Matrix Decomposition and Latent Semantic Indexing (LSI)

Introduction to Information Retrieval
Informatics 141 / CS 121
Donald J. Patterson

Efficient Cosine Ranking

- Find the k docs in the corpus "nearest" to the query
  - the k largest query-doc cosines

CosineScore(q)

Initialize($Scores[d \in D]$)

Initialize($Magnitude[d \in D]$)

for $each\ term(t \in q)$

  do $p \leftarrow$ FetchPostingsList(t)

    $df_t \leftarrow$ GetCorpusWideStats(p)

    $\alpha_{t,q} \leftarrow$ WeightInQuery(t, q, df_t)

    for $each\ \{d, tf_{t,d}\} \in p$

      do $Scores[d] += \alpha_{t,q} \cdot$ WeightInDocument(t, q, df_t)

for $d \in Scores$

  do Normalize($Scores[d], Magnitude[d]$)

return $top\ K \in Scores$

Outline

- Introduction
- Linear Algebra Refresher

Late

Sta

Star Cluster NGC 290 - ESA & NASA

- A picture of the sky is two dimensional
- The stars are not in two dimensions
- When we take a photo of stars we are projecting them into 2-D
  - projecting can be defined mathematically
- When we see two stars that are close..
  - They may not be close in space
- When we see two stars that are far...
  - They may not far in space

Star Cluster NGC 290 - ESA & NASA

- When we see two stars that are close in a photo
  - They really are close for some applications
  - For example pointing a big telescope at them
  - Large shared telescopes order their views according to how “close” they are.

Overhead projector example

Overhead projector example

- Depending on where we put the light (and the wall) we can make things in three dimensions appear close or far away in two dimensions.
- Even though the “real” position of the 3-d objects never moved.

Mathematically speaking

- This is taking a 3-D point and projecting it into 2-D

(x, y, z) → (x, y)

(10, 10, 10) → (10, 10)

$$\begin{bmatrix} 10 \\ 10 \\ 10 \end{bmatrix} \qquad \begin{bmatrix} 10 \\ 10 \end{bmatrix}$$

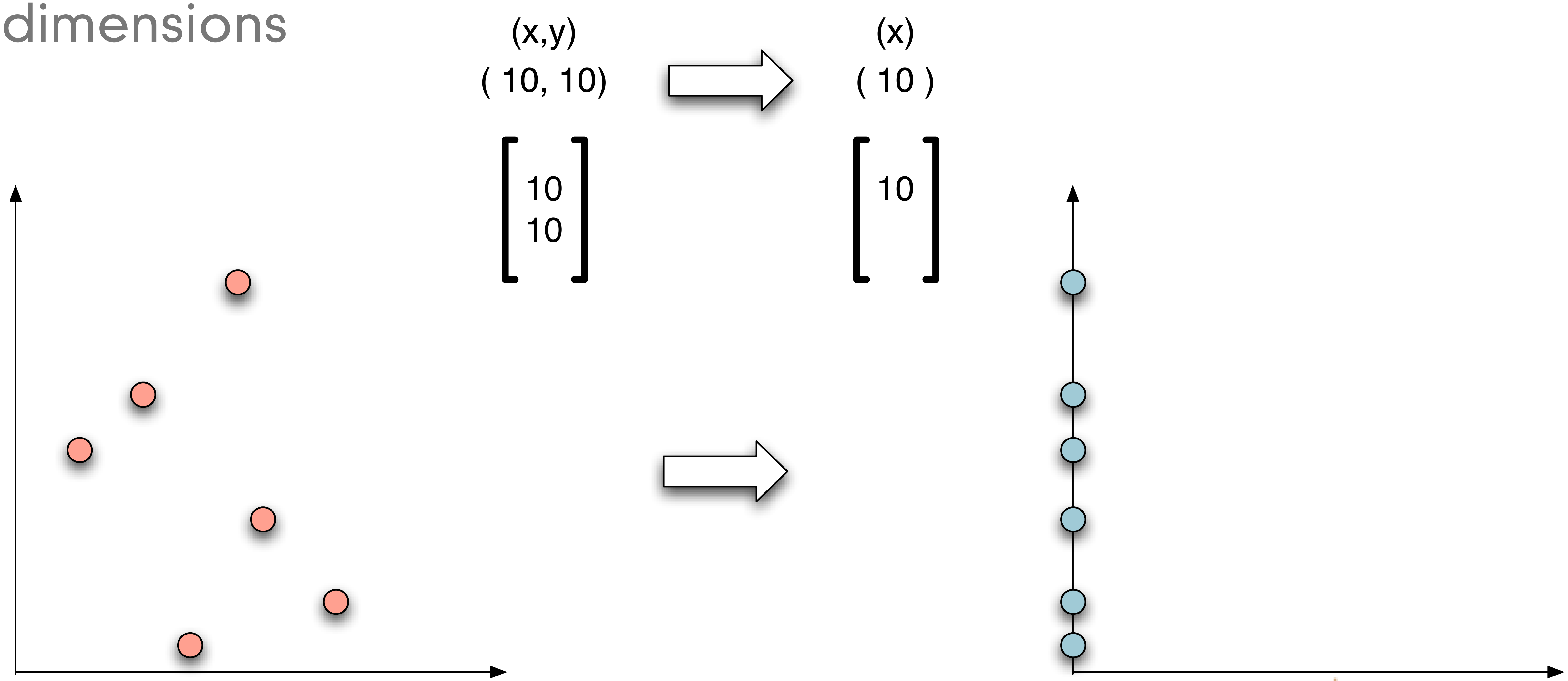
- The arrow in this picture acts like the overhead projector

Mathematically speaking

- We can project from any number of dimensions into any other number of dimensions.
- Increasing dimensions adds redundant information
  - But sometimes useful
  - Support Vector Machines (kernel methods) do this effectively
- Latent Semantic Indexing always reduces the number of dimensions

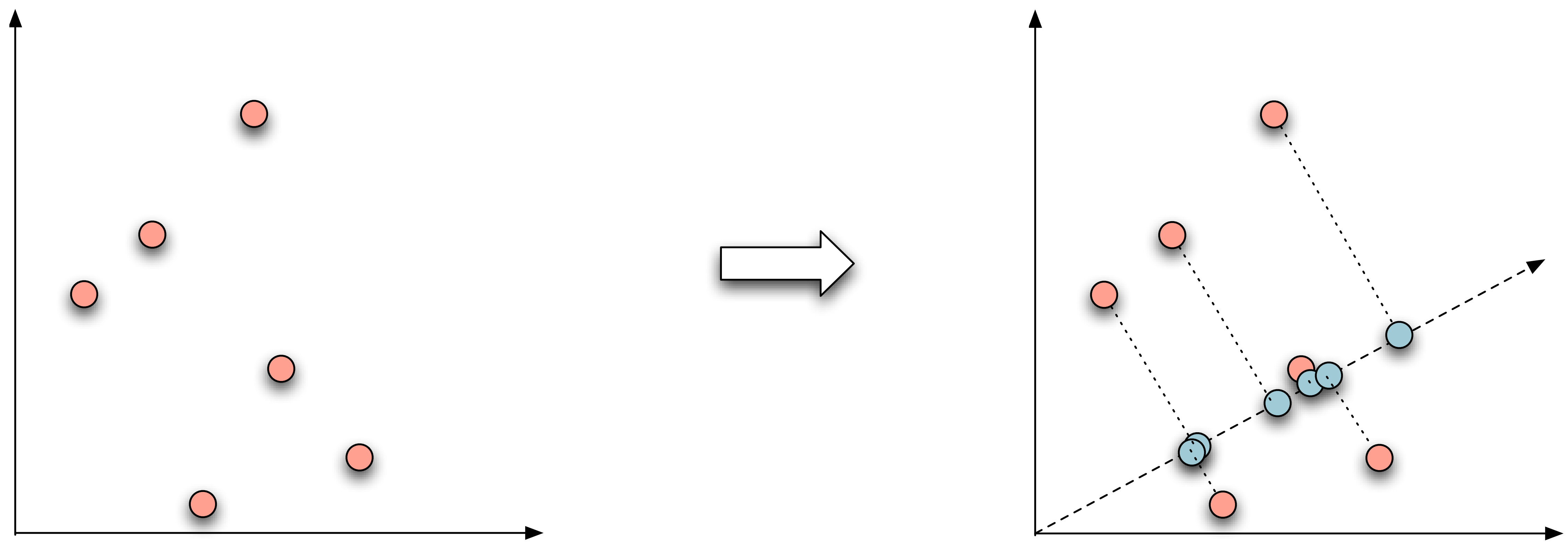
Mathematically speaking

- Latent Semantic Indexing always reduces the number of dimensions



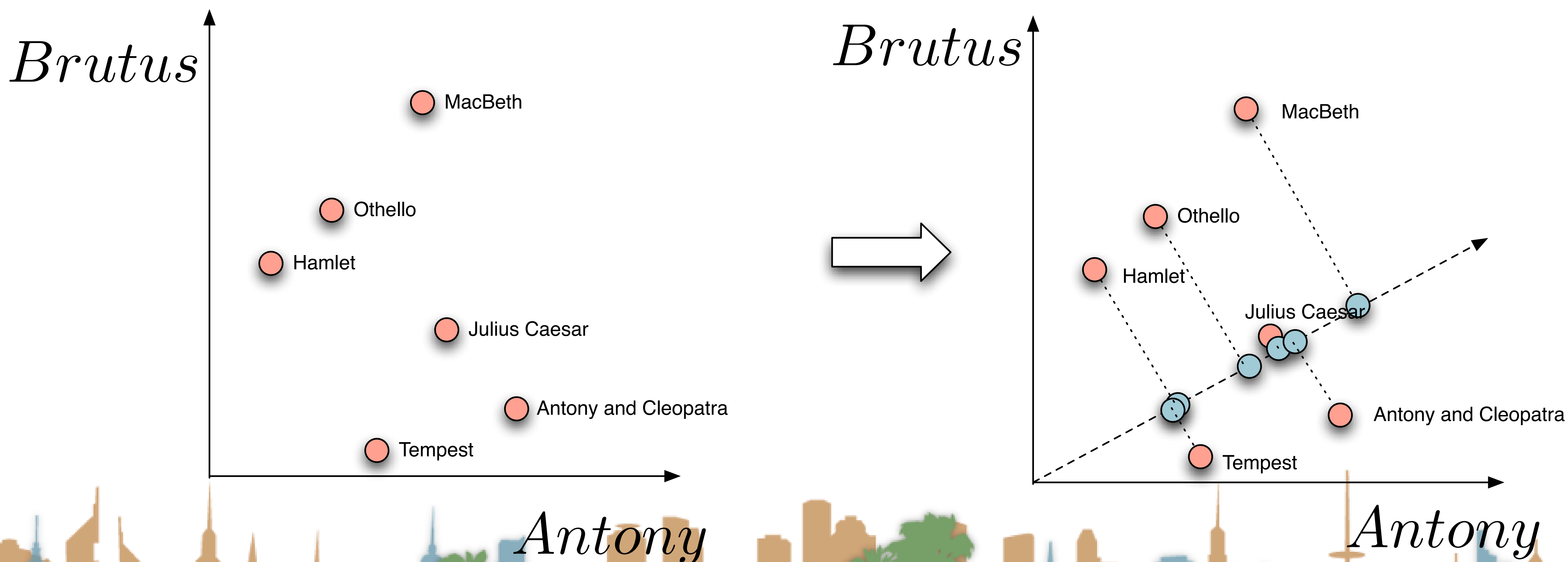
Mathematically speaking

- Latent Semantic Indexing can project on an arbitrary axis, not just a principal axis



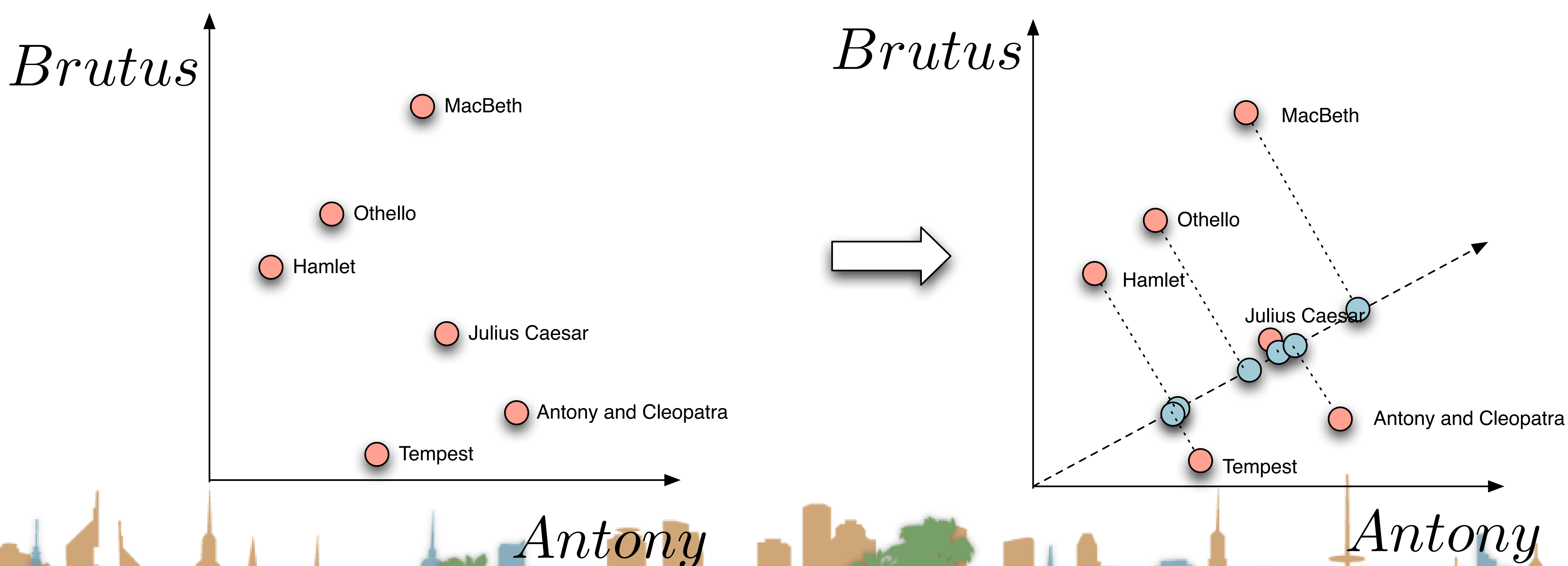
Mathematically speaking

- Our documents were just points in an N-dimensional term space
- We can project them also



Mathematically speaking

- Latent Semantic Indexing makes the claim that these new axes represent semantics - deeper meaning than just a term



Mathematically speaking

- A term vector that is projected on new vectors may uncover deeper meanings
  - For example
    - Transforming the 3 axes of a term matrix from “ball” “bat” and “cave” to
      - An axis that merges “ball” and “bat”
      - An axis that merges “bat” and “cave”
    - Should be able to separate differences in meaning of the term “bat”
    - Bonus: less dimensions is faster

Linear Algebra Refresher

- Let C be an M by N matrix with real-valued entries
  - for example our term document matrix
- A matrix with the same number of rows and columns is called a square matrix
- An M by M matrix with elements only on the diagonal is called a diagonal matrix
- The identity matrix is a diagonal matrix with ones on the main diagnoal

N=5

$$M=3 \quad C = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 2 & 3 & 2 & 1 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix}$$

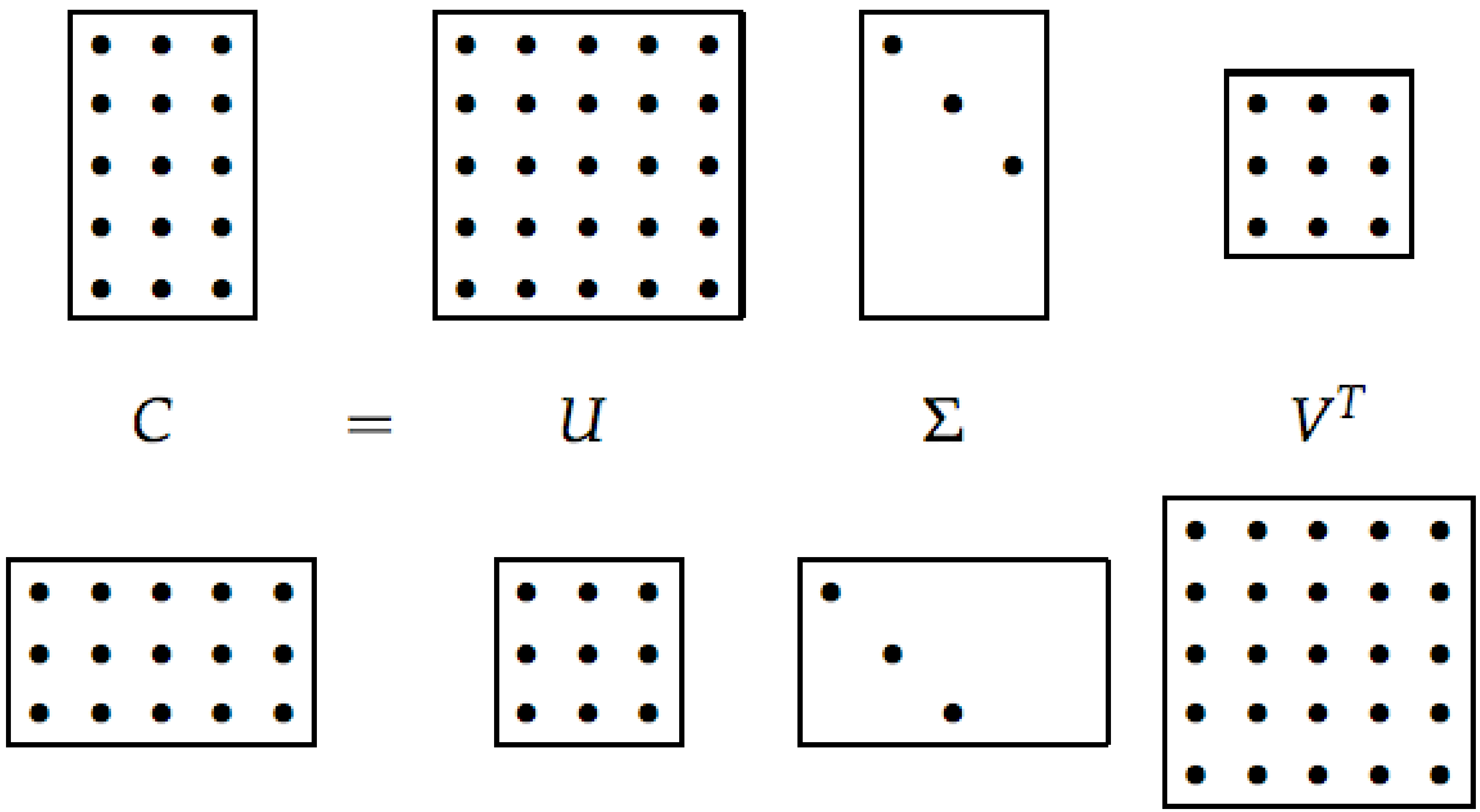
$$\begin{bmatrix} 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

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

Matrix Decomposition

- Singular Value Decomposition
  - Splits a matrix into three matrices $U \quad \Sigma \quad V^T$
  - Such that $C = U\Sigma V^T$
    - If $C\ is\ (M\ by\ N)$
    - then $U\ is\ (M\ by\ M)$
    - and $\Sigma\ is\ (M\ by\ N)$
    - and $V^T\ is\ (N\ by\ N)$
    - also Sigma is almost a diagonal matrix

Matrix Decomposition



Matrix Decomposition

- Singular Value Decomposition
  - Is a technique that splits a matrix into three components with these properties.
  - They also have some other properties which are relevant to latent semantic indexing