Statistical Inference: a Gentle Introduction for Linguists and similar creatures (SIGIL)

With practical examples in GNU R

Designed by Marco Baroni[1] and Stefan Evert[2]

[1]Center for Mind/Brain Sciences (CIMeC)
University of Trento, Italy

[2]Corpus Linguistics Group
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

http://SIGIL.r-forge.r-project.org/

Outline

Outline

Why do we need statistics?

Why do we need statistics?

- **Significance** (control for sampling variation)
  - all linguistic data are samples (of language, speakers, ...)
  - observed effects may be coincidence of particular sample
  - ➥ **inferential statistics**

Why do we need statistics?

- **Significance** (control for sampling variation)
  - all linguistic data are samples (of language, speakers, ...)
  - observed effects may be coincidence of particular sample
  - ➥ **inferential statistics**
- Managing **large data sets**
  - statistical summaries, data analysis, visualisation
  - e.g. collocations as compact summary of word usage
  - ➥ **descriptive statistics**

Why do we need statistics?

- **Significance** (control for sampling variation)
  - all linguistic data are samples (of language, speakers, . . .)
  - observed effects may be coincidence of particular sample
  - ➥ **inferential statistics**
- Managing **large data sets**
  - statistical summaries, data analysis, visualisation
  - e.g. collocations as compact summary of word usage
  - ➥ **descriptive statistics**
- Discovering **latent** (hidden) **properties**
  - clustering, multivariate analysis, distributional semantics
  - advanced statistical modelling (e.g. mixed-effects models)
  - ➥ **exploratory data analysis**

What everyone needs to know about statistics

What everyone needs to know about statistics

- Population vs. sample

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)
- Estimation: MLE, confidence interval

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)
- Estimation: MLE, confidence interval
- Statistical modelling (regression, general linear model, ...)

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)
- Estimation: MLE, confidence interval
- Statistical modelling (regression, general linear model, ...)
- Applicability and limitations of statistical methods

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)
- Estimation: MLE, confidence interval
- Statistical modelling (regression, general linear model, . . .)
- Applicability and limitations of statistical methods

☞ Gain practical experience with statistical software

What everyone needs to know about statistics

- Population vs. sample
- Random variables & scales of measurement
- Basic principles of statistical inference
- Statistical tests: null hypothesis, p-value, assumptions
- Significance vs. effect size (relevance)
- Estimation: MLE, confidence interval
- Statistical modelling (regression, general linear model, ...)
- Applicability and limitations of statistical methods

☞ Gain practical experience with statistical software

What you can do on your own:

- Learn about specific statistical tests and procedures

R – An environment for statistical programming

- "Traditional" statistical software packages offer specialised procedures (e.g. SAS) or interactive GUI (e.g. SPSS)

R – An environment for statistical programming

- "Traditional" statistical software packages offer specialised procedures (e.g. SAS) or interactive GUI (e.g. SPSS)
- New approach: statistical programming language **S** with interactive environment (Bell Labs, since 1976)
  - *White Book* (version 3, 1992); *Green Book* (version 4, 1998)
  - commercial: S-Plus (Insightful Corporation, since 1987)

R – An environment for statistical programming

- "Traditional" statistical software packages offer specialised procedures (e.g. SAS) or interactive GUI (e.g. SPSS)
- New approach: statistical programming language **S** with interactive environment (Bell Labs, since 1976)
  - *White Book* (version 3, 1992); *Green Book* (version 4, 1998)
  - commercial: S-Plus (Insightful Corporation, since 1987)
- **R** is an open-source implementation of the S language
  - originally by Ross Ihaka and Robert Gentleman (Auckland)
  - open-source development since mid-1997

R – An environment for statistical programming

- binary packages available for Linux, Mac OS X and Windows
- 64-bit support for large data sets
- extensive documentation & tutorials
- thousands of add-on packages ready to install from CRAN

http://www.R-project.org/

Recommended cross-platform GUI:
RStudio from `http://www.rstudio.com/ide/`

More about R

- Advantages of R
  - free & open source
  - many add-on packages with state-of-the-art algorithms
  - large, enthusiastic and helpful user community
  - easy to automate and extend (every analysis is a program)
  - no point & click interface

More about R

- Advantages of R
  - free & open source
  - many add-on packages with state-of-the-art algorithms
  - large, enthusiastic and helpful user community
  - easy to automate and extend (every analysis is a program)
  - no point & click interface
- Disadvantages
  - learning curve sometimes rather steep
  - not very good at manipulating non-English text
  - no built-in data editor (spreadsheet)
  - no point & click interface

Outline

Goals of the course

- Basic principles of statistical inference
- Elementary hypothesis tests, estimators & models
- Hands-on work with R on real-life data sets
- Data manipulation and basic R programming skills
- Get to know R implementations of statistical techniques, data analysis and visualisation methods that are useful in various areas of (computational) linguistics along the way

Goals of the course

- Basic principles of statistical inference
- Elementary hypothesis tests, estimators & models
- Hands-on work with R on real-life data sets
- Data manipulation and basic R programming skills
- Get to know R implementations of statistical techniques, data analysis and visualisation methods that are useful in various areas of (computational) linguistics along the way

What this course is *not* about:

- Deeper mathematical foundations of statistics
- Specific (advanced) statistical methods
- Cookbook recipes for particular analyses with R

Course units

1. R Basics: installation, data manipulation, input/output (h)
2. Corpus frequency data & statistical inference (h)
3. Descriptive and inferential statistics for continuous data (f)
4. Co-occurrence, contingency tables and collocations (f)
   + vectorised data processing, high-quality graphs
5. Word frequency distributions with the `zipfR` package (h)
6. Regression and linear models (f)
7. Exploratory data analysis: clustering, visualisation, ML (h)
8. The non-randomness of corpus data: a GLM approach (h)
9. Inter-annotator agreement (h)

(h) = half-day session / (f) = full-day session (optimistic)

Introductions

Who are you?

Recommended textbooks: introductory level

- Baroni, Marco & Evert, Stefan (2008). Statistical methods for corpus exploitation. In A. Lüdeling & M. Kytö (eds.), *Corpus Linguistics. An International Handbook*, Mouton de Gruyter.
- Gries, Stefan Th. (2013). *Statistics for Linguistics with R: A Practical Introduction*, 2nd ed. Mouton de Gruyter. [€29]
  - German original from Vandenhoek & Ruprecht [€25]
- Johnson, Keith (2008). *Quantitative Methods in Linguistics*. Blackwell. [€38]
- Peter Dalgaard (2008). *Introductory Statistics with R*, 2nd ed. Springer. [€52]

Recommended textbooks: advanced level

- R. Harald Baayen (2008). *Analyzing Linguistic Data: A practical introduction to statistics*. CUP. [€29]
  - http://www.sfs.uni-tuebingen.de/~hbaayen/publications/
- Morris H. DeGroot and Mark J. Schervish (2002). *Probability and Statistics*, 4th ed. Pearson Education Ltd. [€74]
- John M. Chambers (2008). *Software for Data Analysis: Programming with R*. Springer. [€85]
- Christopher Butler (1985), *Statistics in Linguistics*. Blackwell.
  - out of print and available online for free download from http://www.uwe.ac.uk/hlss/llas/statistics-in-linguistics/bkindex.shtml

Course materials

- Handouts, example scripts and data sets are available on our homepage for this course:

  http://SIGIL.R-Forge.R-Project.org/

  (includes additional material, software, links, etc.)

Another interesting online course:

- Shravan Vasishth (2006–2009). *The foundations of statistics: A simulation-based approach*.
  - http://www.ling.uni-potsdam.de/~vasishth/SFLS.html

Outline

Installation guide for Linux & Mac OS X

Mac OS X

- Download binary installer from `http://www.R-project.org/`
- Start GUI application `R` (64-bit)
- Alternative: run from TextMate or various other text editors
- Shell commmand `R` available for command-line use

Linux (Ubuntu and other popular distributions)

- Install R with standard package manager (e.g. *Synaptic*)
- Add CRAN repository to obtain up-to-date version of R
  - e.g. http://cran.at.r-project.org/bin/linux/ubuntu precise/
  - pkgs: r-base-core r-base-html r-base-dev r-doc-html r-doc-pdf
- Various GUIs available, e.g. Rkward and R Commander
- Power users: Emacs + ESS or shell command `R` in terminal

Installing add-on packages

Mac OS X

- Select Packages & Data | Package Installer from GUI menu
- Click Get List, then choose packages to be installed
  - you may need to check install dependencies, too
  - installing for all users is only possible on the command line

Linux (Ubuntu and other popular distributions)

- Use standard package manager with CRAN repository
  - offers choice of "difficult" binary packages named r-cran-*
  - make sure that you install the up-to-date CRAN versions!
- Other packages need to be installed from the command line

All Unix platforms

- Install packages from within R (system-wide with `sudo R`)
  - e.g. `install.packages(c("languageR", "corpora"))`
  - select CRAN mirror from pop-up list (recommended: Austria)

Installation on Windows (XP/Vista/7)

Step 1: Download R for Windows installer from `www.R-project.org`

- CRAN | choose mirror (Austria) | R for Windows | base
- Download R ... for Windows, then run the installer
- if Windows complains, allow installer to run & make changes
- select "full installation" and keep defaults for everything else
- start R, adjust GUI preferences and save to default location (make sure to select SDI mode if you use Tinn-R GUI)

Installation on Windows (XP/Vista/7)

Step 1: Download R for Windows installer from `www.R-project.org`

- CRAN | choose mirror (Austria) | R for Windows | base
- Download R … for Windows, then run the installer
- if Windows complains, allow installer to run & make changes
- select "full installation" and keep defaults for everything else
- start R, adjust GUI preferences and save to default location (make sure to select SDI mode if you use Tinn-R GUI)

Step 2: Install some important add-on packages

- Vista/Win 7: run R as administrator to install packages for all users (right-click program icon in Start menu)
- select Packages | Install package from GUI menu
- choose mirror (Austria), then pick the package(s) to install
- check successful installation with these R commands:

```
library(corpora)
help("VSS") # should pop up Web browser with help page
data(VSS); head(VSS, 20)
```

Recommended add-on packages for this course

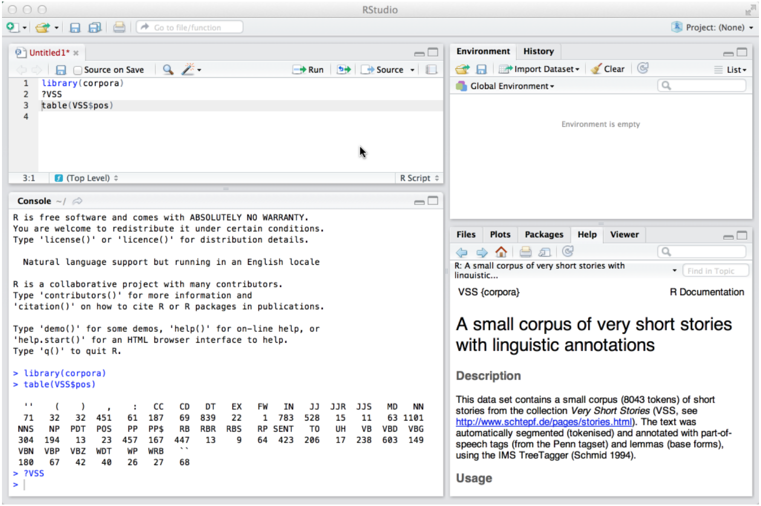
`SIGIL`	data sets and utilities for this course
`corpora`	some additional corpus-related functions
`languageR`	data sets and functions from Baayen (2008)
`exact2x2`	exact inference for 2×2 contingency tables (relevant for corpus frequency comparisons)
`zipfR`	word frequency distributions & Zipf's law
`e1071`	machine learning (SVM) and many other utilities
`MASS`	lots of statistical functions (companion package to *Modern Applied Statistics with S and S-Plus*)

Some other useful packages:

`rgl`	animated 3D graphics with OpenGL (also: `misc3d`)
`vcd`	visualisation of categorical data (contingency tables)
`plyr`, `doBy`, `reshape2`	convenience functions for data manipulation

Recommended cross-platform GUI: RStudio

http://www.rstudio.com/ide/



Outline

R as an oversized calculator

```
> 1+1
[1] 2

> a <- 2          # assignment does not print anything by default

> a * 2
[1] 4

> log(a)          # natural, i.e. base-e logarithm
[1] 0.6931472

> log(a,2)        # base-2 logarithm
[1] 1
```

Basic session management

Some of it is not necessary if you only use the GUI

```
# to start R on command line, simply type ``R''

setwd("path/to/data")   # or use GUI menus

ls()                    # probably empty for now

ls                      # notice difference with previous line

quit()                  # or use GUI menus
quit(save="yes")
quit(save="no")

# NB: at least some interfaces support history recall, TAB completion, etc.
```

Vectorial math

```
> a <- c(1,2,3)  # c (for combine) creates vectors

> a * 2      # operators are applied to each element of a vector
[1] 2 4 6


> log(a)     # also works for most standard functions
[1] 0.0000000 0.6931472 1.0986123

> sum(a)     # basic vector operations: sum, length, product, ...
[1] 6

> length(a)
[1] 3

> sum(a)/length(a)
[1] 2
```

Initializing vectors

```
> a <- 1:100                # integer sequence
> a

> a <- 10^(1:100)

> a <- seq(from=0, to=10, by=0.1) # general sequence

> a <- rnorm(100)           # 100 random numbers

> a <- runif(100, 0, 5) # what you're used to from Java etc.
```

Summary statistics

More about these summary statistics in Unit 3

```
> length(a)

> summary(a)    # statistical summary of numeric vector
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.02717 0.51770 1.05200 1.74300 2.32600 9.11100

> mean(a)

> median(a)

> sd(a)         # standard deviation is not included in summary

> quantile(a)
     0%     25%     50%     75%    100%
0.0272 0.5177 1.0518 2.3261 9.1107
```

Basic plotting

```
> a <- 2^(1:100)              # don't forget the parentheses!
> plot(a)

> x <- 1:100                  # most often: plot x against y
> y <- sqrt(x)
> plot(x, y)

> plot(x, a)
> plot(x, a, log="y")         # various logarithmic plots
> plot(x, a, log="x")
> plot(x, a, log="xy")
> plot(log(x), log(a))

> hist(rnorm(100))            # histogram and density estimation
> hist(rnorm(1000))
> plot(density(rnorm(100000)))
```

(Slightly less) basic plotting

```
> a <- rbinom(10000,100,.5)
> hist(a)

> hist(a, probability=TRUE)
> lines(density(a))

> hist(a, probability=TRUE)
> lines(density(a), col="red", lwd=3)

> hist(a, probability=TRUE,
  main="Some Distribution", xlab="value",
  ylab="probability")   # better to type command on a single line!
> lines(density(a), col="red", lwd=3)
```

Help!

```
> help("hist")     # R has excellent online documentation
> ?hist            # short, convenient form of the help command

> help.search("histogram")

> ?help.search

> help.start()     # searchable HTML documentation

# or use GUI menus to access & search documentation
```

Your first R script

- Simply type R commands into a text file & save it
- Use built-in GUI functionality or external text editor
  - Microsoft Word is *not* a text editor!
  - nor is Apple's TextEdit application . . .

- Execute R script from GUI editor or by typing

```
> source("my_script.R")  # more about files later
> source(file.choose())  # select with file dialog box
```

- Many GUI editors can execute scripts line by line
  - check your editor's documentation for keyboard shortcuts

- Just typing an expression will not automatically print the result in a script: use `print(sd(a))` instead of `sd(a)`

Outline

Input from an external file

- We like to keep our data in space- or TAB-delimited text files with a first row ("header") labeling the fields:

```
word	frequency	cat
dog	15	noun
bark	10	verb
```

- This is an easy format to import into R, and it is easy to convert to/from other tabular formats using standard tools
- We assume that external input is always in this format (or can easily be converted to it)
  - spreadsheet applications prefer CSV (comma-separated values), which R also reads and writes quite well
  - Microsoft Excel is a nice table editor, but beware of localised number formats

Reading a TAB-delimited file with header

```
> brown <- read.table("brown.stats.txt",
  header=TRUE)
# if file is not in working directory, you must specify the full path
# (or use setwd() function we introduced before)

# exact behaviour of file.choose() depends on operating system
> brown <- read.table(file.choose(), header=TRUE)

# more robust if you are sure file is in tab-delimited format
> brown <- read.delim("brown.stats.txt")

# this data set is also included in the SIGIL package
> library(SIGIL)
> brown <- BrownStats
```

Reading and writing CSV files

```
# R can also read and write files in CSV format
> write.csv(brown, "brown.stats.csv",
  row.names=FALSE)
# this is convenient for exchanging data with database and
# spreadsheet software (or using Excel as a data editor)

# NB: comma-separated values are not always separated by commas
# (e.g. in German; use write.csv2 if Excel doesn't recognise columns)
> write.csv2(brown, "brown.stats.csv",
  row.names=FALSE)

# TASK: load brown.stats.csv into Excel or OpenOffice.org

# check generated CSV file (use read.csv2 with write.csv2 above)
> brown.csv <- read.csv("brown.stats.csv")
> all.equal(brown.csv, brown)
```

Data frames

- The commands above create a **data frame**
- This is the basic data structure (object) used to represent statistical tables in R
  - rows = objects or "observations"
  - columns = variables, i.e. measured quantities
- Different types of variables
  - numerical variables (what we've used so far)
  - Boolean variables
  - factor variables (nominal or ordinal classification)
  - string variables
- Technically, data frames are collections of column vectors (of the same length), and we will think of them as such

Data frames

```
> summary(brown)

> colnames(brown)

> dim(brown)            # number of rows and columns

> head(brown)

> plot(brown)
```

Type/token counts and word lengths for Brown & LOB texts

Data files in TAB-delimited format:

- `brown.stats.txt`: information for Brown corpus (AmE)
- `lob.stats.txt`: information for LOB corpus (BrE)

Variables:

to	Token count
ty	Type count (*distinct* words)
se	Sentence count
towl	Average word length (averaged across tokens in document)
tywl	Average word length (averaged across distinct types in document)

Access vectors inside a data frame

```
> brown$to
> head(brown$to)

# TASK: compute summary statistics (length, mean, max, etc.)
# for vectors in the Brown data frame

# what does the following do?
> summary(brown$ty / brown$to)

> attach(brown)    # attach data frame for convenient access
> summary(ty/to)
> detach()         # detach from search path

> with(brown, summary(ty/to)) # a better approach
```

More data access

```
> brown$ty[1]      # vector indexing starts with 1
> brown[1,2]       # row, column

> brown$ty[1:10]  # use arbitrary vectors as indices
> brown[1:10,2]

> brown[1,]
> brown[,2]
```

Conditional selection

```
> brown[brown$to < 2200, ]   # index with Boolean vector
> brown$ty[brown$to >= 2200]
> sum(brown$to >= 2200)      # standard way to count matches

> subset(brown, to < 2200)   # syntactic sugar (similar to with)
> lessdata <- subset(brown, to < 2200)

> a <- brown$ty[brown$to >= 2200]

# equality: == (also works for strings)
# inequality: !=
# complex constraints: and &, or |, not !
# NB: always use single characters, not && or ||
```

Outline

Procedure

The methods used here will be explained in Units 3 and 6

- Collect basic summary statistics for the two corpora
- Check if there is a significant difference in the token counts (since document length was controlled by corpus builders)
- If difference is significant (we will see that it is), then type counts are not directly comparable, and sentence counts should be normalized (divide by token count)
- Is word length correlated to document length? (corpus comparison would also not be appropriate in this case)

Procedure

The methods used here will be explained in Units 3 and 6

- Collect basic summary statistics for the two corpora
- Check if there is a significant difference in the token counts (since document length was controlled by corpus builders)
- If difference is significant (we will see that it is), then type counts are not directly comparable, and sentence counts should be normalized (divide by token count)
- Is word length correlated to document length? (corpus comparison would also not be appropriate in this case)
- Please read the LOB data set into a data frame named `lob` now, and take a look at its basic statistics
  - file `lob.stats.txt`, or `LOBStats` in `SIGIL` package
- Also, plot the data frame for a first impression of correlations between the variables

Comparing token counts

```
> boxplot(brown$to,lob$to)
> boxplot(brown$to,lob$to,names=c("brown","lob"))
> boxplot(brown$to,lob$to,names=c("brown","lob"),
  ylim=c(1500,3000))
> ?boxplot

> t.test(brown$to, lob$to)
> wilcox.test(brown$to, lob$to)

> brown.to.center <-
  with(brown, to[to > 2200 & to < 2400])
> lob.to.center <-
  with(lob, to[to > 2200 & to < 2400])

> t.test(brown.to.center, lob.to.center)
```

Is word length correlated with token count?

average word length by tokens and types is almost the same:

```
> plot(brown$towl, brown$tywl)
> cor.test(brown$towl, brown$tywl)
> cor.test(brown$towl, brown$tywl, method="spearman")
```

correlation with token count

```
> plot(brown$to, brown$towl)
> cor.test(brown$to, brown$towl)
```