

Latent Semantic Analysis (LSA)



Berlin Chen

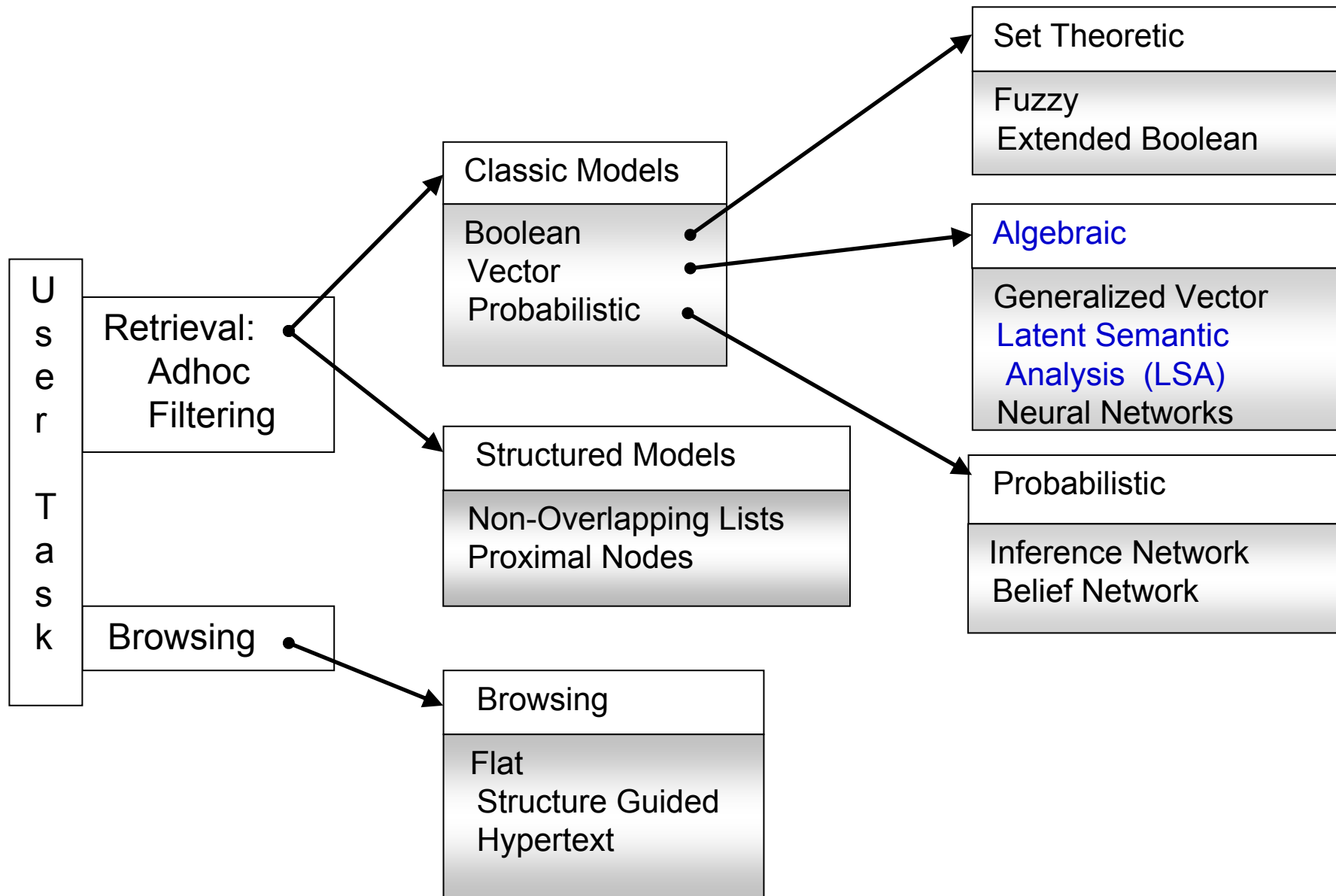
Department of Computer Science & Information Engineering
National Taiwan Normal University



References:

1. G.W.Furnas, S. Deerwester, S.T. Dumais, T.K. Landauer, R. Harshman, L.A. Streeter, K.E. Lochbaum, "Information Retrieval using a Singular Value Decomposition Model of Latent Semantic Structure," ACM SIGIR Conference on R&D in Information Retrieval , 1988
2. J.R. Bellegarda, "Latent semantic mapping," IEEE Signal Processing Magazine, September 2005

Taxonomy of Classic IR Models



Classification of IR Models Along Two Axes

- **Matching Strategy**
 - Literal term matching
 - E.g., Vector Space Model (VSM), Hidden Markov Model (HMM), Language Model (LM)
 - Concept matching
 - E.g., Latent Semantic Analysis (LSA), Probabilistic Latent Semantic Analysis (PLSA), Topical Mixture Model (TMM)
- **Learning Capability**
 - Term weighting, query expansion, document expansion, etc
 - E.g., Vector Space Model, Latent Semantic Indexing
 - Most models are based on linear algebra operations
 - Solid statistical foundations (optimization algorithms)
 - E.g., Hidden Markov Model (HMM), Probabilistic Latent Semantic Analysis (PLSA), Latent Dirichlet Allocation (LDA)
 - Most models belong to the language modeling approach

Two Perspectives for IR Models (cont.)

- Literal Term Matching vs. Concept Matching



香港星島日報篇報導引述軍事觀察家的話表示，到二零零五年台灣將完全喪失空中優勢，原因是中國大陸戰機不論是數量或是性能上都將超越台灣，報導指出中國在大量引進俄羅斯先進武器的同時也得加快研發自製武器系統，目前西安飛機製造廠任職的改進型飛豹戰機即將部署尚未與蘇愷三十通道地對地攻擊住宅飛機，以督促遇到挫折的監控其戰機目前也已經取得了重大階段性的認知成果。根據日本媒體報導在台海戰爭隨時可能爆發情況之下北京方面的基本方針，使用高科技答應局部戰爭。因此，解放軍打算在二零零四年前又有包括蘇愷三十二期在內的兩百架蘇霍伊戰鬥機。

- There are usually many ways to express a given concept, so literal terms in a user's query may not match those of a relevant document

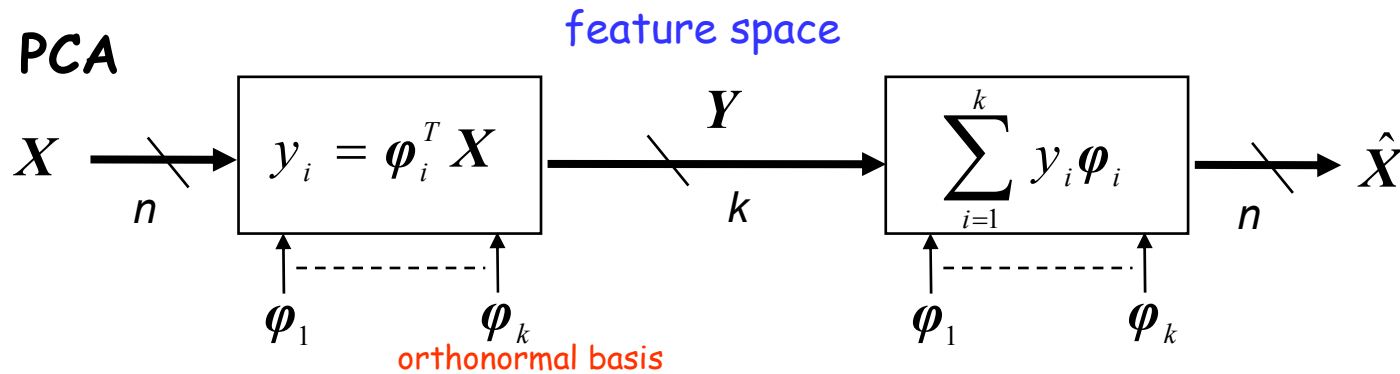
Latent Semantic Analysis

- Also called Latent Semantic Indexing (LSI), Latent Semantic Mapping (LSM), or Two-Mode Factor Analysis

Latent Semantic Analysis: Schematic

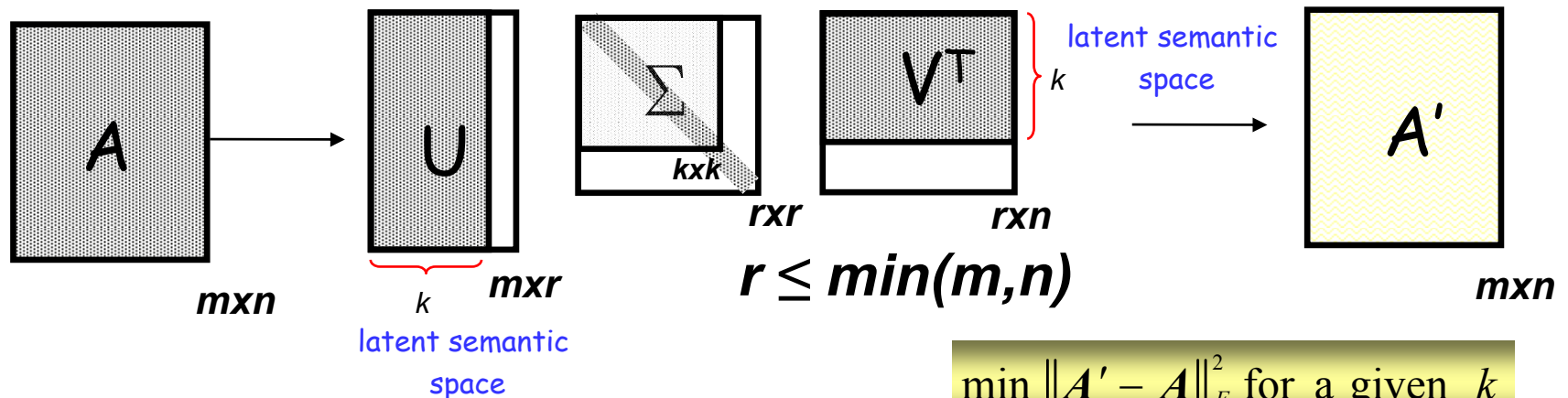
- Dimension Reduction and Feature Extraction

- PCA



$$\min \left\| \hat{X} - X \right\|^2 \text{ for a given } k$$

- SVD (in LSA)



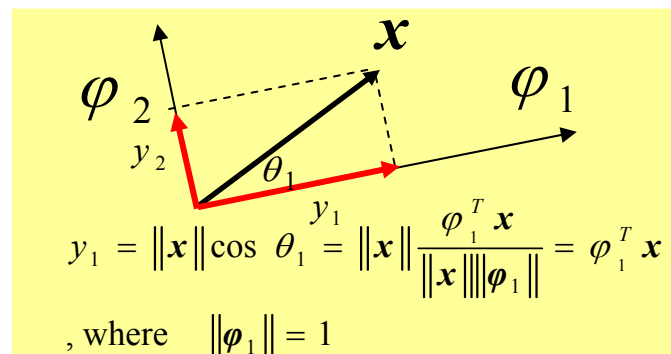
$$\min \left\| A' - A \right\|_F^2 \text{ for a given } k$$

LSA: An Example

- Singular Value Decomposition (SVD) used for the word-document matrix
 - A least-squares method for dimension reduction

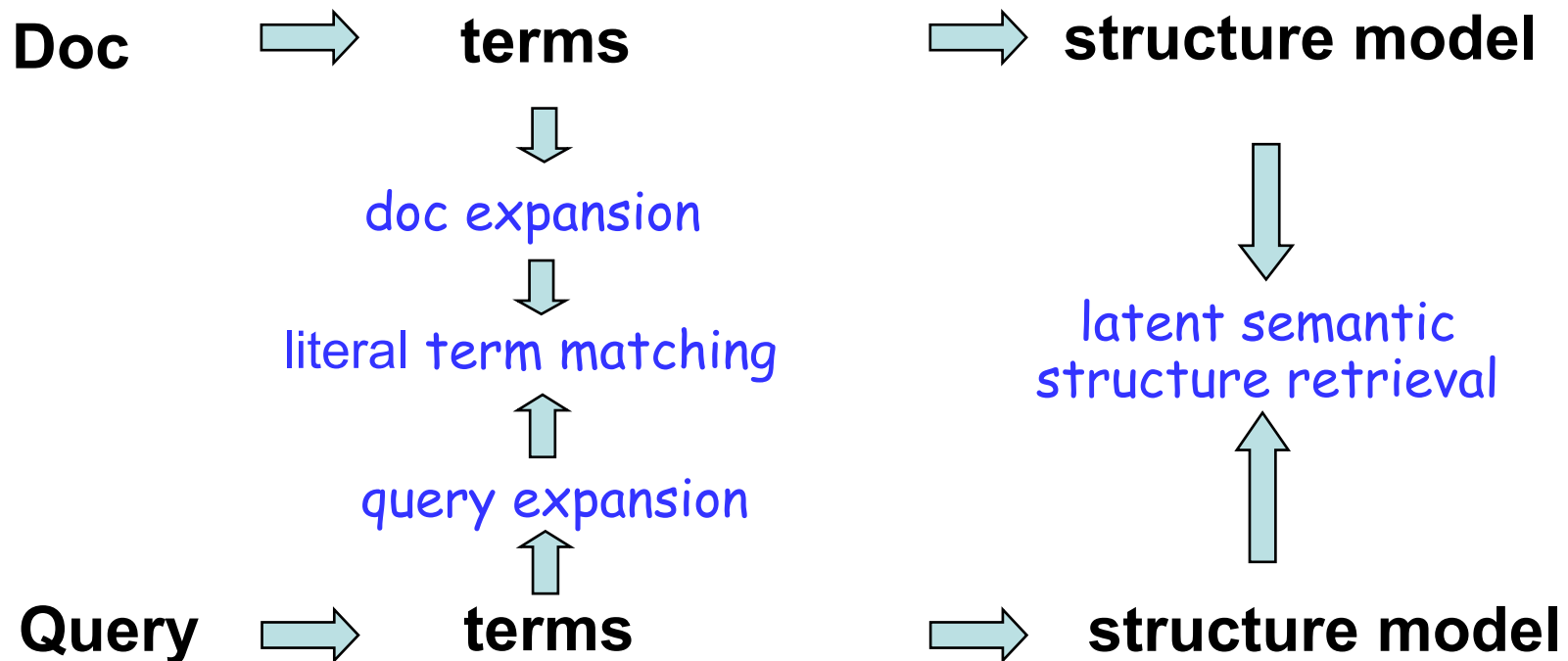
	Term 1	Term 2	Term 3	Term 4
Query	user	interface		
Document 1	user	interface	HCI	interaction
Document 2			HCI	interaction

Projection of a Vector \mathbf{x} :



LSA: Latent Structure Space

- Two alternative frameworks to circumvent vocabulary mismatch



LSA: Another Example (1/2)

Titles

- c1: *Human machine interface for Lab ABC computer applications*
 c2: *A survey of user opinion of computer system response time*
 c3: *The EPS user interface management system*
 c4: *System and human system engineering testing of EPS*
 c5: *Relation of user-perceived response time to error measurement*
 m1: *The generation of random, binary, unordered trees*
 m2: *The intersection graph of paths in trees*
 m3: *Graph minors IV: Widths of trees and well-quasi-ordering*
 m4: *Graph minors: A survey*

Terms

Documents

		c1	c2	c3	c4	c5	m1	m2	m3	m4
1.	<i>human</i>	1	0	0	1	0	0	0	0	0
2.	<i>interface</i>	1	0	1	0	0	0	0	0	0
3.	<i>computer</i>	1	1	0	0	0	0	0	0	0
4.	<i>user</i>	0	1	1	0	1	0	0	0	0
5.	<i>system</i>	0	1	1	2	0	0	0	0	0
6.	<i>response</i>	0	1	0	0	1	0	0	0	0
7.	<i>time</i>	0	1	0	0	1	0	0	0	0
8.	<i>EPS</i>	0	0	1	1	0	0	0	0	0
9.	<i>survey</i>	0	1	0	0	0	0	0	0	1
10.	<i>trees</i>	0	0	0	0	0	1	1	1	0
11.	<i>graph</i>	0	0	0	0	0	0	1	1	1
12.	<i>minors</i>	0	0	0	0	0	0	0	1	1

LSA: Another Example (2/2)

2-D Plot of Terms and Docs from Example

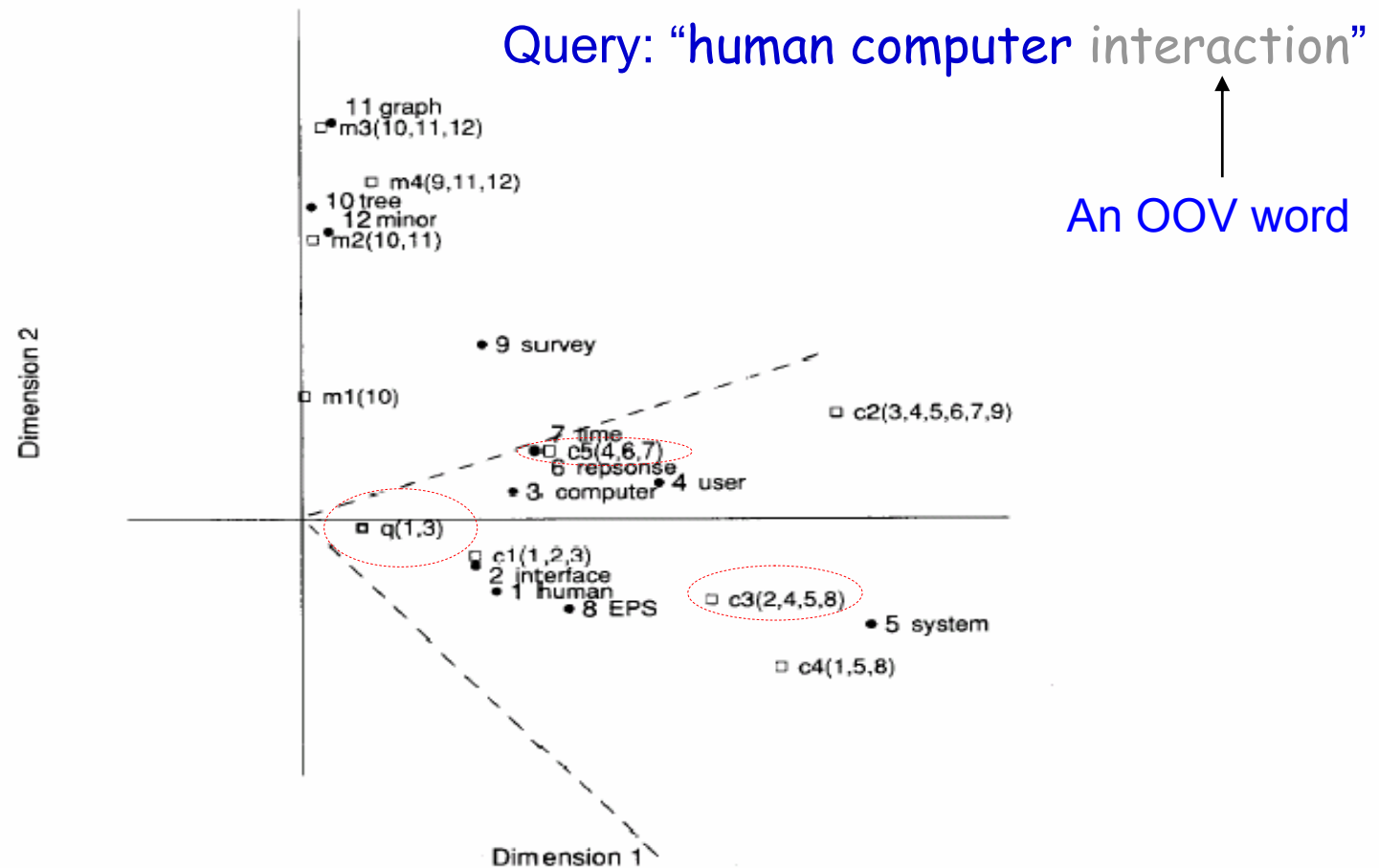
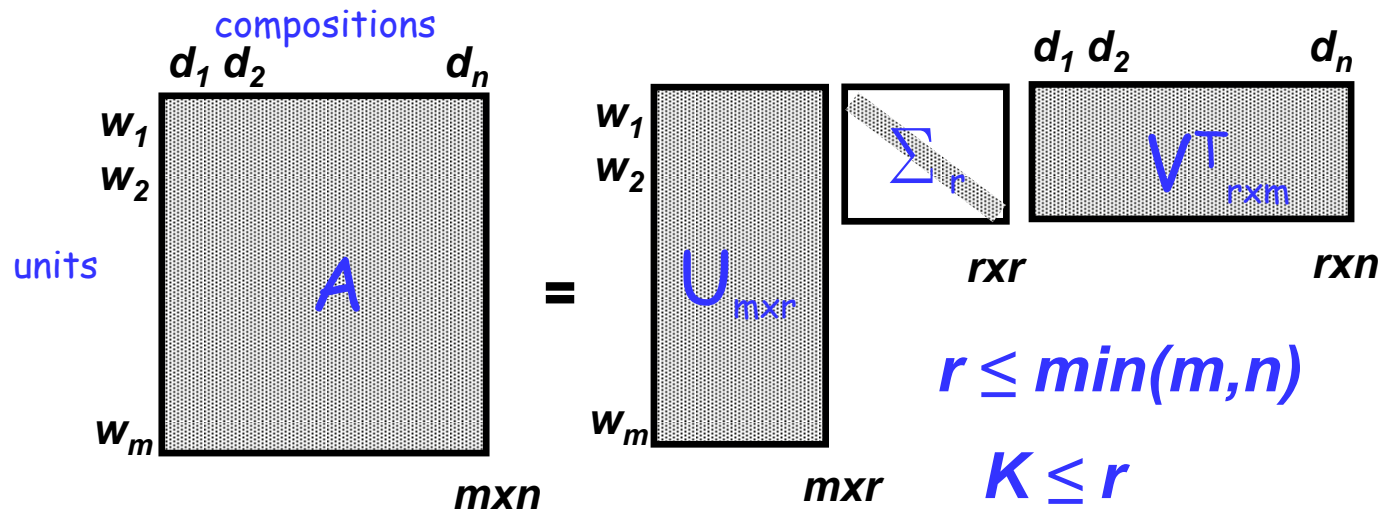


FIG. 1. A two-dimensional plot of 12 Terms and 9 Documents from the same TM set. Terms are represented by filled circles. Documents are shown as open squares, and component terms are indicated parenthetically. The query ("human computer interaction") is represented as a pseudo-document at point q . Axes are scaled for Document-Document or Term-Term comparisons. The dotted cone represents the region whose points are within a cosine of .9 from the query q . All documents about human-computer ($c1$ – $c5$) are "near" the query (i.e., within this cone), but none of the graph theory documents ($m1$ – $m4$) are nearby. In this reduced space, even documents $c3$ and $c5$ which share no terms with the query are near it.

LSA: Theoretical Foundation

- Singular Value Decomposition (SVD)

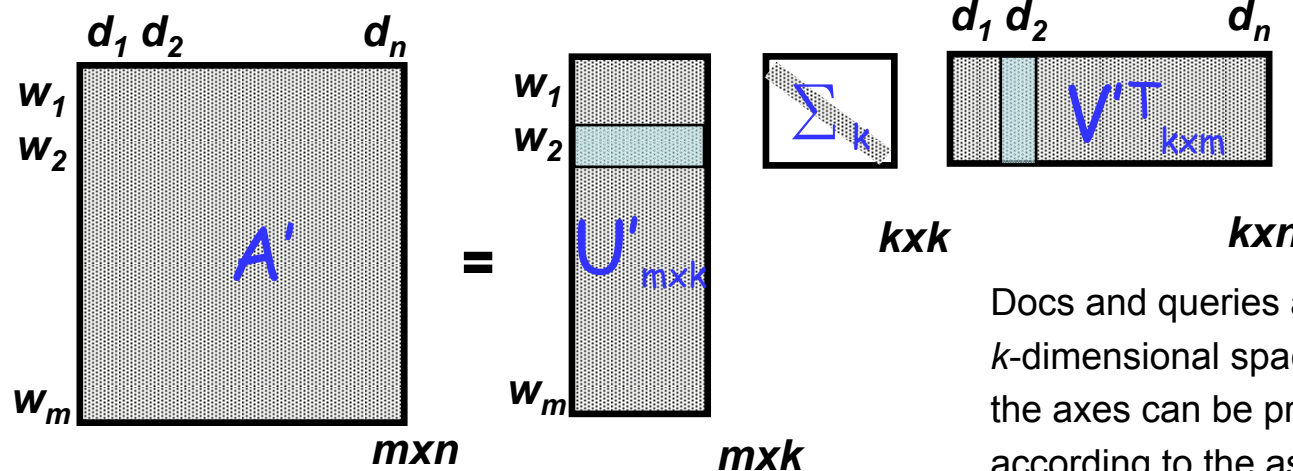
Row $A \in R^n$
 Col $A \in R^m$



Both U and V has orthonormal column vectors

$$U^T U = I_{r \times r}$$

$$V^T V = I_{r \times r}$$



$$\|A\|_F^2 \geq \|A'\|_F^2$$

$$\|A\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n a_{ij}^2$$

Docs and queries are represented in a k -dimensional space. The quantities of the axes can be properly weighted according to the associated diagonal values of Σ_k

LSA: Theoretical Foundation

- “term-document” matrix A has to do with the co-occurrences between terms (or units) and documents (or compositions)
 - Contextual information for words in documents is discarded
 - “bag-of-words” modeling
- **Feature extraction** for the entities $a_{i,j}$ of matrix A
 1. Conventional *tf-idf* statistics
 2. Or, $a_{i,j}$: occurrence frequency weighted by negative entropy

occurrence count

$$a_{i,j} = \frac{f_{i,j}}{|d_j|} \times (1 - \varepsilon_i), \quad |d_j| = \sum_{i=1}^m f_{i,j}$$

negative normalized entropy

document length

normalized entropy of term i

$$\varepsilon_i = -\frac{1}{\log n} \sum_{j=1}^n \left(\frac{f_{i,j}}{\tau_i} \log \frac{f_{i,j}}{\tau_i} \right), \quad \tau_i = \sum_{j=1}^n f_{i,j}$$

occurrence count of term i in the collection

$$0 \leq \varepsilon_i \leq 1$$

LSA: Theoretical Foundation

- Singular Value Decomposition (SVD)

- $A^T A$ is symmetric $n \times n$ matrix

- All eigenvalues λ_j are nonnegative real numbers

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0 \quad \Sigma^2 = \text{diag}(\lambda_1, \lambda_1, \dots, \lambda_n)$$

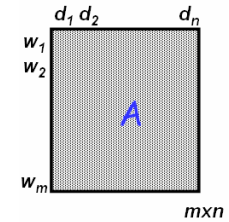
- All eigenvectors v_j are orthonormal ($\in \mathbb{R}^n$)

$$V = [v_1 \ v_2 \ \dots \ v_n] \quad v_j^T v_j = 1 \quad (V^T V = I_{n \times n})$$

- Define **singular values**: sigma $\sigma_j = \sqrt{\lambda_j}$, $j = 1, \dots, n$

- As the square roots of the eigenvalues of $A^T A$

- As the lengths of the vectors Av_1, Av_2, \dots, Av_n



For $\lambda_i \neq 0$, $i=1, \dots, r$,
 $\{Av_1, Av_2, \dots, Av_r\}$ is an
 orthogonal basis of Col A

$$\sigma_1 = \|Av_1\|$$

$$\sigma_2 = \|Av_2\|$$

.....

$$\|Av_i\|^2 = v_i^T A^T Av_i = v_i^T \lambda_i v_i = \lambda_i$$

$$\Rightarrow \|Av_i\| = \sigma_i$$

LSA: Theoretical Foundation

- $\{Av_1, Av_2, \dots, Av_r\}$ is an **orthogonal** basis of **Col A**

$$Av_i \bullet Av_j = (Av_i)^T Av_j = v_i^T A^T Av_j = \lambda_j v_i^T v_j = 0$$

- Suppose that A (or $A^T A$) has rank $r \leq n$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r > 0, \quad \lambda_{r+1} = \lambda_{r+2} = \dots = \lambda_n = 0$$

- Define an **orthonormal** basis $\{u_1, u_2, \dots, u_r\}$ for Col A

$$u_i = \frac{1}{\|Av_i\|} Av_i = \frac{1}{\sigma_i} Av_i \Rightarrow \sigma_i u_i = Av_i$$

U is also an
orthonormal matrix
($m \times r$)

$$\Rightarrow [u_1 \ u_2 \ \dots \ u_r] \Sigma_r = A [v_1 \ v_2 \ \dots \ v_r]$$

V : an orthonormal matrix

Known in advance

- Extend to an orthonormal basis $\{u_1, u_2, \dots, u_m\}$ of R^m

$$\Rightarrow [u_1 \ u_2 \ \dots \ u_r \ \dots \ u_m] \Sigma = A [v_1 \ v_2 \ \dots \ v_r \ \dots \ v_n]$$

$$\Rightarrow U \Sigma = AV \Rightarrow U \Sigma V^T = A \underbrace{V V^T}_{I_{n \times n}}$$

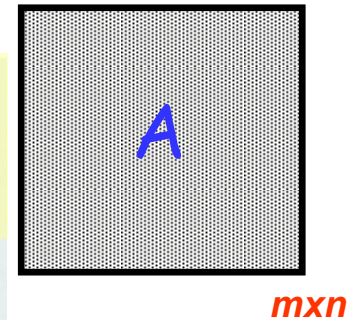
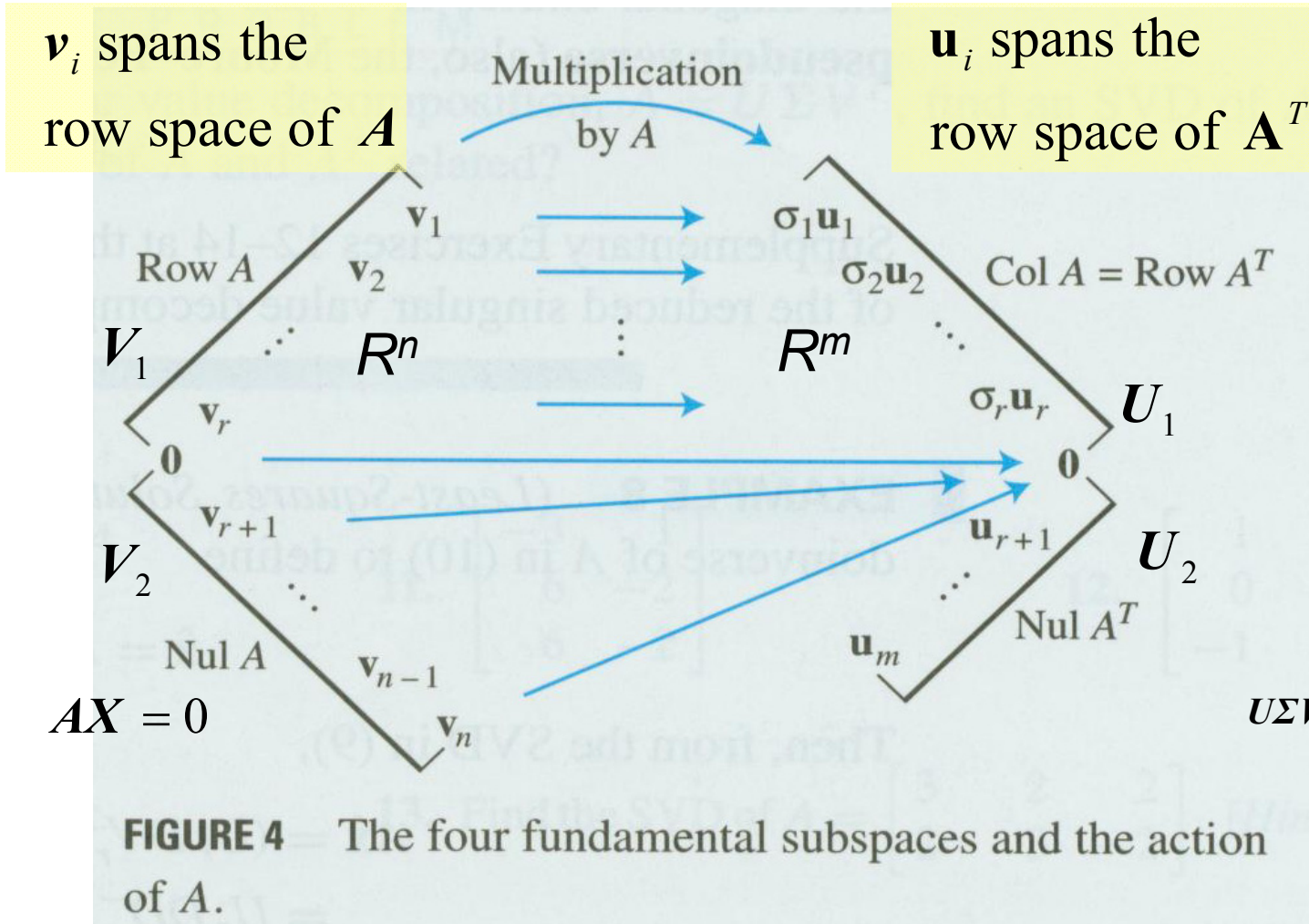
$$\Rightarrow A = U \Sigma V^T \quad I_{n \times n} \quad ?$$

$$\Sigma_{m \times n} = \begin{pmatrix} \Sigma_r & \mathbf{0}_{r \times (n-r)} \\ \mathbf{0}_{(m-r) \times r} & \mathbf{0}_{(m-r) \times (n-r)} \end{pmatrix}$$

$$\|A\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n a_{ij}^2$$

$$\|A\|_F^2 = \sigma_1^2 + \sigma_2^2 + \dots + \sigma_r^2 \quad ?$$

LSA: Theoretical Foundation



$$\begin{aligned}
 U \Sigma V^T &= (U_1 \ U_2) \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\
 &= U_1 \Sigma_1 V_1^T \\
 &= A V_1 V_1^T \\
 &= A
 \end{aligned}$$

$U \Sigma = A V$

LSA: Theoretical Foundation

- Additional Explanations

- Each row of U is related to the projection of a corresponding row of A onto the basis formed by columns of V

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

$$\Rightarrow AV = U\Sigma V^T V = U\Sigma \Rightarrow U\Sigma = AV$$

- the i -th entry of a row of U is related to the projection of a corresponding row of A onto the i -th column of V

- Each row of V is related to the projection of a corresponding row of A^T onto the basis formed by U

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

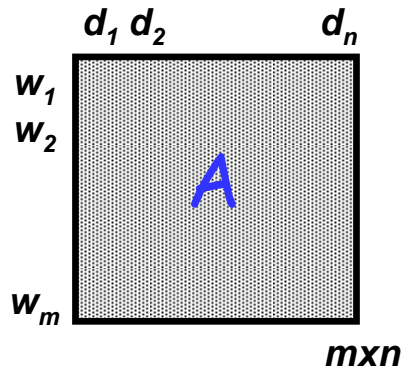
$$\Rightarrow A^T U = (U\Sigma V^T)^T U = V\Sigma U^T U = V\Sigma$$

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

- the i -th entry of a row of V is related to the projection of a corresponding row of A^T onto the i -th column of U

LSA: Theoretical Foundation

- Fundamental comparisons based on SVD
 - The original word-document matrix (A)



- compare two terms \rightarrow dot product of two rows of A
 - or an entry in AA^T
- compare two docs \rightarrow dot product of two columns of A
 - or an entry in $A^T A$
- compare a term and a doc \rightarrow each individual entry of A

– The new word-document matrix (A')

$$U' = U_{m \times k}$$

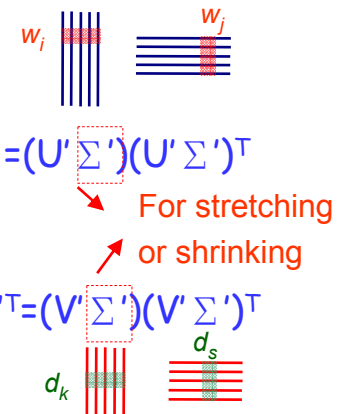
$$\Sigma' = \Sigma_k$$

$$V' = V_{n \times k}$$

- compare two terms \rightarrow dot product of two rows of $U' \Sigma'$

$$A' A'^T = (U' \Sigma' V'^T) (U' \Sigma' V'^T)^T = U' \Sigma' V'^T V' \Sigma'^T U'^T = (U' \Sigma') (U' \Sigma')^T$$
- compare two docs \rightarrow dot product of two rows of $V' \Sigma'$

$$A'^T A' = (U' \Sigma' V'^T)^T (U' \Sigma' V'^T) = V' \Sigma'^T U'^T U' \Sigma' V'^T = (V' \Sigma') (V' \Sigma')^T$$
- compare a query word and a doc \rightarrow each individual entry of A'



LSA: Theoretical Foundation

- **Fold-in:** find representations for pseudo-docs q
 - For objects (new queries or docs) that did not appear in the original analysis
 - Fold-in a new $m \times 1$ query (or doc) vector

See Figure A in next page

$$\hat{q}_{1 \times k} = \left(q^T \right)_{1 \times m} U_{m \times k} \Sigma^{-1}_{k \times k}$$

Just like a row of V

Query represented by the weighted sum of its constituent term vectors

The separate dimensions are differentially weighted

- Represented as the weighted sum of its component word (or term) vectors
- Cosine measure between the query and doc vectors in the latent semantic space

$$\text{sim} \left(\hat{q}, \hat{d} \right) = \text{coine} \left(\hat{q} \Sigma, \hat{d} \Sigma \right) = \frac{\hat{q} \Sigma^2 \hat{d}^T}{\left| \hat{q} \Sigma \right| \left| \hat{d} \Sigma \right|}$$

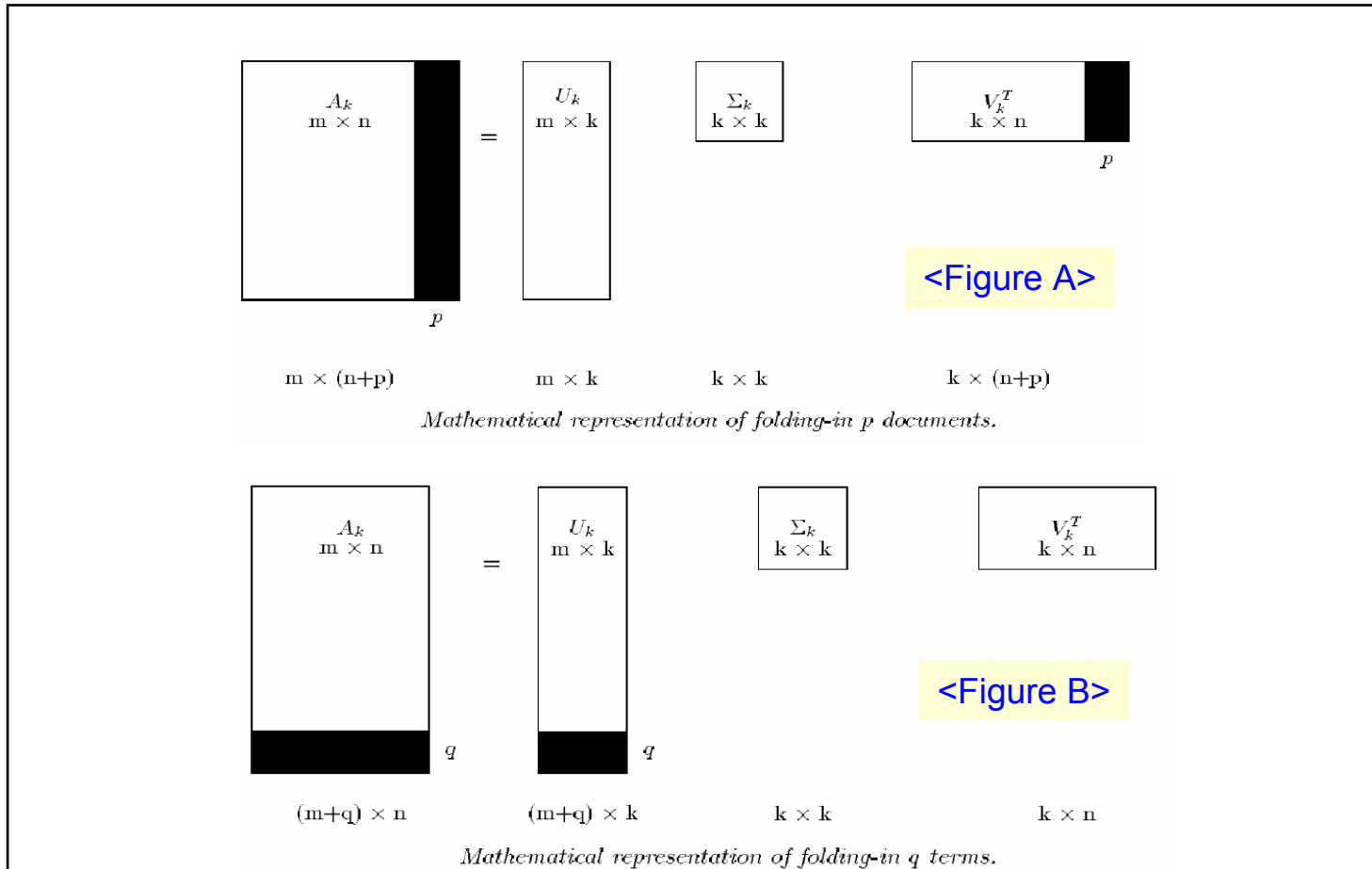
row vectors

LSA: Theoretical Foundation

- Fold-in a new 1 x n term vector

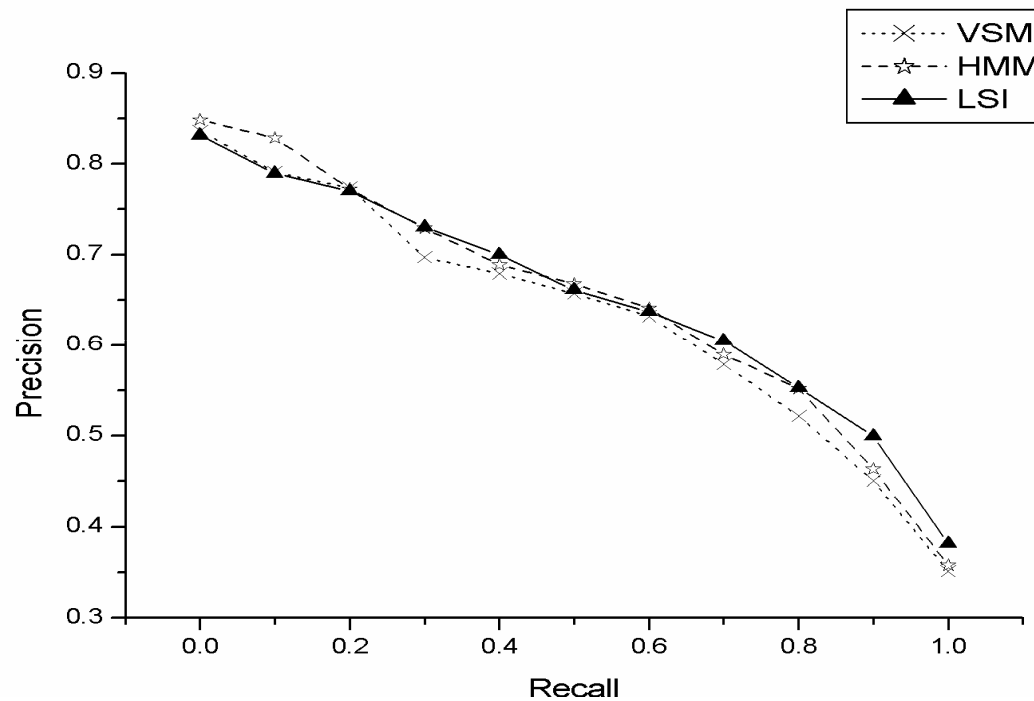
$$\hat{t}_{1 \times k} = t_{1 \times n} V_{n \times k} \Sigma_{k \times k}^{-1} \mathbf{1}_{k \times k}$$

See Figure B below



LSA: A Simple Evaluation

- Experimental results
 - HMM is consistently better than VSM at all recall levels
 - LSA is better than VSM at higher recall levels



Recall-Precision curve at 11 standard recall levels evaluated on TDT-3 SD collection. (Using word-level indexing terms)

LSA: Pro and Con (1/2)

- Pro (Advantages)
 - A clean formal framework and a clearly defined optimization criterion (least-squares)
 - Conceptual simplicity and clarity
 - Handle synonymy problems (“heterogeneous vocabulary”)
 - Replace individual terms as the descriptors of documents by independent “artificial concepts” that can be specified by any one of several terms (or documents) or combinations
 - Good results for high-recall search
 - Take term co-occurrence into account

LSA: Pro and Con (2/2)

- Disadvantages
 - High computational complexity (e.g., SVD decomposition)
 - Exhaustive comparison of a query against all stored documents is needed (cannot make use of inverted files ?)
 - LSA offers only a partial solution to polysemy (e.g. bank, bass,...)
 - Every term is represented as just one point in the latent space (represented as weighted average of different meanings of a term)


LSA: SVDLIBC

- Doug Rohde's SVD C Library version 1.3 is based on the [SVDPACKC](#) library
- Download it at <http://tedlab.mit.edu/~dr/>

LSA: Exercise (1/4)

- Given a sparse term-document matrix
 - E.g., 4 terms and 3 docs

	Doc		
Term	2.3	0.0	4.2
	0.0	1.3	2.2
	3.8	0.0	0.5
	0.0	0.0	0.0



Row #Tem	Col. # Doc	Nonzero entries
4	3	6
2		2 nonzero entries at Col 0
0	2.3	Col 0, Row 0
2	3.8	Col 0, Row 2
1		1 nonzero entry at Col 1
1	1.3	Col 1, Row 1
3		3 nonzero entry at Col 2
0	4.2	Col 2, Row 0
1	2.2	Col 2, Row 1
2	0.5	Col 2, Row 2

- Each entry can be weighted by *TFxIDF* score
- Perform SVD to obtain term and document vectors represented in the latent semantic space
- Evaluate the information retrieval capability of the LSA approach by using varying sizes (e.g., 100, 200, ..., 600 etc.) of LSA dimensionality

LSA: Exercise (2/4)

- Example: term-document matrix

Indexing Term no.	Doc no.	Nonzero entries
51253	2265	218852
77		
508	7.725771	
596	16.213399	
612	13.080868	
709	7.725771	
713	7.725771	
744	7.725771	
1190	7.725771	
1200	16.213399	
1259	7.725771	
.....		

- SVD command (IR_svd.bat)

`svd -r st -o LSA100 -d 100 Term-Doc-Matrix`

Annotations:

- `-r st`: sparse matrix input
- `-o LSA100`: prefix of output files
- `-d 100`: No. of reserved eigenvectors
- `Term-Doc-Matrix`: name of sparse matrix input

output →

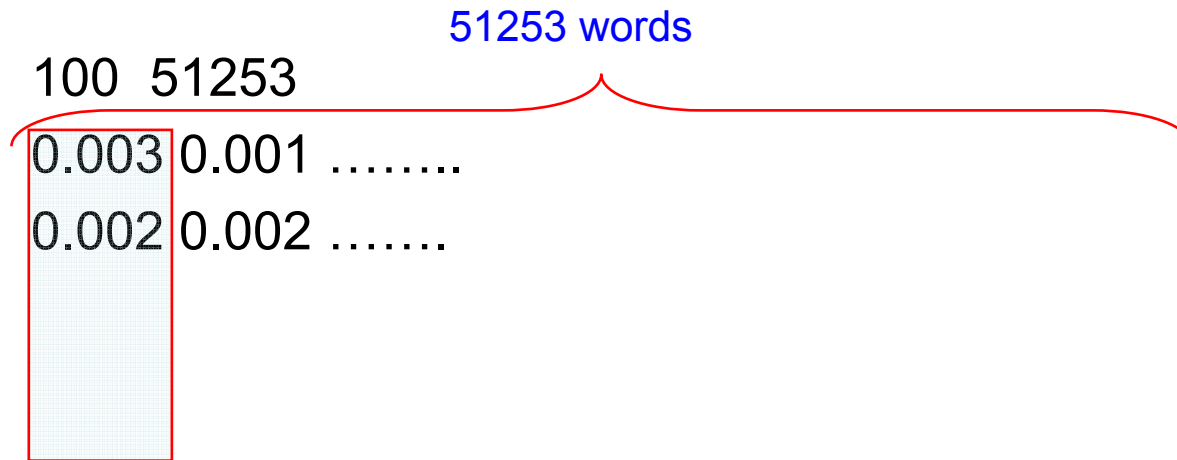
LSA100-Ut

LSA100-S

LSA100-Vt

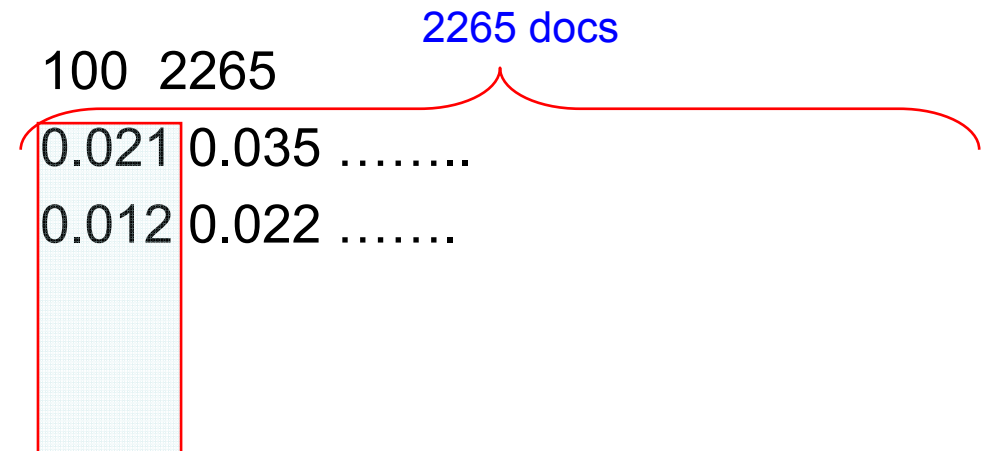
LSA: Exercise (3/4)

- **LSA100-Ut**



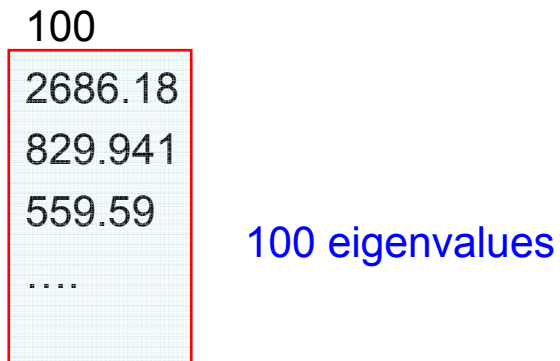
word vector (u^T): 1x100

- **LSA100-Vt**



doc vector (v^T): 1x100

- **LSA100-S**



LSA: Exercise (4/4)

- Fold-in a new $m \times 1$ query vector

$$\hat{q}_{1 \times k} = \left(q^T \right)_{1 \times m} U_{m \times k} \Sigma^{-1}_{k \times k}$$

Just like a row of V Query represented by the weighted sum of its constituent term vectors The separate dimensions are differentially weighted

- Cosine measure between the query and doc vectors in the latent semantic space

$$\text{sim}(\hat{q}, \hat{d}) = \text{coine}(\hat{q}\Sigma, \hat{d}\Sigma) = \frac{\hat{q}\Sigma^T \hat{d}}{|\hat{q}\Sigma| |\hat{d}\Sigma|}$$