TANA15/Lecture 8 - Contents

The Singular Value Decomposition

- Definition. Computing the SVD.
- Fundamental subspaces. Linear Systems and Least Squares. Low rank approximation.

Applications

- Classification of handwritten digits.
- Total Least Squares.

The singular value decomposition

Proposition Every matrix $A \in \mathbb{R}^{m\times n}$ has a decomposition

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

where U and V are orthogonal and $\Sigma \in \mathbb{R}^{m\times n}$ is *diagonal* with diagonal elements $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(n,m)} \geq 0$

Remark The equivalent formula

$$A = \sum_{i=1}^{n} \sigma_i u_i v_i^T$$

writes A as a sum of rank one matrices.

Computing the SVD

Lemma Let $A = U\Sigma V^T \in \mathbb{R}^{m\times n}$. Then

$$A^TA = V(\Sigma^T\Sigma)V^T, \quad \text{and} \quad AA^T = U(\Sigma\Sigma^T)U^T.$$

So (σ_i^2, v_i) and (σ_i^2, u_i) are eigen pairs of A^TA and AA^T.

Remark Suggests we can compute the SVD by solving either of two symmetric eigenvalue problems.

Question How to organize the computations efficiently?

Definition A matrix B is *upper bidiagonal* if $b_{ij} = 0$ unless $j = i$ or $j = i + 1$.

Lemma If B is bidiagonal then BB^T and B^TB are tridiagonal.

Proposition Any matrix $A \in \mathbb{R}^{m\times n}$ can be reduced to bidiagonal form by $A = Q_1BQ_2^T$, where Q_1 and Q_2 are orthogonal.

Reduction to bidiagonal form

Example Suppose A is a 5×4 matrix. First select a reflection such that $H_1 A(1:5,1) = \alpha e_1$. Then

$$\tilde{H}_1 A = \tilde{H}_1 \begin{pmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{pmatrix} = \begin{pmatrix} + & + & + & + \\ 0 & + & + & + \\ 0 & + & + & + \\ 0 & + & + & + \\ 0 & + & + & + \end{pmatrix} = A_2.$$

Next select a reflection such that $H_2 A_2(1,2:4)^T = \alpha e_1$. Then

$$A_2 \tilde{H}_2^T = \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{pmatrix} \tilde{H}_2^T = \begin{pmatrix} x & + & 0 & 0 \\ 0 & + & + & + \\ 0 & + & + & + \\ 0 & + & + & + \\ 0 & + & + & + \end{pmatrix} = A_3.$$

Proceed and find reflections $H_3 A_3(2:5,2) = \alpha e_1$ and $H_4 A_4(2,3:4)^T = \alpha e_1$,

$$\tilde{H}_3 \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{pmatrix} \tilde{H}_4^T = \begin{pmatrix} x & x & 0 & 0 \\ 0 & + & + & + \\ 0 & 0 & + & + \\ 0 & 0 & + & + \\ 0 & 0 & + & + \end{pmatrix} \tilde{H}_4^T = \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & + & 0 \\ 0 & 0 & + & + \\ 0 & 0 & + & + \\ 0 & 0 & + & + \end{pmatrix}$$

Finally apply reflections H_5 and H_6 to obtain

$$\tilde{H}_6 \tilde{H}_5 \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{pmatrix} = \tilde{H}_6 \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & + & + \\ 0 & 0 & 0 & + \\ 0 & 0 & 0 & + \end{pmatrix} = \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \\ 0 & 0 & 0 & + \\ 0 & 0 & 0 & 0 \end{pmatrix} = B.$$

Have reached *bidiagonal form* after $2n-2$ Householder reflections.

The SVD Algorithm

The singular value decomposition is computed by

- Reduction to bidiagonal form $A = \bar{U} B \bar{V}^T$, $\bar{U}$ and $\bar{V}$ orthogonal.
- Apply the symmetric QR algorithm to $B^T B$ or BB^T.

Golub and Kahan, 1965.

- Don't need to form $T = B^T B$ explicitly. The QR step (with shift) can be carried out by applying a sequence of Givens rotations to B directly.
- Many different algorithms for computing the SVD exists. Matlab has `svd` for dense matrices and `svds` for sparse matrices.

The Fundamental Subspaces

Lemma If $\sigma_k > 0$ and $\sigma_{k+1} = 0$ then $\mathrm{Rank}(A) = k$.

Remark This means that

$$A = \sum_{i=1}^{k} \sigma_i u_i v_i^T.$$

Lemma If $\mathrm{rank}(A) = k$ then $\mathrm{Range}(A) = \mathrm{span}\{u_1, \ldots, u_k\}$.

Question How to write a basis for $\mathrm{null}(A)$?

Lemma If $\operatorname{rank}(A) = k$ then $\operatorname{null}(A) = \operatorname{span}\{v_{k+1}, \ldots, v_n\}$.

Example Let $Ax = b$. It is often useful to split x and b into components, e.g.

$$x = x_1 + x_2, \quad \text{where} \quad x_1 \in \operatorname{null}(A)^{\perp} \ \text{and} \ x_2 \in \operatorname{null}(A).$$

Remark It holds that $A^T = V\Sigma U^T$ so $\operatorname{range}(A)^{\perp} = \operatorname{null}(A^T)$.

Linear Systems of Equations

Lemma If $A \in \mathbb{R}^{m\times n}$ then $Ax = b$ has a solution if $b \in \operatorname{range}(A)$. The solution is unique if $\operatorname{rank}(A) = n$.

Remark If $\operatorname{rank}(A) = k$ and $b \in \operatorname{range}(A)$ then the general solution of $Ax = b$ is

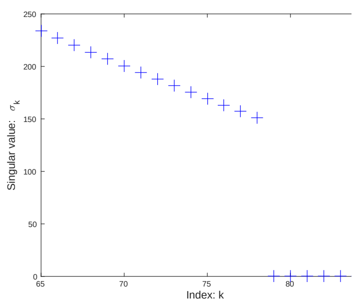
$$x = \sum_{i=1}^{k} \frac{u_i^T b}{\sigma_i} v_i + \sum_{i=k+1}^{n} c_i v_i.$$

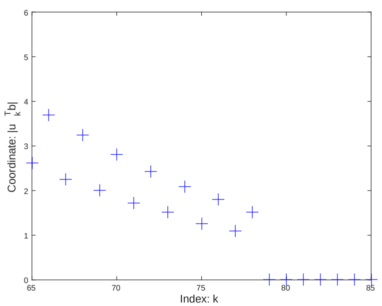
where $c_{k+1}, \ldots, c_n$ are undetermined parameters.

Question How to verify $b \in \operatorname{range}(A)$?

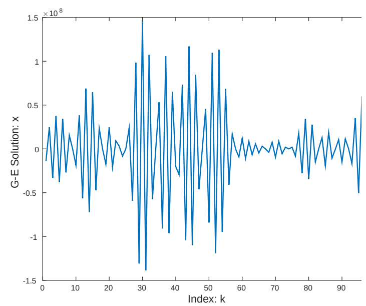
Example In an application we have an 500×100 matrix A and want to solve a linear system $Ax = b$. Since b is obtained by measurements and we know the model is valid $b \in \operatorname{range}(A)$.

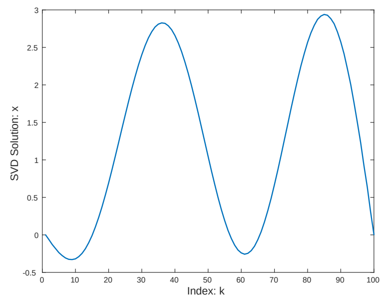
In Matlab Compute the SVD and plot the singular values and also the coefficients $|u_i^T b|$.





Remark We see that $\sigma_{78} = 300.3492$ and $\sigma_{79} = 2.3 \cdot 10^{-10}$ so the rank is $k = \operatorname{rank}(A) = 78$.





Results Solutions using `x=A\b` and `x=Vk*inv(Sk)*Uk'*b`.

After eliminating the small singular values the solution is very good.

The Pseudo Inverse and Least squares problems

Recall Let $A \in \mathbb{R}^{m\times n}$. Previously we defined $A^+ = (A^TA)^{-1}A^T$ and noted that $x = A^+b$ is the vector that minimize $\|Ax - b\|_2$.

Definition If $A \in \mathbb{R}^{m\times n}$ and $\mathrm{rank}(A) = k$ then
$$A^+ = \sum_{i=1}^{k} \frac{v_i u_i^T}{\sigma_i}.$$

Remark If $\mathrm{rank}(A) = n$ then $(A^TA)^{-1}$ exists and the new definition of A^+ coincides with the previous one.

Projections and the SVD

Lemma Suppose $V \in \mathbb{R}^{n\times k}$ has orthonormal columns. Then
$$P = VV^T,$$
is an *orthogonal projection* onto $\mathrm{range}(V)$.

Example Suppose $A = U\Sigma V^T$ and $\mathrm{rank}(A) = k$. Partition
$$U = (U_k, U_{m-k}) \quad \text{and} \quad V = (V_k, V_{n-k}).$$
where, e.g, $U_k = (u_1, \dots, u_k)$.

Question What is the orthogonal projection onto $(\mathrm{null}(A))^\perp$?

Example Suppose that the decomposition $A = U\Sigma V^T$ is available and we want to compute the distance from b to the subspace $\mathrm{range}(A)$, i.e. find the minimum of $\|Ax - b\|_2$.

How should we organize the computations?

Application: Low rank approximation

Theorem If $A \in \mathbb{R}^{m\times n}$ then
$$\min_{\mathrm{rank}(B)=k} \|A - B\|_2 = \sigma_{k+1}, \quad B = \sum_{i=1}^{k} \sigma_i u_i v_i^T.$$

Remark If the number σ_n is small then A is close to rank deficient.

Definition Let $\varepsilon > 0$. The *numerical rank* of A is

$$\text{rank}(A, \epsilon) = \max_k \{\sigma_k > \varepsilon\}.$$

Remark Let μ be the machine precision. If A has full rank but $\text{rank}(A, \mu) < n$ its likely better to treat A as rank deficient.

The Classification Problem

Suppose we study *objects* of a certain type and that objects occur in different variants, or *classes*. Given a new object we want to determine which class it belongs to.

- We collect a large *Reference set* $\{R_k\}$. That is objects of known class.
- Let S be unknown and R_k belong to the reference set. The *distance function* $d(S, R_k)$ measures the similarity between the two objects.

Example Incomming email can either be a spam mail or not.

Nearest Neighbour Classification

Algorithm Let $\{R_k\}$ be the reference set and $d(\cdot, \cdot)$ be the distance function. Do

1. Find k such that $d(S, R_k) = \min_j d(S, R_j)$.
2. The object S is of the same class as R_k.

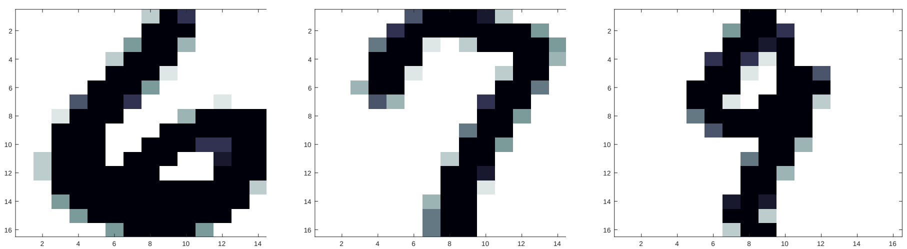
Remark This method is simple, but very accurate assuming the reference set is large enough. It is also too inefficient for practical use.

A good distance function is needed.

Classification of Handwritten Digits

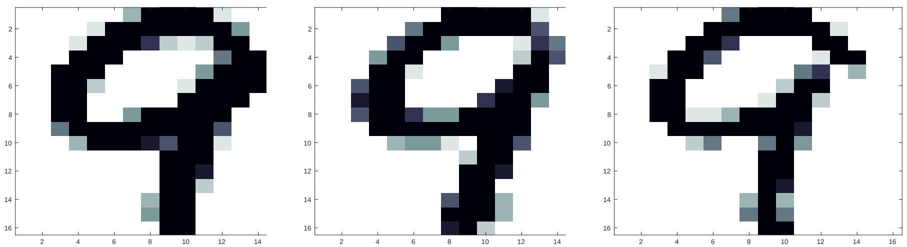
Example A *reference set* consists of $n = 1707$ digits taken from letters (postal codes). The images are stored as 16×16 pixels.

In Matlab `DisplayDigit( RefSet(:,1) );`



Measure distance using Euclidean norm $\|S - R_j\|_2$.

Example The digit S_1 and its two nearest neighbours R_{11} and R_{303}.



This is a successful classification. Of the 20 nearest there are 18 nines and 2 sevens.

Of a (very difficult) *Test Set* of size 2007 a total of 92.8% are classified correctly. Objects are vectors in $\mathbb{R}^{256}$ so have vector space structure.

Classification using Low-Rank approximation

Observation The reference set contains many examples of digits that are very similar.

Let $R^{(k)}$ be a matrix of size $256 \times n_k$ consisting of all reference digits of type k, $k = 0, 1, \ldots, 9$.

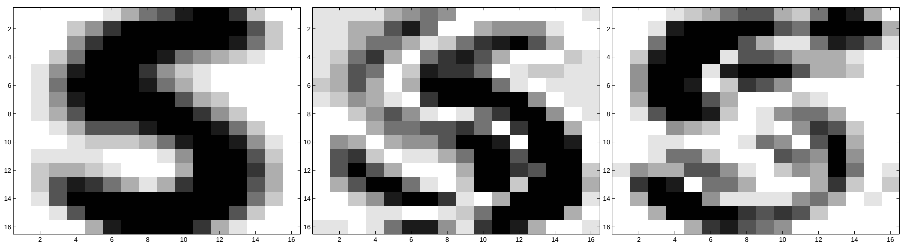
Approximation Compute $R^{(k)} = U^{(k)} \Sigma V^T$ and use

$$\mathrm{span}(R_1^{(k)}, \ldots, R_{n_k}^{(k)}) \approx \mathrm{span}(u_1^{(k)}, \ldots, u_m^{(k)})$$

where m is the dimension of the subspace.

Remark A low dimension m is sufficient to accurately describe the most common variations in writing style.

Example The first 3 basis vectors $u_k^{(5)}$. Created from a total of 88 5:s from the reference set.



Just 5-10 basis vectors very accurately describe the digit 5 and its variations.

For each type of digit we find a low rank approximating subspace $U_m^{(k)} = \{u_1^{(k)}, \ldots, u_m^{(k)}\}$, $k = 0, 1, \ldots, 9$.

Algorithm Classify an unknown object S by

1. Find k such that $d(S, U_m^{(k)}) = \min_j d(S, U_m^{(j)})$.
2. The object S is of class k.

The distance $d(S, U^{(k)})$ is the distance from S to the subspace. This is a least squares problem. The matrices U_m^k has orthogonal columns.

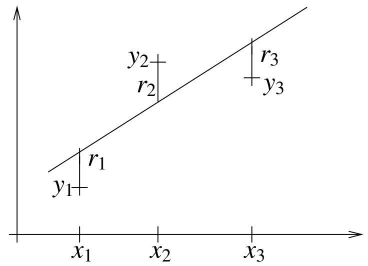
Using subspaces of dimension $m = 10$ we classify 93.2% of the test set correctly. Bad reference digits are removed.

Total least squares

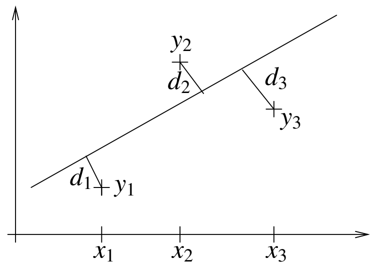
Example Suppose we have a set of points $\{x_i, y_i\}$ and want to find the best possible straight line $y = ax + b$ to this set of data.

Observation A least squares model $y_i = c_0 + c_1 x_i$ would minimize the the distances $|y_i - y|$. Treats y_i and x_i differently.

Can we find a method that treats x_i and y_i the same way? How should we proceed?



Regular: $\min \sum r_i^2$



Total: $\min \sum d_i^2$

In the second case the *orthogonal distance* from the points (x_i, y_i) to the line $y = c_0 + c_1 x$ is minimized.

Definition The *Total least squares* solution x satisfies $(A+E)x = b + r$, where $[E, r]$ is given by

$$\min \|[E, r]\|_2 \text{ such that } (A+E)x = b + r.$$

Remarks The solution always exists since $E = -A$ and $r = -b$ gives a trivial solution. It might not be unique.

Natural to assume errors in both A and b.

Have an over determined linear system $Ax = b$. How to compute the total least squares solution?

Algorithm Compute x_{TLS} by

1. Compute $[A, b] = U\Sigma V^T$. Set $v_{n+1} = V(:, n+1)$.

2. if $v_{n+1}(n+1) \neq 0$ **then**

$$x_{TLS} = -v_{n+1}(1\!:\!n)/v_{n+1}(n+1).$$

end

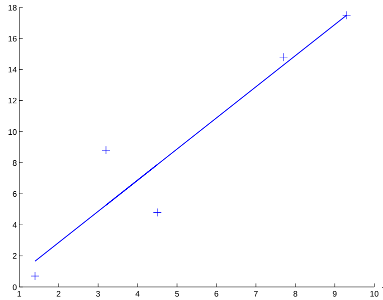
Remark This is sometimes called *orthogonal distance regression*.

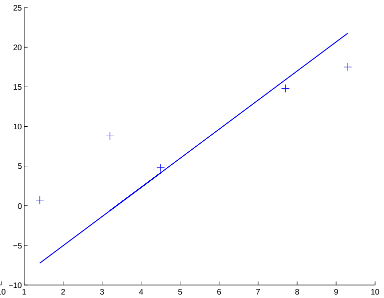
What happens if $v_{n+1}(n+1) = 0$? Not well understood.

Example Fit a straight line to $n = 6$ data points. (x_i, y_i).

In Matlab

```
>> A=[x.^0 , x.^1]; [U,S,V]=svd( [A,y] );
>> x_LS=A\y;
>> x_TLS=-V(1:2,3)/V(3,3);
```





Regular least squares (left) and Total least squares (right).