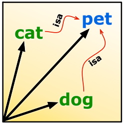
Distributional Semantic Models

Part 2: The parameters of a DSM

Stefan Evert[1]
with Alessandro Lenci[2], Marco Baroni[3] and Gabriella Lapesa[4]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
[2]University of Pisa, Italy
[3]University of Trento, Italy
[4]University of Stuttgart, Germany

http://wordspace.collocations.de/doku.php/course:start



Outline

General definition of DSMs

A **distributional semantic model** (DSM) is a scaled and/or transformed co-occurrence matrix **M**, such that each row **x** represents the distribution of a target term across contexts.

	get	see	use	hear	eat	kill
knife	0.027	-0.024	0.206	-0.022	-0.044	-0.042
cat	0.031	0.143	-0.243	-0.015	-0.009	0.131
dog	-0.026	0.021	-0.212	0.064	0.013	0.014
boat	-0.022	0.009	-0.044	-0.040	-0.074	-0.042
cup	-0.014	-0.173	-0.249	-0.099	-0.119	-0.042
pig	-0.069	0.094	-0.158	0.000	0.094	0.265
banana	0.047	-0.139	-0.104	-0.022	0.267	-0.042

Term = word, lemma, phrase, morpheme, word pair, . . .

General definition of DSMs

Mathematical notation:

- $k \times n$ co-occurrence matrix $\mathbf{M} \in \mathbb{R}^{k \times n}$ (example: 7×6)
  - k rows = **target** terms
  - n columns = **features** or **dimensions**

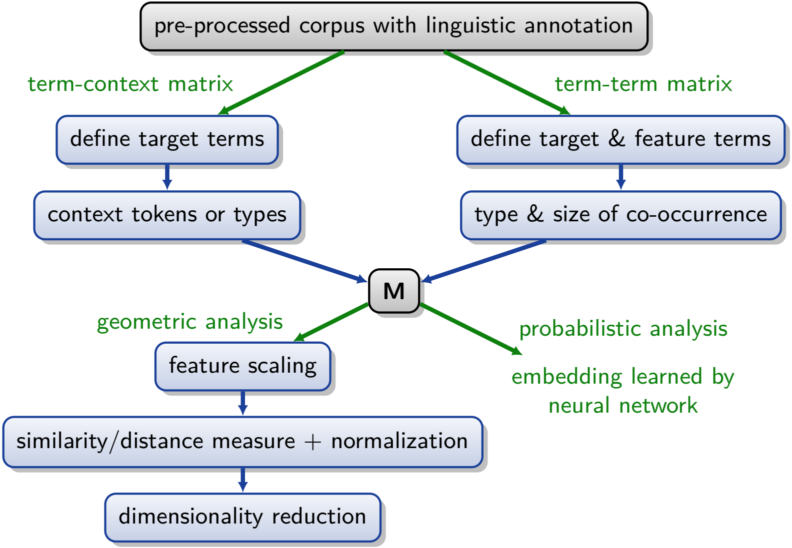
$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & & \vdots \\ m_{k1} & m_{k2} & \cdots & m_{kn} \end{bmatrix}$$

- distribution vector $\mathbf{m}_i$ = i-th row of $\mathbf{M}$, e.g. $\mathbf{m}_3 = \mathbf{m}_{\text{dog}} \in \mathbb{R}^n$
- components $\mathbf{m}_i = (m_{i1}, m_{i2}, \ldots, m_{in})$ = features of i-th term:

$$\begin{aligned} \mathbf{m}_3 &= (-0.026, 0.021, -0.212, 0.064, 0.013, 0.014) \\ &= (m_{31}, m_{32}, m_{33}, m_{34}, m_{35}, m_{36}) \end{aligned}$$

Outline

Overview of DSM parameters



Term-context matrix

Term-context matrix records frequency of term in each individual context (e.g. sentence, document, Web page, encyclopaedia article)

$$\mathbf{F} = \begin{bmatrix} \cdots & \mathbf{f}_1 & \cdots \\ \cdots & \mathbf{f}_2 & \cdots \\ & \vdots & \\ & \vdots & \\ \cdots & \mathbf{f}_k & \cdots \end{bmatrix}$$

	Felidae	Pet	Feral	Bloat	Philosophy	Kant	Back pain
cat	10	10	7	–	–	–	–
dog	–	10	4	11	–	–	–
animal	2	15	10	2	–	–	–
time	1	–	–	–	2	1	–
reason	–	1	–	–	1	4	1
cause	–	–	–	2	1	2	6
effect	–	–	–	1	–	1	–

```
> TC <- DSM_TermContext
> head(TC, Inf) # extract full co-oc matrix from DSM object
```

Term-term matrix

Term-term matrix records co-occurrence frequencies with feature terms for each target term

$$\mathbf{M} = \begin{bmatrix} \cdots & \mathbf{m}_1 & \cdots \\ \cdots & \mathbf{m}_2 & \cdots \\ & \vdots & \\ & \vdots & \\ \cdots & \mathbf{m}_k & \cdots \end{bmatrix}$$

	breed	tail	feed	kill	important	explain	likely
cat	83	17	7	37	–	1	–
dog	561	13	30	60	1	2	4
animal	42	10	109	134	13	5	5
time	19	9	29	117	81	34	109
reason	1	–	2	14	68	140	47
cause	–	1	–	4	55	34	55
effect	–	–	1	6	60	35	17

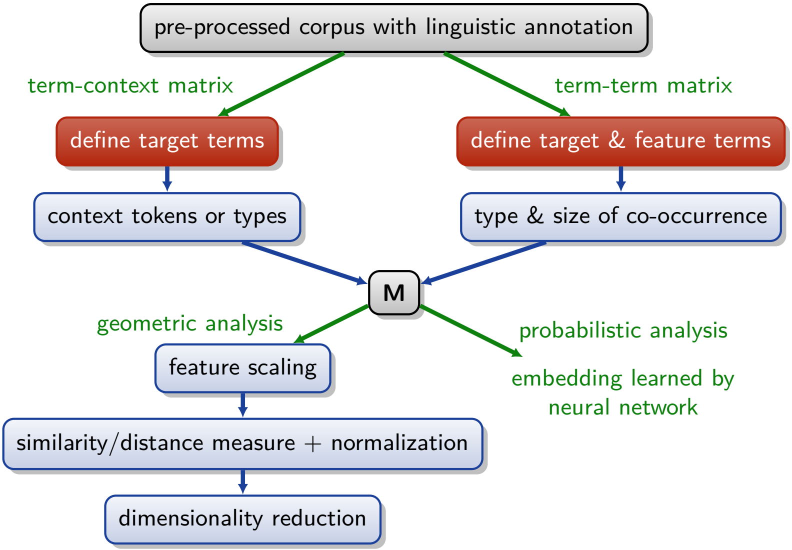
```
> TT <- DSM_TermTerm
> head(TT, Inf)
```

Term-term matrix

Some footnotes:

- Often target terms $\neq$ feature terms
  - e.g. nouns described by co-occurrences with verbs as features
  - identical sets of target & feature terms ➜ symmetric matrix
- Different types of co-occurrence (Evert 2008)
  - **surface context** (word or character window)
  - **textual context** (non-overlapping segments)
  - **syntactic context** (dependency relation)
- Can be seen as smoothing of term-context matrix
  - average over similar contexts (with same context terms)
  - data sparseness reduced, except for small windows
  - we will take a closer look at the relation between term-context and term-term models in part 5 of this tutorial

Overview of DSM parameters



Definition of target and feature terms

- Choice of linguistic unit
  - words
  - bigrams, trigrams, ...
  - multiword units, named entities, phrases, ...
  - morphemes
  - word pairs (☞ analogy tasks)

Definition of target and feature terms

- Choice of linguistic unit
  - words
  - bigrams, trigrams, . . .
  - multiword units, named entities, phrases, . . .
  - morphemes
  - word pairs (☞ analogy tasks)
- Linguistic annotation
  - word forms (minimally requires tokenisation)
  - often lemmatisation or stemming to reduce data sparseness: *go, goes, went, gone, going* ➜ *go*
  - POS disambiguation (*light*/N vs. *light*/A vs. *light*/V)
  - word sense disambiguation (*bank*$_{\text{river}}$ vs. *bank*$_{\text{finance}}$)
  - abstraction: POS tags (or bigrams) as feature terms

Definition of target and feature terms

- Choice of linguistic unit
  - words
  - bigrams, trigrams, ...
  - multiword units, named entities, phrases, ...
  - morphemes
  - word pairs (☞ analogy tasks)
- Linguistic annotation
  - word forms (minimally requires tokenisation)
  - often lemmatisation or stemming to reduce data sparseness: *go, goes, went, gone, going* ➜ *go*
  - POS disambiguation (*light*/N vs. *light*/A vs. *light*/V)
  - word sense disambiguation (*bank*$_{\text{river}}$ vs. *bank*$_{\text{finance}}$)
  - abstraction: POS tags (or bigrams) as feature terms
- Trade-off between deeper linguistic analysis and
  - need for language-specific resources
  - possible errors introduced at each stage of the analysis

Effects of linguistic annotation

Nearest neighbours of *walk* (BNC)

word forms

- stroll
- walking
- walked
- go
- path
- drive
- ride
- wander
- sprinted
- sauntered

lemmatised + POS

- hurry
- stroll
- stride
- trudge
- amble
- wander
- walk (noun)
- walking
- retrace
- scuttle

http://clic.cimec.unitn.it/infomap-query/

Effects of linguistic annotation

Nearest neighbours of *arrivare* (Repubblica)

word forms

- giungere
- raggiungere
- arrivi
- raggiungimento
- raggiunto
- trovare
- raggiunge
- arrivasse
- arriverà
- concludere

lemmatised + POS

- giungere
- aspettare
- attendere
- arrivo (noun)
- ricevere
- accontentare
- approdare
- pervenire
- venire
- piombare

`http://clic.cimec.unitn.it/infomap-query/`

Selection of target and feature terms

- Full-vocabulary models are often unmanageable
  - 762,424 distinct word forms in BNC, 605,910 lemmata
  - large Web corpora have $>$ 10 million distinct word forms
  - low-frequency targets (and features) are not reliable ("noisy")

Selection of target and feature terms

- Full-vocabulary models are often unmanageable
  - 762,424 distinct word forms in BNC, 605,910 lemmata
  - large Web corpora have > 10 million distinct word forms
  - low-frequency targets (and features) are not reliable ("noisy")
- Frequency-based selection
  - minimum corpus frequency: $f \geq F_{\text{min}}$
  - or accept n_w most frequent terms
  - sometimes also upper threshold: $F_{\text{min}} \leq f \leq F_{\text{max}}$

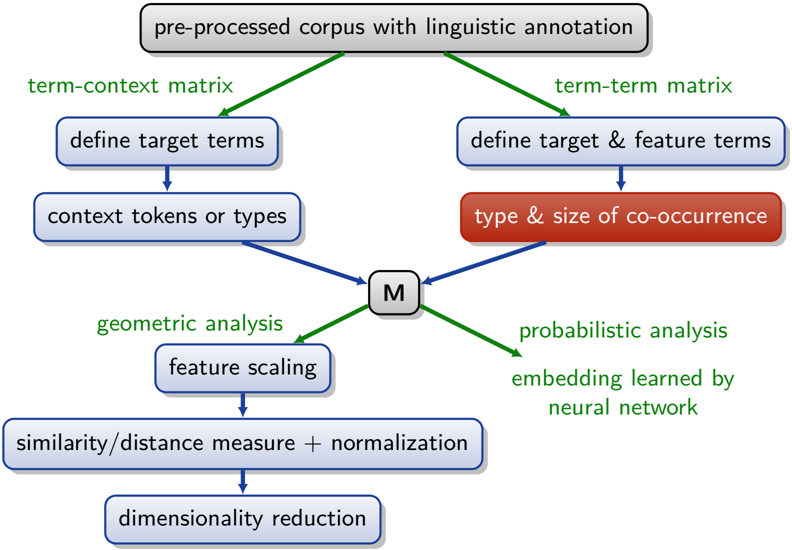
Selection of target and feature terms

- Full-vocabulary models are often unmanageable
  - 762,424 distinct word forms in BNC, 605,910 lemmata
  - large Web corpora have > 10 million distinct word forms
  - low-frequency targets (and features) are not reliable ("noisy")
- Frequency-based selection
  - minimum corpus frequency: $f \geq F_{\text{min}}$
  - or accept n_w most frequent terms
  - sometimes also upper threshold: $F_{\text{min}} \leq f \leq F_{\text{max}}$
- Relevance-based selection
  - criterion from IR: document frequency *df*
  - high *df* ➜ uninformative / low *df* ➜ too sparse to be useful
  - alternatives: entropy H or chi-squared statistic X^2

Selection of target and feature terms

- Full-vocabulary models are often unmanageable
  - 762,424 distinct word forms in BNC, 605,910 lemmata
  - large Web corpora have $>$ 10 million distinct word forms
  - low-frequency targets (and features) are not reliable ("noisy")
- Frequency-based selection
  - minimum corpus frequency: $f \geq F_{\text{min}}$
  - or accept n_w most frequent terms
  - sometimes also upper threshold: $F_{\text{min}} \leq f \leq F_{\text{max}}$
- Relevance-based selection
  - criterion from IR: document frequency *df*
  - high *df* ➜ uninformative / low *df* ➜ too sparse to be useful
  - alternatives: entropy H or chi-squared statistic X^2
- Other criteria
  - POS-based filter: no function words, only verbs, nouns, …
  - general dictionary, words required for particular task, …

Overview of DSM parameters



Surface context

Context term occurs within a span of k words around target.

The silhouette of the sun beyond a wide-open bay on the lake; the sun still glitters although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners. [L3/R3 span, $k = 6$]

Parameters:

- span size (in words or characters)
- symmetric vs. one-sided span
- uniform or "triangular" (distance-based) weighting (don't!)
- spans clamped to sentences or other textual units?

Effect of span size

Nearest neighbours of *dog* (BNC)

2-word span	30-word span
cat	kennel
horse	puppy
fox	pet
pet	bitch
rabbit	terrier
pig	rottweiler
animal	canine
mongrel	cat
sheep	to bark
pigeon	Alsatian

http://clic.cimec.unitn.it/infomap-query/

Textual context

Context term is in the same linguistic unit as target.

The silhouette of the sun beyond a wide-open bay on the lake; the sun still glitters although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners.

Parameters:

- type of linguistic unit
  - sentence
  - paragraph
  - turn in a conversation
  - Web page
  - tweet

Syntactic context

Context term is linked to target by a syntactic dependency (e.g. subject, modifier, . . .).

The silhouette of the sun beyond a wide-open bay on the lake; the sun still glitters although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners.

Parameters:

- types of syntactic dependency (Padó & Lapata 2007)
- direct vs. indirect dependency paths
- homogeneous data (e.g. only verb-object) vs. heterogeneous data (e.g. all children and parents of the verb)
- maximal length of dependency path

"Knowledge pattern" context

Context term is linked to target by a lexico-syntactic pattern (text mining, cf. Hearst 1992, Pantel & Pennacchiotti 2008, etc.).

In Provence, Van Gogh painted with bright colors such as red and yellow. These colors produce incredible effects on anybody looking at his paintings.

Parameters:

- inventory of lexical patterns
  - lots of research to identify semantically interesting patterns (cf. Almuhareb & Poesio 2004, Veale & Hao 2008, etc.)
- fixed vs. flexible patterns
  - patterns are mined from large corpora and automatically generalised (optional elements, POS tags or semantic classes)

Comparison of co-occurrence contexts

Contexts range from general/implict to specific/explicit:

	features are
textual / large span	from same general domain

Comparison of co-occurrence contexts

Contexts range from general/implict to specific/explicit:

	features are
textual / large span	from same general domain
small span	collocations

Comparison of co-occurrence contexts

Contexts range from general/implict to specific/explicit:

	features are
textual / large span	from same general domain
small span	collocations
syntactic (single relation)	attributes (focus on aspect)

Comparison of co-occurrence contexts

Contexts range from general/implict to specific/explicit:

	features are
textual / large span	from same general domain
small span	collocations
syntactic (single relation)	attributes (focus on aspect)
knowledge pattern	properties

Structured vs. unstructured context

- In **unstructured** models, context specification acts as a **filter**
  - determines whether context token counts as co-occurrence
  - e.g. muste be linked by any syntactic dependency relation

Structured vs. unstructured context

- In **unstructered** models, context specification acts as a **filter**
  - determines whether context token counts as co-occurrence
  - e.g. muste be linked by any syntactic dependency relation
- In **structured** models, feature terms are **subtyped**
  - depending on their position in the context
  - e.g. left vs. right context, type of syntactic relation, etc.

Structured vs. unstructured surface context

A dog bites a man. The man's dog bites a dog. A dog bites a man.

unstructured	bite
dog	4
man	3

Structured vs. unstructured surface context

A dog bites a man. The man's dog bites a dog. A dog bites a man.

unstructured	bite
dog	4
man	3

A dog bites a man. The man's dog bites a dog. A dog bites a man.

structured	bite-l	bite-r
dog	3	1
man	1	2

Structured vs. unstructured dependency context

A dog bites a man. The man's dog bites a dog. A dog bites a man.

unstructured	bite
dog	4
man	2

Structured vs. unstructured dependency context

A dog bites a man. The man's dog bites a dog. A dog bites a man.

unstructured	bite
dog	4
man	2

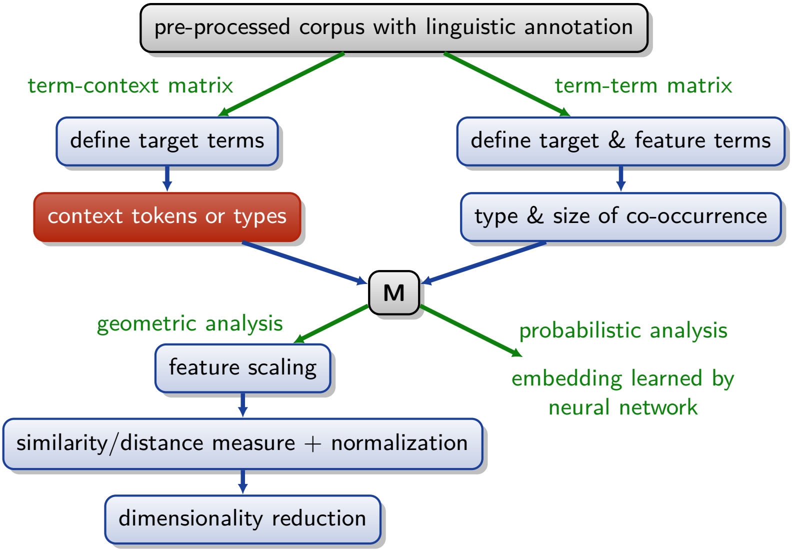
A dog bites a man. The man's dog bites a dog. A dog bites a man.

structured	bite-subj	bite-obj
dog	3	1
man	0	2

Comparison

- Unstructured context
  - data less sparse (e.g. *man kills* and *kills man* both map to the *kill* dimension of the vector $\mathbf{x}_{\text{man}}$)
- Structured context
  - more sensitive to semantic distinctions (*kill-subj* and *kill-obj* are rather different things!)
  - dependency relations provide a form of syntactic “typing” of the DSM dimensions (the “subject” dimensions, the “recipient” dimensions, etc.)
  - important to account for word-order and compositionality

Overview of DSM parameters



Context tokens vs. context types

- Features are usually context **tokens**, i.e. individual instances
  - document, Wikipedia article, Web page, ...
  - paragraph, sentence, tweet, ...
  - "co-occurrence" count = frequency of term in context token

Context tokens vs. context types

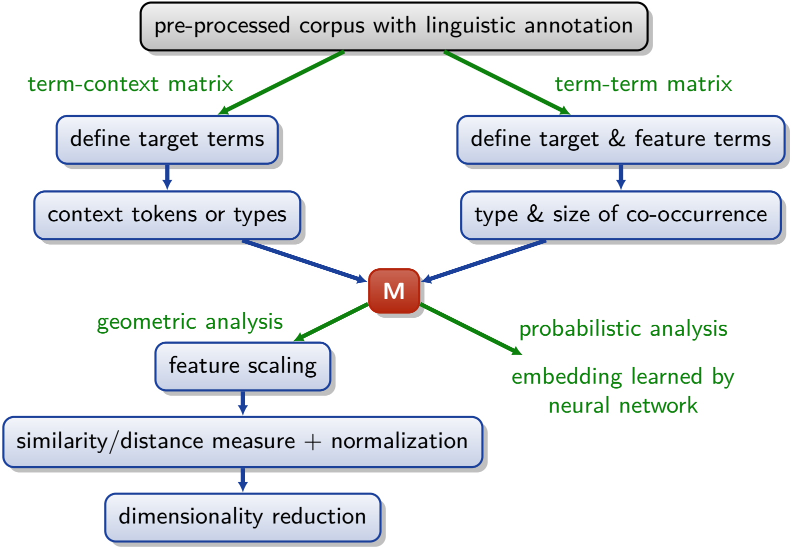
- Features are usually context **tokens**, i.e. individual instances
  - document, Wikipedia article, Web page, ...
  - paragraph, sentence, tweet, ...
  - "co-occurrence" count = frequency of term in context token

- Can also be generalised to context **types**, e.g.
  - type = cluster of near-duplicate documents
  - type = syntactic structure of sentence (ignoring content)
  - type = tweets from same author
  - frequency counts from all instances of type are aggregated

Context tokens vs. context types

- Features are usually context **tokens**, i.e. individual instances
  - document, Wikipedia article, Web page, ...
  - paragraph, sentence, tweet, ...
  - "co-occurrence" count = frequency of term in context token
- Can also be generalised to context **types**, e.g.
  - type = cluster of near-duplicate documents
  - type = syntactic structure of sentence (ignoring content)
  - type = tweets from same author
  - frequency counts from all instances of type are aggregated
- Context types may be anchored at individual tokens
  - n-gram of words (or POS tags) around target
  - subcategorisation pattern of target verb
  - ➥ overlaps with (generalisation of) syntactic co-occurrence

Overview of DSM parameters



Marginal and expected frequencies

- Matrix of observed co-occurrence frequencies not sufficient

target	feature	O
dog	*small*	855
dog	*domesticated*	29

- Notation
  - O = observed co-occurrence frequency

Marginal and expected frequencies

- Matrix of observed co-occurrence frequencies not sufficient

target	feature	O	R	C
dog	*small*	855	33,338	490,580
dog	*domesticated*	29	33,338	918

- Notation
  - O = observed co-occurrence frequency
  - R = overall frequency of target term = row marginal frequency
  - C = overall frequency of feature = column marginal frequency
  - N = sample size $\approx$ size of corpus

Marginal and expected frequencies

- Matrix of observed co-occurrence frequencies not sufficient

target	feature	O	R	C	E
dog	*small*	855	33,338	490,580	134.34
dog	*domesticated*	29	33,338	918	0.25

- Notation
  - O = observed co-occurrence frequency
  - R = overall frequency of target term = row marginal frequency
  - C = overall frequency of feature = column marginal frequency
  - N = sample size ≈ size of corpus
- **Expected** co-occurrence **frequency**

$$E = \frac{R \cdot C}{N} \quad \longleftrightarrow \quad O$$

Obtaining marginal frequencies

- Term-document matrix
  - R = frequency of target term in corpus
  - C = size of document (# tokens)
  - N = corpus size

Obtaining marginal frequencies

- Term-document matrix
  - R = frequency of target term in corpus
  - C = size of document (# tokens)
  - N = corpus size
- Syntactic co-occurrence
  - # of dependency instances in which target/feature participates
  - N = total number of dependency instances
  - can be computed from full co-occurrence matrix $\mathbf{M}$

Obtaining marginal frequencies

- Term-document matrix
  - R = frequency of target term in corpus
  - C = size of document (# tokens)
  - N = corpus size
- Syntactic co-occurrence
  - # of dependency instances in which target/feature participates
  - N = total number of dependency instances
  - can be computed from full co-occurrence matrix $\mathbf{M}$
- Textual co-occurrence
  - R, C, O are "document" frequencies, i.e. number of context units in which target, feature or combination occurs
  - N = total # of context units

Obtaining marginal frequencies

- Surface co-occurrence
  - it is quite tricky to obtain fully consistent counts (Evert 2008)
  - at least correct E for span size k (= number of tokens in span)

$$E = k \cdot \frac{R \cdot C}{N}$$

with R, C = individual corpus frequencies and N = corpus size
  - can also be implemented by pre-multiplying $R' = k \cdot R$
  - ☞ alternatively, compute marginals and sample size by summing over full co-occurrence matrix (➜ E as above, but inflated N)

Obtaining marginal frequencies

- Surface co-occurrence
  - it is quite tricky to obtain fully consistent counts (Evert 2008)
  - at least correct E for span size k (= number of tokens in span)

$$E = k \cdot \frac{R \cdot C}{N}$$

    with R, C = individual corpus frequencies and N = corpus size
  - can also be implemented by pre-multiplying $R' = k \cdot R$
  - ☞ alternatively, compute marginals and sample size by summing over full co-occurrence matrix (➜ E as above, but inflated N)

- NB: shifted PPMI (Levy & Goldberg 2014) corresponds to a post-hoc application of the span size adjustment
  - performs worse than PPMI, but paper suggests they already approximate correct E by summing over co-occurrence matrix

Marginal frequencies in wordspace

DSM objects in wordspace (class dsm) include marginal frequencies as well as counts of nonzero cells for rows and columns.

```
> TT$rows
    term       f nnzero
1    cat   22007      5
2    dog   50807      7
3 animal   77053      7
4   time 1156693      7
5 reason   95047      6
6  cause   54739      5
7 effect  133102      6
> TT$cols
...
> TT$globals$N
[1] 199902178
> TT$M # the full co-occurrence matrix
```

Geometric vs. probabilistic interpretation

- Geometric interpretation
  - row vectors as points or arrows in *n*-dimensional space
  - very intuitive, good for visualisation
  - use techniques from geometry and matrix algebra

Geometric vs. probabilistic interpretation

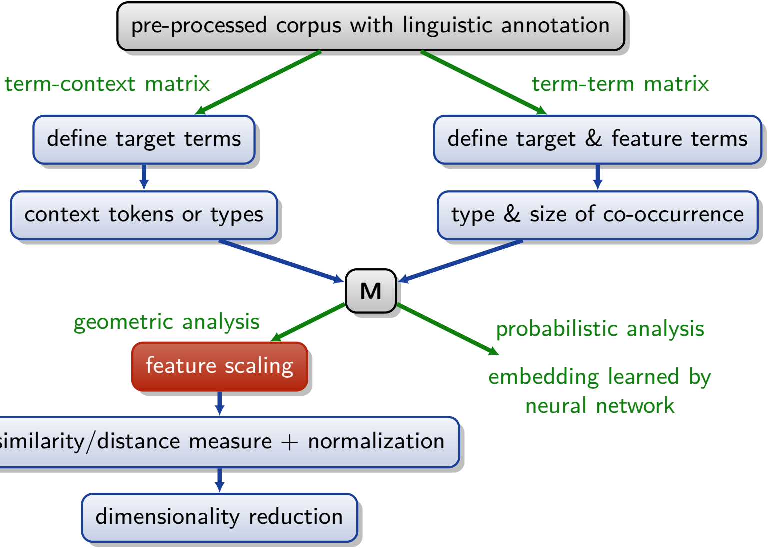
- Geometric interpretation
  - row vectors as points or arrows in *n*-dimensional space
  - very intuitive, good for visualisation
  - use techniques from geometry and matrix algebra
- Probabilistic interpretation
  - co-occurrence matrix as observed sample statistic that is "explained" by a generative probabilistic model
  - e.g. probabilistic LSA (Hoffmann 1999), Latent Semantic Clustering (Rooth *et al.* 1999), Latent Dirichlet Allocation (Blei *et al.* 2003), etc.
  - explicitly accounts for random variation of frequency counts
  - recent work: neural word embeddings

Geometric vs. probabilistic interpretation

- Geometric interpretation
  - row vectors as points or arrows in n-dimensional space
  - very intuitive, good for visualisation
  - use techniques from geometry and matrix algebra
- Probabilistic interpretation
  - co-occurrence matrix as observed sample statistic that is "explained" by a generative probabilistic model
  - e.g. probabilistic LSA (Hoffmann 1999), Latent Semantic Clustering (Rooth *et al.* 1999), Latent Dirichlet Allocation (Blei *et al.* 2003), etc.
  - explicitly accounts for random variation of frequency counts
  - recent work: neural word embeddings

☞ focus on geometric interpretation in this tutorial

Overview of DSM parameters



Feature scaling

Feature scaling is used to "discount" less important features:

- Logarithmic scaling: $O' = \log(O + 1)$
  (cf. Weber-Fechner law for human perception)

Feature scaling

Feature scaling is used to "discount" less important features:

- Logarithmic scaling: $O' = \log(O + 1)$
  (cf. Weber-Fechner law for human perception)
- Relevance weighting, e.g. tf.idf (information retrieval)

$$tf.idf = tf \cdot log(D/df)$$

  - tf = co-occurrence frequency O
  - df = document frequency of feature (or nonzero count)
  - D = total number of documents (or row count of $\mathbf{M}$)

Feature scaling

Feature scaling is used to "discount" less important features:

- Logarithmic scaling: $O' = \log(O + 1)$
  (cf. Weber-Fechner law for human perception)
- Relevance weighting, e.g. tf.idf (information retrieval)

$$tf.idf = tf \cdot log(D/df)$$

  - tf = co-occurrence frequency O
  - df = document frequency of feature (or nonzero count)
  - D = total number of documents (or row count of $\mathbf{M}$)
- Statistical **association measures** (Evert 2004, 2008) take frequency of target term and feature into account
  - often based on comparison of observed and expected co-occurrence frequency
  - measures differ in how they balance O and E

Simple association measures

target	feature	O	E
dog	*small*	855	134.34
dog	*domesticated*	29	0.25
dog	*sgjkj*	1	0.00027

Simple association measures

- pointwise Mutual Information (MI)

$$\text{MI} = \log_2 \frac{O}{E}$$

target	feature	O	E	MI
dog	*small*	855	134.34	2.67
dog	*domesticated*	29	0.25	6.85
dog	*sgjkj*	1	0.00027	11.85

Simple association measures

- pointwise Mutual Information (MI)

$$\text{MI} = \log_2 \frac{O}{E}$$

- local MI

$$\text{local-MI} = O \cdot \text{MI} = O \cdot \log_2 \frac{O}{E}$$

target	feature	O	E	MI	local-MI
dog	*small*	855	134.34	2.67	2282.88
dog	*domesticated*	29	0.25	6.85	198.76
dog	*sgjkj*	1	0.00027	11.85	11.85

Simple association measures

- pointwise Mutual Information (MI)

$$\mathrm{MI} = \log_2 \frac{O}{E}$$

- local MI

$$\text{local-MI} = O \cdot \mathrm{MI} = O \cdot \log_2 \frac{O}{E}$$

- t-score

$$t = \frac{O - E}{\sqrt{O}}$$

target	feature	O	E	MI	local-MI	t-score
dog	*small*	855	134.34	2.67	2282.88	24.64
dog	*domesticated*	29	0.25	6.85	198.76	5.34
dog	*sgjkj*	1	0.00027	11.85	11.85	1.00

Other association measures

- simple log-likelihood ($\approx$ local-MI)

$$G^2 = \pm\, 2 \cdot \left(O \cdot \log_2 \frac{O}{E} - (O - E) \right)$$

with positive sign for $O > E$ and negative sign for $O < E$

Other association measures

- simple log-likelihood ($\approx$ local-MI)

$$G^2 = \pm\, 2 \cdot \left(O \cdot \log_2 \frac{O}{E} - (O - E) \right)$$

  with positive sign for $O > E$ and negative sign for $O < E$

- Dice coefficient

$$\text{Dice} = \frac{2O}{R + C}$$

Other association measures

- simple log-likelihood ($\approx$ local-MI)

$$G^2 = \pm\, 2 \cdot \left(O \cdot \log_2 \frac{O}{E} - (O - E) \right)$$

  with positive sign for $O > E$ and negative sign for $O < E$
- Dice coefficient

$$\text{Dice} = \frac{2O}{R + C}$$

- Many other simple association measures (AMs) available
- Further AMs computed from full contingency tables, see
  - Evert (2008)
  - http://www.collocations.de/
  - http://sigil.r-forge.r-project.org/

Applying association scores in wordspace

```
> options(digits=3)  # print fractional values with limited precision
> dsm.score(TT, score="MI", sparse=FALSE, matrix=TRUE)
       breed   tail   feed   kill important explain  likely
cat     6.21  4.568  3.129  2.801      -Inf  0.0182    -Inf
dog     7.78  3.081  3.922  2.323    -3.774 -1.1888 -0.4958
animal  3.50  2.132  4.747  2.832    -0.674 -0.4677 -0.0966
time   -1.65 -2.236 -0.729 -1.097    -1.728 -1.2382  0.6392
reason -2.30   -Inf -1.982 -0.388     1.472  4.0368  2.8860
cause   -Inf -0.834   -Inf -2.177     1.900  2.8329  4.0691
effect  -Inf -2.116 -2.468 -2.459     0.791  1.6312  0.9221
```

Applying association scores in wordspace

```
> options(digits=3)  # print fractional values with limited precision
> dsm.score(TT, score="MI", sparse=FALSE, matrix=TRUE)
        breed   tail   feed   kill important explain  likely
cat      6.21  4.568  3.129  2.801      -Inf  0.0182    -Inf
dog      7.78  3.081  3.922  2.323    -3.774 -1.1888 -0.4958
animal   3.50  2.132  4.747  2.832    -0.674 -0.4677 -0.0966
time    -1.65 -2.236 -0.729 -1.097    -1.728 -1.2382  0.6392
reason  -2.30   -Inf -1.982 -0.388     1.472  4.0368  2.8860
cause    -Inf -0.834   -Inf -2.177     1.900  2.8329  4.0691
effect   -Inf -2.116 -2.468 -2.459     0.791  1.6312  0.9221
```

- ☞ sparseness of the matrix has been lost!
- ☞ cells with score $x = -\infty$ are inconvenient
- ☞ distribution of scores may be even more skewed than co-occurrence frequencies themselves (esp. for local-MI)

Sparse association measures

- Sparse association scores are cut off at zero, i.e.

$$f(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

- Also known as "positive" scores
  - PPMI = positive pointwise MI (e.g. Bullinaria & Levy 2007)
  - `wordspace` computes sparse AMs by default ➜ `"MI"` = PPMI

Sparse association measures

- Sparse association scores are cut off at zero, i.e.

$$f(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

- Also known as "positive" scores
  - PPMI = positive pointwise MI (e.g. Bullinaria & Levy 2007)
  - `wordspace` computes sparse AMs by default ➜ `"MI"` = PPMI
- Preserves sparseness if $x \leq 0$ for all empty cells ($O = 0$)
  - combine with signed AM ($x > 0$ for $O > E$, $x < 0$ for $O < E$)
  - sparseness may even increase: cells with $x < 0$ become empty

Sparse association measures

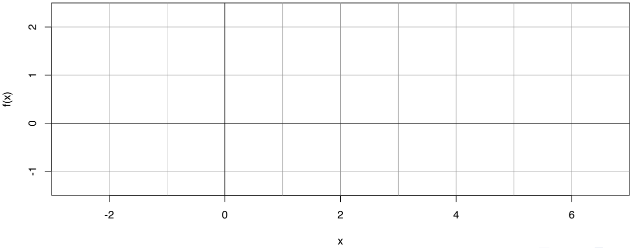
- Sparse association scores are cut off at zero, i.e.

$$f(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

- Also known as "positive" scores
  - PPMI = positive pointwise MI (e.g. Bullinaria & Levy 2007)
  - `wordspace` computes sparse AMs by default ➜ `"MI"` = PPMI
- Preserves sparseness if $x \leq 0$ for all empty cells ($O = 0$)
  - combine with signed AM ($x > 0$ for $O > E$, $x < 0$ for $O < E$)
  - sparseness may even increase: cells with $x < 0$ become empty
- Further thinning may be beneficial (Polajnar & Clark 2014)
  - apply shifted cutoff threshold $x > \theta$ (Levy *et al.* 2015)
  - keep only k top-scoring features for each target

Score transformations

An additional scale transformation can be applied in order to de-skew association scores:

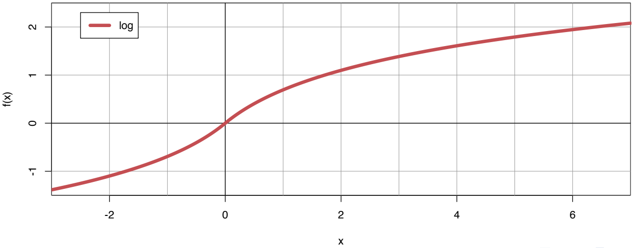


Score transformations

An additional scale transformation can be applied in order to de-skew association scores:

- signed logarithmic transformation

$$f(x) = \pm \log(|x| + 1)$$



Score transformations

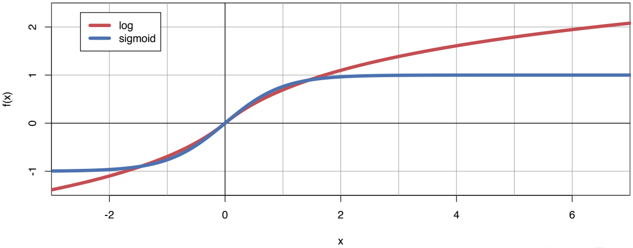
An additional scale transformation can be applied in order to de-skew association scores:

- signed logarithmic transformation

$$f(x) = \pm \log(|x| + 1)$$

- sigmoid transformation as soft binarization

$$f(x) = \tanh x$$



Score transformations

An additional scale transformation can be applied in order to de-skew association scores:

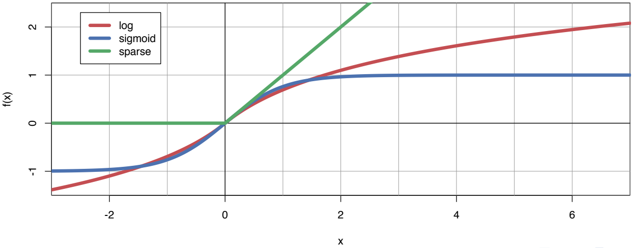
- signed logarithmic transformation

$$f(x) = \pm \log(|x| + 1)$$

- sigmoid transformation as soft binarization

$$f(x) = \tanh x$$

- sparse AM as cutoff transformation



Score transformations

An additional scale transformation can be applied in order to de-skew association scores:

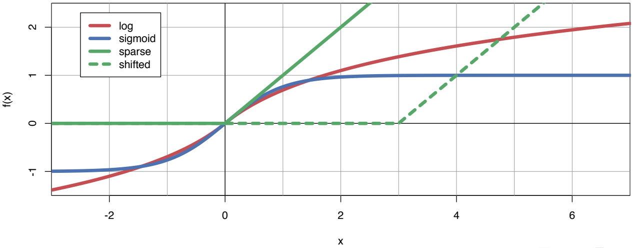
- signed logarithmic transformation

$$f(x) = \pm \log(|x| + 1)$$

- sigmoid transformation as soft binarization

$$f(x) = \tanh x$$

- sparse AM as (shifted) cutoff transformation



Association scores & transformations in wordspace

```
> dsm.score(TT, score="MI", matrix=TRUE) # PPMI
       breed tail feed kill important explain likely
cat     6.21 4.57 3.13 2.80     0.000  0.0182  0.000
dog     7.78 3.08 3.92 2.32     0.000  0.0000  0.000
animal  3.50 2.13 4.75 2.83     0.000  0.0000  0.000
time    0.00 0.00 0.00 0.00     0.000  0.0000  0.639
reason  0.00 0.00 0.00 0.00     1.472  4.0368  2.886
cause   0.00 0.00 0.00 0.00     1.900  2.8329  4.069
effect  0.00 0.00 0.00 0.00     0.791  1.6312  0.922
> dsm.score(TT, score="simple-ll", matrix=TRUE)
> dsm.score(TT, score="simple-ll", transf="log", matrix=T)
# logarithmic co-occurrence frequency
> dsm.score(TT, score="freq", transform="log", matrix=T)

# now try other parameter combinations
> ?dsm.score # read help page for available parameter settings
```

Scaling of column vectors

- In statistical analysis and machine learning, features are usually centered and scaled so that

$$\begin{aligned} \text{mean} \quad & \mu = 0 \\ \text{variance} \quad & \sigma^2 = 1 \end{aligned}$$

Scaling of column vectors

- In statistical analysis and machine learning, features are usually centered and scaled so that

$$\text{mean} \quad \mu = 0$$
$$\text{variance} \quad \sigma^2 = 1$$

- In DSM research, this step is less common for columns of $\mathbf{M}$
  - centering is a prerequisite for certain dimensionality reduction and data analysis techniques (esp. PCA)
  - but co-occurrence matrix no longer sparse!
  - scaling may give too much weight to rare features

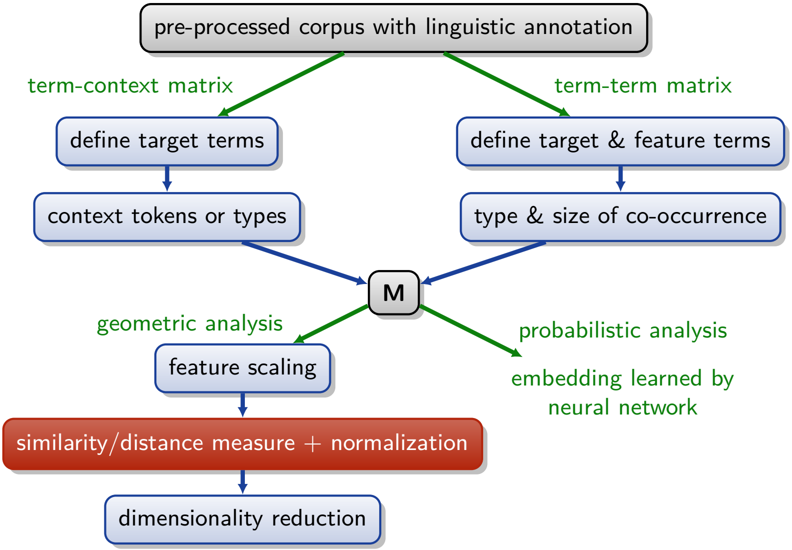
Scaling of column vectors

- In statistical analysis and machine learning, features are usually centered and scaled so that

$$\text{mean} \quad \mu = 0$$
$$\text{variance} \quad \sigma^2 = 1$$

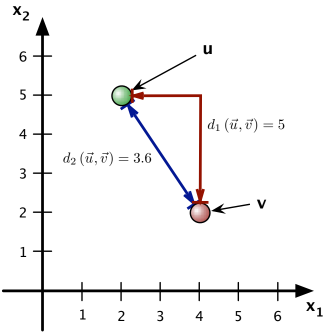
- In DSM research, this step is less common for columns of **M**
  - centering is a prerequisite for certain dimensionality reduction and data analysis techniques (esp. PCA)
  - but co-occurrence matrix no longer sparse!
  - scaling may give too much weight to rare features
- **M** cannot be row-normalised and column-scaled at the same time (result depends on ordering of the two steps)

Overview of DSM parameters



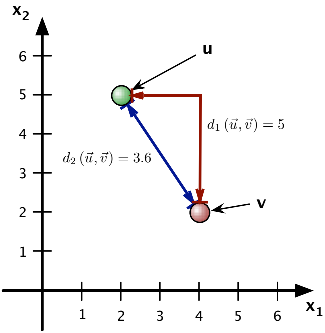
Geometric distance = metric

- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \ldots, u_n)$
  - $\mathbf{v} = (v_1, \ldots, v_n)$



Geometric distance = metric

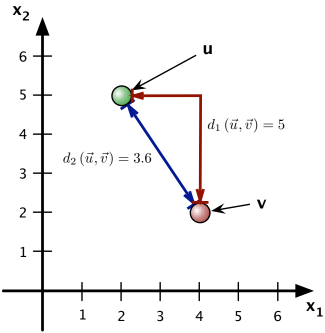
- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \ldots, u_n)$
  - $\mathbf{v} = (v_1, \ldots, v_n)$
- **Euclidean** distance $d_2(\mathbf{u}, \mathbf{v})$



$$d_2(\mathbf{u}, \mathbf{v}) := \sqrt{(u_1 - v_1)^2 + \cdots + (u_n - v_n)^2}$$

Geometric distance = metric

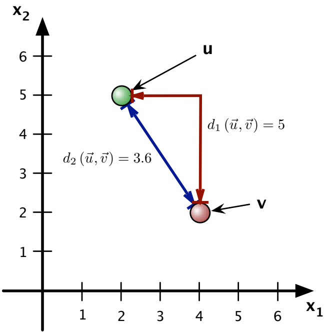
- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \ldots, u_n)$
  - $\mathbf{v} = (v_1, \ldots, v_n)$
- **Euclidean** distance $d_2(\mathbf{u}, \mathbf{v})$
- "City block" **Manhattan** distance $d_1(\mathbf{u}, \mathbf{v})$



$$d_1(\mathbf{u}, \mathbf{v}) := |u_1 - v_1| + \cdots + |u_n - v_n|$$

Geometric distance = metric

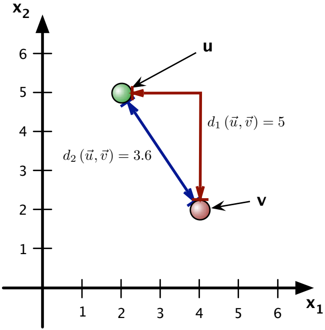
- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \ldots, u_n)$
  - $\mathbf{v} = (v_1, \ldots, v_n)$
- **Euclidean** distance $d_2(\mathbf{u}, \mathbf{v})$
- "City block" **Manhattan** distance $d_1(\mathbf{u}, \mathbf{v})$
- Both are special cases of the **Minkowski** p-distance $d_p(\mathbf{u}, \mathbf{v})$ (for $p \in [1, \infty]$)



$$d_p(\mathbf{u}, \mathbf{v}) := \left(|u_1 - v_1|^p + \cdots + |u_n - v_n|^p\right)^{1/p}$$

Geometric distance = metric

- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \ldots, u_n)$
  - $\mathbf{v} = (v_1, \ldots, v_n)$
- **Euclidean** distance $d_2(\mathbf{u}, \mathbf{v})$
- "City block" **Manhattan** distance $d_1(\mathbf{u}, \mathbf{v})$
- Both are special cases of the **Minkowski** p-distance $d_p(\mathbf{u}, \mathbf{v})$ (for $p \in [1, \infty]$)

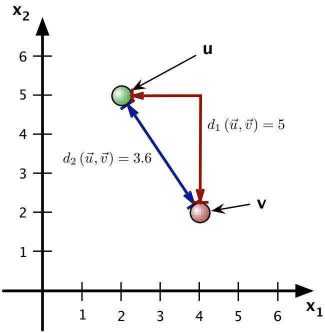


$$d_p(\mathbf{u}, \mathbf{v}) := \left(|u_1 - v_1|^p + \cdots + |u_n - v_n|^p\right)^{1/p}$$

$$d_\infty(\mathbf{u}, \mathbf{v}) = \max\{|u_1 - v_1|, \ldots, |u_n - v_n|\}$$

Geometric distance = metric

- **Distance** between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ➜ (dis)similarity
  - $\mathbf{u} = (u_1, \dots, u_n)$
  - $\mathbf{v} = (v_1, \dots, v_n)$
- **Euclidean** distance $d_2(\mathbf{u}, \mathbf{v})$
- "City block" **Manhattan** distance $d_1(\mathbf{u}, \mathbf{v})$
- Extension of p-distance $d_p(\mathbf{u}, \mathbf{v})$ (for $0 \le p \le 1$)



$$d_p(\mathbf{u}, \mathbf{v}) := |u_1 - v_1|^p + \cdots + |u_n - v_n|^p$$

$$d_0(\mathbf{u}, \mathbf{v}) = \#\{i \mid u_i \neq v_i\}$$

Computing distances

Preparation: store "scored" matrix in DSM object

```
> TT <- dsm.score(TT, score="freq", transform="log")
```

Computing distances

Preparation: store "scored" matrix in DSM object

```
> TT <- dsm.score(TT, score="freq", transform="log")
```

Compute distances between individual term pairs . . .

```
> pair.distances(c("cat","cause"), c("animal","effect"),
                 TT, method="euclidean")
  cat/animal cause/effect
        4.16         1.53
```

Computing distances

Preparation: store "scored" matrix in DSM object

```
> TT <- dsm.score(TT, score="freq", transform="log")
```

Compute distances between individual term pairs . . .

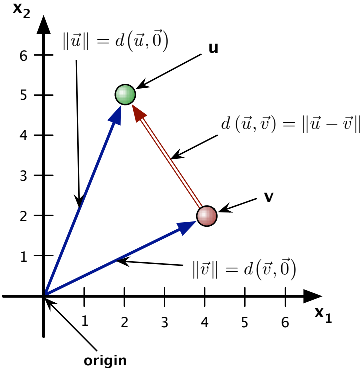
```
> pair.distances(c("cat","cause"), c("animal","effect"),
                 TT, method="euclidean")
  cat/animal cause/effect
        4.16         1.53
```

. . . or full distance matrix.

```
> dist.matrix(TT, method="euclidean")
> dist.matrix(TT, method="minkowski", p=4)
```

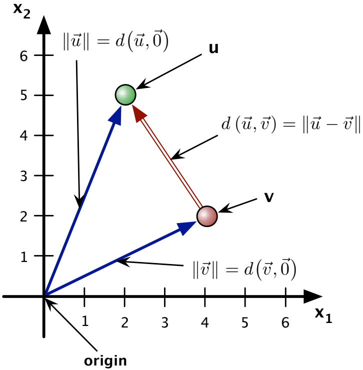
Distance and vector length = norm

- Intuitively, distance $d(\mathbf{u}, \mathbf{v})$ should correspond to length $\|\mathbf{u} - \mathbf{v}\|$ of displacement vector $\mathbf{u} - \mathbf{v}$
  - $d(\mathbf{u}, \mathbf{v})$ is a **metric**
  - $\|\mathbf{u} - \mathbf{v}\|$ is a **norm**
  - $\|\mathbf{u}\| = d(\mathbf{u}, \mathbf{0})$



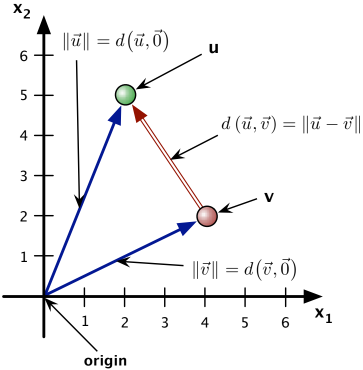
Distance and vector length = norm

- Intuitively, distance $d(\mathbf{u}, \mathbf{v})$ should correspond to length $\|\mathbf{u} - \mathbf{v}\|$ of displacement vector $\mathbf{u} - \mathbf{v}$
  - $d(\mathbf{u}, \mathbf{v})$ is a **metric**
  - $\|\mathbf{u} - \mathbf{v}\|$ is a **norm**
  - $\|\mathbf{u}\| = d(\mathbf{u}, \mathbf{0})$
- Any norm-induced metric is **translation-invariant**



Distance and vector length = norm

- Intuitively, distance $d(\mathbf{u}, \mathbf{v})$ should correspond to length $\|\mathbf{u} - \mathbf{v}\|$ of displacement vector $\mathbf{u} - \mathbf{v}$
  - $d(\mathbf{u}, \mathbf{v})$ is a **metric**
  - $\|\mathbf{u} - \mathbf{v}\|$ is a **norm**
  - $\|\mathbf{u}\| = d(\mathbf{u}, \mathbf{0})$
- Any norm-induced metric is **translation-invariant**

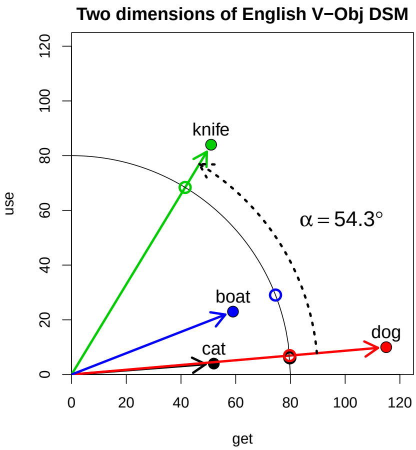


- $d_p(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_p$
- **Minkowski *p*-norm** for $p \in [1, \infty]$ (not $p < 1$):

$$\|\mathbf{u}\|_p := \left(|u_1|^p + \cdots + |u_n|^p\right)^{1/p}$$

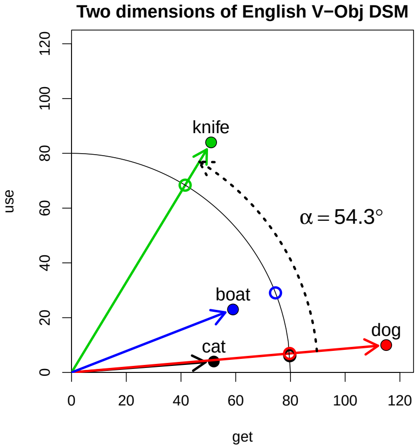
Normalisation of row vectors

- Geometric distances only meaningful for vectors of the same length $\|\mathbf{x}\|$



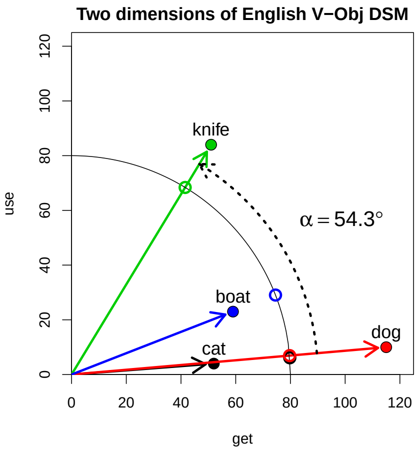
Normalisation of row vectors

- Geometric distances only meaningful for vectors of the same length $\|\mathbf{x}\|$
- Normalize by scalar division: $\mathbf{x}' = \mathbf{x}/\|\mathbf{x}\| = (\frac{x_1}{\|\mathbf{x}\|}, \frac{x_2}{\|\mathbf{x}\|}, \dots)$ with $\|\mathbf{x}'\| = 1$
- Norm must be compatible with distance measure!



Normalisation of row vectors

- Geometric distances only meaningful for vectors of the same length $\|\mathbf{x}\|$
- Normalize by scalar division: $\mathbf{x}' = \mathbf{x}/\|\mathbf{x}\| = \left(\frac{x_1}{\|\mathbf{x}\|}, \frac{x_2}{\|\mathbf{x}\|}, \ldots\right)$ with $\|\mathbf{x}'\| = 1$
- Norm must be compatible with distance measure!
- Special case: scale to relative frequencies with
  $$\|\mathbf{x}\|_1 = |x_1| + \cdots + |x_n|$$
  ➜ probabilistic interpretation



Norms and normalization

```
> rowNorms(TT$S, method="euclidean")
   cat    dog animal   time reason  cause effect
  6.90   8.96   8.82  10.29   8.13   6.86   6.52
```

```
> TT <- dsm.score(TT, score="freq", transform="log",
                  normalize=TRUE, method="euclidean")
> rowNorms(TT$S, method="euclidean") # all = 1 now
> dist.matrix(TT, method="euclidean")
         cat   dog animal  time reason cause effect
cat    0.000 0.224  0.473 0.782  1.121 1.239  1.161
dog    0.224 0.000  0.398 0.698  1.065 1.179  1.113
animal 0.473 0.398  0.000 0.426  0.841 0.971  0.860
time   0.782 0.698  0.426 0.000  0.475 0.585  0.502
reason 1.121 1.065  0.841 0.475  0.000 0.277  0.198
cause  1.239 1.179  0.971 0.585  0.277 0.000  0.224
effect 1.161 1.113  0.860 0.502  0.198 0.224  0.000
```

Distance measures for non-negative vectors

- Information theory: **Kullback-Leibler** (KL) **divergence** for probability vectors (☞ non-negative, $\|\mathbf{x}\|_1 = 1$)

$$D(\mathbf{u}\|\mathbf{v}) = \sum_{i=1}^{n} u_i \cdot \log_2 \frac{u_i}{v_i}$$

Distance measures for non-negative vectors

- Information theory: **Kullback-Leibler** (KL) **divergence** for probability vectors (☞ non-negative, $\|\mathbf{x}\|_1 = 1$)

$$D(\mathbf{u}\|\mathbf{v}) = \sum_{i=1}^{n} u_i \cdot \log_2 \frac{u_i}{v_i}$$

- Properties of KL divergence
  - most appropriate in a probabilistic interpretation of $\mathbf{M}$
  - zeroes in $\mathbf{v}$ without corresponding zeroes in $\mathbf{u}$ are problematic
  - not symmetric, unlike geometric distance measures
  - alternatives: skew divergence, Jensen-Shannon divergence

Distance measures for non-negative vectors

- Information theory: **Kullback-Leibler** (KL) **divergence** for probability vectors (☞ non-negative, $\|\mathbf{x}\|_1 = 1$)

$$D(\mathbf{u}\|\mathbf{v}) = \sum_{i=1}^{n} u_i \cdot \log_2 \frac{u_i}{v_i}$$

- Properties of KL divergence
  - most appropriate in a probabilistic interpretation of $\mathbf{M}$
  - zeroes in $\mathbf{v}$ without corresponding zeroes in $\mathbf{u}$ are problematic
  - not symmetric, unlike geometric distance measures
  - alternatives: skew divergence, Jensen-Shannon divergence

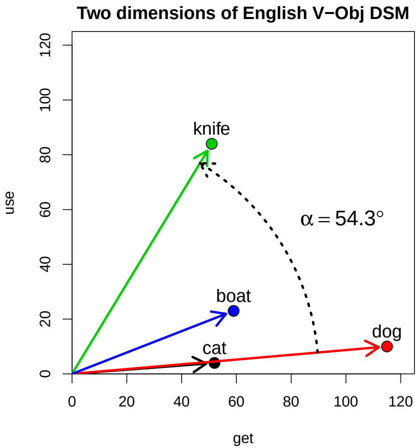
- A symmetric distance metric (Endres & Schindelin 2003)

$$D_{\mathbf{uv}} = D(\mathbf{u}\|\mathbf{z}) + D(\mathbf{v}\|\mathbf{z}) \quad \text{with} \quad \mathbf{z} = \frac{\mathbf{u}+\mathbf{v}}{2}$$

Similarity measures

- Angle α between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ is given by

$$\cos \alpha = \frac{\sum_{i=1}^{n} u_i \cdot v_i}{\sqrt{\sum_i u_i^2} \cdot \sqrt{\sum_i v_i^2}} = \frac{\mathbf{u}^T \mathbf{v}}{\|\mathbf{u}\|_2 \cdot \|\mathbf{v}\|_2}$$

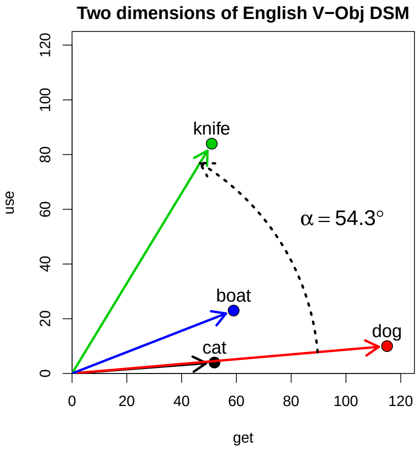


Similarity measures

- Angle α between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ is given by

$$\cos\alpha = \frac{\sum_{i=1}^{n} u_i \cdot v_i}{\sqrt{\sum_i u_i^2} \cdot \sqrt{\sum_i v_i^2}} = \frac{\mathbf{u}^T \mathbf{v}}{\|\mathbf{u}\|_2 \cdot \|\mathbf{v}\|_2}$$

- **cosine** measure of similarity: $\cos\alpha$
  - $\cos\alpha = 1$ ➜ collinear
  - $\cos\alpha = 0$ ➜ orthogonal
- Corresponding metric: **angular distance** α



Euclidean distance or cosine similarity?

$$d_2(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 = \sqrt{\sum_i (u_i - v_i)^2}$$

Euclidean distance or cosine similarity?

$$
\begin{aligned}
d_2(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 &= \sqrt{\sum_i (u_i - v_i)^2} \\
&= \sqrt{\sum_i u_i^2 + \sum_i v_i^2 - 2 \sum_i u_i v_i}
\end{aligned}
$$

Euclidean distance or cosine similarity?

$$
\begin{aligned}
d_2(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 &= \sqrt{\sum_i (u_i - v_i)^2} \\
&= \sqrt{\sum_i u_i^2 + \sum_i v_i^2 - 2 \sum_i u_i v_i} \\
&= \sqrt{\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2 - 2\, \mathbf{u}^T \mathbf{v}}
\end{aligned}
$$

Euclidean distance or cosine similarity?

$$
\begin{aligned}
d_2(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 &= \sqrt{\sum_i (u_i - v_i)^2} \\
&= \sqrt{\sum_i u_i^2 + \sum_i v_i^2 - 2\sum_i u_i v_i} \\
&= \sqrt{\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2 - 2\,\mathbf{u}^T\mathbf{v}} \\
&= \sqrt{2 - 2\cos\phi}
\end{aligned}
$$

☞ $d_2(\mathbf{u}, \mathbf{v})$ is a monotonically increasing function of ϕ

Euclidean distance or cosine similarity?

$$
\begin{aligned}
d_2(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 &= \sqrt{\sum_i (u_i - v_i)^2} \\
&= \sqrt{\sum_i u_i^2 + \sum_i v_i^2 - 2\sum_i u_i v_i} \\
&= \sqrt{\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2 - 2\,\mathbf{u}^T\mathbf{v}} \\
&= \sqrt{2 - 2\cos\phi}
\end{aligned}
$$

☞ $d_2(\mathbf{u}, \mathbf{v})$ is a monotonically increasing function of ϕ

Euclidean distance and cosine similarity are equivalent: if vectors have been normalised ($\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1$), both lead to the same neighbour ranking.

Similarity measures for non-negative vectors

- Generalized **Jaccard coefficient** = shared features

$$J(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1}^{n} \min\{u_i, v_i\}}{\sum_{i=1}^{n} \max\{u_i, v_i\}}$$

- $1 - J(\mathbf{u}, \mathbf{v})$ is a distance **metric** (Kosub 2016)

Similarity measures for non-negative vectors

- Generalized **Jaccard coefficient** = shared features

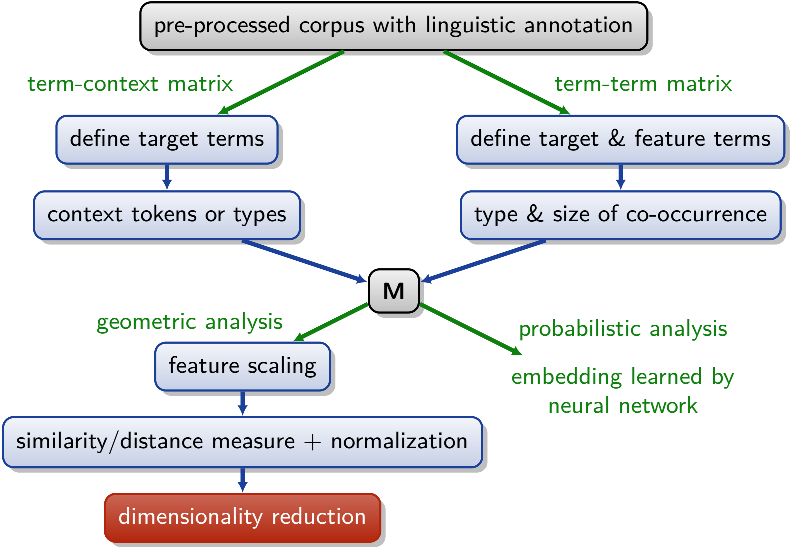
$$J(\mathbf{u},\mathbf{v}) = \frac{\sum_{i=1}^n \min\{u_i, v_i\}}{\sum_{i=1}^n \max\{u_i, v_i\}}$$

- $1 - J(\mathbf{u},\mathbf{v})$ is a distance **metric** (Kosub 2016)

- An asymmetric measure of feature **overlap** (Clarke 2009)

$$o(\mathbf{u},\mathbf{v}) = \frac{\sum_{i=1}^n \min\{u_i, v_i\}}{\sum_{i=1}^n u_i}$$

Overview of DSM parameters



Dimensionality reduction = model compression

- Co-occurrence matrix **M** is often unmanageably large and can be extremely sparse
  - Google Web1T5: 1M × 1M matrix with one trillion cells, of which less than 0.05% contain nonzero counts (Evert 2010)

➥ Compress matrix by reducing dimensionality (= rows)

Dimensionality reduction = model compression

- Co-occurrence matrix **M** is often unmanageably large and can be extremely sparse
  - Google Web1T5: 1M × 1M matrix with one trillion cells, of which less than 0.05% contain nonzero counts (Evert 2010)

➥ Compress matrix by reducing dimensionality (= rows)

- **Feature selection**: columns with high frequency & variance
  - measured by entropy, chi-squared test, nonzero count, …
  - may select similar dimensions and discard valuable information

Dimensionality reduction = model compression

- Co-occurrence matrix **M** is often unmanageably large and can be extremely sparse
  - Google Web1T5: 1M × 1M matrix with one trillion cells, of which less than 0.05% contain nonzero counts (Evert 2010)

➥ Compress matrix by reducing dimensionality (= rows)

- **Feature selection**: columns with high frequency & variance
  - measured by entropy, chi-squared test, nonzero count, . . .
  - may select similar dimensions and discard valuable information

- **Projection** into (linear) subspace
  - principal component analysis (PCA)
  - independent component analysis (ICA)
  - random indexing (RI)
  - ☞ intuition: preserve distances between data points

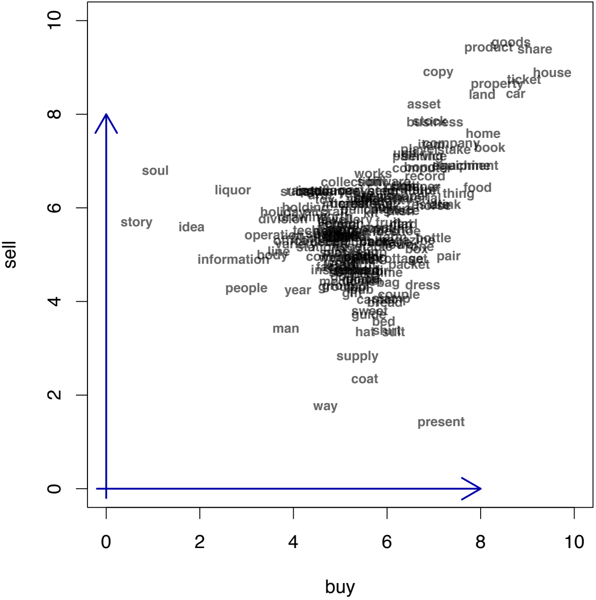
Dimensionality reduction & latent dimensions

Landauer & Dumais (1997) claim that LSA dimensionality reduction (and related PCA technique) uncovers **latent dimensions** by exploiting correlations between features.

- Example: term-term matrix
- V-Obj co-oc. extracted from BNC
  - targets = noun lemmas
  - features = verb lemmas
- feature scaling: association scores (SketchEngine log Dice)
- $k = 186$ nouns with $f_{\text{buy}} + f_{\text{sell}} \geq 25$
- $n = 2$ dimensions: *buy* and *sell*

noun	*buy*	*sell*
antique	5.12	5.50
bread	5.96	3.99
computer	6.75	6.83
factory	4.95	4.72
group	4.93	4.28
jewellery	5.11	5.73
mill	5.14	5.41
people	3.00	4.26
record	6.81	6.68
souvenir	5.45	4.67
ticket	8.93	8.74

Dimensionality reduction & latent dimensions



Motivating latent dimensions & subspace projection

- The **latent property** of being a commodity is "expressed" through associations with several verbs: *sell*, *buy*, *acquire*, ...
- Consequence: these DSM dimensions will be **correlated**

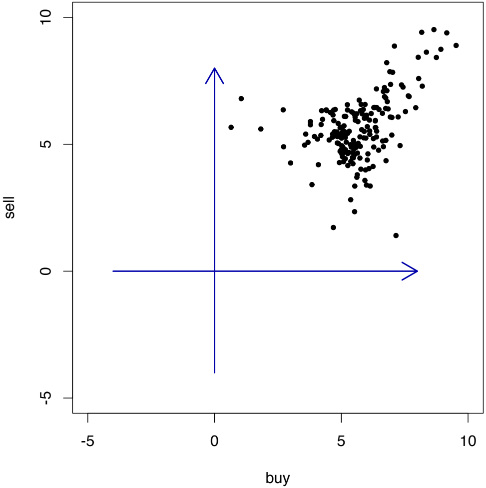
Motivating latent dimensions & subspace projection

- The **latent property** of being a commodity is "expressed" through associations with several verbs: *sell*, *buy*, *acquire*, ...
- Consequence: these DSM dimensions will be **correlated**

- Identify **latent dimension** by looking for strong correlations (or weaker correlations between large sets of features)
- Projection into subspace V of $k < n$ latent dimensions as a "**noise reduction**" technique ➜ **LSA**
- Assumptions of this approach:
  - "latent" distances in V are semantically meaningful
  - other "residual" dimensions represent chance co-occurrence patterns, often particular to the corpus underlying the DSM

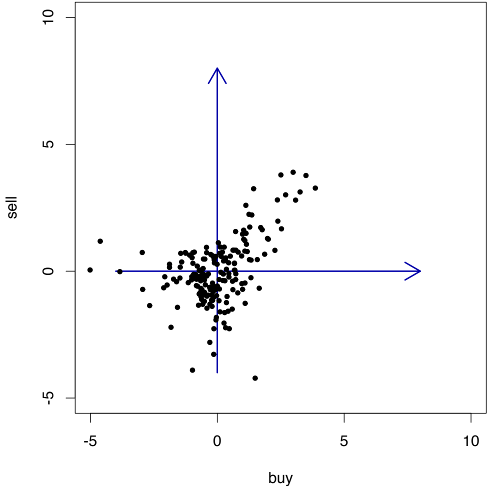
Step 1: Centering the data set

- **Uncentered data set**
- Centered data set
- Distance information = variance



Step 1: Centering the data set

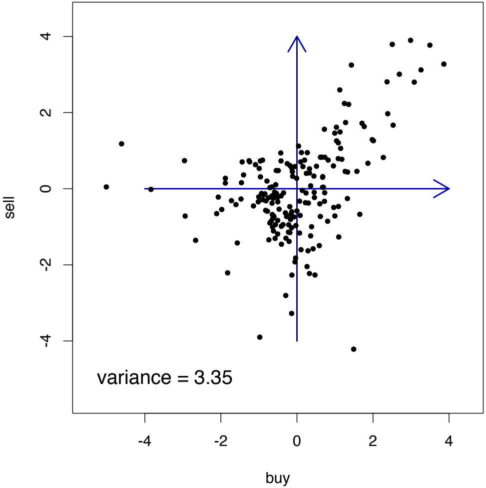
- Uncentered data set
- **Centered data set**
- Distance information = variance



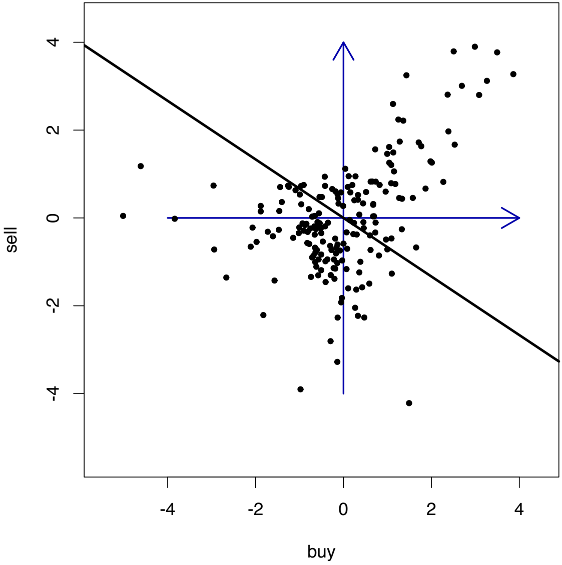
Step 1: Centering the data set

- Uncentered data set
- Centered data set
- Distance information = **variance**

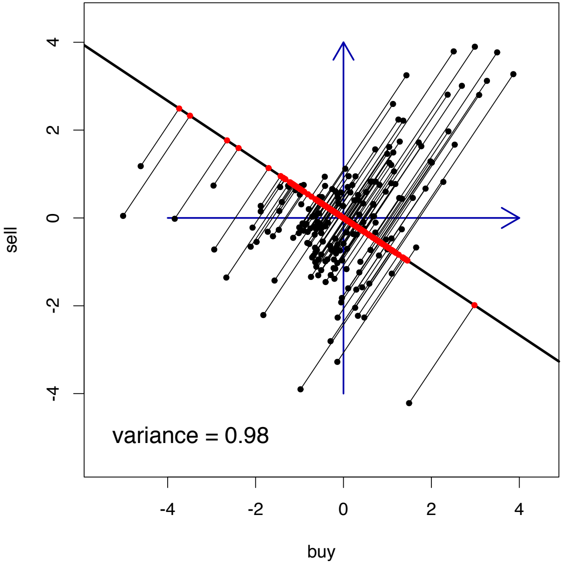
$$\sigma^2 = \frac{1}{k-1} \sum_{i=1}^{k} \|\mathbf{x}^{(i)}\|^2$$



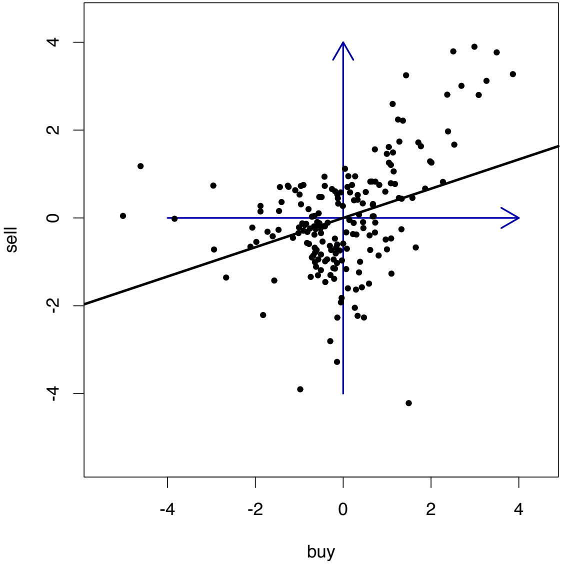
Step 2: Orthogonal projection into optimal subspace



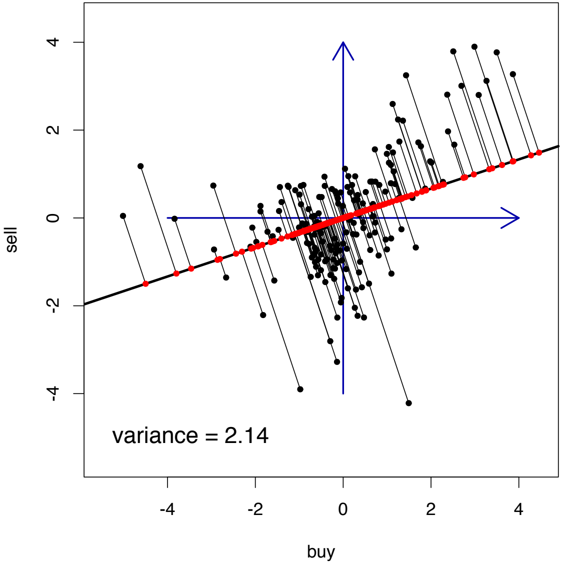
Step 2: Orthogonal projection into optimal subspace



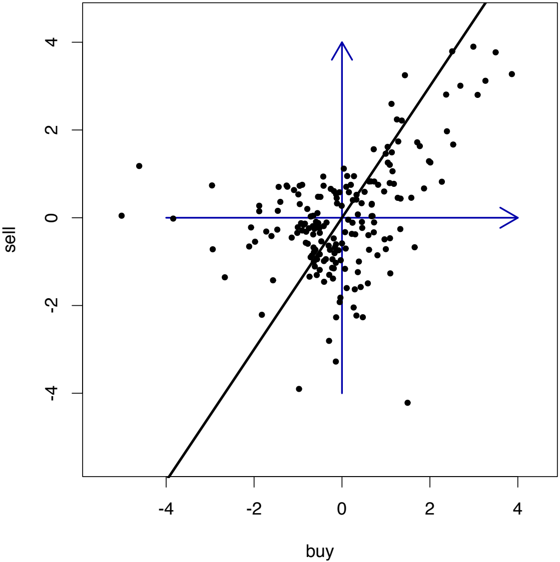
Step 2: Orthogonal projection into optimal subspace



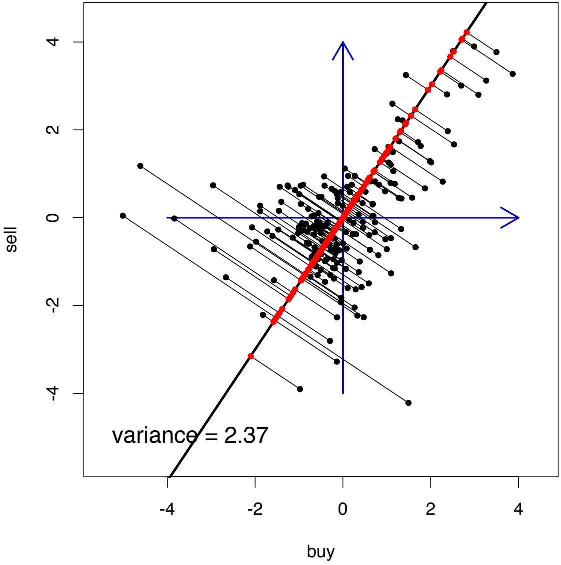
Step 2: Orthogonal projection into optimal subspace



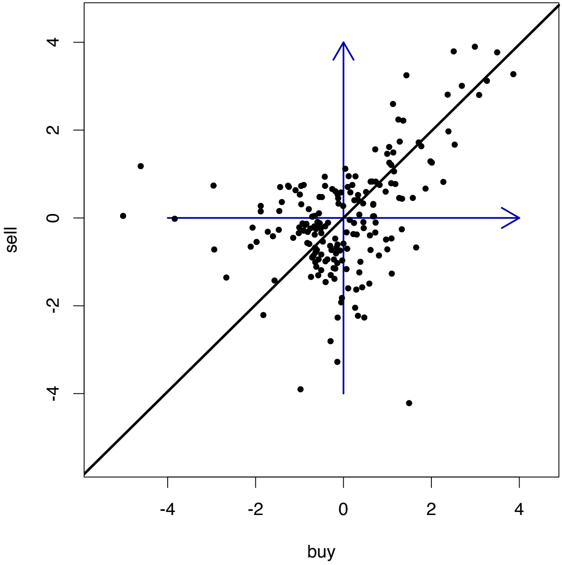
Step 2: Orthogonal projection into optimal subspace



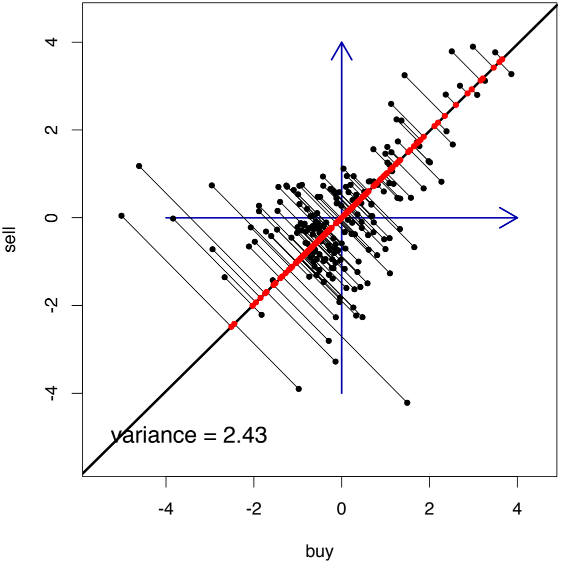
Step 2: Orthogonal projection into optimal subspace



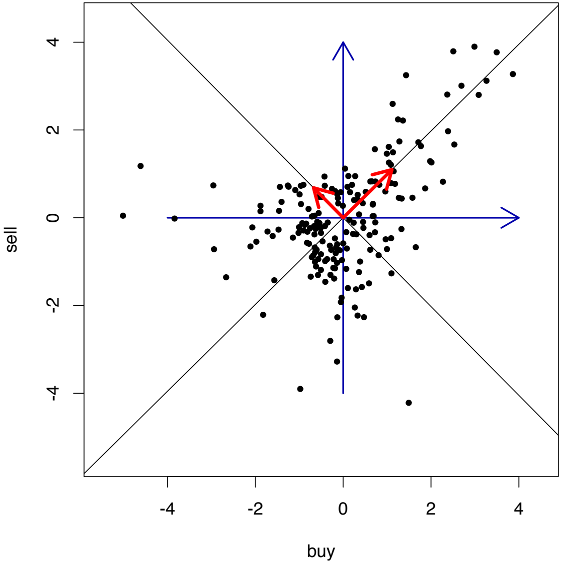
Step 2: Orthogonal projection into optimal subspace



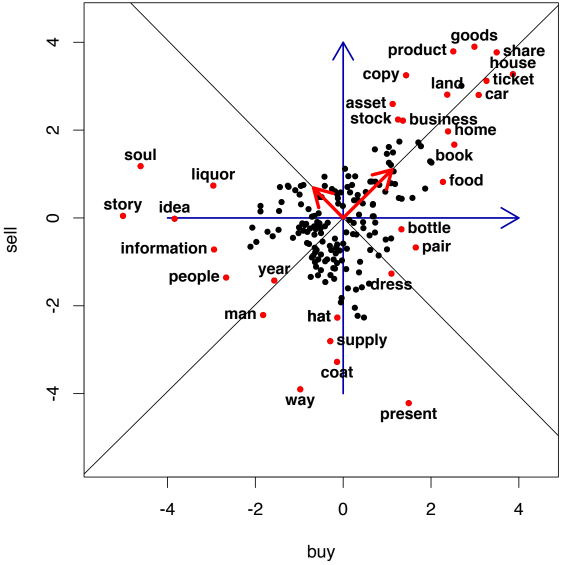
Step 2: Orthogonal projection into optimal subspace



Step 3: Further orthogonal dimensions



Step 3: Further orthogonal dimensions



Dimensionality reduction by PCA

- Principal component analysis (**PCA**)
  - orthogonal projection into orthogonal latent dimensions
  - finds optimal subspace of given dimensionality (such that orthogonal projection preserves distance information)
  - but requires centered features ➜ no longer sparse

Dimensionality reduction by PCA

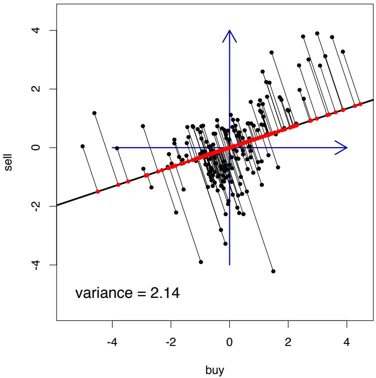
- Principal component analysis (**PCA**)
  - orthogonal projection into orthogonal latent dimensions
  - finds optimal subspace of given dimensionality (such that orthogonal projection preserves distance information)
  - but requires centered features ➜ no longer sparse
- Singular value decomposition (**SVD**)
  - the mathematical algorithm behind PCA
  - often applied without centering in distributional semantics
  - optimality of subspace not guaranteed (☞ part 5)

Dimensionality reduction by PCA

- Principal component analysis (**PCA**)
  - orthogonal projection into orthogonal latent dimensions
  - finds optimal subspace of given dimensionality (such that orthogonal projection preserves distance information)
  - but requires centered features ➜ no longer sparse
- Singular value decomposition (**SVD**)
  - the mathematical algorithm behind PCA
  - often applied without centering in distributional semantics
  - optimality of subspace not guaranteed (☞ part 5)
- NB: row vectors should be renormalised after PCA/SVD
  - unless cosine similarity / angular distance is used
  - ☞ also normalise vectors before dimensionality reduction

Dimensionality reduction by RI

- Random indexing (**RI**)
  - project into random subspace (Sahlgren & Karlgren 2005)
  - reasonably good if there are many subspace dimensions
  - can be performed online w/o collecting full co-oc. matrix



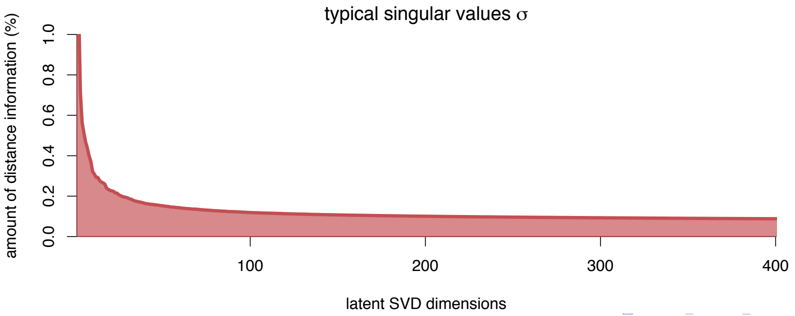
Dimensionality reduction in practice

```
# it is customary to omit the centering: SVD dimensionality reduction
> TT2 <- dsm.projection(TT, n=2, method="svd")
> TT2
          svd1    svd2
cat     -0.733 -0.6615
dog     -0.782 -0.6110
animal  -0.914 -0.3606
time    -0.993  0.0302
reason  -0.889  0.4339
cause   -0.817  0.5615
effect  -0.871  0.4794

> x <- TT2[, 1] # first latent dimension
> y <- TT2[, 2] # second latent dimension
> plot(x, y, pch=20, col="red",
       xlim=extendrange(x), ylim=extendrange(y))
> text(x, y, rownames(TT2), pos=3)
```

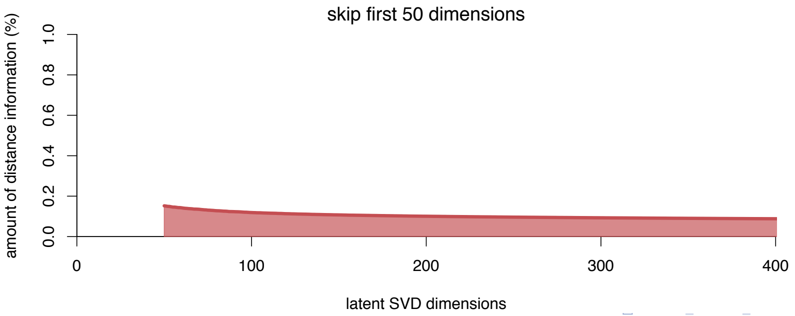
Scaling latent dimensions

- Capture different amounts of distance info (= variance)
- Indicated by **singular values** σ_i of PCA/SVD algorithm



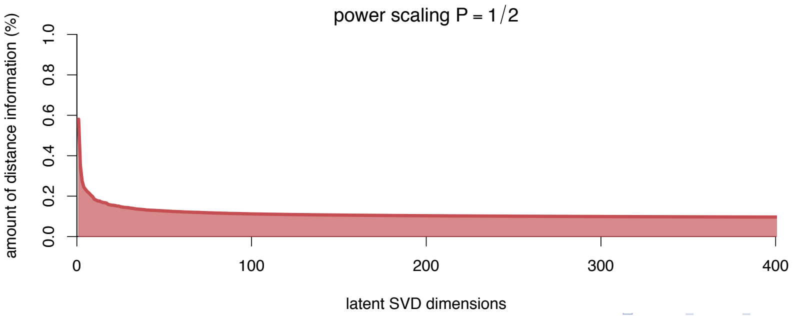
Scaling latent dimensions

- Capture different amounts of distance info (= variance)
- Indicated by **singular values** σ_i of PCA/SVD algorithm
- Skip first k dimensions, e.g. $k = 50$ (Bullinaria & Levy 2012)



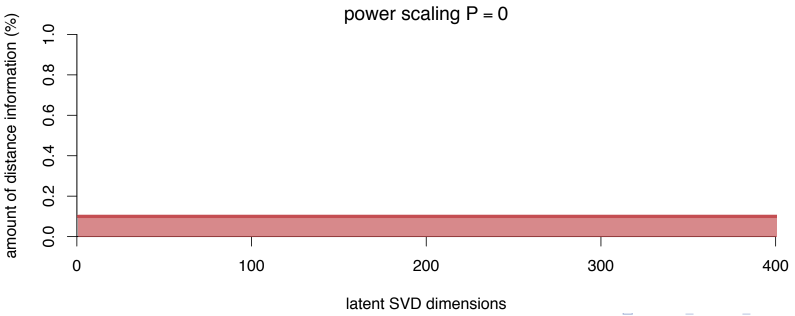
Scaling latent dimensions

- Capture different amounts of distance info (= variance)
- Indicated by **singular values** σ_i of PCA/SVD algorithm
- Skip first k dimensions, e.g. $k = 50$ (Bullinaria & Levy 2012)
- Power-scaling of dimensions: σ^P (Caron 2001)
  - Bullinaria & Levy (2012) report positive effect



Scaling latent dimensions

- Capture different amounts of distance info (= variance)
- Indicated by **singular values** σ_i of PCA/SVD algorithm
- Skip first k dimensions, e.g. $k = 50$ (Bullinaria & Levy 2012)
- Power-scaling of dimensions: σ^P (Caron 2001)
  - Bullinaria & Levy (2012) report positive effect
  - esp. with $P = 0$ to equalize dimensions (**whitening**)



Power-scaling in practice

```
> TT2 <- dsm.projection(TT, n=2, method="svd", power=0)
> TT2
          svd1    svd2
cat    -0.322 -0.5110
dog    -0.343 -0.4721
animal -0.401 -0.2786
time   -0.436  0.0233
reason -0.390  0.3353
cause  -0.359  0.4338
effect -0.383  0.3704

# power-scaling can also be applied post-hoc
> sigma <- attr(TT2, "sigma")        # singular values
> scaleMargins(TT2, cols=sigma^0.5)  # P = 1/2
> scaleMargins(TT2, cols=sigma)      # unscaled (P = 1)
```

Outline

Some well-known DSM examples

Latent Semantic Analysis (Landauer & Dumais 1997)

- term-context matrix with document context
- weighting: log term frequency and term entropy
- distance measure: cosine
- dimensionality reduction: SVD

Some well-known DSM examples

Latent Semantic Analysis (Landauer & Dumais 1997)

- term-context matrix with document context
- weighting: log term frequency and term entropy
- distance measure: cosine
- dimensionality reduction: SVD

Hyperspace Analogue to Language (Lund & Burgess 1996)

- term-term matrix with surface context
- structured (left/right) and distance-weighted frequency counts
- distance measure: Minkowski metric ($1 \leq p \leq 2$)
- dimensionality reduction: feature selection (high variance)

Some well-known DSM examples

Infomap NLP (Widdows 2004)

- term-term matrix with unstructured surface context
- weighting: none
- distance measure: cosine
- dimensionality reduction: SVD

Some well-known DSM examples

Infomap NLP (Widdows 2004)

- term-term matrix with unstructured surface context
- weighting: none
- distance measure: cosine
- dimensionality reduction: SVD

Random Indexing (Karlgren & Sahlgren 2001)

- term-term matrix with unstructured surface context
- weighting: various methods
- distance measure: various methods
- dimensionality reduction: random indexing (RI)

Some well-known DSM examples

Dependency Vectors (Padó & Lapata 2007)

- term-term matrix with unstructured dependency context
- weighting: log-likelihood ratio
- distance measure: PPMI-weighted Dice (Lin 1998)
- dimensionality reduction: none

Some well-known DSM examples

Dependency Vectors (Padó & Lapata 2007)

- term-term matrix with unstructured dependency context
- weighting: log-likelihood ratio
- distance measure: PPMI-weighted Dice (Lin 1998)
- dimensionality reduction: none

Distributional Memory (Baroni & Lenci 2010)

- term-term matrix with structured and unstructured dependencies + knowledge patterns
- weighting: local-MI on type frequencies of link patterns
- distance measure: cosine
- dimensionality reduction: none

. . . and an unexpected application

Authorship attribution (Burrows 2002)

- Burrows's Delta method is very popular in modern literary stylometry and authorship attribution (Evert *et al.* 2017)
- document-term matrix with word forms as features
- weighting: relative frequency of word form in document
- feature selection: 200–5,000 most frequent words (mfw)
- columns are standardized ($\mu = 0$, $\sigma^2 = 1$) ➜ z-scores
- clustering of documents based on various distance metrics (or nearest-neighbour classifier for known authors)
- dimensionality reduction: none
- main result: angle/cosine $\succ$ Manhattan $\succ$ Euclidean

Outline

Scaling up to the real world

- So far, we have worked on minuscule **toy models**
- ☞ We want to scale up to **real world** data sets now

Scaling up to the real world

- So far, we have worked on minuscule **toy models**

☞ We want to scale up to **real world** data sets now

- Example 1: window-based DSM on BNC content words
  - 83,926 lemma types with $f \geq 10$
  - term-term matrix with $83{,}926 \cdot 83{,}926 = 7$ billion entries
  - standard representation requires 56 GB of RAM (8-byte floats)
  - only 22.1 million non-zero entries ($= 0.32\%$)

Scaling up to the real world

- So far, we have worked on minuscule **toy models**
- ☞ We want to scale up to **real world** data sets now

- Example 1: window-based DSM on BNC content words
  - 83,926 lemma types with $f \geq 10$
  - term-term matrix with $83{,}926 \cdot 83{,}926 = 7$ billion entries
  - standard representation requires 56 GB of RAM (8-byte floats)
  - only 22.1 million non-zero entries ($= 0.32\%$)

- Example 2: Google Web 1T 5-grams (1 trillion words)
  - more than 1 million word types with $f \geq 2500$
  - term-term matrix with 1 trillion entries requires 8 TB RAM
  - only 400 million non-zero entries ($= 0.04\%$)

Sparse matrix representation

- Invented example of a **sparsely populated** DSM matrix

	eat	get	hear	kill	see	use
boat	·	59	·	·	39	23
cat	·	·	·	26	58	·
cup	·	98	·	·	·	·
dog	33	·	42	·	83	·
knife	·	·	·	·	·	84
pig	9	·	·	27	·	·

Sparse matrix representation

- Invented example of a **sparsely populated** DSM matrix

	eat	get	hear	kill	see	use
boat	·	59	·	·	39	23
cat	·	·	·	26	58	·
cup	·	98	·	·	·	·
dog	33	·	42	·	83	·
knife	·	·	·	·	·	84
pig	9	·	·	27	·	·

- Store only non-zero entries in compact **sparse matrix format**

row	col	value
1	2	59
1	5	39
1	6	23
2	4	26
2	5	58
3	2	98

row	col	value
4	1	33
4	3	42
4	5	83
5	6	84
6	1	9
6	4	27

Working with sparse matrices

- Compressed format: each row index (or column index) stored only once, followed by non-zero entries in this row (or column)
  - convention: **column-major** matrix (data stored by columns)
- Specialised algorithms for sparse matrix algebra
  - especially matrix multiplication, solving linear systems, etc.
  - take care to avoid operations that create a dense matrix!

Working with sparse matrices

- Compressed format: each row index (or column index) stored only once, followed by non-zero entries in this row (or column)
  - convention: **column-major** matrix (data stored by columns)
- Specialised algorithms for sparse matrix algebra
  - especially matrix multiplication, solving linear systems, etc.
  - take care to avoid operations that create a dense matrix!
- **R** implementation: `Matrix` package
  - essential for real-life distributional semantics
  - `wordspace` provides additional support for sparse matrices (vector distances, sparse SVD, ...)
- Other software: Matlab, Octave, Python + SciPy

Outline

Triplet tables

- A sparse DSM matrix can be represented as a table of triplets (target, feature, co-occurrence frequency)
  - for syntactic co-occurrence and term-document matrices, marginals can be computed from a complete triplet table
  - for surface and textual co-occurrence, marginals have to be provided in separate files (see `?read.dsm.triplet`)

noun	rel	verb	f	mode
dog	subj	bite	3	spoken
dog	subj	bite	12	written
dog	obj	bite	4	written
dog	obj	stroke	3	written
...	...	...	...	...

- `DSM_VerbNounTriples_BNC` contains additional information
  - syntactic relation between noun and verb
  - written or spoken part of the British National Corpus

Constructing a DSM from a triplet table

- Additional information can be used for filtering (verb-object relation), or aggregate frequencies (spoken + written BNC)

```
> tri <- subset(DSM_VerbNounTriples_BNC, rel == "obj")
```

- Construct DSM object from triplet input
  - `raw.freq=TRUE` indicates raw co-occurrence frequencies (rather than a pre-weighted DSM)
  - constructor aggregates counts from duplicate entries
  - marginal frequencies are automatically computed

```
> VObj <- dsm(target=tri$noun, feature=tri$verb,
              score=tri$f, raw.freq=TRUE)
> VObj  # inspect marginal frequencies (e.g. head(VObj$rows, 20))
```

Exploring the DSM

```
> VObj <- dsm.score(VObj, score="MI", normalize=TRUE)

> nearest.neighbours(VObj, "dog") # angular distance
   horse       cat    animal    rabbit      fish       guy
    73.9      75.9      76.2      77.0      77.2      78.5
 cichlid       kid       bee  creature
    78.6      79.0      79.1      79.5

> nearest.neighbours(VObj, "dog", method="manhattan")
# NB: we used an incompatible Euclidean normalization!

> VObj50 <- dsm.projection(VObj, n=50, method="svd")
> nearest.neighbours(VObj50, "dog")
```

Practice

- How many different models can you build from `DSM_VerbNounTriples_BNC`?
- Apply different filters, scores, transformations and metrics
  - ☞ explore nearest neighbours of selected words

- Code examples for this part show additional options

- Download practice session (`part2_input_formats.R`)
  ➜ different ways of loading your own co-occurrence data
- Build real-life DSMs from pre-compiled co-occurrence data
  - http://wordspace.collocations.de/doku.php/course:material
  - also download R script with instructions (`part2_exercise.R`)

References I

Baroni, Marco and Lenci, Alessandro (2010). Distributional Memory: A general framework for corpus-based semantics. *Computational Linguistics*, **36**(4), 673–712.

Blei, David M.; Ng, Andrew Y.; Jordan, Michael, I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, **3**, 993–1022.

Bullinaria, John A. and Levy, Joseph P. (2007). Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, **39**(3), 510–526.

Bullinaria, John A. and Levy, Joseph P. (2012). Extracting semantic representations from word co-occurrence statistics: Stop-lists, stemming and SVD. *Behavior Research Methods*, **44**(3), 890–907.

Burrows, John (2002). 'Delta': a measure of stylistic difference and a guide to likely authorship. *Literary and Linguistic Computing*, **17**(3), 267–287.

Caron, John (2001). Experiments with LSA scoring: Optimal rank and basis. In M. W. Berry (ed.), *Computational Information Retrieval*, pages 157–169. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Clarke, Daoud (2009). Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 112–119, Athens, Greece.

References II

Endres, Dominik M. and Schindelin, Johannes E. (2003). A new metric for probability distributions. *IEEE Transactions on Information Theory*, **49**(7), 1858–1860.

Evert, Stefan (2004). *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Dissertation, Institut für maschinelle Sprachverarbeitung, University of Stuttgart.

Evert, Stefan (2008). Corpora and collocations. In A. Lüdeling and M. Kytö (eds.), *Corpus Linguistics. An International Handbook*, chapter 58, pages 1212–1248. Mouton de Gruyter, Berlin, New York.

Evert, Stefan (2010). Google Web 1T5 n-grams made easy (but not for the computer). In *Proceedings of the 6th Web as Corpus Workshop (WAC-6)*, pages 32–40, Los Angeles, CA.

Evert, Stefan; Proisl, Thomas; Jannidis, Fotis; Reger, Isabella; Pielström, Steffen; Schöch, Christof; Vitt, Thorsten (2017). Understanding and explaining Delta measures for authorship attribution. *Digital Scholarship in the Humanities*, **22**(suppl_2), ii4–ii16.

Hoffmann, Thomas (1999). Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*.

References III

Karlgren, Jussi and Sahlgren, Magnus (2001). From words to understanding. In Y. Uesaka, P. Kanerva, and H. Asoh (eds.), *Foundations of Real-World Intelligence*, chapter 294–308. CSLI Publications, Stanford.

Kosub, Sven (2016). A note on the triangle inequality for the Jaccard distance. *CoRR*, **abs/1612.02696**.

Landauer, Thomas K. and Dumais, Susan T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, **104**(2), 211–240.

Levy, Omer and Goldberg, Yoav (2014). Neural word embedding as implicit matrix factorization. In *Proceedings of Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Levy, Omer; Goldberg, Yoav; Dagan, Ido (2015). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, **3**, 211–225.

Lin, Dekang (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING-ACL 1998)*, pages 768–774, Montreal, Canada.

References IV

Lund, Kevin and Burgess, Curt (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, **28**(2), 203–208.

Padó, Sebastian and Lapata, Mirella (2007). Dependency-based construction of semantic space models. *Computational Linguistics*, **33**(2), 161–199.

Polajnar, Tamara and Clark, Stephen (2014). Improving distributional semantic vectors through context selection and normalisation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 230–238, Gothenburg, Sweden.

Rooth, Mats; Riezler, Stefan; Prescher, Detlef; Carroll, Glenn; Beil, Franz (1999). Inducing a semantically annotated lexicon via EM-based clustering. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

Sahlgren, Magnus and Karlgren, Jussi (2005). Automatic bilingual lexicon acquisition using random indexing of parallel corpora. *Natural Language Engineering*, **11**, 327–341.

Widdows, Dominic (2004). *Geometry and Meaning*. Number 172 in CSLI Lecture Notes. CSLI Publications, Stanford.