Data Mining and Matrices

03 – Singular Value Decomposition

Rainer Gemulla, Pauli Miettinen

April 25, 2013

The SVD is the Swiss Army knife of matrix decompositions

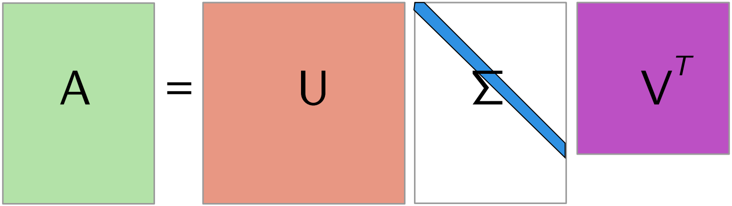
—Diane O'Leary, 2006

Outline

The definition

Theorem. For every $\mathbf{A} \in \mathbb{R}^{m \times n}$ there exists $m \times m$ orthogonal matrix $\mathbf{U}$ and $n \times n$ orthogonal matrix $\mathbf{V}$ such that $\mathbf{U}^T\mathbf{A}\mathbf{V}$ is an $m \times n$ diagonal matrix $\mathbf{\Sigma}$ that has values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min\{n,m\}} \geq 0$ in its diagonal.

- I.e. every $\mathbf{A}$ has decomposition $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$
  - The **singular value decomposition** (SVD)
- The values σ_i are the **singular values** of $\mathbf{A}$
- Columns of $\mathbf{U}$ are the **left singular vectors** and columns of $\mathbf{V}$ the **right singular vectors** of $\mathbf{A}$



Outline

The fundamental theorem of linear algebra

The **fundamental theorem of linear algebra** states that every matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ induces four fundamental subspaces:

- The **range** of dimension $\text{rank}(\mathbf{A}) = r$
  - The set of all possible linear combinations of columns of $\mathbf{A}$
- The **kernel** of dimension $n - r$
  - The set of all vectors $\mathbf{x} \in \mathbb{R}^n$ for which $\mathbf{Ax} = \mathbf{0}$
- The **coimage** of dimension r
- The **cokernel** of dimension $m - r$

The bases for these subspaces can be obtained from the SVD:

- Range: the first r columns of $\mathbf{U}$
- Kernel: the last $(n - r)$ columns of $\mathbf{V}$
- Coimage: the first r columns of $\mathbf{V}$
- Cokernel: the last $(m - r)$ columns of $\mathbf{U}$

Pseudo-inverses

Problem.

Given $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$, find $\mathbf{x} \in \mathbb{R}^n$ minimizing $\|\mathbf{Ax} - \mathbf{b}\|_2$.

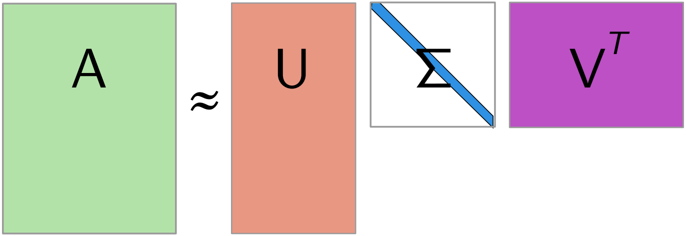
- If $\mathbf{A}$ is invertible, the solution is $\mathbf{A}^{-1}\mathbf{Ax} = \mathbf{A}^{-1}\mathbf{b} \Leftrightarrow \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$
- A **pseudo-inverse** $\mathbf{A}^{+}$ captures some properties of the inverse $\mathbf{A}^{-1}$
- The **Moose–Penrose pseudo-inverse** of $\mathbf{A}$ is a matrix $\mathbf{A}^{+}$ satisfying the following criteria
  - $\mathbf{AA}^{+}\mathbf{A} = \mathbf{A}$ (but it is possible that $\mathbf{AA}^{+} \neq \mathbf{I}$)
  - $\mathbf{A}^{+}\mathbf{AA}^{+} = \mathbf{A}^{+}$ (cf. above)
  - $(\mathbf{AA}^{+})^T = \mathbf{AA}^T$ ($\mathbf{AA}^{+}$ is symmetric)
  - $(\mathbf{A}^{+}\mathbf{A})^T = \mathbf{A}^{+}\mathbf{A}$ (as is $\mathbf{A}^{+}\mathbf{A}$)
- If $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the SVD of $\mathbf{A}$, then $\boxed{\mathbf{A}^{+} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T}$
  - $\mathbf{\Sigma}^{-1}$ replaces σ_i's with $1/\sigma_i$ and transposes the result

Theorem.

The optimum solution for the above problem can be obtained using $\mathbf{x} = \mathbf{A}^{+}\mathbf{b}$.

Truncated (thin) SVD

- The rank of the matrix is the number of its non-zero singular values
  - Easy to see by writing $\mathbf{A} = \sum_{i=1}^{\min\{n,m\}} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$
- The truncated (or thin) SVD only takes the first k columns of $\mathbf{U}$ and $\mathbf{V}$ and the main $k \times k$ submatrix of $\boldsymbol{\Sigma}$
  - $\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \mathbf{U}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^T$
  - $\text{rank}(\mathbf{A}_k) = k$ (if $\sigma_k > 0$)
  - $\mathbf{U}_k$ and $\mathbf{V}_k$ are no more orthogonal, but they are **column-orthogonal**
- The truncated SVD gives a low-rank approximation of $\mathbf{A}$



SVD and matrix norms

Let $\mathbf{A} = \mathbf{U\Sigma V}^T$ be the SVD of $\mathbf{A}$. Then

- $\|\mathbf{A}\|_F^2 = \sum_{i=1}^{\min\{n,m\}} \sigma_i^2$
- $\|\mathbf{A}\|_2 = \sigma_1$
  - Remember: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{n,m\}} \geq 0$
- Therefore $\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_F \leq \sqrt{n}\|\mathbf{A}\|_2$
- The Frobenius of the truncated SVD is $\|\mathbf{A}_k\|_F^2 = \sum_{i=1}^{k} \sigma_i^2$
  - And the Frobenius of the difference is $\|\mathbf{A} - \mathbf{A}_k\|_F^2 = \sum_{i=k+1}^{\min\{n,m\}} \sigma_i^2$

The Eckart–Young theorem

Let $\mathbf{A}_k$ be the rank-k truncated SVD of $\mathbf{A}$. Then $\mathbf{A}_k$ is the closest rank-k matrix of $\mathbf{A}$ in the Frobenius sense. That is

$$\|\mathbf{A} - \mathbf{A}_k\|_F \leq \|\mathbf{A} - \mathbf{B}\|_F \quad \text{for all rank-}k\text{ matrices } \mathbf{B}.$$

Eigendecompositions

- An **eigenvector** of a square matrix $\mathbf{A}$ is a vector $\mathbf{v}$ such that $\mathbf{A}$ only changes the magnitude of $\mathbf{v}$
  - I.e. $\mathbf{Av} = \lambda\mathbf{v}$ for some $\lambda \in \mathbb{R}$
  - Such λ is an **eigenvalue** of $\mathbf{A}$
- The **eigendecomposition** of $\mathbf{A}$ is $\mathbf{A} = \mathbf{Q}\mathbf{\Delta}\mathbf{Q}^{-1}$
  - The columns of $\mathbf{Q}$ are the eigenvectors of $\mathbf{A}$
  - Matrix $\mathbf{\Delta}$ is a diagonal matrix with the eigenvalues
- Not every (square) matrix has eigendecomposition
  - If $\mathbf{A}$ is of form $\mathbf{BB}^T$, it always has eigendecomposition
- The SVD of $\mathbf{A}$ is closely related to the eigendecompositions of $\mathbf{AA}^T$ and $\mathbf{A}^T\mathbf{A}$
  - The left singular vectors are the eigenvectors of $\mathbf{AA}^T$
  - The right singular vectors are the eigenvectors of $\mathbf{A}^T\mathbf{A}$
  - The singular values are the square roots of the eigenvalues of both $\mathbf{AA}^T$ and $\mathbf{A}^T\mathbf{A}$

Outline

Factor interpretation

- The most common way to interpret SVD is to consider the columns of **U** (or **V**)
  - Let **A** be objects-by-attributes and $\mathbf{U\Sigma V}^T$ its SVD
  - If two columns have similar values in a row of $\mathbf{V}^T$, these attributes are somehow similar (have strong correlation)
  - If two rows have similar values in a column of **U**, these users are somehow similar

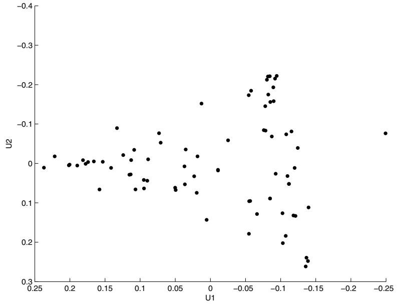
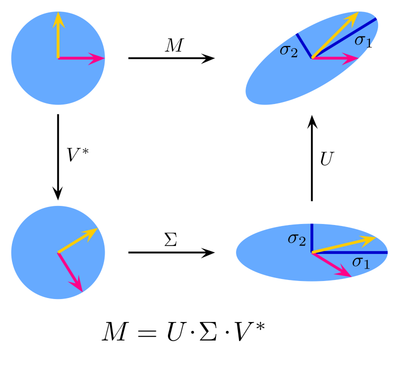


Figure 3.2. *The first two factors for a dataset ranking wines.*

- Example: people's ratings of different wines
- Scatterplot of first and second column of **U**
  - left: likes wine
  - right: doesn't like
  - up: likes red wine
  - bottom: likes white vine
- Conclusion: winelovers like red and white, others care more

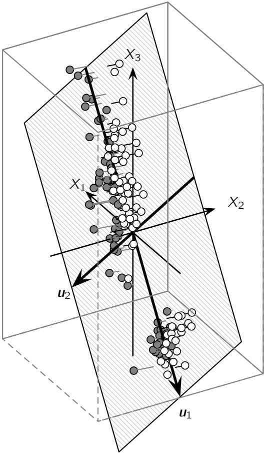
Geometric interpretation



- Let $\mathbf{U\Sigma V}^T$ be the SVD of $\mathbf{M}$
- SVD shows that every linear mapping $\mathbf{y} = \mathbf{Mx}$ can be considered as a series of rotation, stretching, and rotation operations
  - Matrix $\mathbf{V}^T$ performs the first rotation $\mathbf{y}_1 = \mathbf{V}^T\mathbf{x}$
  - Matrix $\mathbf{\Sigma}$ performs the stretching $\mathbf{y}_2 = \mathbf{\Sigma}\mathbf{y}_1$
  - Matrix $\mathbf{U}$ performs the second rotation $\mathbf{y} = \mathbf{U}\mathbf{y}_2$

Wikipedia user Georg-Johann

Dimension of largest variance



- The singular vectors give the dimensions of the variance in the data
  - The first singular vector is the dimension of the largest variance
  - The second singular vector is the orthogonal dimension of the second largest variance
    - First two dimensions span a hyperplane
- From Eckart–Young we know that if we project the data to the spanned hyperplanes, the distance of the projection is minimized

Component interpretation

- Recall that we can write $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \sum_{i=1}^{r} \mathbf{A}_i$
  - $\mathbf{A}_i = \sigma_i \mathbf{v}_i \mathbf{u}_i^T$
- This explains the data as a sums of (rank-1) layers
  - The first layer explains the most
  - The second corrects that by adding and removing smaller values
  - The third corrects that by adding and removing even smaller values
  - ...
- The layers don't have to be very intuitive

Outline

Outline

Problem

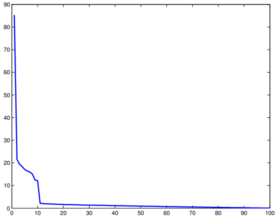
- Most data mining applications do not use full SVD, but truncated SVD
  - To concentrate on “the most important parts”
- But how to select the rank k of the truncated SVD?
  - What is important, what is unimportant?
  - What is structure, what is noise?
  - Too small rank: all subtlety is lost
  - Too big rank: all smoothing is lost
- Typical methods rely on singular values in a way or another

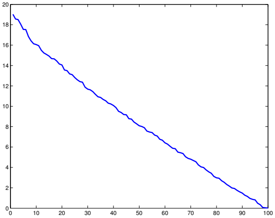
Guttman–Kaiser criterion and captured energy

- Perhaps the oldest method is the Guttman–Kaiser criterion:
  - Select k so that for all $i > k$, $\sigma_i < 1$
  - Motivation: all components with singular value less than unit are uninteresting
- Another common method is to select enough singular values such that the sum of their squares is 90% of the total sum of the squared singular values
  - The exact percentage can be different (80%, 95%)
  - Motivation: The resulting matrix "explains" 90% of the Frobenius norm of the matrix (a.k.a. energy)
- **Problem:** Both of these methods are based on arbitrary thresholds and do not consider the "shape" of the data

Cattell's Scree test

- The **scree plot** plots the singular values in decreasing order
  - The plot looks like a side of the hill, thence the name
- The scree test is a subjective decision on the rank based on the shape of the scree plot
- The rank should be set to a point where
  - there is a clear drop in the magnitudes of the singular values; or
  - the singular values start to even out
- **Problem:** Scree test is subjective, and many data don't have any clear shapes to use (or have many)
  - Automated methods have been developed to detect the shapes from the scree plot





Entropy-based method

- Consider the relative contribution of each singular value to the overall Frobenius norm
  - Relative contribution of σ_k is $f_k = \sigma_k^2 / \sum_i \sigma_i^2$
- We can consider these as probabilities and define the (normalized) **entropy** of the singular values as

$$E = -\frac{1}{\log(\min\{n, m\})} \sum_{i=1}^{\min\{n,m\}} f_i \log f_i$$

  - The basis of the logarithm doesn't matter
  - We assume that $0 \cdot \infty = 0$
  - Low entropy (close to 0): the first singular value has almost all mass
  - High entropy (close to 1): the singular values are almost equal
- The rank is selected to be the smallest k such that $\sum_{i=1}^{k} f_i \geq E$
- **Problem:** Why entropy?

Random flip of signs

- Multiply every element of the data $\mathbf{A}$ randomly with either 1 or -1 to get $\tilde{\mathbf{A}}$
  - The Frobenius norm doesn't change ($\|\mathbf{A}\|_F = \|\tilde{\mathbf{A}}\|_F$)
  - The spectral norm does change ($\|\mathbf{A}\|_2 \neq \|\tilde{\mathbf{A}}\|_2$)
    - ★ How much this changes depends on how much "structure" $\mathbf{A}$ has
- We try to select k such that the residual matrix contains only noise
  - The residual matrix contains the last $m-k$ columns of $\mathbf{U}$, $\min\{n,m\}-k$ singular values, and last $n-k$ rows of $\mathbf{V}^T$
  - If $\mathbf{A}_{-k}$ is the residual matrix of $\mathbf{A}$ after rank-k truncated SVD and $\tilde{\mathbf{A}}_{-k}$ is that for the matrix with randomly flipped signs, we select rank k to be such that $(\|\mathbf{A}_{-k}\|_2 - \|\tilde{\mathbf{A}}_{-k}\|_2)/\|\mathbf{A}_{-k}\|_F$ is small
- **Problem:** How small is small?

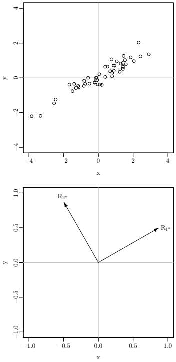
Outline

Normalization

- Data should usually be normalized before SVD is applied
  - If one attribute is height in meters and other weights in grams, weight seems to carry much more importance in data about humans
  - If data is all positive, the first singular vector just explains where in the positive quadrant the data is
- The ***z*-scores** are attributes whose values are transformed by
  - centering them to 0
    - Remove the mean of the attribute's values from each value
  - normalizing the magnitudes
    - Divide every value with the standard deviation of the attribute
- Notice that the z-scores assume that
  - all attributes are equally important
  - attribute values are approximately normally distributed
- Values that have larger magnitude than importance can also be normalized by first taking logarithms (from positive values) or cubic roots
- The effects of normalization should always be considered

Removing noise

- Very common application of SVD is to remove the noise from the data
- This works simply by taking the truncated SVD from the (normalized) data
  - The big problem is to select the rank of the truncated SVD
- Example:



$\sigma_1 = 11.73$
$\sigma_2 = 1.71$

- Original data
  - Looks like 1-dimensional with some noise
- The right singular vectors show the directions
  - The first looks like the data direction
  - The second looks like the noise direction
- The singular values confirm this

Removing dimensions

- Truncated SVD can also be used to battle the **curse of dimensionality**
  - All points are close to each other in very high dimensional spaces
  - High dimensionality slows down the algorithms
- Typical approach is to work in a space spanned by the columns of $\mathbf{V}^T$
  - If $\mathbf{U\Sigma V}^T$ is the SVD of $\mathbf{A} \in \mathbb{R}^{m \times n}$, project $\mathbf{A}$ to $\mathbf{AV}_k \in \mathbb{R}^{m \times k}$ where $\mathbf{V}_k$ has the first k columns of $\mathbf{V}$
  - This is known as the **Karhunen–Loève transform** (KLT) of the rows of $\mathbf{A}$
    - ⋆ Matrix $\mathbf{A}$ must be normalized to z-scores in KLT

Visualization

- Truncated SVD with $k = 2, 3$ allows us to visualize the data
  - We can plot the projected data points after 2D or 3D Karhunen–Loève transform
  - Or we can plot the scatter plot of two or three (first, left/right) singular vectors

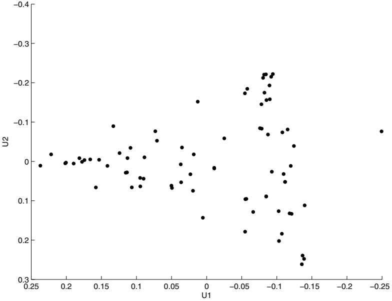
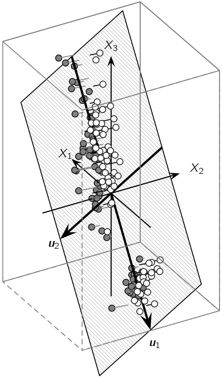


Figure 3.2. *The first two factors for a dataset ranking wines.*



Latent semantic analysis

- The **latent semantic analysis** (LSA) is an information retrieval method that uses SVD
- The data: a term–document matrix $\mathbf{A}$
  - the values are (weighted) term frequencies
  - typically tf/idf values (the frequency of the term in the document divided by the global frequency of the term)
- The truncated SVD $\mathbf{A}_k = \mathbf{U}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^T$ of $\mathbf{A}$ is computed
  - Matrix $\mathbf{U}_k$ associates documents to topics
  - Matrix $\mathbf{V}_k$ associates topics to terms
  - If two rows of $\mathbf{U}_k$ are similar, the corresponding documents "talk about same things"
- A query q can be answered by considering its term vector $\mathbf{q}$
  - $\mathbf{q}$ is projected to $\mathbf{q}_k = \mathbf{q}\mathbf{V}\boldsymbol{\Sigma}^{-1}$
  - $\mathbf{q}_k$ is compared to rows of $\mathbf{U}$ and most similar rows are returned

Outline

Algorithms for SVD

- In principle, the SVD of $\mathbf{A}$ can be computed by computing the eigendecomposition of $\mathbf{A}\mathbf{A}^T$
  - This gives us left singular vectors and squares of singular values
  - Right singular vectors can be solved: $\mathbf{V}^T = \boldsymbol{\Sigma}^{-1}\mathbf{U}^T\mathbf{A}$
  - **Bad for numerical stability!**
- Full SVD can be computed in time $O\big(nm\min\{n, m\}\big)$
  - Matrix $\mathbf{A}$ is first reduced to a bidiagonal matrix
  - The SVD of the bidiagonal matrix is computed using iterative methods (similar to eigendecompositions)
- Methods that are faster in practice exist
  - Especially for truncated SVD
- Efficient implementation of an SVD algorithm requires considerable work and knowledge
  - Luckily (almost) all numerical computation packages and programs implement SVD

Outline

Lessons learned

- SVD is the Swiss Army knife of (numerical) linear algebra
  $\rightarrow$ ranks, kernels, norms, ...
- SVD is also very useful in data analysis
  $\rightarrow$ noise removal, visualization, dimensionality reduction, ...
- Selecting the correct rank for truncated SVD is still a problem

Suggested reading

- Skillicorn, Ch. 3
- Gene H. Golub & Charles F. Van Loan: *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996
  - Excellent source for the algorithms and theory, but very dense

Outline

Basic information

- Assignment sheet will be made available later today/early tomorrow
  - We'll announce it in the mailing list
- DL in two weeks, delivery by e-mail
  - Details in the assignment sheet
- Hands-on assignment: data analysis using SVD
- Recommended software: R
  - Good alternatives: Matlab (commercial), GNU Octave (open source), and Python with NumPy, SciPy, and matplotlib (open source)
  - Excel is **not** a good alternative (too complicated)
- What you have to return?
  - Single document that answers to all questions (all figures, all analysis of the results, the main commands you used for the analysis if asked)
  - Supplementary material containing the transcript of all commands you issued/all source code
  - **Both files in PDF format**