Latent Semantic Indexing

Man-Kwan Shan
CS, NCCU

Indexing by Latent Semantic Analysis

Scott Deerwester
Center for Information and Language Studies, University of Chicago, Chicago, IL 60637

Susan T. Dumais*, George W. Furnas, and Thomas K. Landauer
Bell Communications Research, 445 South St., Morristown, NJ 07960

Richard Harshman
University of Western Ontario, London, Ontario Canada

A new method for automatic indexing and retrieval is described. The approach is to take advantage of implicit higher-order structure in the association of terms with documents ("semantic structure") in order to improve the detection of relevant documents on the basis of terms found in queries. The particular technique used is singular-value decomposition, in which a large term by document matrix is decomposed into a set of ca. 100 orthogonal factors from which the original matrix can be approximated by linear combination. Documents are represented by ca. 100 item vectors of factor weights. Queries are represented as pseudo-document vectors formed from weighted combinations of terms, and documents with supra-threshold cosine values are returned. Initial tests find this completely automatic method for retrieval to be promising.

Introduction

We describe here a new approach to automatic indexing and retrieval. It is designed to overcome a fundamental problem that plagues existing retrieval techniques that try to match words of queries with words of documents. The problem is that users want to retrieve on the basis of conceptual content, and individual words provide unreliable evidence about the conceptual topic or meaning of a document. There are usually many ways to express a given concept, so the literal terms in a user's query may not match those of a relevant document. In addition, most words have multiple meanings, so terms in a user's query will literally match terms in documents that are not of interest to the user.

The proposed approach tries to overcome the deficiencies of term-matching retrieval by treating the unreliability of observed term-document association data as a statistical problem. We assume there is some underlying latent semantic structure in the data that is partially obscured by the randomness of word choice with respect to retrieval. We use statistical techniques to estimate this latent structure, and get rid of the obscuring "noise." A description of terms and documents based on the latent semantic structure is used for indexing and retrieval.[1]

The particular "latent semantic indexing" (LSI) analysis that we have tried uses singular-value decomposition. We take a large matrix of term-document association data and construct a "semantic" space wherein terms and documents that are closely associated are placed near one another. Singular-value decomposition allows the arrangement of the space to reflect the major associative patterns in the data, and ignore the smaller, less important influences. As a result, terms that did not actually appear in a document may still end up close to the document, if that is consistent with the major patterns of association in the data. Position in the space then serves as the new kind of semantic indexing. Retrieval proceeds by using the terms in a query to identify a point in the space, and documents in its neighborhood are returned to the user.

Deficiencies of Current Automatic Indexing and Retrieval Methods

A fundamental deficiency of current information retrieval methods is that the words searchers use often are not the same as those by which the information they seek has been indexed. There are actually two sides to the issue; we will call them loosely *synonymy* and *polysemy*. We use *synonymy* in a very general sense to describe the fact that

*To whom all correspondence should be addressed.

Received August 26, 1987; revised April 4, 1988; accepted April 5, 1988.

[1]By "semantic structure" we mean here only the correlation structure in the way in which individual words appear in documents; "semantic" implies only the fact that terms in a document may be taken as referents to the document itself or to its topic.

Indexing by Latent Semantic Analysis,
Journal of American Society for Information Science, Vol. 41, No. 6, Sep.,1990.

學術搜尋 任何時間 至少包含摘要 建立電子郵件快訊 共約有32,200項查詢結果

提示：如只要搜尋中文（繁體）的結果，可使用學術搜尋偏好.指定搜尋語言。

[PDF] Indexing by latent semantic analysis — psu.edu 提供的 [PDF] / Full text@NCCU (政大)
S Deerwester, ST Dumais, GW Furnas... - Journal of the American ..., 1990 - Citeseer
A new method for automatic indexing and retrieval is described. The approach is to take advantage of implicit higher-order structure in the association of terms with documents ("semantic structure") in order to improve the detection of relevant documents on the basis of terms ...
被引用 5927 次 - 相關文章 - HTML 版 - 全部共 160 個版本

Probabilistic latent semantic indexing — psu.edu 提供的 [PDF]
T Hofmann - Proceedings of the 22nd annual international ACM ..., 1999 - portal.acm.org
Probabilistic Latent Semantic Indexing is a novel approach to automated document indexing which is based on a sta- tistical latent class model for factor analysis of count data. Fitted from a training corpus of text documents by a gen- eralization of the Expectation Maximization ...
被引用 1536 次 - 相關文章 - Find it@NCCU (政大) - 全部共 47 個版本

Latent semantic indexing: A probabilistic analysis — psu.edu 提供的 [PDF]
CH Papadimitriou, H Tamaki... - Proceedings of the ..., 1998 - portal.acm.org
Latent semantic indexing (LX) is an information re- trieval technique based on the spectral analysis of the term-document matrix, whose empirical success had hereto- fort been without rigorous prediction and explanation. We prove that, under certain conditions, LSI does suc- ceed ...
被引用 398 次 - 相關文章 - Find it@NCCU (政大) - 全部共 27 個版本

Recovering documentation-to-source-code traceability links using latent semantic indexing — psu.edu 提供的 [PDF]
A Marcus... - ··· of the 25th International Conference on ..., 2003 - portal.acm.org
Abstract An information retrieval technique, latent semantic indexing, is used to automatically

Google 學術搜尋 C. L. Liu 搜尋 進階學術搜尋

搜尋所有網站 搜尋所有中文網頁 搜尋繁體中文網頁

學術搜尋 任何時間 至少包含摘要 建立電子郵件快訊 共約有2,030,000項查詢結果

提示：如只要搜尋中文（繁體）的結果，可使用學術搜尋偏好.指定搜尋語言。

Scheduling algorithms for multiprogramming in a hard-real-time environment — psu.edu 提供的 [PDF] Full text@NCCU (政大)

CL Liu... - Journal of the ACM (JACM), 1973 - portal.acm.org

ABSTR.~kCT. The problem of multiprogram scheduling on a single processor is studied from the viewpoint of the characteristics peculiar to the program functions that need guaranteed ser- vice. It is shown that an optimum fixed priority scheduler possesses an upper bound ...

被引用 7221 次 - 相關文章 - 全部共 109 個版本

Randomized controlled trial of transarterial lipiodol chemoembolization for unresectable hepatocellular carcinoma — ucsf.edu 提供的 [PDF]

CM Lo, H Ngan, WK Tso, CL Liu, CM Lam... - ···, 2002 - Wiley Online Library

This randomized, controlled trial assessed the efficacy of transarterial Lipiodol (Lipiodol Ultrafluide, Laboratoire Guerbet, Aulnay-Sous-Bois, France) chemoembolization in pa- tients with unresectable hepatocellular carcinoma. From March 1996 to October 1997, 80 out of ...

被引用 809 次 - 相關文章 - Find it@NCCU (政大) - 全部共 12 個版本

Adult-to-adult living donor liver transplantation using extended right lobe grafts. — nih.gov 提供的 [PDF]

···, ST Fan, CL Liu, WI Wei, RJ Lo, CL Lai... - Annals of ..., 1997 - ncbi.nlm.nih.gov

Chung-Mau Lo, MB, BS,* Sheung-Tat Fan, MS,* Chi-Leung Liu, MB, BS,* William 1. Wei, MS,* Ronald J. W. Lo, MB, BS,t Ching-Lung Lai, MD,: John KF Chan, MB, BS,§ Irene 0. L. Ng, MD,ll Amy Fung, Ph.D.,¶ and John Wong, Ph.D.* ... From the Departments of ...

被引用 438 次 - 相關文章 - 全部共 11 個版本

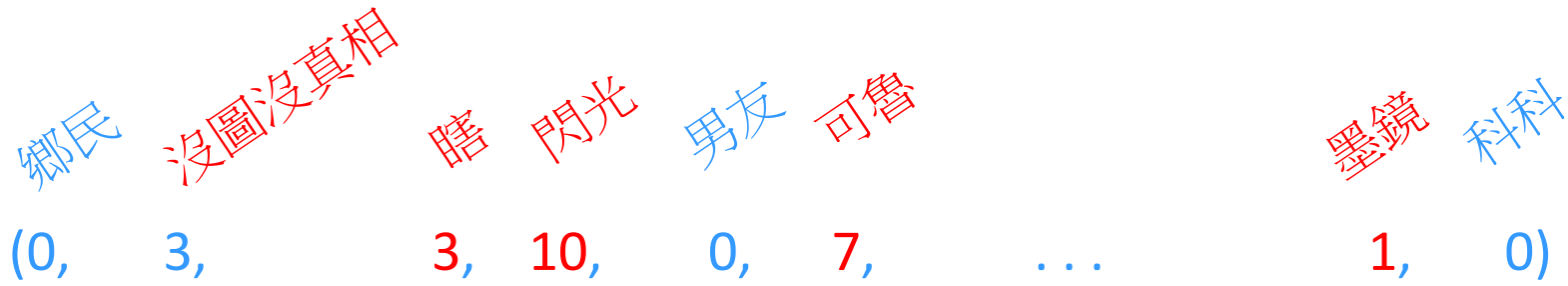
A new algorithm for floorplan design — psu.edu 提供的 [PDF]

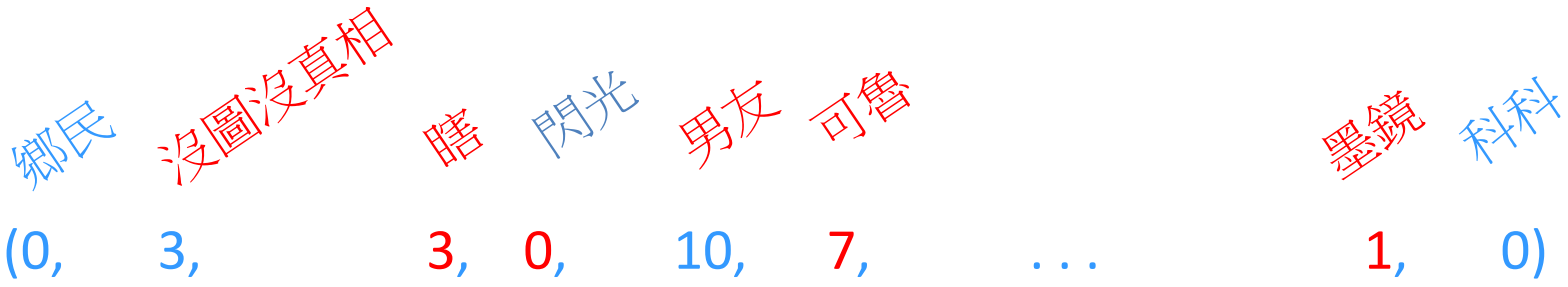
···, CL Liu - Proceedings of the 23rd ACM/IEEE Design ..., 1986 - portal.acm.org


Google 學術搜尋
The anatomy of a large-scale hypertextual Web
搜尋
進階學術搜尋
搜尋所有網站 搜尋所有中文網頁 搜尋繁體中文網頁
學術搜尋 任何時間 至少包含摘要 建立電子郵件快訊
共約有8,270項查詢結果
提示： 如只要搜尋中文（繁體）的結果，可使用學術搜尋偏好指定搜尋語言。
The anatomy of a large-scale hypertextual Web search engine* 1
S Brin... - Computer networks and ISDN systems, 1998 - Elsevier
To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of Web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on ...
被引用 7642 次 - 相關文章 - 全部共 458 個版本
psu.edu 提供的 [PDF]
Full text@NCCU (政大)
The Anatomy of a Large-ScaleHypertextual Web Search Engine
B Sergey, P Lawrence - 2006 - citeulike.org
... Tags. The Anatomy of a Large-Scale Hypertextual Web Search Engine. ... View FullText article. No URLs defined. Abstract. Abstract In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. ...
被引用 192 次 - 相關文章 - 頁庫存檔 - 全部共 2 個版本
The Anatomy of a Large-Scale Hyper Textual Web Search Engine
U Sehgal, K Kaur... - Computer and Electrical ..., 2009 - ieeexplore.ieee.org
Abstract—In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing ...
被引用 1 次 - 相關文章 - Find it@NCCU (政大) - 全部共 3 個版本
The PageRank Citation Ranking: Bringing Order to the Web.
L Page, S Brin, R Motwani... - 1999 - ilpubs.stanford.edu
... Spertus Spe97 discusses information that can be obtained from the link structure for a variety
stanford.edu 提供的 [PDF]

Deficiency of Vector Space Model

- Vector space model
  - Each document is represented as a length-m vector





Deficiency of Vector Space Model (cont.)

- Synonymy
  - More than one way to refer to the same object
  - e.g. human, user, people
  - People choose the same key word for a single well-known object less than 20% of the time
  - poor recall
- Polysemy

SVD

  - One word have more than one distinct meaning
  - e.g. Chip, Jordan
  - Poor precision

Poor Recall for Synonymy

- **Synonymy** occurs when multiple words or phrases refer to the same object or concept (e.g., “human,” “user,” “people”).
- In the **Vector Space Model**, queries typically rely on exact matches of terms between the query and documents.
- **Impact on Recall:** If a query uses one synonym (e.g., “human”), but a relevant document uses a different synonym (e.g., “people”), the system may fail to retrieve that document, resulting in **poor recall**. Recall measures the proportion of relevant documents retrieved out of all relevant documents in the collection, and synonymy limits the retrieval of some relevant documents.

Poor Precision for Polysemy

- **Polysemy** refers to a single word having multiple meanings depending on context (e.g., “chip” could mean a computer chip or a potato chip, and “Jordan” could mean the country or a person’s name).
- In the **Vector Space Model**, matching is based solely on word occurrences without understanding context.
- **Impact on Precision:** A query for “chip” intending to find information about computer chips might retrieve irrelevant documents about potato chips, reducing **precision**. Precision measures the proportion of retrieved documents that are actually relevant to the query, and polysemy introduces irrelevant results due to ambiguity.

Summary

- **Synonymy → Poor Recall:** Relevant documents using different synonyms might not be retrieved.
- **Polysemy → Poor Precision:** Irrelevant documents are retrieved due to word ambiguity.

These issues highlight the limitations of basic Vector Space Models and the need for more advanced techniques like **Latent Semantic Analysis (LSA)** or **contextual embeddings** to handle synonymy and polysemy effectively.

SVD

	d1	d2	d3	d4
a	2	2	0	0
b	2	2	0	0
c	3	3	0	0
d	0	0	2	2
e	0	0	1	1
f	0	0	2	2

=

	f1	f2	f3	f4
a	-0.48	0	-0.48	0.48
b	-0.48	0	0.84	0.15
c	-0.72	0	-0.23	-0.42
d	0	-0.66	0	0.55
e	0	-0.33	0	-0.22
f	0	-0.66	0	-0.44

left singular

X

	f1	f2	f3	f4
f1	5.83	0	0	0
f2	0	4.24	0	0
f3	0	0	0	0
f4	0	0	0	0

diagonal

X

	d1	d2	d3	d4
f1	-0.7	-0.7	0	0
f2	0	0	-0.7	-0.7
f3	-0.7	0.7	0	0
f4	0	0	-0.7	07

right singular

Example (1/3)

Technical Memo Titles

c1: *Human* machine *interface* for 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, ordered *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*

Query: human, computer interaction ?

Example (2/3)

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Query: human, computer interaction ?

Example (3/3)

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

corcoeff(human,user) = -0.38 , corcoeff(human,minors) = -0.29

Rationale of Latent Semantic Indexing

	Access	Document	Retrieval	Information	Theory	Database	Indexing	Computer	REL	MATCH
Doc 1	x	x	x			x	x		R	
Doc 2				x*	x			x*		M
Doc 3			x	x*				x*	R	M

Query: IDF computer-based information look-up

- What if 95% documents containing “access” also contain “retrieval”

Rationale of Latent Semantic Indexing

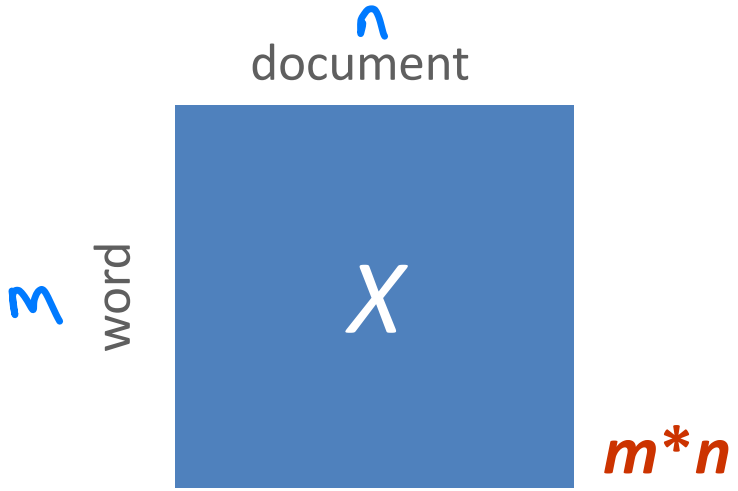
- Correlation between the occurrence of one term and another
- Occurrence of some patterns of words gives a clue as to the likely (unlikely) occurrence of others
- Up-weight synonymy terms,

  down-weight polysemy terms

Preliminary

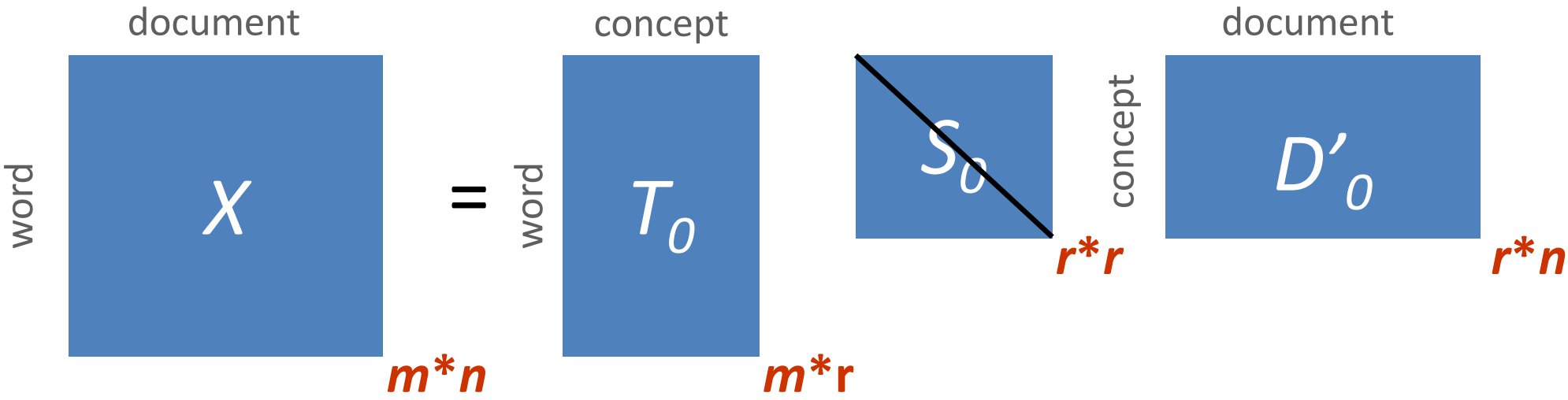
- Given ***n*** documents, word size ***m***
  - Documents
    - A sentence, paragraph, chapter
  - Words
    - Remove stop-words
- Generate a word-documents co-occurrence matrix ***X***

Column: number of w_i occurs in d_j
Row: number of words present in d_j



Singular Value Decomposition

- Consider a term-document co-occurrence data
  - ***m*** words, ***n*** documents



T_0: left singular, orthonormal matrix, $T_0 \times T'_0 = I$
D_0: right singular, orthonormal matrix, $D'_0 \times D_0 = I$
S_0: diagonal matrix, r: rank of X

Example

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

12x9

Example

$T_0 =$

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

12 x 9

$S_0 =$

3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

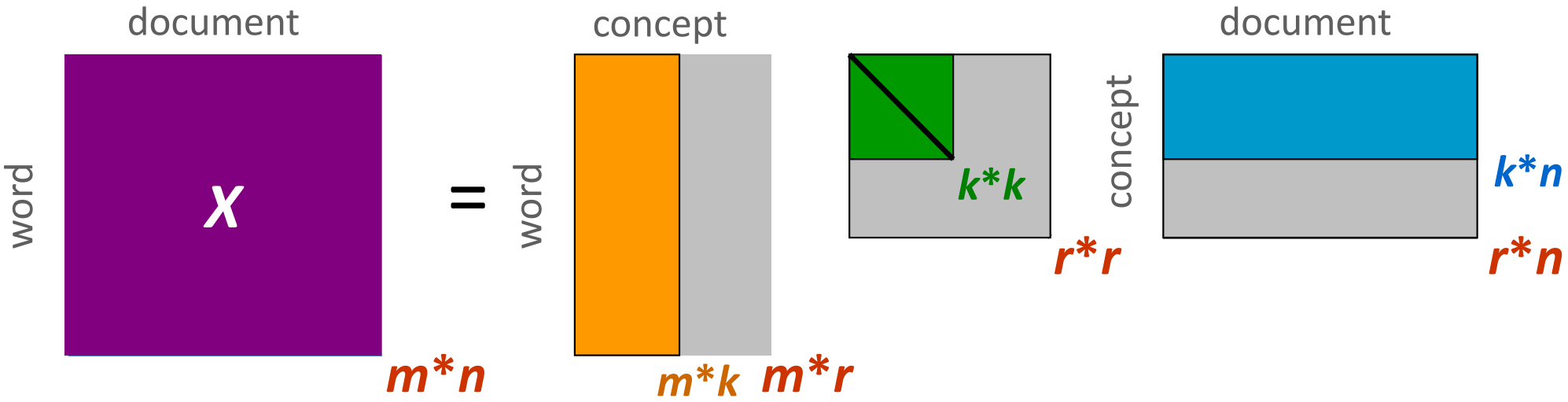
9x9

$D_0 =$

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

9x9

SVD (cont.)



- **Reduce** to a ***k***-dimensional space
  - Each singular vector is regarded as a latent concept
  1. Capture salient words combined with documents
  2. Singular value = importance of the concept
- $X = T_0 S_0 D'_0 \approx \overline{X} = TSD'$

Example (3/4)

$T_0 =$

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

$S_0 =$

3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

$D_0 =$

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

Example

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

Example

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Example (4/4)

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

corcoeff(human,user) = 0.94 , corcoeff(human,minors) = -0.83

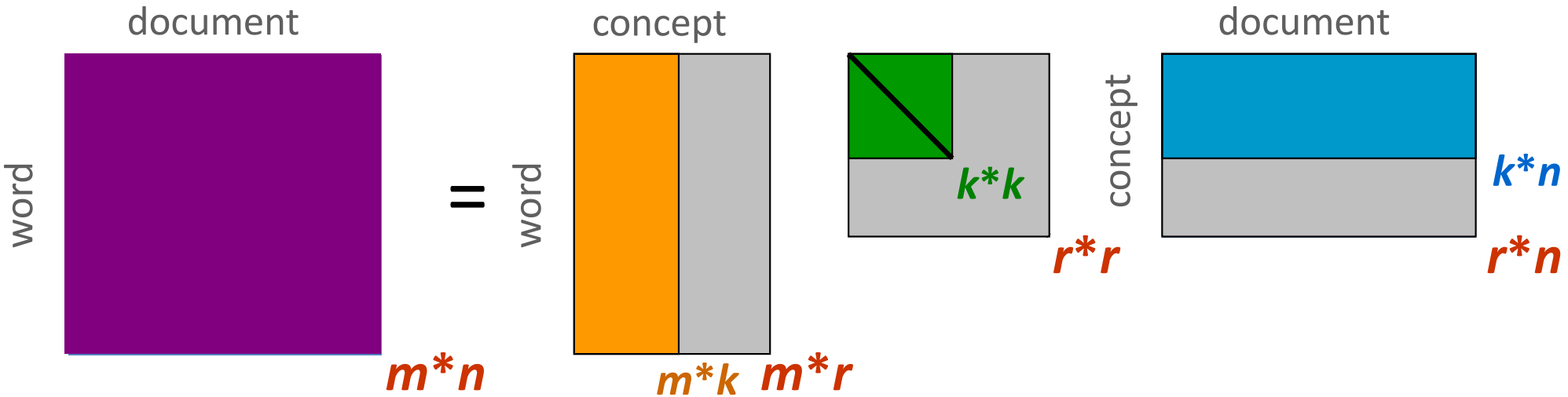
Comparing Two Terms

dimension reduction

X' : X^T

S' = S (diagonal)

- $X = T_0 S_0 D_0' \approx \overline{X} = TSD'$
- $\overline{X}\overline{X}' = (TSD')(TSD')' = (TSD')(DS'T') = (TS)(S'T') = TS^2T'$
- *i*, *j* cell of $\overline{X}\overline{X}'$ can be obtained by taking dot product between the *i* and *j* rows of matrix *TS*
- *TS*: stretched version of *T*



Example (3/4)

$T_0 =$

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

$S_0 =$

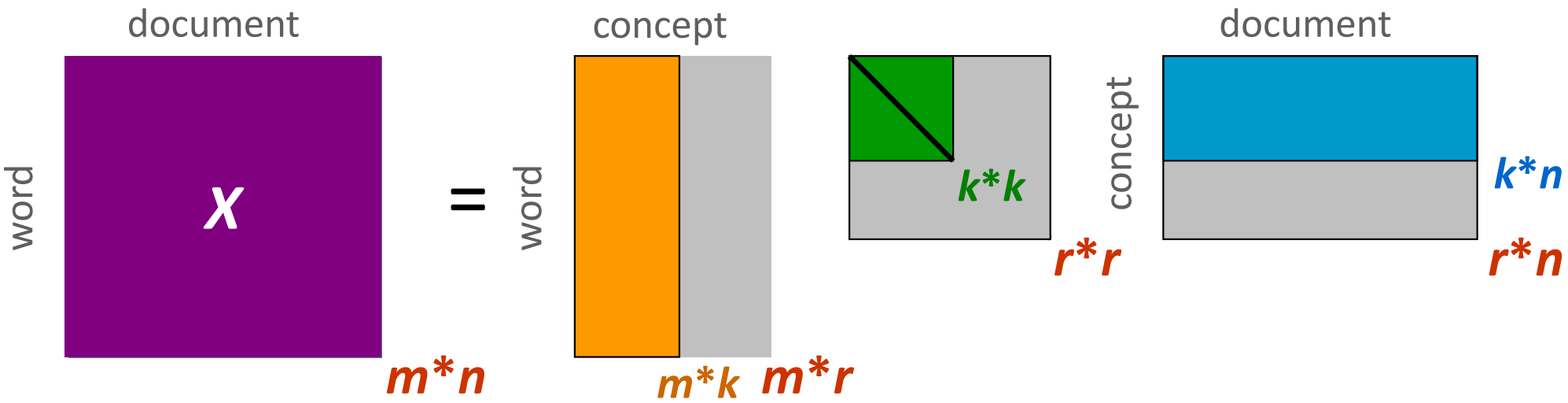
3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

$D_0 =$

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

Comparing Two Documents

- $X = T_0 S_0 D_0' \approx \overline{X} = TSD'$
- $\overline{X}'\overline{X} = (TSD')'(TSD') = (DS'T')(TSD') = (DS')(SD') = DS^2D'$
- *i*, *j* cell of $\overline{X}'\overline{X}$ can be obtained by taking dot product between the *i* and *j* rows of matrix *DS*



Example (3/4)

$T_0 =$

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

$S_0 =$

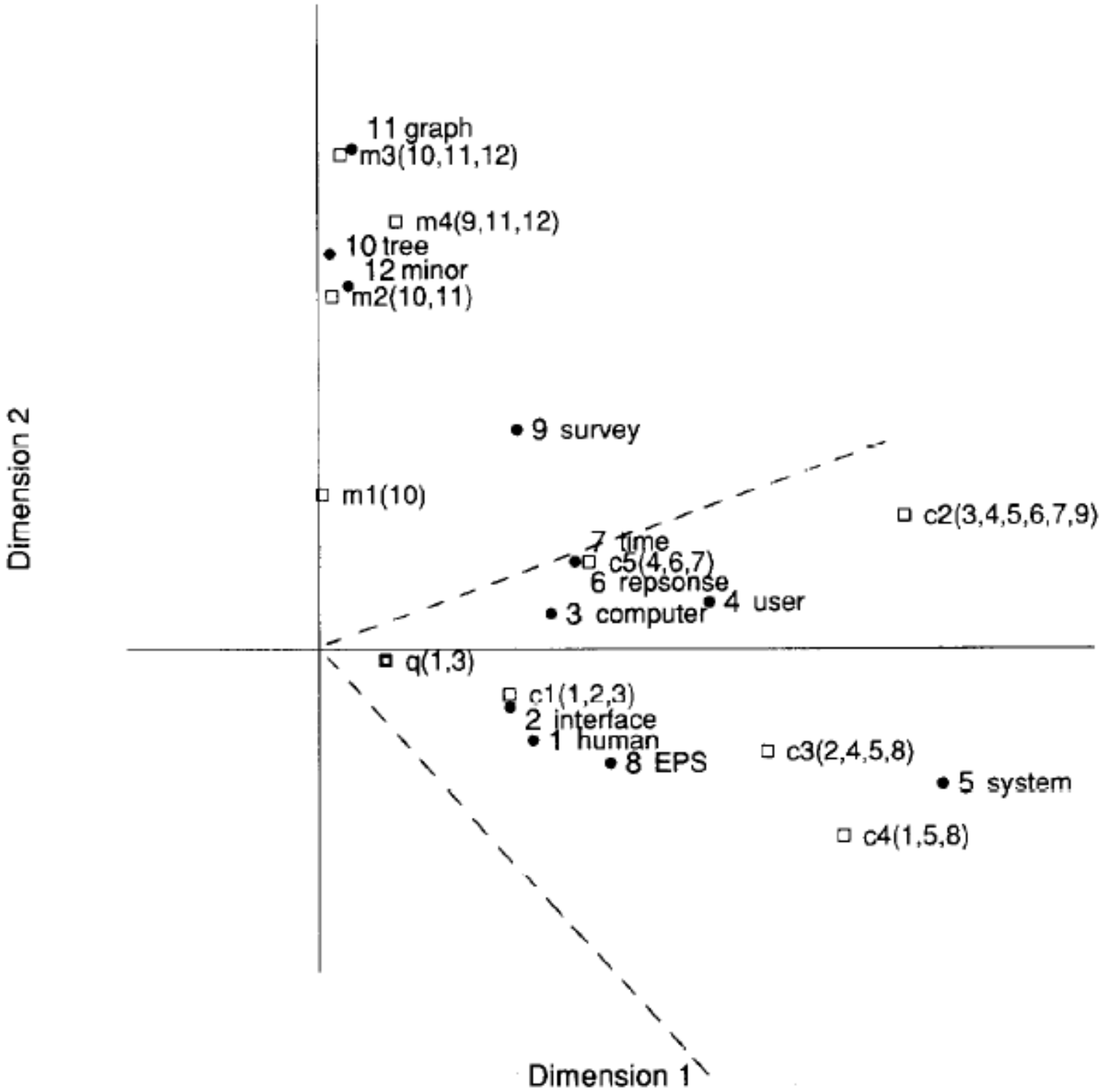
3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

$D_0 =$

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

Example

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1


11 graph
m3(10,11,12)
m4(9,11,12)
10 tree
12 minor
m2(10,11)
9 survey
m1(10)
c2(3,4,5,6,7,9)
7 time
c5(4,6,7)
6 repsonse
4 user
3 computer
q(1,3)
c1(1,2,3)
2 interface
1 human
8 EPS
c3(2,4,5,8)
5 system
c4(1,5,8)
Dimension 2
Dimension 1

Properties of SVD

- SVD method is equivalent to obtaining

$$\widehat{A}_k = \arg\min_{A_k} \|A - A_k\|$$

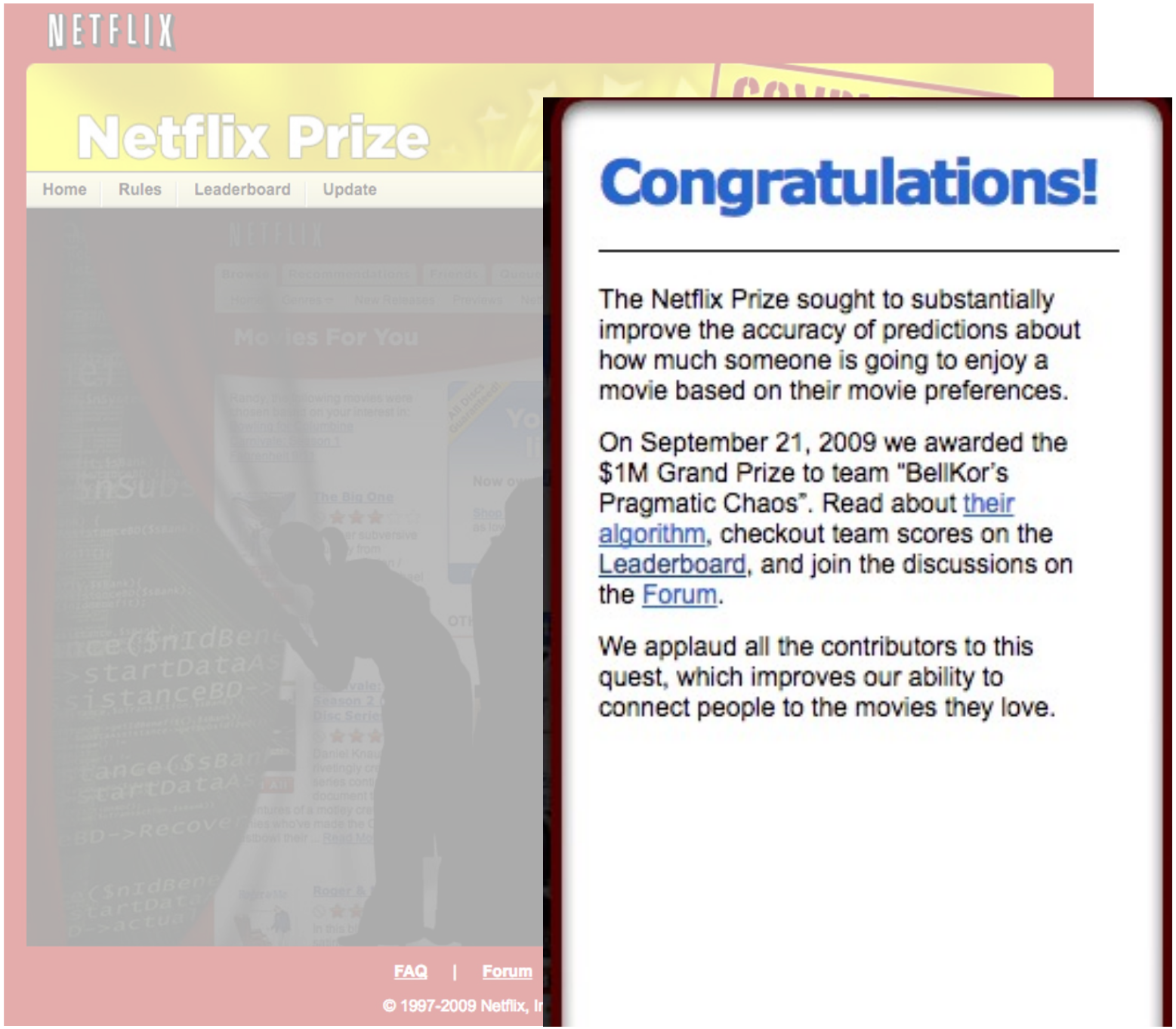
A_k is a rank k matrix

- $\widehat{A}_k$ is the rank k matrix which has the minimal distance to A among all possible rank k matrices.

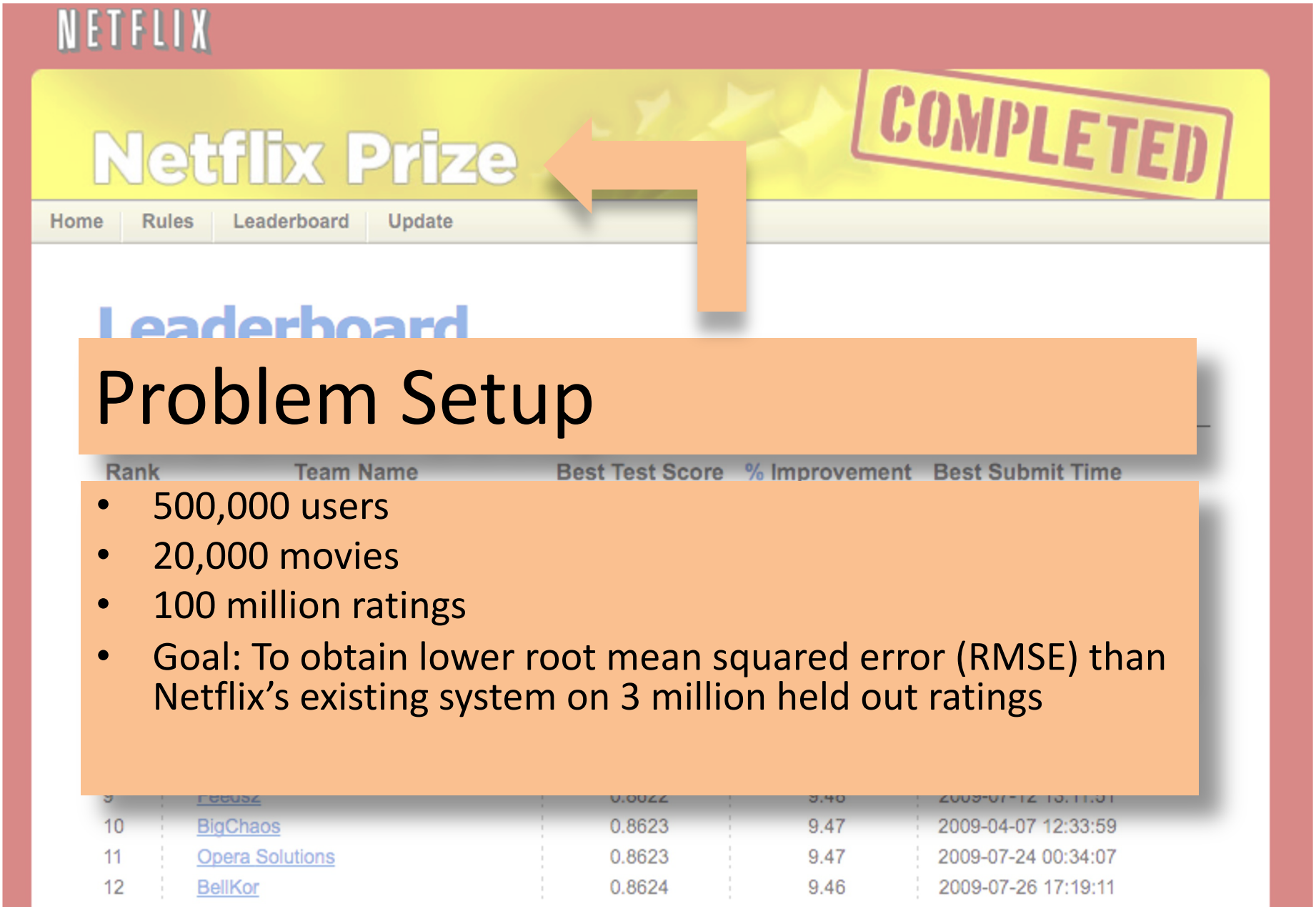
Matrix Factorization

Netflix Prize

Netflix Prize (cont.)

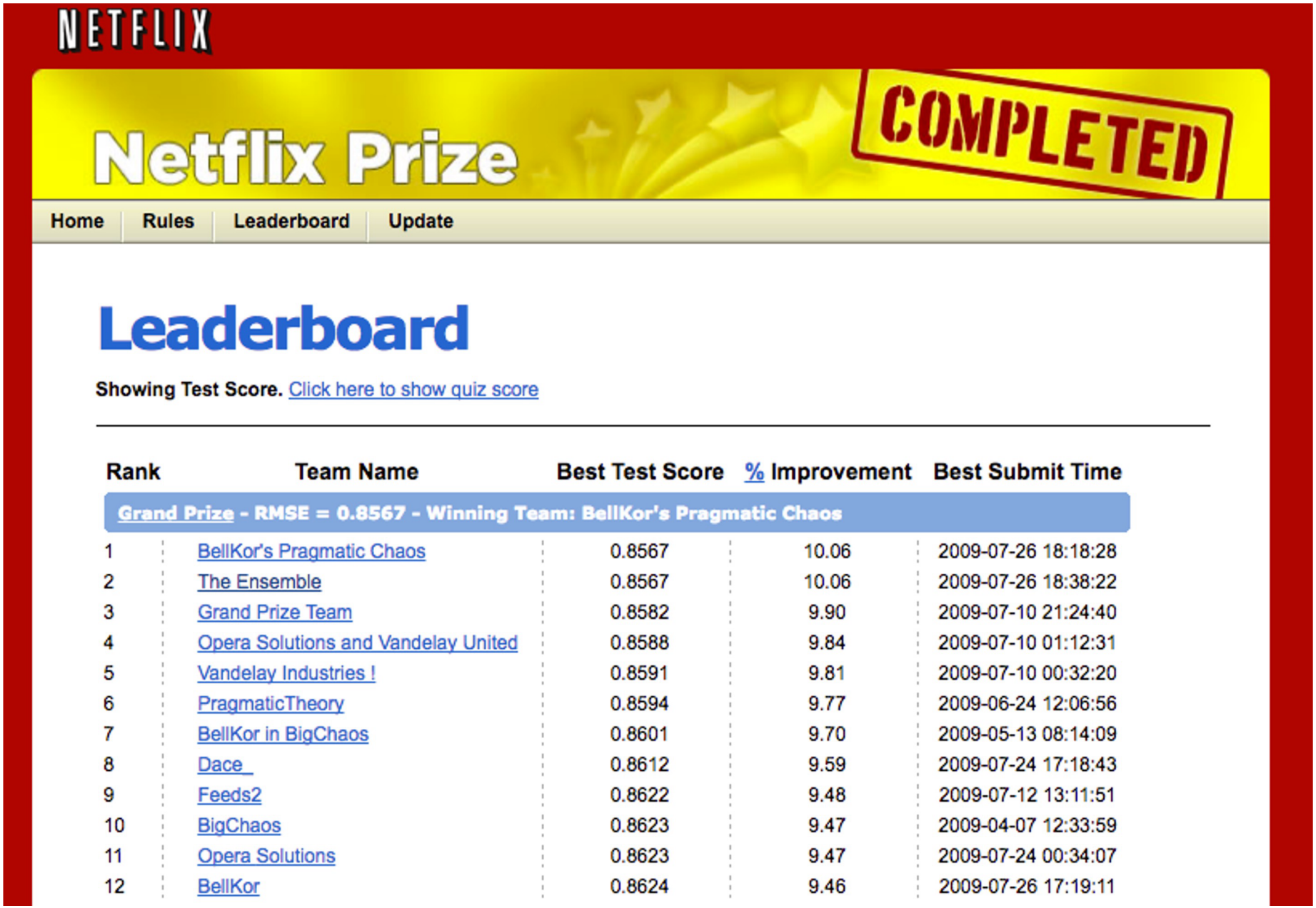


Netflix Prize (cont.)

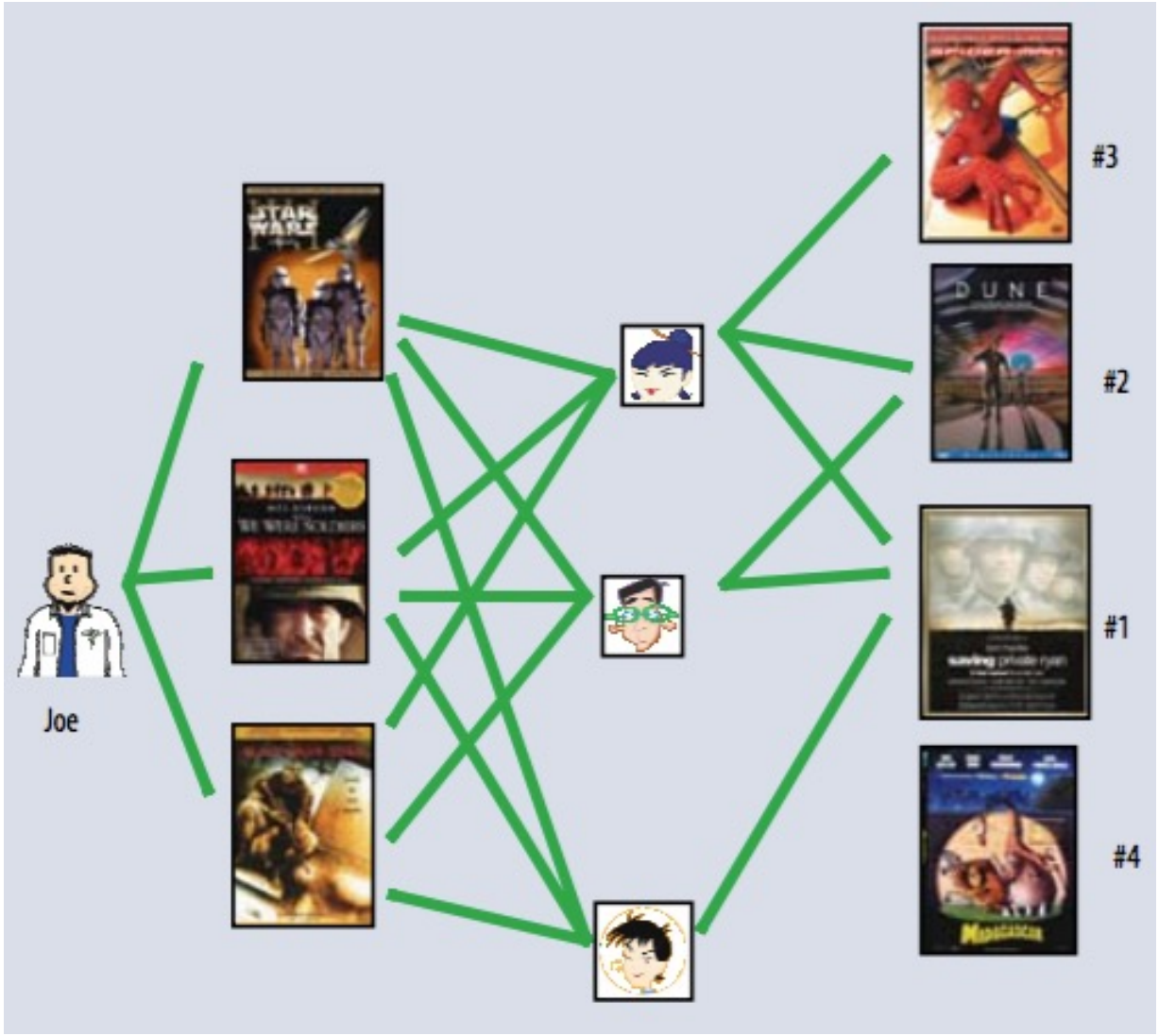


- 500,000 users
- 20,000 movies
- 100 million ratings
- Goal: To obtain lower root mean squared error (RMSE) than Netflix's existing system on 3 million held out ratings

Netflix Prize (cont.)



Collaborative Filtering: Neighborhood Methods



A green line indicates the movie was **watched**

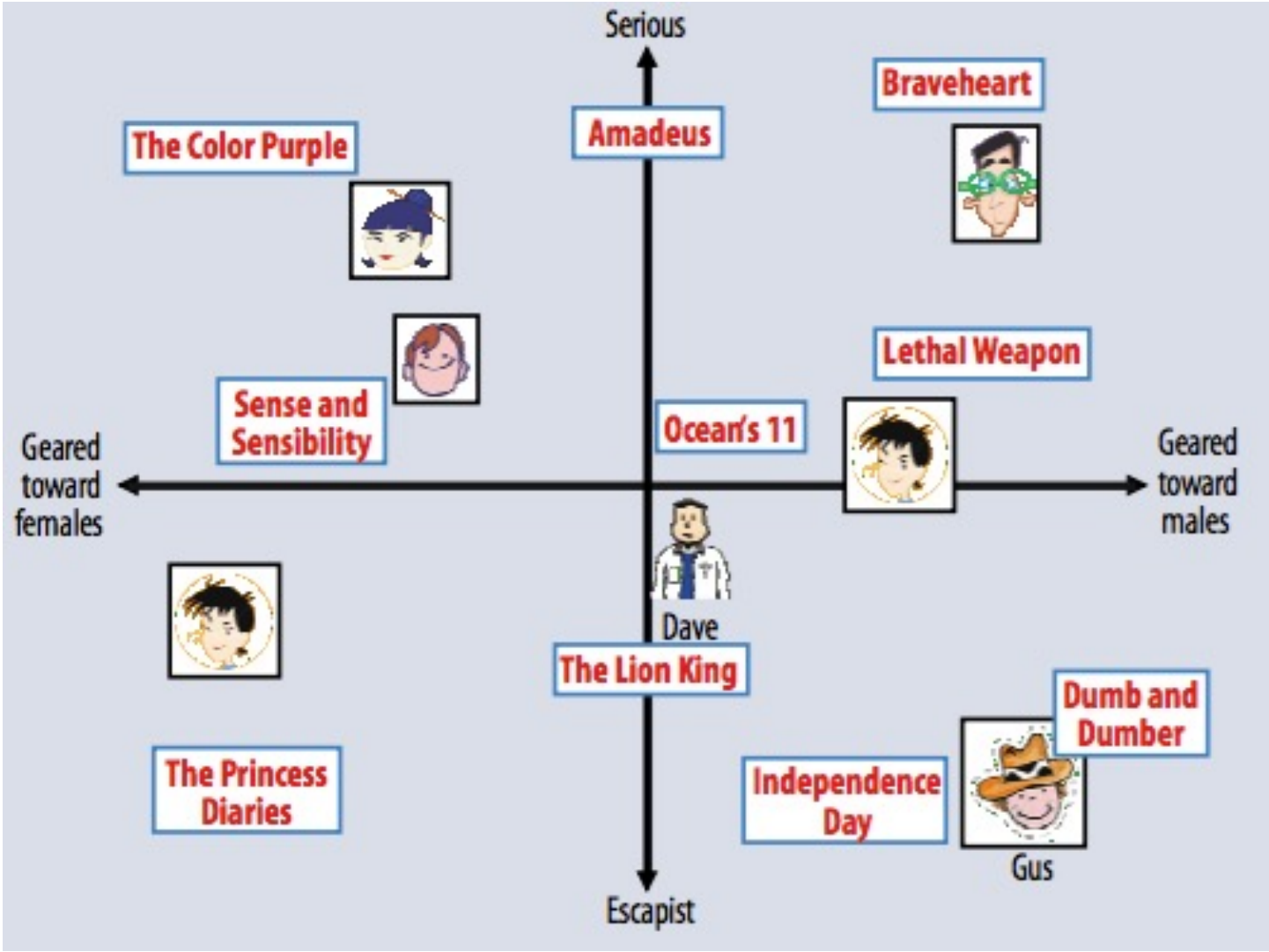
Algorithm:

1. **Find neighbors** based on similarity of movie preferences
2. **Recommend** movies that those neighbors watched

Figures from Koren et al. (2009)

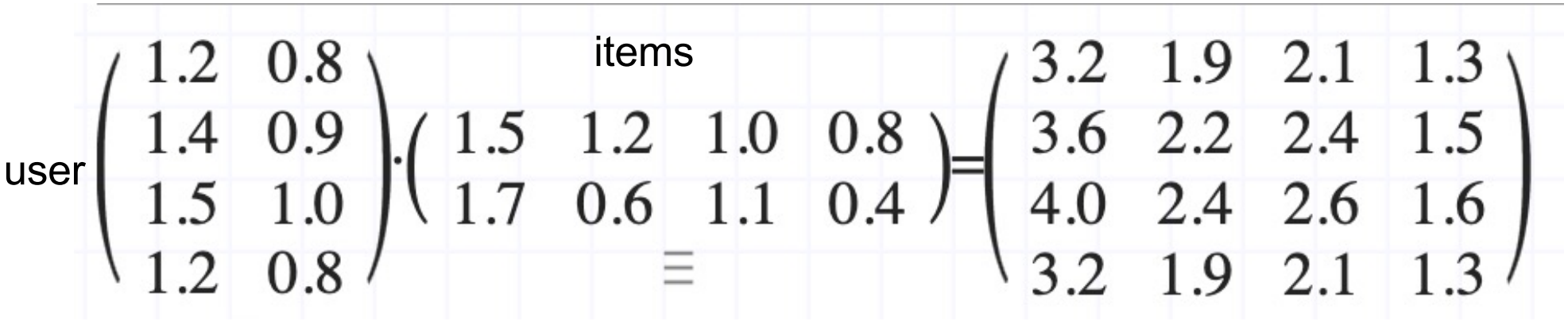
Collaborative Filtering: Latent Factor Methods

- Assume that both movies and users live in some **low-dimensional space** describing their properties
- **Recommend** a movie based on its **proximity** to the user in the latent space

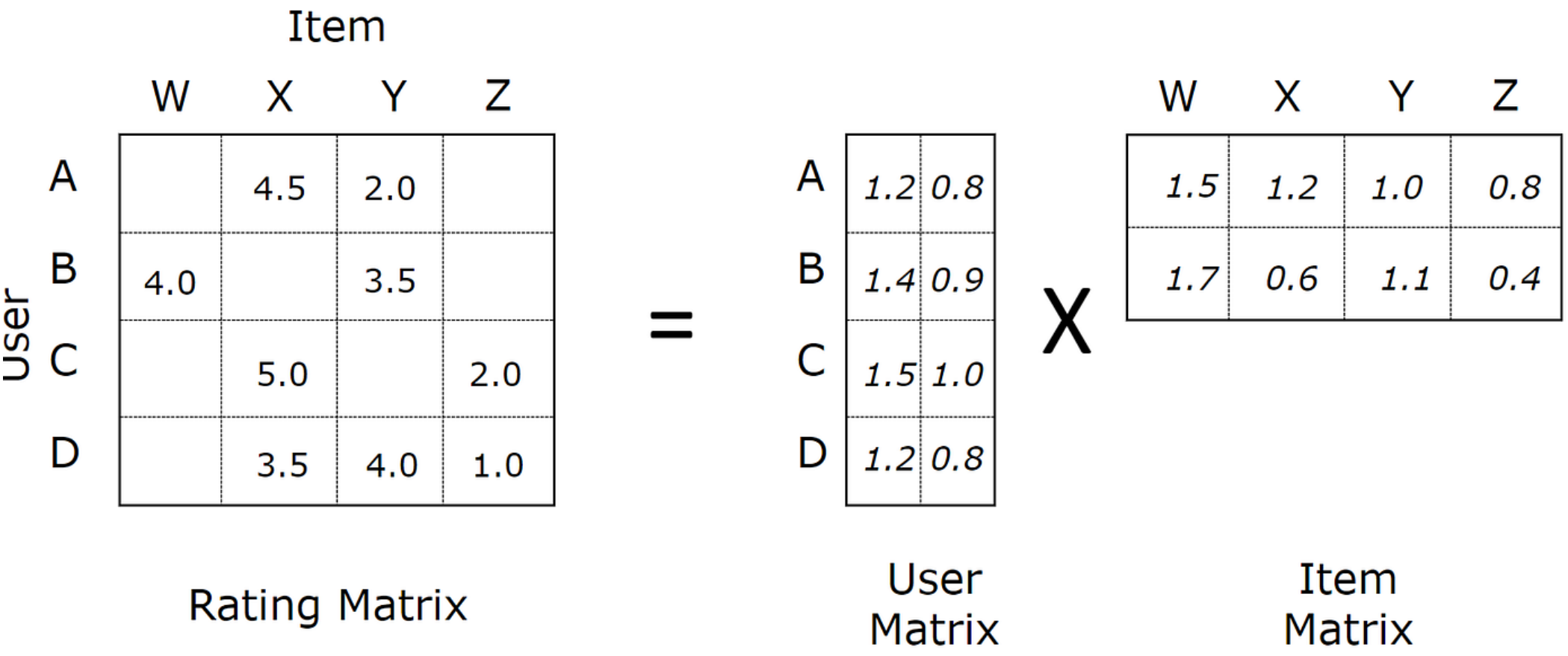


Figures from Koren et al. (2009)

Matrix Factorization



Matrix Factorization



A Basic Matrix Factorization Model

- Each item *i* → vector $q_i \in \mathbb{R}^f$
- Each user *u* → vector $p_u \in \mathbb{R}^f$
- r_{ui} : real rating of item *i* by user *u*
- $\hat{r}_{ui} = q_i^T p_u$: estimated rating

$$\min_{q^*, p^*} \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2$$

We want to learn the factor vectcp_u and q_i

Example:

item

$$\text{user} \begin{bmatrix} ? & ? & 2 \\ ? & 4 & ? \\ 3 & ? & 5 \end{bmatrix}$$

$$\arg\min_{p,q}[(2 - p_1 \cdot q_3)^2 + (4 - p_2 \cdot q_2)^2 + (3 - p_3 \cdot q_1)^2 + (2 - p_3 \cdot q_3)^2]$$

A Basic Matrix Factorization Model

- Each item *i* → vector $q_i \in \mathbb{R}^f$
- Each user *u* → vector $p_u \in \mathbb{R}^f$
- r_{ui} : real rating of item *i* by user *u*
- $\hat{r}_{ui} = q_i^T p_u$: estimated rating

$$\min_{q^*, p^*} \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(|| q_i ||^2 + || p_u ||^2)$$

We want to learn the factor vectc p_u and q_i

regularization term

Learning Algorithms

- Stochastic gradient descent
  - For each given training case, the system predicts r_{ui} and computes the associated prediction error

$$e_{ui} \overset{def}{=} r_{ui} - q_i^T p_u$$

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$

$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

- Alternating least squares (ALS)
  - If we fix one of the unknowns, the optimization problem becomes quadratic and can be solved optimally.
  - Thus, ALS techniques rotate between fixing the q_i's and fixing the p_u's.
  - When all p_u's *are fixed, the system* recomputes the q_i's by solving a least-squares problem, and vice versa.

More Improvements

- Adding biases
- Additional input sources
- Temporal dynamics
- Varying confidence levels

Adding Biases

- Much of the observed variation in rating values is due to effects associated with either users or items, known as ***biases*** or ***intercepts***.

$$b_{ui} = \mu + b_i + b_u$$

μ : overall average rating

b_i and b_u :observed deviations of user *u* and item *i*, respectively.

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

Adding Biases

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in\kappa} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda$$

$$(|| p_u ||^2 + || q_i ||^2 + b_u^2 + b_i^2)$$

Additional Input Sources

Boolean implicit feedback

$N(u)$: A set of items for which user *u* expressed an implicit prefer

$$\sum_{i \in N(u)} x_i \quad \xrightarrow{\text{normalize}} \quad |N(u)|^{-0.5} \sum_{i \in N(u)} x_i$$

Boolean user attributes

$A(u)$: A set of attributes where user *u* corresponds to

$$\sum_{a \in A(u)} y_a$$

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T [p_u + |N(u)|^{-0.5} \sum_{i \in N(u)} x_i + \sum_{a \in A(u)} y_a]$$

Temporal Dynamics

- The system should account for the temporal effects reflecting the dynamic, time-drifting nature of user-item interactions.

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

$$\hat{r}_{ui}(t) = \mu + b_i(t) + b_u(t) + q_i^T p_u(t)$$

Varying confidence Levels

- In several setups, not all observed ratings deserve the same weight or confidence.

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in\kappa} c_{ui}(r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda\,(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2)$$

Netflix Prize Competition

- In 2006, the online DVD rental company Netflix announced a contest to improve the state of its recommender system
- *Training set* include more than 100 million ratings spanning about 500,000 anonymous customers and their ratings on more than 17,000 movies.
- Participating teams submit predicted ratings for a test set of approximately 3 million ratings, and Netflix calculates a root-mean-square error (RMSW) based on the held-out truth.

Netflix Prize Competition

- The first team that can improve on the Netflix algorithm's RMSE performance by 10 percent or more wins a $1 million prize.
- If no team reaches the 10 percent goal, Netflix gives a$50,000 *Progress Prize* to the team in first place after each year of the competition.
- More than 48,000 teams from 182 different countries.

Evaluation

- *BellKor* won the 2007 Progress Prize with the best score at the time: 8.43 better than Netflix.
- *BellKor* aligned with team *Big Chaos* to win the 2008 Progress Prize with a score of 9.46.
- They were still in first place at the time of this writing.

Karhunen-Loève Transform & Principle Component Analysis

- Principal Components Analysis (PCA)
  - can be used to simplify a dataset;
  - a linear transformation

    that chooses a new coordinate system for the data set such that

    the greatest variance by any projection of the data set comes to lie on the first axis

    the second greatest variance on the second axis

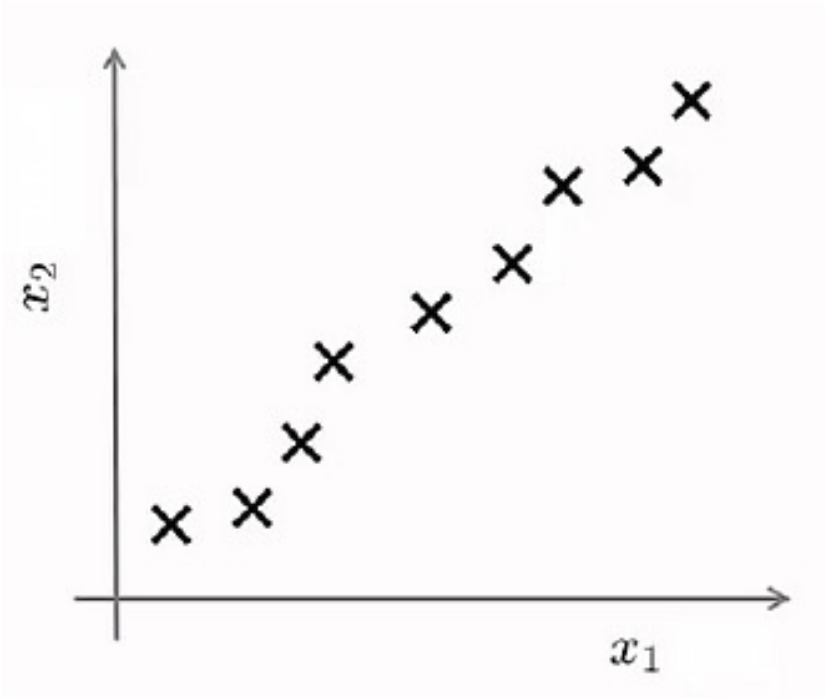
    and so on
  - can be used for reducing dimensionality in a dataset while retaining those characteristics of the dataset

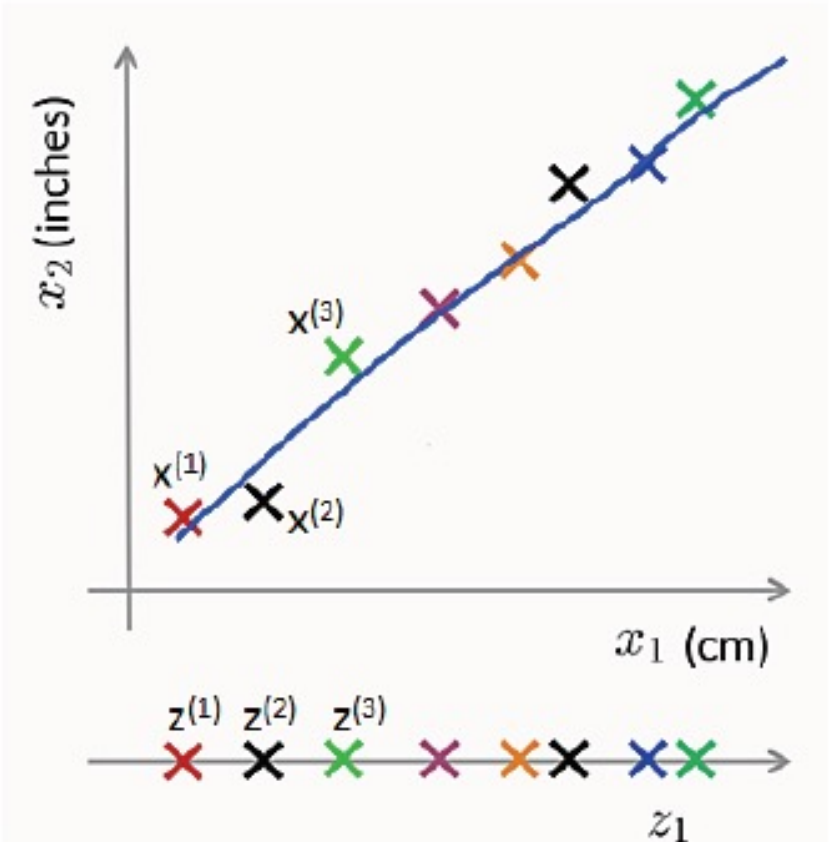
    that contribute most to its variance

    by eliminating the later principal components

Principle Component Analysis

Principle Components





Linear Transformation

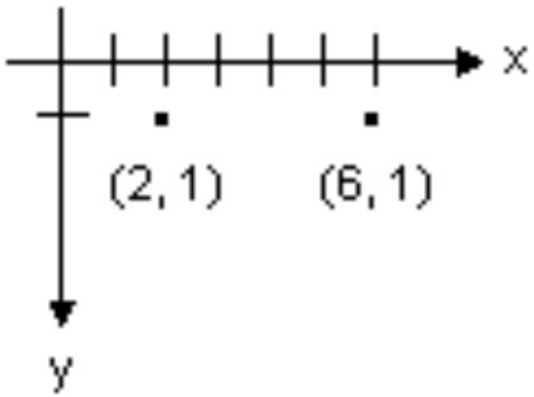
- A linear transformation of a vector x can be described by a square matrix A.
- Ax changes both the magnitude and the direction of x.

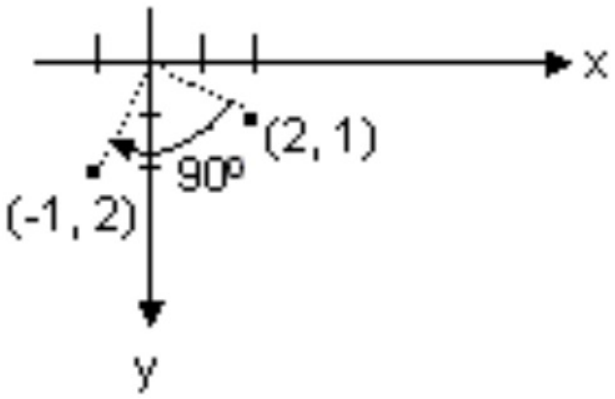
Change magnitude

$$A = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}, \quad Ax = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 1 \end{bmatrix}$$

Change direction

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad Ax = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

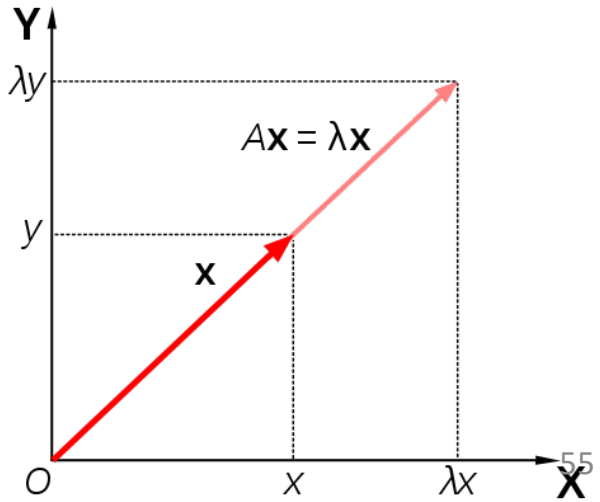




Linear Transformation (cont.)

- A linear transformation of a vector *x* can be described by a square matrix *A*.
- *Ax* changes both the magnitude and the direction of *x*.
- Special case: *Ax* changes
  - only the scale (magnitude) of the vector *x* & leaves the direction unchanged, or
  - switches the vector *x* to the opposite direction,

  then *x* is an *eigenvector* of *A*



Eigenvector & Eigenvalue

- Eigenvector & eigenvalue
  - Let A be an *n* * *n* matrix,

    if $\exists$ a nonzero vector *x* such that $Ax=\lambda x$,

    scalar λ: eigenvalue of *A*

    vector *x*: eigenvector of *A*.
  - e.g. $$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix} = 4 \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

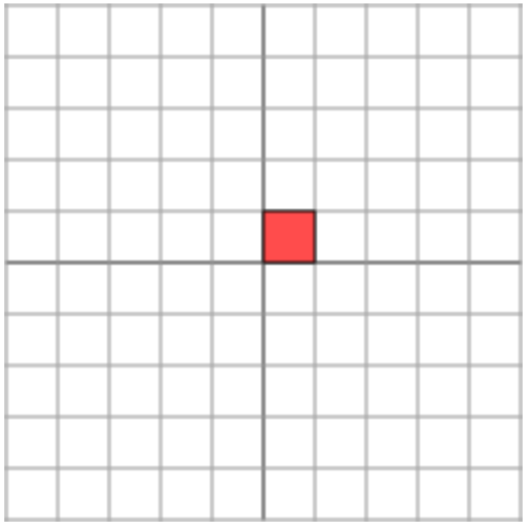
Eigenvector & Eigenvalue (cont.)

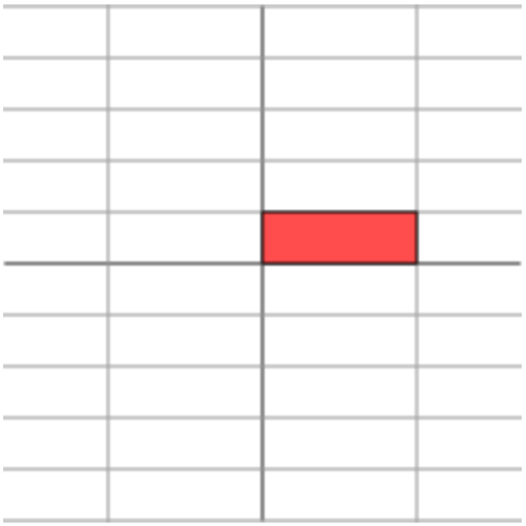
- *Ax* changes only the magnitude of *x*.

$$A = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Ax = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3x \\ y \end{bmatrix}$$

$$Ax = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

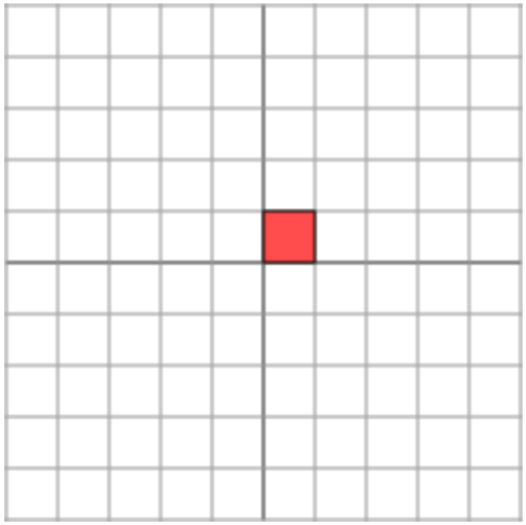


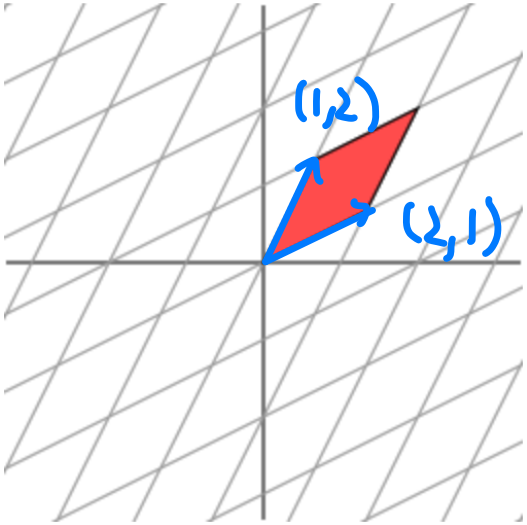


Eigenvector & Eigenvalue (cont.)

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$Ax = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x + y \\ x + 2y \end{bmatrix}$$

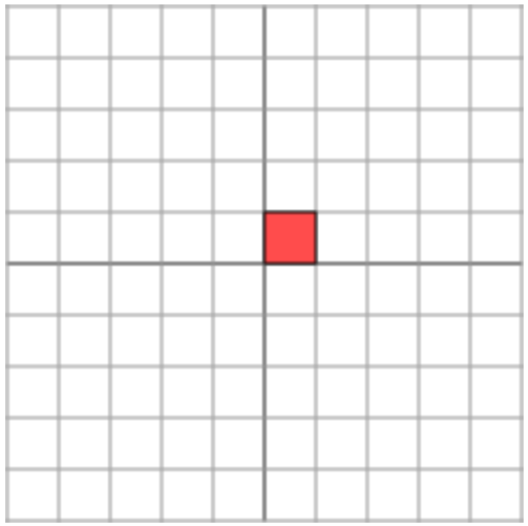


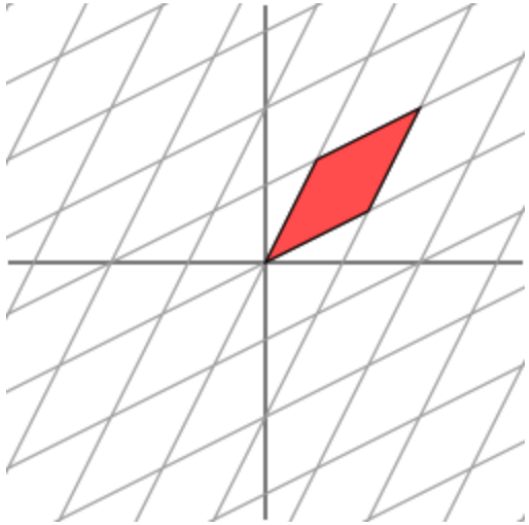


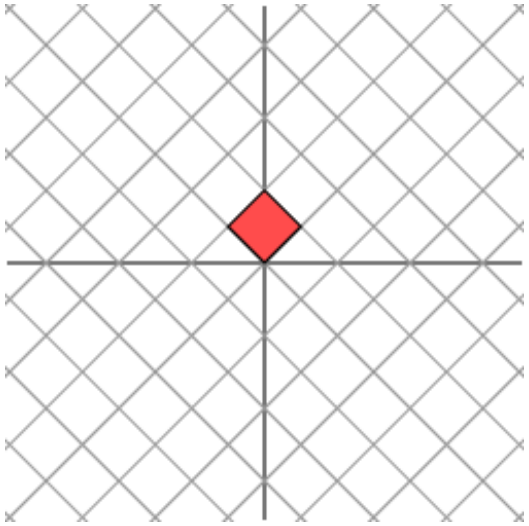
Eigenvector & Eigenvalue (cont.)

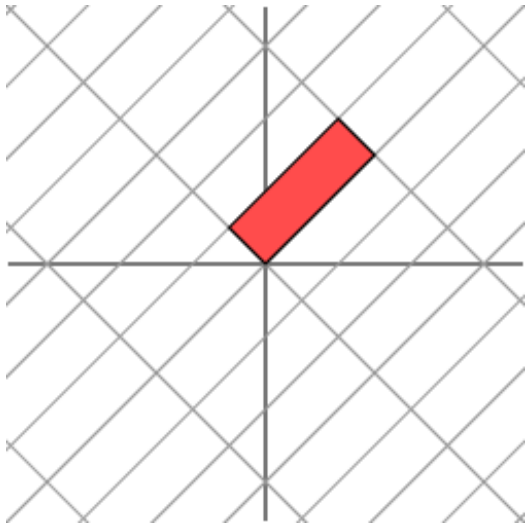
$$Ax = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x + y \\ x + 2y \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$Ax = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x + y \\ x + 2y \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = 1 \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$







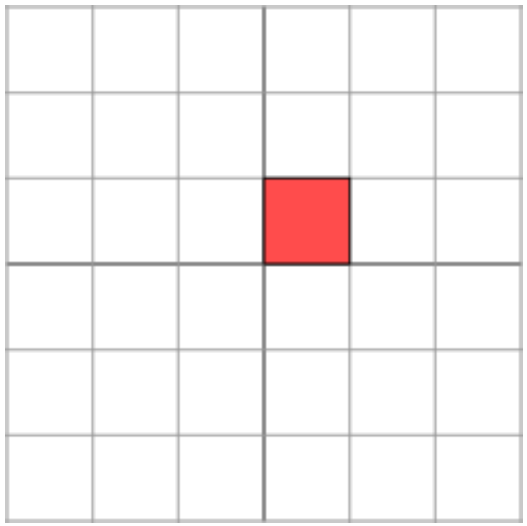


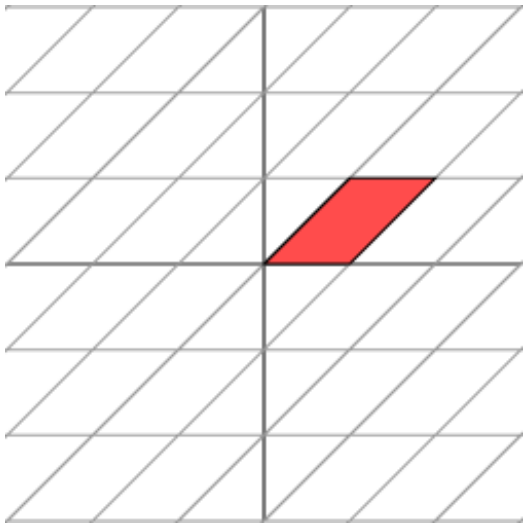
Eigenvector & Eigenvalue (cont.)

- A linear transformation of a vector *x* can be described by a square matrix *A*.
- *Ax* changes both the magnitude and the direction of *x*.

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

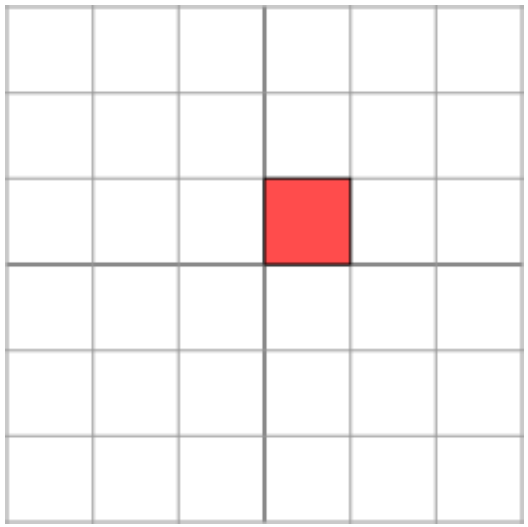
$$Ax = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + y \\ y \end{bmatrix}$$

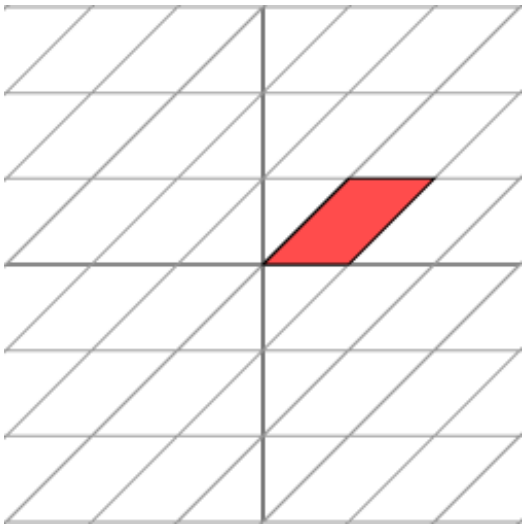


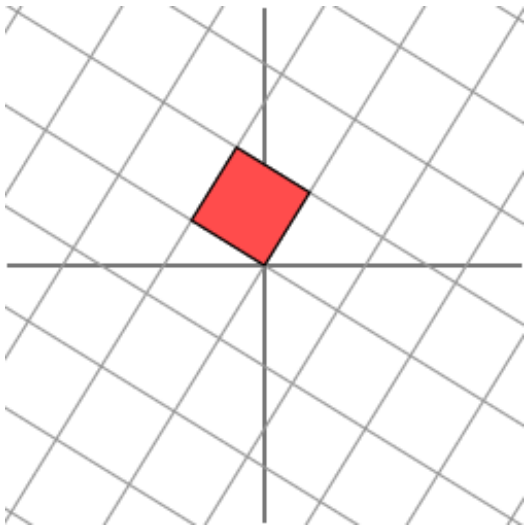


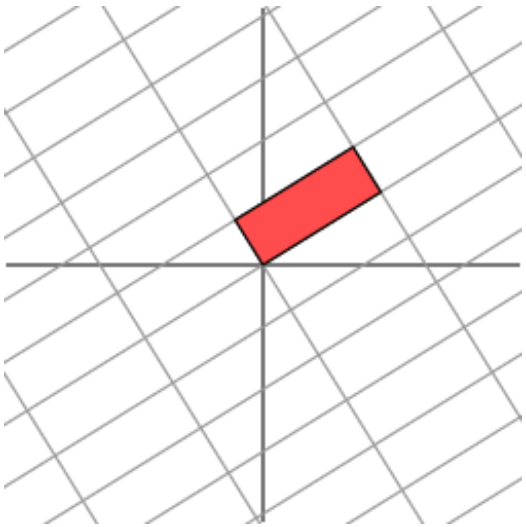
Eigenvector & Eigenvalue (cont.)

$$Ax = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + y \\ y \end{bmatrix}$$



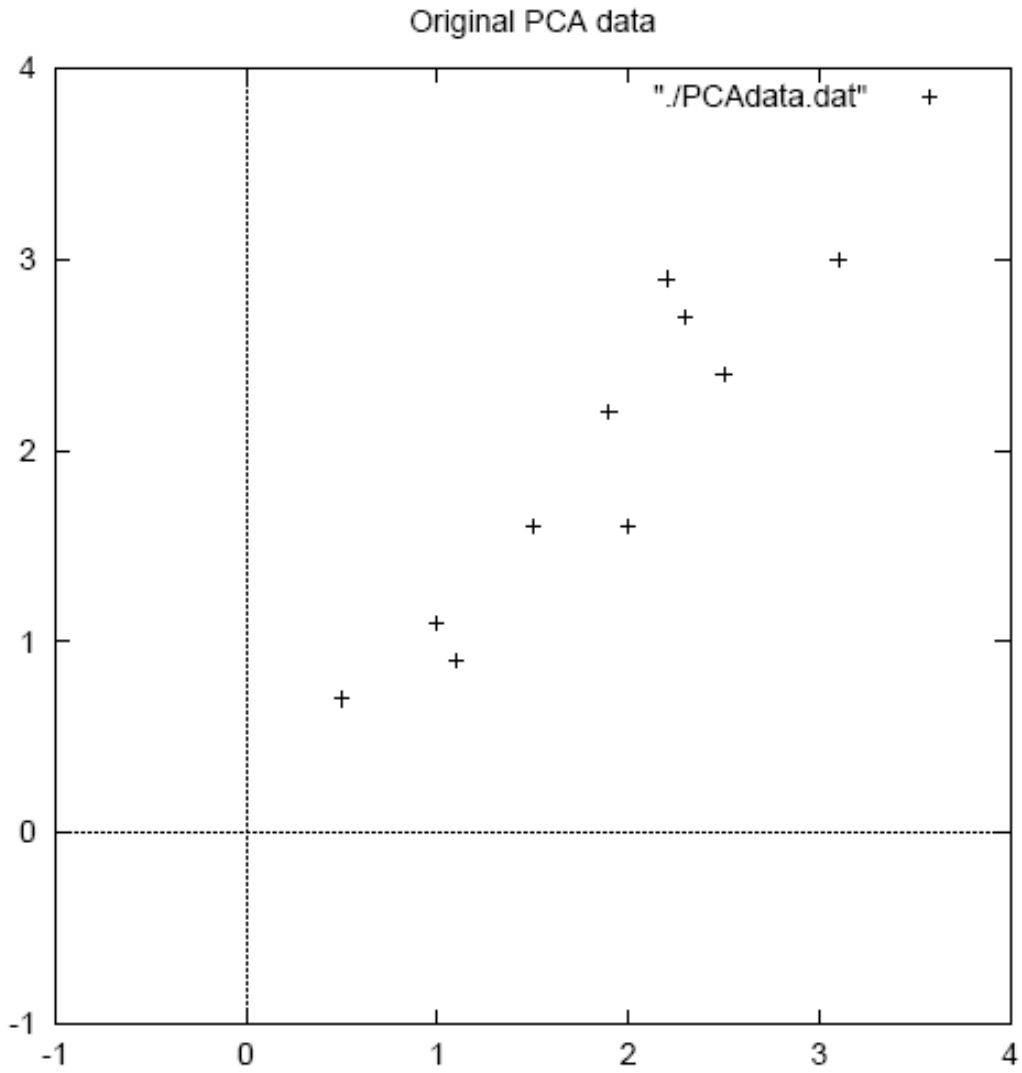






Example of Principle Component Analysis

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9



Example of Principle Component Analysis (cont.)

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

μ 1.81 1.91

input vector

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

subtracting the mean vector

$\det(cov - \lambda \cdot I) \cdot U = 0$

$$cov = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

Var(x') Cov(x', y')

Cov(y', x') Var(y')

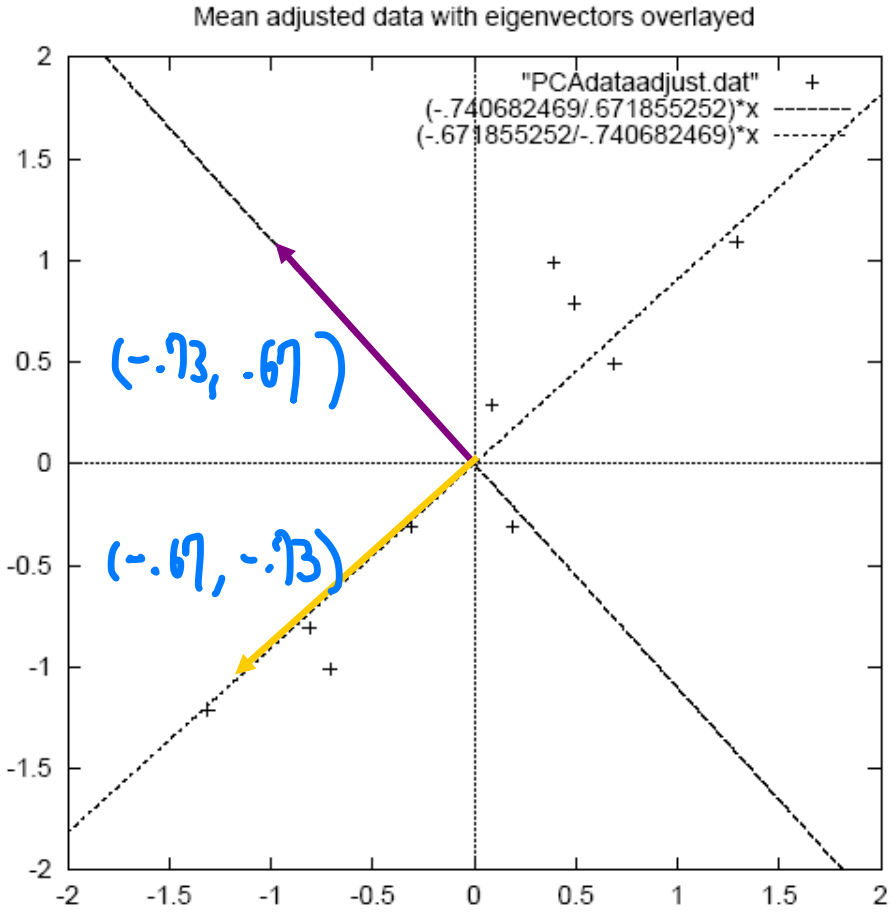
Covariance matrix

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

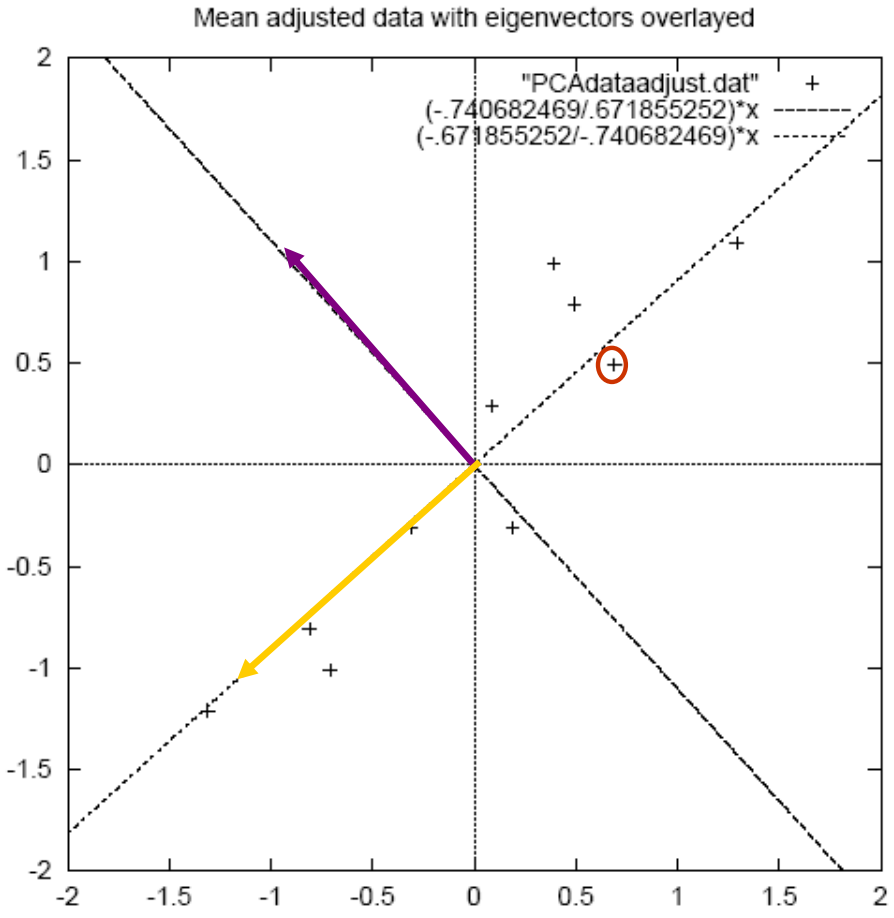
Example of Principle Component Analysis (cont.)

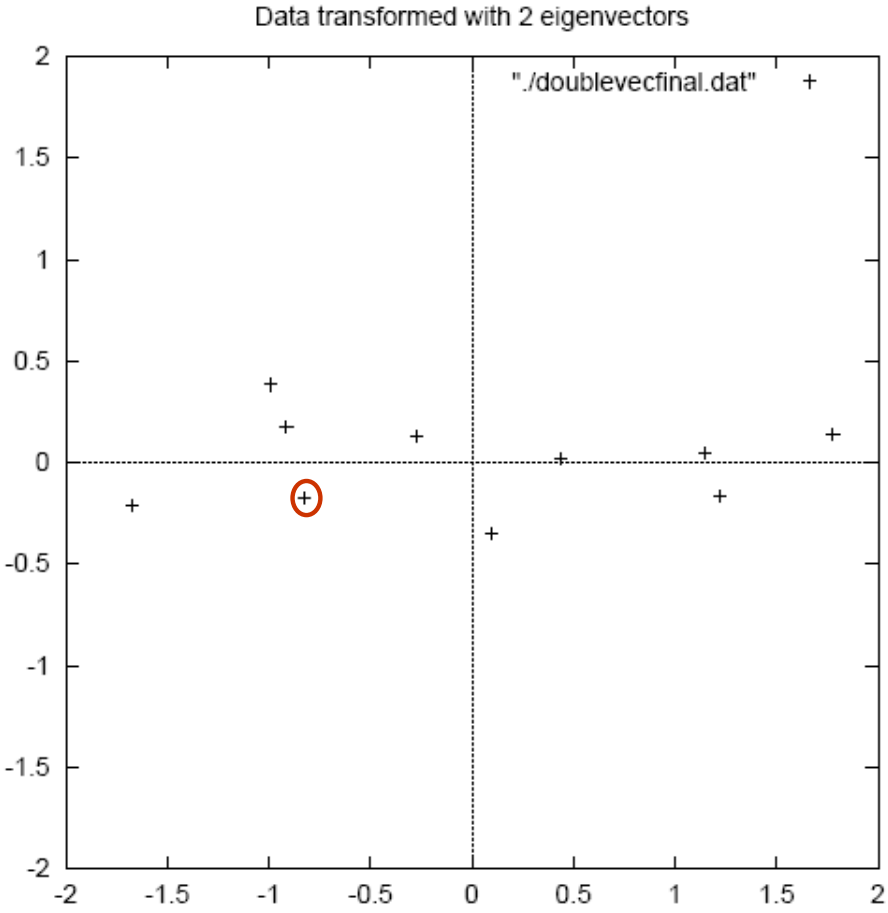
x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01



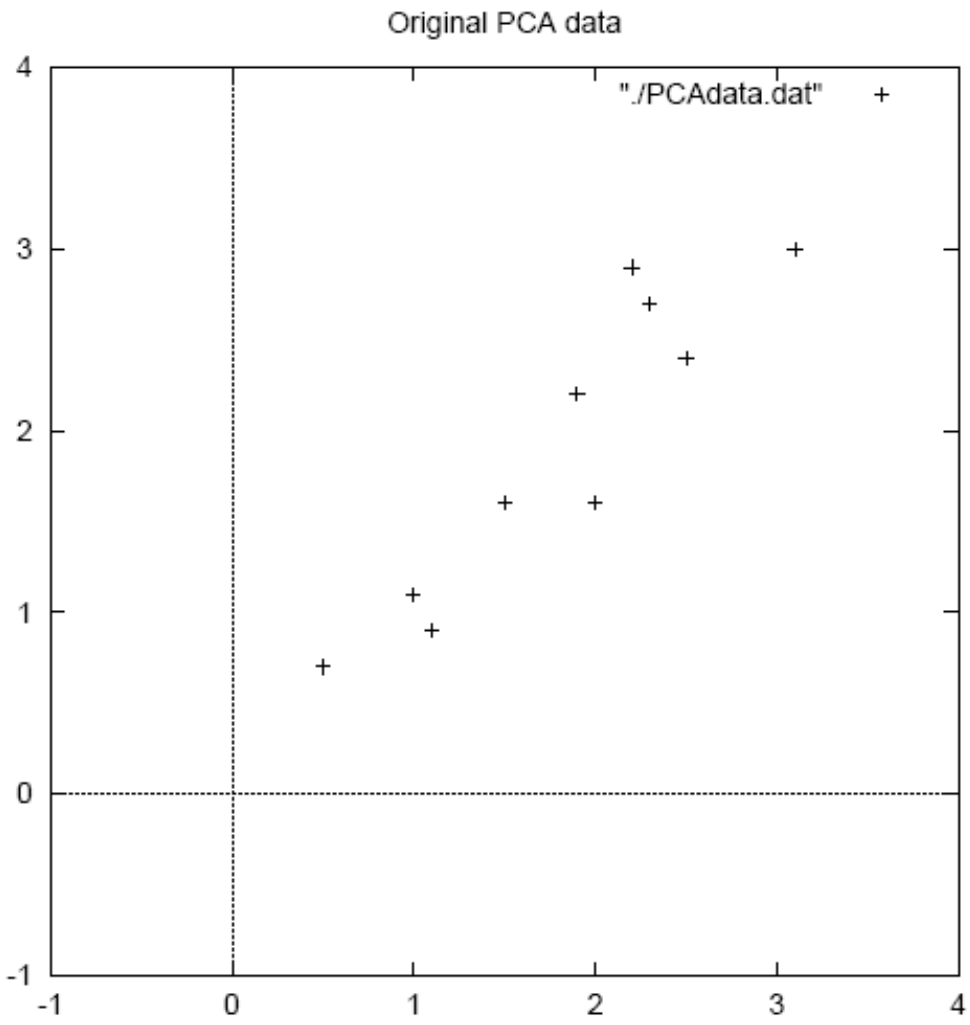
$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

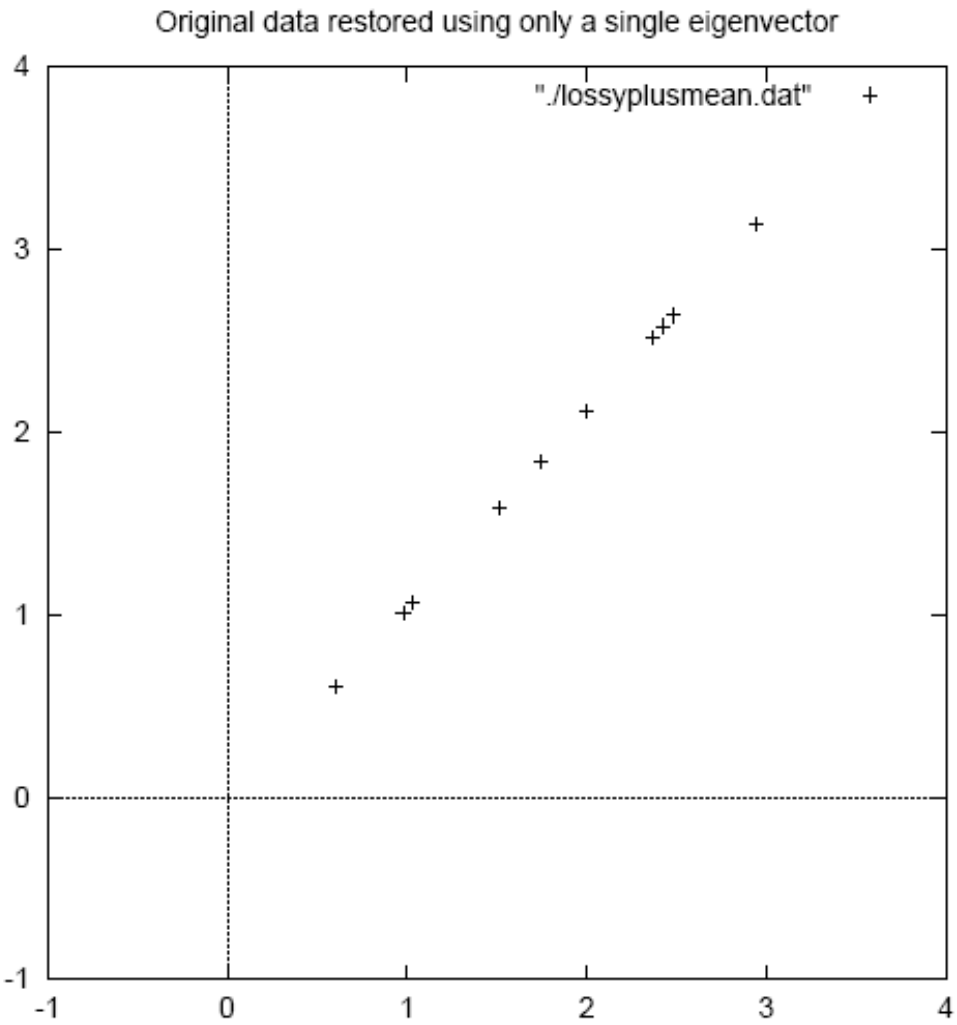
Example of Principle Component Analysis (cont.)





PCA


Original PCA data
"./PCAdata.dat"
4
3
2
1
0
-1
-1
0
1
2
3
4


Original data restored using only a single eigenvector
"./lossyplusmean.dat"
4
3
2
1
0
-1
-1
0
1
2
3
4