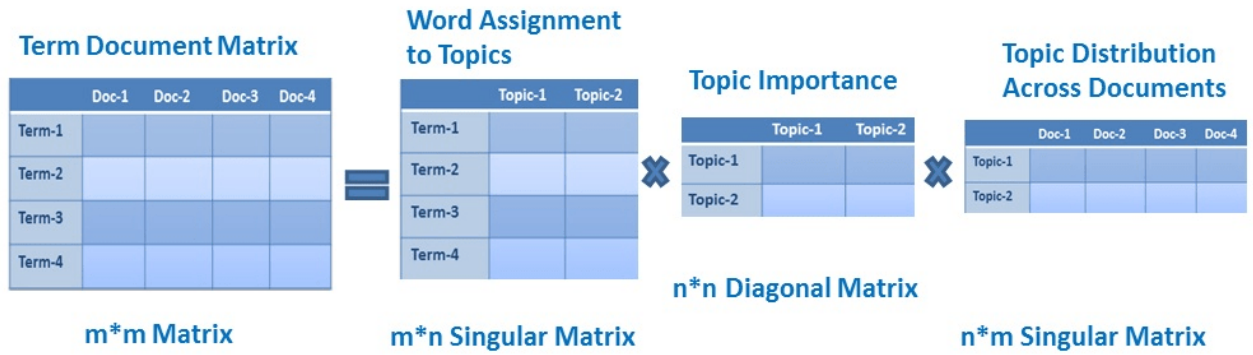
Text Retrieval & Search Engine (CP423B)
Latent Semantic Indexing

Course Instructor: Stanley (Zhaohui) Liang, PhD

Why we need latent semantic indexing

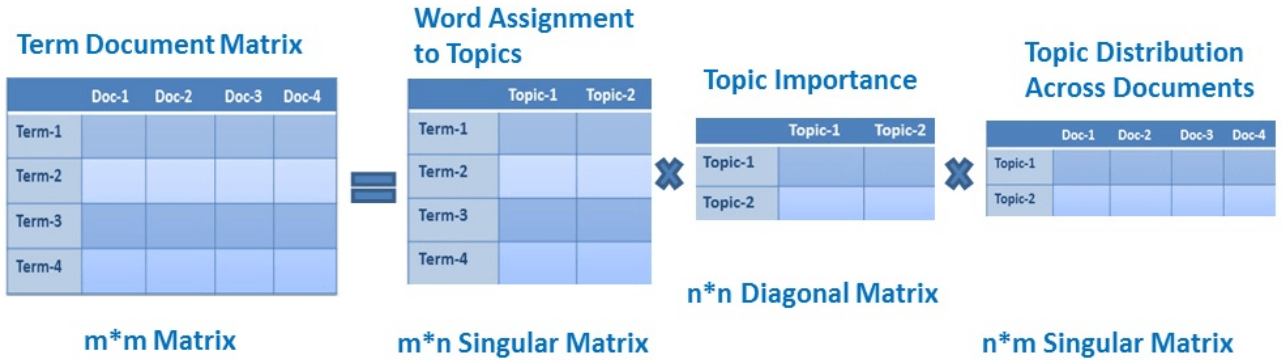
- Term-document matrices are very large
- The # of topics that people search is relatively small
  - E.g., clothes, food, movies, politics, sports
- To find a way to represent the term-document space by a lower dimensional latent space
  - Improve storage efficiency
  - Improve ambiguity search



LATENT SEMANTIC INDEXING (LSI)

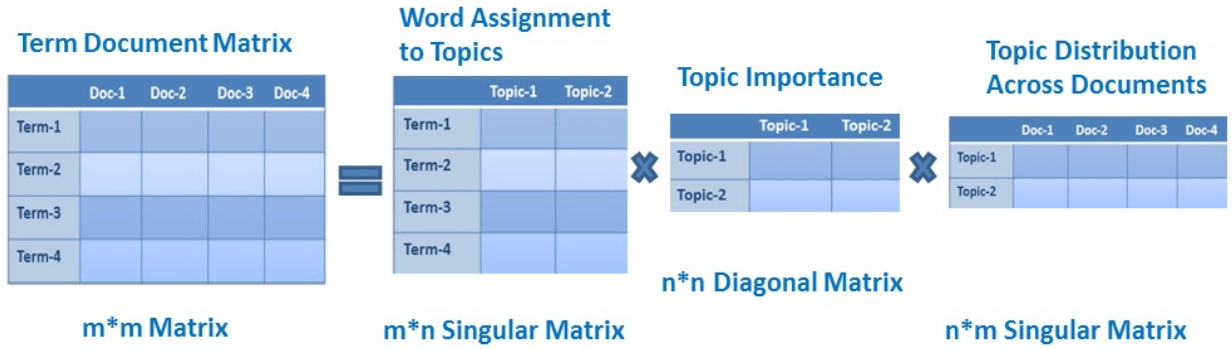
LSI uses a mathematical technique called singular value decomposition (SVD) to identify patterns in the relationships between the terms and concepts in an unstructured document collection

LSI can analyze relationship between a set of documents and the terms in the docs, and extract a set of concepts related to the documents and terms



Latent Semantic Indexing (LSI)

- Main idea
  - Map each document into some "concepts"
  - Map each term into some "concepts"
- Concept: a set of terms with corresponding weights to other terms
  - DBMS as concept:
    - Data – 0.8
    - System – 0.5
    - Retrieval – 0.6



Latent Semantic Indexing (LSI)

term-concept matrix

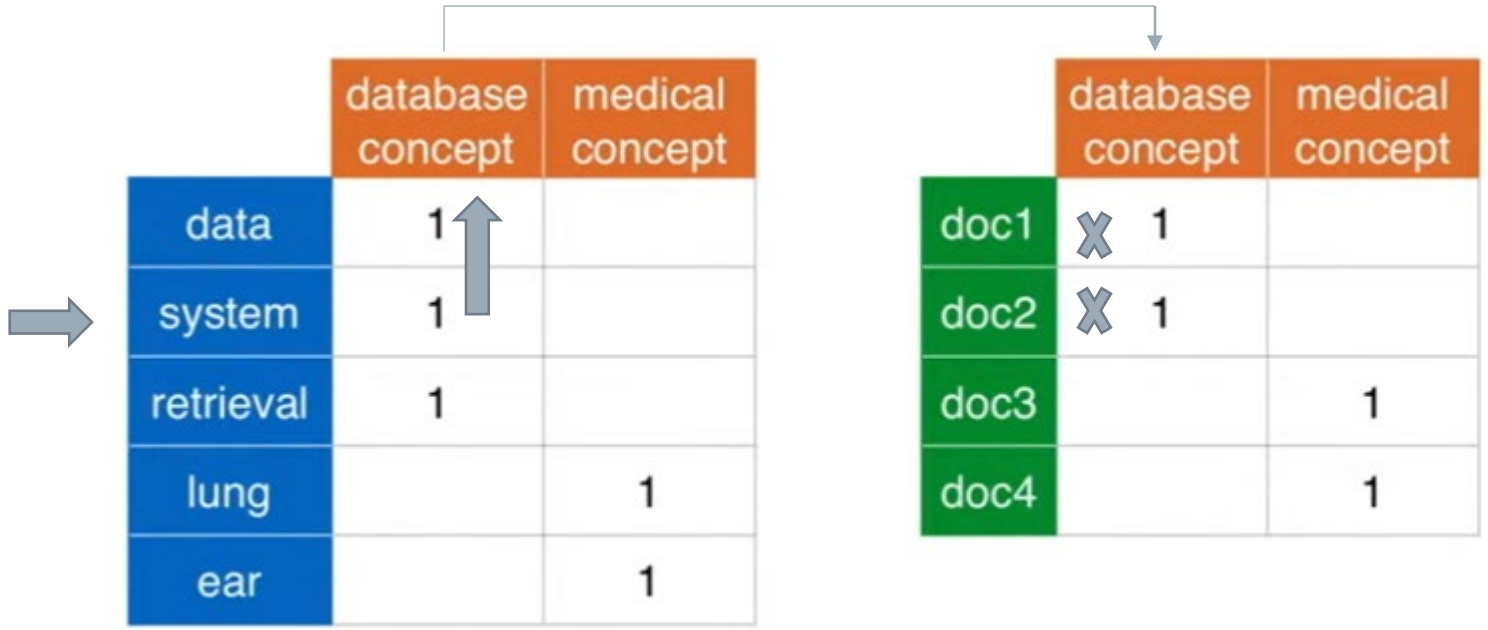
	database concept	medical concept
data	1	
system	1	
retrieval	1	
lung		1
ear		1

document-concept matrix

	database concept	medical concept
doc1	1	
doc2	1	
doc3		1
doc4		1

LSI can search for concept

- Search the docs for the relevant concept related to “system”
- Find the concepts in the term-concept table, then map the concept to the doc-concept table to retrieve the docs



By the tables, we may retrieve documents with the term “system”, but contain the concept database and all relevant terms such as “data” and “retrieval”, so LSI acts like an automatic thesaurus

The Pros of using vector space model

- **Automatic** selection of index terms
- **Partial matching** of queries and documents (dealing with the case where no document contains all search terms)
- **Ranking** according to **similarity score** (dealing with large result sets)
- **Term weighting** schemes (improves retrieval performance)
- Various extensions
  - Document clustering
  - Relevance feedback (modifying query vector)

Problems with Lexical Semantics 1

- **Synonymy**: Different terms may have an **identical or a similar meaning** (weaker: words indicating the same topic).
- No associations between words are made in the vector space representation.

$$\mathrm{sim}_{\mathrm{true}}(d, q) > \cos(\angle(\vec{d}, \vec{q}))$$

Problems with Lexical Semantics 2

- Ambiguity and association in natural language
  - Polysemy: Words often have a multitude of meanings and different types of usage (more severe in very heterogeneous collections).
  - The vector space model is unable to discriminate between different meanings of the same word.

$$\mathrm{sim}_{\mathrm{true}}(d, q) < \cos(\angle(\vec{d}, \vec{q}))$$

Advantage of using LSI

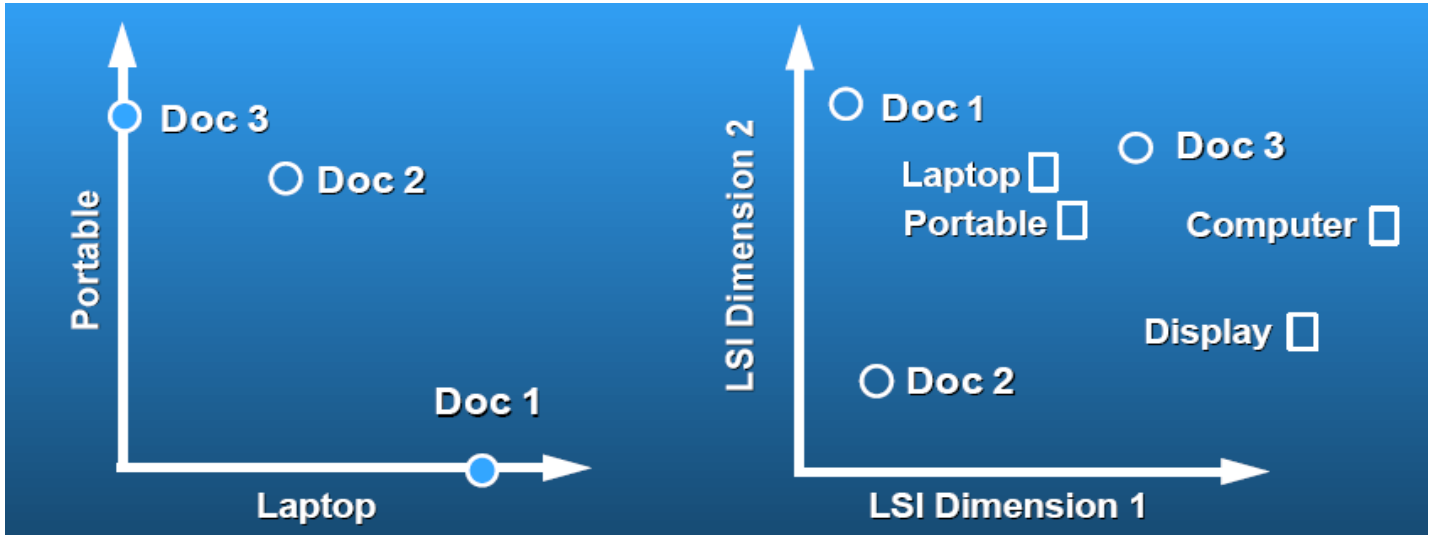
- Synonymy – two different words have the identical meaning
  - d = "Alex gave me a **gift** even if I didn't buy her a **present**"
  - q = "gift"
  - The similarity will be underestimated
- Polysemy – a term has different meaning
  - d = "I wrote the solution of problem 5 on this paper"
  - q = "solution"
  - The similarity will be overestimated ("solution" occurs in a document containing "problem" versus in a document containing "chemistry")

Main idea: Latent Semantic Indexing (LSI)

- Perform a **low-rank approximation** of **document-term matrix** (typical rank **100–300**)
- General idea
  - Map documents (and terms) to a low-dimensional representation
  - Design a mapping such that the low-dimensional space reflects **semantic associations** (latent semantic space)
  - Compute document similarity based on the **inner product** in this **latent semantic space**

Goals of LSI

- LSI takes documents that are semantically similar (i.e., the same topics), but are not similar in the vector space (because they use different words) and re-represents them in a reduced vector space in which they have higher similarity.
- Similar terms map to similar location in low dimensional space
- Noise reduction by dimension reduction



Singular Value Decomposition (SVD) for LSI

- Mathematical technique for LSI to reduce the dimensionality of the data
- $A \approx TSD^T$
  - the original matrix A is presumed to be too large to be computationally addressed
  - reducing the dimensionality helps remove some of the noise inherently present in the original matrix
  - the matrix A is sparse, so it is possible to significantly reduce the cardinality and at the same time preserve much of the information

term-concept doc-concept

$$\mathbf{A}_{[n \times m]} = \mathbf{U}_{[n \times r]} \Lambda_{[r \times r]} (\mathbf{V}_{[m \times r]})^T$$

  - U is a n by r term-concept vector matrix
  - Λ is a r by r singular values matrix with r ranked concepts in the diagonal
  - V is a m by r concept concept-doc vector matrix

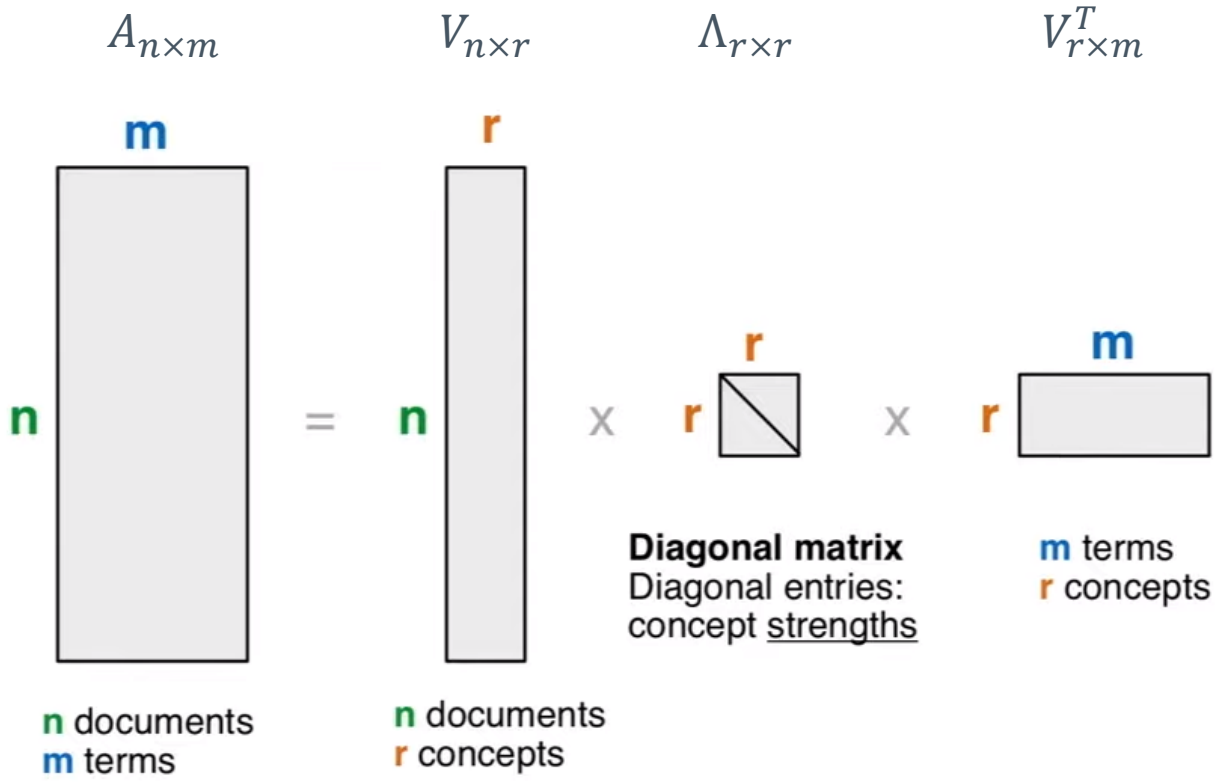
SVD for LSI

$$\mathbf{A}_{[n \times m]} = \mathbf{U}_{[n \times r]} \Lambda_{[r \times r]} (\mathbf{V}_{[m \times r]})^T$$

$U^T U = I_r, V^T V = I_r$
$\Lambda_{11} \geq \Lambda_{22} \geq \cdots \geq \Lambda_{rr} > 0, \Lambda_{ij} = 0 \; \forall \; i \neq j$

- Then we truncate the SVD and keep the largest $k \ll r$ diagonal entries in Λ
- Typical value for k is between 100 to 300 dimensions
- The dimension of term-concept matrix: $n \times r \rightarrow n \times k$
- The dimension of doc-concept matrix: $m \times r \rightarrow m \times k$
- SVD operation and dimensional reduction

SVD definition

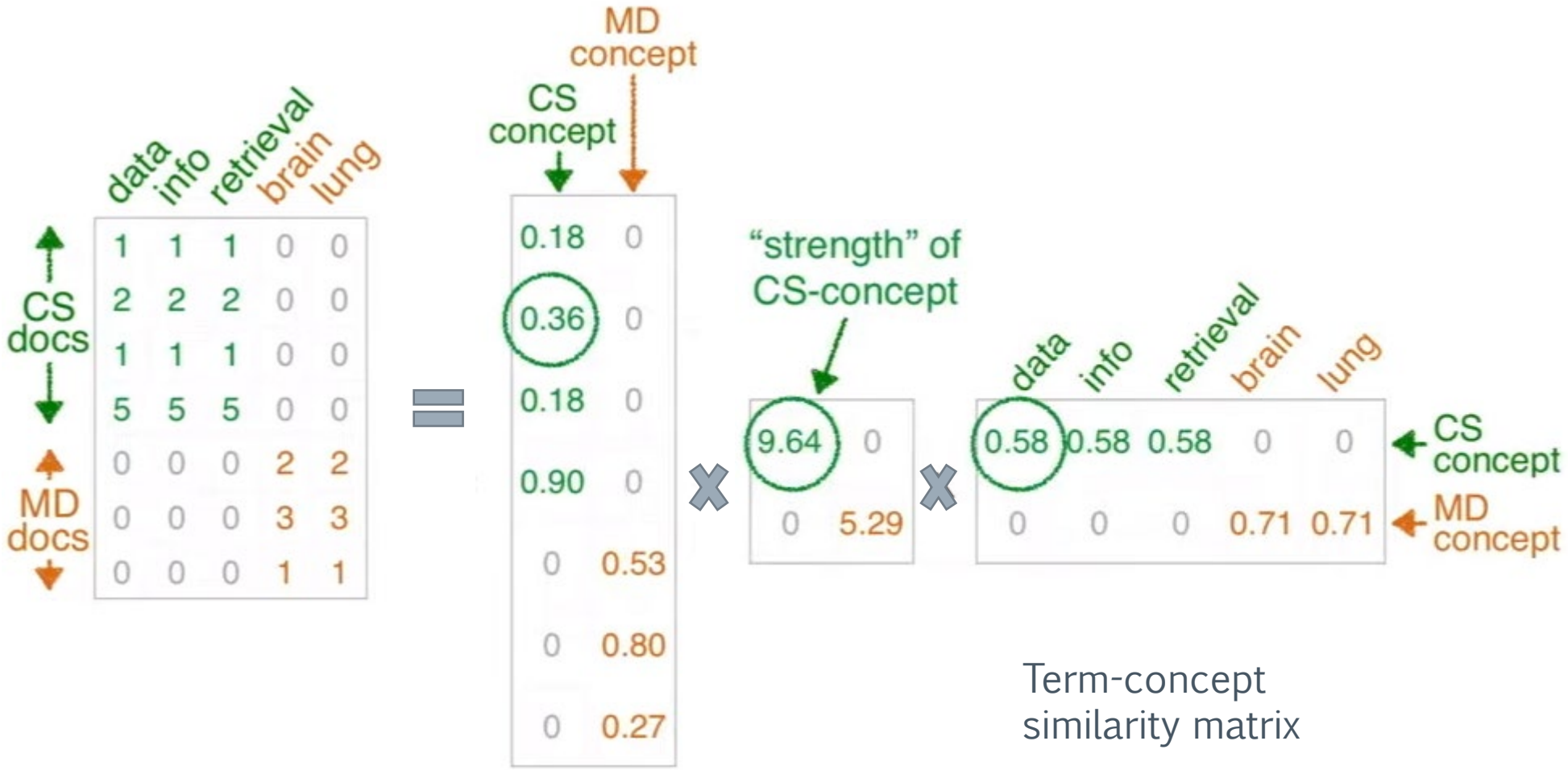


- Theorem: always possible to decompose matrix A into $A = U\Lambda V^T$
- U, Λ, V are unique for most of the time
- The columns of U & V are unit vectors and orthogonal to each other: $U^T U = I, V^T V = I$
- Λ: diagonal matrix with non-negative diagonal entries sorted in descending order
- A can be approximated by the top-k from r

Perform the maps

- Each row and column of A gets mapped into the k-dimensional LSI space, by the SVD.
- Not only the mapping with the best (Frobenius error) approximation to A, but in fact improves retrieval.
- The entered query q is also mapped into the space by $q_k = q^T U_k \Lambda_k^{-1}$
- Note that the query is NOT a sparse vector

LSI example 1



Document-concept similarity matrix

Term-concept similarity matrix

LSI example 2

Term-doc matrix - A

A	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

Term-concept matrix - U

U	1	2	3	4	5
ship	−0.44	−0.30	0.57	0.58	0.25
boat	−0.13	−0.33	−0.59	0.00	0.73
ocean	−0.48	−0.51	−0.37	0.00	−0.61
wood	−0.70	0.35	0.15	−0.58	0.16
tree	−0.26	0.65	−0.41	0.58	−0.09

Diagonal singular value matrix - Λ

Λ	1	2	3	4	5
1	2.16	0.00	0.00	0.00	0.00
2	0.00	1.59	0.00	0.00	0.00
3	0.00	0.00	1.28	0.00	0.00
4	0.00	0.00	0.00	1.00	0.00
5	0.00	0.00	0.00	0.00	0.39

Document-concept matrix - V^T

V^T	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.28	−0.75	0.45	−0.20	0.12	−0.33
4	0.00	0.00	0.58	0.00	−0.58	0.58
5	−0.53	0.29	0.63	0.19	0.41	−0.22

Original term table vs reduced term table

A	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

A_2	d_1	d_2	d_3	d_4	d_5	d_6
ship	0.85	0.52	0.28	0.13	0.21	−0.08
boat	0.36	0.36	0.16	−0.20	−0.02	−0.18
ocean	1.01	0.72	0.36	−0.04	0.16	−0.21
wood	0.97	0.12	0.20	1.03	0.62	0.41
tree	0.12	−0.39	−0.08	0.90	0.41	0.49

Benefit of the reduced dimension matrix

- Similarity of d2 and d3 in the original space: 0
- Similarity of d2 and d3 in the reduced space: $0.52 * 0.28 + 0.36 * 0.16 + 0.72 * 0.36 + 0.12 * 0.20 + -0.39 * -0.08 \approx 0.52$
- Typically, LSA increases recall and hurts precision

Empirical evidence

- Precision at or above median TREC precision
  - Top scorer on almost 20% of TREC topics
- Slightly better on average than straight vector spaces
- Effect of dimensionality:

Dimensions	Precision
250	0.367
300	0.371
346	0.374

Some extrapolation

- The “dimensionality” of a corpus is the number of distinct topics represented in it.
- More mathematical extrapolation:
  - if A has a rank k approximation of low Frobenius error, then there are no more than k distinct topics in the corpus.
  - Dimension reduction through LSI brings together “related” axes in the vector space -- clustering

Other applications of LSI

- In many settings in pattern recognition and retrieval, we have a feature-object matrix.
  - For text, the terms are features and the docs are objects.
  - Could be opinions and users …
  - This matrix may be redundant in dimensionality.
  - Can work with low-rank approximation.
  - If entries are missing (e.g., users' opinions), can recover if dimensionality is low.
- Powerful general analytical technique
  - Close, principled analog to clustering methods.

Exercise questions

› Latent semantic indexing is used to

A. Dimension reduction
B. Text classification
C. Text summarization
D. All above

- Which Latent semantic indexing (LSI) technique is used to reduce the number of rows while preserving the similarity structure among columns
  A. Principal component analysis
  B. Singular value decomposition
  C. Both principal component analysis and singular value decomposition
  D. None of these