Lecture 7: Matrix decomposition and LSA

William Webber (`william@williamwebber.com`)

COMP90042, 2014, Semester 1, Lecture 7

What we'll learn in this lecture

- A tutorial on matrix algebra
- A simple matrix transformation (Principal Component Analysis) which aligns data with most "important" correlated dimensions
- A related matrix decomposition called Singular Value Decomposition (SVD)
- How to interpret SVD when performed on a TDM
- An initial look at Latent Semantic Analysis (LSA) which uses reduced-rank SVD to find "concepts" in a document corpus

Matrix concepts

- $\mathbf{X}$ is a matrix with m rows $\{r_1, r_2, \ldots, r_m\}$ and n columns $\{c_1, c_2, \ldots, c_n\}$ ($\mathbf{X}_{m \times n}$ for short)
  - Applied to TDM, rows are terms, columns are docs (NB)
- x_{ij} is the element in row i, column j of $\mathbf{X}$
  - In TDM, this is a (possibly 0) term posting
- $(\mathbf{X}_{n \times m})^T$ is the transpose of $\mathbf{X}mn$, where $x_{ij}^T = x_{ji}$
  - In TDM, transposing is analogous to view points as terms in document space, rather than documents in term space
- A *square* matrix has the same number of rows as columns ($m = n$)
- A *diagonal* matrix is one which has non-zero values only on the diagonal (i.e., $x_{ij} = 0$ if $i \neq j$).

Matrix multiplication and geometry

- If $\mathbf{X}$ is $m \times n$ and $\mathbf{Y}$ is $n \times p$, then $\mathbf{Z} = \mathbf{XY}$ is $m \times p$ (matrix multiplication)
- Matrix multiplication is associative:

$$\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C} \tag{1}$$

- If $\mathbf{X}$ is $m \times d$, and $\mathbf{Y}$ is square $d \times d$, then:
  - $\mathbf{X}$ can be interpreted as locating m items in d-dimensional space
  - $\mathbf{Y}$ can be interpreted as some (combined) geometrical transformation (rotate, shear, scale, translate)
- In particular, if $\mathbf{Y}$ is a diagonal vector, it can be interpreted as a scale (dimensions scaled, but remain independent)

More matrix concepts

- The identity matrix $\mathbf{I}$ is a square matrix with 1 in the diagonals, 0 elsewhere
  - $\mathbf{M}_{\cdot \times n} \cdot \mathbf{I}_{n \times n} = \mathbf{M}$
- $\mathbf{M}^{-1}$ is the inverse of $\mathbf{M}$ if $\mathbf{M}\mathbf{M}^{-1} = \mathbf{I}$
- For a diagonal matrix $\mathbf{d}$, $\mathbf{d}^{-1}$ has the diagonal values $d_{i,i}^{-1} = 1/d_{i,i}$ (and 0 elsewhere)

Rank and orthogonality

- A set of vectors $\mathcal{V} = \{v_1, v_2, \cdot, v_n\}$ is linearly independent if no vector v_i can be expressed as a weighted combination of the other vectors $v_1, \cdots, v_{i-1}, v_{i+1}, v_n$.
- An $m \times n$ matrix $\mathbf{M}$ has rank r ($r \leq \min(m, n)$) if r is the size of the largest set of linearly independent row (or column) vectors of $\mathbf{M}$
- Two vectors $\mathbf{v}$, $\mathbf{w}$, of same length n, are orthogonal ("at right angles") if $\mathbf{v} \cdot \mathbf{w} = 0$.
- $\mathbf{v}$ and $\mathbf{w}$ are orthonormal if in addition they are unit vectors.
- If we have a set $\mathcal{V}$ of n orthonormal vectors $\{v_1, v_2, \cdots, v_n\}$, and each vector is also of length n, then $\mathcal{V}$ is an *orthonormal basis*.
- Necessarily, $\mathcal{V}$ is also linearly independent
- An *orthogonal* matrix $\mathbf{Q}$ is one in which columns (or rows) form an orthonormal basis. (Necessarily square.)
- If $\mathbf{Q}$ orthogonal, $\mathbf{Q}^T = \mathbf{Q}^{-1}$ (very handy for algebraic manipulations)

Orthonormal basis

- An orthonormal basis can be thought of as a set of axes
- So the standard 3-d Cartesian axes are:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

- An orthogonal matrix $\mathbf{N}$ can be interpreted as a rotation (around the origin) operation
- Specifically, $\mathbf{N}$ is the rotation that transforms points into the orthonormal basis space defined by the columns of $\mathbf{N}$
- So, $\mathbf{MN}$ can be viewed as either:
  - Rotating $\mathbf{M}$ by $\mathbf{N}$; or
  - "Viewing" $\mathbf{M}$ from the basis space ("axes") of $\mathbf{N}$

Eigenvalues and eigenvectors

- Let $\mathbf{A}$ be an $n \times n$ matrix.
- Let there be some vector $\mathbf{x}$ of size $n \times 1$ (that is, n rows and 1 column), such that there exists a scalar (i.e. single real value) λ such that:

$$\mathbf{A}\mathbf{x} = \lambda x \tag{3}$$

- Then we say that:
  - $\mathbf{x}$ is an *eigenvector* of $\mathbf{A}$
  - λ is an *eigenvalue* of $\mathbf{A}$; and more specifically
  - λ is the eigenvalue of $\mathbf{A}$ that corresponds to $\mathbf{x}$.

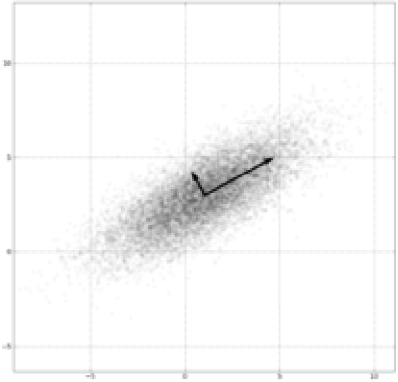
Properties of eigenvalues and eigenvectors

- An nXn matrix has no more than n eigenvalues
- The eigenvectors of the one matrix **A** are linearly independent
- If **A** is symmetric and of rank $r \leq n$, the eigenvectors are orthogonal
- ... and there are exactly r non-zero eigenvalues
- If the eigenvectors are normalized to unit length, they define an orthonormal basis

Principle component analysis (PCA): motivation

- Data may have many variables, but fewer (important) relations (*components*) as some variables (e.g. terms) may be highly correlated
- Would like to shift variables (axes) so that they aligned along important components:
  - $x = x_1$ axis along most important component
  - $y = x_2$ axis along next most important (orthogonal to x)
  - $z = x_3$ axis along next most important (orthogonal to x and y)
  - x_k axis long most important axis orthogonal to $\{x_1, x_2, \cdots, x_{k-1}\}$
- We can then also "drop" the unimportant dimensions

PCA illustrated



- Center origin in mean of each dimension
- Align orthogonal axes along decreasing covariances [1]

[1] Image source: Wikipedia

PCA (with dimensionality reduction)

- Start with $m \times n$ matrix $\mathbf{M}$
- Shift each variable so that it has 0 mean, $\rightarrow \mathbf{X}$
- Calculate $n \times n$ covariance matrix $\mathbf{C} = \frac{1}{n}\mathbf{X}^T\mathbf{X}$
- Calculate the size n, unit-length eigenvectors (orthonormal basis) of $\mathbf{C}$, and corresponding eigenvalues
- Choose d top eigenvalues, and concat to $n \times d$ matrix $\mathbf{P}$
- $\mathbf{P}$ represents a rotation that drops some dimensions
- Now $m \times d$ matrix $\mathbf{N} = \mathbf{XP}$ is the original data, zero-centered, then transformed into the reduced, (concept-)transformed space.

Singular value decomposition (SVD)

$\mathbf{X}$ is an $m \times n$ matrix. It can be decomposed into:

$$\mathbf{X} = \mathbf{U\Sigma V}^T \tag{4}$$

where:

- $\mathbf{U}$ is $m \times m$ and orthogonal
- $\mathbf{\Sigma}$ is $m \times n$ and diagonal (but not square!)
- $\mathbf{V}$ is $n \times n$ and orthogonal

- Orthogonal matrices interpretable as rotation around origin
- Diagonal matrices as scales
- So Equation (4) interpretable as decomposing transform represented by $\mathbf{M}$ into a rotation, then a scale, then another rotation
- Important distinction from PCA: we don't zero-center before rotating!

SVD: singular values

$$\mathbf{\Sigma} = \begin{pmatrix} \sqrt{\lambda_1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sqrt{\lambda_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sqrt{\lambda_r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{5}$$

- r is rank of $\mathbf{X}_{m \times n}$ (at most $\min(m, n)$, but can be less)
- $\lambda_1 > \lambda_2 > \ldots > \lambda_r$ are non-zero eigenvalues of $\mathbf{X}^T\mathbf{X}$,
  - Note: $\frac{1}{n}\mathbf{X}^T\mathbf{X}$ is covaraince matrix (used in PCA)
- $\sigma_i = \sqrt{\lambda_i}$ are *singular values*.
- Redundant dimensions are diagonal 0
- Not necessarily square, though extra column or row all 0

SVD: singular vectors

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \begin{pmatrix} \sigma_{11} & 0 & 0 & 0 \\ 0 & \sigma_{22} & 0 & 0 \\ 0 & 0 & \sigma_{33} & 0 \end{pmatrix} \begin{pmatrix} v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ v_{41} & v_{42} & v_{43} & v_{44} \end{pmatrix}$$

$$\mathbf{X}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\left(\mathbf{V}_{n\times n}\right)^T \tag{6}$$

- $\hat{\mathbf{v}}_i$ is $n \times 1$ unit eigenvector for the eigenvalue λ_i.
- $\mathbf{V} = [\hat{\mathbf{v}}_1, \cdots, \hat{\mathbf{v}}_r, \hat{\mathbf{v}}_{r+1}, \cdots, \hat{\mathbf{v}}_n]$
  - $[\hat{\mathbf{v}}_1, \cdots, \hat{\mathbf{v}}_r]$ is orthogonal
  - $\hat{\mathbf{v}}_j, r < j \leq n$ orthonormal "fillers"
- $\hat{\mathbf{u}}_i$ is the $m \times 1$ vector defined by $\hat{\mathbf{u}}_i = \frac{1}{\sigma_i}\mathbf{X}\hat{\mathbf{v}}_i$
- $\mathbf{U} = [\hat{\mathbf{u}}_1, \cdots, \hat{\mathbf{u}}_r, \hat{\mathbf{u}}_{r+1}, \cdots, \hat{\mathbf{u}}_n]$
  - similarly "filled out" with $m - r$ orthonormal vectors
- $\mathbf{U}$ and $\mathbf{V}$ hold left and right *singular vectors* of $\mathbf{X}$

Interpreting SVD for TDM

$$\mathbf{X}_{t\times d} = \mathbf{T}_{t\times t}\mathbf{\Sigma}_{t\times d}\left(\mathbf{D}_{d\times d}\right)^T \tag{7}$$

- Each SV relates to a "semantic dimension" ("topic")
- $\mathbf{\Sigma}$ gives importance of topic
- $\mathbf{T}$ a change of basis op, shifting terms into semantic space:

$$\mathbf{T}^T\mathbf{X} = \mathbf{\Sigma}\mathbf{D}^T \tag{8}$$

- $\mathbf{\Sigma}\mathbf{D}^T$ are documents in semantic space
- $\mathbf{D}^T$ change of basis op, shifting terms into semantic space:

$$\mathbf{D}^T\mathbf{X}^T = \mathbf{\Sigma}\mathbf{T}^T \tag{9}$$

  (and $\mathbf{\Sigma}\mathbf{D}^T$ are the terms projected)
- $\mathbf{T}$ relates terms to topics; value gives strength. Interpretation of negative values unclear.
- $\mathbf{D}$ relates docs to topics

Dimensionality reduction in SVD

- The $\{\sigma_1, \cdots, \sigma_r\}$ values on the diagonal of $\mathbf{\Sigma}$ are ordered by decreasing "importance" of the corresponding dimension
- We can reduce dimensionality to only top k concepts by setting $\{\sigma_{k+1}, \cdots, \sigma_r\}$ to 0.
- This gives reduced representation:

$$\mathbf{X}_{t\times d} \approx \mathbf{X_K}_{t\times d} = \mathbf{T_K}_{t\times k}\mathbf{\Sigma_K}_{k\times k}\left(\mathbf{D_K}_{d\times k}\right)^T \qquad (10)$$

- $\mathbf{\Sigma_K}\mathbf{D_K}^T$ $(k \times d)$ represents docs (cols) in k-d latent space
- $\mathbf{\Sigma_K}\mathbf{T_K}^T$ $(k \times t)$ represents terms (cols) in k-d latent space
- $\mathbf{T_K}$, $\mathbf{D_K}$ retain term–topic, doc–topic relations for top k topics

Latent Semantic Analysis

$$\mathbf{X} \approx \mathbf{T_K} \mathbf{\Sigma_K} \mathbf{D_K}^T \tag{11}$$

- Rank-lowering SVD on the TDM is used in the cluster of related techniques known as *Latent Semantic Analysis*
- The "big claim" for LSA that this captures the "semantic structure" of the collection
- Matches by "topic", not term
- Automatically expands term into underlying topic
- Allows semantically related documents (queries) to match, even if different terms used
- (Also referred to as "Latent Semantic Indexing", or LSI)

Document comparison

- $\mathbf{Z_K} = \mathbf{\Sigma_K} \mathbf{D_K}^T$ represents docs (cols) in semantic space
- Documents d_i and d_j can be compared using cosine distance on i and j columns of $\mathbf{Z_K}$
- Similar to comparison on TDM, except:
  - Compares by "concepts" (useful for short documents, e.g. sentences)
  - Dense, k-d representation, rather than sparse t-d
    - Suits vector hardware, e.g. GPU
- Clustering can also be done in semantic space
  - Again, faster due to short, dense vectors
  - (though doing the SVD itself can be slow!)

Term comparison

- $\mathbf{Y_K} = \mathbf{\Sigma_K}\mathbf{T_K}^T$ represents terms (cols) in semantic space
- Terms t_i and t_j can be compared as cosine distance on i and j columns of $\mathbf{Y_K}$.
- And clustering can be done (as with docs)
- Also, $\mathbf{Z_K}$ and $\mathbf{Y_K}$ are in same k-d space
- So we can directly compare terms with documents (though what this precisely means...)

Searches in LSA space

- Search for a query q similar in LSA space to TDM space
- treat q as doc, calculate cosine with true docs in $\mathbf{D_K}$
- But q must first be converted into the k-dim form
- $\mathbf{D_K}$ calculable as:

$$\mathbf{D_K} = \mathbf{X}^T \mathbf{T_K} \mathbf{\Sigma_K}^{-1} \tag{12}$$

- Therefore q_K calculated as

$$\mathbf{q_K} = \mathbf{q} \mathbf{T_K} \mathbf{\Sigma_K}^{-1} \tag{13}$$

- Semantic space performs automatic (global) query expansion
- Note: practicalities of query evaluation change, because:
  - Even short queries have many "concepts"
  - Docvecs no longer large and sparse, but short and dense

Folding new documents into space

- Recalculating full SVD when new documents added expensive
  - (though there are now incremental algorithms available)
- But new documents can be “folded in” in same way as queries
- That is, calculate their k-d representation as $\mathbf{d_K} = \mathbf{d T_K \Sigma_K}^{-1}$
- Then add $\mathbf{d_K}$ as new row to $\mathbf{D_K}$
- Folded-in documents, however, did not contribute to semantic decomposition
- As more are added, representativeness of decomposition declines
- Particularly if new documents are significantly different (e.g. represent different concepts) from old ones

Latent-SVD as semantic tool

- Concept of "folding" alerts that not all documents need to be included in SVD
- Provided coverage of co-occurrences is adequate
- For instance, could sample documents
  - though this will miss rarer co-occurences (even though this may be significant)
- One can view LSA-SVD not as index, but as semantic transformation tool

Topic analysis

$$\mathbf{X}_{t \times d} = \mathbf{T_K}_{t \times k} \mathbf{\Sigma_K}_{k \times k} (\mathbf{D_K}_{d \times k})^T \tag{14}$$

- Left singular vectors $\mathbf{T_K}$ map between k terms and “semantic dimensions” (topics)
- Then column k of $\mathbf{T_K}$ “describes” topic by giving strength of association with each term
  - Interpretation of negative weights unclear
  - Many terms have some non-zero association with each topic, though most are not “significant”

Topic analysis example

Tpc	Terms	Labels
0	iraq, percent, bank, rate, trad, shar, ...	??Overall
1	iraq, kurd, saddam, missil, attack, baghdad, ...	Iraq
2	net, profit, loss, shar, incom, tax, dividend, ...	Financials
3	bank, govern, minist, israel, elect ...	?Israel election
4	ton, wheat, oil, chin, trad ...	?Resources
5	shar, stock, point, index, clos ...	Sharemarket

- Took LYRL-30k collection.
- Performed $k = 100$ LSA analysis using `gensim` toolkit (needed 88 seconds on my laptop)
- Top positive terms for top 6 topics given above, with possible labels (that I came up with)
- What do you think of these topics?

Topic analysis by documents

- Right singular vectors $\mathbf{T_K}$ map between topics and documents
- (though these are not so easy to get out of `gensim`)
- Could in principle tell us what a document was "about"
- As with terms, one document can be associated with many topics

LSA: computational complexity

$$\mathbf{X}_{l \times d} = \mathbf{T}_{t \times t} \mathbf{\Sigma}_{t \times d} \left(\mathbf{D}_{d \times d}\right)^T \tag{15}$$

- Time complexity of full SVD is $O(\min\{t^2 d, td^2\})$ (ouch!)[2]
- For reduced dimension k, this can be reduced[3] to $O(tdk)$
- For sparse matrices (and the TDM is sparse) and (approximate) incremental methods, faster still
  - e.g. `gensim` claims[4] $O(duk + tk^2)$, where u is the average number of words (terms?) per document. I.e. $\approx O(z)$, where z is the number of postings in collection (non-zero cells in TDM).

NOTE: Computational complexity of LSA (equiv.: low-rank or thin SVD on sparse matrices) not well documented; would make good final project for someone with strong matrix algebra

---

[2] Holmes, Gray, Isbell, "Fast SVD", 2007.

[3] Brand, "Fast Low-Rank Modifications of the Thin SVD", 2006

[4] `http://bit.ly/1kZuEU0`

LSA in practice

- LSA widely used, particularly in industry and in non-core CS tasks (e.g. automatic marking of student essays)
- Has not been widely adopted in "core" IR:
  - SVD was too compute intensive (still is for large corpora)
  - Pseudo-relevance feedback techniques (e.g. Rocchio) and other "local" query expansion techniques work as well or better
  - Inability to do exact term matching a drawback
- LSA may be useful as component in larger system (e.g. for global expansion, topic analysis), especially if built on sample to reduce computation

Topic modelling

- Through the left and right singular vectors, LSA provides a form of topic modelling (viz. identification of semantic concepts to which terms and documents are co-clustered)
- Has been criticism of the (lack of) theoretical basis on which LSA topics stand
- Also difficulty of interpreting the term–topic association scores
- Recent attention has turned to probabilistic topic models

Looking back and forward

Back: SVD

- PCA shifts and rotates TDM to align dimensions along term covariances
- SVD splits $\mathbf{X}$ into $\mathbf{U\Sigma V}$
- We can reduce from full rank r to k-dimensional space by dropping smaller singular values in $\mathbf{\Sigma}$
- In LSA, the SVD is seen as mapping from “terms” to “concepts”
- Reduction to k dimensions extracts k “key concepts”

Looking back and forward

Back

LSA

- LSA uses reduced-rank SVD to project TDM into “semantic space”
- Reduced dimensions make clustering faster
- Term co-association in topics provides term expansions (particularly for queries or very short documents)
- LSA provides a form of topic modelling

Looking back and forward

Forward

- Probablistic LSA and LDA (week after next) provide a probabilistic approach to extracting concepts from TDM space
- Next week, will look at geometric approaches to text classification

Further reading

- Jonathon Shlens, "A Tutorial on Principal Component Analysis"[5] (2005 (?)). Also discusses singular value decomposition.
- Deerwester, Dumais, Furnas, Landauer, and Harshman, "Indexing by Latent Semantic Analysis", *JASIST*, 1990.
- Berry, Dumais, and O'Brien, "Using Linear Algebra for Intelligent Information Retrieval", *SIAM*, 1995.
- Chapter 18, "Matrix decompositions and latent semantic indexing"[6], of Manning, Raghavan, and Schutze, *Introduction to Information Retrieval*, CUP, 2009.

[5] http://www.cs.princeton.edu/picasso/mats/PCA-Tutorial-Intuition_jp.pdf

[6] http://nlp.stanford.edu/IR-book/pdf/18lsi.pdf

Appendix: Dimensionality reduction example

$$\underset{\mathbf{X}_{4\times 5}}{} = \underset{\mathbf{T}_{4\times 4}}{\begin{pmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ t_{41} & t_{42} & t_{43} & t_{44} \end{pmatrix}} \underset{\mathbf{\Sigma}_{4\times 5}}{\begin{pmatrix} \sigma_1 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}} \underset{(\mathbf{D}_{5\times 5})^T}{\begin{pmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} \\ d_{41} & d_{42} & d_{43} & d_{44} & d_{45} \\ d_{51} & d_{52} & d_{53} & d_{54} & d_{55} \end{pmatrix}}$$

- $t = 4$ terms, $d = 5$ documents
- Here, rank $r = 3$
  - $r < t$, implies term made redundant by others
- d_5, d_4, and t_4 can be dropped, and $\mathbf{\Sigma}$ shrunk to $r \times r = 3 \times 3$

Appendix: Dimensionality reduction example

$$\begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \\ t_{41} & t_{42} & t_{43} \end{pmatrix} \quad \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix} \quad \begin{pmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} \end{pmatrix}$$

$$\mathbf{T}_{\mathbf{R}4\times 3} \qquad \mathbf{\Sigma}_{\mathbf{R}3\times 3} \qquad (\mathbf{D}_{\mathbf{R}5\times 3})^T$$

- $t = 4$ terms, $d = 5$ documents
- Here, rank $r = 3$
  - $r < t$, implies term made redundant by others
- d_5, d_4, and t_4 can be dropped, and $\mathbf{\Sigma}$ shrunk to $r \times r = 3 \times 3$
- Dimensionality can be lower from $r = 3$ to $k = 2$ by setting lowest-weight SV σ_3 to 0

Appendix: Dimensionality reduction example

$$\begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \\ t_{41} & t_{42} & t_{43} \end{pmatrix} \quad \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} \end{pmatrix}$$

$$\mathbf{T}_{\mathbf{R}4\times3} \qquad \mathbf{\Sigma}_{\mathbf{R}3\times3} \qquad (\mathbf{D}_{\mathbf{R}5\times3})^T$$

- $t = 4$ terms, $d = 5$ documents
- Here, rank $r = 3$
  - $r < t$, implies term made redundant by others
- d_5, d_4, and t_4 can be dropped, and $\mathbf{\Sigma}$ shrunk to $r \times r = 3 \times 3$
- Dimensionality can be lower from $r = 3$ to $k = 2$ by setting lowest-weight SV σ_3 to 0
- And then d_3 and t_3 can be dropped, and $\mathbf{\Sigma}$ shrunk to $k \times k = 2 \times 2$

Appendix: Dimensionality reduction example

$$\begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{31} & t_{32} \\ t_{41} & t_{42} \end{pmatrix} \quad \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \quad \begin{pmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} \end{pmatrix}$$

$$\mathbf{T}_{\mathbf{K}4\times 2} \qquad \mathbf{\Sigma}_{\mathbf{K}2\times 2} \qquad (\mathbf{D}_{\mathbf{K}5\times 2})^T$$

- $t = 4$ terms, $d = 5$ documents
- Here, rank $r = 3$
  - $r < t$, implies term made redundant by others
- d_5, d_4, and t_4 can be dropped, and $\mathbf{\Sigma}$ shrunk to $r \times r = 3 \times 3$
- Dimensionality can be lower from $r = 3$ to $k = 2$ by setting lowest-weight SV σ_3 to 0
- And then d_3 and t_3 can be dropped, and $\mathbf{\Sigma}$ shrunk to $k \times k = 2 \times 2$