# R TUTORIAL - VTT (Lecture 1)

Oguzhan (Ouz) Gencoglu

VTT Technical Research Center of Finland

27 April 2016

# Outline

# Outline

# What is R? Why/Why not R? Practical Issues.

## What exactly is this R thing?

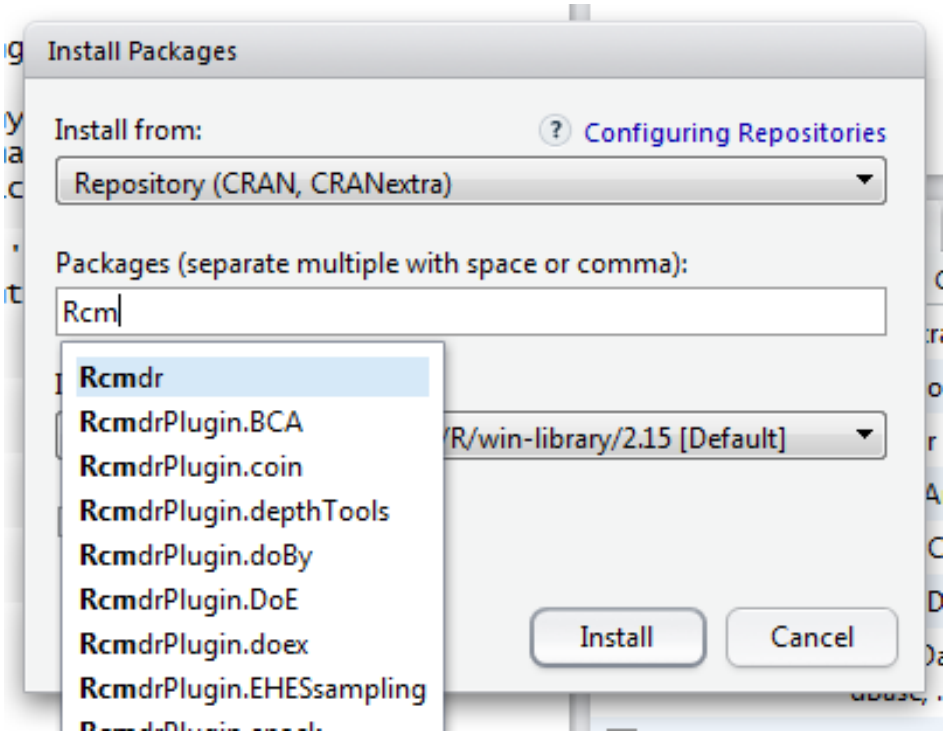
R is a language and environment for statistical computing and graphics. It provides a wide variety of statistical (linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible.

# Outline

# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

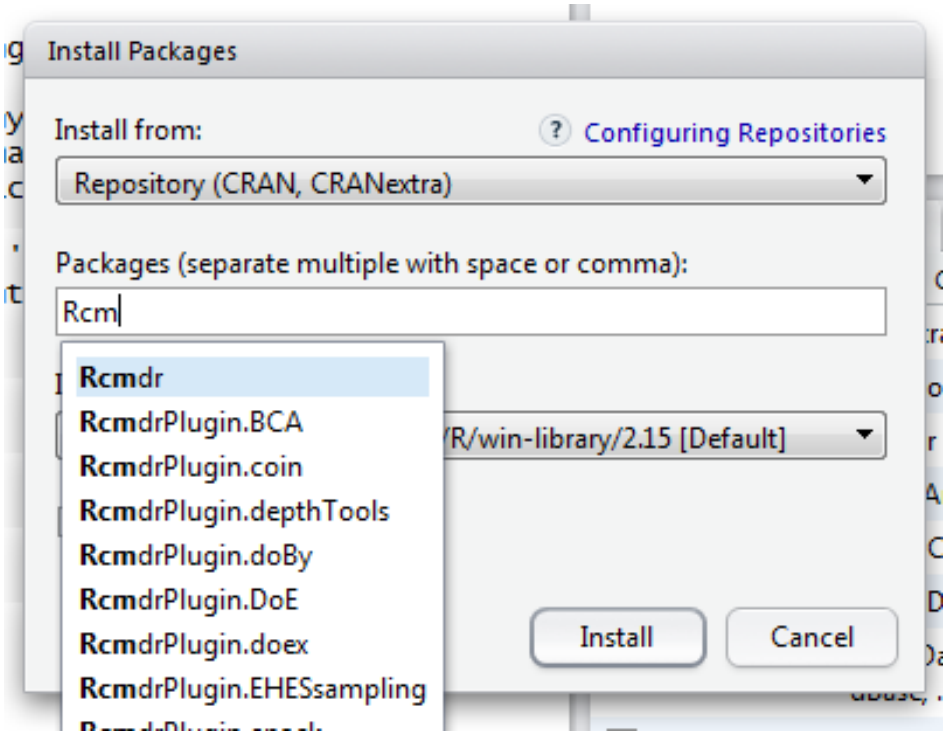
- Developed by statisticians & scientists



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

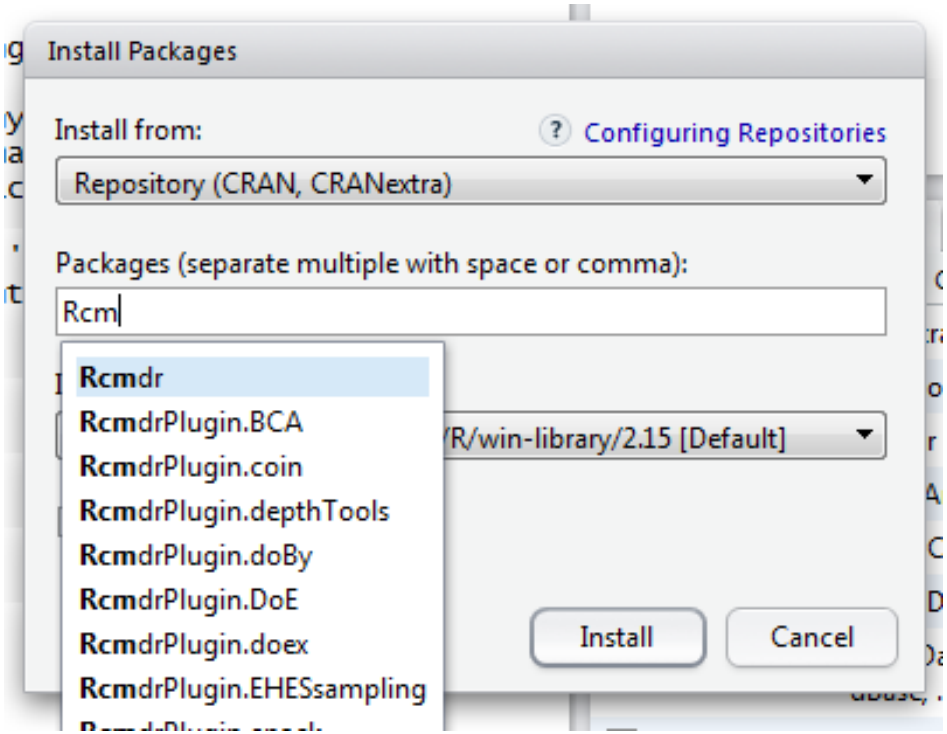
- Developed by statisticians & scientists
- Free (Speech & Sandwich)



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

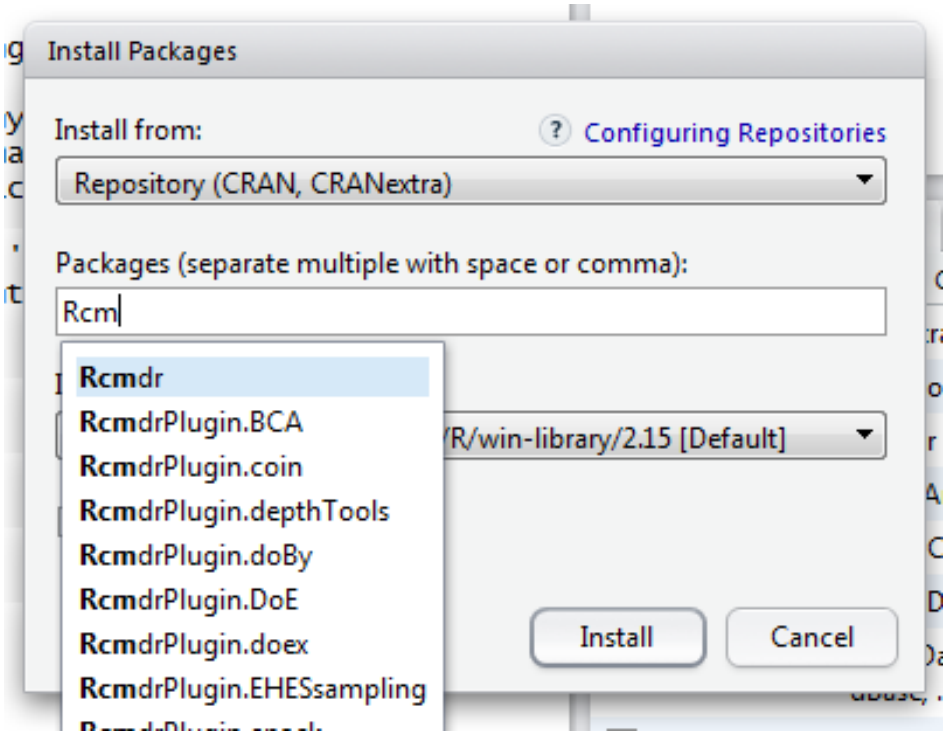
- Developed by statisticians & scientists
- Free (Speech & Sandwich)
- Most comprehensive statistical analysis environment



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

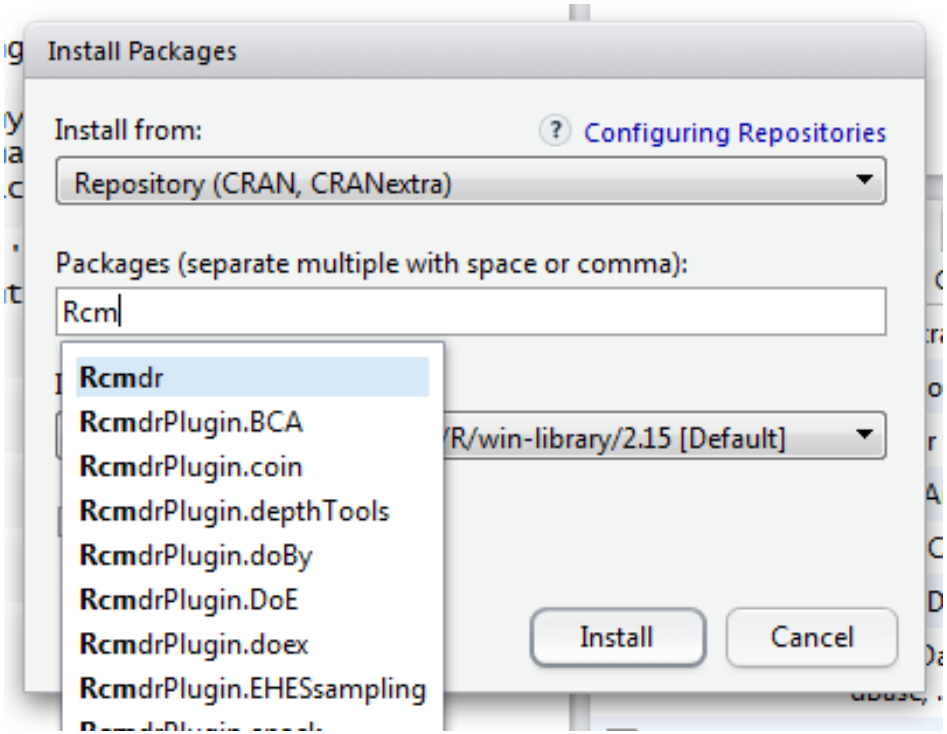
- Developed by statisticians & scientists
- Free (Speech & Sandwich)
- Most comprehensive statistical analysis environment
- Outstanding graphical capabilities



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

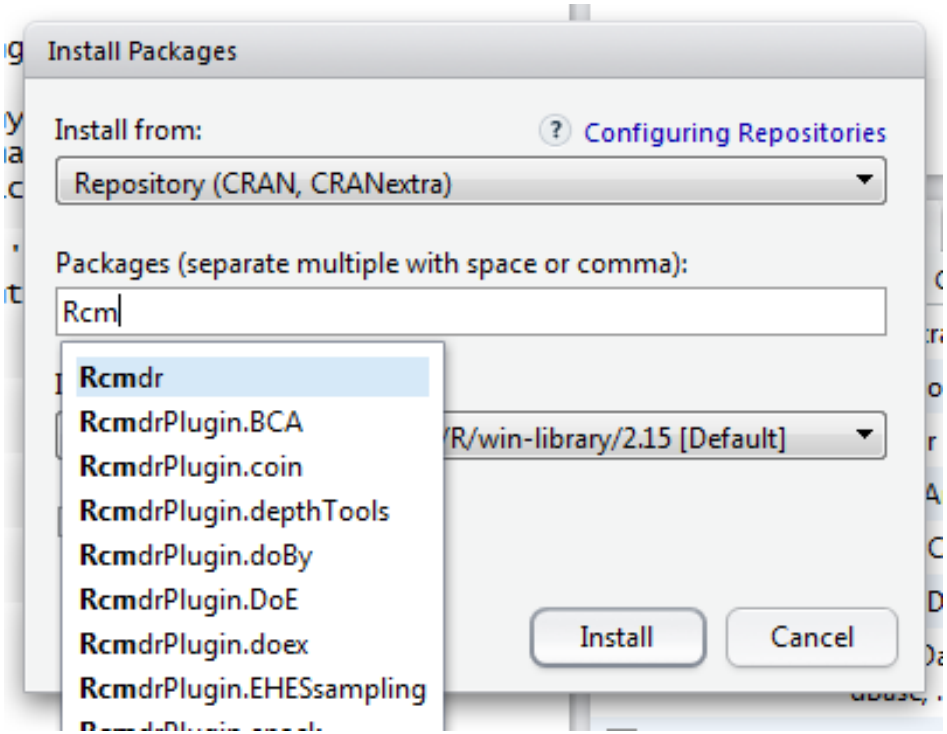
- Developed by statisticians & scientists
- Free (Speech & Sandwich)
- Most comprehensive statistical analysis environment
- Outstanding graphical capabilities
- Almost 5000 packages



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

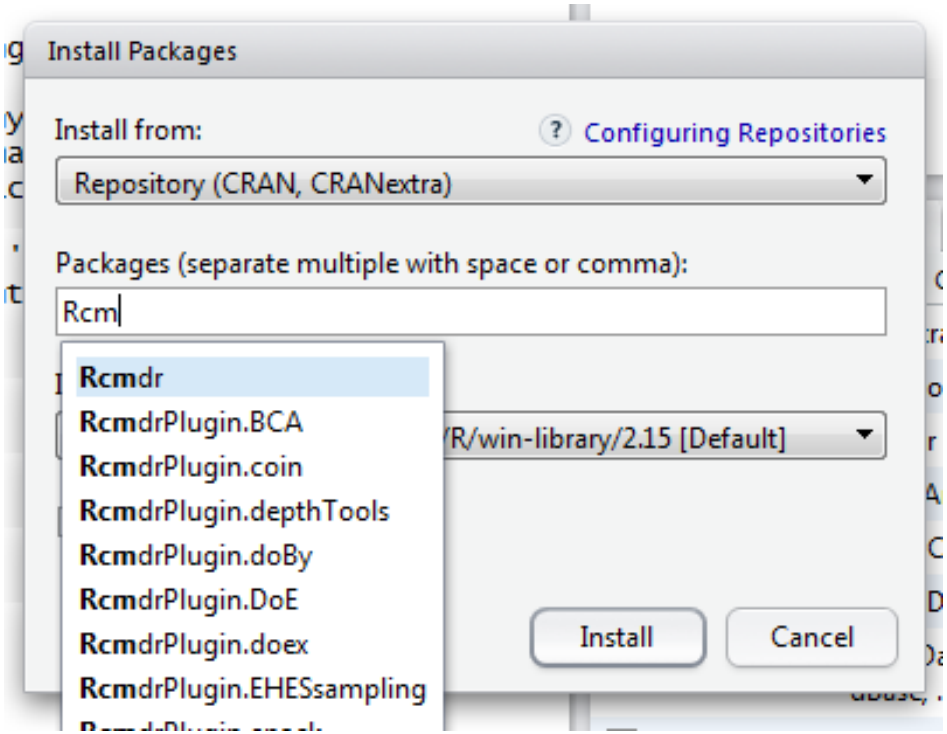
- Developed by statisticians & scientists
- Free (Speech & Sandwich)
- Most comprehensive statistical analysis environment
- Outstanding graphical capabilities
- Almost 5000 packages
- Cross-platform



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- Developed by statisticians & scientists
- Free (Speech & Sandwich)
- Most comprehensive statistical analysis environment
- Outstanding graphical capabilities
- Almost 5000 packages
- Cross-platform
- Active user group



# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- ☹ Developed by statisticians & scientists

```
<?xml version="1.0" encoding="UTF-8"?>
```

although it does not require it.

The XML package provides general facilities for reading and writing XML documents within R. A description of the facilities c details and examples. Package StatDataML on CRAN is one example building on XML.

NB: XML is available as a binary package for Windows, normally from the 'CRAN extras' repository (which is selected by def

Next: Importing from other statistical systems, Previous: Introduction, Up: Top [Contents][Index]

2 Spreadshe

- Variations on read.table:
- Fixed-width-format files:
- Data Interchange Format (DIF):
- Using scan directly:
- Re-shaping data:
- Flat contingency tables:

In Export to text files we saw a number of variations on the format of a spreadsheet-like text file, in which the data are presentec

Next: Fixed-width-format files, Previous: Spreadsheet-like data, Up: Spreadsheet-like data [Contents][Index]

### 2.1 Variations on read.table

The function `read.table` is the most convenient way to read in a rectangular grid of data. Because of the many possibilities, ther

Beware that `read.table` is an inefficient way to read in very large numerical matrices: see `scan` below.

Some of the issues to consider are:

1. **Encoding**
   If the file contains non-ASCII character fields, ensure that it is read in the correct encoding. This is mainly an issue for rea

   ```
   read.table("file.dat", fileEncoding="latin1")
   ```

   Note that this will work in any locale which can represent Latin-1 strings, but not many Greek/Russian/Chinese/Japanese
2. **Header line**
   We recommend that you specify the `header` argument explicitly, Conventionally the header line has entries only for the co presented with a file that has a (possibly empty) header field for the row labels, read it in by something like

   ```
   read.table("file.dat", header = TRUE, row.names = 1)
   ```

   Column names can be given explicitly via the `col.names`; explicit names override the header line (if present).
3. **Separator**
   Normally looking at the file will determine the field separator to be used, but with white-space separated files there may b `"\t"`. Note that the choice of separator affects the input of quoted strings.

   If you have a tab-delimited file containing empty fields be sure to use `sep = "\t"`.
4. **Quoting**
   By default character strings can be quoted by either `"` or `'`, and in each case all the characters up to a matching quote ar argument. For `sep = "\n"` the default is changed to `quote = ""`.

   If no separator character is specified, quotes can be escaped within quoted strings by immediately preceding them by '\',

   If a separator character is specified, quotes can be escaped within quoted strings by doubling them as is conventional in sp

# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- ☹ Developed by statisticians & scientists
- ☹ Learning Curve

```
<?xml version="1.0" encoding="UTF-8"?>
```

although it does not require it.

The XML package provides general facilities for reading and writing XML documents within R. A description of the facilities c details and examples. Package StatDataML on CRAN is one example building on XML.

NB: XML is available as a binary package for Windows, normally from the 'CRAN extras' repository (which is selected by def

Next: Importing from other statistical systems. Previous: Introduction. Up: Top [Contents][Index]

2 Spreadshe

- Variations on read.table:
- Fixed-width-format files:
- Data Interchange Format (DIF):
- Using scan directly:
- Re-shaping data:
- Flat contingency tables:

In Export to text files we saw a number of variations on the format of a spreadsheet-like text file, in which the data are presentec

Next: Fixed-width-format files. Previous: Spreadsheet-like data. Up: Spreadsheet-like data [Contents][Index]

2.1 Variations on read.table

The function `read.table` is the most convenient way to read in a rectangular grid of data. Because of the many possibilities, ther

Beware that `read.table` is an inefficient way to read in very large numerical matrices: see `scan` below.

Some of the issues to consider are:

1. **Encoding**
   If the file contains non-ASCII character fields, ensure that it is read in the correct encoding. This is mainly an issue for rea

   ```
   read.table("file.dat", fileEncoding="latin1")
   ```

   Note that this will work in any locale which can represent Latin-1 strings, but not many Greek/Russian/Chinese/Japanese
2. **Header line**
   We recommend that you specify the `header` argument explicitly, Conventionally the header line has entries only for the co presented with a file that has a (possibly empty) header field for the row labels, read it in by something like

   ```
   read.table("file.dat", header = TRUE, row.names = 1)
   ```

   Column names can be given explicitly via the `col.names`; explicit names override the header line (if present).
3. **Separator**
   Normally looking at the file will determine the field separator to be used, but with white-space separated files there may b "\t". Note that the choice of separator affects the input of quoted strings.

   If you have a tab-delimited file containing empty fields be sure to use `sep = "\t"`.
4. **Quoting**
   By default character strings can be quoted by either '"' or "'", and in each case all the characters up to a matching quote ar argument. For `sep = "\n"` the default is changed to `quote = ""`.

   If no separator character is specified, quotes can be escaped within quoted strings by immediately preceding them by '\',

   If a separator character is specified, quotes can be escaped within quoted strings by doubling them as is conventional in sp

# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- ☹ Developed by statisticians & scientists
- ☹ Learning Curve
- ☹ Memory and Speed Issues for certain tasks

```
<?xml version="1.0" encoding="UTF-8"?>
```

although it does not require it.

The XML package provides general facilities for reading and writing XML documents within R. A description of the facilities c details and examples. Package StatDataML on CRAN is one example building on XML.

NB: XML is available as a binary package for Windows, normally from the 'CRAN extras' repository (which is selected by def

Next: Importing from other statistical systems. Previous: Introduction. Up: Top [Contents][Index]

2 Spreadshe

- Variations on read.table:
- Fixed-width-format files:
- Data Interchange Format (DIF):
- Using scan directly:
- Re-shaping data:
- Flat contingency tables:

In Export to text files we saw a number of variations on the format of a spreadsheet-like text file, in which the data are presentec

Next: Fixed-width-format files. Previous: Spreadsheet-like data. Up: Spreadsheet-like data [Contents][Index]

### 2.1 Variations on read.table

The function `read.table` is the most convenient way to read in a rectangular grid of data. Because of the many possibilities, ther

Beware that `read.table` is an inefficient way to read in very large numerical matrices: see `scan` below.

Some of the issues to consider are:

1. **Encoding**
   If the file contains non-ASCII character fields, ensure that it is read in the correct encoding. This is mainly an issue for rea

   ```
   read.table("file.dat", fileEncoding="latin1")
   ```

   Note that this will work in any locale which can represent Latin-1 strings, but not many Greek/Russian/Chinese/Japanese
2. **Header line**
   We recommend that you specify the `header` argument explicitly, Conventionally the header line has entries only for the co presented with a file that has a (possibly empty) header field for the row labels, read it in by something like

   ```
   read.table("file.dat", header = TRUE, row.names = 1)
   ```

   Column names can be given explicitly via the `col.names`; explicit names override the header line (if present).
3. **Separator**
   Normally looking at the file will determine the field separator to be used, but with white-space separated files there may b `"\t"`. Note that the choice of separator affects the input of quoted strings.

   If you have a tab-delimited file containing empty fields be sure to use `sep = "\t"`.
4. **Quoting**
   By default character strings can be quoted by either `"` or `'`, and in each case all the characters up to a matching quote ar argument. For `sep = "\n"` the default is changed to `quote = ""`.

   If no separator character is specified, quotes can be escaped within quoted strings by immediately preceding them by '\',

   If a separator character is specified, quotes can be escaped within quoted strings by doubling them as is conventional in sp

# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- ☹ Developed by statisticians & scientists
- ☹ Learning Curve
- ☹ Memory and Speed Issues for certain tasks
- ☹ Ugly syntax (my opinion) - compare Python

```
<?xml version="1.0" encoding="UTF-8"?>
```

although it does not require it.

The XML package provides general facilities for reading and writing XML documents within R. A description of the facilities c details and examples. Package StatDataML on CRAN is one example building on XML.

NB: XML is available as a binary package for Windows, normally from the 'CRAN extras' repository (which is selected by def

Next: Importing from other statistical systems. Previous: Introduction. Up: Top [Contents][Index]

2 Spreadshe

- Variations on read.table:
- Fixed-width-format files:
- Data Interchange Format (DIF):
- Using scan directly:
- Re-shaping data:
- Flat contingency tables:

In Export to text files we saw a number of variations on the format of a spreadsheet-like text file, in which the data are presentec

Next: Fixed-width-format files. Previous: Spreadsheet-like data. Up: Spreadsheet-like data [Contents][Index]

2.1 Variations on read.table

The function `read.table` is the most convenient way to read in a rectangular grid of data. Because of the many possibilities, ther

Beware that `read.table` is an inefficient way to read in very large numerical matrices: see `scan` below.

Some of the issues to consider are:

1. **Encoding**
   If the file contains non-ASCII character fields, ensure that it is read in the correct encoding. This is mainly an issue for rea

   ```
   read.table("file.dat", fileEncoding="latin1")
   ```

   Note that this will work in any locale which can represent Latin-1 strings, but not many Greek/Russian/Chinese/Japanese
2. **Header line**
   We recommend that you specify the `header` argument explicitly, Conventionally the header line has entries only for the co presented with a file that has a (possibly empty) header field for the row labels, read it in by something like

   ```
   read.table("file.dat", header = TRUE, row.names = 1)
   ```

   Column names can be given explicitly via the `col.names`; explicit names override the header line (if present).
3. **Separator**
   Normally looking at the file will determine the field separator to be used, but with white-space separated files there may b "\t". Note that the choice of separator affects the input of quoted strings.

   If you have a tab-delimited file containing empty fields be sure to use `sep = "\t"`.
4. **Quoting**
   By default character strings can be quoted by either '"' or "'", and in each case all the characters up to a matching quote ar argument. For `sep = "\n"` the default is changed to `quote = ""`.

   If no separator character is specified, quotes can be escaped within quoted strings by immediately preceding them by '\',

   If a separator character is specified, quotes can be escaped within quoted strings by doubling them as is conventional in sp

# What is R? Why/Why not R? Practical Issues.

## The Good, The Bad & the Ugly (Sides of R)

- ☹ Developed by statisticians & scientists
- ☹ Learning Curve
- ☹ Memory and Speed Issues for certain tasks
- ☹ Ugly syntax (my opinion) - compare Python
- ☹ Documentation style

```
<?xml version="1.0" encoding="UTF-8"?>
```

although it does not require it.

The XML package provides general facilities for reading and writing XML documents within R. A description of the facilities c details and examples. Package StatDataML on CRAN is one example building on XML.

NB: XML is available as a binary package for Windows, normally from the 'CRAN extras' repository (which is selected by def

Next: Importing from other statistical systems. Previous: Introduction. Up: Top [Contents][Index]

2 Spreadshe

- Variations on read.table:
- Fixed-width-format files:
- Data Interchange Format (DIF):
- Using scan directly:
- Re-shaping data:
- Flat contingency tables:

In Export to text files we saw a number of variations on the format of a spreadsheet-like text file, in which the data are presentec

Next: Fixed-width-format files. Previous: Spreadsheet-like data. Up: Spreadsheet-like data [Contents][Index]

2.1 Variations on read.table

The function `read.table` is the most convenient way to read in a rectangular grid of data. Because of the many possibilities, ther

Beware that `read.table` is an inefficient way to read in very large numerical matrices: see `scan` below.

Some of the issues to consider are:

1. **Encoding**
   If the file contains non-ASCII character fields, ensure that it is read in the correct encoding. This is mainly an issue for rea

   ```
   read.table("file.dat", fileEncoding="latin1")
   ```

   Note that this will work in any locale which can represent Latin-1 strings, but not many Greek/Russian/Chinese/Japanese
2. **Header line**
   We recommend that you specify the `header` argument explicitly, Conventionally the header line has entries only for the co presented with a file that has a (possibly empty) header field for the row labels, read it in by something like

   ```
   read.table("file.dat", header = TRUE, row.names = 1)
   ```

   Column names can be given explicitly via the `col.names`; explicit names override the header line (if present).
3. **Separator**
   Normally looking at the file will determine the field separator to be used, but with white-space separated files there may b "\t". Note that the choice of separator affects the input of quoted strings.

   If you have a tab-delimited file containing empty fields be sure to use `sep = "\t"`.
4. **Quoting**
   By default character strings can be quoted by either '"' or "'", and in each case all the characters up to a matching quote ar argument. For `sep = "\n"` the default is changed to `quote = ""`.

   If no separator character is specified, quotes can be escaped within quoted strings by immediately preceding them by '\',

   If a separator character is specified, quotes can be escaped within quoted strings by doubling them as is conventional in sp

# Outline

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive
- Expressions are printed, assignments not

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive
- Expressions are printed, assignments not
- Assigning with <- (or ->)

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive
- Expressions are printed, assignments not
- Assigning with <- (or ->)
- Commenting with #

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive
- Expressions are printed, assignments not
- Assigning with <- (or ->)
- Commenting with #
- Indexing starts from 1

# What is R? Why/Why not R? Practical Issues.

## Starting the Journey

- R - `http://ftp.sunet.se/pub/lang/CRAN/`
- RStudio IDE - `http://www.rstudio.com/`
- Case sensitive
- Expressions are printed, assignments not
- Assigning with <- (or ->)
- Commenting with #
- Indexing starts from 1
- NaN (not a number), NA (not available)

# Outline

# Assignment, Lists, Arrays etc.

**introduction.R**
**control_structures.R**

- Assignment

# Assignment, Lists, Arrays etc.

**introduction.R**

**control_structures.R**

- Assignment
- Indexing

# Assignment, Lists, Arrays etc.

**introduction.R**

**control_structures.R**

- Assignment
- Indexing
- Arrays

# Assignment, Lists, Arrays etc.

**introduction.R**

**control_structures.R**

- Assignment
- Indexing
- Arrays
- List

# Assignment, Lists, Arrays etc.

**introduction.R**

**control_structures.R**

- Assignment
- Indexing
- Arrays
- List
- Control structures

# Outline

# Data Frames & Data Wrangling

**data_frames.R data_wrangling.R**

- Creating DFs

# Data Frames & Data Wrangling

**data_frames.R data_wrangling.R**

- Creating DFs
- Using DFs

# Data Frames & Data Wrangling

**data_frames.R data_wrangling.R**

- Creating DFs
- Using DFs
- Aggregation

# Data Frames & Data Wrangling

**data_frames.R data_wrangling.R**

- Creating DFs
- Using DFs
- Aggregation
- Reshaping

# Outline

# 'apply' functions

**apply_functions.R**

- lapply

# 'apply' functions

**apply_functions.R**

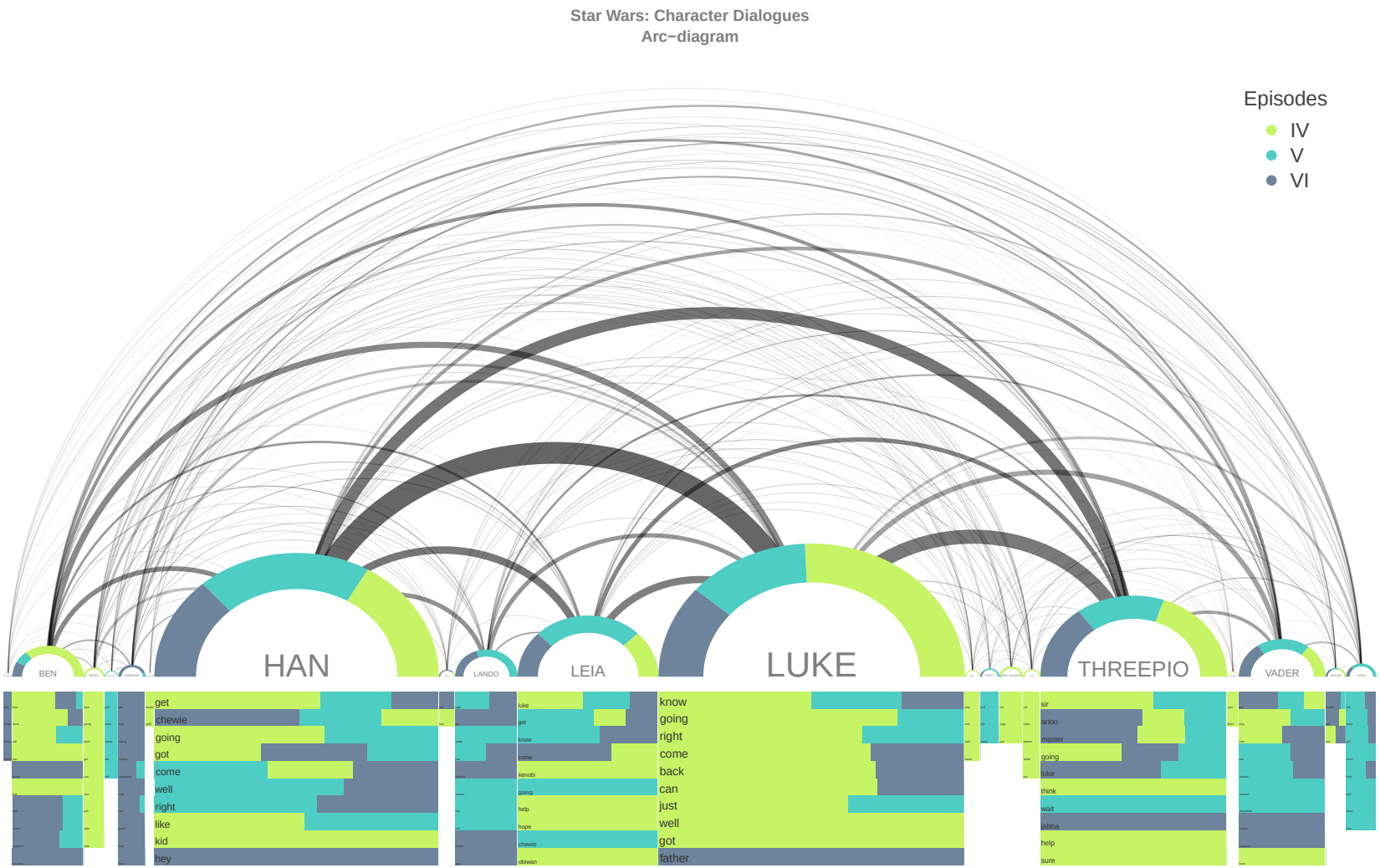
- lapply
- sapply

# 'apply' functions

**apply_functions.R**

- lapply
- sapply
- tapply

# Coming Up


Star Wars: Character Dialogues
Arc−diagram
Episodes
IV
V
VI
BEN
HAN
LANDO
LEIA
LUKE
THREEPIO
VADER
get
chewie
going
got
come
well
right
like
kid
hey
know
going
right
come
back
can
just
well
got
father