Using Corpora in Social Science Research

Andreas Blaette

2018-01-23

Contents

Chapter 1

Introduction

1.1 Corpora in Social Science Research

1.1.1 Corpora, Big Data, and Text Mining

1.1.2 Corpus and Content Analysis

1.2 Getting Started

1.2.1 Hardware: Know Your Machine

1.2.2 Software: Tools, Toolkits, Frameworks

Chapter 2

Corpora

2.1 Corpora as Linguistic Data for Social Science

2.1.1 Corpora that Exist and Might Exits

2.1.2 Corpora and the Wealth of Text

2.2 Data Formats

2.2.1 The Beauty and Limits of Plain Text

2.2.2 Extended Markup Language (XML)

2.2.2.1 Standardization: Text Encoding Initiative (TEI)

2.2.2.2 Alternative Standards

2.2.3 From Words to Numbers: Indexing and Query Engines

2.3 Annotated Corpora

2.3.1 Linguistic Annotation

2.3.2 Structural Annotation (Metadata)

2.3.3 Subcorpora, Contrastive and Comparative Research

2.4 The Licencensing Issue

Chapter 3

Queries and Contexts

3.1 Formulating Queries

3.1.1 Basics of Regulary Expression Syntax

3.1.2 The Corpus Query Processor

3.2 Concordances and KWIC Analysis

3.2.1 The Art and Science of Concordancing

3.2.2 A Tool for Validation?

Chapter 4

The Art of Counting

Counting is a very basic task when working with text. Counting is essentially simple: It is nothing but making statements how often a word, or words, or linguistic patterns occurr in a corpus, or a subcorpus. However, a social scientist counting words or linguistic patterns should see counting is measuring. Counts are stable and objective, the reliability and intersubjectivity of the measurement are minor problems. Words and linguistic patterns can be observed directly, many issues associated with measuring latent variables do not arise. But counting is not trivial. Without conscious reflection, what is actually measured by counting lexical items, without an awareness that counting is measuring and needs to meet standards of validity, a simple move "from words to numbers" (. . .) can violate fundamental concerns of social science. Counting without methodological reflection will lead to flawed and bad science.

If done well, counting can be powerful instruments. Corpora offer efficient ways to extract information from vast amounts of text efficiently. In particular, when we move from counts of patterns in one subcorpus to the dispersion of counts in one or two dimensions, the basis for comparative statements that involve a synchronic or diachronic dimension can be generated quickly. Sometimes, simply counting single words may do. Often, we may want to count more complex linguistic patterns. The following sections will discuss the move from counts to dispersions.

The corpus used for the examples is the GermaParl corpus. Before moving on, load the polmineR library, and activate the corpus.

```
library(polmineR)
use("GermaParl")
```

```
## ... resetting CORPUS_REGISTRY environment variable:

## ... setting registry: /Library/Frameworks/R.framework/Versions/3.4/Resources/library/GermaParl/extda

## ... unloading rcqp library

## ... reloading rcqp library

## ... ... status: OK
```

4.1 The Basics of Counting

4.2 Counting Words

The method in the polmineR-package to perform counts, it is not necessarily a surprise, is called 'count'. It can be used for a variety of scenarios. To learn about the uses of the count-method, call the documentation (i.e. the help page) for the method.

```
?count
```

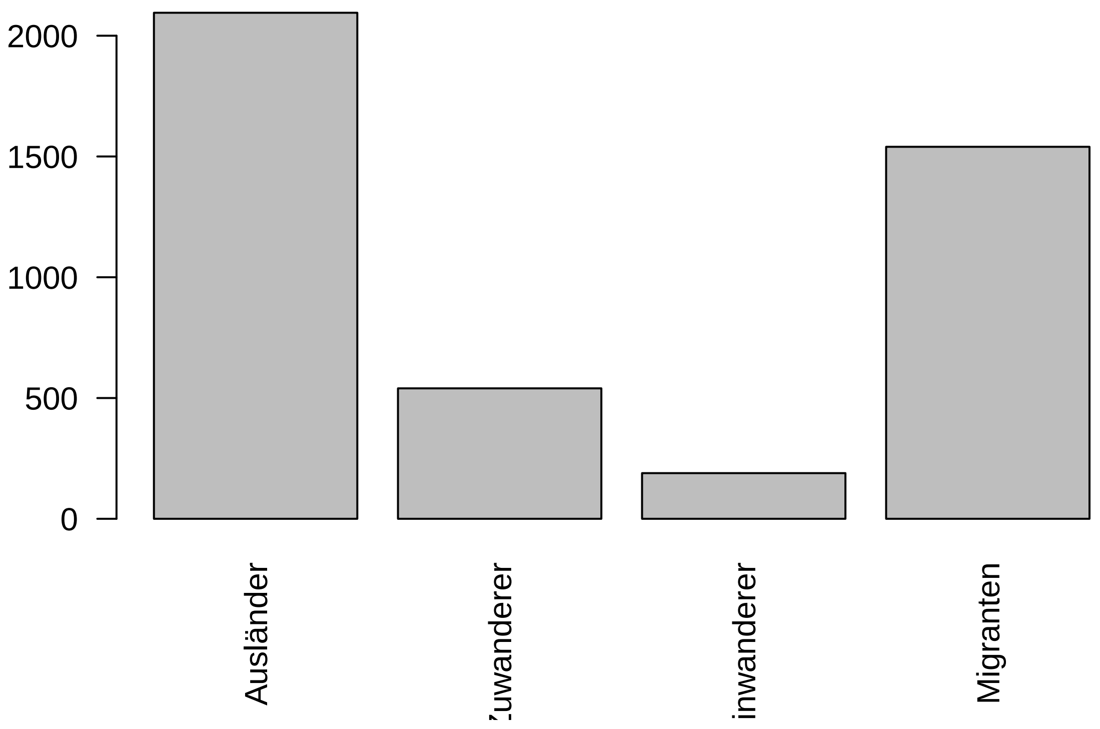
Count can be applied to different objects. We start by

```
count("GERMAPARL", query = "Zuwanderer")
```

```
##          query count         freq
## 1: Zuwanderer   540 5.345809e-06
```

```
auslaender <- count("GERMAPARL", query = "Ausländer")[["count"]]
zuwanderer <- count("GERMAPARL", query = "Zuwanderer")[["count"]]
einwanderer <- count("GERMAPARL", query = "Einwanderer")[["count"]]
migranten <- count("GERMAPARL", query = "Migranten")[["count"]]
gastarbeiter <- count("GERMAPARL", query = "Gastarbeiter")[["count"]]
```

```
barplot(
  c(auslaender, zuwanderer, einwanderer, migranten),
  names.arg = c("Ausländer", "Zuwanderer", "Einwanderer", "Migranten"),
  las = 2, cex.axis = 1,
)
```



```
y <- dispersion("GERMAPARL", query = "Flüchtlinge", sAttribute = "year")
```

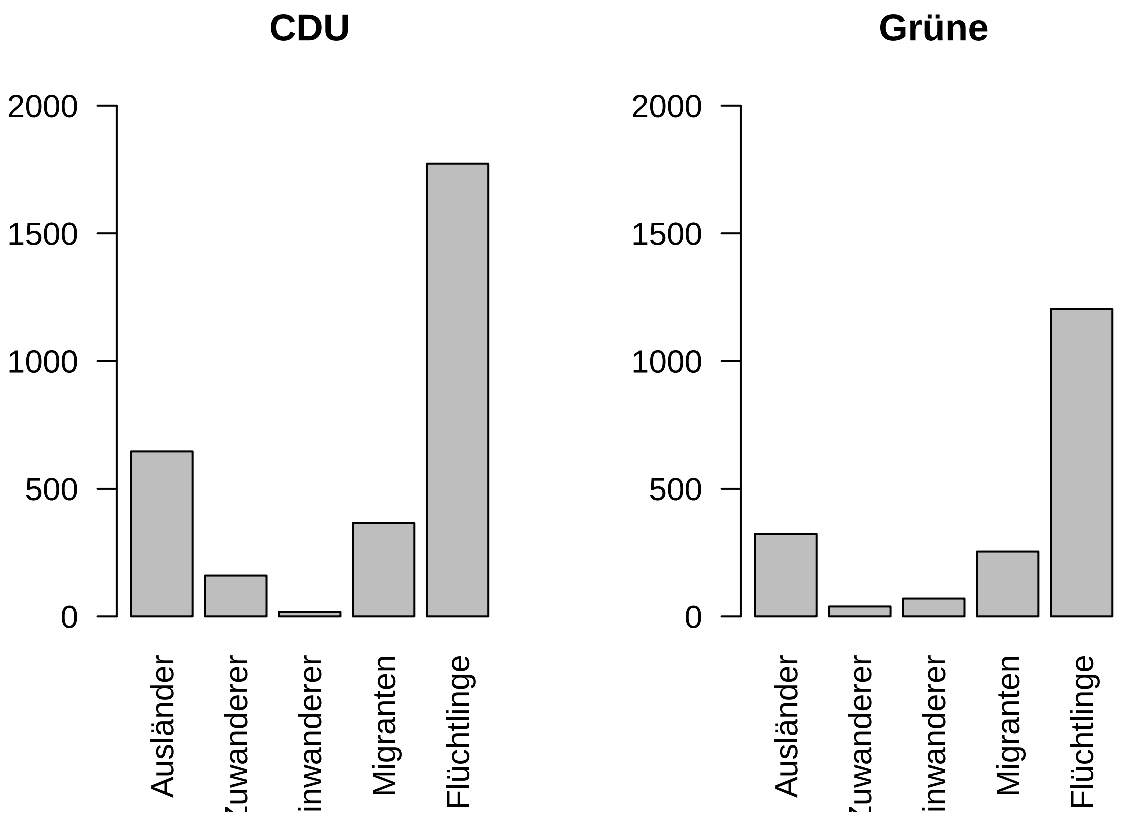
```
cdu <- partition("GERMAPARL", party = "CDU")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1

## ... get cpos and strucs

## ... get partition size
```

```
gruene <- partition("GERMAPARL", party = "GRUENE")
```

```
## ... Setting up partition
## ... type of the corpus is plpr
## ... get encoding: latin1
## ... get cpos and strucs
## ... get partition size
```

```
cnt_cdu <- count(
  cdu,
  query = c("Ausländer", "Zuwanderer", "Einwanderer", "Migranten", "Flüchtlinge")
  )
cnt_gruene <- count(
  gruene,
  query = c("Ausländer", "Zuwanderer", "Einwanderer", "Migranten", "Flüchtlinge")
  )

par(mfrow = c(1,2))
barplot(
  cnt_cdu[["count"]], names.arg = cnt_cdu[["query"]], las = 2,
  ylim = c(0,2000), main = "CDU"
  )
barplot(
  cnt_gruene[["count"]], names.arg = cnt_gruene[["query"]], las = 2,
  ylim = c(0,2000), main = "Grüne"
  )
```



Now we move to regular expressions.

```
count("GERMAPARL", '"Multikult.*"', cqp = TRUE)
```

```
##            query count         freq
## 1: "Multikult.*"   101 9.998643e-07
```

```
count("GERMAPARL", '"Multikult.*"', cqp = TRUE, breakdown = TRUE)
```

```
##             query                        match count share
##  1: "Multikult.*"                   Multikulti    48 47.52
##  2: "Multikult.*"           Multikulturalität     6  5.94
##  3: "Multikult.*"           Multikultiromantik     5  4.95
##  4: "Multikult.*"             Multikultidenken     3  2.97
##  5: "Multikult.*"       Multikultigesellschaft     3  2.97
##  6: "Multikult.*"          Multikulturalismus     3  2.97
##  7: "Multikult.*"          Multikulti-Ideologie     2  1.98
##  8: "Multikult.*"             Multikultibasar     2  1.98
##  9: "Multikult.*"           Multikultiideologie     2  1.98
## 10: "Multikult.*"            Multikultipolitik     2  1.98
## 11: "Multikult.*"          Multikultiseligkeit     2  1.98
## 12: "Multikult.*"             Multikultitruppe     2  1.98
## 13: "Multikult.*"          Multikulti-fantasien     1  0.99
## 14: "Multikult.*"           Multikulti-Gesäusel     1  0.99
## 15: "Multikult.*"              Multikulti-Gout     1  0.99
## 16: "Multikult.*"            Multikulti-Irrtum     1  0.99
## 17: "Multikult.*" Multikulti-Kuschelpädagogik     1  0.99
## 18: "Multikult.*"        Multikulti-Mannschaft     1  0.99
## 19: "Multikult.*"        Multikulti-Mischmasch     1  0.99
## 20: "Multikult.*"         Multikulti-Spektakel     1  0.99
## 21: "Multikult.*"         Multikulti-Sportclub     1  0.99
## 22: "Multikult.*"           Multikultidelirium     1  0.99
## 23: "Multikult.*"        Multikultiexperimente     1  0.99
## 24: "Multikult.*"          Multikultigläubigen     1  0.99
## 25: "Multikult.*"           Multikultikonzepte     1  0.99
## 26: "Multikult.*"         Multikultimannschaft     1  0.99
## 27: "Multikult.*"        Multikultiseligkeiten     1  0.99
## 28: "Multikult.*"            Multikultisierung     1  0.99
## 29: "Multikult.*"             Multikultiträume     1  0.99
## 30: "Multikult.*"        Multikultiträumereien     1  0.99
## 31: "Multikult.*"                  Multikultur     1  0.99
## 32: "Multikult.*"               Multikulturell     1  0.99
## 33: "Multikult.*"              Multikulturhaus     1  0.99
##             query                        match count share
```

```
sAttributes("GERMAPARL", "party")
```

```
##  [1] "CDU"          "FDP"          "SPD"          "CSU"
##  [5] "GRUENE"       "PDS"          ""             "parteilos"
##  [9] "LINKE"        "fraktionslos"
```

```
csu <- partition("GERMAPARL", party = "CSU")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1
## ... get cpos and strucs
## ... get partition size
```

```r
cdu <- partition("GERMAPARL", party = "CDU")
```

```
## ... Setting up partition
## ... type of the corpus is plpr
## ... get encoding: latin1
## ... get cpos and strucs
## ... get partition size
```

```r
spd <- partition("GERMAPARL", party = "SPD")
```

```
## ... Setting up partition
## ... type of the corpus is plpr
## ... get encoding: latin1
## ... get cpos and strucs
## ... get partition size
```

```r
gru <- partition("GERMAPARL", party = "GRUENE")
```

```
## ... Setting up partition
## ... type of the corpus is plpr
## ... get encoding: latin1
## ... get cpos and strucs
## ... get partition size
```

4.3 Multi-word expressions and complex queries

```r
count("GERMAPARL", query = '"Menschen" "mit" "Migrationshintergrund"')
```

```
##                                      query count         freq
## 1: "Menschen" "mit" "Migrationshintergrund"   302 2.989693e-06
```

```r
cnt <- count("GERMAPARL", query = '[pos = "ADJA"] "Integration"', breakdown = TRUE)
head(cnt, 10)
```

```
##                           query                          match count share
##  1: [pos = "ADJA"] "Integration"      europäischen Integration   482 19.55
##  2: [pos = "ADJA"] "Integration"       europäische Integration   421 17.07
##  3: [pos = "ADJA"] "Integration"      erfolgreiche Integration    81  3.28
##  4: [pos = "ADJA"] "Integration"          bessere Integration    70  2.84
##  5: [pos = "ADJA"] "Integration"          soziale Integration    64  2.60
##  6: [pos = "ADJA"] "Integration"        gelungene Integration    47  1.91
##  7: [pos = "ADJA"] "Integration"         sozialen Integration    38  1.54
##  8: [pos = "ADJA"] "Integration"      europäischer Integration    36  1.46
##  9: [pos = "ADJA"] "Integration"       politische Integration    35  1.42
## 10: [pos = "ADJA"] "Integration" gesellschaftliche Integration    34  1.38
```

4.3.1 From Counts to Dispersion

4.3.2 Visualising Dispersions

4.3.2.1 Comparing Subcorpora

4.3.2.2 Time-Series Analysis

Chapter 5

Cooccurrences and Neighborhoods

The neighborhoods of words, and more generally, of lexical items are interesting for many reason. What patterns occurr in a context of 5, 10 or 15 words to the left and to the right of a word, or the match of a query? "You shall know a word by the neighborhood it keeps" (Firth 1957:11) is a frequently quoted, programmatic statement in linguistics. It has inspired statistical approaches to analyse word contexts. In fact, computing collocations is a core technique in corpus linguistics, and extremely productive to gain insights into to the way language is used in a data-driven manner. Analysing collocations has become an essential tool in lexicography, for instance.

There are many reasons why social scientists are fascinated by the idea to adapt working with collocations. Collocations come with the promise that the framing of groups and issues may be explored efficiently and productively. But social scientists usually have different research interests than corpus linguists. Rules of thumb and choices that guide corpus linguists in a reasoned and established manner may deserve reconsideration, as we relocate analysing collocations to another discipline. This concerns word context size and statistical measures, for instance.

The most important concern of this chapter on the analysis of word context is that deriving statements about meaning from a statistical identification of cooccurrences requires interpretation. This interpretation cannot be derived from statistical measures, but from reading relevant passages of text the statistical measures indicate as being potentially interesting. Moving from words to numbers is extremely productive to reveal patterns, but the interpretation of patterns will require moving from numbers to word again.

Before we look into cooccurrences, a remark on terminology. The same statistical procedures are used to identify collocations and cooccurrences. The statement that a word is a collocate is an observation about a linguistic pattern, the stickiness of one word towards another. But the concern here is not linguistic patterns. Just as social scientists are careful not to confuse correlation and causation, the more technical, and less substantial term 'cooccurrences' somewhat pushes us to see that findings about cooccurrences are merely statements about a statistical regularity - in need of further inquiry to obtain substantial statements.

```r
library(polmineR)
use("GermaParl")
```

```
## ... resetting CORPUS_REGISTRY environment variable:

## ... setting registry: /Library/Frameworks/R.framework/Versions/3.4/Resources/library/GermaParl/extda

## ... unloading rcqp library

## ... reloading rcqp library

## ... ... status: OK
```

In addition to polmineR, we will use the packages, tm, magrittr and wordcloud.

```r
if (!"tm" %in% available.packages()[,"Package"]) install.packages("tm")
if (!"magrittr" %in% available.packages()[,"Package"]) install.packages("magrittr")
if (!"wordcloud" %in% available.packages()[,"Package"]) install.packages("wordcloud")
```

```r
library(tm)
library(wordcloud)
library(magrittr)
```

We use different context sizes.

```r
cooccurrences("GERMAPARL", query = "Islam")
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", query = "Islam", left = 10, right = 10)
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", query = "Islam", left = 15, right = 15)
```

```
## ... using polmineR.Rcpp for counting
```

Set the context size to a value (15) for the session.

```r
options("polmineR.left" = 15)
options("polmineR.right" = 15)
```

We also define "junk" vocabulary.

```r
junk <- c(tm::stopwords("de"), c(".", "''", ",", "``", ")", "(", "-", "!", "[", "]"), "000")
```

```r
cooccurrences("GERMAPARL", "Muslime") %>% subset(!word %in% junk)
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", "Aussiedler") %>% subset(!word %in% junk)
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", "Flüchtlinge") %>% subset(!word %in% junk)
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", "Asylbewerber") %>% subset(!word %in% junk)
```

```
## ... using polmineR.Rcpp for counting
```

```r
cooccurrences("GERMAPARL", "Asylsuchende") %>% subset(!word %in% junk)
```

```
## ... using polmineR.Rcpp for counting
```

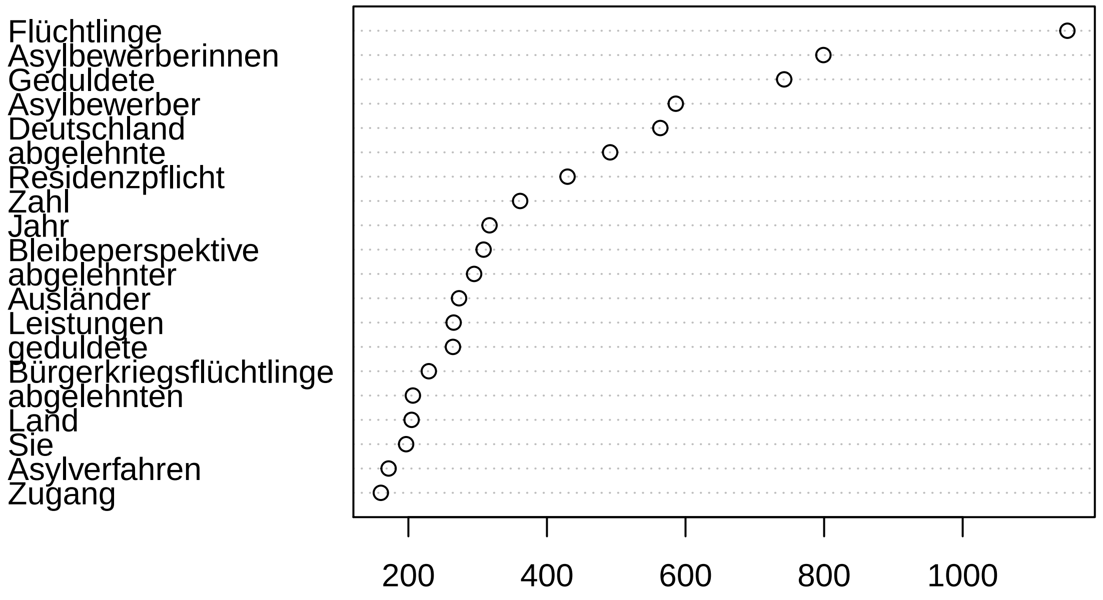
A very common visualisation is to use wordclouds.

```r
hv <- cooccurrences("GERMAPARL", "Heimatvertriebene") %>%
  subset(!word %in% junk) %>% subset(rank_ll <= 75)
pal <- brewer.pal(8,"Dark2")
wordcloud(
  words = hv[["word"]], scale = c(2.5, 0.5), freq = hv[["ll"]] / 5,
  color = pal, random.color = TRUE
  )
```

However, it is much better to use dotplots.

```
cooccurrences("GERMAPARL", "Asylbewerber") %>% subset(!word %in% junk) %>% dotplot()
```

```
## ... using polmineR.Rcpp for counting
```



Very often, we want to work with partitions. So what do we see, when we distinguish parties?

```
cdu <- partition("GERMAPARL", party = "CDU", interjection = "FALSE")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1

## ... get cpos and strucs

## ... get partition size
```

```
csu <- partition("GERMAPARL", party = "CSU", interjection = "FALSE")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1

## ... get cpos and strucs

## ... get partition size
```

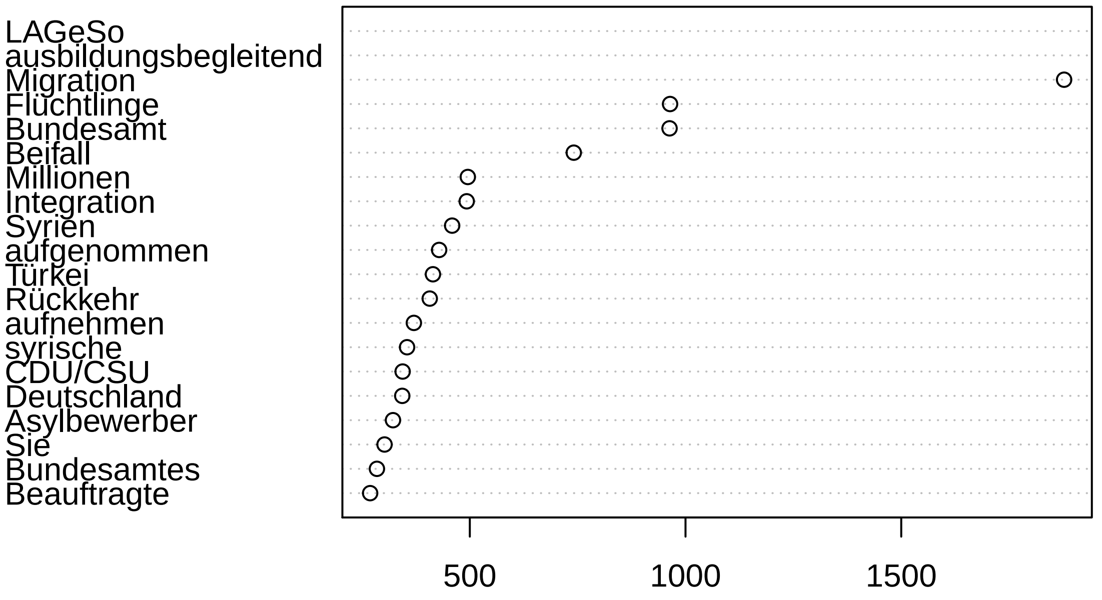
```
spd <- partition("GERMAPARL", party = "SPD", interjection = "FALSE")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1

## ... get cpos and strucs

## ... get partition size
```

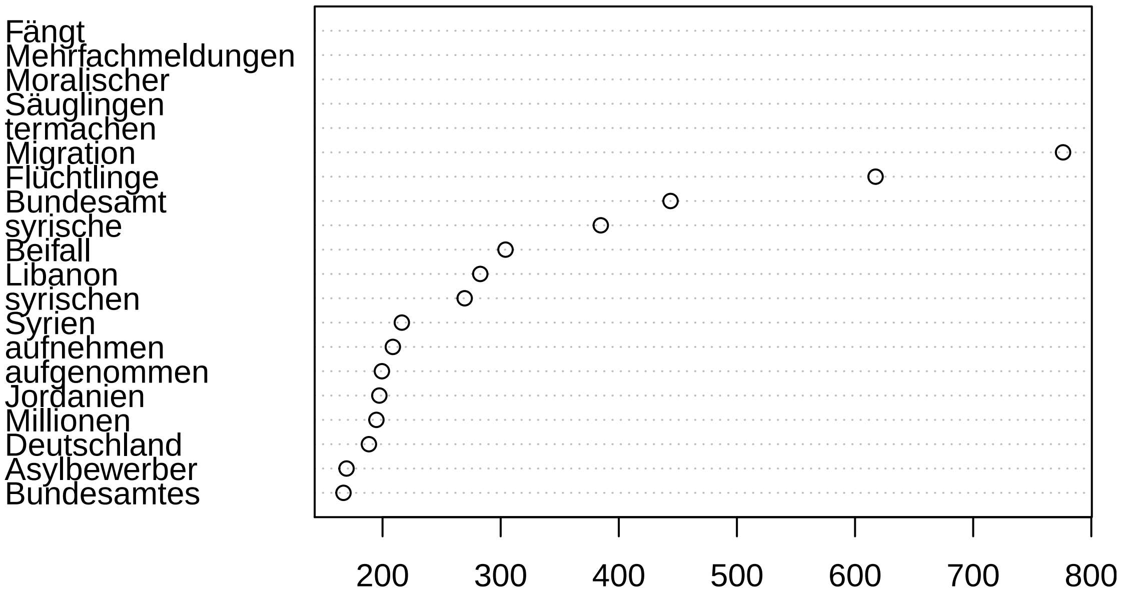
```
gruene <- partition("GERMAPARL", party = "GRUENE", interjection = "FALSE")
```

```
## ... Setting up partition

## ... type of the corpus is plpr

## ... get encoding: latin1

## ... get cpos and strucs

## ... get partition size
```

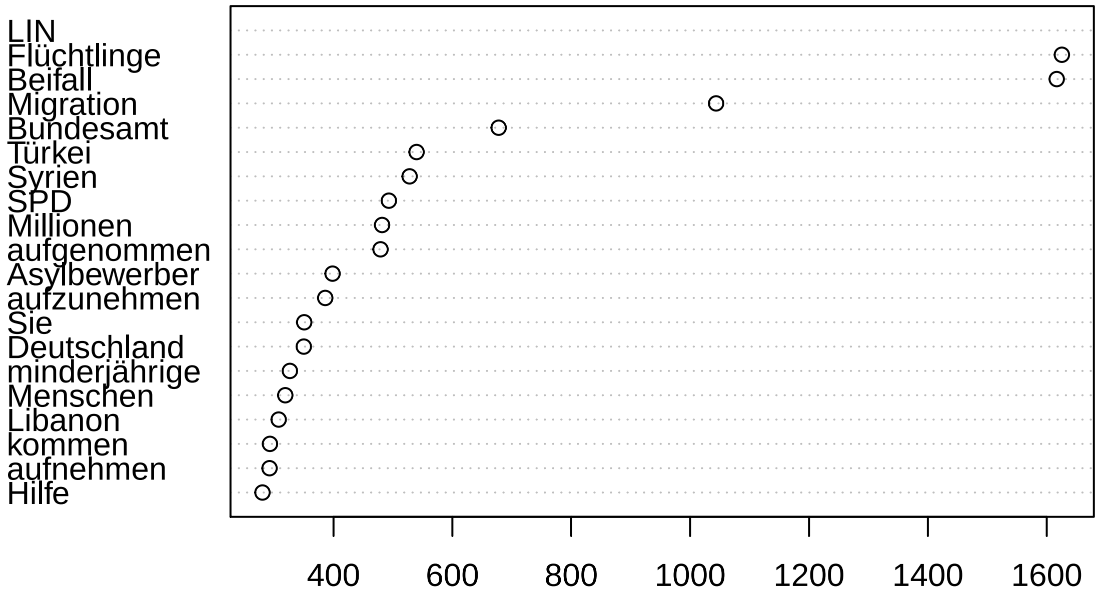
```
cdu %>% cooccurrences("Flüchtlinge") %>% subset(!word %in% junk) %>% dotplot()
```



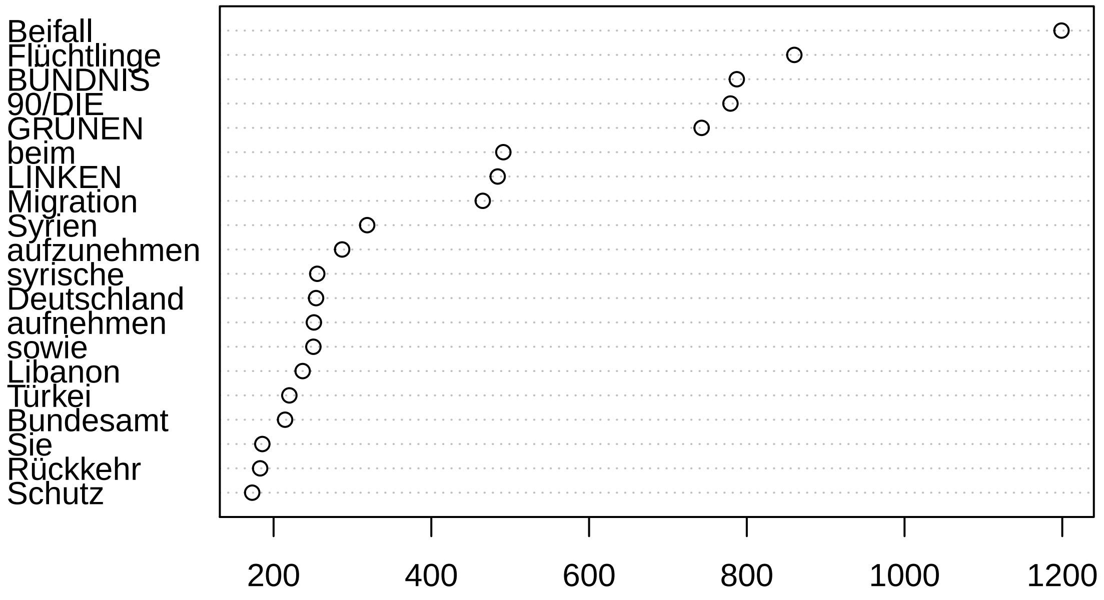
```
csu %>% cooccurrences("Flüchtlinge") %>% subset(!word %in% junk) %>% dotplot()
```



```
spd %>% cooccurrences("Flüchtlinge") %>% subset(!word %in% junk) %>% dotplot()
```



```
gruene %>% cooccurrences("Flüchtlinge") %>% subset(!word %in% junk) %>% dotplot()
```



5.1 Uses of Analyzing Cooccurrences

5.2 Statistical Measures

5.3 Validity: From Numbers to Words

5.4 Advanced Applications

5.4.1 Cooccurrence Graphs

5.4.2 Word Embeddings

Chapter 6

Keywords

6.1 Using Term Extraction to Get Keywords

6.2 Statistical Measures

6.3 Using Keywords to Generate Dictionaries

Chapter 7

Dictionaries

7.1 Available Resources

7.2 Developing Dictionaries

7.3 Scoring Texts Using Dictionaries

7.4 Use Case: Sentiment Analysis

Chapter 8

Bags of Words and Matrices

8.1 Uses and Limits of Bag of Word Approaches

8.2 Parsimony: Sparse Matrices

Chapter 9

Topicmodelling

9.1 Optimizing Topicmodels

9.2 Validating Topicmodels

9.3 From Unsupervised to Supervised Learning