Latent Semantic Indexing

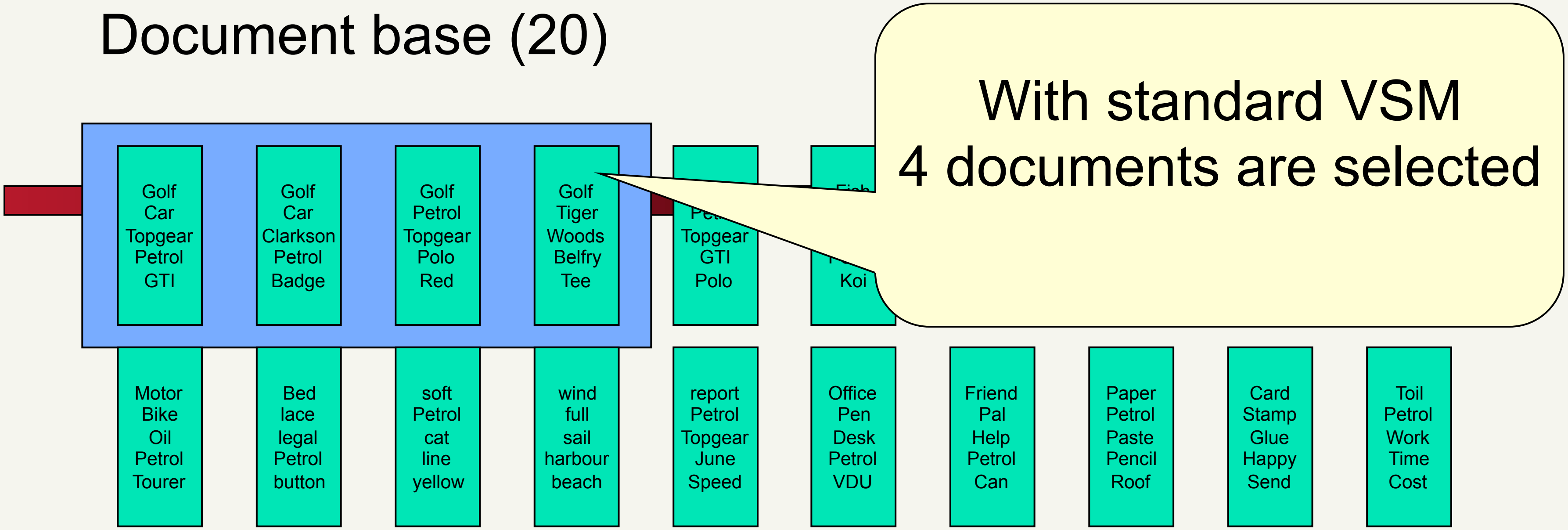
Latent Semantic Indexing

- Term-document matrices are very large, though most cells are “zeros”
- But the number of **topics** that people talk about is small (in some sense)
  - Clothes, movies, politics, …
  - Each topic can be represented as a cluster of (semantically) related terms, e.g.: clothes=golf, jacket, shoe..
- Can we represent the term-document space by a lower dimensional “latent” space (**latent space=set of topics**)?

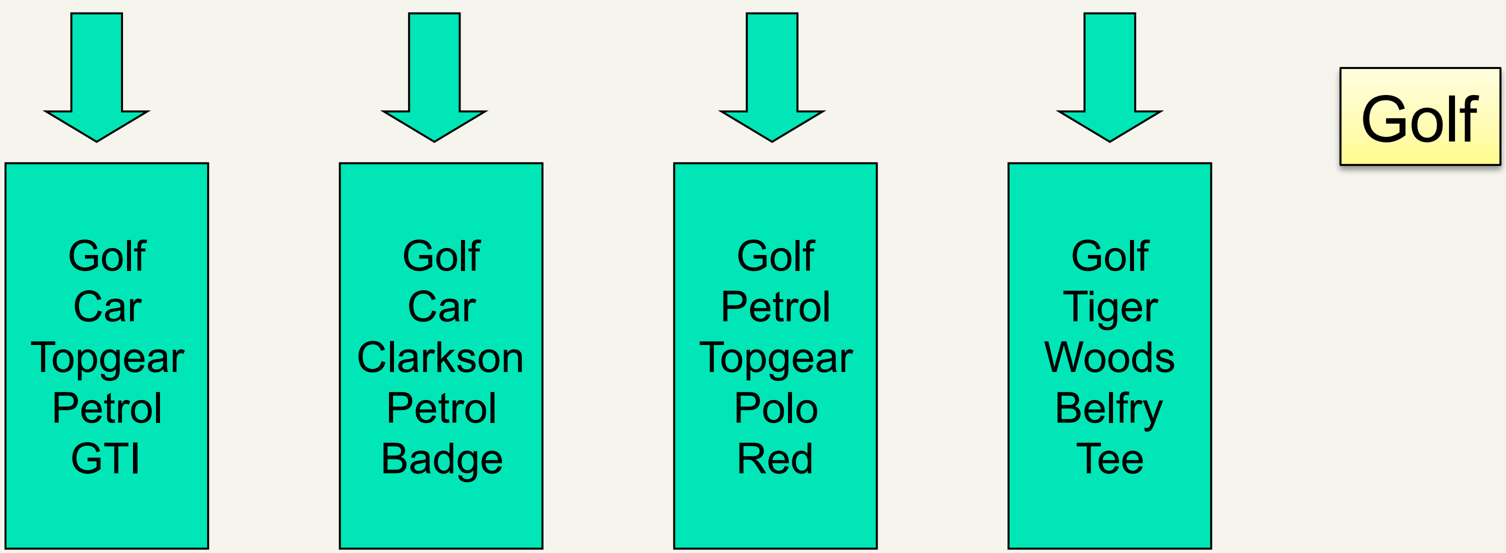
Searching with latent topics

- Given a collection of documents, LSI learns clusters of frequently co-occurring terms (ex: `information retrieval`, `ranking` and `web)`
- If you query with `ranking`, **`information retrieval`** LSI "automatically" extends the search to documents including also (and even ONLY) `web`

Document base (20)



Selection based on ‘Golf’



20 documents

Golf Car Topgear Petrol GTI

Golf Car Clarkson Petrol Badge

Golf Petrol Topgear Polo Red

Golf Tiger Woods Belfry Tee

Car Petrol Topgear GTI Polo

Fish Pond gold Petrol Koi

PC Dell RAM Petrol Floppy

Core Petrol Apple Pip Tree

Pea Pod Fresh Green French

Lupin Petrol Seed May April

Motor Bike Oil Petrol Tourer

Bed lace legal Petrol button

soft Petrol cat line yellow

wind full sail harbour beach

report Petrol Topgear June Speed

Office Pen Desk Petrol VDU

Friend Pal Help Petrol Can

Paper Petrol Paste Pencil Roof

Card Stamp Glue Happy Send

Toil Petrol Work Time Cost

Selection based on ‘Golf’

Golf Car Topgear Petrol GTI

Golf Car Clarkson Petrol Badge

Golf Petrol Topgear Polo Red

Golf Tiger Woods Belfry Tee

Rank of selected documents

tf.idf

Car

2 *(20/3) = 13

Topgear

2 *(20/3) = 13

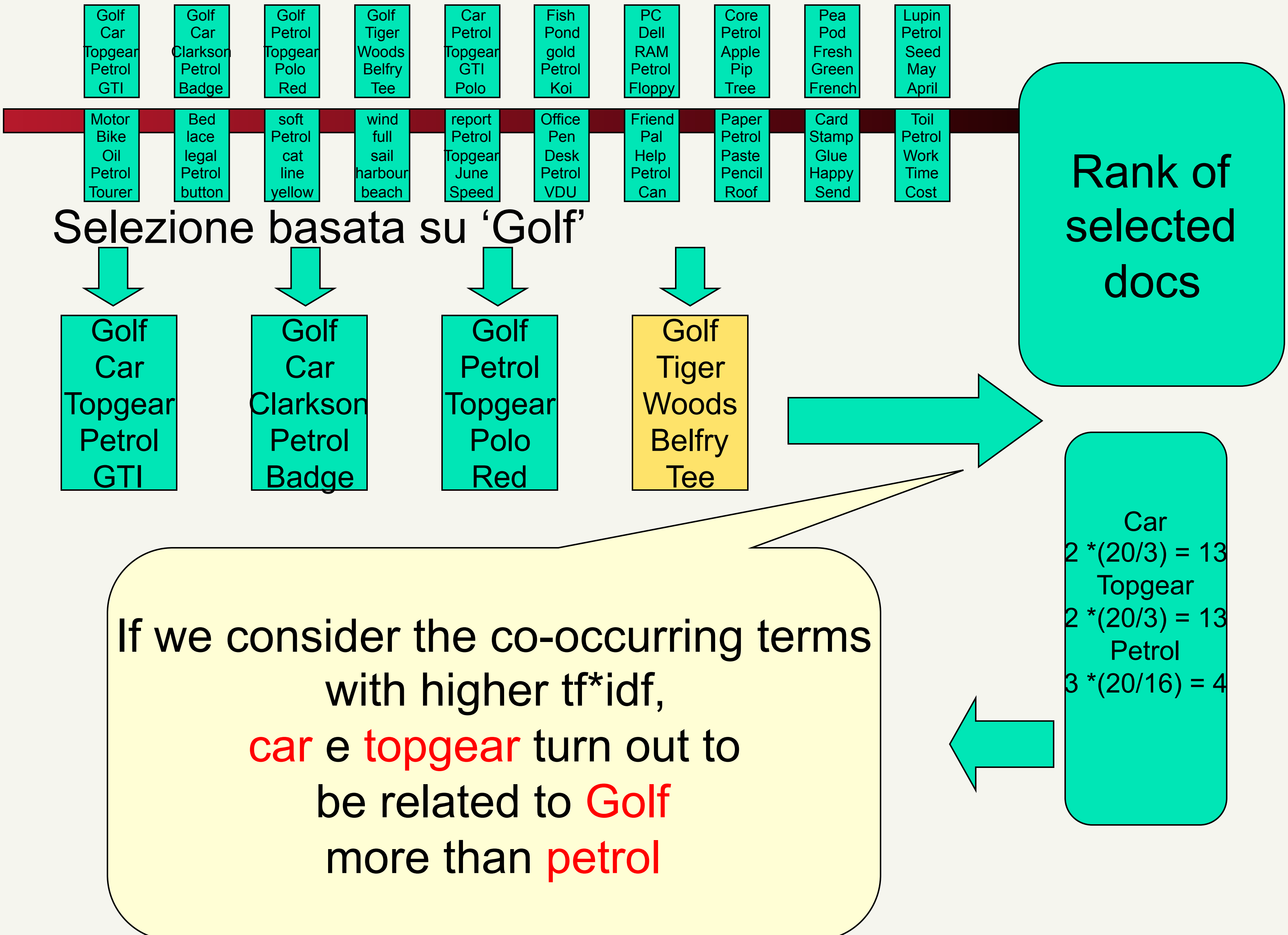
Petrol

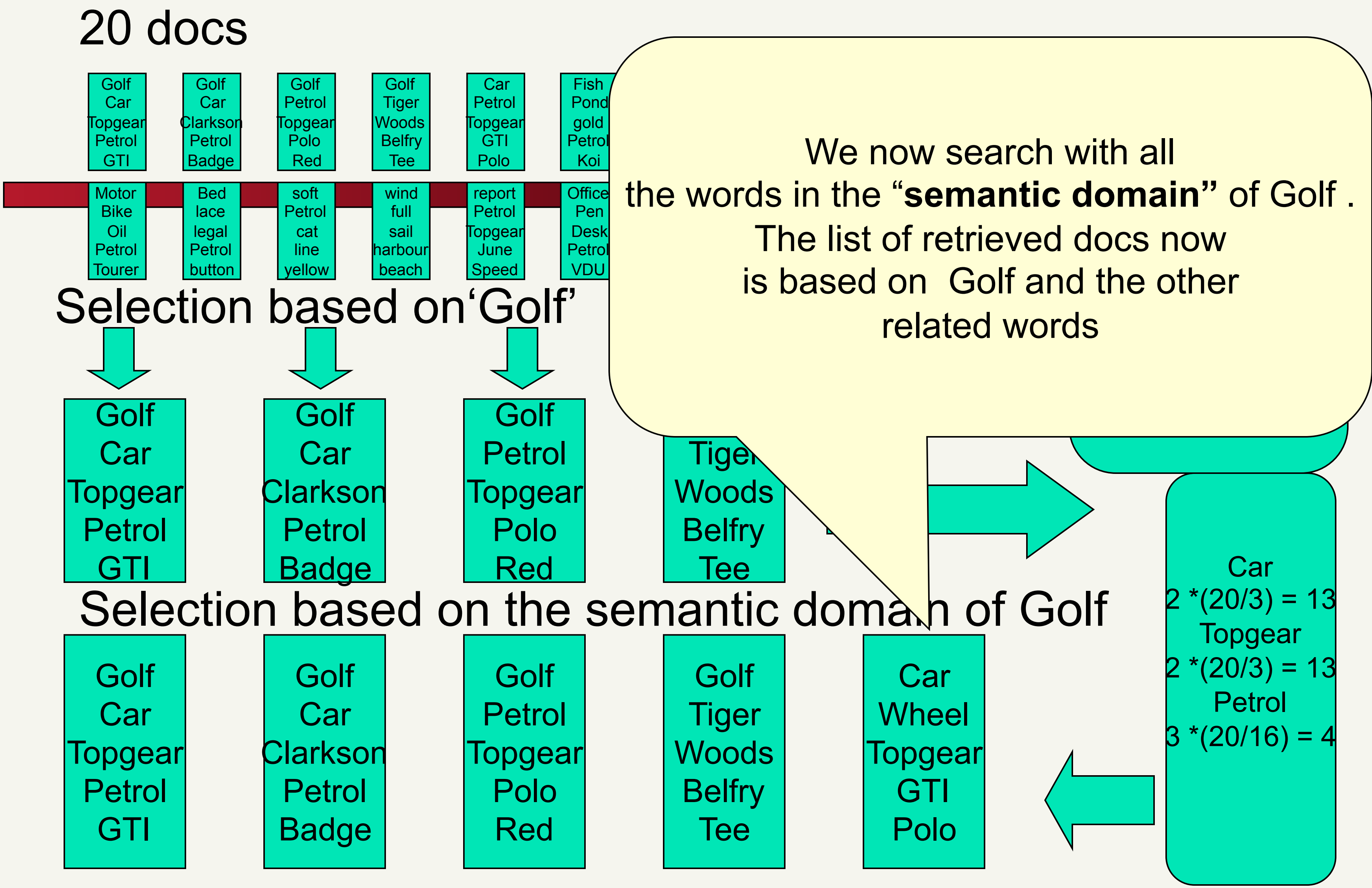
3 *(20/16) = 4

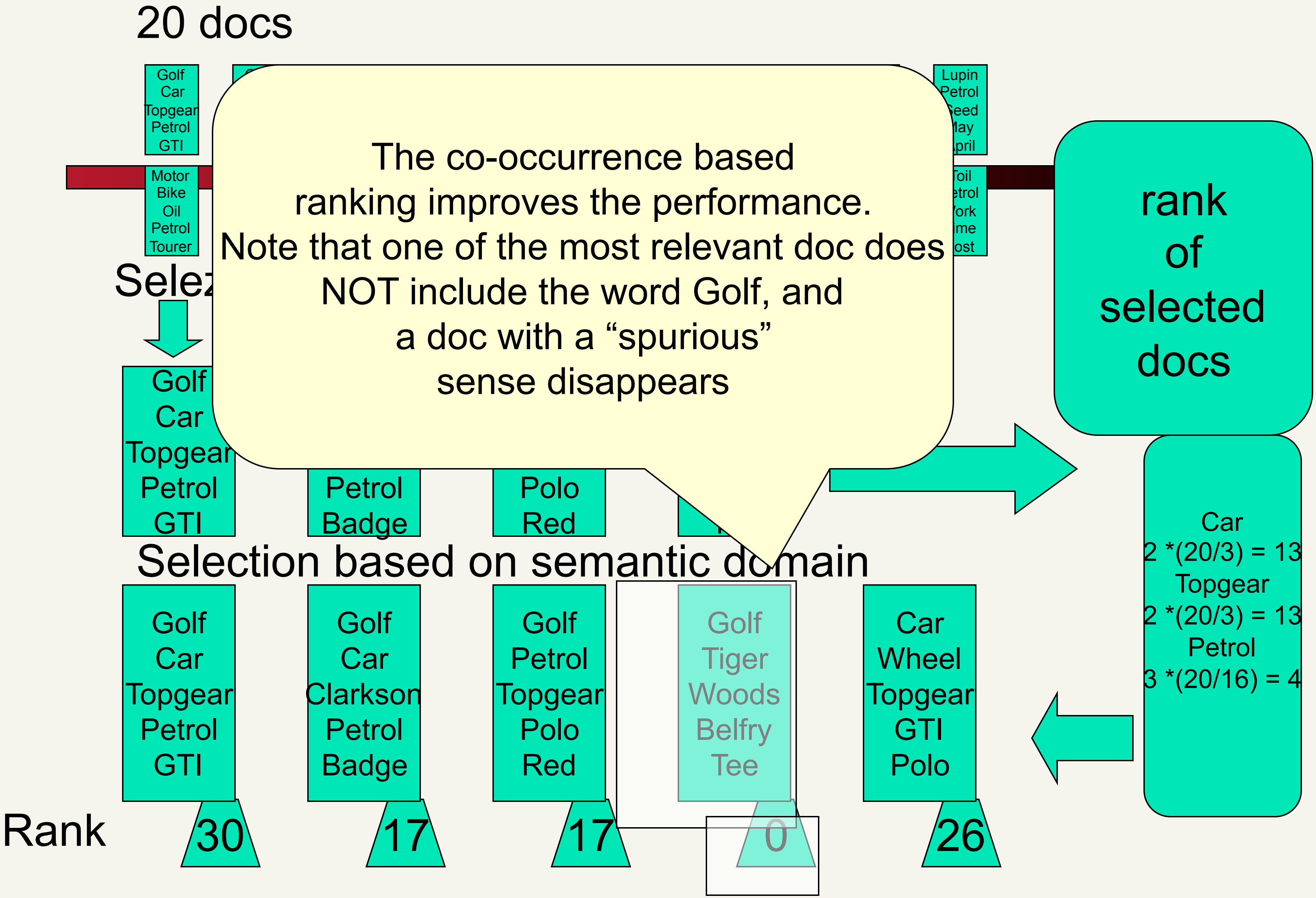
The most relevant words associated with golf in these docs are:

Car, Topgear and Petrol

20 docs


Golf Car Topgear Petrol GTI
Golf Car Clarkson Petrol Badge
Golf Petrol Topgear Polo Red
Golf Tiger Woods Belfry Tee
Car Petrol Topgear GTI Polo
Fish Pond gold Petrol Koi
PC Dell RAM Petrol Floppy
Core Petrol Apple Pip Tree
Pea Pod Fresh Green French
Lupin Petrol Seed May April
Motor Bike Oil Petrol Tourer
Bed lace legal Petrol button
soft Petrol cat line yellow
wind full sail harbour beach
report Petrol Topgear June Speed
Office Pen Desk Petrol VDU
Friend Pal Help Petrol Can
Paper Petrol Paste Pencil Roof
Card Stamp Glue Happy Send
Toil Petrol Work Time Cost
Selezione basata su ‘Golf’
Golf Car Topgear Petrol GTI
Golf Car Clarkson Petrol Badge
Golf Petrol Topgear Polo Red
Golf Tiger Woods Belfry Tee
Rank of selected docs
Car
2 *(20/3) = 13
Topgear
2 *(20/3) = 13
Petrol
3 *(20/16) = 4
If we consider the co-occurring terms with higher tf*idf, car e topgear turn out to be related to Golf more than petrol


20 docs
Golf Car Topgear Petrol GTI
Motor Bike Oil Petrol Tourer
Golf Car Clarkson Petrol Badge
Bed lace legal Petrol button
Golf Petrol Topgear Polo Red
soft Petrol cat line yellow
Golf Tiger Woods Belfry Tee
wind full sail harbour beach
Car Petrol Topgear GTI Polo
report Petrol Topgear June Speed
Fish Pond gold Petrol Koi
Office Pen Desk Petrol VDU
We now search with all the words in the "**semantic domain**" of Golf . The list of retrieved docs now is based on Golf and the other related words
Selection based on'Golf'
Golf Car Topgear Petrol GTI
Golf Car Clarkson Petrol Badge
Golf Petrol Topgear Polo Red
Tiger Woods Belfry Tee
Car 2 *(20/3) = 13
Topgear 2 *(20/3) = 13
Petrol 3 *(20/16) = 4
Selection based on the semantic domain of Golf
Golf Car Topgear Petrol GTI
Golf Car Clarkson Petrol Badge
Golf Petrol Topgear Polo Red
Golf Tiger Woods Belfry Tee
Car Wheel Topgear GTI Polo



20 docs
Golf Car Topgear Petrol GTI
Motor Bike Oil Petrol Tourer
Lupin Petrol
The co-occurrence based ranking improves the performance. Note that one of the most relevant doc does NOT include the word Golf, and a doc with a “spurious” sense disappears
rank of selected docs
Golf Car Topgear Petrol GTI
Petrol Badge
Polo Red
Car 2 *(20/3) = 13
Topgear 2 *(20/3) = 13
Petrol 3 *(20/16) = 4
Selection based on semantic domain
Golf Car Topgear Petrol GTI
Golf Car Clarkson Petrol Badge
Golf Petrol Topgear Polo Red
Golf Tiger Woods Belfry Tee
Car Wheel Topgear GTI Polo
Rank
30
17
17
0
26

Ranking with latent Semantic Indexing

- Previous example just gives the intuition
- Latent Semantic Indexing is an **algebraic** method to identify clusters of co-occurring terms, called "latent topics", and to compute query-doc similarity in a **latent space**, in which every coordinate is a latent topic.
- A "latent" quantity is one which cannot directly observed, what is observed is a measurement which may include some amount of random errors (topics are "latent" in this sense: we observe them, but they are an approximation of "true" semantic topics)
- Since it is an algebraic method, needs some linear algebra background

The LSI method: how to detect “topics”

Linear Algebra Background

Eigenvalues & Eigenvectors

- **Eigenvectors** (for a square $m \times m$ matrix $\mathbf{S}$)

$$\mathbf{A}\underline{\mathbf{v}} = \lambda \underline{\mathbf{v}}$$

(right) eigenvector $\mathbf{v} \in \mathbb{R}^m \neq \mathbf{0}$

eigenvalue $\lambda \in \mathbb{R}$

Example

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$\mathbf{A}\underline{\mathbf{v}} = \lambda \underline{\mathbf{v}}$$

- Def: A vector $\underline{v} \in R^n$, $\underline{v} \neq \underline{0}$, is an **eigenvector** of a matrix *mxm* A with corresponding **eigenvalue** λ, if:

  $\mathbf{A}\underline{\mathbf{v}} = \lambda \underline{\mathbf{v}}$

Algebraic method

- How many eigenvalues are there at most?

$$\mathbf{A}\underline{\mathbf{v}} = \lambda\underline{\mathbf{v}}$$

equation has a non-zero solution if $|\mathrm{A} - \lambda\mathbf{I}| = 0$

$$\mathrm{A}\mathbf{v} = \lambda\mathbf{v} \iff (\mathrm{A} - \lambda\mathbf{I})\,\mathbf{v} = \mathbf{0}$$

Where I is the identity matrix
this is a m-th order equation in λ which can have **at most m distinct solutions** (roots of the characteristic polynomial) – <u>can be complex even though **A** is real.</u>

Example of eigenvector/eigenvalues

$$A = \begin{pmatrix} 1 & -1 \\ 3 & 5 \end{pmatrix}, \quad v = \begin{pmatrix} 1 \\ -3 \end{pmatrix}, \quad \lambda = 4$$

$$A\underline{v} = \lambda\underline{v}$$

$$\begin{pmatrix} 1 & -1 \\ 3 & 5 \end{pmatrix}\begin{pmatrix} 1 \\ -3 \end{pmatrix} = 4\begin{pmatrix} 1 \\ -3 \end{pmatrix}$$

$$\begin{pmatrix} 1 + -3(-1) \\ 3 + 5(-3) \end{pmatrix} = \begin{pmatrix} 4 \\ -12 \end{pmatrix}$$

$$\begin{pmatrix} 4 \\ -12 \end{pmatrix} = \begin{pmatrix} 4 \\ -12 \end{pmatrix}$$

Example of computation

remember

$$\det M = |M| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

$$A = \begin{pmatrix} 1 & -1 \\ 3 & 5 \end{pmatrix}$$

$$\det(A - \lambda I) = 0$$

$$\left| \begin{pmatrix} 1 & -1 \\ 3 & 5 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 0$$

$$\left| \begin{pmatrix} 1 & -1 \\ 3 & 5 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} \right| = 0$$

$$\left| \begin{pmatrix} 1-\lambda & -1 \\ 3 & 5-\lambda \end{pmatrix} \right| = 0$$

$$(1-\lambda)(\lambda - 5) + 3 = 0$$

$$\lambda^2 - 6\lambda + 5 + 3 = 0$$

$$\lambda^2 - 6\lambda + 8 = 0$$

$$(\lambda - 4)(\lambda - 2) = 0$$

$$\boxed{\lambda_1 = 2 \quad \lambda_2 = 4}$$

Characteristic polynomial

2 and 4 are the eigenvalues of S

$$(A - \lambda I)\underline{v} = \underline{0}$$

$$\begin{pmatrix} 1-4 & -1 \\ 3 & 5-4 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} -3 & -1 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{cases} -3\alpha - \beta = 0 \\ 3\alpha + \beta = 0 \end{cases} \qquad \beta = -3\alpha$$

$$v = \begin{pmatrix} \alpha \\ -3\alpha \end{pmatrix}$$

$$(A - \lambda I)\underline{v} = \underline{0}$$

$$\begin{pmatrix} 1-2 & -1 \\ 3 & 5-2 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

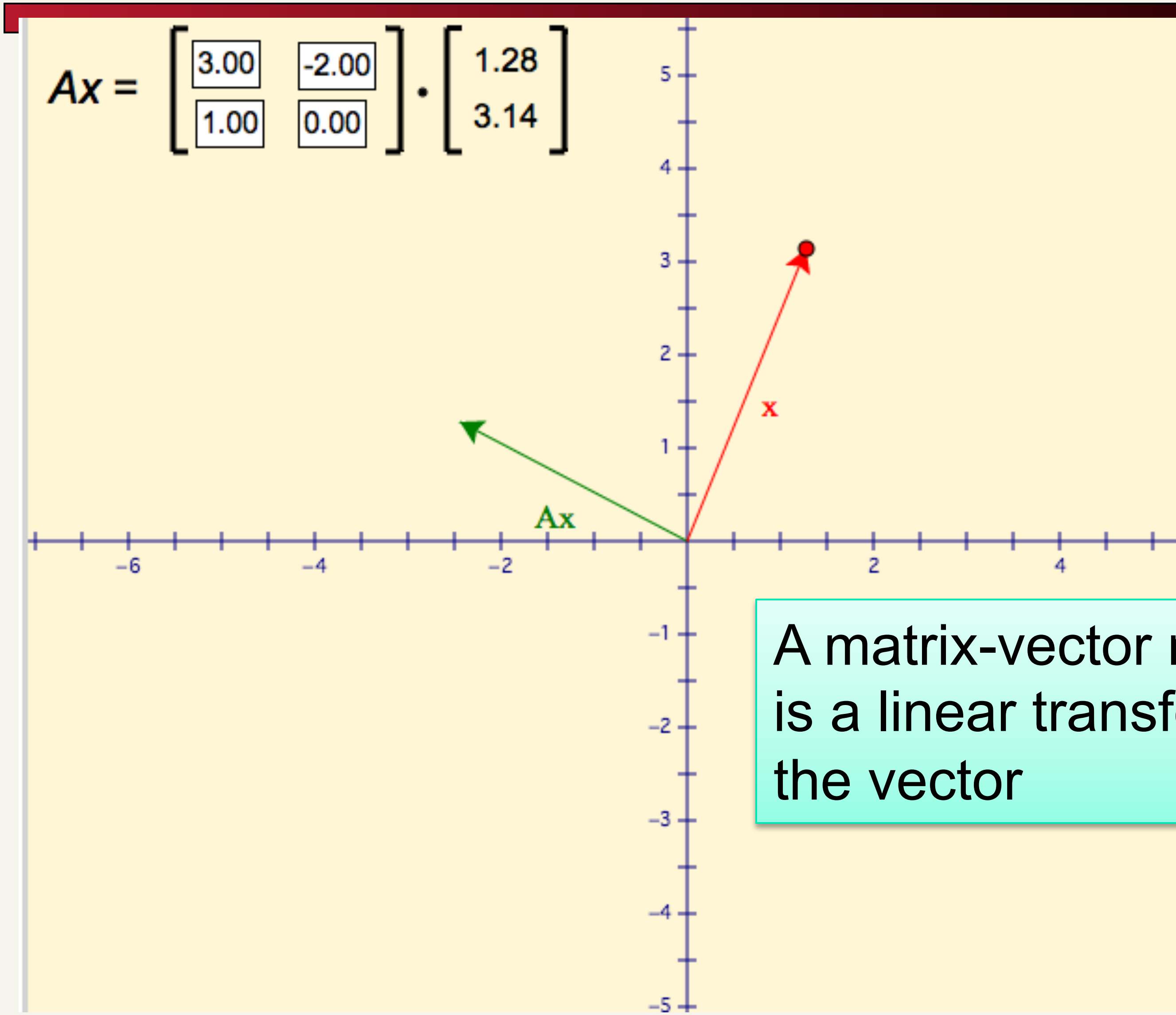
$$\begin{pmatrix} -1 & -1 \\ 3 & 3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{cases} -\alpha - \beta = 0 \\ 3\alpha + 3\beta = 0 \end{cases} \qquad \alpha = -\beta$$

$$v = \begin{pmatrix} -\beta \\ \beta \end{pmatrix}$$

Note that we compute only the **DIRECTION** of eigenvectors

Geometric interpretation



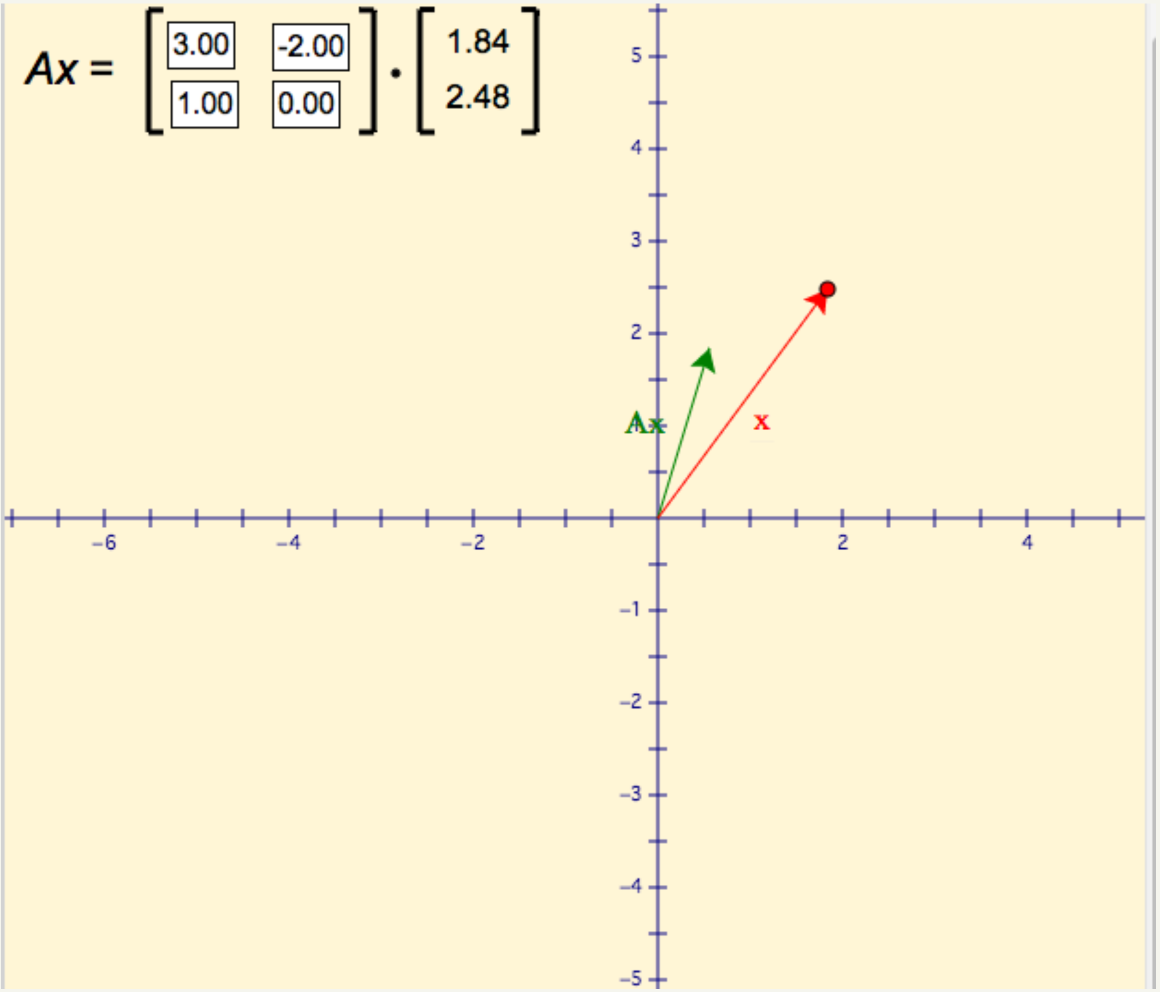
A matrix-vector multiplication Ax is a linear transformation over the vector

Matrix vector multiplication

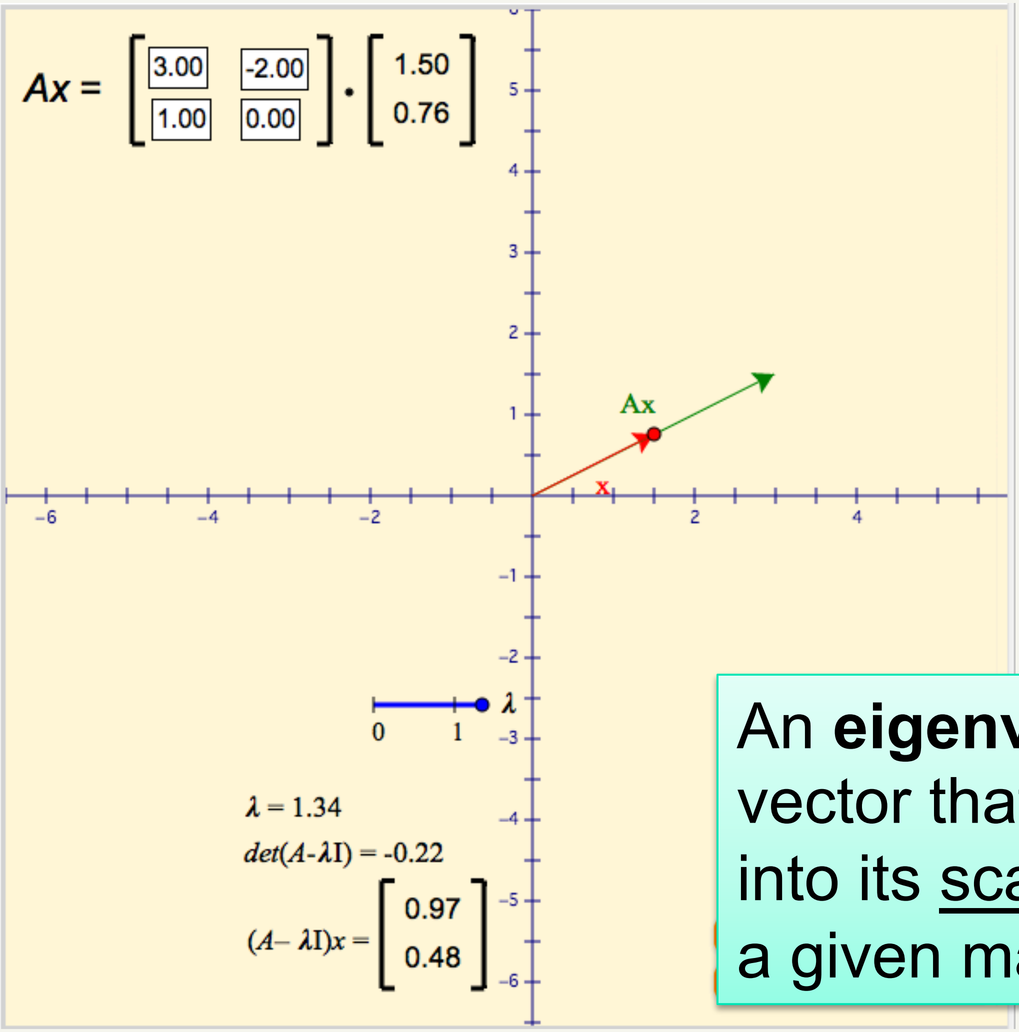
$$A\mathbf{x} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix}.$$

Matrix multiplication by a vector = a linear transformation of the initial vector, that implies **rotation** and **translation**

Geometric interpretation



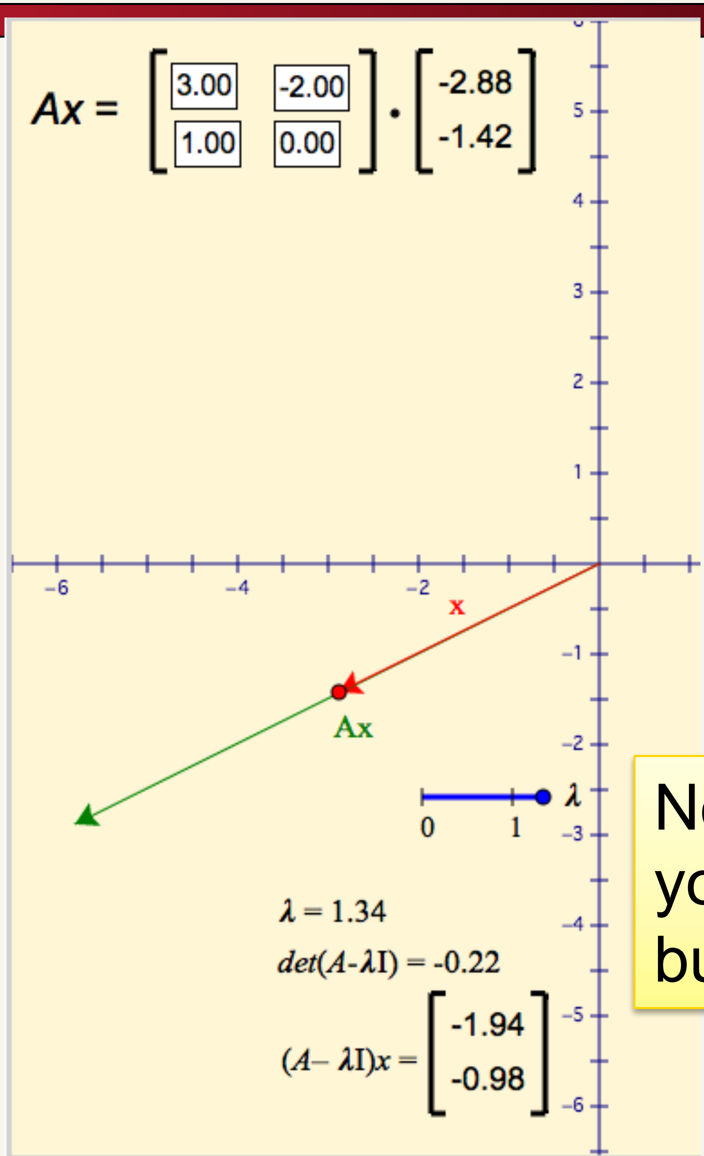
Geometric interpretation



Ax=1.34x=λx

An **eigenvector** is a special vector that is transformed into its scalar multiple under a given matrix (no rotation!)

Geometric interpretation



Here we found another eigenvector for the matrix A

Note that for any single eigenvalue you have infinite eigenvectors, but they have **the same direction**

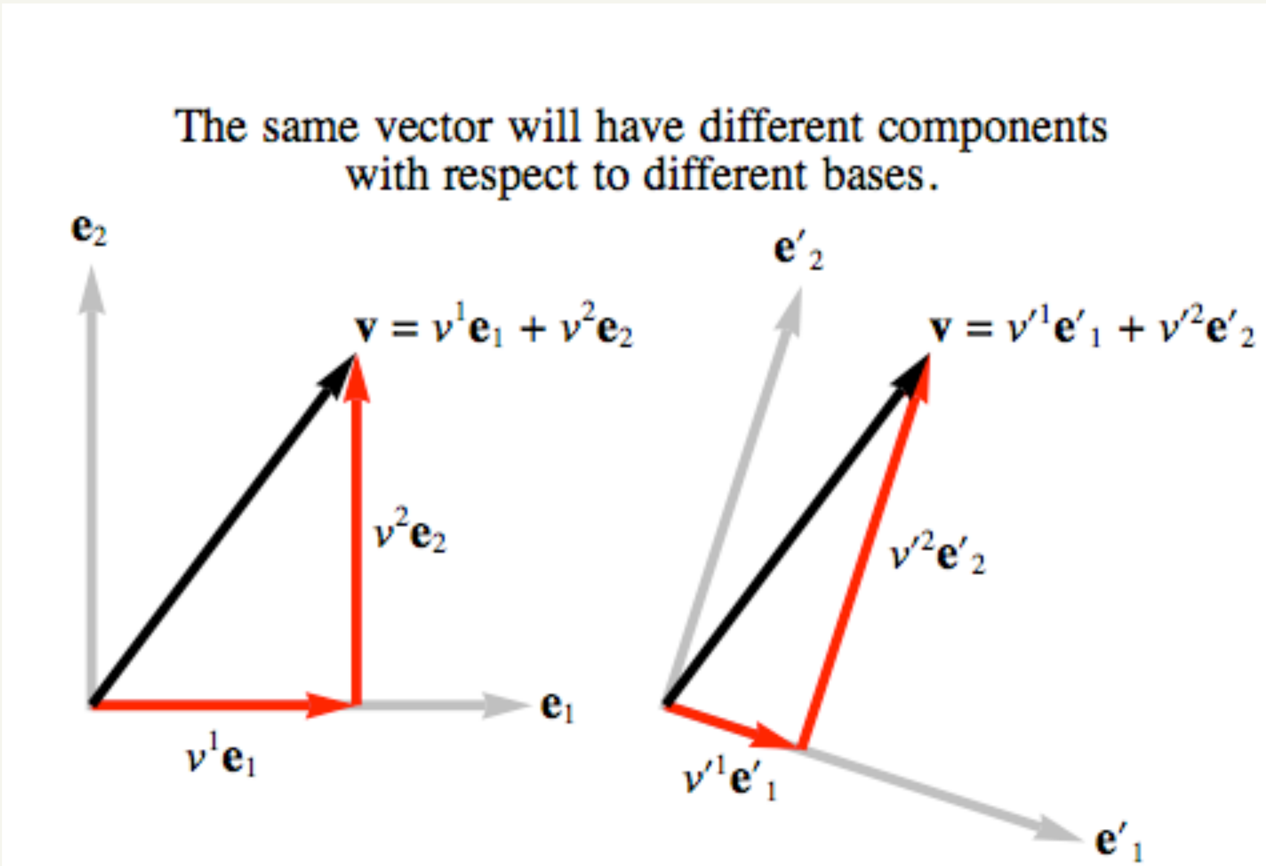
Matrix-vector multiplication

- Eigenvectors of <u>different</u> eigenvalues are **linearly independent** (i.e. $\forall \alpha_1 .. \alpha_n \rightarrow \alpha_1 v_1 + .. \alpha_n v_n \neq 0$)
- For **square normal** matrixes eigenvectors of different eigenvalues define an orthonormal space and they are othogonal.
- A square matrix is NORMAL iff it commutes with its transpose, i.e. $AA^T = A^T A$
- Example:
- $A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$ $\rightarrow AA^T = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} = A^T A$

Difference between orthonormal and orthogonal?

- Orthogonal mean that the dot product is null (the cosin of the angle is zero).
  Orthonormal mean that the dot product is null **and** the norm of the vectos is equal to 1.
  What we are actually saying is that eigenvectors define a set of DIRECTIONS wich are orthogonal.
- If two or more vectors are orthonormal they are also orthogonal but the inverse is not true.

Example: projecting a vector on 2 orthonormal spaces (or “bases”)



$\mathbf{e}_1$,$\mathbf{e}'_1$, $\mathbf{e}_2$,$\mathbf{e}'_2$ are unary vectors and v_1,v'_1, v_2,v'_2 are the coordinates of **v** along the directions of $\mathbf{e}_1$,$\mathbf{e}'_1$, $\mathbf{e}_2$,$\mathbf{e}'_2$

The effect of a matrix-vector multiplication is governed by eigenvectors and eigenvalues

- A matrix-vector multiplication such as *Ax* (*A normal matrix*, *x* a ***generic* vector** as in the previous slide) can be rewritten in terms of the eigenvalues/vectors **of A**.
  Example:

$$x = \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$$

$$Ax = A(2v_1 + 4v_2 + 6v_3)$$

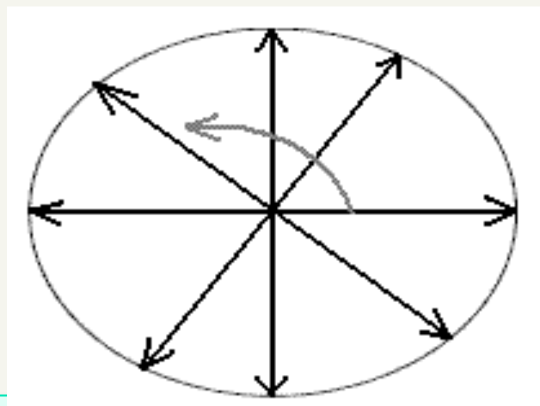
$$Ax = 2Av_1 + 4Av_2 + 6Av_3 = 2\lambda_1 v_1 + 4\lambda_2 v_2 + 6\lambda_3 v_3$$

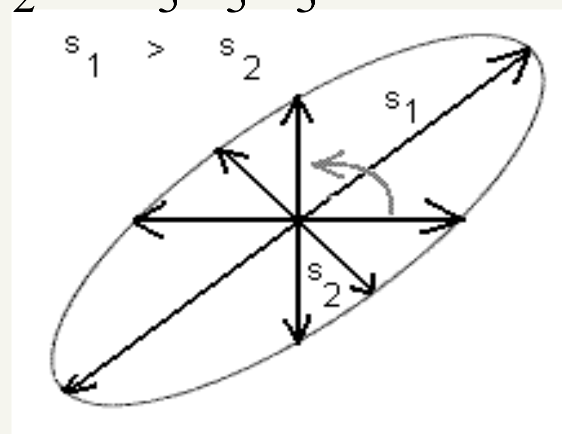
- Where v_1, v_2 v_3 are the (orthogonal) eigenvectors of A
- Even though *x* is an arbitrary vector, the action of *A* on *x* (transformation) is determined by the eigenvalues/vectors.
- Why?

Geometric explanation

x is a generic vector with coordinates x_i; λ_i, v_i are the eigenvalues and eigenvectors of A

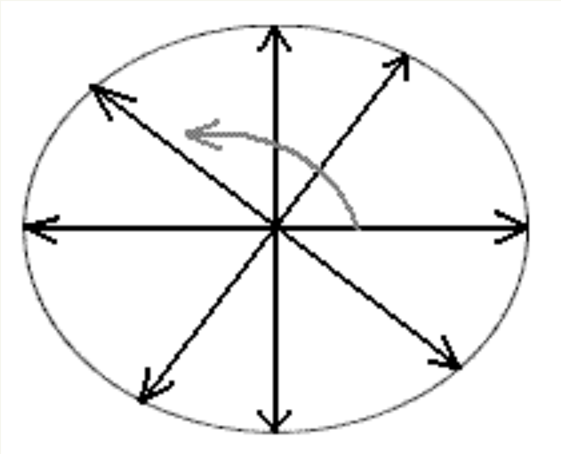
$$\boldsymbol{Ax} = \boldsymbol{x}_1\lambda_1\boldsymbol{v}_1 + \boldsymbol{x}_2\lambda_2\boldsymbol{v}_2 + \boldsymbol{x}_3\lambda_3\boldsymbol{v}_3$$

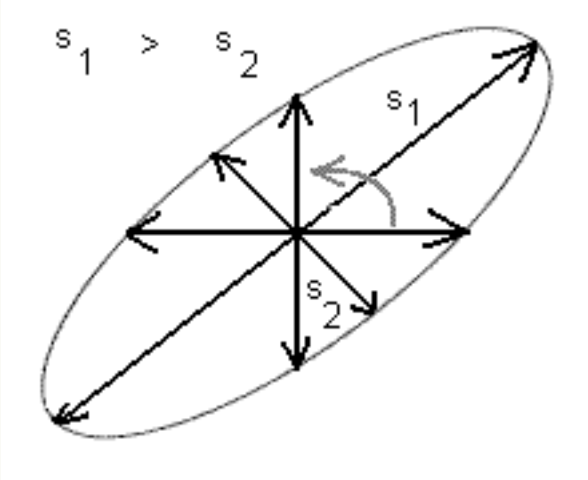




Multiplying a matrix and a vector has two effects over the vector: **rotation** (the coordinates of the vector change) and **scaling** (the length changes). The max compression and rotation **depends on the matrix's eigenvalues** λi (s1,s2 and s3 in the picture)

Geometric explanation





In the distortion, the max effect **is played by the biggest eigenvalues** (s1 and s2 in the picture)

The eigenvalues describe the distorsion operated by the matrix on the original vector

Summary so far

- A matrix A has eigenvectors v and eigenvalues λ, defined by A**v**=λ**v**
- Eigenvalues can be computed as:

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{v} \iff (\mathbf{A} - \lambda \mathbf{I})\,\mathbf{v} = \mathbf{0}$$

- We can compute only the the direction of eigenvectors, since for any eigenvalue there are infinite eigenvectors lying on the same direction
- If A is normal (i.e. if $AA^T=A^TA$) then the eigenvector form an othonormal basis
- The product of A by ANY vector x is a linear transformation of x where the **rotation** is determined by eigenvectors and the **translation** is determined by the eigenvalues. The biggest role in this transformation is played by the biggest eigenvalues.

Eigen/diagonal Decomposition

- Let A be a **square** matrix with ***m*** **orthogonal eigenvectors (hence, A is normal)**
- **Theorem**: Exists an **eigen decomposition**
  - $A=U\Lambda U^{-1}$
  - Λ is a diagonal matrix (all zero except the diagonal cells)

$$\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_m), \quad \lambda_i \geq \lambda_{i+1}$$

- Columns of ***U*** are **eigenvectors** of *A*
- Diagonal elements of Λ are **eigenvalues** of A

Diagonal decomposition: why/how

Let ***U*** have the eigenvectors as columns: $U = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}$

Then, ***AU*** can be written

$$AU = A\begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} = \begin{bmatrix} \lambda_1 v_1 & \dots & \lambda_n v_n \end{bmatrix} = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}\begin{bmatrix} \lambda_1 & & \\ & \dots & \\ & & \lambda_n \end{bmatrix}$$

Thus $AU=U\Lambda$, or $U^{-1}AU=\Lambda$

And $A=U\Lambda U^{-1}$.

Example

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

AU=UΛ

$$A\,[\mathbf{v}_1 \mathbf{v}_2] = \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} v_{11} & v_{21} \\ v_{12} & v_{22} \end{bmatrix} = \begin{bmatrix} v_{11} & v_{21} \\ v_{12} & v_{22} \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = [\mathbf{v}_1 \mathbf{v}_2] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

From this we compute $\lambda_1=1$, $\lambda_2=3$

$$\begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = 1 \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix}$$

from which we get $v_{11}=-2v_{12}$

$$\begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = 3 \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix}$$

From which we get $v_{21}=0$ and v_{22} any real

Diagonal decomposition – example 2

Recall $A=\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$; $\lambda_1=1, \lambda_2=3.$

The eigenvectors $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ form $U=\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$

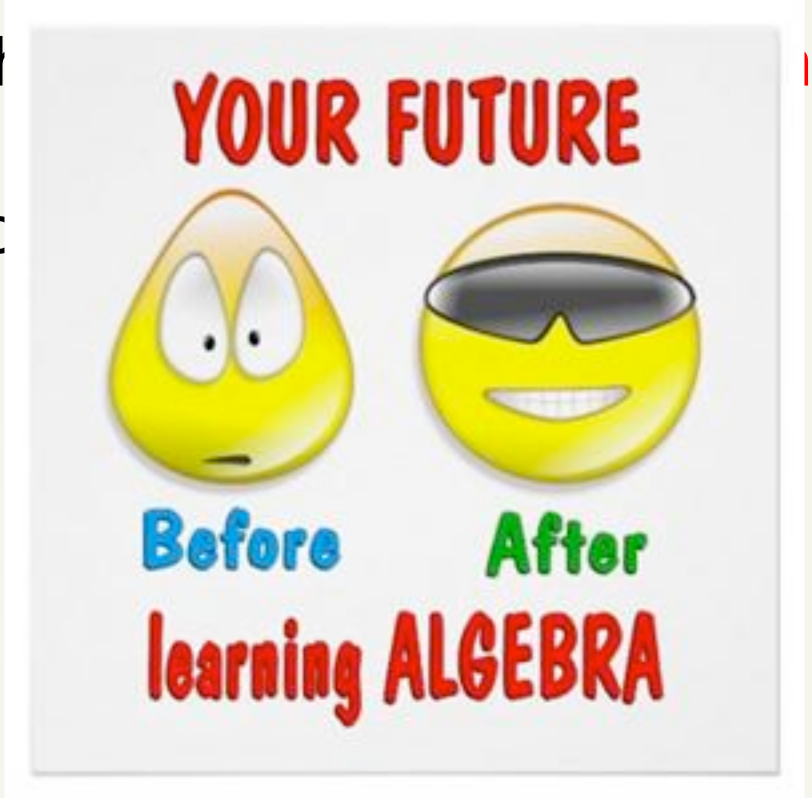
Inverting, we have $U^{-1}=\begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$

Recall $UU^{-1}=1.$

Then, ***A=UΛU*** $^{-1}$ **=** $\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}\begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$

So what?

- What do these matrices have to do with Information Retrieval and document ranking?
- Recall $M \times N$ term-document matrices …
- But everyth ormal matrices – and learn one last no



Singular Value Decomposition for non-square matrixes

For a non-square $M \times N$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

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

$M\times M$ | $M\times N$ | V is $N\times N$

The columns of ***U*** are orthogonal eigenvectors of $\boldsymbol{AA^T}$ ***(left singular vectors)***.

The columns of ***V*** are orthogonal eigenvectors of $\boldsymbol{A^TA}$ (right singular eigenvector). NOTE THAT $\boldsymbol{AA^T}$ ***and*** $\boldsymbol{A^TA}$ are square symmetric (and hence **NORMAL**)

Eigenvalues $\lambda_1 \dots \lambda_r$ of $\boldsymbol{AA^T}$ = eigenvalues of $\boldsymbol{A^TA}$ and: $\sigma_i = \sqrt{\lambda_i}$

$$\Sigma = diag\left(\sigma_1 \dots \sigma_r\right)$$

Singular values.

An example

Find the SVD of A, $U\Sigma V^T$, where $A = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$

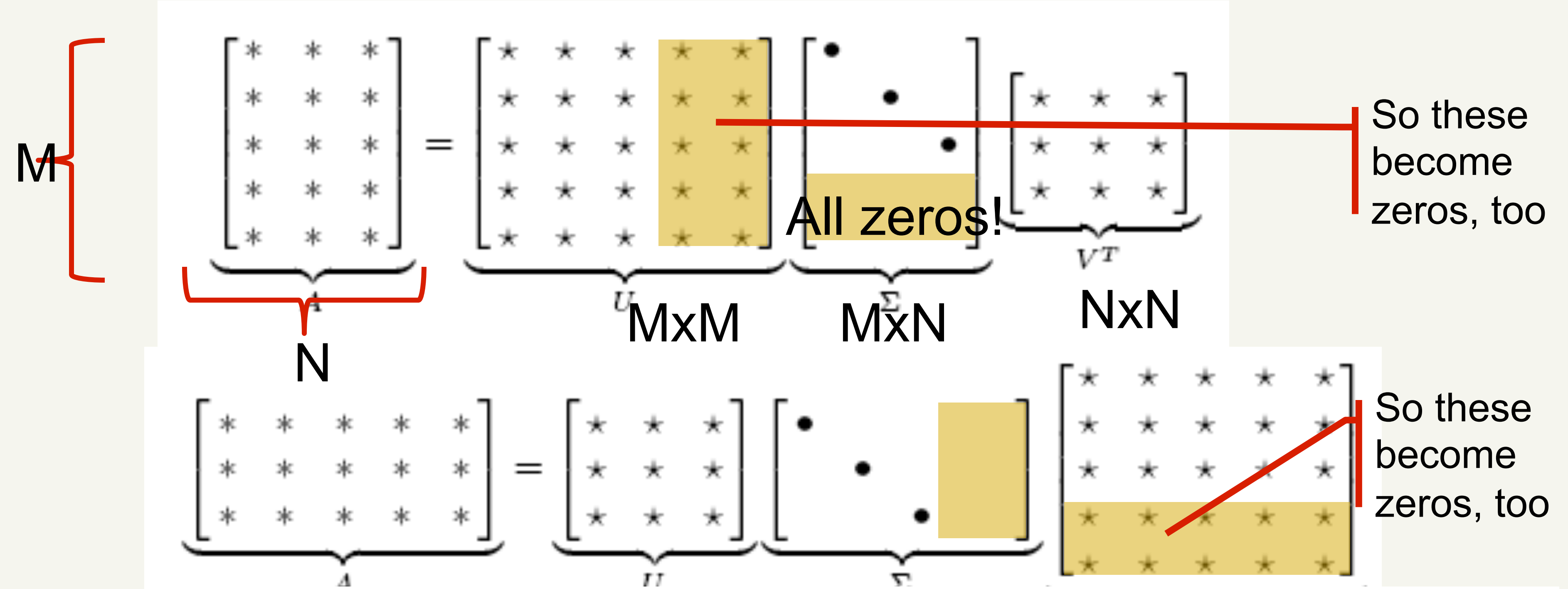
$$A = U\Sigma V^T = U \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}$$

$$A = U\Sigma V^T = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}$$

$$v_3 = \begin{pmatrix} 2/3 \\ -2/3 \\ -1/3 \end{pmatrix}$$

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness



$$\begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}$$

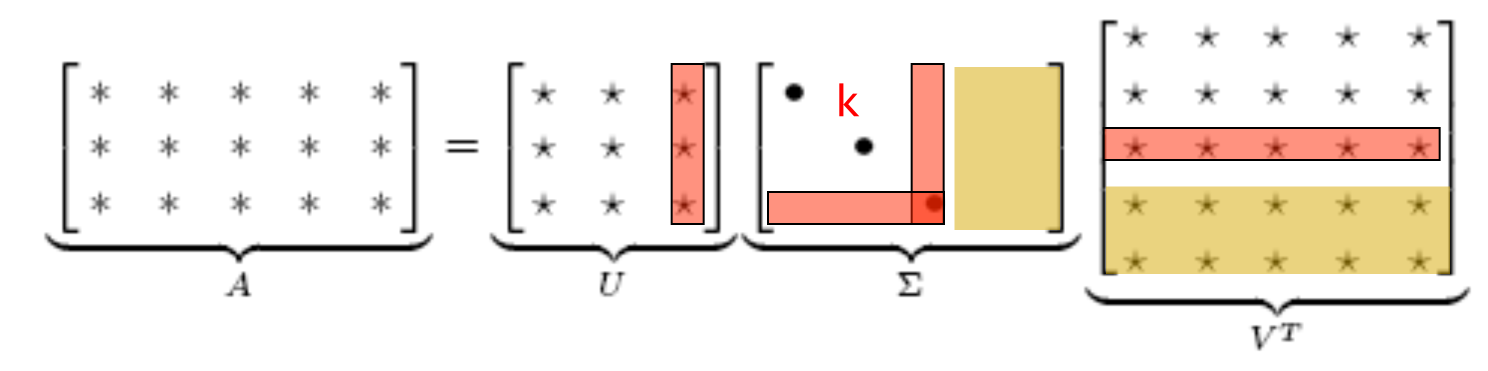
Back to matrix-vector multiplication

- Remember what we said? In a matrix vector multiplication the biggest role is played by the biggest eigenvalues
- The diagonal matrix Σ has the eigenvalues of $A^T A$ (called the singular values of A) in decreasing order along the diagonal
- We can therefore apply an approximation by setting $\sigma_i=0$ if $\sigma_i \leq \theta$ and only consider the first k singular values

Reduced SVD

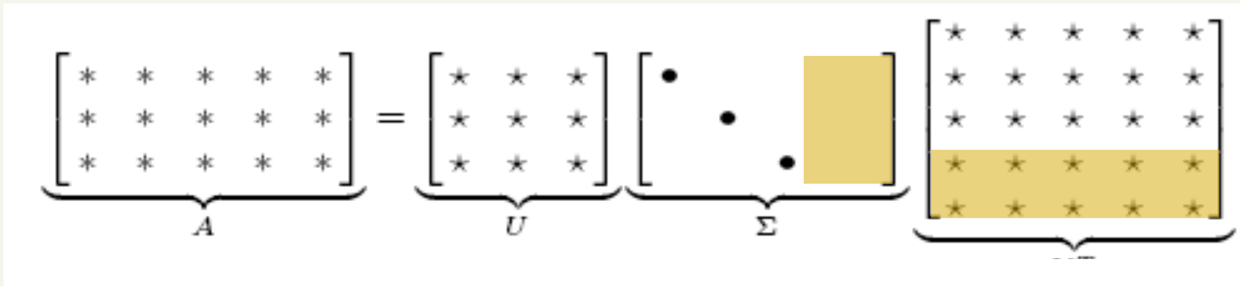
- If we retain only k singular values, and set the rest to 0, then **we don't need the matrix parts in red**
- Then Σ is $k \times k$, U is $M \times k$, V^T is $k \times N$, and A_k is $M \times N$
- This is referred to as the ***reduced SVD, or rank k approximation***

Now all the red and yellow parts are zeros!!

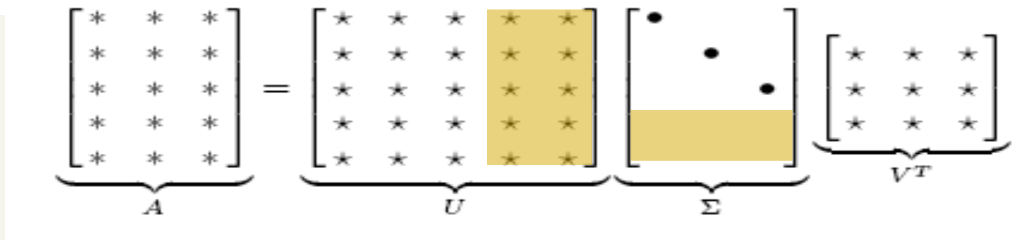


Let's recap

M<N



M>N



Since the yellow part is zero, an **exact** representation of A is:

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \ldots + \sigma_r u_r v_r^T$$

$$r = \min(M, N)$$

But "for some" k<r, a **good approximation** is:

$$A_k = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \ldots + \sigma_k u_k v_k^T$$

Example of rank approximation

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 4 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & \text{[redacted]} \\ 0 & 1 & 0 & \\ 0 & 0 & 0 & \\ 1 & 0 & 0 & \end{bmatrix} \times \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ \text{[redacted]} & & & & \end{bmatrix}$$

A

$$[A]^* = \begin{bmatrix} 001 \\ 010 \\ 000 \\ 100 \end{bmatrix} \times \begin{bmatrix} 400 \\ 030 \\ 00\sqrt{5} \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \end{bmatrix}$$

$$A^* = \begin{matrix} 0.981 & 0.000 & 0.000 & 0.000 & 1.985 \\ 0.000 & 0.000 & 3.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 4.000 & 0.000 & 0.000 & 0.000 \end{matrix} \cong A$$

Approximation error

- How good (bad) is this approximation?
- It's the best possible, measured by the Frobenius norm of the error:

$$\min_{X:rank(X)=k} \|A - X\|_F = \|A - A_k\|_F = \sigma_{k+1} \qquad \sigma_i = \sqrt{\lambda_i}$$

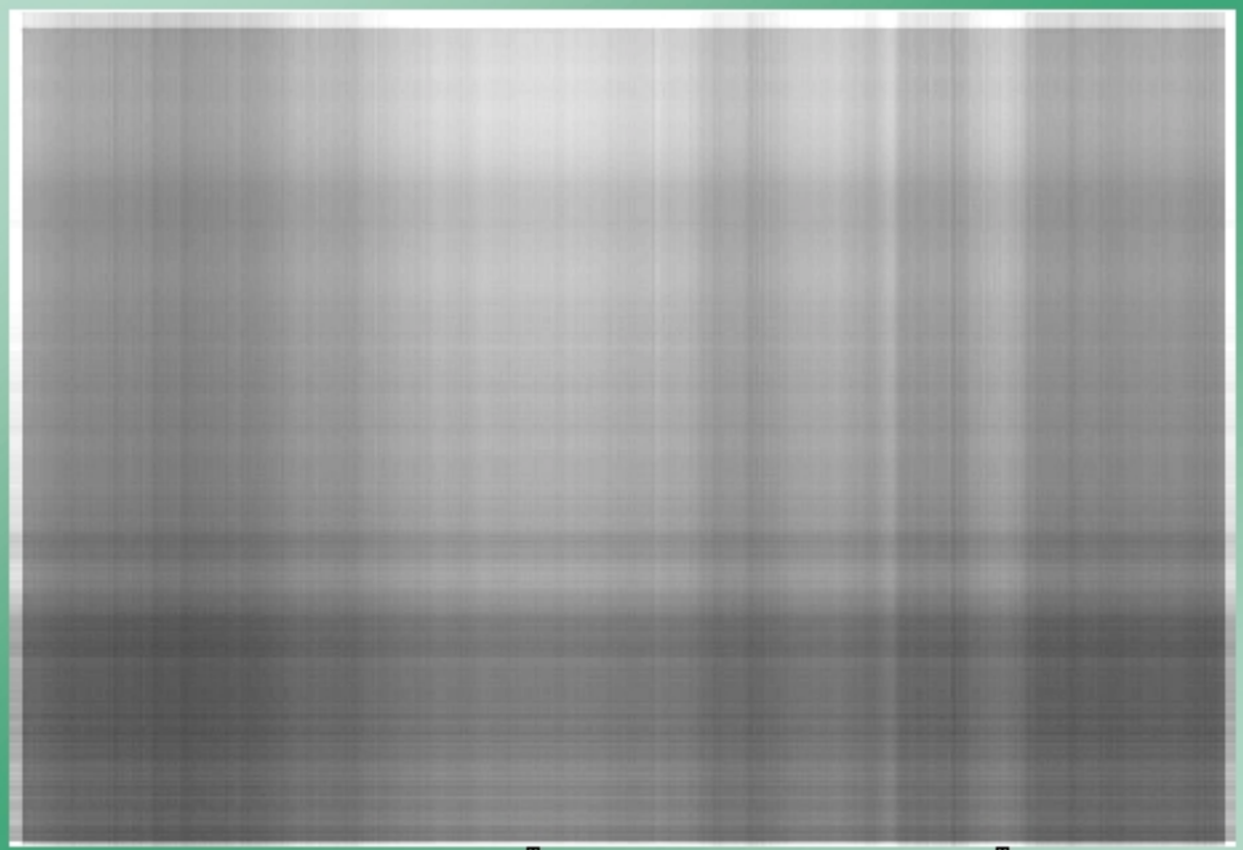
where the σ_i are ordered such that $\sigma_i \geq \sigma_{i+1}$.

- Suggests why Frobenius error drops as *k* increases.

Images gives a better intuition (image = matrix of pixels)

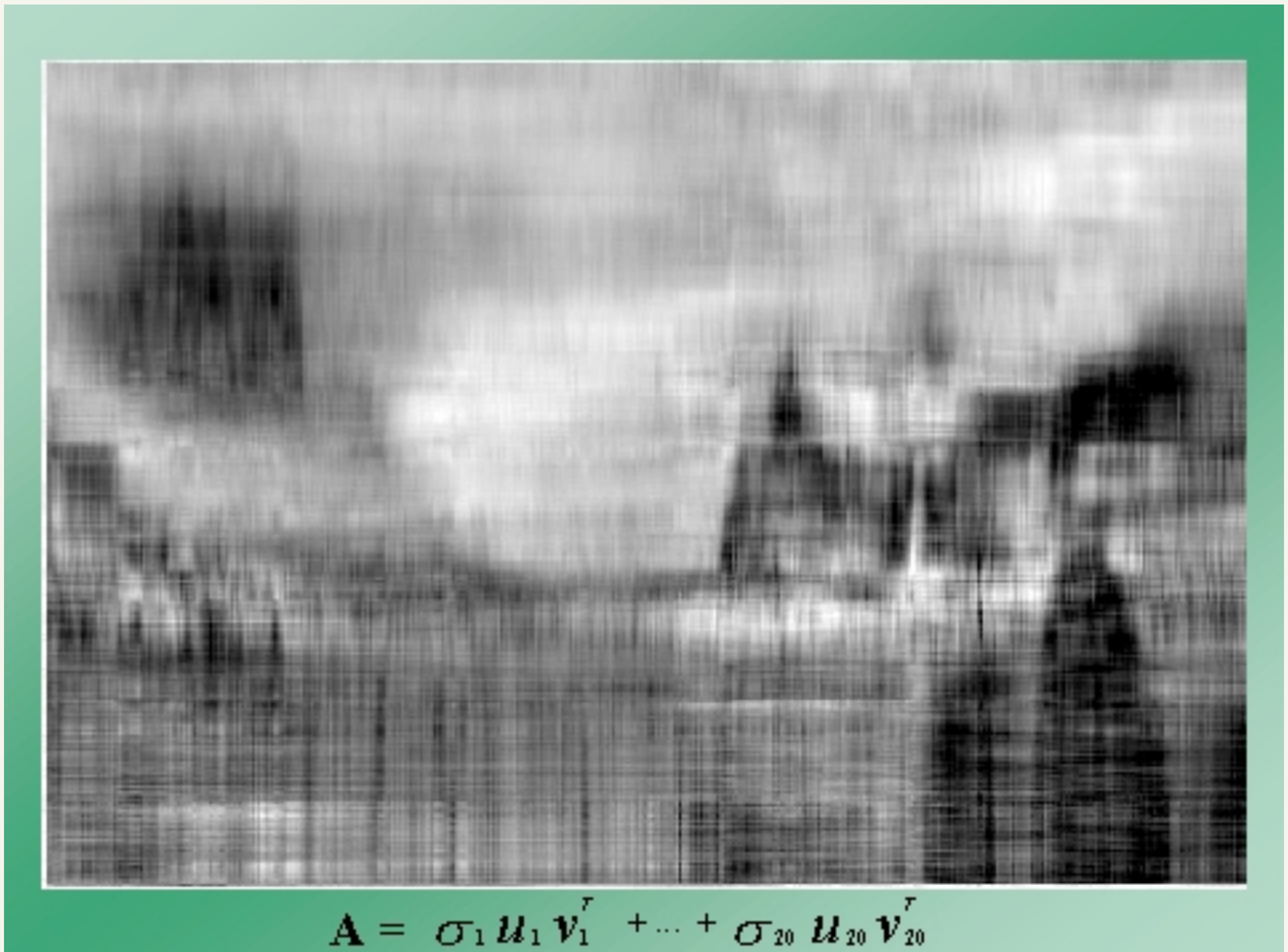
The original image

K=10

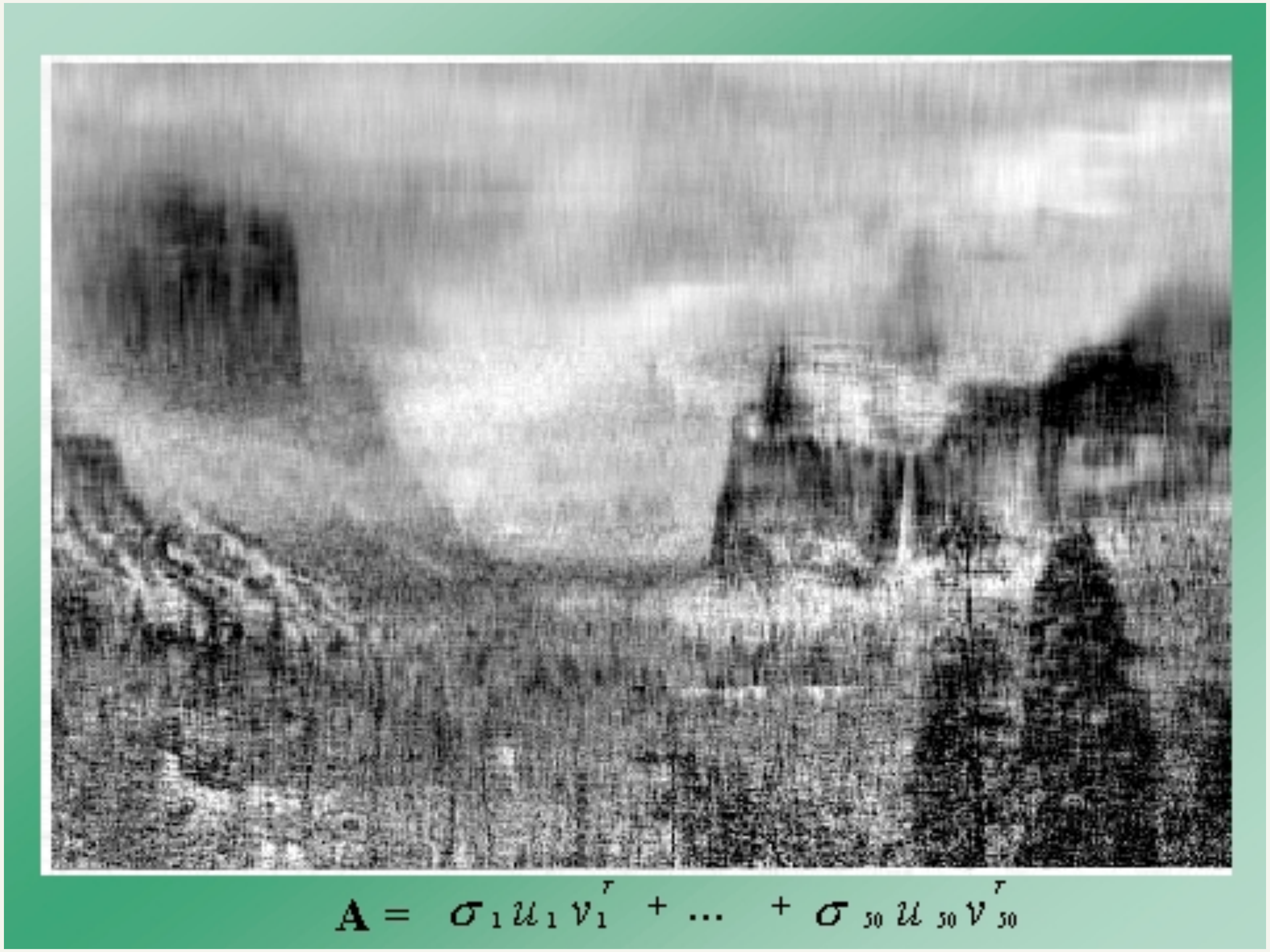


$$A = \sigma_1 u_1 v_1^T + \dots + \sigma_{10} u_{10} v_{10}^T$$

K=20



K=30



K=100

$$\mathbf{A} = \sigma_1 u_1 v_1^T + \dots + \sigma_{100} u_{100} v_{100}^T$$

K=322 (the original image)

The original image

We obtained our approximation by summing up only the first 100 terms of the singular value decomposition.

This approximation reduced the amount of information in our image by nearly **70% !!!**

We save space!! But this is only one issue

So, finally, back to IR

- Our initial problem was:
  - the term-document (MxN) matrix A is highly sparse (has many zeros)
  - However, since groups of terms tend to co-occur together, can we identify the semantic space of these clusters of terms, and apply the vector space model in the semantic space defined by such clusters?
- What we learned so far:
  - Matrix A can be decomposed and rank k approximated using SVD
  - Does this help solving our initial problems?

A is our term document matrix

- **Latent Semantic Indexing via the SVD**

$$A = \underset{M\times M}{U}\ \underset{M\times N}{\Sigma}\ \underset{V \text{ is } N\times N}{V^T}$$

The columns of $\boldsymbol{U}$ are orthogonal eigenvectors of $\boldsymbol{AA^T}$.

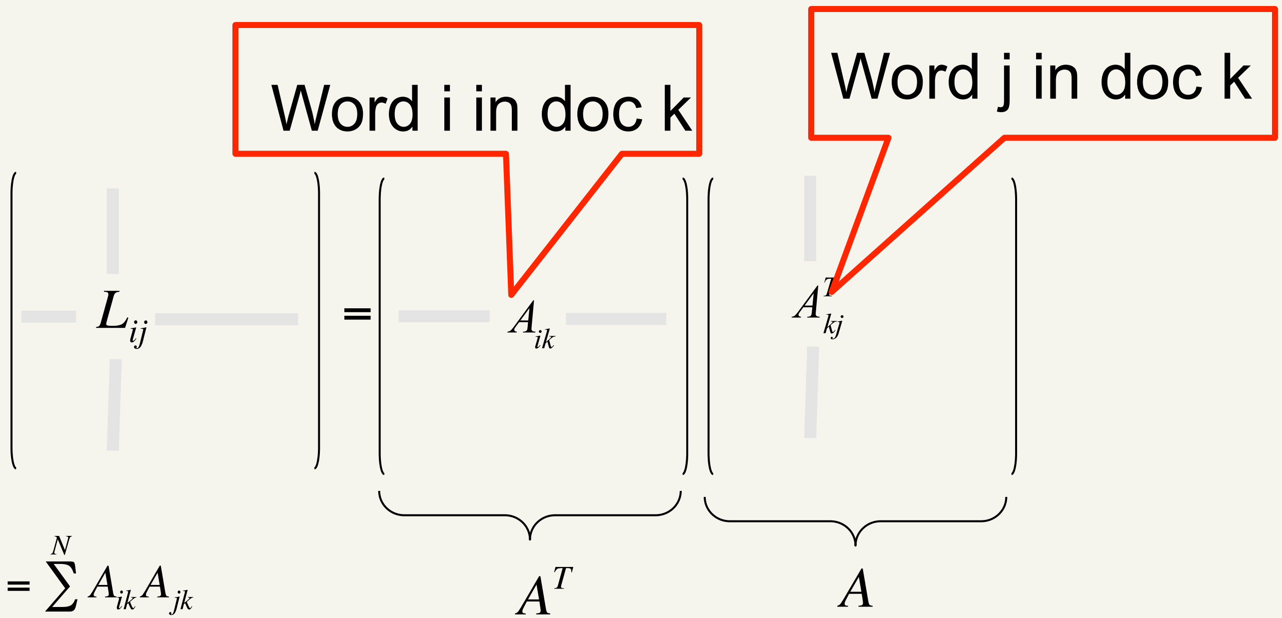
The columns of $\boldsymbol{V}$ are orthogonal eigenvectors of $\boldsymbol{A^TA}$.

Eigenvalues $\lambda_1 \dots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

- If A is a term/document matrix, then AA^T and $A^T A$ are the matrixes of term and document co-occurrences, repectively

Meaning of A^TA and AA^T

$$L = A\ A^T =$$



$$L_{ij} = \sum_{k=1}^{N} A_{ik} A^T_{kj} = \sum_{k=1}^{N} A_{ik} A_{jk}$$

L_{ij} is the number of documents in which wi and wj co-occurr

Similarly, L^T_{ij} is the number of co-occurring words in docs di and dj (or viceversa if A is a document-term matrix rather than term-document)

Example

Example of text data: Titles of Some Technical Memos

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*

Term-document matrix

A =

	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

Term co-occurrences 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

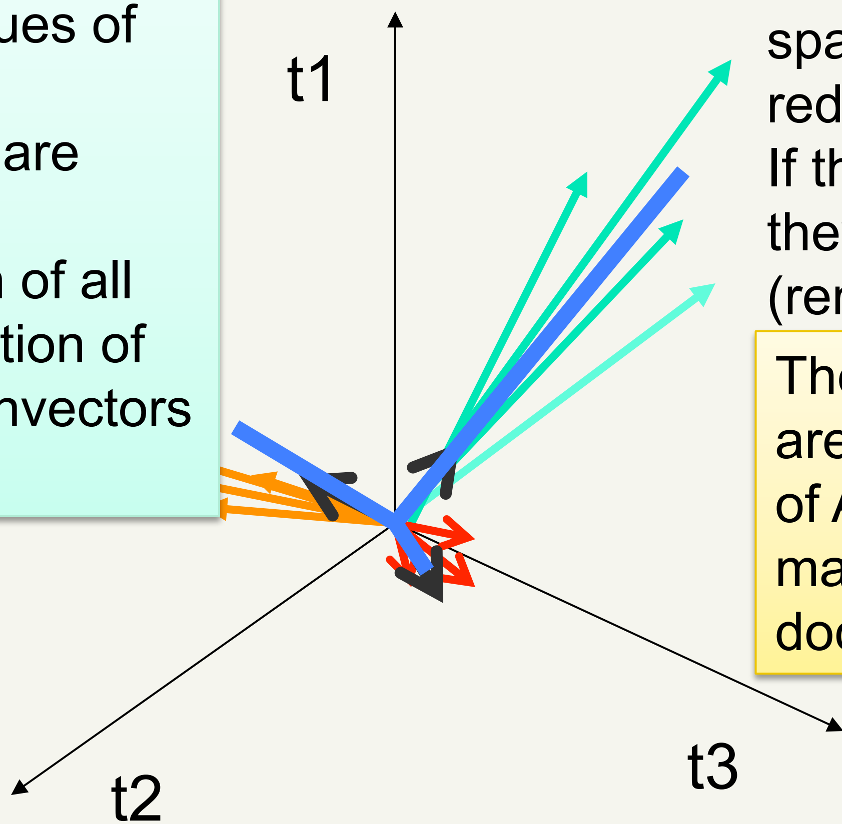
$$L_{\text{trees,graph}} = (000001110) \bullet (000000111)^T = 2$$

So the matrix L=AA^T is the matrix of term co-occurrences

- **Remember**: eigenvectors of a matrix define an orthonormal space
- **Remember**: bigger eigenvalues define the “main” directions of this space
- **But**: Matrix L and L^T are SIMILARITY matrixes (respectively, of terms and of documents). They define a SIMILARITY SPACE (the orthonormal space of their eigenvectors)
- **If the matrix elements are word co-occurrences, bigger eigenvalues are associated to bigger groups of similar words**
- **Similarly, bigger eigenvalues of L^T=A^{T}A are associated with bigger groups of similar documents (those in which co-occur the same terms)**

LSI: the intuition

The blue segments give the intuition of eigenvalues of $L^T=A^TA$
Bigger eigenvalues are those for
which the projection of all vectors on the direction of correspondent eigenvectors is higher



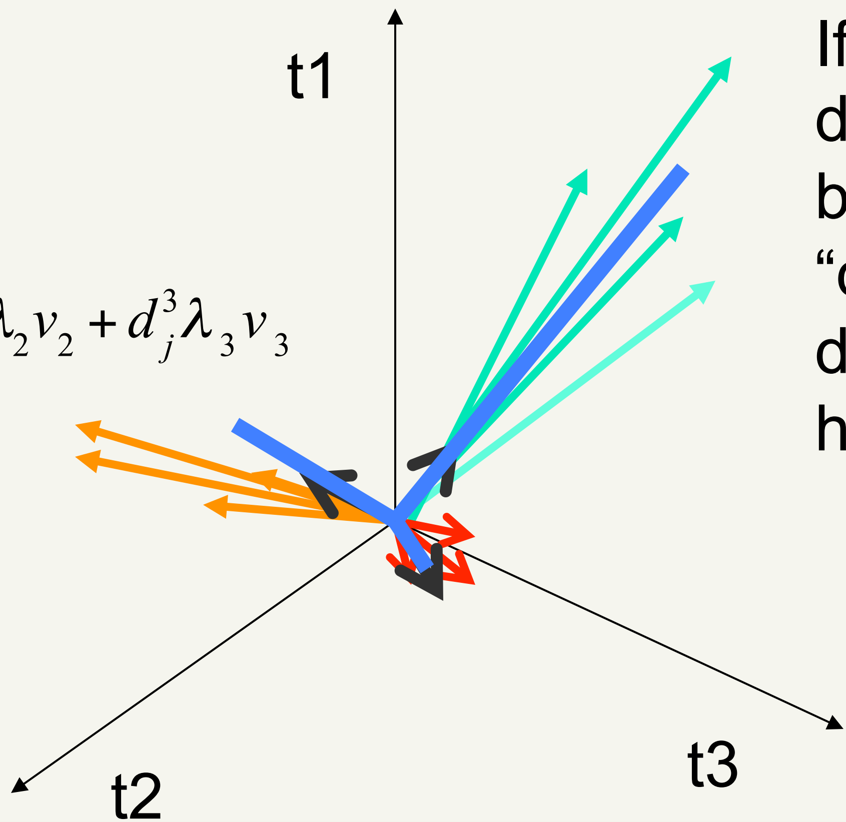
Projecting *A* in the term space: green, yellow and red vectors are documents. If they form small angles, they have common words (remember cosin-sim)

The black vector
are the unary eigenvector of A: they represent the main “directions” of the document vectors

LSI intuition

$$\vec{d}_j = (d_j^1, d_j^2, d_j^3)$$

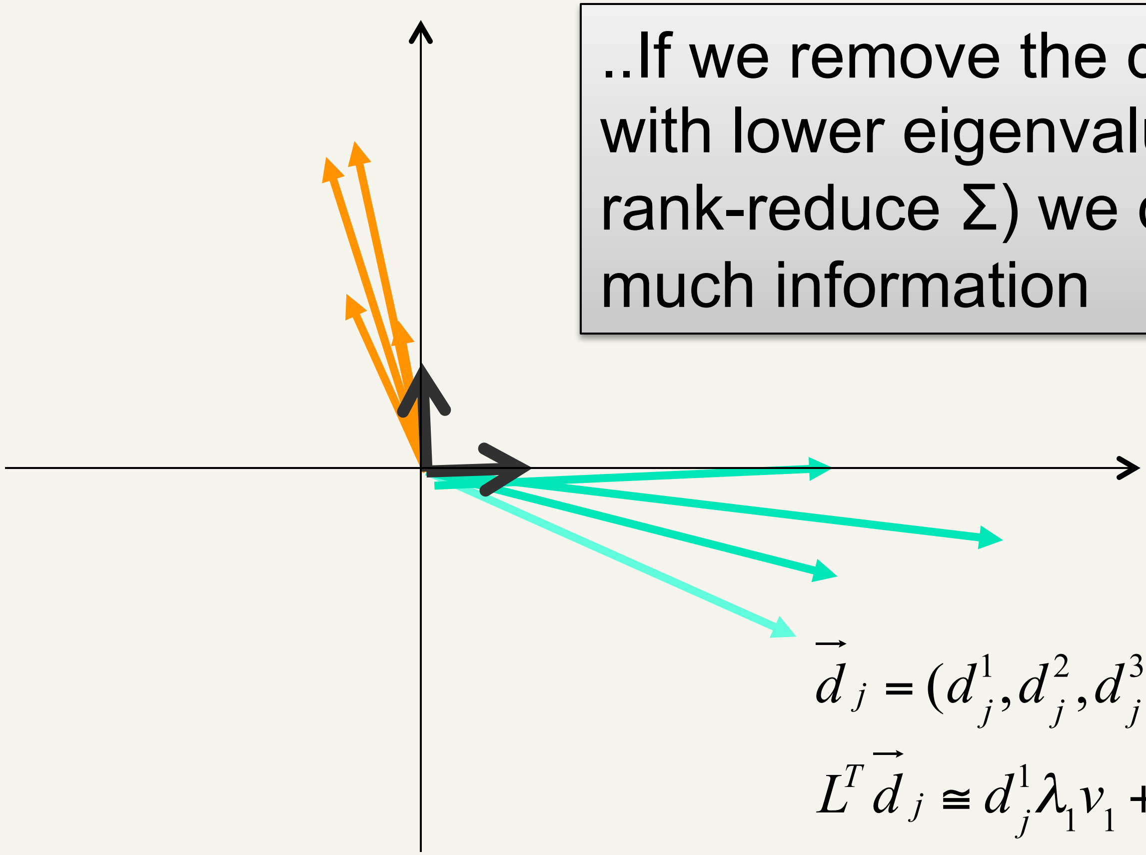
$$L^T \vec{d}_j = d_j^1 \lambda_1 v_1 + d_j^2 \lambda_2 v_2 + d_j^3 \lambda_3 v_3$$



If we multiply all document vectors by $L^T = A^T A$, their "distorsion" is mostly determined by the highest eigenvalues

We now project our document vectors on the reference orthonormal system represented by the 3 black vectors

LSI intuition



..If we remove the dimension(s) with lower eigenvalues (i.e. if we rank-reduce Σ) we don't loose much information

$$\vec{d}_j = (d_j^1, d_j^2, d_j^3)$$

$$L^T \vec{d}_j \cong d_j^1 \lambda_1 v_1 + d_j^2 \lambda_2 v_2$$

Remember that the two "new" axis represent a combination of co-occurring words e.g. a **latent semantic space**

Example

	d1	d2	d3	d4
t1	1.000	1.000	0.000	0.000
t2	1.000	0.000	0.000	0.000
t3	0.000	0.000	1.000	1.000
t4	0.000	0.000	1.000	1.000

We project terms and docs on two dimensions, **v1** and **v2** (the principal eigenvectors)

$$\begin{bmatrix} \boxed{\begin{matrix} 0.000 & -0.851 \\ 0.000 & -0.526 \\ -0.707 & 0.000 \\ -0.707 & 0.000 \end{matrix}} & \begin{matrix} -0.526 & 0.000 \\ 0.851 & 0.000 \\ 0.000 & -0.707 \\ 0.000 & 0.707 \end{matrix} \end{bmatrix} \times \begin{bmatrix} 2.000 & 0.000 \\ 0.000 & 1.618 \end{bmatrix} \times \begin{bmatrix} \boxed{\begin{matrix} 0.000 & 0.000 & -0.707 & -0.707 \\ -0.851 & -0.526 & 0.000 & 0.000 \end{matrix}} \\ \begin{matrix} 0.526 & -0.851 & 0.000 & 0.000 \\ 0.000 & 0.000 & -0.707 & 0.707 \end{matrix} \end{bmatrix}$$

Note that the direction of each eigenvector is determined by the direction of just two terms: (t1,t2) or (t3,t4)

tic

coordinates: s1:(t1,t2) and

s2:(t3,t4)

matrix

0.000 0.000 1.000 1.000
0.000 0.000 1.000 1.000

Even if t2 does not occur in d2, now if we query with t2 the system will return also d2!!

Co-occurrence space

- In principle, the space of document or term co-occurrences is much (much!) higher than the original space of terms!!
- But with SVD we consider only the most relevant ones, trough **rank reduction**

$$A = U \sum V^T \cong U_k \Sigma_k V_k^T = A_k$$

Summary so-far

- We compute the SVD rank-k approximation for the term-document matrix A
- This approximation is based on considering only the principal eigenvalues of the term co-occurrence and document similarity matrixes ($L=AA^T$ and $LT=A^TA$)
- The eigenvectors of the eigenvalues of $L=AA^T$ and $LT=A^TA$ represent the main “directions” identified by term vectors and document vectors, respectively.

LSI: what are the steps

- From term-doc matrix A, compute the approximation $A_{k.}$ with SVD
- Project docs, terms and queries in a space of *k<<r* dimensions (the k "survived" eigenvectors) and compute cos-similarity as usual
  - These dimensions are **not** the original axes, but those defined by the orthonormal space of the reduced matrix Ak

$$A\vec{q} \cong A_k \vec{q} = \sigma_1 q_1 \vec{v}_1 + \sigma_2 q_2 \vec{v}_2 + \ldots \sigma_k q_k \vec{v}_k$$

Where $\sigma_i q_i$ (i=1,2..k<<*r)* are the new coordinates of q in the orthonormal space of Ak

Projecting terms documents and queries in the LS space

$$(\mathbf{t}_i^T) \rightarrow \overset{\substack{\boxed{A} \\ (\mathbf{d}_j) \\ \downarrow}}{\begin{bmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \dots & x_{m,n} \end{bmatrix}} = (\hat{\mathbf{t}}_i^T) \rightarrow \overset{U}{\left[\begin{bmatrix} \\ \mathbf{u}_1 \\ \\ \end{bmatrix} \dots \begin{bmatrix} \\ \mathbf{u}_l \\ \\ \end{bmatrix} \right]} \overset{\Sigma}{\begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_l \end{bmatrix}} \cdot \overset{\substack{V^T \\ (\hat{\mathbf{d}}_j) \\ \downarrow}}{\begin{bmatrix} [& \mathbf{v}_1 &] \\ & \vdots & \\ [& \mathbf{v}_l &] \end{bmatrix}}$$

If $A=U\Sigma V^T$ we also have that:

- $V = A^T U \Sigma^{-1}$
- $t = t^{\wedge T} \Sigma V^T$
- $d = d^{\wedge T} U \Sigma^{-1}$
- $q = q^{\wedge T} U \Sigma^{-1}$

After rank k-approximation :

- $A \cong Ak = U_k \Sigma_k V_k^T$
- $d_k \cong d^T U_k \Sigma_k^{-1}$
- $q_k \cong q^T U_k \Sigma_k^{-1}$
- $sim(q, d) = sim(q^T U_k \Sigma_k^{-1}, d^T U_k \Sigma_k^{-1})$

Consider a term-doc matrix MxN (M=11, N=3) and a query q

$$
\begin{array}{lcccc}
\text{Terms} & & \begin{array}{ccc} d1 & d2 & d3 \end{array} & & \text{query} \\
\begin{array}{l}
\text{a} \\ \text{arrived} \\ \text{damaged} \\ \text{delivery} \\ \text{fire} \\ \text{gold} \\ \text{in} \\ \text{of} \\ \text{shipment} \\ \text{silver} \\ \text{truck}
\end{array}
& A = &
\begin{bmatrix}
1 & 1 & 1 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
1 & 0 & 0 \\
1 & 0 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 0 & 1 \\
0 & 2 & 0 \\
0 & 1 & 1
\end{bmatrix}
& \mathbf{q} = &
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1
\end{bmatrix}
\end{array}
$$

1. Compute SVD: $A = U\Sigma V^T$

$$U = \begin{bmatrix} -0.4201 & 0.0748 & -0.0460 \\ -0.2995 & -0.2001 & 0.4078 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.1576 & -0.3046 & -0.2006 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.3151 & -0.6093 & -0.4013 \\ -0.2995 & -0.2001 & 0.4078 \end{bmatrix}$$

$$S = \begin{bmatrix} 4.0989 & 0.0000 & 0.0000 \\ 0.0000 & 2.3616 & 0.0000 \\ 0.0000 & 0.0000 & 1.2737 \end{bmatrix}$$

$$V = \begin{bmatrix} -0.4945 & 0.6492 & -0.5780 \\ -0.6458 & -0.7194 & -0.2556 \\ -0.5817 & 0.2469 & 0.7750 \end{bmatrix}$$

$$V^T = \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \\ -0.5780 & -0.2556 & 0.7750 \end{bmatrix}$$

2. Obtain a low rank approximation (k=2) $A_k = U_k \Sigma_k V^T_k$

$$U = \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \qquad S = \begin{bmatrix} 4.0989 & 0.0000 \\ 0.0000 & 2.3616 \end{bmatrix}$$

$$V = \begin{bmatrix} -0.4945 & 0.6492 \\ -0.6458 & -0.7194 \\ -0.5817 & 0.2469 \end{bmatrix} \qquad V^T = \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \end{bmatrix}$$

3a. Compute doc/query similarity

- For N documents, A_k has N columns, each representing the coordinates of a document d_i projected in the k LSI dimensions
- A query is considered like a document, and is projected in the LSI space

3c. Compute the query vector

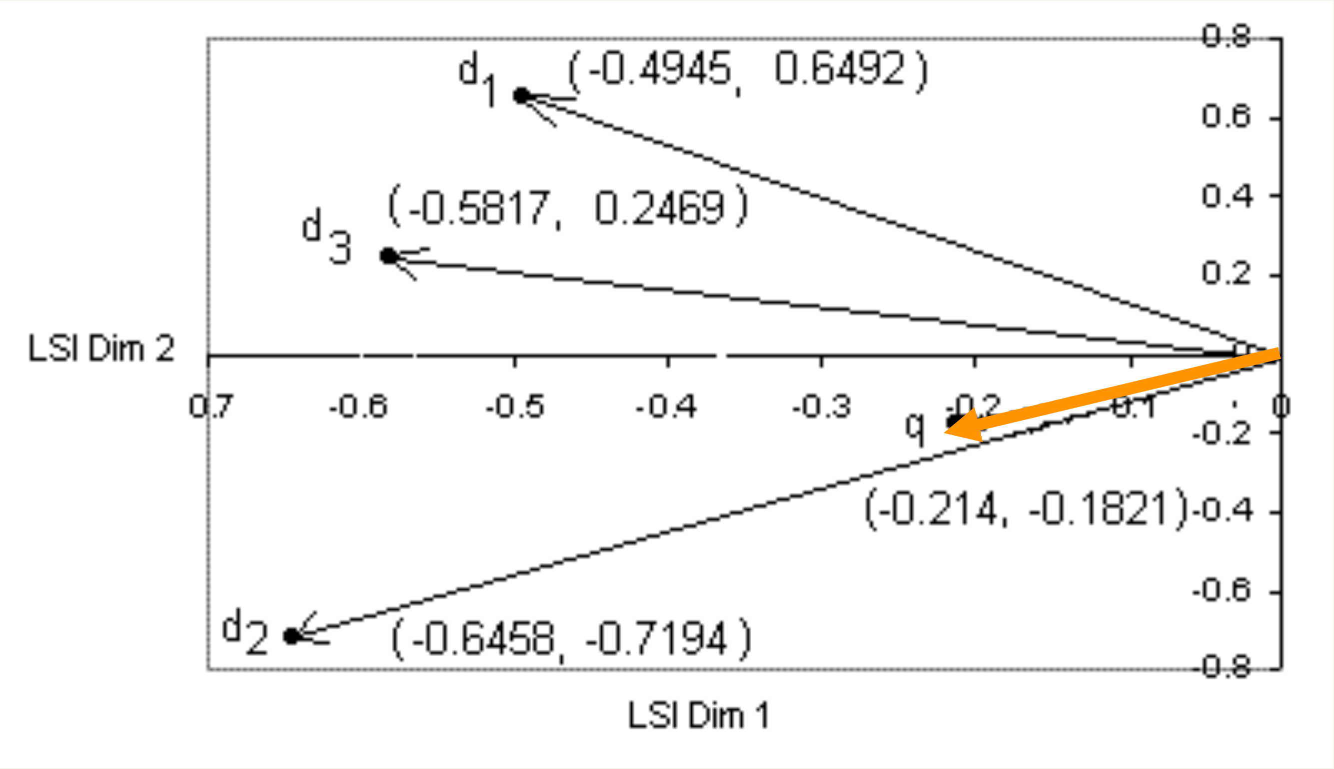
$q_k = q^T U_k \Sigma_k^{-1}$

$$\mathbf{q} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \begin{bmatrix} \frac{1}{4.0989} & 0.0000 \\ 0.0000 & \frac{1}{2.3616} \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} -0.2140 & -0.1821 \end{bmatrix}$$

q is projected in the 2-dimension LSI space!

Documents and queries projected in the LSI space



q/d similarity

$$\text{sim}(q, d) = \frac{q \bullet d}{|q||d|}$$

$$\text{sim}(q, d_1) = \frac{(-0.2140)(-0.4945) + (-0.1821)(0.6492)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.4945)^2 + (0.6492)^2}} = -0.0541$$

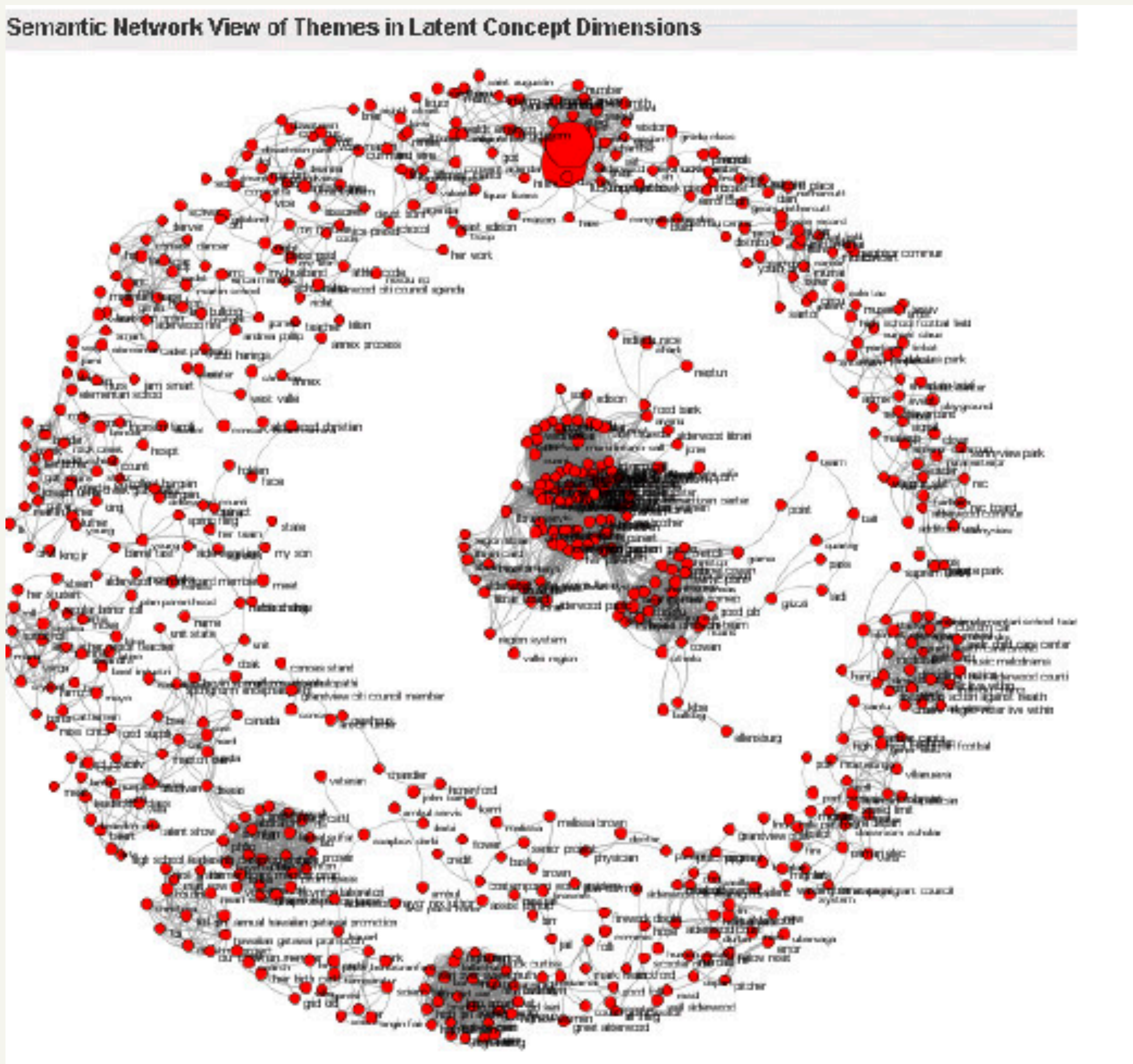
$$\text{sim}(q, d_2) = \frac{(-0.2140)(-0.6458) + (-0.1821)(-0.7194)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.6458)^2 + (-0.7194)^2}} = 0.9910$$

$$\text{sim}(q, d_3) = \frac{(-0.2140)(-0.5817) + (-0.1821)(0.2469)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.5817)^2 + (0.2469)^2}} = 0.4478$$

Ranking documents in descending order

$$d_2 > d_3 > d_1$$

An overview of a semantic network of terms based on the top 100 most significant latent semantic dimensions (Zhu&Chen)



Conclusion

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

Another LSI Example


specific
wise
management
website
sem
singapore
plan
consultant
offer
internet
company
affordable
engine
service
outsourcing
seo
web
design
blog
miami
software
consulting
optimization
expert
linkateer
professional
corporate
market
japan
faq
dakota

Input matrix A:

	d1	d2	d3
t1	1.000	1.000	0.000
t2	1.000	1.000	0.000
t3	0.000	1.000	0.000
t4	0.000	0.000	1.000

Input matrix B:

A^T

1.000	1.000	0.000	0.000
1.000	1.000	1.000	0.000
0.000	0.000	0.000	1.000

AA^T

Matrix product A*B

Term co-occurrences

t1	2.000	2.000	1.000	0.000
t2	2.000	2.000	1.000	0.000
t3	1.000	1.000	1.000	0.000
t4	0.000	0.000	0.000	1.000

Input matrix:

1.000	1.000	0.000
1.000	1.000	0.000
0.000	1.000	0.000
0.000	0.000	1.000

$A_k=U_k\Sigma_k V_k^T \approx A$

0.863	1.106	0.000
0.863	1.106	0.000
0.485	0.621	0.000
0.000	0.000	1.000

Singular Value Decomposition:

U:

-0.657	0.000	0.261
-0.657	0.000	0.261
-0.369	0.000	-0.929
0.000	-1.000	0.000

S:

2.136	0.000	0.000
0.000	1.000	0.000
0.000	0.000	0.662

V^T

-0.615	-0.788	0.000
0.000	0.000	-1.000
0.788	-0.615	0.000

Now it is like if t3 belongs to d1!