

# Latent Semantic Indexing and the SVD

---

# Computational Information Systems

---

- Documents: papers, books, web pages, info blobs
- Terms: words or phrases, acronyms, moving windows of sets of words (n-grams)
- Three basic problems in computational info systems
  - document retrieval: find relevant documents from a text database, via a query
  - classification: assign a new doc to its proper group, category or class [pre-existing categories]
  - clustering: find homogeneous clusters of docs [define the category structure, instead of using predetermined ones]

# Computational Information Systems

---

- Difficulties
  - Handling multiple languages
  - Polysemy: words with multiple meanings
  - Document purification
    - formatting: `<comment>`, `<alt text>`, `{\it ... }`
    - stop words: *a, an, the, of, ...*
    - stemming: *prefixes, prefixing, prefix, prefixed*
  - Scale of problem;  $10^5$  English words,  $10^{10}$  web pages
- Need to use some form of dimension reduction (information compression, data reduction, ...)

# SVD Properties

---

- Recall: if  $Q_1$  and  $Q_2$  are  $m \times m$  and  $n \times n$  orthogonal matrices, then  $\|Q_1 A Q_2\|_2 = \|A\|_2$  for any  $m \times n$  matrix  $A$ .
- Same holds for Frobenius norm.
- Let  $A = USV^T$  be full SVD, with singular values ordered as
  - $s_1 \geq s_2 \geq \dots \geq s_r > s_{r+1} = \dots = s_p = 0$ , with  $p = \min(m,n)$
  - $\text{rank}(A) = r$
  - $\|A\|_2 = s_1$
  - $\|A\|_F = \sqrt{s_1^2 + s_2^2 + \dots + s_p^2}$
  - $\text{range}(A) = \text{span}\{u_1, u_2, \dots, u_r\}$
  - $\text{null}(A) = \text{span}\{v_{r+1}, v_{r+2}, \dots, v_n\}$

# SVD Properties

---

- Let  $A = USV^T$  be the full SVD, and let  $r = \text{rank}(A)$
- $\min \|Ax - b\|_2$  solved by  $x = V S^\Psi U^T b$ , where
$$S^\Psi = \text{diag}(s_1^\Psi, s_2^\Psi, \dots, s_r^\Psi)$$
 is  $n \times m$ , and
$$s_i^\Psi = 1/s_i, \text{ if } s_i \neq 0 \text{ but } s_i^\Psi = 0 \text{ otherwise}$$
- More precisely, this solution  $x$  is the *unique* minimum norm least squares solution
- Sometimes the matrix  $VS^\Psi U^T$  is called the *pseudoinverse* of  $A$ . What are its dimensions?

# SVD Properties

---

- Can write the SVD as a sum of rank-one matrices:
- $A = \sum_{i=1}^r s_i u_i v_i^T$ , sum from  $i=1$  to  $i=r = \text{rank}(A)$  or as
- $A = \sum_{i=1}^p s_i u_i v_i^T$ , sum from  $i=1$  to  $i=p = \max(m,n)$
- Define  $A_k = \sum_{i=1}^k s_i u_i v_i^T$  with sum from  $i=1$  to  $i=k$
- Then  $A_k$  is the best rank- $k$  approximation to  $A$  in the 2-norm ...

# SVD Properties

---

- Then  $A_k$  is the best rank- $k$  approximation to  $A$  in the 2-norm:
  - $\|A_k - A\|_2 = \min \|B - A\|_2$  over all  $m \times n$   $B$  of rank  $k$
  - $\|A_k - A\|_2 = s_{k+1}$ , the  $k+1$ -th singular value
  - Same minimization result holds for Frobenius norm
  - This is one way to achieve information compression in computational information retrieval
  - The matrix  $A$  requires in general up to  $m \times n$  doubles to be stored to represent it
  - The matrix  $A_k$  requires at most  $k \cdot (m+n) + 1$  doubles
  - Full storage of  $m \times n$  doubles is for a *dense* matrix, but even for *sparse* matrices savings is immense

# Term-document matrix

---

- Suppose we have  $t$  terms and  $d$  documents
- Define  $A$  as the  $t \times d$  matrix with  $A(i,j) =$  number of times term  $i$  appears in document  $j$ .
- Columns of  $A \rightarrow$  documents; rows of  $A \rightarrow$  terms
- Matrix  $A$  will generally be *sparse*: most of its entries are going to be zeros.
- Sparse matrices are far more common in modern scientific computing than dense or *full* matrices (which are often stored as 2D arrays)
- Definition of sparse is not precise: the matrix has enough zeros in it to be worthwhile to take advantage of (and so to not store them or do flops with them)

# Term-document matrix

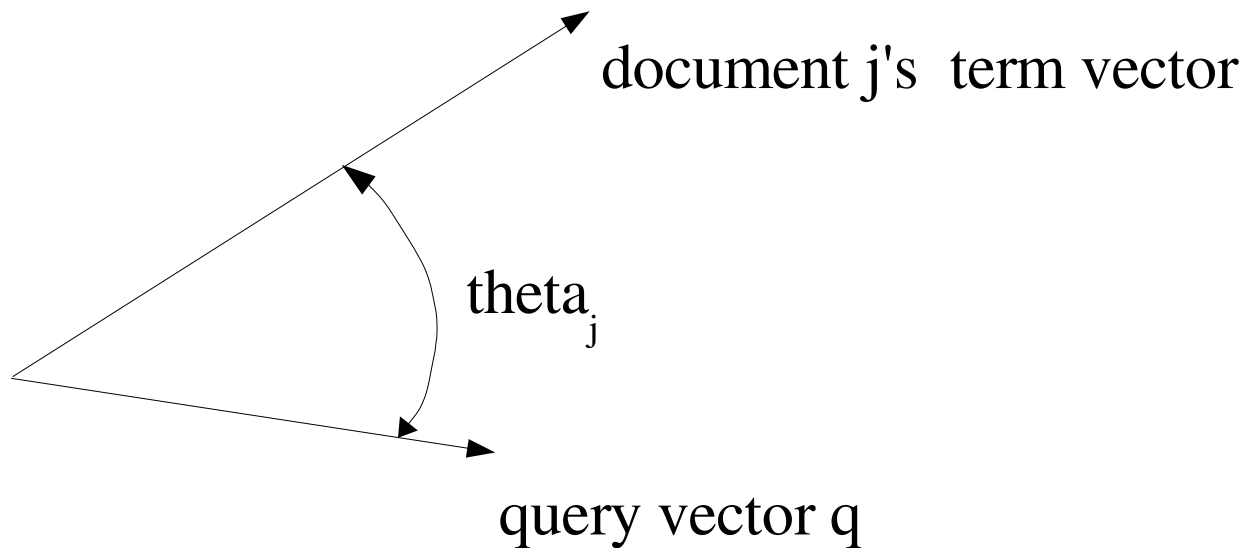
---

- A query can be treated as another document
  - $q(i) = 1$  if term is keyword in query
  - $q(i) = 0$  otherwise
- Relevance of a document vector is measured by the angle it makes with the query vector
  - angle close to zero: document is highly relevant (the vectors are nearly aligned and hence are close)
  - angle close to 90 degrees: document is unrelated (the vectors point in completely different directions)
  - we ignore the norm of the vector, or equivalently, normalize all the vectors to have norm 1
  - will use the cosine of the angle. So should cosine be large or small for a document closely related to the query?

# Term-document and queries

---

- Measure of angle between a query and documents:
  - $\cos(\theta_j) = \frac{a_j^T q}{(\|q\| \|a_j\|)}$
  - can get all cosines by  $A^T q$ , matrix-vector product



# Examples (using titles for the full docs)

---

- D1: Infant and Toddler First Aid
- D2: Babie's and Children's Room For Your Home
- D3: Child Safety at Home
- D4: Your Babie's Health and Safety: from Infant to Toddler
- D5: Baby Proofing Basics
- D6: Your Guide to Easy Rust Proofing
- D7: Beanie Babies Collectors Guide

# Example Terms (Dictionary)

---

- T1: bab(y,ies,y's)
- T2: child(ren, ren's)
- T3: guide
- T4: health
- T5: home
- T6: infant
- T7: proofing
- T8: safety
- T9: toddler

# Term-document matrix example

---

D1	D2	D3	D4	D5	D6	D7	
0	1	0	1	1	0	1	(baby)
0	1	1	0	0	0	0	(child)
0	0	0	0	0	1	1	(guide)
0	0	0	1	0	0	0	(health)
0	1	1	0	0	0	0	(home)
1	0	0	1	0	0	0	(infant)
0	0	0	0	1	1	0	(proofing)
0	0	1	1	0	0	0	(safety)
1	0	0	1	0	0	0	(toddler)

# Term-document query: child proofing

---

q:

0 (baby)

1 (child)

0 (guide)

0 (health)

0 (home)

0 (infant)

1 (proofing)

0 (safety)

0 (toddler)

# Preprocessing: Normalize Columns

---

- Divide each column of  $A$  by its norm.
  - Reason: a lengthy document would attract more hits than it really should
  - Relevance should be measured by *relative* frequency of term, not by its absolute count
- What norm should be used here?
- Should we also divide each row by its norm?

# Normalized Matrix (x 10)

---

0	5.7735	0	4.4721	7.0711	0	7.0711
0	5.7735	5.7735	0	0	0	0
0	0	0	0	0	7.0711	7.0711
0	0	0	4.4721	0	0	0
0	5.7735	5.7735	0	0	0	0
7.0711	0	0	4.4721	0	0	0
0	0	0	0	7.0711	7.0711	0
0	0	5.7735	4.4721	0	0	0
7.0711	0	0	4.4721	0	0	0

# Normalized Matrix (x 10)

---

0	5.7735	0	4.4721	7.0711	0	7.0711	0
0	5.7735	5.7735	0	0	0	0	1
0	0	0	0	0	7.0711	7.0711	0
0	0	0	4.4721	0	0	0	0
0	5.7735	5.7735	0	0	0	0	0
7.0711	0	0	4.4721	0	0	0	0
0	0	0	0	7.0711	7.0711	0	1
0	0	5.7735	4.4721	0	0	0	0
7.0711	0	0	4.4721	0	0	0	0

Query vector: "Child Proofing"

# Normalized Matrix (x 10)

---

0	5.7735	0	4.4721	7.0711	0	7.0711	0
0	5.7735	5.7735	0	0	0	0	0.7071
0	0	0	0	0	7.0711	7.0711	0
0	0	0	4.4721	0	0	0	0
0	5.7735	5.7735	0	0	0	0	0
7.0711	0	0	4.4721	0	0	0	0
0	0	0	0	7.0711	7.0711	0	0.7071
0	0	5.7735	4.4721	0	0	0	0
7.0711	0	0	4.4721	0	0	0	0
0	4.0825	4.0825	0	5.0	5.0	0	

Query vector: "Child Proofing"

Cosines: say best match with docs **D5** and **D6**: "Baby Proofing Basics", "Your Guide to Easy Rust Proofing"

# SVD and information retrieval

---

- $A = USV^T$
- $\text{range}(A) = \text{range}(u_1, \dots, u_p)$ : span of term vectors
- $\text{range}(A^T) = \text{range}(v_1, \dots, v_p)$ : span of doc vectors
- $u_1$  is a dense vector in general. Corresponds to a unit term that best spans all the terms.
- $v_1$  is dense vector in general. Corresponds to an uberdocument that best spans all the documents
- $u_1$  dense means that there is not really a single term corresponding to it – it has some linear combination of all of the terms.

# SVD and Latent Semantic Indexing

---

- The term-document matrix has an underlying semantic structure
  - concealed by wide variety of words used ("class", "category", "group") for single idea : *synonymy*
  - concealed by a single word used for disparate things ("proofing", "code") : *polysemy*
  - spellings and misspellings ("color", "colour", "coulor")  
[Andrew Jackson: *I have nothing but sympathy for a man who knows but one way to spell a word*]

# SVD and Latent Semantic Indexing

---

- Two documents with similar topics may not share keywords, but will share *associating* words
  - SVD puts those documents into same cluster, or retrieves them together on a query
  - Documents will have close semantic structures even after dimension reduction via SVD
  - Reason a web search can return a relevant document even when none of your keywords appear in the returned document

# SVD and Latent Semantic Indexing

---

- Explaining idea 1:
  - First few singular vectors contain global information from the document database. Signal in data
  - Trailing ones contain local peculiarities: variant spellings, nonstandard use of terms. Noise in data
- Idea behind this idea:  $A = \sum_{i=1}^p s_i u_i v_i^T$ , sum from  $i=1$  to  $i=p = \max(m,n)$ 
  - $u_i, v_i$  are orthonormal (same weight over  $i$ )
  - Weighting comes from  $s_i$
  - Larger weight corresponds to signal, smaller to noise

# SVD and Latent Semantic Indexing

---

- Explaining idea 2:
  - dotproduct between two doc vectors measures similarity
  - $A^T A$  measures similarities among all the docs (the doc-doc similarity matrix)
  - This similarity matrix is in term space (a  $t \times t$  matrix)
  - dotproduct between two term vectors measures their *co-occurrences* among all the documents of the database
  - $A A^T$  is the word co-occurrence matrix
  - This similarity matrix is in document space ( $d \times d$  matrix)
  - SVD gives eigenvalues/eigenvectors of those 2 matrices

# SVD and LSI

---

- $A = USV^T$  is full SVD, with singular values  
 $s_1 \geq s_2 \geq \dots \geq s_r > s_{r+1} = \dots = s_p = 0$ , with  $p = \min(m,n)$
- $A^T A = VS^2V^T$  and so columns of  $V$  are eigenvectors of the document-document similarity matrix
- Approximating  $A$  with  $A_k$  means approximating

$$A_k^T A_k = V_k S_k^2 V_k^T \text{ where}$$

$$S_k = \text{diag}(s_1, s_2, \dots, s_k, 0, 0, \dots, 0)$$

# Interpretation of Queries in SVD

---

- $\cos(\theta_j) = \mathbf{a}_j^T \mathbf{q} / (\|\mathbf{q}\| \|\mathbf{a}_j\|)$ , with columns of  $A$
- $\mathbf{q}^T A \sim \mathbf{q}^T A_k = \mathbf{q}^T (\mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^T) = (\mathbf{q}^T \mathbf{U}_k) (\mathbf{S}_k \mathbf{V}_k^T)$
- So the query vector in reduced dimension space is given by  $\mathbf{U}_k^T \mathbf{q}$ , which is a vector of length  $k$
- Columns of  $\mathbf{S}_k \mathbf{V}_k^T$  represent the document vectors in reduced dimension space
- Variant treatment: sometimes  $\mathbf{S}_k^{-1} \mathbf{U}_k^T \mathbf{q}$  is used as the reduced query

# More with the Example

---

- D1: Infant and Toddler First Aid
- D2: Babie's and Children's Room For Your Home
- D3: Child Safety at Home
- D4: Your Babie's Health and Safety: from Infant to Toddler
- D5: Baby Proofing Basics
- D6: Your Guide to Easy Rust Proofing
- D7: Beanie Babies Collectors Guide