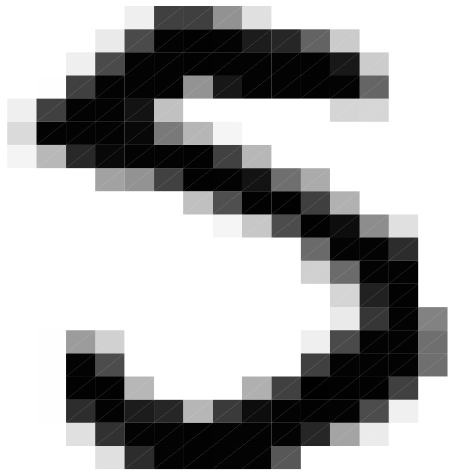
SVD appears under different names:

- Principal Component Analysis (PCA)
- Latent Semantic Indexing (LSI)/Latent Semantic Analysis (LSA)
- Karhunen-Loeve expansion/Hotelling transform (in image processing)

Example: Classification of handwritten digits



Digitized images, 28×28, grayscale. Task: classify unknown digits.



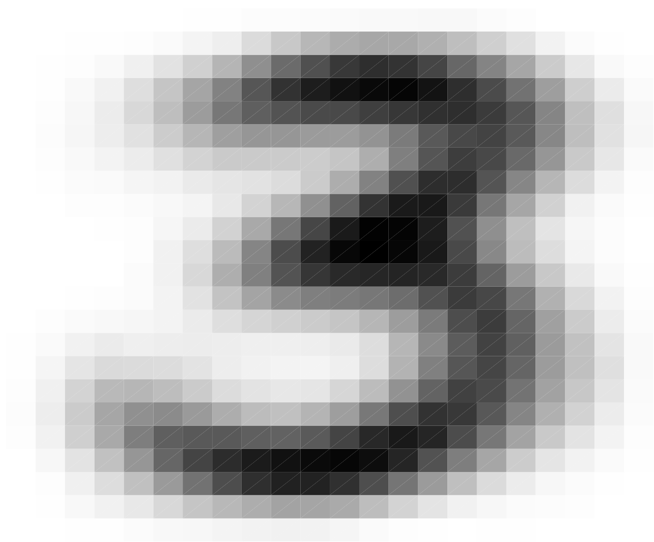
Digitized images, 28×28, grayscale.

The image can also be considered as a function of two variables: $c = c(x, y)$, where c is the intensity of the color.

- The image of one digit is a matrix (of size 28×28).
- Stacking all the columns of each image matrix above each other gives a vector (of size 784×1).
- The image vectors can then be collected into a data matrix (of size $784 \times N$, where N is the number of images).
- Distance between images: Euclidean distance in $\mathbb{R}^{784}$ or cosine distance.

Naive method

Given a data base of handwritten digits (training set), compute the mean (centroid) of all digits of one kind. Centroid of threes:



Naive method

- Given a data base of handwritten digits (training set), compute the mean (centroid) of all digits of one kind.
- Compare an unknown digit (from the test set) with all means, and classify as the digit that is closest (using e.g. euclidean or cosine distance).
- Good results for well-written digits, worse for bad digits.
- does not use any information about the variation of the digits of one kind.

Results using the naive method

- Success rate around 75%. Good... but not good enough!

- For the homework set (training set=4000, test set size = 2000) the number of misclassifications per digit look like this:

digit	0	1	2	3	4	5	6	7	8	9
missed	25	9	82	51	42	51	47	49	76	49
total	175	234	219	207	217	179	178	205	192	194

- Some digits are easier to recognize than others.

SVD

- Any matrix $\mathbf{A}$ can be written as a sum of rank 1 matrices:

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

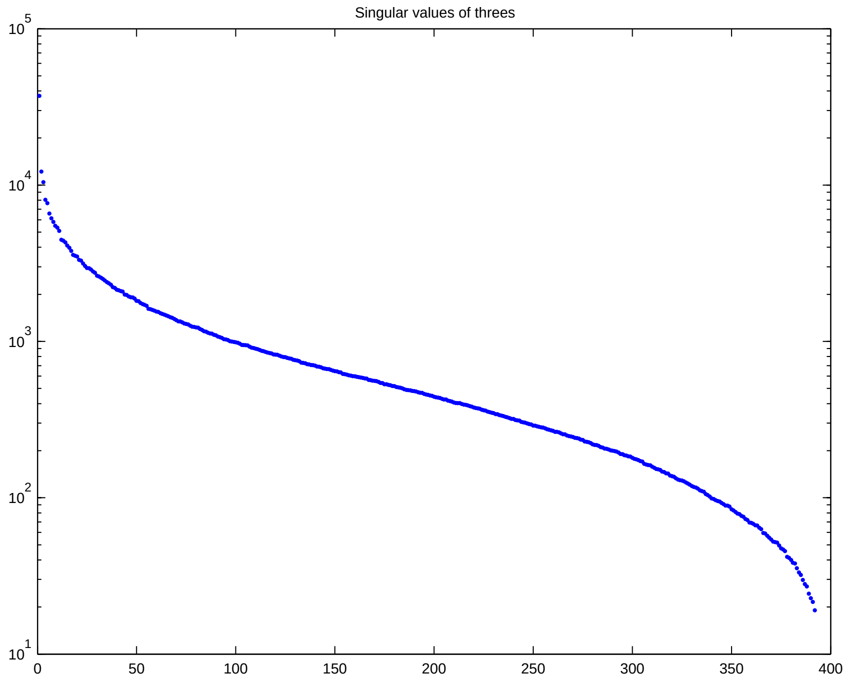
- Consider a data matrix $\mathbf{A}$,where each column of $\mathbf{A}$ is one image of a digit. Column j: $\mathbf{a}_j = \sum_{i=1}^{n} (\sigma_i v_{ij}) \mathbf{u}_i$.

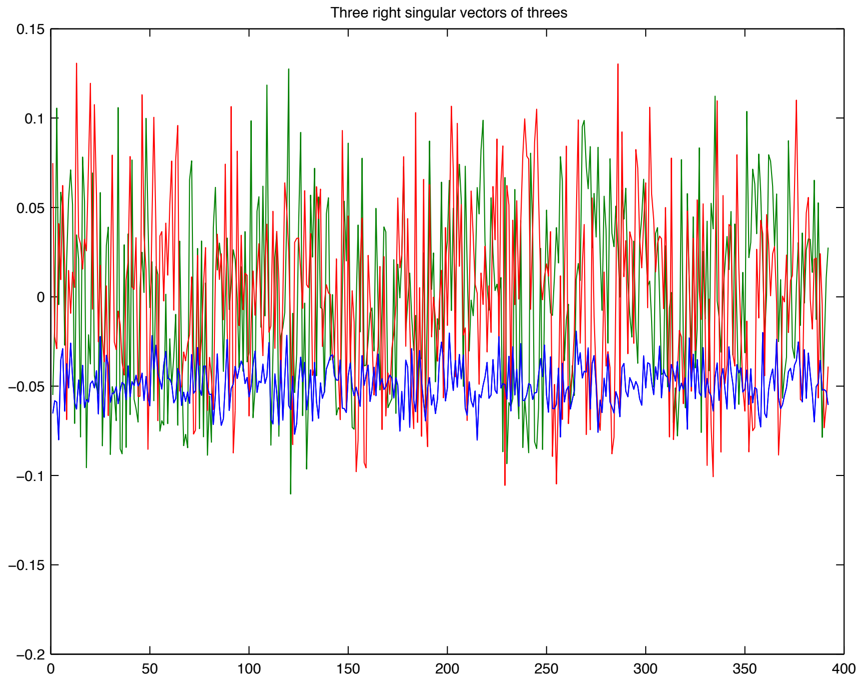
- $\mathbf{u}_j$ is the same size as $\mathbf{a}_j$. We can fold it back to get an image.

- We can think of the left singular vectors $\mathbf{u}_j$ as "singular images".

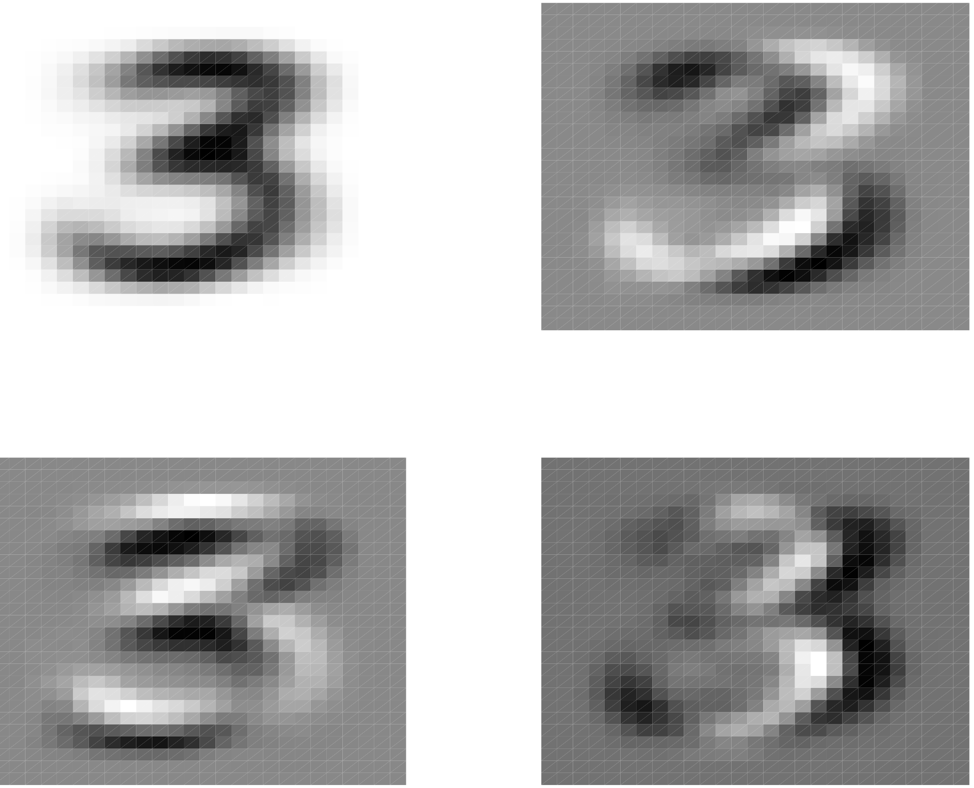
Compute the SVD of the matrix fo one digit (=3), (392 images).

Singular values and right singular vectors $\mathbf{v}_j$:





Left singular vectors or "singular images" $\mathbf{u}_j$:



Classification

Let $\mathbf{z}$ be and unknown digit, and consider the least squares problem (for some predetermined k):

$$\min_{\alpha_j} \|\mathbf{z} - \sum_{j=1}^{k} \alpha_j \mathbf{u}_j\|.$$

Then, if $\mathbf{z}$ is the same digit as that represented by the $\mathbf{u}_j$, then the residual should be small. I.e. $\mathbf{z}$ should be well approximated by some linear combination of the $\mathbf{u}_j$.

Principal components (SVD) regression

Assume

- Each digit is well characterize by a few of the first ”singular images” of its own kind.
- Each digit is NOT characterized very well by a few of the first ”singular images” of some other digit.

We can use this to classify unknown digits:

Algorithm

Training:

- For the training set of digits, compute the SVD of each set of digits of one kind. We get 10 sets of ”singular images”, one for each digit.

Classification:

1. Take an unknown digit $\mathbf{z}$. For each set of singular images $\mathbf{u}_j$, solve the least squares problem

$$\min_{\alpha_j} \|\mathbf{z} - \sum_{j=1}^{k} \alpha_j \mathbf{u}_j\|.$$

   and compute the relative residual $\|\mathbf{r}\|/\|\mathbf{z}\|$, where $\mathbf{r}$ is the residual.

2. Classify $\mathbf{z}$ to be the digit corresponding to the smallest relative residual. Or: find the smallest residual $\mathbf{r}_0$ and the next smallest residual $\mathbf{r}_1$. If $\mathbf{r}_0 \leq \tau \mathbf{r}_1$, where τ is the tolerance, then classify, else give up.

Results

To be reported by you in you homework assignment.

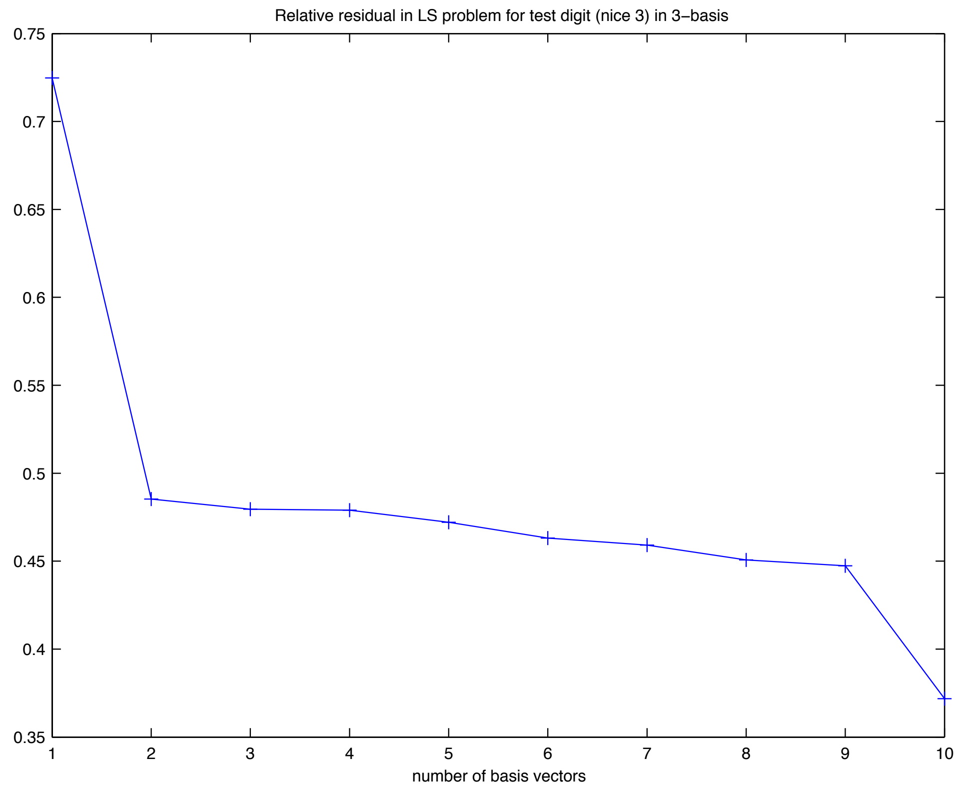
Performance better: success rate over 90%.

Still not up to the best (more complicated) methods.

Nice test digit 3 in basis 3

Original image of digit and approximations using 1,3,5,7 and 9 basis vectors in 3-basis:

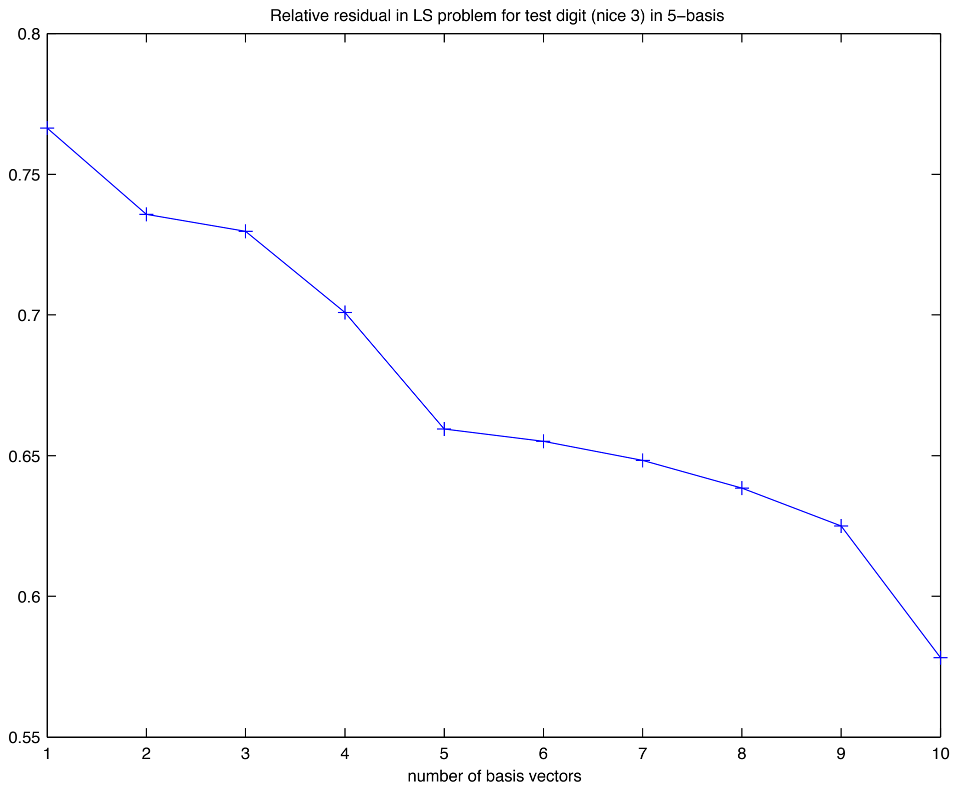
Relative residual in least squares problem for nice test digit 3 in 3-basis:



Nice test digit 3 in basis 5

Original image of digit and approximations using 1,3,5,7 and 9 basis vectors in 5-basis:

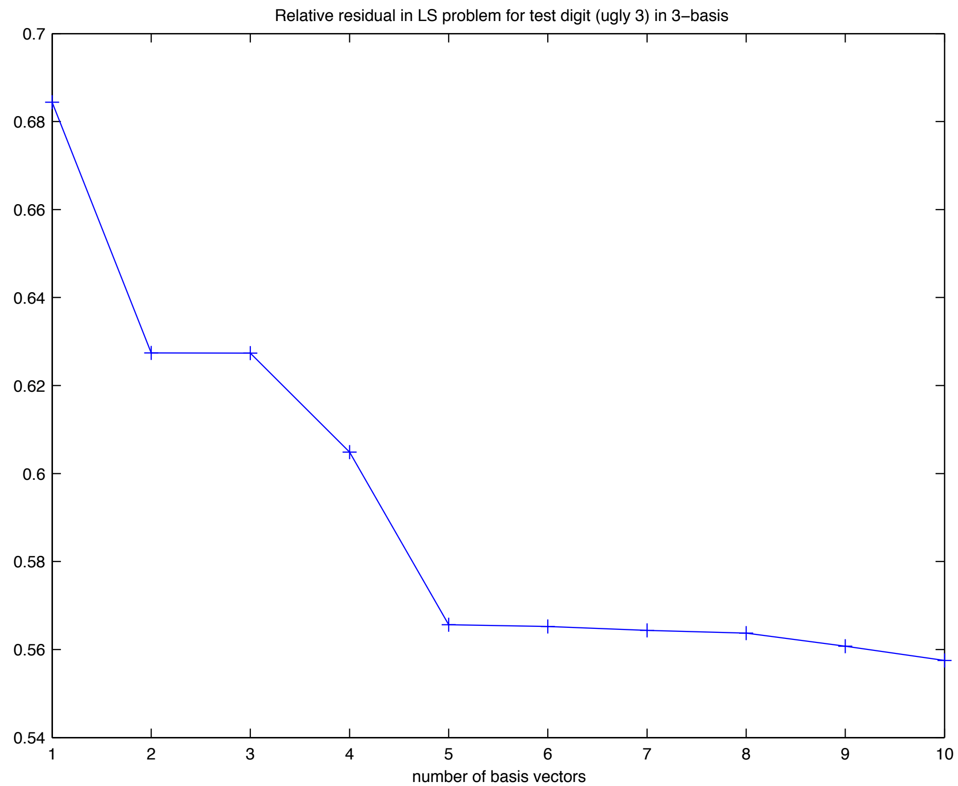
Relative residual in least squares problem for nice test digit 3 in 5-basis:



Ugly test digit 3 in basis 3

Original image of digit and approximations using 1,3,5,7 and 9 basis vectors in 3-basis:

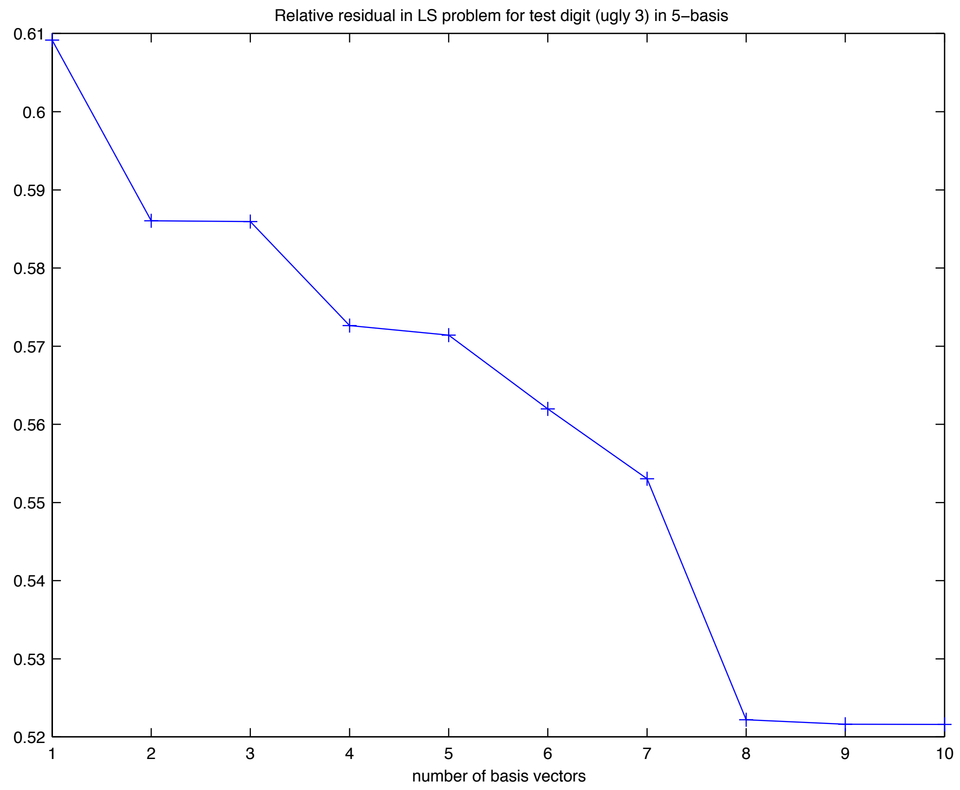
Relative residual in least squares problem for ugly test digit 3 in 3-basis:



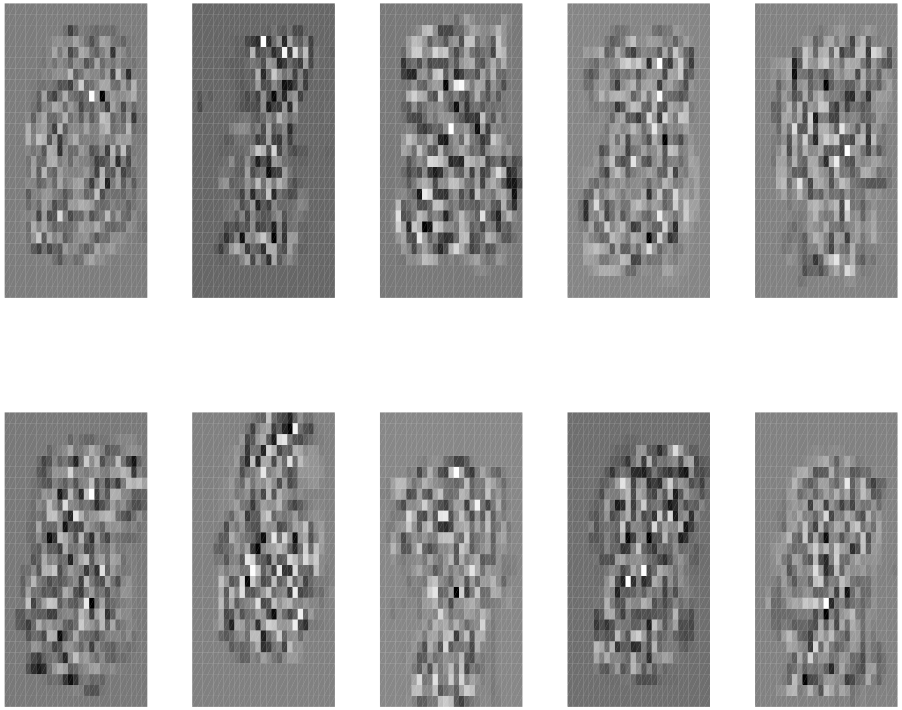
Ugly test digit 3 in basis 5

Original image of digit and approximations using 1,3,5,7 and 9 basis vectors in 5-basis:

Relative residual in least squares problem for ugly test digit 3 in 5-basis:



Conclusion: do not use many terms in the basis.



The 100th singular images of all vectors.

Work

- Training: compute SVD's of 10 matrices of dimension $m^2 \times n_i$, each image of a digit is a $m \times m$ matrix; $n_i =$ number of training digits i.

- Solve 10 least squares problems; multiply test digit by 10 matrices with orthogonal columns.

Fast test phase! Suitable for real time computations!

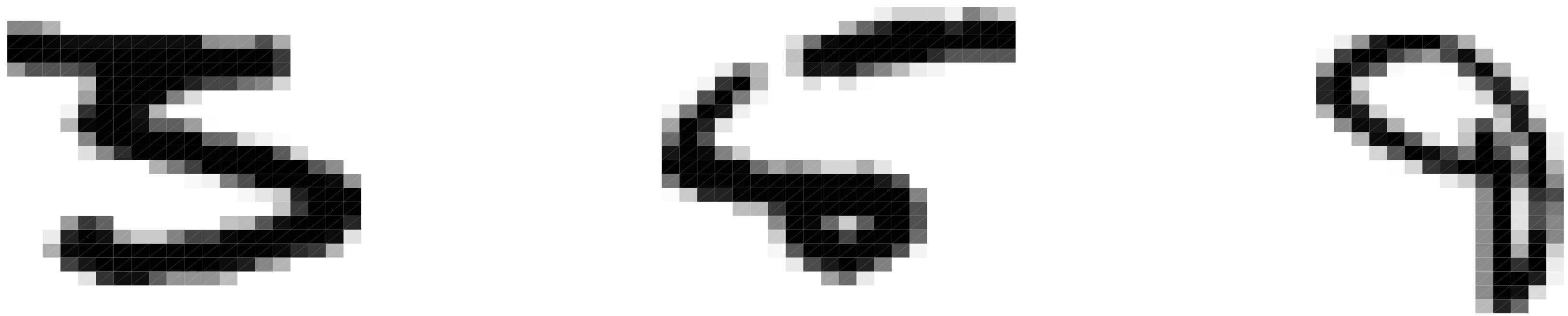
Test results

Correct classification as a function fo the number of basis vectors for each digit vector:

number of vectors	1	2	4	6	8	10
	76	82	88	90	90	91.3

How to improve performance?

Ugly digits are difficult to handle automatically.



The Singular Value Decomposition

- Any $m \times n$ matrix $\mathbf{A}$, with $m \geq n$, can be factorized

$$\mathbf{A} = \mathbf{U} \begin{pmatrix} \mathbf{\Sigma} \\ \mathbf{0} \end{pmatrix} \mathbf{V}^T,$$

where $\mathbf{U} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are orthogonal,
and $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ is diagonal:

$$\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \sigma_2, ..., \sigma_n), \quad \sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0.$$

- "Skinny version": $\mathbf{A} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{V}^T$, $\mathbf{U}_1 \in \mathbb{R}^{m \times n}$.

We can write

$$\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^T,$$

so we get

$$\mathbf{A}^T\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{\Sigma}^2.$$

Let $\mathbf{v}_j$ be the j^{th} column of $\mathbf{V}$. Write the above column by column:

$$\mathbf{A}^T\mathbf{A}\mathbf{v}_j = \sigma_j^2\mathbf{v}_j.$$

Equivalently, we can write $\mathbf{A}\mathbf{A}^T = ... = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T$ to get ($\mathbf{u}_j$ is the j^{th} column of $\mathbf{U}$)

$$\mathbf{A}\mathbf{A}^T\mathbf{u}_j = \sigma_j^2\mathbf{u}_j.$$

Singular values and vectors

- The singular values are the nonnegative square roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$, and hence uniquely determined.

- The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}^T\mathbf{A}$, arranged in the same order as the σ_j.

- The columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{A}\mathbf{A}^T$, arranged in the same order as the σ_j.

- If $\mathbf{A}$ is real, then $\mathbf{V}$, $\mathbf{\Sigma}$ and $\mathbf{U}$ can be taken to be real.

Uniqueness considerations.

- $\mathbf{\Sigma}$ is unique: the eigenvalues of $\mathbf{A}^T\mathbf{A}$ are unique, the positive square roots of them = the singular values are also unique, and the ordering

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0$$

fixes $\mathbf{\Sigma}$.

- But how about $\mathbf{U}$ and $\mathbf{V}$?

Example from lecture 4 revisited:

```
%Rank deficient case: A is not of full rank.
%The columns of A are not linearly independent.
A=[1 1 1 1;1 2 3  4]'

A=     1.0000    1.0000
       1.0000    2.0000
       1.0000    3.0000
       1.0000    4.0000

A(:,3)=A(:,1)+A(:,2)*0.5

A=     1.0000    1.0000    1.5000
       1.0000    2.0000    2.0000
       1.0000    3.0000    2.5000
       1.0000    4.0000    3.0000
```

```
%Matlab version 7
[U,S,V]=svd(A,0)

U=   -0.2612   -0.7948    0.0985
     -0.4032   -0.3708    0.2703
     -0.5451    0.0533   -0.8360
     -0.6871    0.4774    0.4672

S=    7.3944         0         0
           0    0.9072         0
           0         0    0.0000

V=   -0.2565   -0.6998    0.6667
     -0.7372    0.5877    0.3333
     -0.6251   -0.4060   -0.6667
```

```
%Matlab version 6.5
[U,S,V]=svd(A,0)

U=   -0.2612    0.7948    0.0236
     -0.4032    0.3708   -0.4393
     -0.5451   -0.0533    0.8079
     -0.6871   -0.4774   -0.3921

S=    7.3944         0         0
           0    0.9072         0
           0         0         0

V=   -0.2565    0.6998   -0.6667
     -0.7372   -0.5877   -0.3333
     -0.6251    0.4060    0.6667
```

Uniqueness of $\mathbf{U}$ and $\mathbf{V}$

- Let $\mathbf{A}$ be a $m \times n$, real valued matrix, with $m \geq n$.

- If the singular values σ_j are distinct, and $\mathbf{A} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{V}^T = \mathbf{U}_2 \mathbf{\Sigma} \mathbf{W}^T$, the following holds for the columns of $\mathbf{V}$ and $\mathbf{W}$:
  $\mathbf{w}_j = (-1)^{k_j} \mathbf{v}_j$, $k_j \in (0, 1)$.

- If $m = n =$ rank of $\mathbf{A}$, then, given $\mathbf{V}$, $\mathbf{U}$ is uniquely determined.

- But if $m > n$, $\mathbf{U}$ is never uniquely determined! (Only the k first columns are, up to a multiplication by ± 1, where $k = \text{rank}(\mathbf{A})$.)

Example: 1st singular images of handwritten digits

References

[1] Lars Eldén: Matrix Methods in Data Mining and Pattern Recognition, SIAM 2007.
item[[2]] T. Hastie, R. Tibshirani, J. Friedman: The Elements of Statistical Learning. Data mining, Inference and Prediction, Springer Verlag, New York, 2001.

[3] The data is from the MNIST database of handwritten digits, http://yann.lecun.com/exdb/mnist.