CS665: Advanced Data Mining

Lecture#16: SVD-2

U Kang
KAIST

Outline

- **Multi-lingual IR; LSI queries**
- Compression
- PCA – ‘ratio rules’
- Karhunen-Lowe transform
- Conclusion

Case study - LSI

- Q1: How to do queries with LSI?

  Q2: multi-lingual IR (english query, on spanish text?)

Case study - LSI

Q1: How to do queries with LSI?

Problem: Eg., find documents with 'data'

$$
\begin{array}{c|c}
 & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\
\begin{array}{c} \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \\ \end{array} &
\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}
= \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix}
\times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix}
\times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}
\end{array}
$$

Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into 'concept space' – how?

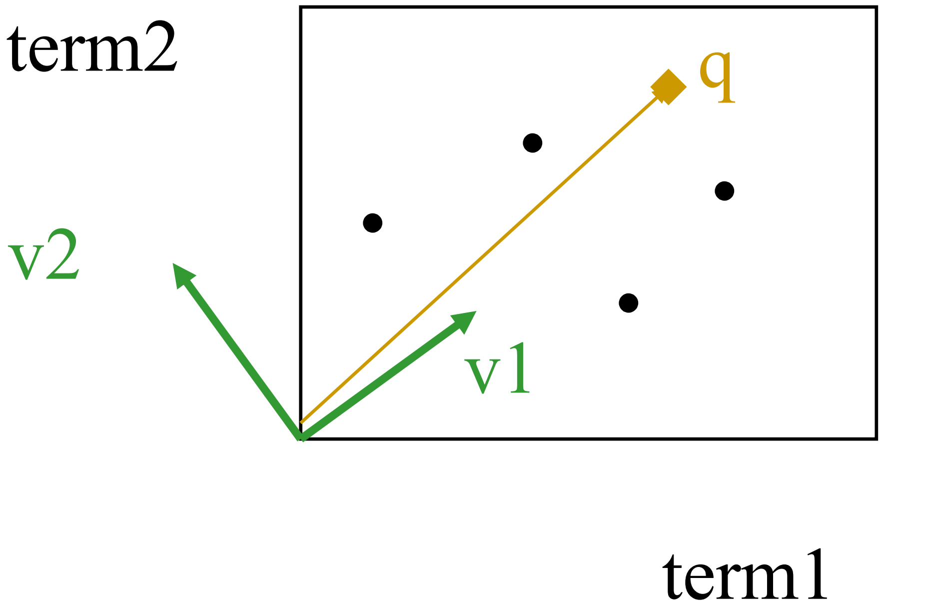
$$
\begin{array}{cc}
 & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\
\begin{array}{c} \\ \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \end{array} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}
\end{array}
=
\begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix}
\text{x}
\begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix}
\text{x}
\begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}
$$

Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into 'concept space' – how?

$$q = \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\ \end{array}$$
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

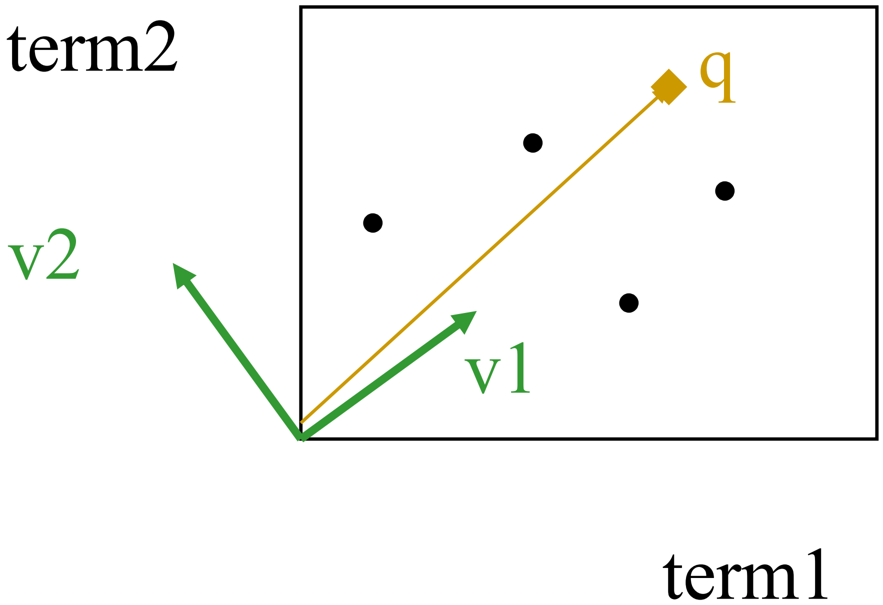


Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$q = \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\ \hline \end{array}$$
$$q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$



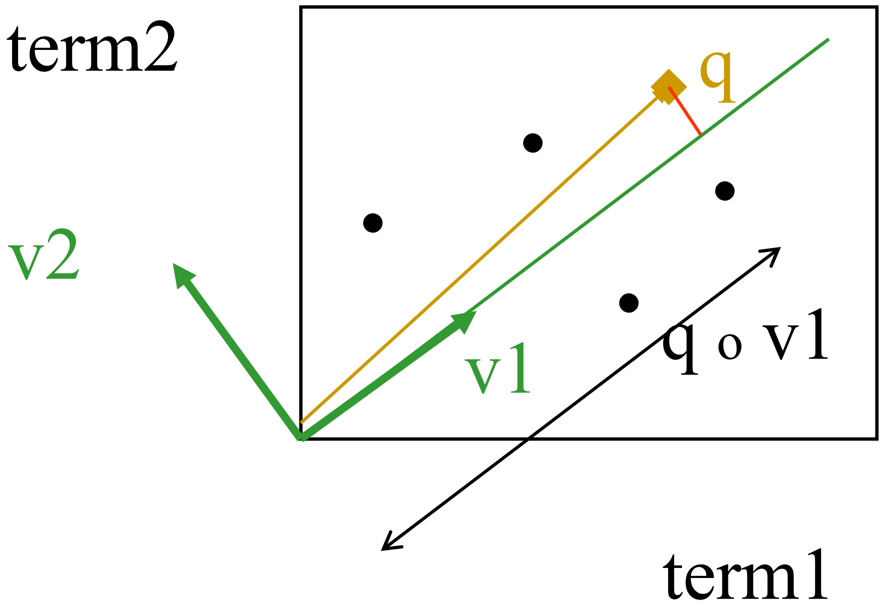
A: inner product (cosine similarity) with each ‘concept’ vector v_i

Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$q = \begin{array}{c} \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$



A: inner product (cosine similarity) with each ‘concept’ vector v_i

Case study - LSI

compactly, we have:

$$q\,\mathbf{V} = q_{concept}$$

Eg:

$$\mathbf{q} = \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\ \end{array}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{bmatrix} = \begin{array}{cc} \text{CS-concept} & \\ \downarrow & \\ \end{array} \begin{bmatrix} 0.58 & 0 \end{bmatrix}$$

term-to-concept similarities

Case study - LSI

Q: how would the document ('information', 'retrieval') be handled by LSI?

Case study - LSI

Q: how would the document ('information', 'retrieval ') be handled by LSI? A: SAME:

$d_{concept} = d\,\mathbf{V}$

Eg:

$$d = \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\ \end{array}$$

$$d = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{bmatrix} = \begin{bmatrix} 1.16 & 0 \end{bmatrix}$$

CS-concept

term-to-concept similarities

Case study - LSI

Observation: document ('information', 'retrieval') will be retrieved by query ('data'), although it does not contain 'data'!!

$$\begin{array}{cc} & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\ d= & \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \\ q= & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} \\ \dashrightarrow \\ \dashrightarrow \end{array} \begin{array}{c} \begin{array}{cc} \text{CS-concept} & \end{array} \\ \begin{bmatrix} 1.16 & 0 \end{bmatrix} \\ \begin{bmatrix} 0.58 & 0 \end{bmatrix} \end{array}$$

Case study - LSI

- Q1: How to do queries with LSI?
- Q2: multi-lingual IR (english query, on spanish text?)

Case study - LSI

- Problem:
  - Given: many documents, translated to both languag es (e.g., English and Spanish)
  - Task: answer queries across languages
    - E.g., Given an English query, find relevant Spanish docu ments

Case study - LSI

- Solution: ~ LSI

$$
\begin{array}{c|ccccc:ccccc}
 & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} & \text{datos} & \text{informacion} & & & \\
\hline
\text{CS} & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\text{CS} & 2 & 2 & 2 & 0 & 0 & 1 & 2 & 2 & 0 & 0 \\
\text{CS} & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\text{CS} & 5 & 5 & 5 & 0 & 0 & 5 & 5 & 4 & 0 & 0 \\
\text{MD} & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 2 & 2 \\
\text{MD} & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 2 & 3 \\
\text{MD} & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1
\end{array}
$$

Outline

- ☑ Multi-lingual IR; LSI queries
- ☐ **Compression**
- ☐ PCA – ‘ratio rules’
- ☐ Karhunen-Lowe transform
- ☐ Conclusion

Case study: compression

[Korn+97]

Problem:

- Given: a matrix
- Task: compress it, but maintain ‘random access’

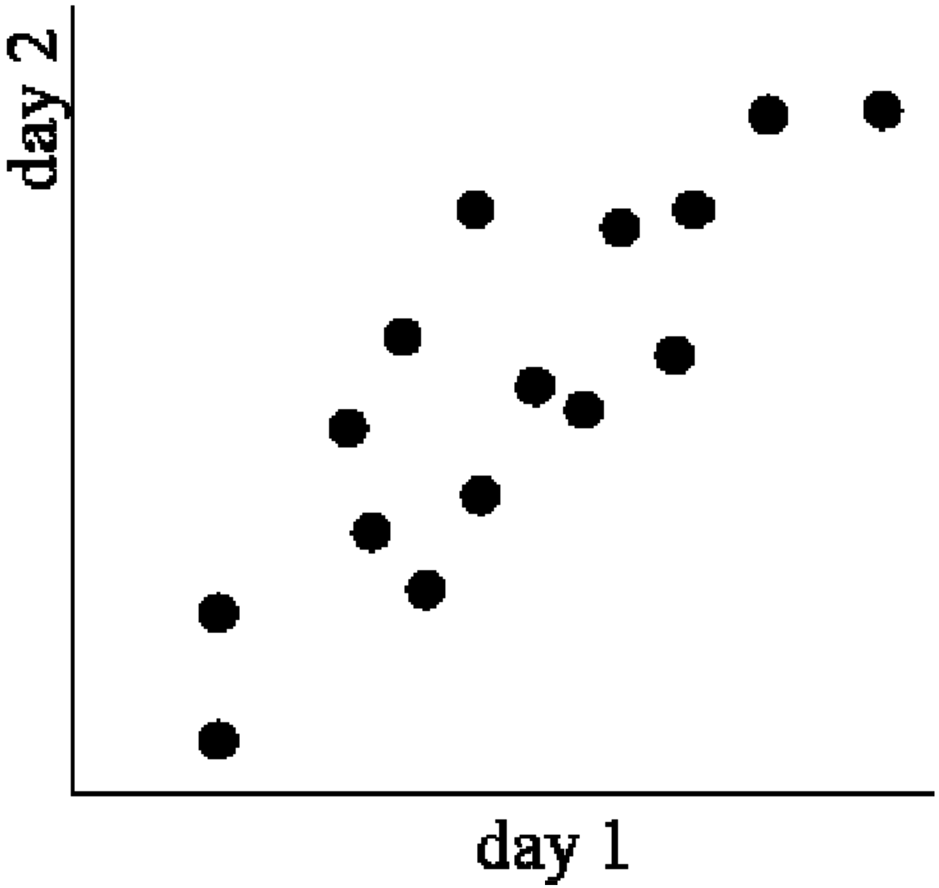
(surprisingly, its solution leads to data mining and vi sualization...)

Problem - specs

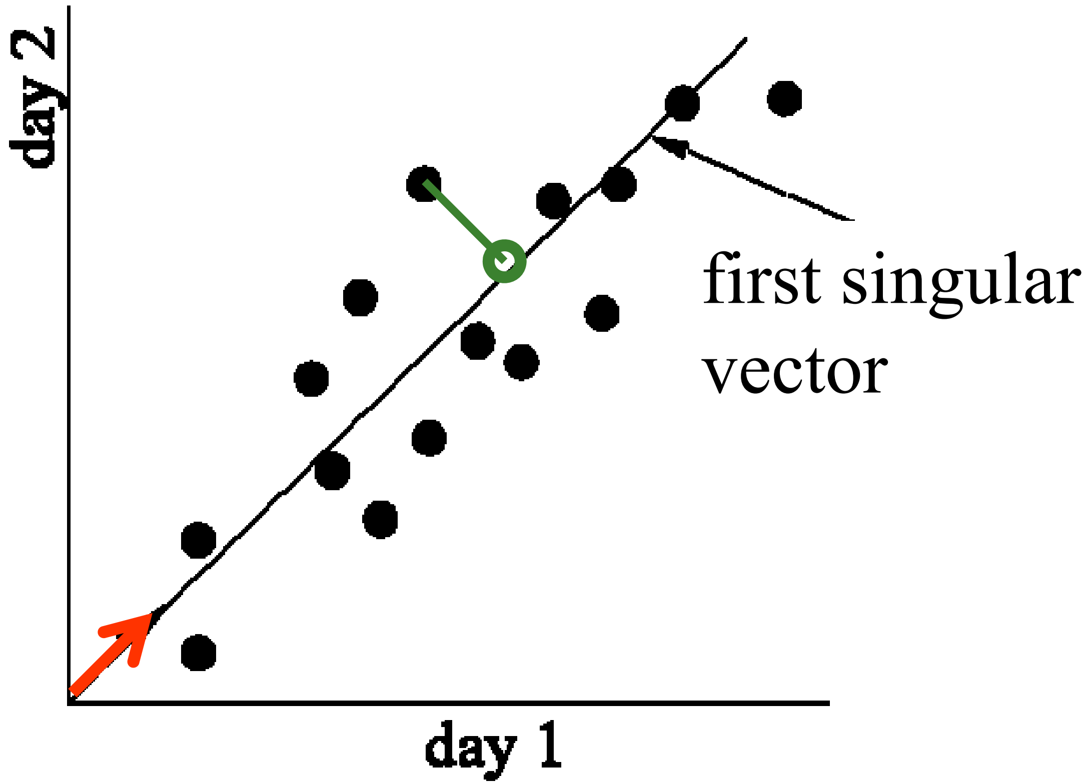
- $\sim 10^6$ rows; $\sim 10^3$ columns; no updates;
- random access to any cell(s) ; small error: OK

day / customer	We 7/10/96	Th 7/11/96	Fr 7/12/96	Sa 7/13/96	Su 7/14/96
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	1	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

Idea



SVD - reminder



- space savings: 2:1
- minimum RMS error

Compression - Algorithm

- Compression in SVD
  - If we want to decrease the error, we can simply increase k (# of singular values)
  - However, we need (N+M) additional space to increase k by 1, for N x M matrix
  - Is there better way (i.e., minimize error *more* with *the same* space)?

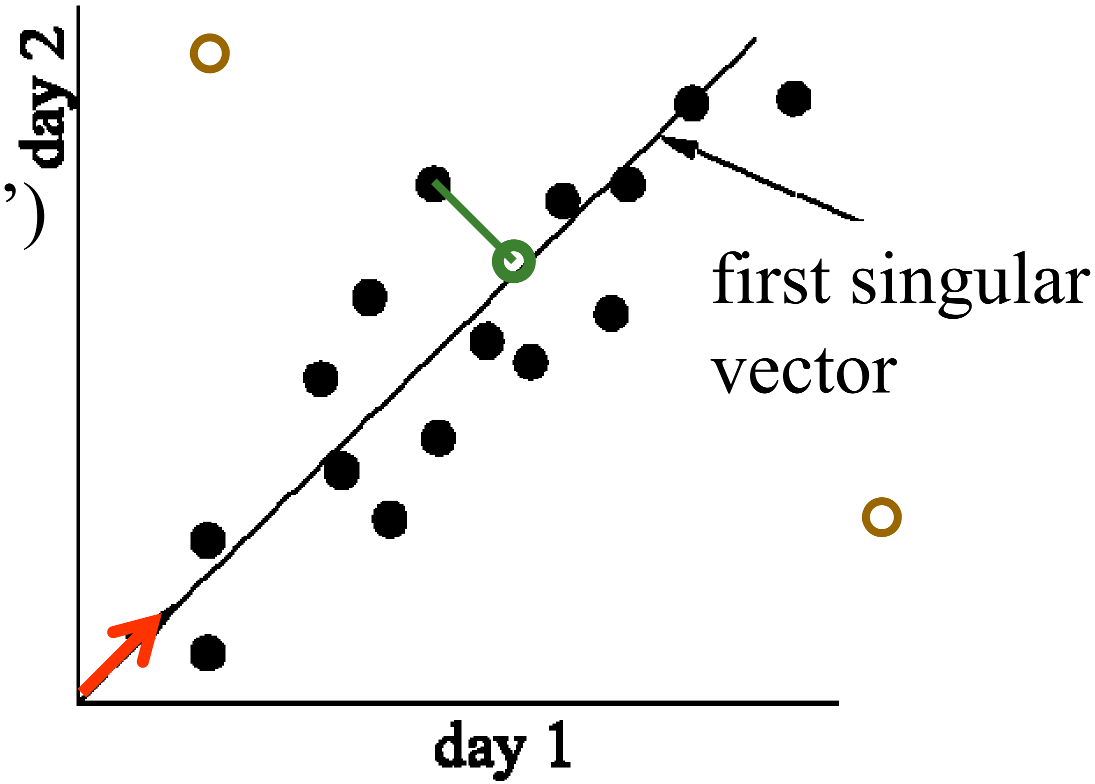
Compression - Algorithm

- Compression in SVD
  - If we want to decrease the error, we can simply increas e k (# of singular values)
  - However, we need (N+M) additional space to increase k by 1, for N x M matrix
  - Is there better way (i.e., minimize error *more* with *the s ame* space)?
- SVDD (SVD with Deltas)
  - Instead of increasing k, store outliers(=points with the highest errors) separately

Case study: compression

outliers?

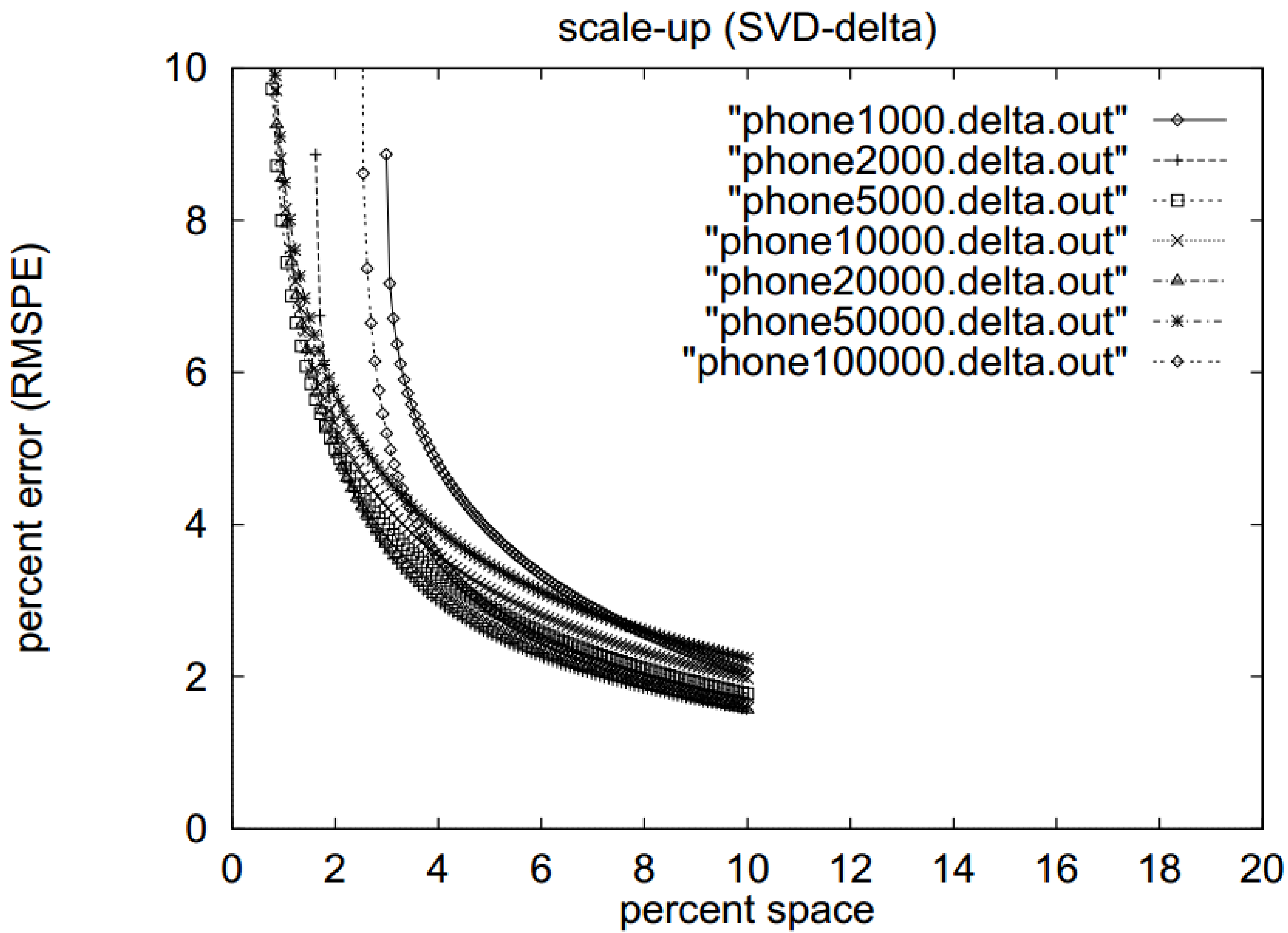
A: treat separately
(SVD with ‘Deltas’)



Compression - Performance

- random cell(s) reconstruction
- 10:1 compression with < 2% error

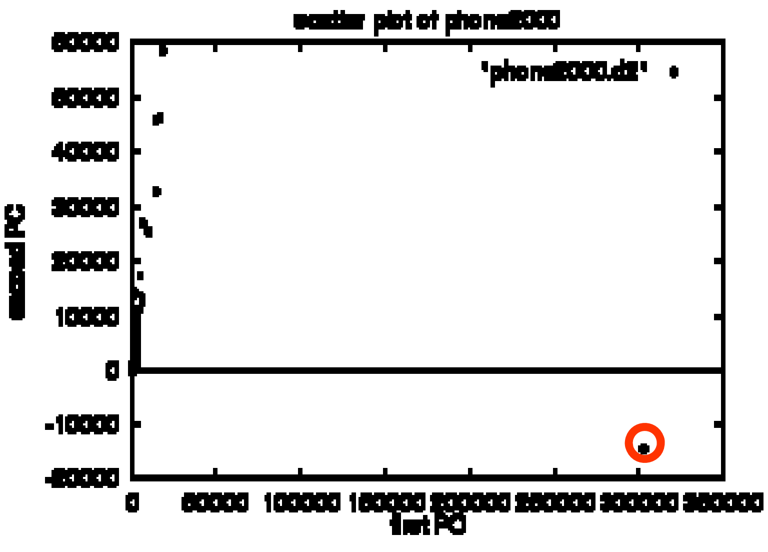
Performance - scaleup



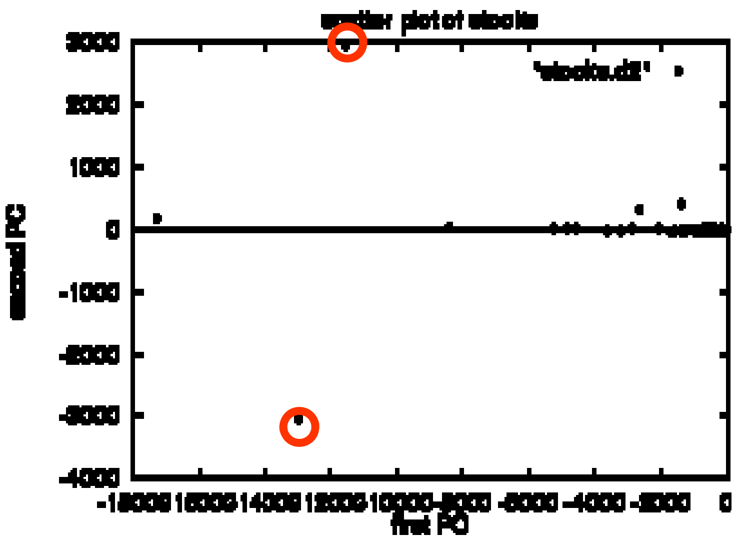
SVDD.

Compression - Visualization

- no Gaussian clusters; Zipf-like distribution



(a) 'phone2000'



(b) 'stocks'

Outline

- ☑ Multi-lingual IR; LSI queries
- ☑ Compression
- ➡ ☐ **PCA – 'ratio rules'**
- ☐ Karhunen-Lowe transform
- ☐ Conclusion

PCA - 'Ratio Rules'

[Korn+00]

Typically: '*Association Rules*' (eg.,

{bread, milk} -> {butter}

But:

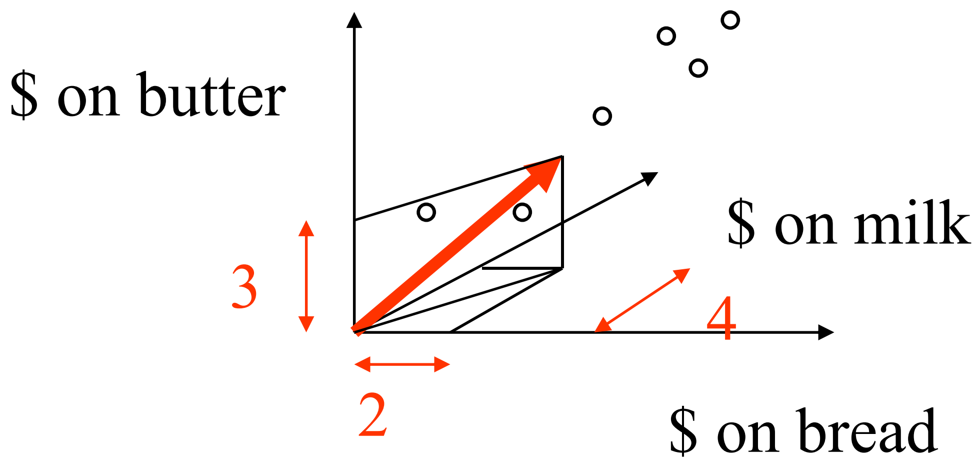
- which set of rules is 'better'?
- how to reconstruct missing/corrupted values?
- need binary/bucketized values

PCA - 'Ratio Rules'

Idea: try to find 'concepts':

- singular vectors dictate rules about ratios:

  bread:milk:butter = 2:4:3



PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- Q1: which set of rules is 'better'?
- Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= 'principal components ')?

PCA - 'Ratio Rules'

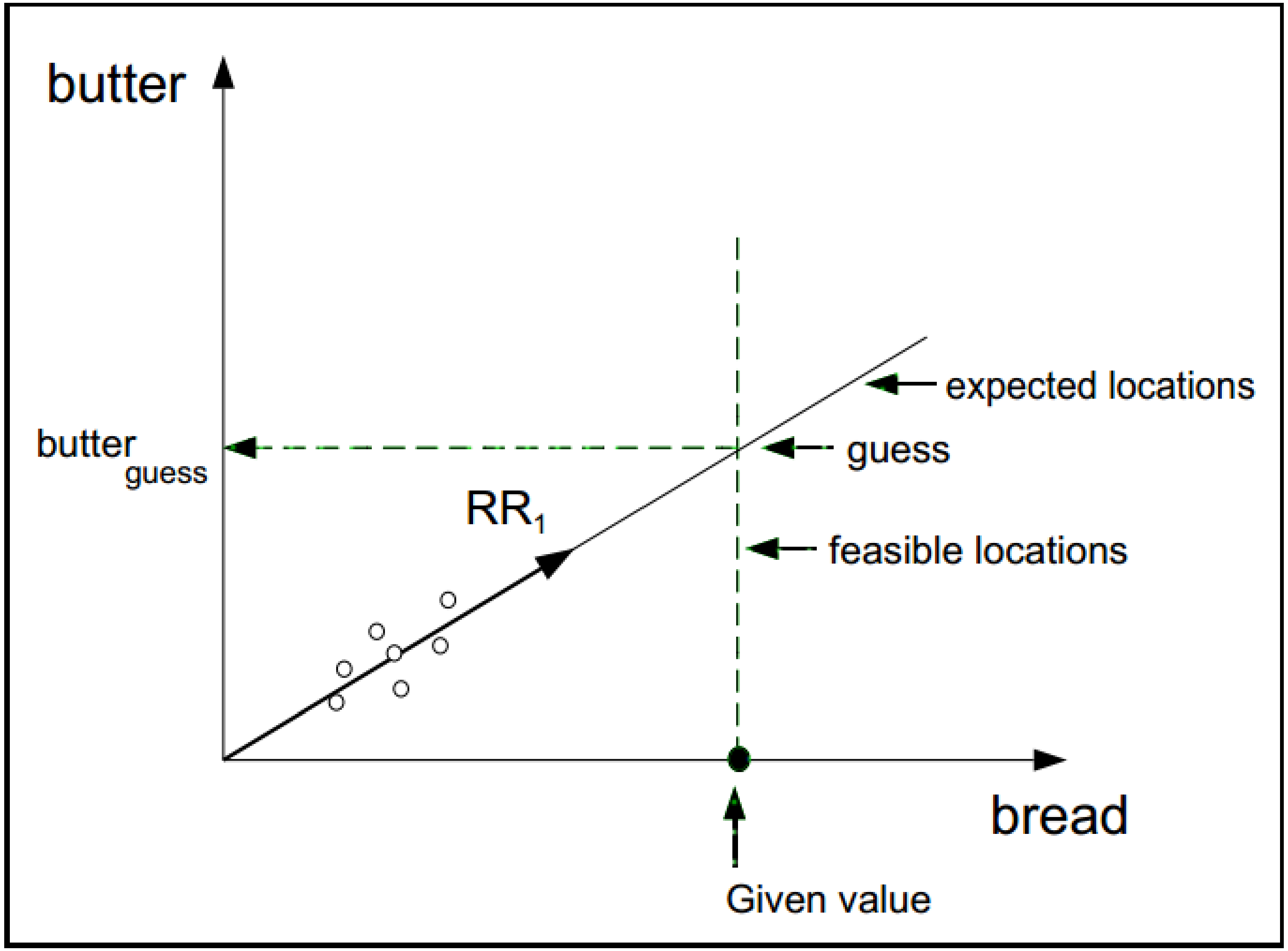
Q2: how to reconstruct missing/corrupted values?

Eg:

- rule: bread:milk = 3:4
- a customer spent \$6 on bread - how about milk?

PCA - 'Ratio Rules'

pictorially:



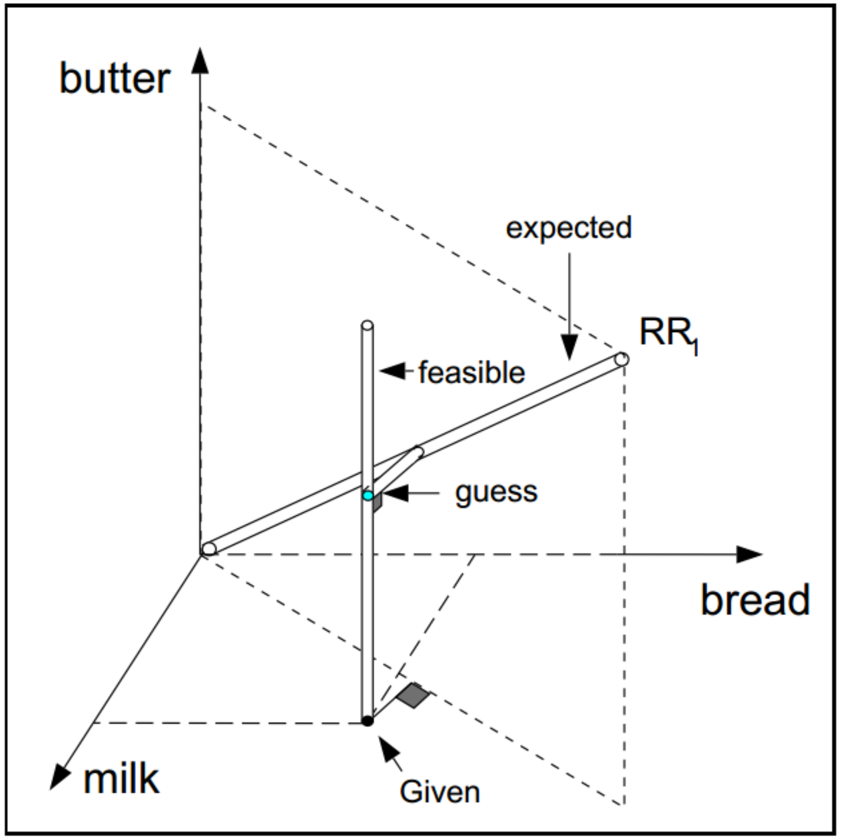
PCA - 'Ratio Rules'

details

harder cases: overspecified/underspecified

over-specified:
- milk:bread:butter = 1:2:3
- a customer got
  - $2 bread and $4 milk
- how much butter?

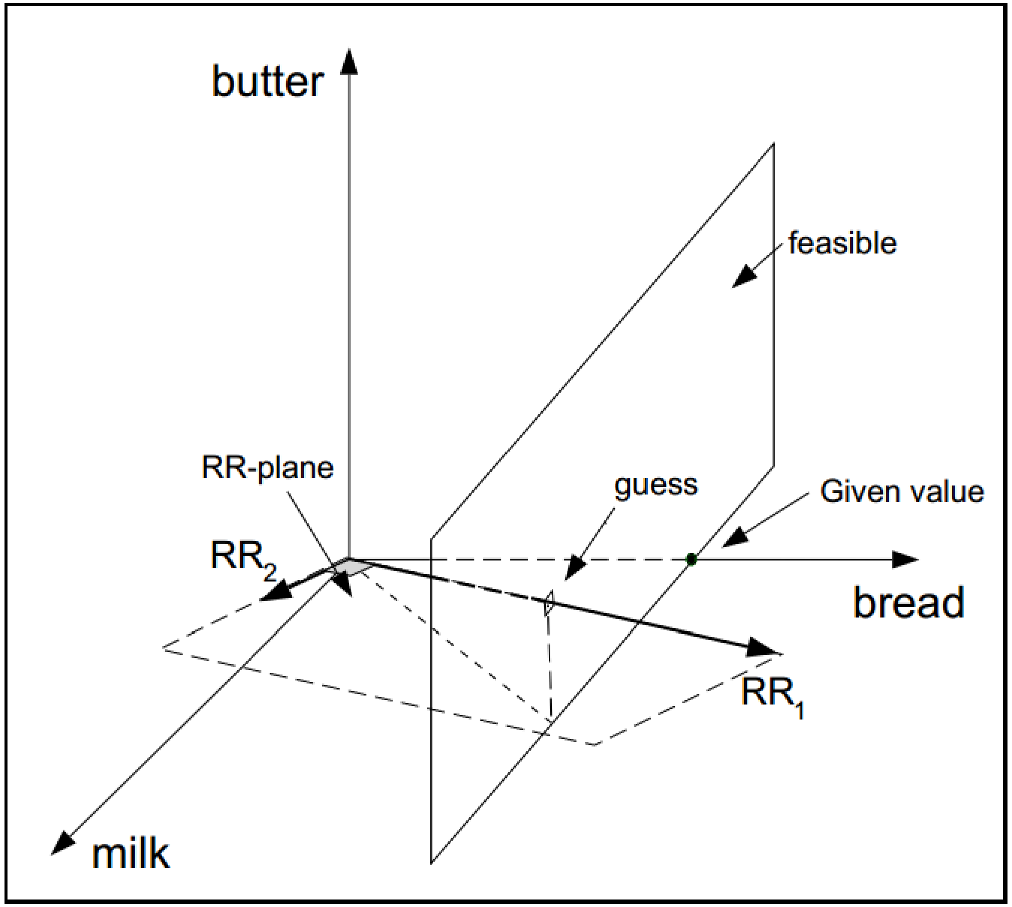
Answer: minimize distance between 'feasible' and 'expected' values



PCA - 'Ratio Rules'

details

harder cases: underspecified



PCA - 'Ratio Rules'

bottom line: we can reconstruct any count of missin g values

This is very useful:

- can spot outliers (how?)
- can measure the 'goodness' of a set of rules (how ?)

PCA - ‘Ratio Rules’

Identical to PCA = Principal Components Analysis

- Q1: which set of rules is ‘better’?
- Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= ‘principal components’)?

PCA - ‘Ratio Rules’

- Q1: which set of rules is ‘better’?
- A: the ones that need the fewest outliers:
  - pretend we don’t know a value (eg., $ of ‘Smith’ on ‘br ead’)
  - reconstruct it
  - and sum up the squared errors, for all our entries

PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- ✓ Q1: which set of rules is 'better'?
- ✓ Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= 'principal components ')?

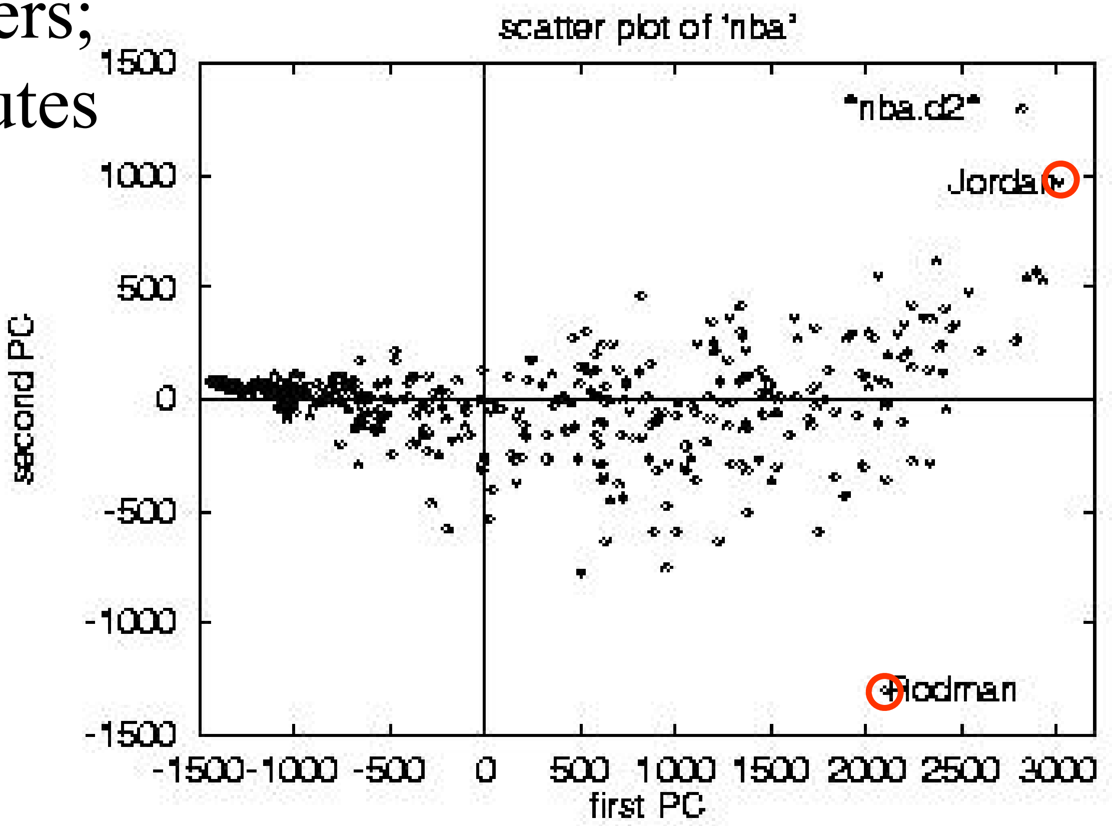
PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- Q1: which set of rules is 'better'?
- Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values? NO
- Q4: how to interpret the rules (= 'principal components')?

PCA - Ratio Rules

NBA dataset
~500 players;
~30 attributes



PCA - Ratio Rules

- PCA: get singular vectors v1, v2, ...
- ignore entries with small abs. value
- try to interpret the rest

PCA - Ratio Rules

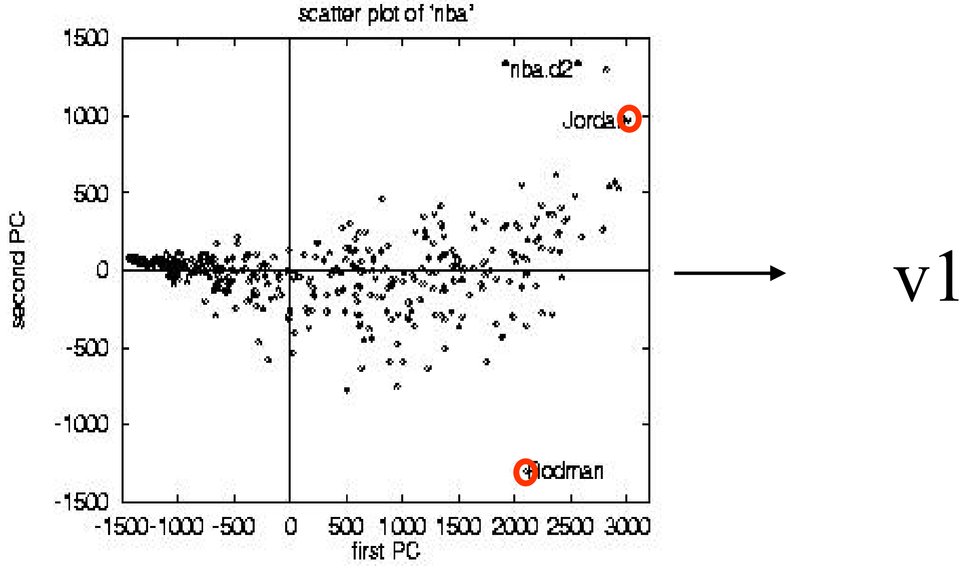
NBA dataset - **V** matrix (term to 'concept' similarities)

field	RR_1	RR_2	RR_3
minutes played	.808	$-.4$	
field goals			
goal attempts			
points	.406	.199	
total rebounds		$-.489$	.602
assists			$-.486$
steals			$-.07$

v1

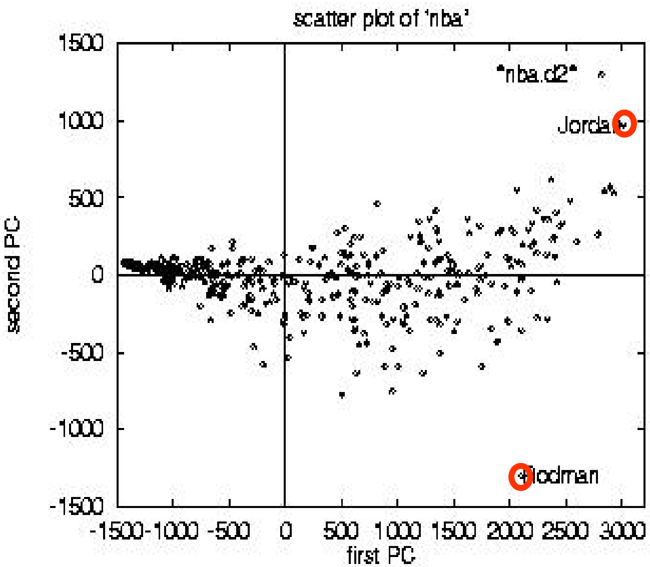
Ratio Rules - example

- RR1: minutes:points = 2:1
- corresponding concept?



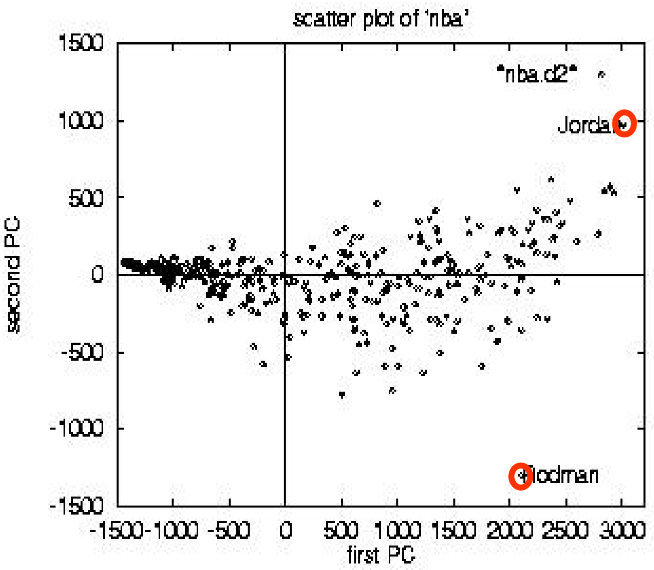
Ratio Rules - example

- RR1: minutes:points = 2:1

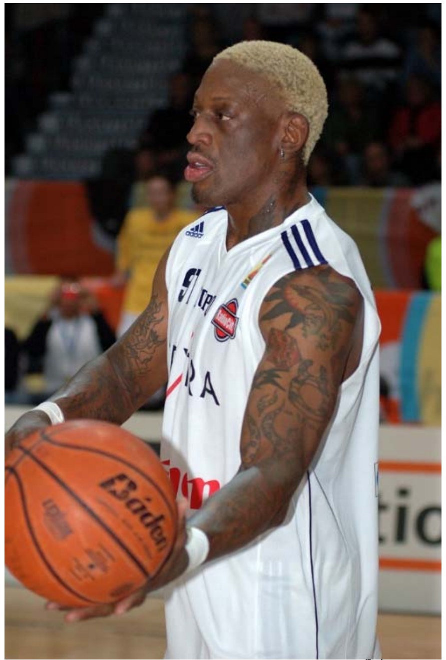


Ratio Rules - example

- RR1: minutes:points = 2:1



v1



Ratio Rules - example

- RR1: minutes:points = 2:1
- corresponding concept?
- A: ‘goodness’ of player

Ratio Rules - example

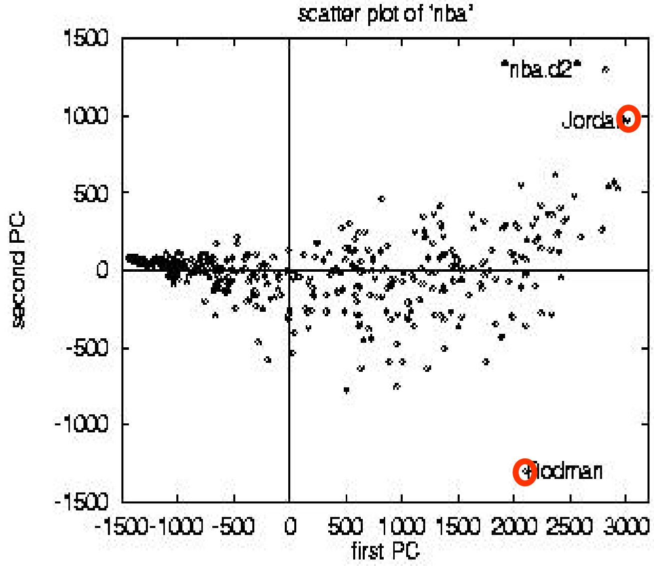
- RR2: points: rebounds negatively correlated(!)

field	RR_1	RR_2	RR_3
minutes played	.808	$-.4$	
field goals			
goal attempts			
points	.406	.199	
total rebounds		$-.489$	.602
assists			$-.186$
steals			$-.07$

Ratio Rules - example

- RR2: points: rebounds negatively correlated(!) - c oncept?

v2



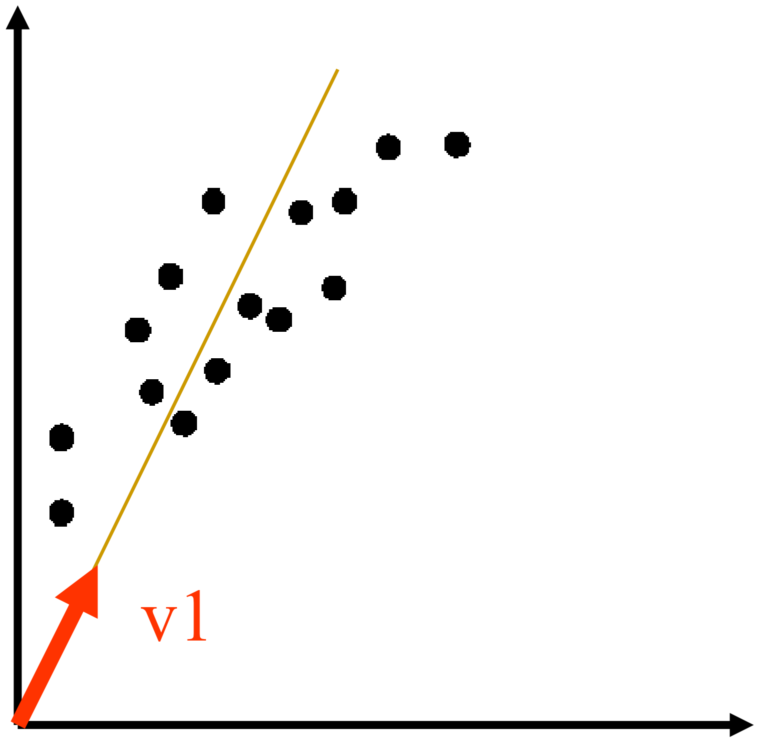
Ratio Rules - example

- RR2: points: rebounds negatively correlated(!) - c oncept?
- A: position: offensive/defensive

Outline

- ☑ Multi-lingual IR; LSI queries
- ☑ Compression
- ☑ PCA – ‘ratio rules’
- ☐ **Karhunen-Lowe transform**
- ☐ Conclusion

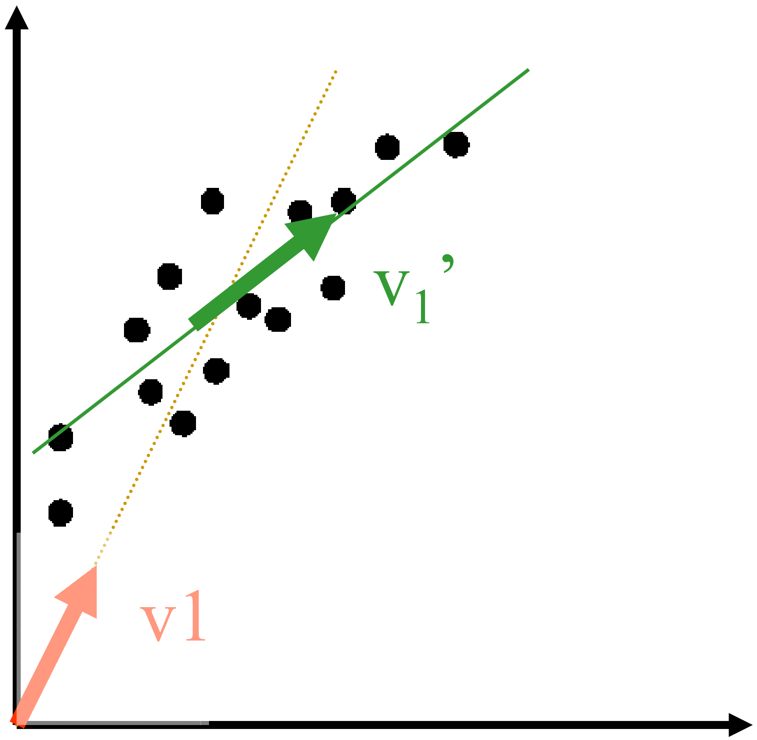
K-L transform (=PCA)



[Duda & Hart]; [Fukunaga]

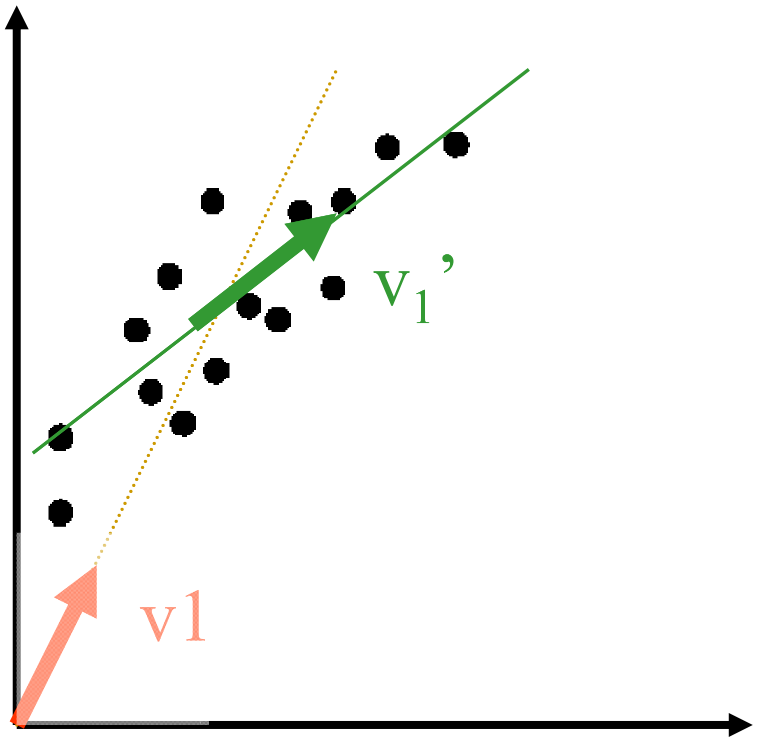
A subtle point:
SVD will give vectors that go through the origin

K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1' ?

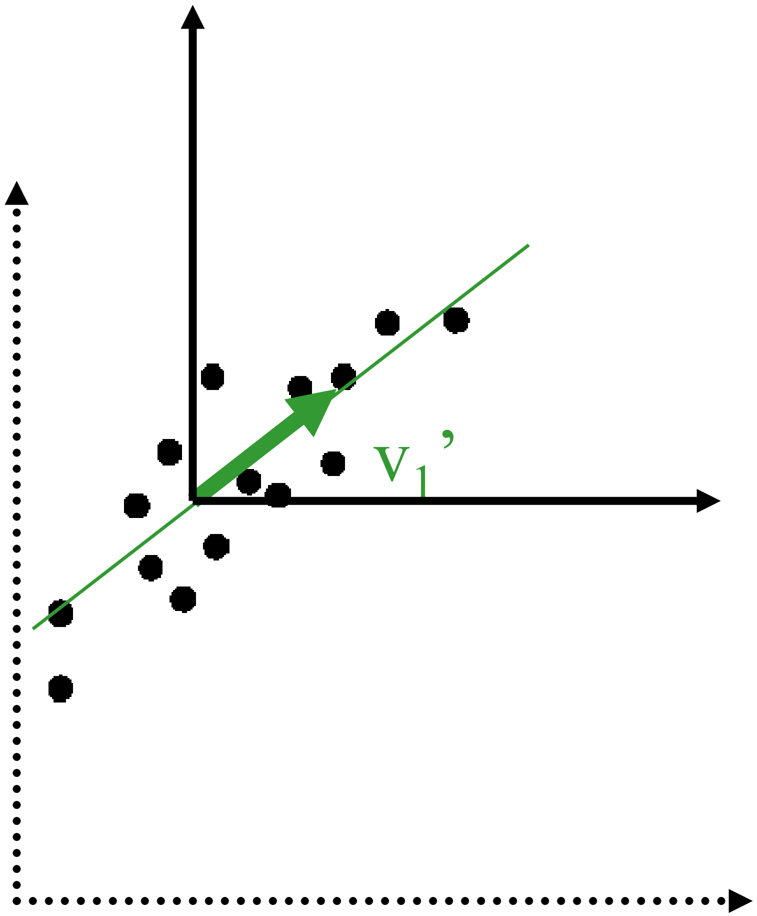
K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1' ?

A: 'centered' PCA, ie., move the origin to center of gravity

K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1' ?

A: 'centered' PCA, ie.,
move the origin to center of gravity
and THEN do SVD

K-L transform

- How to ‘center’ a set of vectors (= data matrix)?
- What is the covariance matrix?

K-L transform

- How to 'center' a set of vectors (= data matrix)?
- What is the covariance matrix?
  - Let A = N x n data matrix
  - Covariance matrix C = $B^T B$ where $b_{ij} = a_{ij} - \overline{a_{:j}}$
  - I.e., $c_{pq} = \sum_{i=1}^{N} (a_{ip} - \overline{a_{:p}})(a_{iq} - \overline{a_{:q}})$

  - Right singular vector is given by the eigenvector of C

Outline

- ☑ Multi-lingual IR; LSI queries
- ☑ Compression
- ☑ PCA – ‘ratio rules’
- ☑ Karhunen-Lowe transform
- ☐ **Conclusion**

Conclusions

- SVD: popular for dimensionality reduction / com pression
- SVD is the 'engine under the hood' for PCA (prin cipal component analysis)
- ... as well as the Karhunen-Lowe transform
- (and there is more to come ...)

References

- Duda, R. O. and P. E. Hart (1973). Pattern Classification and Scene Analysis. New York, Wiley.
- Fukunaga, K. (1990). Introduction to Statistical Pattern Recognition, Academic Press.
- Jolliffe, I. T. (1986). Principal Component Analysis, Springer Verlag.

References

- Korn, F., H. V. Jagadish, et al. (May 13-15, 1997). Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences. ACM SIGMOD, Tucson, AZ.
- Korn, F., A. Labrinidis, et al. (1998). Ratio Rules: A New Paradigm for Fast, Quantifiable Data Mining. VLDB, New York, NY.

References

- Korn, F., A. Labrinidis, et al. (2000). "Quantifiable Data Mining Using Ratio Rules." VLDB Journal 8(3-4): 254-266.
- Press, W. H., S. A. Teukolsky, et al. (1992). Numerical Recipes in C, Cambridge University Press.