R Environment

Why R?

- It's free!
- It runs on a variety of platforms including Windows, Unix and MacOS.
- It provides an unparalleled platform for programming new statistical methods in an easy and straightforward manner.
- It contains advanced statistical routines not yet available in other packages.
- It has state-of-the-art graphics capabilities.

R paradigm is different

Supports interactive analysis rather than setting up a complete analysis at once

Installing R & R Studio

Getting Session Info

To get a description of the version of R and its attached packages used in the current session, we can use the sessionInfo function:

```
sessionInfo()

## R version 3.1.1 (2014-07-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## locale:
## [1] LC_COLLATE=English_United  States.1252
## [2] LC_CTYPE=English_United  States.1252
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] xlsx_0.5.7       xlsxjars_0.6.0  rJava_0.9-6     dplyr_0.3.0.2
##  [5] lattice_0.20-29 vcd_1.3-1       GGally_0.4.7    ggplot2_1.0.0
##
## loaded via a namespace (and not attached):
##  [1] assertthat_0.1   colorspace_1.2-4 DBI_0.3.1        digest_0.6.4
##  [5] evaluate_0.5.5   formatR_0.10     gtable_0.1.2     highr_0.3
```

R Shell

- Results of calculations can be stored in objects using the assignment operators:
  - An arrow (<-) formed by a smaller than character and a hyphen without a space!
  - The equal character (=).

Working Directory

```
getwd() # print the current working directory

setwd(mydirectory)   # change to mydirectory

#view and set options for the session
    help(options) # learn about available options
    options() # view current option settings
    options(digits=3) # number of digits to print on output
```

R Object Naming

- Object names cannot contain `strange' symbols like !, +, -, #.
- A dot (.) and an underscore ( ) are allowed, also a name starting with a dot.
- Object names can contain a number but cannot start with a number.
- R is case sensitive, X and x are two different objects, as well as temp and temP.

Packages

R Packages

- Package is a group of related functions and objects.
- R allows us to write new functions and group those functions in a so called `R package' (or `R library'). The R package may also contain other R objects, for example data sets.
- There is a lively R user community and many R packages have been written and made available on CRAN(Comprehensive R Archive Network) for other users.

Installed Packages

- The function library can also be used to list all the available libraries on your system with a short description. Run the function without any arguments

```
> library()
Packages in library 'C:/PROGRA~1/R/R-25~1.0/library':
base                    The R Base Package
Boot            Bootstrap R (S-Plus) Functions  (Canty)
class                   Functions for Classification
cluster                 Cluster Analysis Extended Rousseeuw  et al.
codetools               Code Analysis Tools for R
datasets                The R Datasets Package
DBI                     R Database Interface
foreign                 Read Data Stored by Minitab, S, SAS,
      SPSS, Stata, Systat, dBase, ...
graphics                The R Graphics Package
```

Attached Packages

- When you download R, already a number (around 30) of packages are downloaded as well.

- To use a function in an R package, that package has to be attached to the system. When you start R, not all of the downloaded packages are attached, only seven packages are attached to the system by default.

- You can use the function search to see a list of packages that are currently attached to the system, this list is also called the search path.

```
> search()
[1] ".GlobalEnv" "package:stats" "package:graphics"
[4] "package:grDevices" "package:datasets" "package:utils"
[7] "package:methods" "Autoloads" "package:base"
```

Installing Packages from CRAN

Commandline: Use install.packages function, which typically downloads the package from CRAN and installs it for use.

```
install.packages("dplyr")
install.packages("reshape2")
install.packages("ggplot2")
```

GUI option: Tools -> Install Packages

Installing Packages from any Repository

install.packages(“dplyr”, repos="http://lib.stat.cmu.edu/R/CRAN")

install.packages(“dplyr.zip”, repos=NULL)

Loading Packages

CommandLine: If we know we will need a particular package for our current R session, we must load it into the R environment using the library or require functions

library(dplyr) or require(dplyr)
library(ggplot2) or require(ggplot2)

GUI option: Select the `Packages' menu and select `Load package...'

Exploring a Package

```
ls(pos="package:stats")
```

Help System

Help System

Once **R** is installed, there is a comprehensive built-in help system. At the program's command prompt you can use any of the following:

help.start() # general help
help(*foo*) # help about function *foo*
?*foo* # same thing
apropos("*foo*") # list all function containing string foo
example(*foo*) # show an example of function *foo*

Workspace

R Workspace

- ✓Objects that you create during an R session are hold in memory, the collection of objects that you currently have is called the workspace.

- ✓The workspace is not saved on disk unless you do explicitly i.e., your objects are lost when you close R without explicit save.

- ✓Saved workspace will be available in .Rdata file. It is a binary file located in the working directory of R.

Saving Workspace

- ✓ When you close the RGui or the R console window, the system will ask if you want to save the workspace image.

- ✓ During your R session you can also explicitly save the workspace image.
  - **GUI option**: Go to the `File‘ menu and then select `Save Workspace...‘
  - **CommandLine:** Use the save.image function.

```
## save to the current working directory
save.image()
## just checking what the current working directory is
getwd()
## save to a specific file and location
save.image("C:\\Program Files\\R\\R-2.5.0\\bin\\.RData")
```

Loading Workspace

- If you have saved a workspace image and you start R the next time, it will restore the workspace. So all your previously saved objects are available again.

- You can also explicitly load a saved workspace i.e.,, that could be the workspace image of someone else. Go the `File' menu and select `Load workspace...'.

Listing workspace objects

- To list the objects that you have in your current R session use the function ls or the function objects.
  - ls()

- To list all objects starting with the letter x:

  > x2 = 9

  > y2 = 10

  - ls(pattern="x")

  - To get a list of all masked objects use the function conflicts.

Removing workspace objects

- If you assign a value to an object that already exists then the contents of the object will be overwritten with the new value (without a warning!).

- Use the function rm to remove one or more objects from your session.

```
> rm(x, x2)
```

Command History

Command History

Commands are entered interactively at the **R** user prompt. **Up** and **down arrow keys** scroll through your command history.

Command History

```
# work with your previous commands
    history()     #display last 25 commands
    history(max.show=Inf)
                        #display all previous commands

# save your command history
      savehistory(file="myfile") # default is ".Rhistory"
# recall your command history
      loadhistory(file="myfile") # default is ".Rhistory“
```

Default datasets in R

Datasets in R

R comes with a number of sample datasets that you can experiment with. Type

> data( )

to see the available datasets. The results will depend on which packages you have loaded. Type

help(*datasetname***)**

for details on a sample dataset.