

Introduction to Data Analysis in R-Studio

Raim,Ostrouchov,Neerchal

UMBC

About R

- R is an open source statistical programming language and environment for statistical computing and graphics
- R supports user defined functions, and is capable of run-time calls to C, C++, Fortran codes
- Capability of R can be extended by packages (1300)
- R needs to be downloaded from <https://cran.r-project.org/> and installed (like any other software)
- Available for Windows, Mac or Linux
- R feels and looks are the same regardless of the underlying operating system (for the most part)
- Developed by Ross Ihaka and Robert Gentleman, University of Auckland, in 1995.

Installing R

The screenshot shows the official website for The R Project for Statistical Computing. At the top, there's a navigation bar with icons for back, forward, search, and other site functions. The URL https://www.r-project.org is visible. The main title "The R Project for Statistical Computing" is prominently displayed above the "Getting Started" section. Below the title, there's a brief introduction to R, a "News" section with a list of recent releases, and links to various R-related projects and documentation.

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and Mac OS. To download R, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers](#) to frequently asked questions before you send an email.

News

- The [useR! 2017 conference](#) will take place in Brussels, July 4 - 7, 2017, and details will be appear here in due course.
- R version 3.3.1 ([Bug in Your Hair](#)) has been released on Tuesday 2016-06-21.
- R version 3.2.5 ([Very, Very Secure Dishes](#)) has been released on 2016-04-14. This is a rebadging of the quick-fix release 3.2.4-revised.
- Notice [XQuartz users \(Mac OS X\)](#) A security issue has been detected with the Sparkle update mechanism used by XQuartz. Avoid updating over insecure channels.
- The [R Logo](#) is available for download in high-resolution PNG or SVG formats.
- useR! 2016, has taken place at Stanford University, CA, USA, June 27 - June 30, 2016.
- The [R Journal Volume 7/2](#) is available.
- R version 3.2.3 ([Wooden Christmas-Tree](#)) has been released on 2015-12-10.
- R version 3.1.3 ([Smooth Sidewalk](#)) has been released on 2015-03-09.

R Project

[\[Home\]](#)

Download

[CRAN](#)

R Project

[About R](#)

[Logo](#)

[Contributors](#)

[What's New?](#)

[Mailing Lists](#)

[Reporting Bugs](#)

[Development Site](#)

[Conferences](#)

[Search](#)

R Foundation

[Foundation](#)

[Board](#)

[Members](#)

[Donors](#)

[Donate](#)

Documentation

[Manuals](#)

[FAQs](#)

[The R Journal](#)

[Books](#)

[Certification](#)

Installing R

<https://www.stats.ox.ac.uk/R/>
<http://stat.ethz.ch/CRAN/>

Taiwan
<http://ftp.yzu.edu.tw/CRAN/>
<http://cran.csie.ntu.edu.tw/>

Thailand
<http://mirrors.psu.ac.th/pub/cran/>

Turkey
<http://cran.r-project.org/>
<http://cran.ncr.metu.edu.tr/>

UK
<https://www.stats.bris.ac.uk/R/>
<http://www.stats.bris.ac.uk/R/>
<https://mirrors.ebi.ac.uk/CRAN/>
<http://mirrors.ebi.ac.uk/CRAN/>
<http://cran.ma.imperial.ac.uk/>
<https://cran.r-project.org/>

ETH ZÜRICH
ETH Zürich

Department of Computer Science and Engineering, Yuan Ze University
National Taiwan University, Taipei

Prince of Songkla University, Hatyai

Pamukkale University, Denizli
Middle East Technical University Northern Cyprus Campus, Mersin

University of Bristol
University of Bristol
EMBL-EBI (European Bioinformatics Institute)
EMBL-EBI (European Bioinformatics Institute)
Imperial College London

Installing R

The screenshot shows a web browser displaying the CRAN (Comprehensive R Archive Network) website at mirrors.psu.ac.th/pub/cran/. The page title is "The Comprehensive R Archive Network". The main content area is titled "Download and Install R". It contains instructions for Windows and Mac users to download precompiled binary distributions. Below this, a list of download links is provided: "Download R for Linux", "Download R for (Mac) OS X", and "Download R for Windows". The link for "Download R for Windows" is circled in red. A note below the links states: "R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above." Another section, "Source Code for all Platforms", provides links for Windows and Mac users to download precompiled binaries, while Linux users are directed to their package management systems. A third section, "Questions About R", contains a single bullet point: "If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email." At the bottom of the page, a footer links to "What are R and CRAN?"

mirrors.psu.ac.th/pub/cran/ The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Tuesday 2016-06-21, Bug in Your Hair) [R-3.3.1.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

What are R and CRAN?

Installing R

The screenshot shows a web browser window with the URL mmon-psu.ac.th/pub/cran/. The page title is "R for Windows". On the left, there's a large blue "R" logo. Below it are links for "CRAN", "Mirrors", "What's new?", "Task Views", and "Search". Under "About R", there are links for "R Homepage" and "The R Journal". Further down are links for "Software", "R Sources", "R Binaries", and "Packages". The main content area has a heading "Subdirectories:" followed by a list of directory names: "base", "contrib", "old contrib", and "Rtools". To the right of each name is a brief description. A callout bubble highlights the link "[install R for the first time](#)". At the bottom of the page, there are notes about not submitting binaries to CRAN, reading the R FAQ and Windows FAQ, and being cautious with executables.

Subdirectories:

- [base](#) Binaries for base distribution (managed by Duncan Murdoch). This is what you want to [install R for the first time](#).
- [contrib](#) Binaries of contributed CRAN packages (for $R \geq 2.11.x$; managed by Uwe Ligges). There is also information on [third party software](#) available for CRAN Windows services and corresponding environment and make variables.
- [old contrib](#) Binaries of contributed CRAN packages for outdated versions of R (for $R < 2.11.x$; managed by Uwe Ligges).
- [Rtools](#) Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

Installing R

The screenshot shows a web browser window with the URL mirrors.psu.ac.th/pub/cran/ in the address bar. The main content is the R 3.3.1 for Windows (32/64 bit) download page. A red oval highlights the URL in the address bar. Another red oval highlights the download link "Download R 3.3.1 for Windows". The page includes links for "Installation and other instructions" and "New features in this version". Below these, a note about verifying the package's integrity using md5sum is displayed. A "Frequently asked questions" section lists three bullet points: "Does R run under my version of Windows?", "How do I update packages in my previous version of R?", and "Should I run 32-bit or 64-bit R?". A "Software" sidebar on the left lists various R-related resources like CRAN Mirrors, What's new?, Task Views, Search, About R, R Homepages, and The R Journal. A "Documentation" sidebar lists Manuals, FAQs, and Contributed packages. At the bottom, a note for webmasters and the last change date (2016-06-21 by Duncan Murdoch) are shown. A red oval highlights the URL mirrors.psu.ac.th/pub/cran/bin/windows/base/R-3.3.1-win.exe at the very bottom of the page.

Installing R

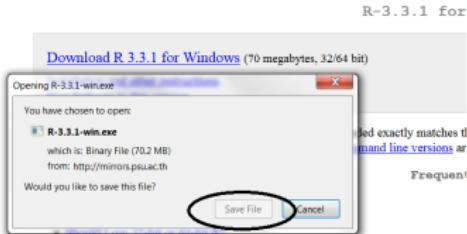


CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

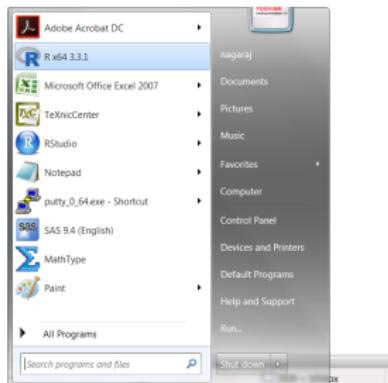


Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for W

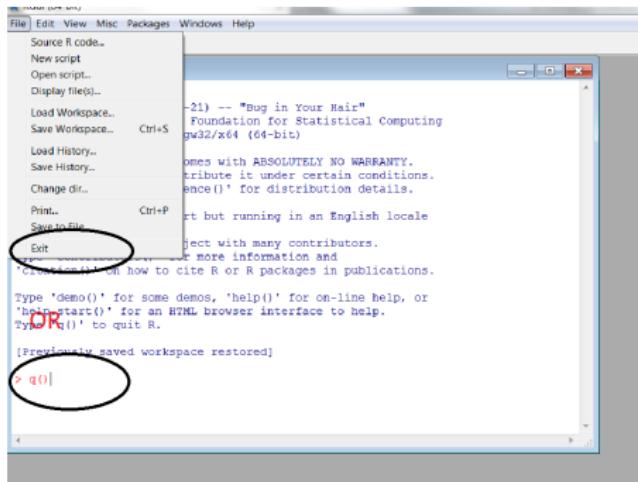
- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release) is available at [r-devel binary packages](#).
- [Previous releases](#)

Note to webmasters: A stable link which will redirect to the current Windows binary is
<http://<CRAN MIRROR>/bin/windows/base/release.htm>

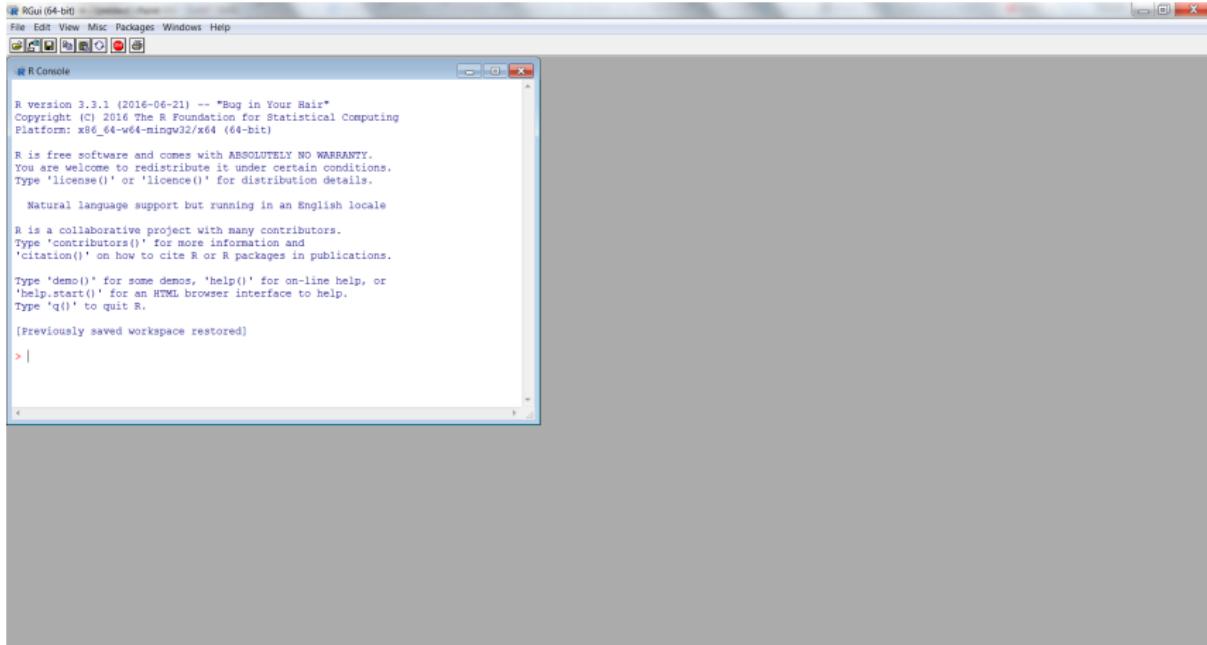
Starting R



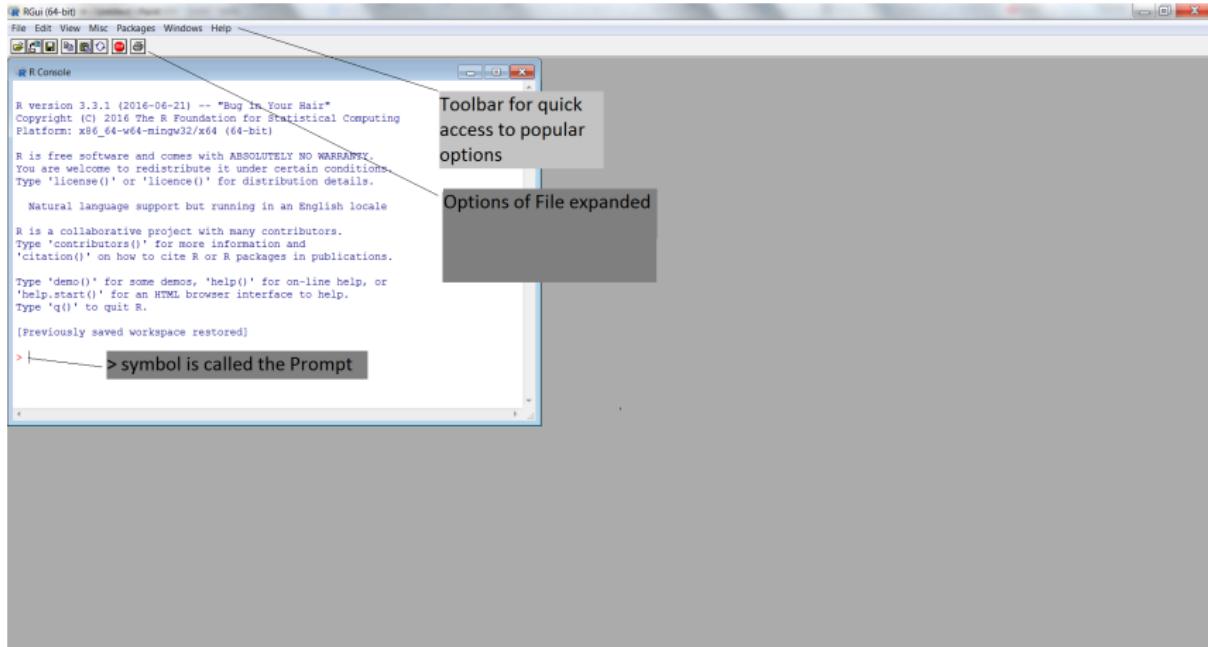
Quitting or Exiting R



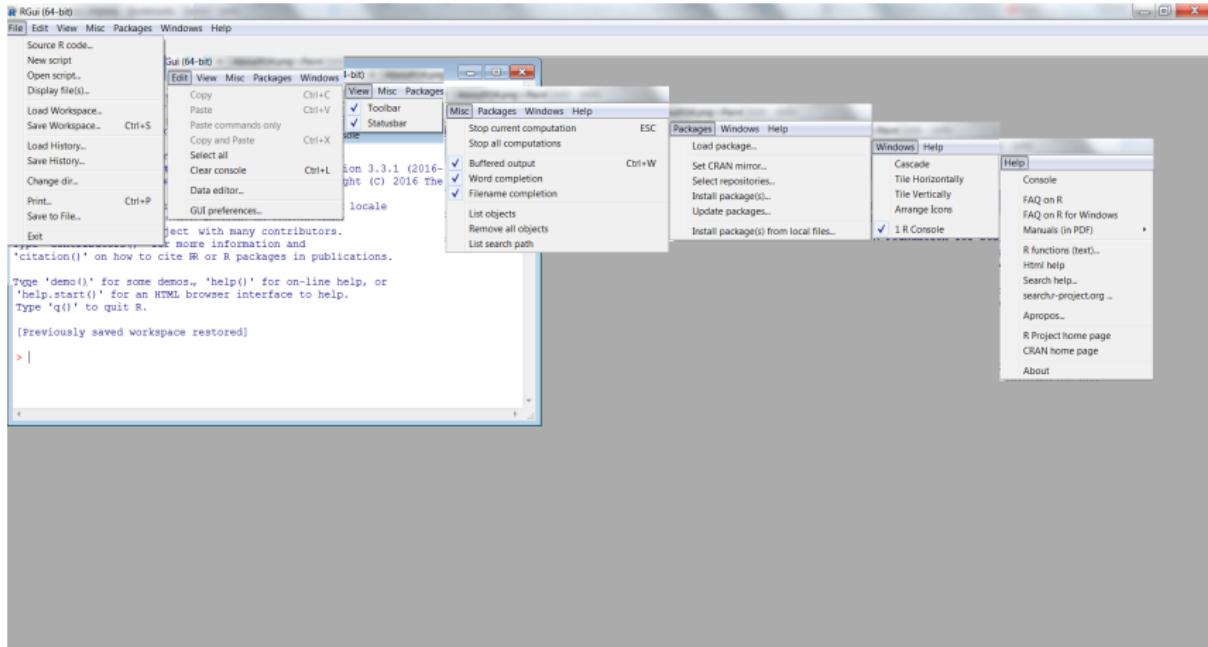
R Interface



R Interface Components

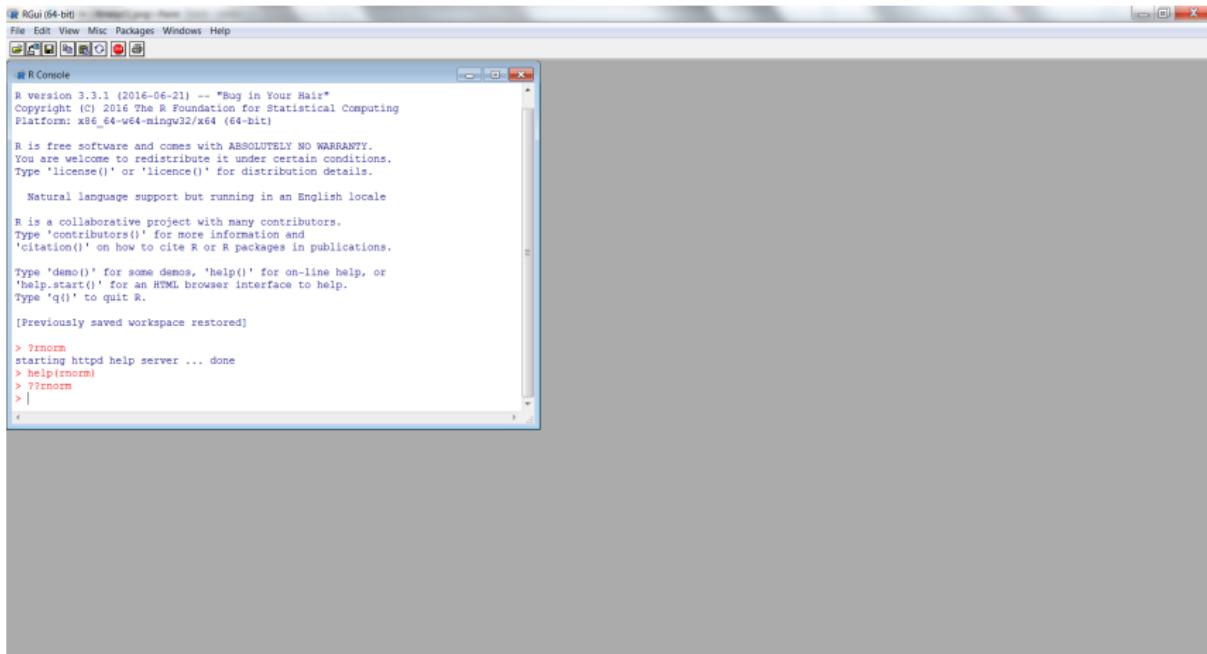


The menu tree of plain vanilla R



Help in R

- R has an easy-to-use help facility system. For example, if you want to get more information about function "rnorm", you can either type the command:
`> help(rnorm) or >?rnorm`



The screenshot shows the RGui (64-bit) application window. The title bar reads "RGui (64-bit)". The menu bar includes File, Edit, View, Misc, Packages, Windows, Help. The main window is titled "R Console". It displays the standard R startup message, including the version (3.3.1), copyright notice, and license information. Below this, the console shows the user's session, starting with the command `> ?rnorm`. The R help system starts a local HTTP server to serve documentation, and the user types `help(rnorm)` and `?rnorm` to search for information about the rnorm function.

```
R version 3.3.1 (2016-06-21) -- "Bug in Your Hair"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> ?rnorm
starting httpd help server ... done
> help(rnorm)
> ?rnorm
> |
```

Help in R

- `>help(rnorm)` or `>?rnorm` or `>??rnorm` will take us to a help page

The image shows two side-by-side screenshots of a web browser displaying R documentation. The left screenshot shows the help page for the `Normal` distribution, which includes the function signature, usage examples, and arguments. The right screenshot shows the search results for `rnorm`, listing related functions and vignettes.

Normal (stats)
The Normal Distribution
Description
Density, distribution function, quantile function and random generation for the normal distribution with mean equal to `mean` and standard deviation equal to `sd`.
Usage

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

Arguments

?rnorm or help(rnorm)

Search Results
R

Vignettes:

| | | | |
|--|----------------------|------------------------|------------------------|
| lme4::lmerPerf lmer Performance Tips | HTML | source | R code |
| mvtnorm::MVT_Rnews Using mvtnorm | PDF | source | R code |
| R6::Performance Reference and R6 class performance tests | HTML | source | R code |

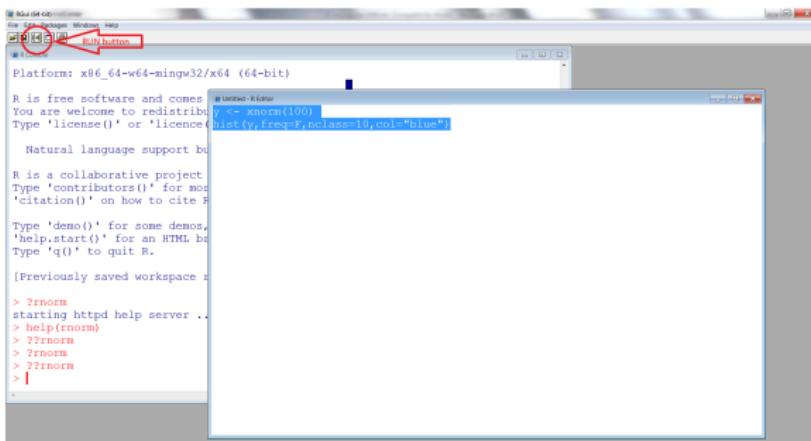
Help pages:

| | |
|--|---|
| car::testTransform | Likelihood-Ratio Tests for Univariate or Multivariate Power Transformations to Normality |
| e1071::fclustIndex | Fuzzy Cluster Indexes (Validity/Performance Measures) |
| gridExtra::annotation_raster | Annotation: High-performance rectangular tiling. |
| gridExtra::grid.arrange | Wrap up a selection of summary functions from Hmisc to make it easy to use with 'stat_summary'. |
| itable::z_normalise | Normalizes z values within a gtble object |
| Hmisc::smean.cl.normal | Compute Summary Statistics on a Vector |
| Hmisc::spower | Simulate Power of 2-Sample Test for Survival under Complex Conditions |
| knitr::knit_print | A custom printing function |
| matrixcalc::entrywise_norm | Compute the entrywise norm of a matrix |
| matrixcalc::frobenius.norm | Compute the Frobenius norm of a matrix |
| matrixcalc::hilbert.schmidt.norm | Compute the Hilbert-Schmidt norm of a matrix |
| matrixcalc::inf.norm | Compute the infinity norm of a matrix |

??rnorm

Writing a Program in R

- To create a program, File → New Script; Type the program into the editor
- To execute the program, select all program by CTRL+A, and then press the run button. A histogram will appear in the graphics window
- To save the R script, either click on the Save icon or from selecting the file menu: File → Save as to have the Save as dialogue box. Follow the prompts to save the script file.

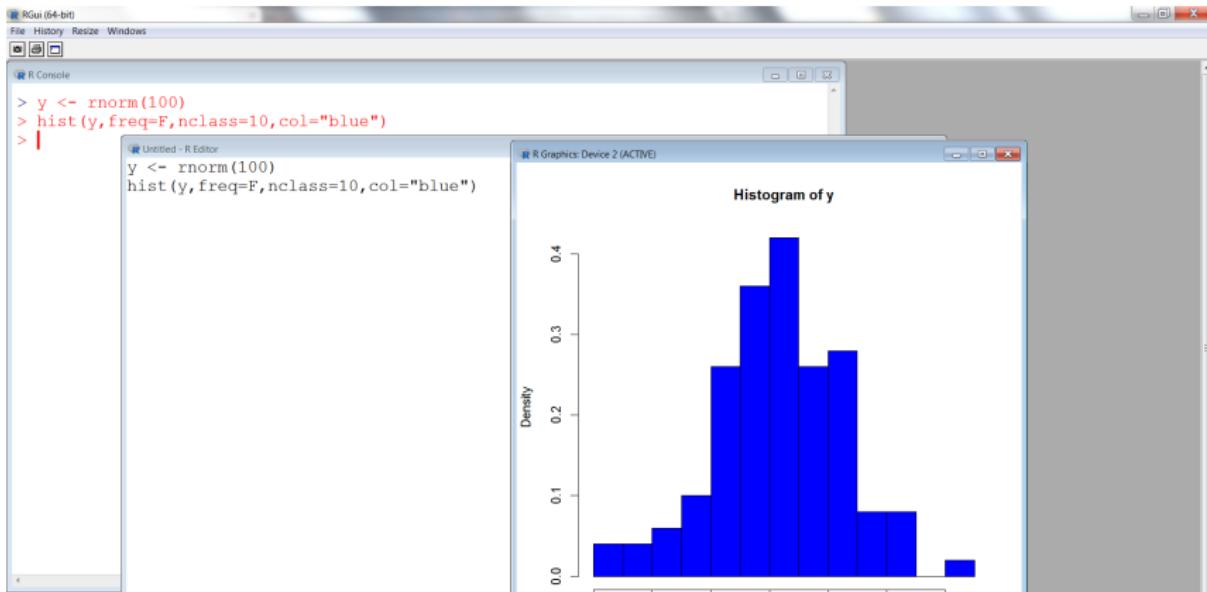


The screenshot shows the R console window with a histogram displayed in the plot area. The console output is as follows:

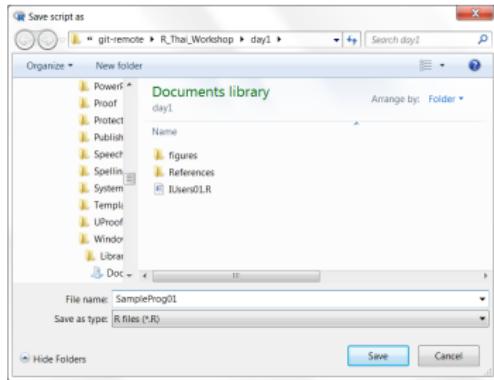
```
R> x <- rnorm(100)
R> hist(x,freq=F,nbins=10,col="blue")
```

A red circle highlights the "Run button" icon in the toolbar above the console window.

Running an R Program



Saving a Program as .R file (Script)



About R Studio

- RStudio is an integrated development environment (IDE) for R.
- RStudio allows the user to run R in a more user-friendly environment.
- Can be downloaded from <http://www.rstudio.com/> either open source or commercial editions
- Runs on the desktop (Windows, Mac, and Linux) or in a browser connected to RStudio Server
- RStudio looks and feels the same regardless of the platform
- Created by J.J. Allaire in 2008. Supported by Foundation for Open Access Statistics (FOAS) and R Consortium
- To start either click on the RStudio icon or by type library("Rstudio")

Installing RStudio

The screenshot shows the RStudio website at https://www.rstudio.com/. The page has a blue header bar with the RStudio logo and navigation links for Products, Resources, Pricing, About Us, Blog, and a search icon. Below the header is a large blue section containing the text "Welcome to RStudio - Open source and enterprise-ready professional software for R". It features three buttons: "Download RStudio" (which is highlighted with a black oval), "Discover Shiny", and "shinyapps.io Login". To the right is a large circular graphic with a blue gradient and a white stylized letter "R" inside. At the bottom of the main content area are three small white circles followed by three blue semi-circular arrows pointing right.

Welcome to RStudio - Open source
and enterprise-ready professional
software for R

Download RStudio

Discover Shiny

shinyapps.io Login

Installing RStudio

The screenshot shows the RStudio website at <https://www.rstudio.com/products/rstudio/>. The page features a large blue header with the RStudio logo. Below the header, a main heading says "Take control of your R code". A text block explains that RStudio is an IDE for R, supporting a console and syntax-highlighting. To the right is a dark sidebar with a heart icon and a paper airplane icon. The main content area has two sections: "License" (AGPL v3) and "Pricing" (Free). A large blue button labeled "DOWNLOAD RSTUDIO DESKTOP" is highlighted with a black oval. Another blue button labeled "BUY NOW" is also visible.

Products Resources Pricing About Us Blog

Take control of your R code

RStudio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports

License AGPL v3

Pricing Free

[DOWNLOAD RSTUDIO DESKTOP](#)

[BUY NOW](#)

Installing RStudio

The screenshot shows the RStudio website at <https://www.rstudio.com/products/rstudio/download/>. The page displays a comparison chart of RStudio's features across five columns: Integrated Development Environment for R, Priority support, Access via Web Browser, Enterprise Security and Access Controls, Project Sharing, Access to Multiple Versions of R, Multiple Concurrent Sessions, Administrative Dashboard, and Load Balancing and Resource Management. Below the chart, there is a section about RStudio's purpose, a note for Linux server users, and a link for commercial support. At the bottom, there are four download buttons: AGPL (circled in black), Commercial, AGPL, and Commercial.

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

If you run R on a Linux server and want to enable users to remotely access RStudio using a web browser please download RStudio Server.

Do you need support or a commercial license? Check out our commercial offerings.

<https://www.rstudio.com/products/rstudio/download/#download> | RStudio Desktop 0.99.903 — Release Notes

Installing RStudio

The screenshot shows the RStudio download page at <https://www.rstudio.com/products/rstudio/download/>. The desktop installers section is highlighted with a red oval.

Installers for Supported Platforms

| | Size | Date | MD5 |
|--|---------|------------|-------------------------------------|
| RStudio Desktop 0.99.903 - Windows 7/8/10 | 77.1 MB | 2016-07-18 | T16E228E21143c0e21f4acca3752e2024f8 |
| RStudio 0.99.903 - Ubuntu 12.04+ (Debian 8+) (32-bit) | 60 MB | 2016-07-18 | d14a150a5a5a0a03365079a0d4464d6 |
| RStudio 0.99.903 - Ubuntu 12.04+ (Debian 8+) (64-bit) | 82.8 MB | 2016-07-18 | 741aae10b0baa49a0d9051a0d2a2c44e0 |
| RStudio 0.99.903 - Fedora 23+ (Red Hat® openSUSE 12.1+ (32-bit)) | 88.5 MB | 2016-07-18 | 98ea593b01e007303e405351a77446 |
| RStudio 0.99.903 - Fedora 23+ (Red Hat® openSUSE 12.1+ (64-bit)) | 81.9 MB | 2016-07-18 | 0e9ek112303931790b959ef57e446e0 |
| RStudio 0.99.903 - Fedora 23+ (Red Hat® openSUSE 12.1+ (64-bit)) | 81.9 MB | 2016-07-18 | 1522f47235e4f904cf3354afbc7b09 |

Zip/Tarballs

| | Size | Date | MD5 |
|--|----------|------------|----------------------------------|
| RStudio 0.99.903 - Windows Vista™/7/8/10 | 103.8 MB | 2016-07-18 | 639817697093a1fc730e313e409133e1 |
| RStudio 0.99.903 - Ubuntu 12.04+ (Debian 8+) (32-bit) | 62.3 MB | 2016-07-18 | 4c2c15bafe0ed5202212ea7b2a2c03 |
| RStudio 0.99.903 - Ubuntu 12.04+ (Debian 8+) (64-bit) | 69.4 MB | 2016-07-18 | 4c123d514e39924120150079fb2 |
| RStudio 0.99.903 - Fedora 23+ (Red Hat® openSUSE 12.1+ (32-bit)) | 81.8 MB | 2016-07-18 | c05a4e533fb711f97442ea7aece3fb5 |
| RStudio 0.99.903 - Fedora 23+ (Red Hat® openSUSE 12.1+ (64-bit)) | 82.8 MB | 2016-07-18 | ad5761417fd40c4db7db21ax5303a |

Source Code

A tarball containing source code for RStudio v0.99.903 can be downloaded from [here](#).

Footer Information:

250 Northern Ave, Boston, MA 02210
844-448-1212
info@rstudio.com

Links:

- [DMCA](#)
- [Trademark](#)
- [Support](#)
- [ECCN](#)

Twitter: [@jthompson@rstudio](#) (Yes) Type first few letters, then Ctrl+Up (Cmd+Up on Mac) to see matches.
2 weeks ago

<https://download1.rstudio.org/RStudio-0.99.903.exe>

Installing RStudio

RStudio Desktop 0.99.903 — Release Notes

RStudio requires R 2.11.1+. If you don't already have R, download it [here](#).

Installers for Supported Platforms

| Platform | Size | Date | MD5 |
|--|---------|------------|------------------------------------|
| RStudio 0.99.903 - Windows Vista/7/8/10 | 77.1 MB | 2016-07-18 | T16Z2BZC2143c0e21f4acca3752e2021e5 |
| RStudio 0.99.903 - Mac OS X 10.8+ (64-bit) | 60 MB | 2016-07-18 | d144e150fa5a5aa038365079a0d4464d6 |
| RStudio 0.99.903 - Ubuntu 12.04+ (64-bit) | 82.8 MB | 2016-07-18 | 741a4a10b0ba494cd9051a02a2c41e2 |
| RStudio 0.99.903 - Ubuntu 12.04+ (32-bit) | 88.3 MB | 2016-07-18 | 9feea593b01e07830e+05351a7746d |
| RStudio 0.99.903 - Fedora 20+ (64-bit) | 81.1 MB | 2016-07-18 | 0e3ek11230393179a0b95ef57e4469e7 |
| RStudio 0.99.903 - Fedora 20+ (32-bit) | 81.3 MB | 2016-07-18 | 152z247235e54904cf3354afbc7b3b9 |

Zip/Tarballs

| Platform | Size | Date | MD5 |
|---|----------|------------|----------------------------------|
| RStudio 0.99.903 - Windows Vista/7/8/10 | 110.8 MB | 2016-07-18 | 53981769791a9fcf3ba311e49133e1d |
| RStudio 0.99.903 - Ubuntu 12.04+ (64-bit) | 82.3 MB | 2016-07-18 | 42c215be86ed05202212ea7b-a2c0304 |
| RStudio 0.99.903 - Ubuntu 12.04+ (32-bit) | 88.0 MB | 2016-07-18 | 41c123d14e399294125017fb2a7fb2 |
| RStudio 0.99.903 - Fedora 20+ (64-bit) | 81.8 MB | 2016-07-18 | c03a4e53eb711f9714d9a7aee3fb5 |
| RStudio 0.99.903 - Fedora 20+ (32-bit) | 82.8 MB | 2016-07-18 | ad5761417fd7cc4db7dfb21ax53303a |

Source Code

A tarball containing source code for RStudio v0.99.903 can be downloaded from [here](#).

250 Northern Ave, Boston, MA 02210
844-448-1212
info@rstudio.com

RStudio

DMCA
Trademark
Support
ECCN

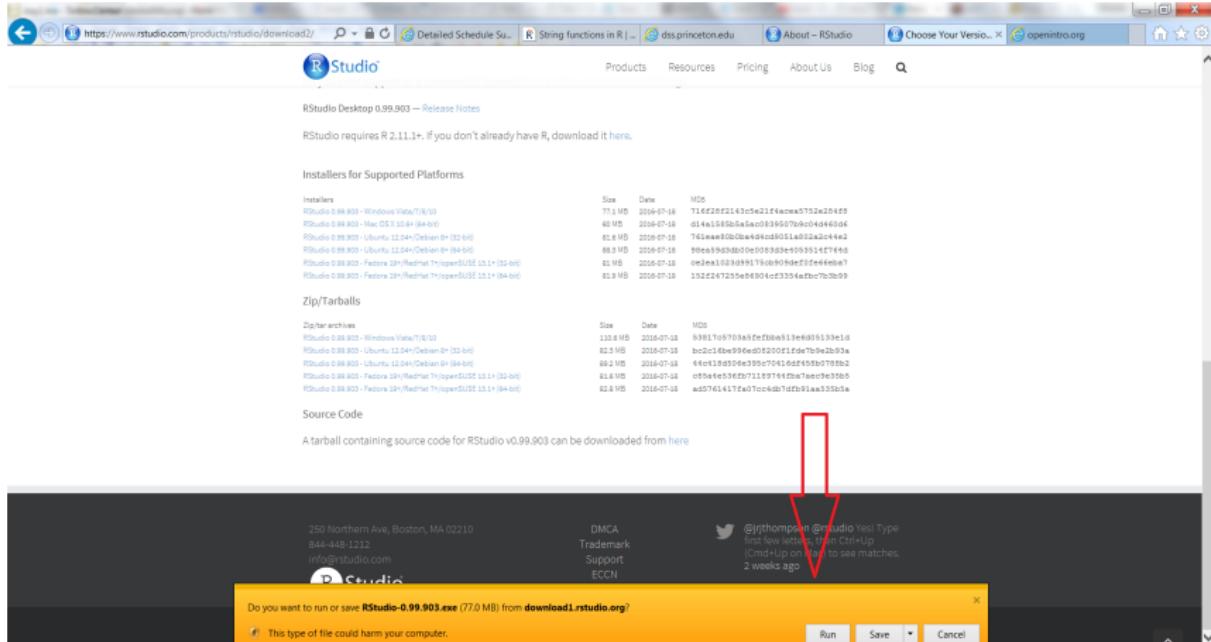
Do you want to run or save **RStudio-0.99.903.exe** (77.0 MB) from [download1.rstudio.org](https://www.rstudio.com/products/rstudio/download/)?

This type of file could harm your computer.

Run Save Cancel

Twitter icon: @jthompson @rstudio
Type first few letters, then Ctrl+Up
(Cmd+Up on Mac) to see matches.
2 weeks ago

Installing RStudio



RStudio Desktop 0.99.903 — Release Notes

RStudio requires R 2.11.1+. If you don't already have R, download it [here](#).

Installers for Supported Platforms

| | Size | Date | MD5 |
|--|---------|------------|-----------------------------------|
| RStudio 0.99.903 - Windows Vista (7.0) | 77.1 MB | 2016-07-18 | T16228E2113c0e21f4acca3752e2021e5 |
| RStudio 0.99.903 - Mac OS X 10.8+ (64-bit) | 60.0 MB | 2016-07-18 | d144e150fa5a5aa0e3365079e0d4464d6 |
| RStudio 0.99.903 - Ubuntu 12.04+ (64-bit) | 82.8 MB | 2016-07-18 | 741aae10b0baa494cd9051ea02a2c4e2 |
| RStudio 0.99.903 - Ubuntu 12.04+ (32-bit) | 88.3 MB | 2016-07-18 | 9feae593b01e07830e+05351a7746d |
| RStudio 0.99.903 - Fedora 20+ (64-bit) | 81.1 MB | 2016-07-18 | ce9ek11230393179e0b95ef57e4469e7 |
| RStudio 0.99.903 - Fedora 20+ (32-bit) | 81.3 MB | 2016-07-18 | 1522f47235e54904cf3354afbc7b3b9 |

Zip/Tarballs

| | Size | Date | MD5 |
|---|----------|------------|----------------------------------|
| RStudio 0.99.903 - Windows Vista (7.0) | 110.8 MB | 2016-07-18 | 63911767919a1fcf3ba311e401133e1d |
| RStudio 0.99.903 - Ubuntu 12.04+ (64-bit) | 82.2 MB | 2016-07-18 | 2c2c15be86ed05202212e7b-ec2a3034 |
| RStudio 0.99.903 - Ubuntu 12.04+ (32-bit) | 89.0 MB | 2016-07-18 | 4e123d514e39929412501579fb2 |
| RStudio 0.99.903 - Fedora 20+ (64-bit) | 81.8 MB | 2016-07-18 | c03a4e53fb711f9714d9a7aee3fb5 |
| RStudio 0.99.903 - Fedora 20+ (32-bit) | 82.0 MB | 2016-07-18 | ad5761417fd40cc4db7dfb1a55393a |

Source Code

A tarball containing source code for RStudio v0.99.903 can be downloaded from [here](#).

250 Northern Ave, Boston, MA 02210
844-448-1212
info@rstudio.com

RStudio

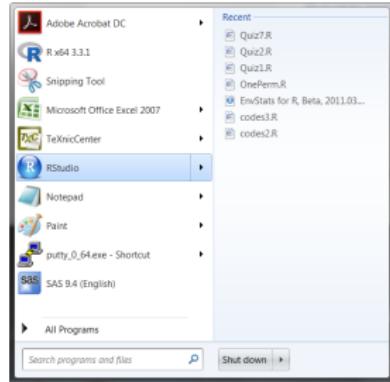
DMCA
Trademark
Support
ECCN

[@jthompson@rstudio](#) Yes Type first few letters, then Ctrl+Up (Cmd+Up on Mac) to see matches.
2 weeks ago

Do you want to run or save **RStudio-0.99.903.exe** (77.0 MB) from [download1.rstudio.org](#)?
This type of file could harm your computer.

Run Save Cancel

Starting RStudio



RStudio Dashboard

The screenshot shows the RStudio interface with the following components:

- Top Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Tools, Help.
- Environment Tab:** Shows the R code and its output:

```
library("Rcmdr", lib.loc="~/R/win-library/3.2")
detach("package:Rcmdr", unload=TRUE)
```
- Packages Tab:** Shows a list of installed packages with their descriptions and versions:

| Name | Description | Version |
|------------|---|---------|
| abind | Combine Multidimensional Arrays | 1.4-5 |
| acepack | ace() and avas() for selecting regression transformations | 1.3-3.3 |
| afex | Another Fit All Effects: stemleaf, bogplot, faces, spm3R, plotsometry, plotfulls, and some slider functions | 1.3-0 |
| base64enc | Tools for base64 encoding | 0.1-3 |
| bitops | Bitwise Operations | 1.0-6 |
| car | Companion to Applied Regression | 2.3-2 |
| caTools | Tools: moving window statistics, GB, Base64, ROC AUC, etc. | 1.27.1 |
| chron | Chronological Objects which can Handle Dates and Times | 2.3-47 |
| colorspace | Output Analysis and Diagnostics for MCMC | 0.19-1 |
| data.table | Color Space Manipulation | 1.2-6 |
| dichromat | Extension of Dataframe | 1.3-6 |
| digest | Color Schemes for Dichromats | 2.0-0 |
| e1071 | Create Compact Hash Digests of R Objects | 0.8-8 |
| effects | Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien | 1.6-7 |
| evaluate | Effect Displays for Linear, Generalized Linear, and Other Models | 3.2-1 |
| formatR | Parsing and Evaluation Tools that Provide More Details than the Default | 0.8 |
| Formulas | Format R Code Automatically | 1.4 |
| gridExtra | Extended Model Formulas | 1.2-1 |
| grid | An Implementation of the Grammar of Graphics | 2.2.0 |
- Left Panel:** Shows the R console output:

```
y
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from ~/.RData]
```
- Bottom Right:** Project pane labeled "Project (None)".

Quitting RStudio

The screenshot shows the RStudio application window. The 'File' menu is open, displaying various file operations like 'New File', 'Open File...', 'Save', and 'Quit RStudio...'. The 'Quit RStudio...' option is circled in red. Below the menu, there's a message about R being free software. The main workspace shows a script named 'Untitled2.R' with some R code. The bottom left shows the console output: '[workspace loaded from ~/.RData]' and '> | q()'. The bottom right shows the package manager interface.

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

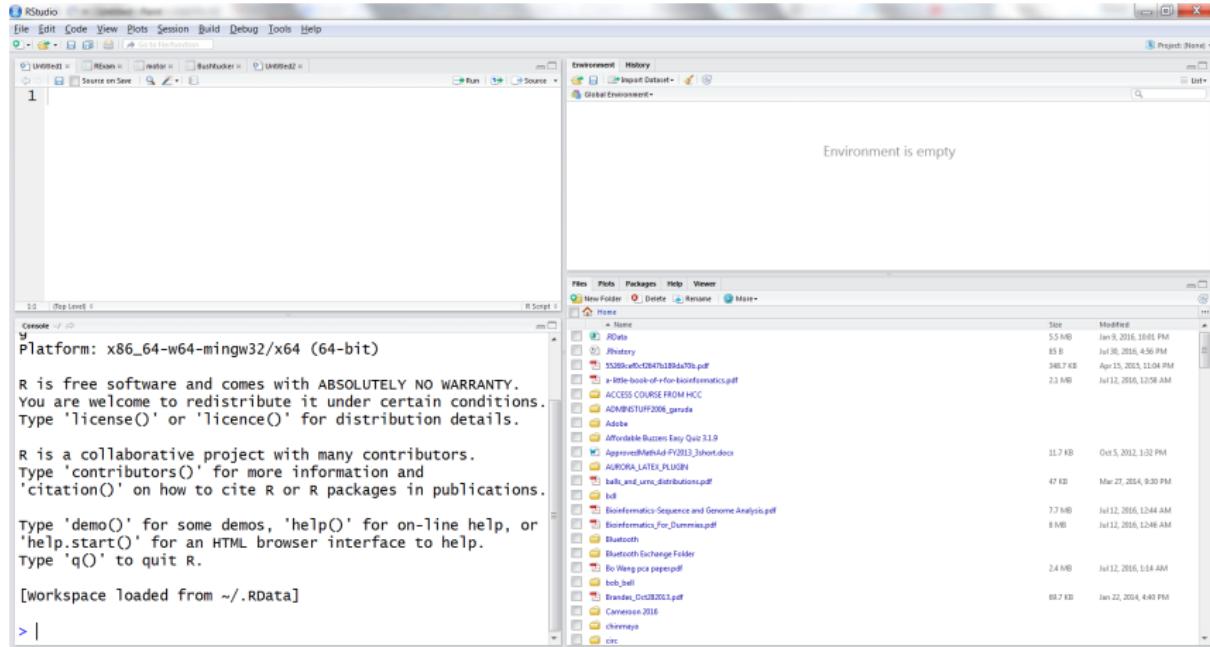
QR
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from ~/.RData]

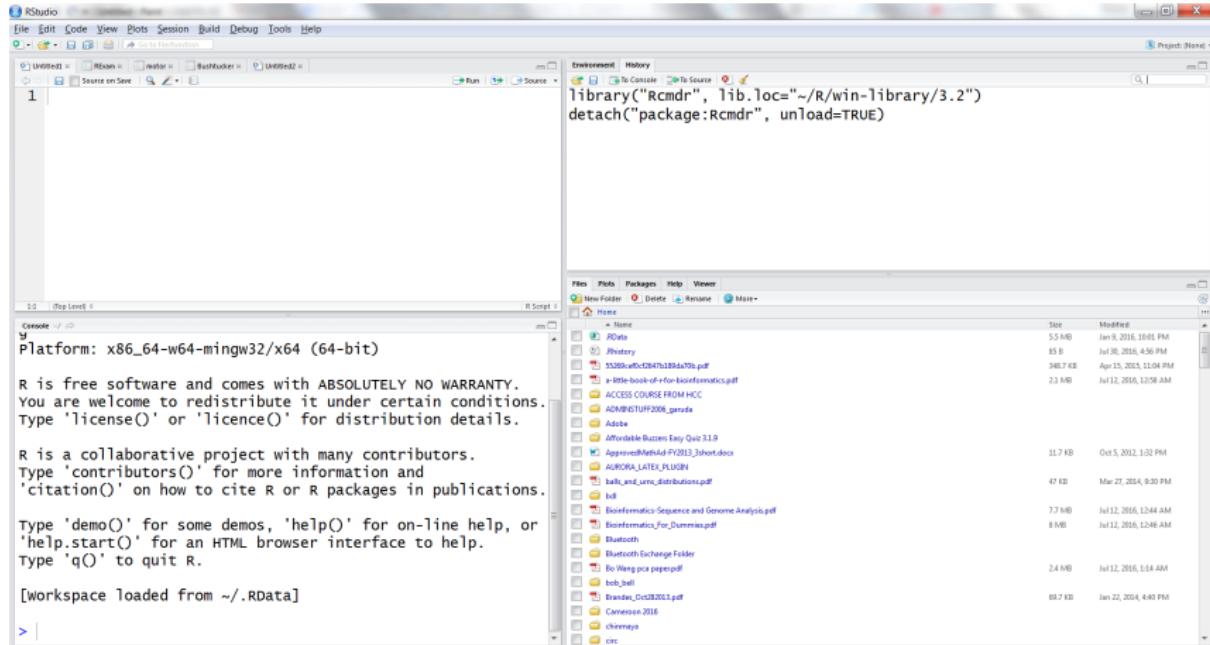
> | q()

| Name | Description | Version |
|------------|--|---------|
| abind | Combine Multidimensional Arrays | 1.4-5 |
| acepack | ace() and avas() for selecting regression transformations | 1.3-3.3 |
| afex | Another Fit All Possible Effects | 1.3.0 |
| arm | Data Analysis Using Regression and Multilevel/Hierarchical Models | 1.8-6 |
| base64enc | Tools for base64 encoding | 0.1-3 |
| bitops | Bitwise Operations | 1.0-6 |
| car | Companion to Applied Regression | 2.2-2 |
| caTools | Tools involving window statistics, GIF, Base64, ROC AUC, etc. | 1.27.1 |
| chron | Chronological Objects which can Handle Dates and Times | 2.3-47 |
| colorspace | Output Analysis and Diagnostics for MCMC | 0.19-1 |
| data.table | Extensions of Dataframe | 1.2-6 |
| dichromat | Color Schemes for Dichromats | 1.8.0 |
| digest | Create Compact Hash Digests of R Objects | 0.8.8 |
| effDD | Miss Displays of the Department of Statistics, Probability Theory Group (Formerly: EI071), TU Wien | 1.8-7 |
| effects | Effect Displays for Linear, Generalized Linear, and Other Models | 3.2-1 |
| evaluate | Parsing and Evaluation Tools that Provide More Details than the Default | 0.8 |
| formatR | Format R Code Automatically | 1.4 |
| Formulas | Extended Model Formulas | 1.2.1 |
| gridExtra | An Implementation of the Grammar of Graphics | 2.2.0 |

RStudio Dashboard: Windows and Panes



RStudio Dashboard: Windows and Panes



RStudio Dashboard: Windows and Panes

The screenshot displays the RStudio desktop application with several windows open:

- Environment pane:** Shows the R environment with a history of commands. The current command is:

```
library("Rcmdr", lib.loc="~/R/win-library/3.2")
detach(package:Rcmdr", unload=TRUE)
```
- Code pane:** Shows the R code being run.
- Console pane:** Shows the R console output, including the platform information and various help messages.
- File browser pane:** Shows the file system structure under the "Home" directory. The contents include RData, RHistory, and numerous PDF files and other documents.
- Plot pane:** Shows a small preview of a plot.
- Session pane:** Shows the workspace loaded from `~/.RData`.
- Help pane:** Shows the help page for `q()`.

RStudio Dashboard: Windows and Panes

Usually, RStudio Dashboard has four windows: 1) Console. 2) Workspace and history. 3) Files, plots, packages and help. 4)The R script(s) and data view. Each window has one or more panes (or tabs).

- R commands are entered and submitted to run in the **Console**
- **Environment** pane shows all the active objects. The **history** pane shows a list of commands used so far.
- **R Script or Source Editor** is used for creating scripts and save it for later or recalling and running a saved script or also to view R objects
- The **Packages** shows all the packages that are available to be loaded. This pane also provides *search* and *install*.
- **Help** can be used to get help on functions and packages
- **Viewer** pane is used for viewing local web content

Installing and Loading a package to RStudio

- Activate the pane **packages**
- Enter the first few letters from the name of the package. RStudio will list the package(s) starting with those letters
- Check the package to be loaded
- If the package is not already installed, click on the install button and a dialog box will appear to walk you through the steps to install. Once installed, it will appear in this list to be loaded (or checked off).

RStudio: Installing a package

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays R code for generating a histogram.
- Cards View:** Shows the output of the R code, including a warning about R Commander and errors related to the 'car' package.
- Environment Viewer:** A table showing the current environment variables:

| Name | Type | Length | Size | Value |
|--------|------------|--------|---------|-----------------------------|
| trt | factor | 60 | 928 B | Factor w/ 5 levels "1",... |
| viewer | function | 1 | 7 KB | function (url, height,...) |
| wide | data.frame | 12 | 5.1 KB | 6 obs. of 12 variables |
| x | numeric | 10000 | 7.6 MB | Large numeric (1000000 ...) |
| x1 | numeric | 10000 | 78.2 KB | num [1:10000] 0.7111 0... |
| x2 | numeric | 10000 | 78.2 KB | num [1:10000] 0.621 0.1... |
| x3 | numeric | 10000 | 78.2 KB | num [1:10000] 0.397 0.2... |
| x4 | numeric | 10000 | 78.2 KB | num [1:10000] 0.351 0.1... |
| x | numeric | 10000 | 78.2 KB | num [1:10000] 2.08 1.45 |

- Packages View:** Shows the installed packages: Rcmdr and RcmdrMisc.

RStudio: Installing a package

The screenshot shows the RStudio interface with the following components visible:

- Code Editor:** Shows R code for generating a histogram.
- Environment View:** Displays the current environment variables.
- Console:** Shows error messages related to R Commander and package dependencies.
- Install Packages Dialog:** A modal window titled "Install Packages" is open, showing the "Repository (CRAN, CRANextra)" dropdown set to "CRAN". The "Packages" field contains "psidt karray". The "Install dependencies" checkbox is checked. Buttons for "Install" and "Cancel" are at the bottom.

RStudio: Installing a package

The screenshot shows the RStudio interface with the following components:

- Top Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Tools, Help.
- Environment View:** Shows the global environment with objects like trt, viewer, wide, x, x1, x2, x3, x4, and v.
- Console View:** Displays the R session history:

```
1 y <- rnorm(100)
2 hist(y,freq=F,nclass=10,col="blue")
3
4
> library(package.car , uneload=TRUE)
Error: package 'car' is required by 'Rcmdr' so will not be detached
d
> install.packages("pastecs")
Installing package into 'C:/Users/nagaraj/Documents/R/win-library/3.3'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.3/paste
cs_1.3-18.zip'
Content type 'application/zip' length 1636284 bytes (1.6 MB)
downloaded 1.6 MB

package 'pastecs' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\nagaraj\AppData\Local\Temp\Rtmppestiwa\downloaded_
packages
>
```

RStudio: Installing a package

The screenshot shows the RStudio interface with the following details:

- Environment View:** Shows the global environment with objects like trt, viewer, wide, data.fra_, x, x1, x2, x3, x4, and v.
- Console View:** Displays the R session output:

```
1 y <- rnorm(100)
2 hist(y,freq=F,nclass=10,col="blue")
3
4
package 'pastecs' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
  C:\Users\nagaraj\AppData\Local\Temp\RtmpmestiwA\downloaded_
packages
> library('pastecs', lib.loc="~/R/win-library/3.3")
Loading required package: boot

Attaching package: 'boot'

The following object is masked from 'package:car':

  logit

> |
```
- Packages View:** Shows the 'pastecs' package listed with the following details:

| Name | Description | Version |
|---------|--|---------|
| pastecs | Package for Analytic of Space-Time Ecological Series | 1.3-18 |

RStudio: Installing a package

The screenshot shows the RStudio interface. In the Environment pane, there is a list of objects:

| Name | Type | Length | Size | Value |
|--------|------------|--------|---------|-----------------------------|
| trt | factor | 60 | 928 B | Factor w/ 5 levels '1',... |
| viewer | function | 1 | 7 KB | function (url, height,...) |
| wide | data.frame | 12 | 5.1 KB | 6 obs. of 12 variables |
| x | numeric | 10000 | 7.6 MB | Large numeric (1000000 ...) |
| x1 | numeric | 10000 | 78.2 KB | num [1:10000] 0.7111 0... |
| x2 | numeric | 10000 | 78.2 KB | num [1:10000] 0.621 0.1... |
| x3 | numeric | 10000 | 78.2 KB | num [1:10000] 0.397 0.2... |
| x4 | numeric | 10000 | 78.2 KB | num [1:10000] 0.351 0.1... |
| x | numeric | 10000 | 78.2 KB | num [1:10000] 2.08 1.45 |

In the Packages tab, the 'Available' section is highlighted with a red circle. It lists several packages:

| Name | Description | Version |
|-------------|---|----------|
| Rcmdr | R Commander | 2.2-5 |
| RcmdrMisc | R Commander Miscellaneous Functions | 1.0-4 |
| ColorBrewer | ColorBrewer Palettes | 1.1-2 |
| Rcpp | Seamless R and C++ Integration | 0.12.6 |
| RcppEigen | 'Rcpp' Integration for the 'Eigen' Templated Linear Algebra Library | 0.3.2.81 |
| arm | Data Analysis Using Regression and Multilevel/Hierarchical Models | 1.3-4 |
| highr | Syntax Highlighting for Source Code | 0.8 |
| THData | TH's Data Archive | 1.0-7 |

Note that RStudio comes with Rcmdr installed.
They work well together.

```
1 y <- rnorm(100)
2 hist(y,freq=F,nclass=10,col="blue")
3
4
Content-type: application/x-7z-compressed
length: 2020267
bytes: 1210 MB
downloaded 1.6 MB

package 'pastecs' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\nagaraj\AppData\Local\Temp\RtmppestiWA\downloaded_
packages
> library('pastecs', lib.loc="~/R/win-library/3.3")
Loading required package: boot

Attaching package: 'boot'

The following object is masked from 'package:car':

  logit

> |
```

RStudio Console

- Code completion
- Retrieving previous commands
- Console title bar – current working directory
- Using the command history
- Keyboard short cuts:
 - Ctrl+1 - Move focus to the Source Editor
 - Ctrl+2 - Move focus to the Console
 - Ctrl+L - Clear the Console
 - Esc - Interrupt R

Working inside RStudio

- To create a new file: File -> New File or Recent Files menu.
- To switch between many open files: View -> Switch to Tab...
RStudio supports syntax highlighting and other specialized code-editing features for: R scripts, R Markdown documents, HTML files, TeX documents, etc
- Code completion, Find-and-Replace, Comment/Uncomment blocks of code
- Executing the code (Running a program)
Select the lines and use the Run toolbar button
To run the entire document use the Source toolbar button.
- Extract a function
RStudio can automatically convert segments of codes into a function
Highlight a block of code and use Code -> Extract Function
- Code Folding and Sections

RStudio is an IDE

- RStudio is an Integrated Development Environment (IDE)
- It is not a Graphical User Interface (GUI) for statistical analysis
- Note that RStudio has a lot of cool tools for managing script and output files, but does not have a menu for computing descriptive statistics
- We will now discuss a couple of packages which convert R into a GUI based statistical analysis system

Useful Tips for Session Management

- Exiting and saving the script:
 - ▶ Helps keep track of the session
 - ▶ A saved script can be opened in a new session to start from the last point of save
 - ▶ Better option than saving the workspace
- Saving and printing the output
 - ▶ Cut (ctrl-c) -and- paste (ctrl-v) output (ongoing) into an open [word/text] report document
 - ▶ Best practice is to write notes as the output is moved to the report
 - ▶ Graphs may also be similarly pasted or use File > Save as method described earlier
- Commands generated by the R Commander appear in the script window:
 - ▶ This window can be edited so that only those commands that worked are saved
 - ▶ To send this script you have to highlight the relevant text and press the 'Submit' button.

Basic Programming

- When launching R we will see a prompt similar to below.

```
R version 3.3.1 (2016-06-21) -- "Bug in Your Hair"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

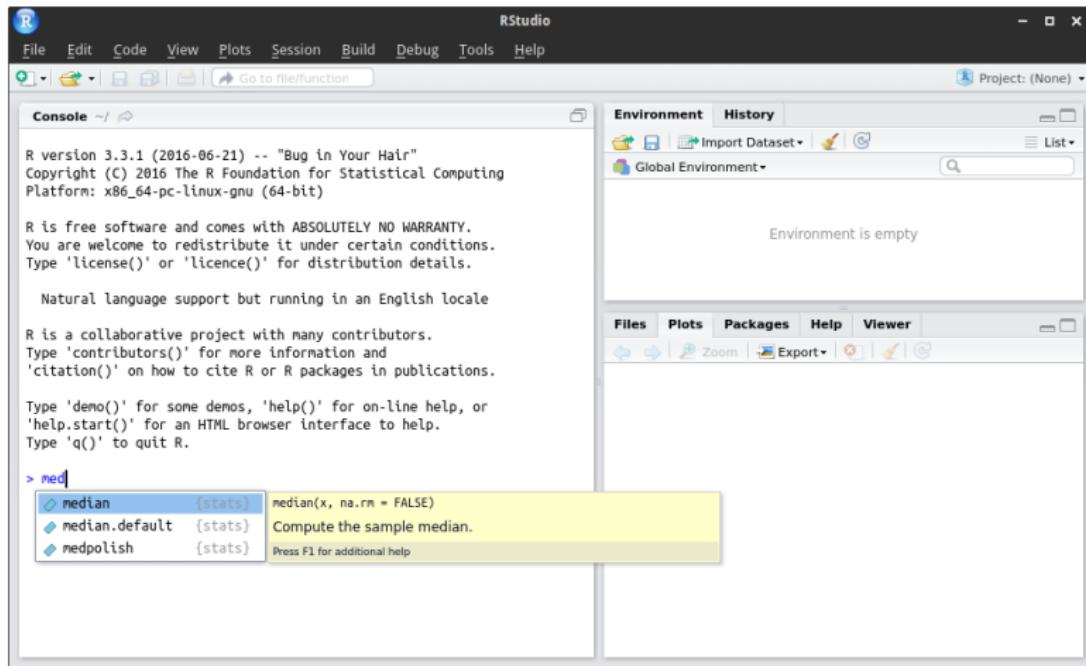
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> print("hello world")
[1] "hello world"
>
```

- We enter commands after the > character.

Command Completion

- Use tab or ctrl+space to get inline suggestions while you program.



- Sometimes suggestions will be offered as you type normally.

Help System

```
> ?median
```

The screenshot shows the R Help System window with the title "R: Median Value". The main content area displays the documentation for the `median` function. It includes sections for Description, Usage, Arguments, Details, and Examples. The `Usage` section shows the command `median(x, na.rm = FALSE)`. The `Arguments` section describes `x` as a numeric vector and `na.rm` as a logical value. The `Details` section notes that it's a generic function. The `Examples` section shows two examples of using the `median` function.

median {stats} R Documentation

Median Value

Description

Compute the sample median.

Usage

```
median(x, na.rm = FALSE)
```

Arguments

`x` an object for which a method has been defined, or a numeric vector containing the values whose median is to be computed.

`na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.

Details

This is a generic function for which methods can be written. However, the default method makes use of `is.na`, `sort` and `mean` from package **base** all of which are generic, and so the default method will work for most classes (e.g., ["Date"](#)) for which a median is a reasonable concept.

Value

The default method returns a length-one object of the same type as `x`, except when `x` is logical or integer of even length, when the result will be double.

If there are no values or if `na.rm = FALSE` and there are NA values the result is NA of the same type as `x` (or more generally the result of `x[FALSE][NA]`).

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[quantile](#) for general quantiles.

Examples

```
median(1:4)          # = 2.5 [even number]
median(c(1:3, 100, 1000)) # = 3 [odd, robust]
```

Arithmetic

- Basic arithmetic

```
> 2 + 3  
[1] 5  
> 2 - 3  
[1] -1  
> 2 * 3  
[1] 6  
> pi / 2  
[1] 1.570796  
> 2^3  
[1] 8  
> 21 %% 10  
[1] 1  
> 2 < 3  
[1] TRUE
```

- Built-in functions

```
> exp(1)  
[1] 2.718282  
> log(10)  
[1] 2.302585  
> log2(10)  
[1] 3.321928  
> abs(-1 * 1)  
[1] 1  
> sqrt(5)  
[1] 2.236068
```

Sequences

- Generate a sequence of integers.

```
> 1:9  
[1] 1 2 3 4 5 6 7 8 9  
> seq(1, 9)  
[1] 1 2 3 4 5 6 7 8 9  
> seq(from = 1, to = 9)  
[1] 1 2 3 4 5 6 7 8 9  
> seq(from = 1, to = 9, by = 2)  
[1] 1 3 5 7 9
```

- 1:9 is a shorthand for seq(1,9).
- Argument by has a default of 1.
- We can specify arguments with or without a label. Arguments without labels can be given in any order.
- Use parentheses to call a function, even if there are no arguments.
- If function name is used without parentheses, R will display its definition.

```
> seq  
function (...)  
  UseMethod("seq")  
<bytecode: 0x623dc18>  
<environment: namespace:base>
```

Variables

- The result of an expression can be assigned it to a named variable with the `<-` or `=` operators.

```
> x <- 2
> y <- x + 3
> x
[1] 2
> y
[1] 5
> x <- x + 1
> x
[1] 3
> y
[1] 5
```

- Variable names can be composed from letters, digits, `'_'`, and `'.'`.
- Some configurations are illegal; e.g. a `'.'` followed by a digit.
- Some names such as `c`, `q`, `t`, and `median` are already defined. R will not prevent them from being reused, but your code might become confusing.

```
> q <- 1
> q
[1] 1
> q()
Save workspace image to ~/.RData? [y/n/c]:
```

Vectors

- A vector is an one-dimensional array of scalars.
- Three basic functions for constructing vectors are
 1. `c(...)` (combine),
 2. `seq(from, to, by)` (sequence),
 3. `rep(x, times)` (repeat).

```
> print(x <- seq(1, 20, by = 2))
[1] 1 3 5 7 9 11 13 15 17 19

> print(y <- rep(3, 4))
[1] 3 3 3 3

> print(z <- c(y, x))
[1] 3 3 3 3 1 3 5 7 9 11 13 15 17 19
```

- The `:` operator takes precedence over operators such as `+` and `-`. Use parentheses when in doubt, or to. improve readability

```
> n <- 10
> print(x <- 1:n+1)
[1] 2 3 4 5 6 7 8 9 10 11
> print(x <- 1:(n+1))
[1] 1 2 3 4 5 6 7 8 9 10 11
```

Indexing into Vectors

```
> x <- seq(10,20)
> x[9]
[1] 18

> x[12]           # Past the end
[1] NA

> x[2:5]
[1] 11 12 13 14

> ind <- c(1,3,5)
> x[ind]
[1] 10 12 14

> x[-1]
[1] 11 12 13 14 15 16 17 18 19 20

> x[-c(1,3,5,7)]
[1] 11 13 15 17 18 19 20

> x[x > 11]
[1] 12 13 14 15 16 17 18 19 20

> ind <- which(x > 11)
> ind
[1] 3 4 5 6 7 8 9 10 11
> x[ind]
[1] 12 13 14 15 16 17 18 19 20
```

Arithmetic on Vectors

```
> x <- c(1, 2, 3)
> y <- c(4, 5, 6)

> x + y           # Most operations are elementwise.
[1] 5 7 9

> y - x
[1] 3 3 3

> x * y
[1] 4 10 18

> y / x
[1] 4.0 2.5 2.0

> y^x
[1] 4 25 216

> 1:4 + 1:2       # Unequal length: shorter vector is repeated.
[1] 2 4 4 6

> 1 + 1:10        # Scalar plus vector
[1] 2 3 4 5 6 7 8 9 10 11

> c(1,2,3) + c(1,2)
[1] 2 4 4
Warning message:
In c(1, 2, 3) + c(1, 2) :
  longer object length is not a multiple of shorter object length
```

Some Useful Functions on Vectors

```
> x <- c(5,4,3,6,7,8,9,10,1,2)
> sort(x)
[1] 1 2 3 4 5 6 7 8 9 10

> x <- c(1,2,3,4,5,6,7,8,9,10)
> rev(x)
[1] 10 9 8 7 6 5 4 3 2 1

> x <- seq(1,10)
> mean(x)
[1] 5.5
> sum(x)
[1] 55
> prod(x)
[1] 3628800
> y <- sample(x)
> min(y)
[1] 1

> # Draw a sample (with or without replacement) from x at random
> x <- seq(1,10)
> sample(x)
[1] 5 7 1 9 10 4 8 6 3 2
> sample(x, size = 5)
[1] 3 1 6 8 9
> sample(x, replace = TRUE)
[1] 9 3 2 1 3 8 10 7 5 7
> sample(x, replace = TRUE, size = 5)
[1] 10 4 6 5 2
```

Other scalar types

- Complex numbers

```
> a <- complex(5, 3); b <- complex(7, 2)
> a + b
[1] 12+5i
```

- Logicals

```
> x <- c(TRUE, TRUE, FALSE); y <- c(TRUE, FALSE, FALSE)
> x & y
[1] TRUE FALSE FALSE
> !x
[1] FALSE FALSE TRUE
> c(sum(x), prod(x))
[1] 2 0
```

- Strings

```
> a <- "my string"
> length(a)
[1] 1
> nchar(a)
[1] 9

> x <- 5
> cat("The answer is", x, ".\n")
The answer is 5 .

> str <- sprintf("The answer is %d.", x)
> print(str)
[1] "The answer is 5."
```

Other scalar types

- Factors represent categorical data. They give descriptive labels, and also emphasize that values are not numbers.

```
> x <- sample(1:3, size = 10, replace = TRUE)
> x
[1] 3 1 3 1 1 3 3 2 2 2
> y <- factor(x, labels=c("urban", "suburban", "rural"))
> y
[1] rural      urban      rural      urban      urban      rural      rural
[8] suburban   suburban   suburban
Levels: urban suburban rural
> levels(y)
[1] "urban"    "suburban" "rural"
```

- Splitting numerical data into intervals, using the `cut` function.

```
> x <- rnorm(10)
> x
[1] 0.3866017 -0.5058934 -2.2927512 -0.7278835  2.6448711
[6] -0.3614214  1.2178368  0.6046139  0.2895143 -0.9899157
> cut(x, 3)
[1] (-0.649,1]  (-0.649,1]  (-2.3,-0.649]  (-2.3,-0.649]
[5] (1,2.65]    (-0.649,1]  (1,2.65]       (-0.649,1]
[9] (-0.649,1]  (-2.3,-0.649]
Levels: (-2.3,-0.649]  (-0.649,1]  (1,2.65]
```

Special scalar types

- Some scalar values with special meanings.
 - ▶ NA means that no value has been assigned, or the value is missing
 - ▶ NaN means *not a number*
 - ▶ NULL represents something which is undefined
 - ▶ Inf represents infinity
- Functions to check for these states.

```
> y <- log(0)
> y
[1] -Inf
> is.infinite(y)
[1] TRUE
> is.na(y)
[1] FALSE

> x <- log(-1)
> is.na(x)
[1] TRUE

> z <- c(1, 2, 3, NA, 4)
> is.na(z)
[1] FALSE FALSE FALSE  TRUE FALSE
> mean(z)
[1] NA
> mean(z, na.rm = TRUE)
[1] 2.5
```

Rounding error

- To avoid issues due to rounding error, use `all.equal` to test for numerical equality.

```
> 2 * 2 == 4  
[1] TRUE  
  
> sqrt(2) * sqrt(2) == 2  
[1] FALSE  
  
> all.equal(sqrt(2) * sqrt(2), 2)  
[1] TRUE
```

Matrices

```
> matrix(NA, nrow = 3, ncol = 3)
   [,1] [,2] [,3]
[1,]    NA    NA    NA
[2,]    NA    NA    NA
[3,]    NA    NA    NA

> matrix(c(1,2,3,4), nrow = 2, ncol = 2)
   [,1] [,2]
[1,]    1    3
[2,]    2    4

> matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE)
   [,1] [,2]
[1,]    1    2
[2,]    3    4

> A <- diag(5, 3)           # A diagonal matrix with custom labels
> rownames(A) <- c("First Row", "Second Row", "Third Row")
> colnames(A) <- c("A", "B", "C")
> print(A)
      A     B     C
First Row 5     0     0
Second Row 0     5     0
Third Row  0     0     5
> nrow(A)
[1] 3
> ncol(A)
[1] 3
```

Indexing Matrices

```
> A <- matrix(seq(1,9), nrow = 3, ncol = 3)
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> A[2,3]      # Access some specific elements
[1] 8
> A[3,2]
[1] 6

> A[1,]        # First row
[1] 1 4 7

> A[,2]        # Second column
[1] 4 5 6

> A[c(1,2),]   # Rows 1 and 2
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8

> A[1,c(2,3)] # Columns 2 and 3 of row 1
[1] 4 7
```

Matrix Operations

```
> A <- B <- matrix(seq(1,9), nrow = 3, ncol = 3)
> A + B
 [,1] [,2] [,3]
[1,]    2    8   14
[2,]    4   10   16
[3,]    6   12   18

> 3 * A          # Multiply by a scalar
 [,1] [,2] [,3]
[1,]    3   12   21
[2,]    6   15   24
[3,]    9   18   27

> A              # Multiply two matrices
 [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
> B
 [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
> A %*% B
 [,1] [,2] [,3] [,4]
[1,]   15   15   15   15
[2,]   18   18   18   18
[3,]   21   21   21   21
[4,]   24   24   24   24
```

Matrix Operations

```
> A <- matrix(c(1,2,3), nrow = 3, ncol = 3)
> A
 [,1] [,2] [,3]
[1,]    1    1    1
[2,]    2    2    2
[3,]    3    3    3
> t(A)          # Transpose
 [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    2    3
[3,]    1    2    3

> A <- matrix(c(1,0,0,0,2,0,0,0,3), nrow = 3, ncol = 3)
> A
 [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    2    0
[3,]    0    0    3
> solve(A)       # Inverse
 [,1] [,2]      [,3]
[1,]    1  0.0  0.0000000
[2,]    0  0.5  0.0000000
[3,]    0  0.0  0.3333333

> x <- matrix(c(1,2,3), 1, 3)
> y <- matrix(c(1,0,0), 1, 3)
> x %x% y        # Kronecker product
 [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,]    1    0    0    2    0    0    3    0    0
```

Matrix Operations

```
> A <- matrix(c(1,0,0,0,2,0,0,0,3), nrow = 3, ncol = 3)
> eigen(A)      # Eigen value/vector decomposition
$values
[1] 3 2 1

$vectors
 [,1] [,2] [,3]
[1,]    0    0    1
[2,]    0    1    0
[3,]    1    0    0

> svd(A)        # Singular value decomposition
$d
[1] 3 2 1

$u
 [,1] [,2] [,3]
[1,]    0    0    1
[2,]    0    1    0
[3,]    1    0    0

$v
 [,1] [,2] [,3]
[1,]    0    0    1
[2,]    0    1    0
[3,]    1    0    0
```

These functions return Lists, which we will discuss shortly.

Stacking Matrices

```
> A <- matrix(0, nrow = 3, ncol = 3)
> B <- matrix(1, nrow = 3, ncol = 3)

> rbind(A, B)
 [,1] [,2] [,3]
[1,] 0 0 0
[2,] 0 0 0
[3,] 0 0 0
[4,] 1 1 1
[5,] 1 1 1
[6,] 1 1 1

> cbind(A, B)
 [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 0 0 0 1 1 1
[2,] 0 0 0 1 1 1
[3,] 0 0 0 1 1 1
```

Functions to Operate on matrices

```
> A <- matrix(seq(1,9), nrow = 3, ncol = 3)
> sqrt(A)           # sqrt works elementwise on matrix
      [,1]      [,2]      [,3]
[1,] 1.000000 2.000000 2.645751
[2,] 1.414214 2.236068 2.828427
[3,] 1.732051 2.449490 3.000000

# Note: see the chol function to compute B such that B %*% B = A

> A <- matrix(seq(1,9), nrow = 3, ncol = 3)
> A
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> sum(A)          # sum and max treat the matrix like a vector
[1] 45
> max(A)
[1] 9

> apply(A, 1, max) # Apply max to each row of A
[1] 7 8 9
> apply(A, 2, max) # Apply to columns
[1] 3 6 9
```

Arrays

Arrays are similar to matrices, but can have > 2 dimensions.

```
> A <- array(seq(1,27), dim = c  
  (3,3,3))  
> A  
, , 1  
 [,1] [,2] [,3]  
[1,] 1 4 7  
[2,] 2 5 8  
[3,] 3 6 9  
, , 2  
 [,1] [,2] [,3]  
[1,] 10 13 16  
[2,] 11 14 17  
[3,] 12 15 18  
, , 3  
 [,1] [,2] [,3]  
[1,] 19 22 25  
[2,] 20 23 26  
[3,] 21 24 27  
  
> A[,2,1] # 2nd col from 1st slice  
[1] 4 5 6  
  
> A[, , 1] # 1st slice  
 [,1] [,2] [,3]  
[1,] 1 4 7  
[2,] 2 5 8  
[3,] 3 6 9  
  
> A[, , 1:2] # 1st and 2nd slice  
, , 1  
 [,1] [,2] [,3]  
[1,] 1 4 7  
[2,] 2 5 8  
[3,] 3 6 9  
  
, , 2  
 [,1] [,2] [,3]  
[1,] 10 13 16  
[2,] 11 14 17  
[3,] 12 15 18  
  
> apply(A, 3, sum)  
[1] 45 126 207
```

Lists

- Vectors, matrices, and arrays are intended for scalars, usually numbers.
- Lists can hold elements of varying types.
- Lists can be used to construct more complicated data structures.

```
> x <- list(a = "string value", b = 5.5, c = TRUE)
> print(x)
$a
[1] "string value"

$b
[1] 5.5

$c
[1] TRUE

> x$a
[1] "string value"
> x[["a"]]
[1] "string value"
> x[[1]]
[1] "string value"
```

- Lists are used to return multiple results from a function (e.g. eigen).

Data Frames

- Data frames are convenient for data in table format. Each column represents a variable, and can have its own data type.
- The CO2 dataset comes with R. First, check the help page using ?CO2.

```
CO2           package:datasets          R
Documentation

Carbon Dioxide uptake in grass plants

Description:

The 'CO2' data frame has 84 rows and 5 columns of data from an
experiment on the cold tolerance of the grass species
_Echinochloa crus-galli_.

...
```

- Now let's view the dataset

```
> CO2
   Plant      Type Treatment conc uptake
1   Qn1      Quebec nonchilled  95   16.0
2   Qn1      Quebec nonchilled 175   30.4
3   Qn1      Quebec nonchilled 250   34.8
...
83  Mc3 Mississippi     chilled  675   18.9
84  Mc3 Mississippi     chilled 1000   19.9
```

Data Frames

- Access each field as a vector using \$ operator.

```
> CO2$uptake  
[1] 16.0 30.4 34.8 37.2 35.3 39.2 39.7 13.6 27.3 37.1 41.8  
...  
[76] 13.7 14.4 10.6 18.0 17.9 17.9 17.9 18.9 19.9
```

- Access rows or columns using matrix indexing.

```
> CO2[3,]  
  Plant    Type Treatment conc uptake  
3  Qn1 Quebec nonchilled  250    34.8  
> CO2[,4]  
[1] 95 175 250 350 500 675 1000 95 175 250 350  
...  
[76] 675 1000 95 175 250 350 500 675 1000  
> CO2[3,4]  
[1] 250
```

- Create a new data.frame

```
> data.frame(new.conc = CO2$conc, new.uptake = CO2$uptake)  
new.conc new.uptake  
1        95      16.0  
2       175      30.4  
3       250      34.8  
4       350      37.2  
5       500      35.3
```

Data Frames

We can sort a data.frame by one or more columns using order

```
> idx <- order(CO2$uptake)
> CO2[idx,]
  Plant      Type Treatment conc uptake
71  Mc2 Mississippi    chilled   95    7.7
29  Qc2     Quebec    chilled   95    9.3
64  Mc1 Mississippi    chilled   95   10.5
43  Mn1 Mississippi nonchilled   95   10.6
...
19  Qn3     Quebec nonchilled  500   42.9
20  Qn3     Quebec nonchilled  675   43.9
14  Qn2     Quebec nonchilled 1000   44.3
21  Qn3     Quebec nonchilled 1000   45.5
```

Data Frames

- We can use `merge` to do an “inner join” of `data.frames`, like SQL.

```
> data1
  id  color
1  1    red
2  2  green
3  3   blue
4  4 yellow

> data2
  id      shape letter
1  2    square     a
2  3 triangle     b
3  4   circle     c
4  5      star     d

> merge(data1, data2, by="id")
  id  color      shape letter
1  2  green    square     a
2  3  blue triangle     b
3  4 yellow   circle     c
```

- For more examples of manipulating `data.frames`, see www.statmethods.net/management.

Functions

- Functions are objects and can be passed like any other data type in R.

```
> f <- function(x) { x + 1 }
> f(1)
[1] 2
> g <- function(f, x) { f(x)^2 }
> g(f,1)
[1] 4
```

- The argument types and return type are not declared in advance.
- We can also explicitly specify what to return.

```
f <- function(x) {
  if (x %% 2 == 0)
    return("even")
  else
    return("odd")
}
```

Listing 1: Script file: function.R

- Running this script produces the following output.

```
> source("function.R")
> print(f(1))
[1] "odd"
> print(f(2))
[1] "even"
```

Functions

- Can set default values for arguments.

```
> f <- function(x,y,z = 100) { c(x, y, z) }
> f(y = 1, x = 3)
[1] 3 1 100
```

- Argument types can be checked manually.

```
> f <- function(x) { stopifnot(is.numeric(x)); return(x^2) }
> f(1)
[1] 0
> f("a")
Error: is.numeric(x) is not TRUE
```

- Functions can be constructed around variables that are defined outside. This can be a source of confusion.

```
> y <- 5
> f <- function(x) { x + y }
> f(1)
[1] 6
```

Control Flow

- if statement

```
u <- runif(1)
if (u < 0.25) {
  y <- 0
} else if (u < 0.5) {
  y <- 1
} else {
  y <- 2
}
```

- switch statement

```
x <- "1"
switch(x,
  "1" = "blue",
  "2" = "green",
  "3" = "red",
  "default")
```

- for loop

```
for (i in 1:10) { cat("Iteration", i, "\n") }
```

- while loop

```
u <- 1
while (u > 0.1) {
  u <- runif(1)
}
```

Control Flow

- try/catch statement

```
f <- function(x) { stop("Error!!") }
ret <- tryCatch(f(5),
  error = function(e) {
    return(e)
  }
)
## ret is the message e

g <- function(x) { x }
ret <- tryCatch(g(5),
  error = function(e) {
    return(e)
  }
)
## ret is 5
```

Control Flow

- apply statement

```
> X <- matrix(rnorm(1000*5), 1000, 5)
> apply(X, 2, sd)
[1] 0.9932699 0.9643422 0.9976412 0.9784432 1.0072679
```

- sapply statement

```
f <- function(x) { if (x %% 2 == 0) { "even" } else { "odd" } }
x <- rpois(8, 5)
sapply(x, f)

> sapply(x, f)
[1] "odd"   "odd"   "odd"   "even"  "odd"   "odd"   "odd"   "even"
```

- lapply statement

```
> L <- list()
> L[[1]] <- rnorm(100)
> L[[2]] <- rnorm(100)
> L[[3]] <- rnorm(100)
> lapply(L, function(x) { c(length(x), mean(x), sd(x)) })
[[1]]
[1] 100.00000000  0.07369905  1.03257221

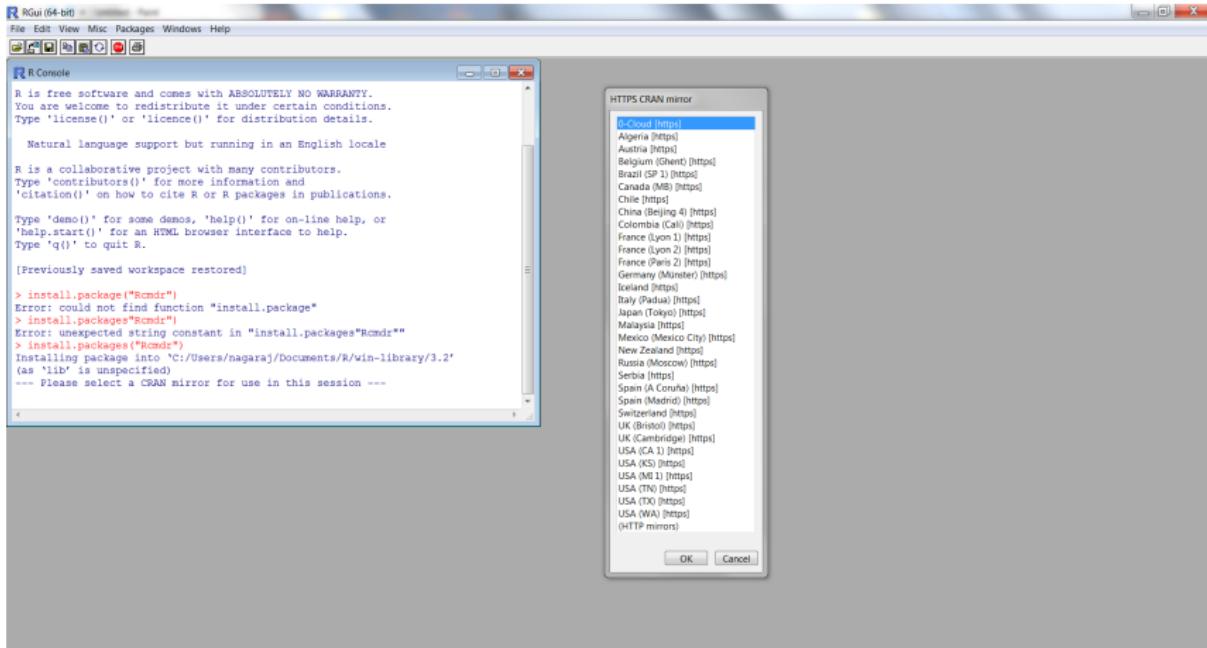
[[2]]
[1] 100.00000000  0.0376098   0.8865248

[[3]]
[1] 100.00000000 -0.1576617   1.0352769
```

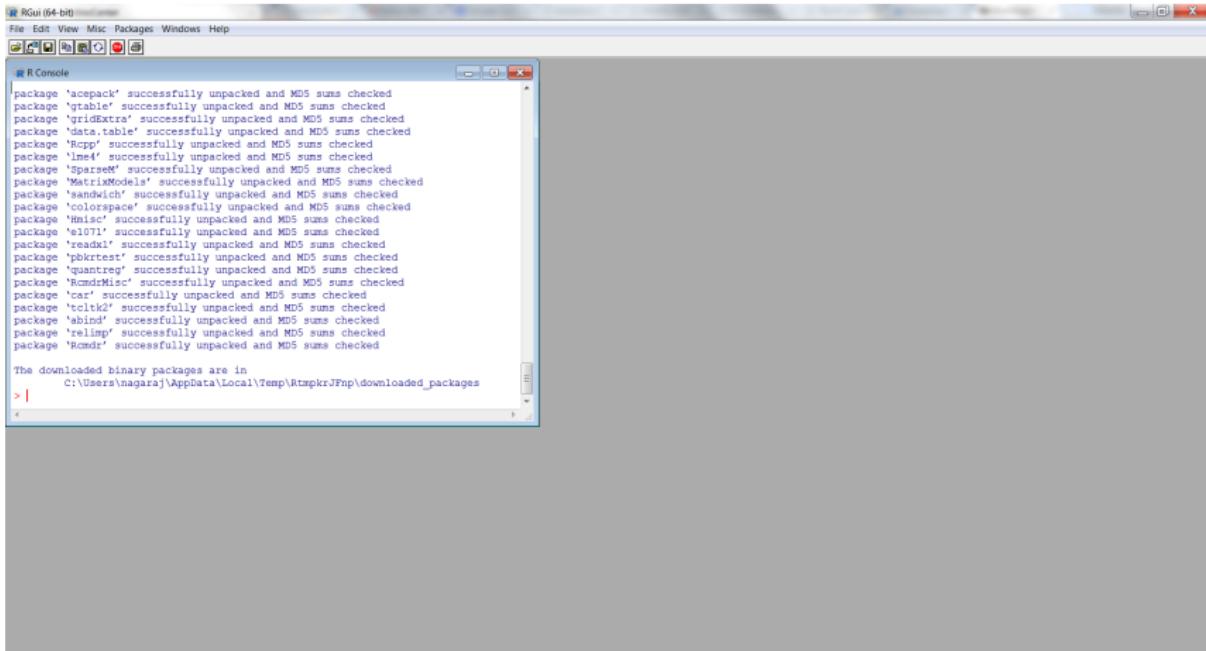

About R Commander

- R commander is a package, to be installed and loaded.
- R commander can be invoked with `Rcmdr()` or `load("Rcmdr")`
- `Rcmdr` opens up a GUI which can be used to
 - ▶ read data file types (.txt, .xlsx, etc)
 - ▶ use graphical tools such as histogram, boxplot, scatterplot matrix, etc
 - ▶ use most commonly used parametric and nonparametric statistical procedures, and implement several linear and generalized linear models
- Many `Rcmdr` Plugin's are also available: `RcmdrPlugin.BCA`, `RcmdrPlugin.DoE`, etc
- Created by John Fox, McMaster University in 2003
- <http://www.rcommander.com/>,
<http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/>

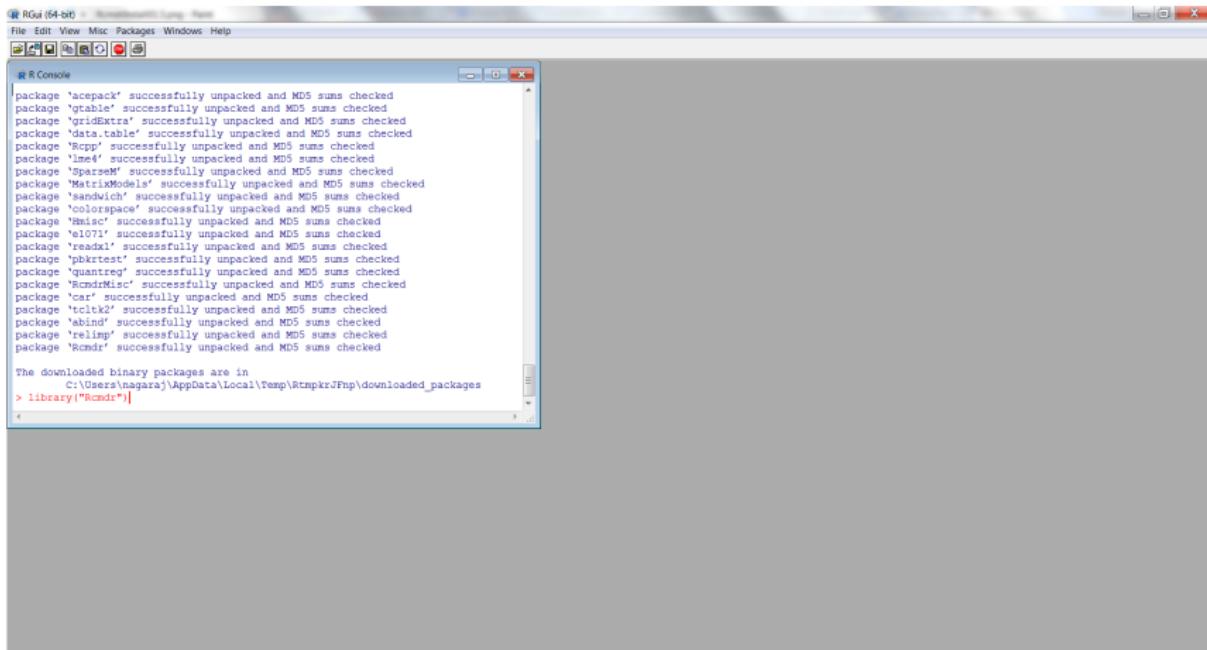
Installing R Commander



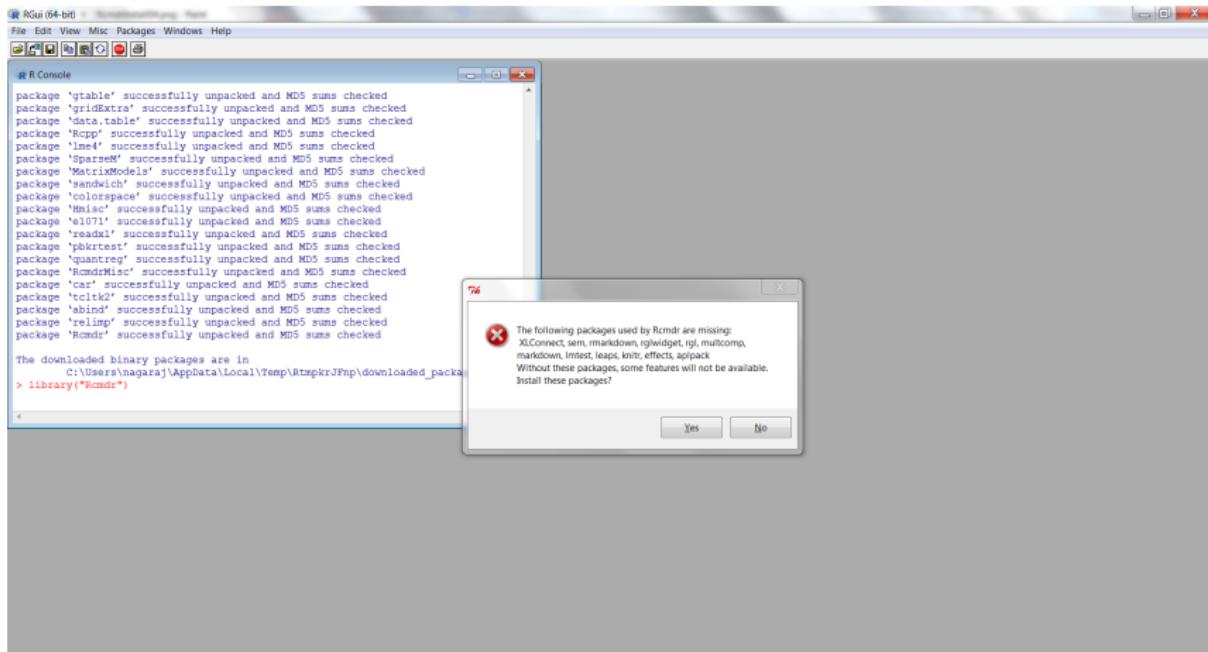
Installing R Commander



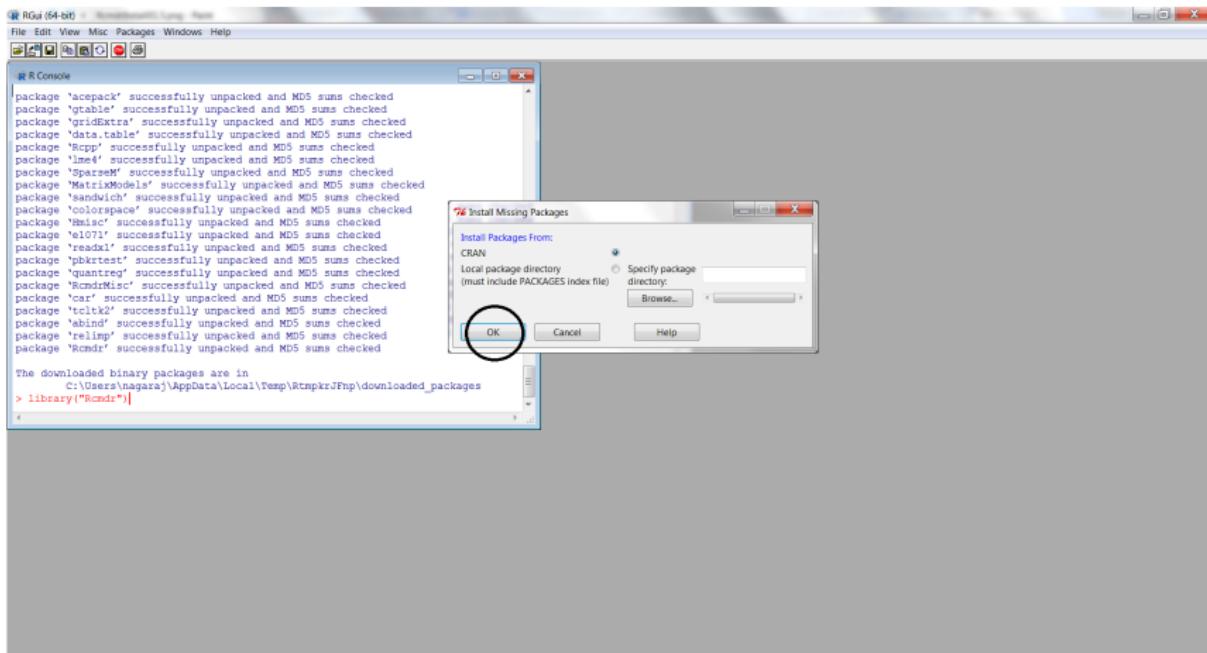
Invoking Rcmdr



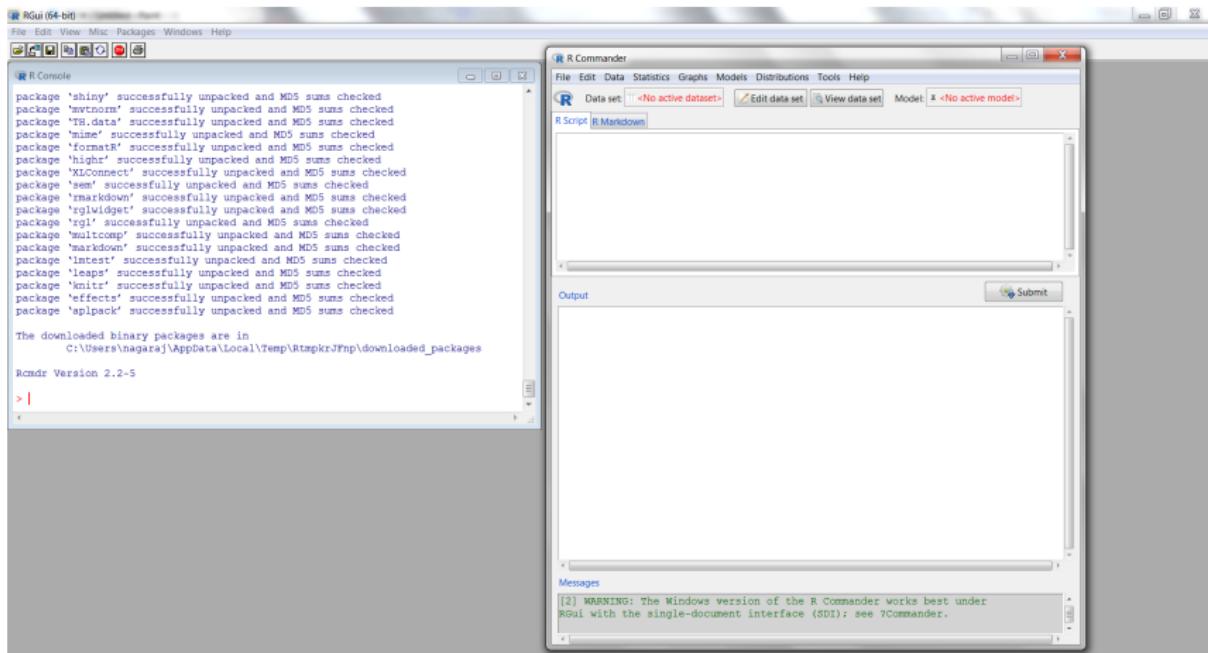
Invoking Rcmdr



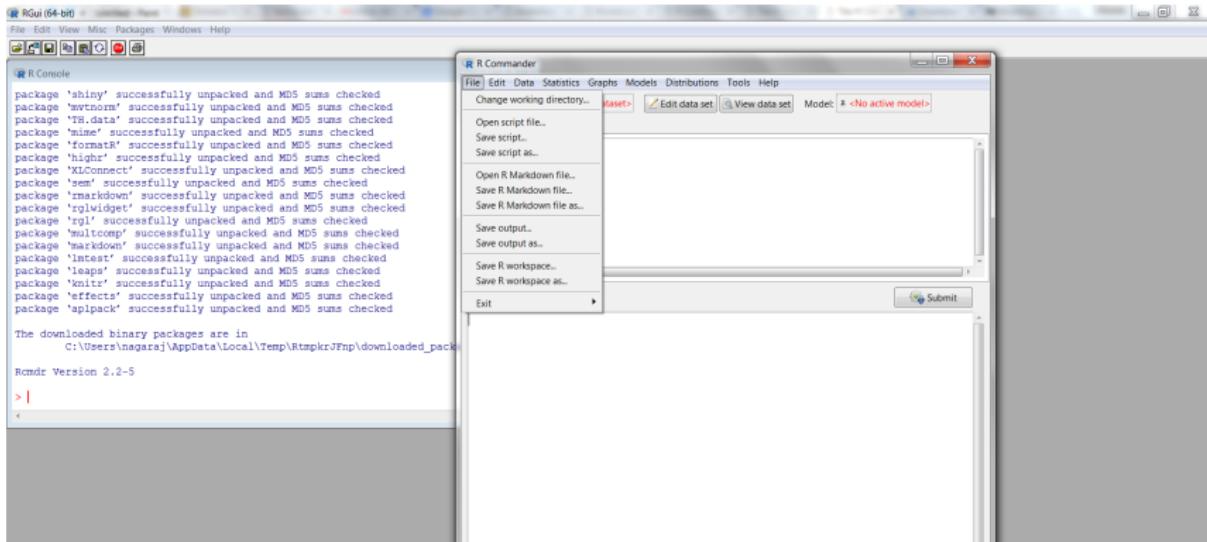
Invoking Rcmdr



Invoking Rcmdr



Menus of Rcmdr

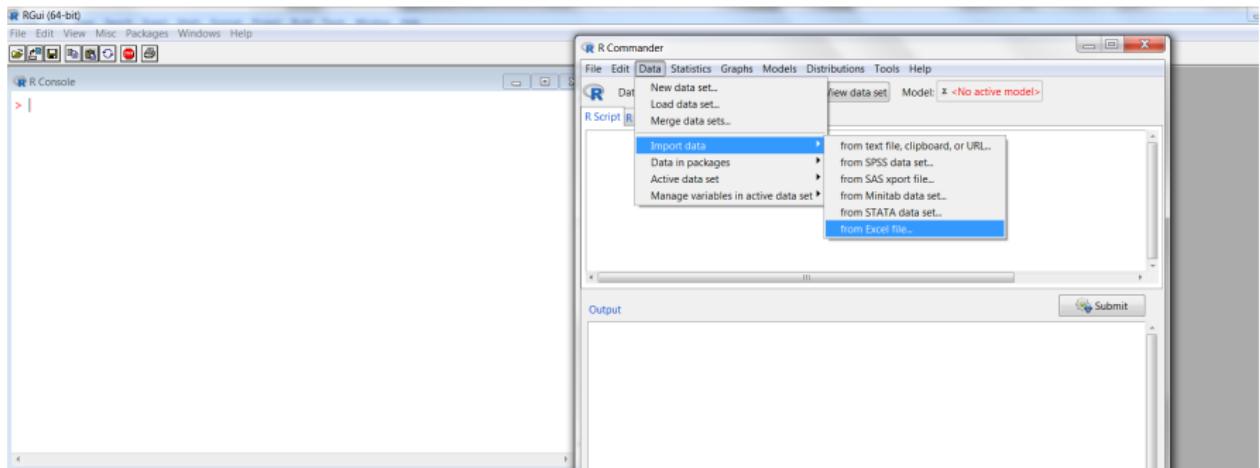


Complete Menu "tree" Rcmdr

Reading Data in Rcmdr: Example

- Data set consists of annual data (2000-2014) on the 'Number of Internet users per 100' for 244 countries.
- Rcmdr has many options for reading datasets. A typical dataset with both character and numeric variables, either .csv or .xlsx type seem to be quite stable.
- Data can also be read from other R packages.
- InternetUsers.xlsx data set consists of 244 rows and 17 rows. First two rows are 'Country Name' and 'Country Code'. The following 15 columns, IntUsers2000-IntUsers2014, are numeric data.

Reading Data in Rcmdr: Example



Reading Data in Rcmdr: Example

The screenshot illustrates the workflow for reading an Excel dataset into R using the Rcmdr interface.

Import Excel Data Set Dialog: This window is open, prompting the user to enter the name of the data set ("Users") and specifying options: "Variable names in first row of spreadsheet" (checked), "Row names in first column of spreadsheet" (unchecked), and "Convert character data to factors" (checked). A "Missing data indicator" field contains a question mark. Buttons include "Help", "OK" (highlighted in green), and "Cancel".

Open File Dialog: This window shows the "Documents library" with "Internet Users.xlsx" selected. The "File name" dropdown also displays "Internet Users.xlsx". Buttons are "Open" (highlighted in blue) and "Cancel".

Select one table Dialog: This modal window lists "Sheet1" and "Sheet2". "Sheet2" is highlighted with a blue selection bar. Buttons are "OK" and "Cancel".

R Commander Main Window: The "Data set" tab is selected, showing the dataset "Users" is currently loaded. The "Script" tab is active. The "R Script" pane contains the R code used to read the file:

```
Users <-  
readXL("C:/Users/nagaraj/Documents/git-remote/R_Thai_Workshop/day1/References/  
rownames=FALSE, header=TRUE, na=".", sheet="Sheet2",  
stringsAsFactors=TRUE)
```

The "Output" pane shows the results of the R command:

```
> Users <-  
> readXL("C:/Users/nagaraj/Documents/git-remote/R_Thai_Workshop/day1/Reference  
> rownames=FALSE, header=TRUE, na=".", sheet="Sheet2",  
> stringsAsFactors=TRUE)
```

The "Messages" pane at the bottom displays two notes:

- [6] NOTE: The dataset Users has 244 rows and 17 columns.
- [7] NOTE: The dataset Users has 244 rows and 17 columns.

Reading Data in Rcmdr: Example

The screenshot shows the Rcmdr interface with two windows open. The main window is titled "R Commander" and contains an "Output" pane displaying R code and its results. The code reads an Excel file named "Users" from the current directory. The results show the first 30 rows of the dataset "Iusers". The second window is titled "Iusers" and is a data viewer showing the same 30 rows of data. The columns include Country, Name, Country.Code, InttUsers2000, InttUsers2000, and Intt. The data shows various countries and their corresponding values.

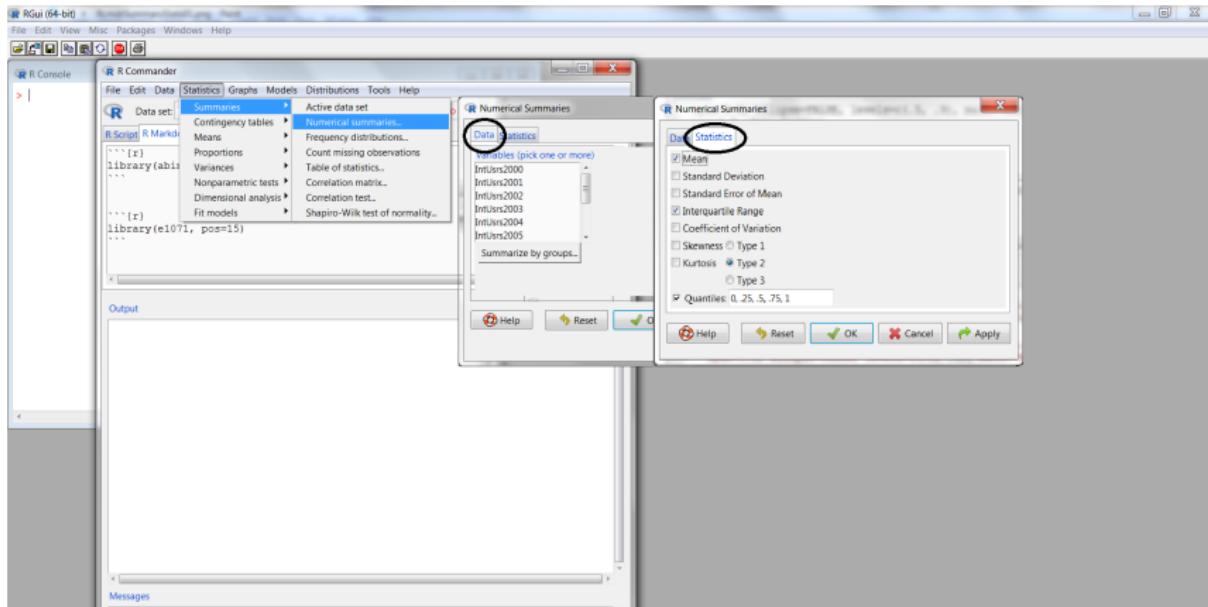
```
Iusers <-  
readXLS("C:/Users/  
rownames=FALSE  
stringsAsFactor  
1 Afghanistan ARG NA 4.722568e-03 4.561395e-03 0.0 *  
2 Albania ALB 0.114097347 3.257984e-01 3.900813e-01 0.9  
3 Algeria DZA 0.491705679 6.461140e-01 1.591641e+00 2.1 *  
4 American Samoa ASM NA NA NA NA NA NA  
5 Andorra ADN 10.53883559 3.08e-01 1.260476e-01 13.5  
6 Aruba ABW 0.105045562 1.360128e-01 2.703767e-01 0.2  
7 Antigua and Barbuda ATG 6.492225737 8.399286e+00 1.250000e+01 17.2  
8 Argentina ARG 7.038482086 9.780807e-01 1.688212e+01 11.8  
9 Armenia ARM 1.300470022 1.631095e+00 1.860405e+00 4.5  
10 Aruba ABW 15.442822948 1.710000e+01 1.880000e+01 20.8  
11 Australia AUS 46.756115612 5.26927e+01 NA  
12 Austria AUT 33.730132952 3.918545e+01 3.656000e+01 42.7  
13 Azerbaijan AZE 0.147757576 3.055646e-01 4.999714e+00  
14 Bahamas, The BHS 8.000000000 1.180000e+01 1.800000e+01 20.0  
15 Bahrain BHR 6.153732545 1.503863e+01 1.805072e+01 21.5  
16 Bangladesh BDG 0.071039423 1.290080e-01 1.399203e+01 0.1  
17 Belarus Belarus BYL 1.193645e+01 1.193645e+01 1.193645e+01 39.6  
18 Belize BEL 1.860398126 4.300616e+00 8.350971e+00  
19 Belgium BLR 29.431691692 3.128840e+01 4.632000e+01 49.8  
20 Benin BEN 5.963835303 NA 5.684251e+00  
21 Benin BEN 0.225247851 3.63179e-01 7.029456e-01 0.9  
22 Bermuda BDN 42.949860015 4.750970e+01 5.203160e+01 56.5  
23 Bhutan BTN 0.400944447 8.646286e-01 1.675803e+00 2.4  
24 Bolivia BOL 1.442763589 2.120443e+00 3.117193e+00 3.5  
25 Bosnia and Herzegovina BIH 1.082960759 1.200527e+00 2.648268e+00 3.9  
26 Botswana BWA 2.902666622 3.430887e+00 3.385592e+00 3.3  
27 Brazil BRA 2.870865159 4.528495e+00 9.149425e+00 13.2  
28 Brunei Darussalam BRN 8.996284534 1.291777e+01 1.532980e+01 19.5  
29 Bulgaria BGR 5.370923469 7.612299e+00 9.080000e+00 12.0  
30 Burkina Faso BFR 0.077080169 1.577325e-01 2.009926e-01 0.3 =
```

[6] NOTE: The dataset Iusers has 244 rows and 17 columns.
[7] NOTE: The dataset Iusers has 244 rows and 17 columns.

Descriptive Statistics in Rcmdr: Example

- Summary statistics of the uploaded data can be obtained using the menu 'Statistics'
- Statistics > Summaries can access computation of summary statistics for both quantitative as well as qualitative variables
- Summary can be computed for the whole data or by groups (Grouping determined by a 'factor' variable in the data set)
- Results are displayed in the output window of Rcmdr

Descriptive Statistics in Rcmdr: Example



Descriptive Statistics in Rcmdr: Example

The screenshot shows the R Commander interface with the 'users' dataset selected. The 'Data' menu is open, and the 'Statistics' option is highlighted. The 'Output' pane displays the results of the 'numSummary' function applied to the 'users' dataset, grouped by year from 2008 to 2014. The results include mean, IQR, and quantiles for each year. The 'Messages' pane at the bottom shows two error messages: 'ERROR: No factors selected.' and 'ERROR: You must select a variable.'

```
R Commander
File Edit Data Statistics Graphic Modes Distributions Tools Help
R Data set: users Edit data set View data set Model: <No active model>
R Script R Markdown
```
IntUsrs2008, "IntUsrs2009", "IntUsrs2010", "IntUsrs2011", "IntUsrs2012",
"IntUsrs2013", "IntUsrs2014")]
, statistics=c("mean", "IQR", "quantiles"),
quantiles=c(0,.25,.5,.75,1)
```

```{r}
numSummary(Iusers[,c("IntUsrs2000", "IntUsrs2001", "IntUsrs2002", "IntUsrs2003", "IntUsrs2004",
"IntUsrs2005", "IntUsrs2006", "IntUsrs2007", "IntUsrs2008", "IntUsrs2009", "IntUsrs2010",
"IntUsrs2011", "IntUsrs2012", "IntUsrs2013", "IntUsrs2014")], statistics=c("mean", "IQR",
"quantiles"), quantiles=c(0,.25,.5,.75,1))
```

Output
Generate report
```
numSummary(Iusers[,c("IntUsrs2000", "IntUsrs2001", "IntUsrs2002", "IntUsrs2003", "IntUsrs2004",
"IntUsrs2005", "IntUsrs2006", "IntUsrs2007", "IntUsrs2008", "IntUsrs2009", "IntUsrs2010",
"IntUsrs2011", "IntUsrs2012", "IntUsrs2013", "IntUsrs2014")], statistics=c("mean", "IQR",
"quantiles"), quantiles=c(0,.25,.5,.75,1))
 mean IQR 0% 25% 50% 75% 100% n NA
IntUsrs2000 8.521371 0.55247 0.0 0.4794528 2.320831 9.031923 52.00000 224 20
IntUsrs2001 10.501471 12.56598 0.0 0.6410089 2.944368 13.206986 64.00000 225 19
IntUsrs2002 13.437392 17.52950 0.0 1.0829414 4.626175 18.612446 79.12000 227 17
IntUsrs2003 15.976100 21.28329 0.0 1.6864900 6.725435 22.969784 83.14000 221 23
IntUsrs2004 18.386798 25.07230 0.0 2.3954570 8.697180 27.467757 83.89000 224 20
IntUsrs2005 20.505668 29.91544 0.0 3.1420618 11.415329 33.057500 87.00000 226 18
IntUsrs2006 23.011282 33.58144 0.0 4.2899330 14.182028 38.980000 88.31000 228 19
IntUsrs2007 25.106282 35.58144 0.0 4.1820618 13.383334 40.415000 91.00000 232 12
IntUsrs2008 27.006216 38.20000 0.0 6.2150000 21.000000 44.4125000 91.00000 231 13
IntUsrs2009 30.673224 43.27652 0.0 7.3500000 24.700000 50.626521 93.00000 230 14
IntUsrs2010 33.757442 43.49000 0.0 10.0000000 29.030000 53.490000 93.39000 231 13
IntUsrs2011 36.726756 45.56992 0.0 11.5000000 33.985000 57.045919 94.81969 232 12
IntUsrs2012 39.818382 48.36237 0.0 13.4761500 36.970000 61.838520 96.20980 230 14
IntUsrs2013 42.448618 50.82190 0.9 15.3500000 41.000000 66.171896 96.54680 231 13
IntUsrs2014 44.897096 51.56500 0.0 17.7350000 43.550000 69.300000 98.16000 231 13
```

Messages
[9] ERROR: No factors selected.
[10] ERROR: You must select a variable.
```

Descriptive Statistics in Rcmdr: Example

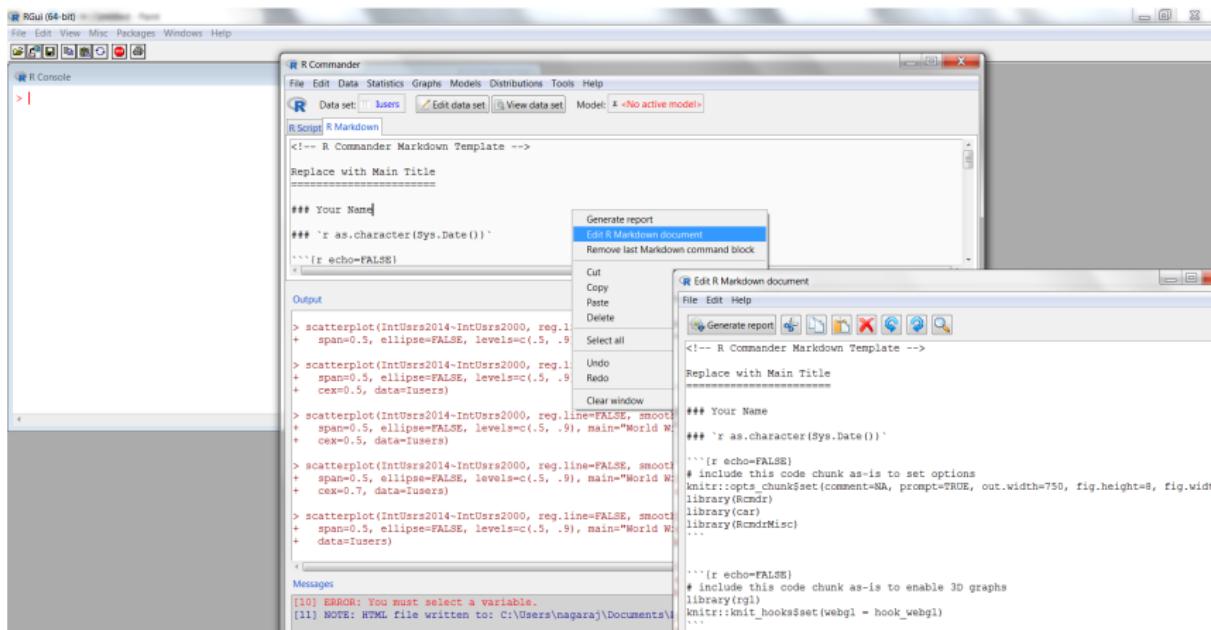
- Understanding the output:

| parameter | What is it? |
|-----------|--|
| mean | Measure of central tendency |
| sd | Standard deviation - a measure of variability in the data |
| N | Number of readings |
| NA | Number of missing values |
| 0% | Minimum value |
| 25% | The value below which 25 percent of the observations may be found. |
| 50% | The value below which 50 percent of the observations may be found. |
| 75% | The value below which 75 percent of the observations may be found. |
| 100% | Maximum value |

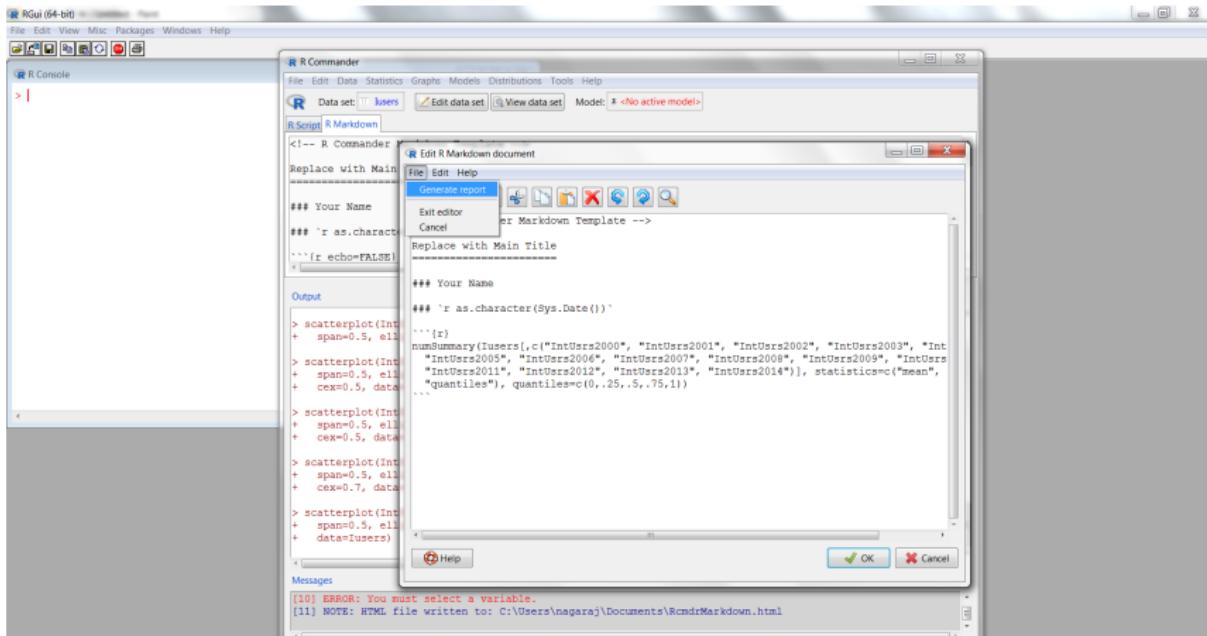
Descriptive Statistics in Rcmdr: Example

- Interpretation of the output:
 - ▶ Mean is much higher than the median in the early years due to the highly skewed nature of this data.
 - ▶ Over the years, they are becoming closer as the Internet usage is becoming ubiquitous.
 - ▶ Even in 2014, 50% of the countries of the world have less than 43 Internet users per 100.
 - ▶ Lower percentiles are climbing at a much slower rate than the higher percentiles.
- The results may be cut-and-paste to reports
- R Markdown can also be used for creating reports

Creating Reports



Creating Reports



Creating Reports

The screenshot shows a web browser window with multiple tabs open. The tabs include "R: Read xls and xlsx files.", "R: Read an Excel File", "R Tutorial on Reading and ...", "Karp-Commander-intro.pdf", and "Replace with Main Title". The main content area displays a report titled "Replace with Main Title" with sections for "Your Name" (left blank), "2016-07-31", and a large code block for "mutSummary".

```
## mutSummary(tusers[,c("Intusr2000", "Intusr2001", "Intusr2002", "Intusr2003", "Intusr2004", "Intusr2005", "Intusr2006", "Intusr2007", "Intusr2008", "Intusr2009", "Intusr2010", "Intusr2011", "Intusr2012", "Intusr2013", "Intusr2014")], statistics=c("mean", "IQR", "quantiles"), quantiles=c(0.25, 0.5, 0.75, 1))
```

| | mean | IQR | 0% | 25% | 50% | 75% | 100% |
|---------------|-----------|----------|-----|------------|-----------|-----------|----------|
| ## Intusr2000 | 8.521371 | 8.55247 | 0.0 | 0.4794528 | 2.320831 | 9.031923 | 52.00000 |
| ## Intusr2001 | 10.501471 | 12.36598 | 0.0 | 0.6410089 | 2.944368 | 13.206986 | 64.00000 |
| ## Intusr2002 | 11.437392 | 17.52930 | 0.0 | 1.0829414 | 4.626175 | 18.612446 | 79.12000 |
| ## Intusr2003 | 15.076100 | 21.28329 | 0.0 | 1.6864900 | 6.725435 | 22.969784 | 83.14000 |
| ## Intusr2004 | 18.386798 | 25.07230 | 0.0 | 2.3954570 | 8.697186 | 27.467757 | 83.89000 |
| ## Intusr2005 | 20.505668 | 29.91544 | 0.0 | 3.1420618 | 11.415329 | 33.057500 | 87.00000 |
| ## Intusr2006 | 23.011282 | 33.69007 | 0.0 | 4.2899330 | 14.182020 | 37.980000 | 89.51000 |
| ## Intusr2007 | 25.199310 | 35.38414 | 0.0 | 4.8142039 | 16.345674 | 40.398347 | 96.60000 |
| ## Intusr2008 | 28.006218 | 38.20000 | 0.0 | 6.2150000 | 21.000000 | 44.415000 | 91.00000 |
| ## Intusr2009 | 30.673226 | 43.27652 | 0.0 | 7.3500000 | 24.700000 | 50.626521 | 93.00000 |
| ## Intusr2010 | 33.726741 | 43.49000 | 0.0 | 10.0000000 | 29.030000 | 53.490000 | 93.19000 |
| ## Intusr2011 | 36.726756 | 45.56992 | 0.0 | 11.5000000 | 33.985000 | 57.069919 | 94.81969 |
| ## Intusr2012 | 39.818382 | 48.36237 | 0.0 | 13.4761500 | 36.970000 | 61.838520 | 96.20980 |
| ## Intusr2013 | 42.448618 | 50.82190 | 0.9 | 15.3500000 | 41.000000 | 66.171896 | 96.54680 |
| ## Intusr2014 | 44.897096 | 51.56500 | 0.0 | 17.7350000 | 43.550000 | 69.300000 | 98.16000 |
| ## | n | NA | | | | | |
| ## Intusr2000 | 224 | 20 | | | | | |
| ## Intusr2001 | 225 | 19 | | | | | |
| ## Intusr2002 | 227 | 17 | | | | | |
| ## Intusr2003 | 221 | 23 | | | | | |
| ## Intusr2004 | 224 | 20 | | | | | |
| ## Intusr2005 | 226 | 18 | | | | | |
| ## Intusr2006 | 225 | 19 | | | | | |

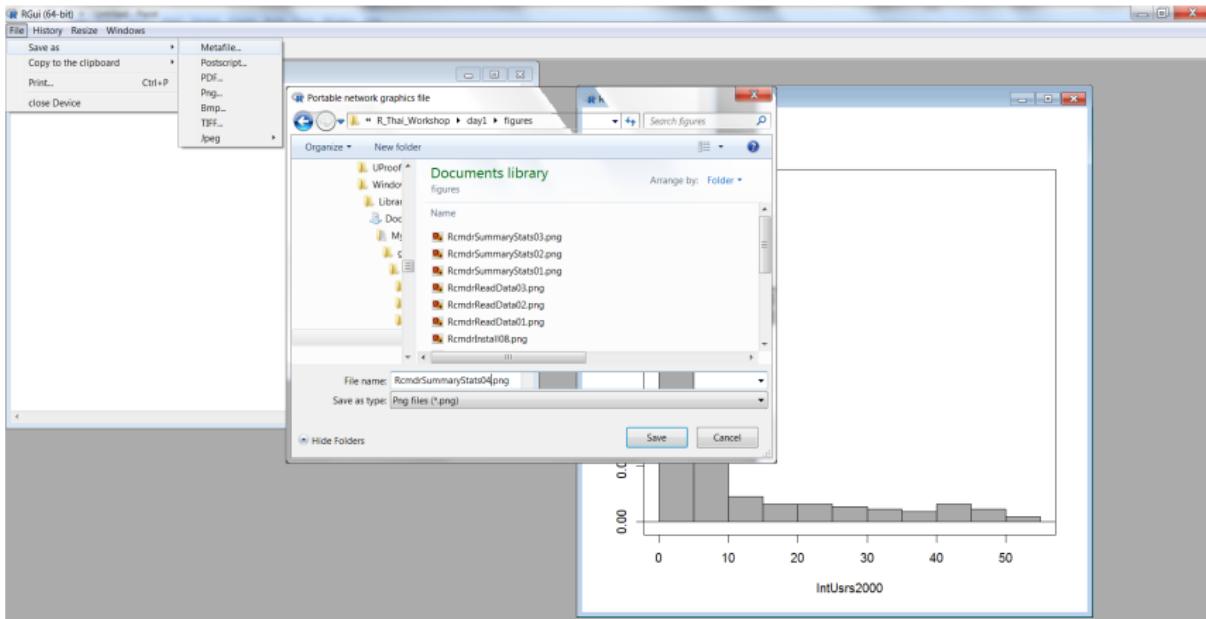
Graphs in Rcmdr

- Rcmdr is also a convenient tool for creating graphical representation of data.
- Next slide shows how to obtain a simple scatterplot.
- There are many choices for [practically] every feature of the plot.
- Try various combinations of choices and determine the best looking graph.

Plotting a Histogram

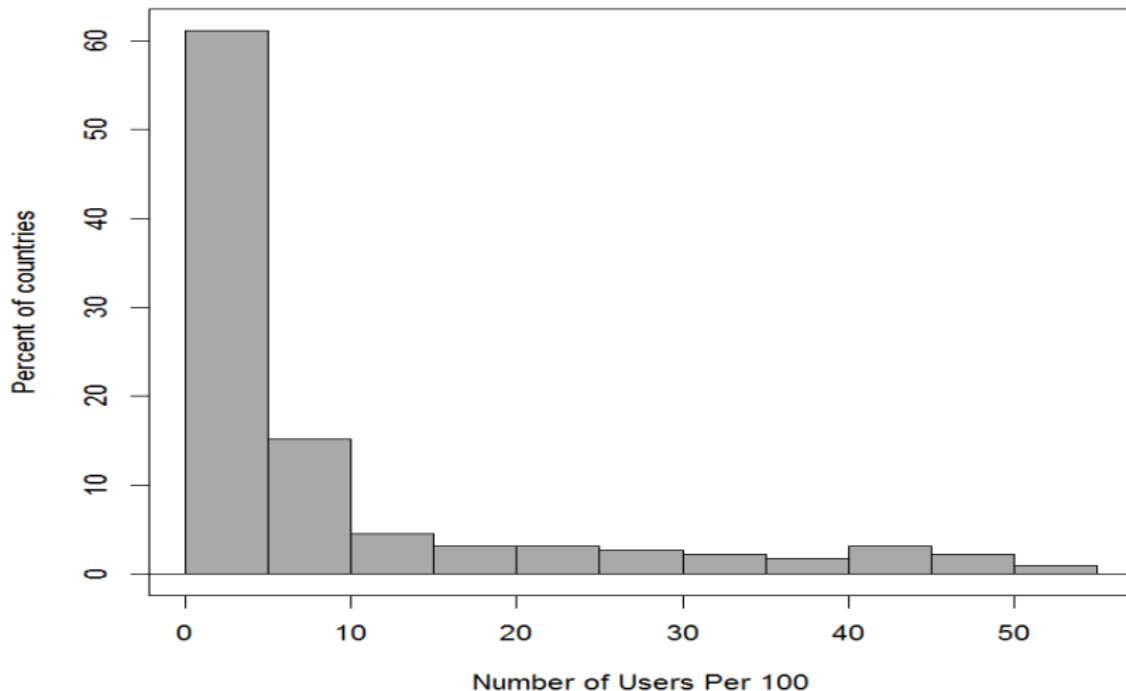
The screenshot shows the RGui interface with the R Commander add-on open. The R Commander window has tabs for Data, Statistics, Graphs, Models, Distributions, Tools, and Help. The Graphs tab is selected. A dropdown menu under Graphs shows various plot types: Index plot..., Dot plot..., Histogram... (which is circled in red), Density estimate..., Stem-and-leaf display..., Boxplot..., Quantile-comparison plot..., Scatterplot..., Scatterplot matrix..., Line graph..., XY conditioning plot..., Plot of means..., Strip chart..., Bar graph..., Pie chart..., 3D graph..., Save graph to file..., and Print... (disabled). Below this is an Output pane displaying R code and its results. The results show data for various years from 2000 to 2014, including IntUsrs2000 through IntUsrs2014. The Data pane of the Histogram dialog box is active, showing variables: IntUsr2000, IntUsr2001, IntUsr2002, IntUsr2003, IntUsr2004, and IntUsr2005. The Options pane shows 'Plot by groups...' checked. The Plot Labels pane shows 'x-axis label' set to 'IntUsr'. The Plot Options pane shows 'Number of bins' set to 'auto'. The Axis Scaling pane shows 'Frequency counts' checked. The Data pane of a second Histogram dialog box is also visible, showing the same variable list and options.

Saving a Plot

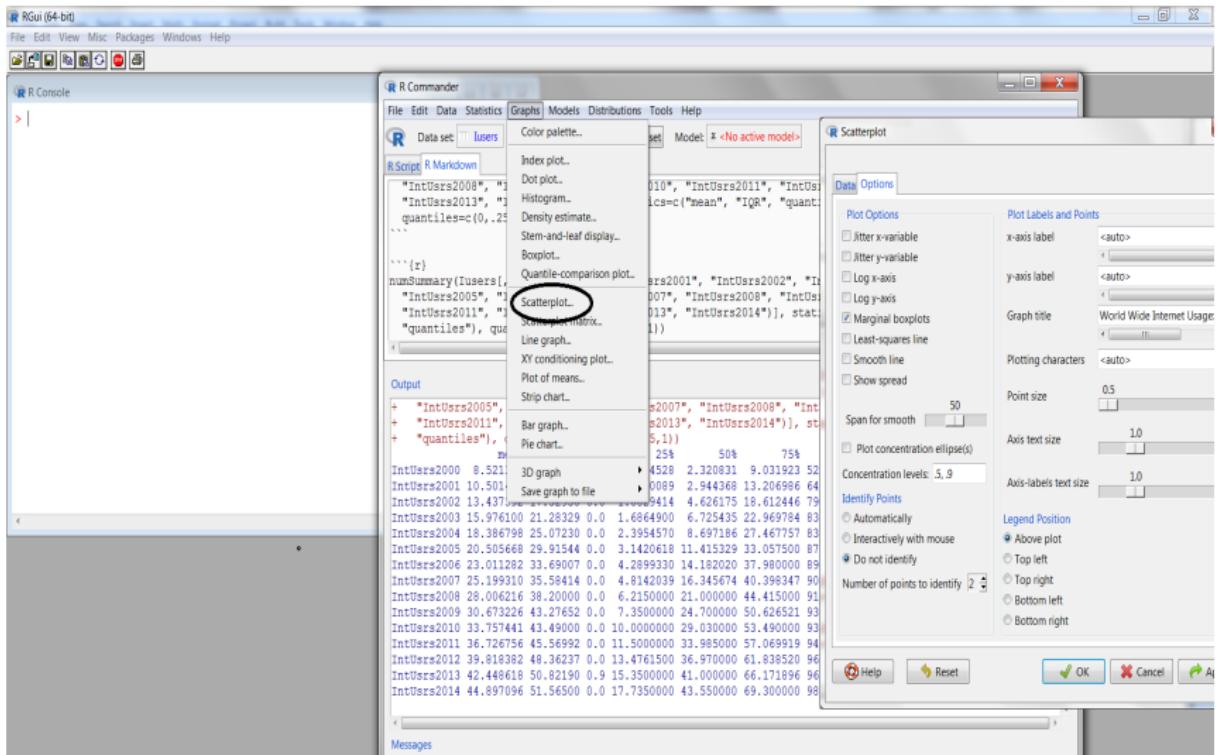


Histogram

Internet Usage in the World (Year 2000)

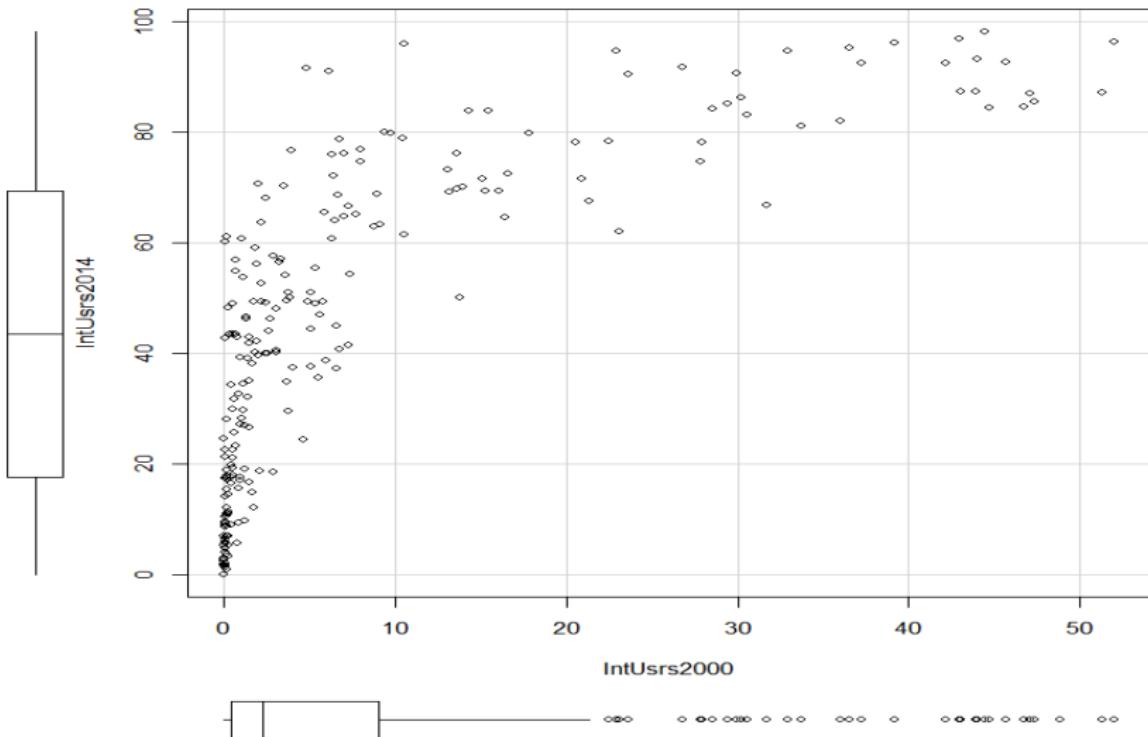


Making a Scatterplot



Scatterplot

World Wide Internet Usage: Years 2014 v 2000



Modifying a Dataset

- Rcmdr also provides several tools to modify the active data sets. Some examples are:
- Compute a new variable
- Convert a numeric to a factor variable
- Bin a numeric variable
- Sub-dividing the data (either by columns or by rows)

Useful Tips for Session Management

- Exiting and saving the script:
 - ▶ Helps keep track of the session
 - ▶ A saved script can be opened in a new session to start from the last point of save
 - ▶ Better option than saving the workspace
- Saving and printing the output
 - ▶ Cut (ctrl-c) -and- paste (ctrl-v) output (ongoing) into an open [word/text] report document
 - ▶ Best practice is to write notes as the output is moved to the report
 - ▶ Graphs may also be similarly pasted or use File > Save as method described earlier
- Commands generated by the R Commander appear in the script window:
 - ▶ This window can be edited so that only those commands that worked are saved
 - ▶ To send this script you have to highlight the relevant text and press the 'Submit' button.

Reading Data in RStudio: Example

- Data set consists of annual data (2000-2014) on the 'Number of Internet users per 100' for 244 countries.
- Rcmdr has many options for reading datasets. A typical dataset with both character and numeric variables, either .csv or .xlsx type seem to be quite stable.
- rattle can also read R data frames, or directly from R scripts etc
- InternetUsers.csv data set consists of 244 rows and 17 rows. First two rows are 'Country Name' and 'Country Code'. The following 15 columns, IntUsers2000-IntUsers2014, are numeric data.
- Tools -> Import Data ->
- Browse down to the desired file

Reading Data in RStudio: Example

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Displays R script code for reading data from a CSV file. The code includes logic to handle missing data and print messages to the console.
- Console:** Shows the output of the R script. It includes messages about saving version 1.0.12.2, attaching the 'rattle' package, and loading the 'Hmisc' package. It also lists masked objects from 'package:base'.
- Environment View:** Shows the global environment with various objects listed by name, type, length, size, and value.
- Packages View:** Shows the installed packages, their descriptions, and versions.

```
152 end.y<-xbr[1]
153 }
154 if (flag.vec)
155 txt<-paste(
156 par(xpd=T)
157 text(xbr[i+1]-xbr[1], ylim.height-par('cxy')[2]*(a
158 )
159 })
160 } else print('no spikes or more than one spike')
161 }
162 }
163 
```

```
0.40 [r] mnormmix, pnorm = 0, pnorm.link = "tanh", rts.density = "truncated normal", rts.parame = c(0, 20) : 

Console
> ## OR SAVE VERSIONS PRIOR TO 1.0.12.2 DEPRECATEED
> rattle()
> rattle()

Attaching package: 'Hmisc'

The following objects are masked from 'package:base':

  format.pval, round.POSIXt, trunc.POSIXt, units

Error in viewData() : could not find function "gladexMLNew"
Error: bad restore file magic number (file may be corrupted) -- no
data loaded
In addition: Warning message:
file 'InternetUsers.csv' has magic number 'Count'
#> 
```

| Name | Description | Version |
|-----------|---|----------|
| rattle | Graphical User Interface for Data Mining in R | 4.1.0 |
| bitops | Bitwise Operations | 1.0-6 |
| jsonlite | A Robust, High-Performance JSON Parser and Generator for R | 1.0 |
| kable | A General-Purpose Package for Dynamic Report Generation in R | 1.13 |
| magrittr | A Forward-Pipe Operator for R | 1.5 |
| minqa | Derivative-free optimization algorithms by quadratic approximation | 1.2.4 |
| Rcpp | Seamless R and C++ Integration | 0.12.6 |
| RcppEigen | 'Rcpp' Integration for the 'Eigen' Templated Linear Algebra Library | 0.3.2.81 |
| stringr | Simple, Consistent Wrappers for Common String Operations | 1.0.0 |

Reading Data in RStudio: Example

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Displays R code in the "Internet.R" script. The code includes conditional statements for reading data from files like "xbr" and "InternetUsers.txt". It also shows a warning about deprecated save versions and a message about attaching the "Hmisc" package.
- Console:** Shows the output of the R code. It includes error messages related to the "viewdata" function and corrupted restore files, as well as a warning about deprecated save versions.
- Environment View:** Shows the global environment with objects like "AnovaModel1.1", "AnovaModel1.2", and "AnovaModel1.3".
- File Import Dialog:** A modal dialog titled "Select File to Import" is open. It shows a list of files in the "Documents library" folder:

| Name | Date modified | Type |
|-------------------------|-------------------|------------------|
| Internet Users.txt | 7/31/2016 9:24 AM | Text Document |
| Internet Users.xlsx | 7/31/2016 9:06 AM | Microsoft Office |
| InternetUsers.csv | 7/31/2016 9:55 AM | Microsoft Office |
| Popular Indicators.xlsx | 7/30/2016 9:26 PM | Microsoft Office |
- Taskbar:** Shows various application icons including Internet Explorer, Firefox, Google Chrome, and R Commander.
- System Tray:** Shows the date and time as "11:28 AM 8/4/2016".

Reading Data in RStudio: Example

The screenshot shows the RStudio interface with several panes:

- Code pane:** Displays R code for reading a CSV file. A portion of the code is highlighted in yellow.
- Environment pane:** Shows the global environment with objects like AnovaModel1.1, AnovaModel1.2, AnovaModel1.3, and rra_10 through rra_1001.
- File browser pane:** Shows a tree view of files and folders, including a file named "InternetUsers.csv".
- Console pane:** Displays the output of the R code, including error messages about corrupted files and deprecated save versions.

A large oval highlights the "Import Dataset" dialog box in the center of the interface. This dialog is used for importing the "InternetUsers.csv" file. It includes fields for "Name" (set to "InternetUsers"), "Input File" (set to "InternetUsers.csv"), "Encoding" (set to "Automatic"), "Heading" (set to "Yes"), "Row names" (set to "Automatic"), "Separator" (set to "Tab"), "Decimal" (set to "Period"), "Quote" (set to "None"), and "Comment" (set to "None"). The "nesting" dropdown is set to "No" with a checked checkbox for "Strings as Factors". The "Import" button is visible at the bottom right of the dialog.

Reading Data in RStudio: Example

The screenshot shows the RStudio interface with several panes open:

- Environment** pane: Shows a list of objects in the global environment, including `datal`, `ex6p2`, `ex6p2df`, `Gumbelsamples`, `Hematoma1`, `Hematoma2`, `Hematoma3`, and `Internet.Use`.
- Files** pane: Shows a list of installed packages: `rattle`, `bitops`, `jsonlite`, `kable`, `magritr`, `minqa`, `Rcpp`, `RcppEigen`, and `stringr`. The `rattle` package is selected.
- Console** pane: Displays the R code used to read the data and the resulting warning messages. The code includes reading from a CSV file and an Excel file, and the warning messages indicate issues with embedded nulls in the CSV file.
- Internet Users** pane: Shows a data frame with columns: Country, Name, Country.Code, InternetUsers2000, InternetUsers2001, InternetUsers2002, InternetUsers2003, InternetUsers2004, InternetUsers2005, and InternetUsers2006. The data includes rows for countries like Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Antigua and Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Belarus, Belgium, Bulgaria, Cambodia, Canada, Chile, China, Colombia, Costa Rica, Czech Republic, Denmark, Ecuador, Egypt, El Salvador, Finland, France, Germany, Greece, Hungary, India, Indonesia, Ireland, Italy, Japan, Jordan, Kenya, Malaysia, Mexico, Morocco, Netherlands, Norway, Oman, Pakistan, Peru, Poland, Portugal, Russia, Saudi Arabia, South Africa, Spain, Sweden, Switzerland, Turkey, United Kingdom, United States, Uruguay, Venezuela, Vietnam, Yemen, and Zambia.

Descriptive Statistics in RStudio: Example

The screenshot shows the RStudio interface with several panes:

- Code Editor:** Displays R code for reading a CSV file and viewing its contents.
- Console:** Shows the output of the R code, including the data frame structure and a list of objects.
- Environment:** Shows the global environment with various data frames and objects.
- Packages:** Shows available packages and their versions.
- Viewer:** Shows the contents of the 'Internet.Users' data frame.

A red oval highlights the 'interaction' object in the Environment pane, and a red arrow points from it to the text "Example of code completion..".

```
R> # This file appears to contain embedded nulls
3: In read.table(path, encoding = encoding, header = header, sep =
sep, :
  incomplete final line found by readTableHeader on '~/git-remote/R_Thai_workspace/Rstudio stuff/Internet Users.xlsx'
> View(Internet.Users)
> ls()
[1] "AnovaModel.1"   "AnovaModel.2"   "AnovaModel.3"
[4] "AnovaModel.4"   "data1"          "ex6p2"
[7] "ex6p2df"        "GumbelSamples"  "Hematomal"
[10] "Hematoma2"      "Hematoma3"     "Internet.Users"
[13] "sweetcorn"       "sweetcorn2"    "sweetcorn3"
[16] "t.test.summary" "tmp1"          "trt"
[19] "wid"             "``"            "x1"
[22] "x2"             "``"            "x4"
[25] "y"              "interaction  {base}  Example of code completion..
There were 5 interaction.plot {stats} to see them)
> summary(Intern
```

Descriptive Statistics in RStudio: Example

The screenshot shows the RStudio interface with the following components:

- Source Panel:** Displays the R code and its output. The code runs the `summary` function on the `Internet.users` dataset, showing descriptive statistics for various variables like Country.Name, Country.Code, and IntUsrs2000.
- Environment Panel:** Shows the global environment with objects like `datal`, `ex6p2`, `ex6p2df`, `Gumbelsamples`, `Hematoma1`, `Hematoma2`, `Hematoma3`, and `Internet.Use`.
- Viewer Panel:** Displays a list of installed packages, including `R Commander`, `RColorBrewer`, `Rcpp`, `RcppEigen`, `arm`, `Nigra`, and `THData`.

Using Rcmdr from within RStudio

- Once the data is in RStudio as a dataframe, it is available to R or any package there of
- All packages including those with their own GUI can be active simultaneously with RStudio
- Invoke Rcmdr as illustrated before and Rcmdr will open its GUI
- Results of Rcmdr commands are displayed in the RStudio Console
- The Environment pane is also updated accordingly. For example, if a new dataset is read into Rcmdr it will show up in the RStudio Environment pane. (Still need to attach it.)
- The History pane does not seem to be updated.

Descriptive Statistics in RStudio: Example

- First, attach the dataframe to Rcmdr as follows

The screenshot shows the RStudio interface with the 'R Commander' window open. The 'Data' menu is highlighted, and a context menu is displayed over a data frame named 'data1'. The context menu includes options like 'Select active data set...', 'Refresh active data set', and 'Save active data set...'. The 'Global Environment' pane on the right lists various objects and their details.

```
> summary(Internet.users)
   Country.Name Country.Code IntUsrs2000
Afghanistan : 1 ABW : 1 Min.    : 0.000
Albania     : 1 ADO : 1 1st Qu.: 0.500
Algeria     : 1 AFG : 1 Median   : 2.900
American Samoa: 1 AGO : 1 Mean    : 10.500
Andorra     : 1 ALB : 1 3rd Qu.:13.200
Angola      : 1 ARB : 1 Max.    :64.000
(Other)     :238 (Other):238
   IntUsrs2001 IntUsrs2002 In
Min.    : 0.00  Min.    : 0.00 Min.
1st Qu.: 0.60 1st Qu.: 1.10 1st
Median  : 2.90 Median : 4.60 Med
Mean    :10.50 Mean   :13.44 Mea
3rd Qu.:13.20 3rd Qu.:18.60 3rd
Max.    :64.00 Max.    :79.10 Max.
NA's   :19 NA's   :17 NA
   IntUsrs2004 IntUsrs2005
Min.    : 0.000 Min.    : 0.00
1st Qu.: 2.375 1st Qu.: 3.15
Median  : 8.700 Median :11.40
Mean    :18.387 Mean   :20.51
3rd Qu.:27.475 3rd Qu.:33.02
Max.    :83.900 Max.    :87.00
NA's   :20 NA's   :18 NA's :19
   IntUsrs2007 IntUsrs2008 IntUsrs2009
Min.    : 0.000 Min.    : 0.00 Min.    : 0.00
1st Qu.: 4.775 1st Qu.: 6.20 1st Qu.: 7.35
Median  :16.350 Median :21.00 Median :24.70
Mean    :25.204 Mean   :28.01 Mean   :30.67
3rd Qu.:40.425 3rd Qu.:44.40 3rd Qu.:50.60
```

Figure

Descriptive Statistics in RStudio: Example

- First, attach the dataframe to Rcmdr

The screenshot shows the RStudio interface with the following details:

- Console:** Displays the command `> summary(Internet.Users)` and its output, which includes descriptive statistics for various countries across three years (2000, 2001, 2004).
- Environment:** Shows the global environment with objects like `data1` and `ex6p2`.
- R Commander:** A modal dialog titled "Select Data Set" is open, listing available datasets: `Hemato1`, `Hemato2`, `Hemato3`, `Internet.Users` (which is selected), `swetwcom`, `swetwcom3`, and `wide`. It has "OK" and "Cancel" buttons.
- Help:** Shows the help documentation for the `lapply` function.
- Session:** Shows the session history and workspace.

Descriptive Statistics in RStudio: Example

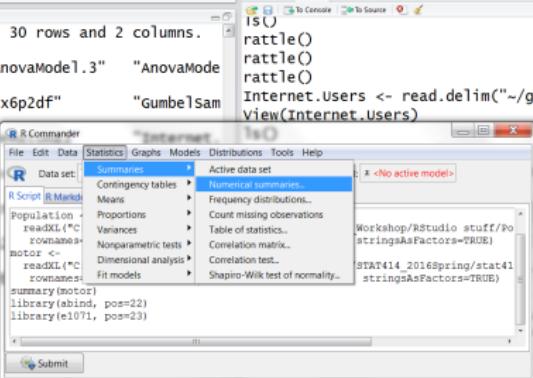
- Second, view the data set

The screenshot shows the RStudio interface. On the left, the Source pane displays R code and its output. A callout bubble points to the 'View' button in the R Commander toolbar, which is highlighted with a black circle. The R Commander window is open, showing a dataset named 'InternetUsers'. The Data set tab is selected. Below it, a preview window displays the first 10 rows of the dataset:

| | Country.Name | Country.Code | IntUsrs2000 | Int% |
|----|---------------------|--------------|-------------|------|
| 1 | Afghanistan | AFG | NA | |
| 2 | Albania | ALB | 0.1 | |
| 3 | Algeria | DZA | 0.5 | |
| 4 | American Samoa | ASM | NA | |
| 5 | Andorra | ADO | 10.5 | |
| 6 | Angola | AGO | 0.1 | |
| 7 | Antigua and Barbuda | ATG | 6.5 | |
| 8 | Argentina | ARG | 7.0 | |
| 9 | Armenia | ARM | 1.3 | |
| 10 | Aruba | ABW | 15.4 | |

Descriptive Statistics in RStudio: Example

- Third, Statistics > Summaries > Numerical summaries



The screenshot shows the RStudio interface with the R Commander add-on open. The code pane on the left contains R code for data manipulation and statistical analysis. The R Commander window is in the foreground, with the 'Summaries' menu open, specifically the 'Numerical summaries' option. The status bar at the bottom indicates 'R Commander'.

```
g: [4] NOTE: The dataset motor has 30 rows and 2 columns.  
novaModel.1" "AnovaModel.2" "AnovaModel.3" "AnovaMode  
atal" "ex6p2" "ex6p2df" "GumbelSam  
ematoma1" "Hematoma2"  
otor" "sweetcorn"  
.test.summary" "tmp1"  
"  
"x1"  
4" "y"  
  
summary(motor)  
ion..microns. Brand  
:11.60 Min. :1  
.13.40 1st Qu.:2  
:13.95 Median :3  
:14.22 Mean :3  
.14.90 3rd Qu.:4  
:17.20 Max. :5  
  
library(abind, pos=22)
```

Descriptive Statistics in RStudio: Example

- Fourth, choose variable(s) and select options
- Finally, Click Apply or OK

```
RMSY: ~4 NOTE: THE DATASET MOTOR HAS 30 ROWS AND 2 COLUMNS.  
| "AnovaModel.1" "AnovaModel.2" "AnovaModel.3" "AnovaMode  
| "data1" "ex6p2" "ex6p2df" "GumbelSam  
| "Hematoma1" "Hematoma2"  
| "motor" "sweetcorn"  
| "t.test.summary" "tm"  
| "x" "x1"  
| "x4" "y"  
  
ir> summary(motor)  
>ratanion..microns.  
. Min. :11.60  
: Qu.:13.40 1st :  
ian :13.95 Medi:  
n :14.22 Mean:  
f Qu.:14.90 3rd :  
c. :17.20 Max.  
  
ir> library(abind, pos=22)  
In: 14library(abind, pos=22)
```

The screenshot shows the R Commander interface. In the background, there is a help window titled 'R Commander' with the text: 'rattle() rattle() rattle() Internet.Users <- read.delim("~/git-remote/R_Thai_Workshop/RStudio/Internet.Users") view(Internet.Users)'.

Two 'Numerical Summaries' dialog boxes are overlaid. The left dialog box has 'Data' selected in the tabs and contains a list of variables: 'IntUsers2000', 'IntUsers2001', 'IntUsers2002', 'IntUsers2003', 'IntUsers2004', 'IntUsers2005'. Below the list is a button 'Summarize by groups...'. At the bottom are buttons for 'Help', 'Reset', 'OK', 'Cancel', and 'Apply'.

The right dialog box has 'Statistics' selected in the tabs and contains a list of statistical measures: 'Mean' (checkbox checked), 'Standard Deviation' (checkbox unselected), 'Standard Error of Mean' (checkbox unselected), 'Interquartile Range' (checkbox checked), 'Coefficient of Variation' (checkbox unselected), 'Skewness' (radio buttons 'Type 1' and 'Type 2' available), 'Kurtosis' (radio buttons 'Type 2' and 'Type 3' available), and 'Quantiles: 0, 25, 5, 75, 1' (checkbox checked). Below the list are buttons for 'Help', 'Reset', 'OK', 'Cancel', and 'Apply'.

Descriptive Statistics in RStudio: Example

- Formatting of the Rcmdr summaries function is not great. We can look for other options. For example, there is a function called stat.desc(). How do we find it?

The screenshot shows the RStudio interface with the search results for 'stat.desc' highlighted in the 'Search Results' pane.

Source:

```
Rcmdr+   IntUsrs2000 ,  IntUsrs2004 ,  IntUsrs2005 ,  IntUsrs2006 , "IntUsrs2007" ,  
Rcmdr+   "IntUsrs2008" , "IntUsrs2009" , "IntUsrs2010" , "IntUsrs2011" , "IntUsrs2012" ,  
Rcmdr+   "IntUsrs2013" , "IntUsrs2014" ) , statistics=c("mean" , "sd" , "IQR" ,  
Rcmdr+   "quantiles") , quantiles=c(0.25 , .5 , .75 , 1))  
      mean      sd     IQR 0% 25% 50% 75% 100%  
n NA  
IntUsrs2000 8.51875 13.04039 8.525 0.0 0.500 2.30 9.025 52.0  
224 20  
IntUsrs2001 10.50178 15.22477 12.600 0.0 0.600 2.90 13.200 64.0  
225 19  
IntUsrs2002 13.43744 18.13099 17.500 0.0 1.100 4.60 18.600 79.1  
227 17  
IntUsrs2003 15.97692 20.03524 21.300 0.0 1.700 6.70 23.000 83.1  
221 23  
IntUsrs2004 18.38750 21.42359 25.100 0.0 2.375 8.70 27.475 83.9  
224 20  
IntUsrs2005 20.50531 22.62030 29.875 0.0 3.150 11.40 33.025 87.0  
226 18  
IntUsrs2006 23.01156 23.65560 33.700 0.0 4.300 14.20 38.000 89.5  
225 19  
IntUsrs2007 25.20388 24.77963 35.650 0.0 4.775 16.35 40.425 90.6  
232 12  
IntUsrs2008 28.00996 25.66116 38.200 0.0 6.200 21.00 44.400 91.0  
231 13  
IntUsrs2009 30.67391 26.29545 43.250 0.0 7.350 24.70 50.600 93.0  
230 14  
IntUsrs2010 33.75844 26.66192 43.500 0.0 10.000 29.00 53.500 93.4  
231 13  
IntUsrs2011 36.72759 27.04104 45.600 0.0 11.500 34.00 57.100 94.8
```

Environment:

```
rattle()  
rattle()  
rattle()  
Internet.Users <- read.delim("~/git-remote/R_Thai_Work/  
view(Internet.Users)  
ls()  
summary(Internet.users)  
library("Rcmdr", lib.loc = "/R/win-library/3.3")  
ls()
```

Help:

Name of the package obtained here
Type name of the function here

The search a...
Help pages:
[pastecs::stat.desc](#) Descriptive statistics on a data frame or time series

Descriptive Statistics in RStudio: Example

- Now we can install the package `pastecs`, and then get help for `stat.desc()`:

The screenshot shows the RStudio interface with two windows open. The left window is the 'Console' tab, displaying R code and its output. The right window is the 'Help' tab, showing the help documentation for the `stat.desc` function from the `pastecs` package.

Console Output:

```
Warning in .HTMLsearch(query) : unrecognized search field: keyword
Warning in .HTMLsearch(query) : Unrecognized search field: alias
Making 'packages.html' ... done
Warning in .HTMLsearch(query) : Unrecognized search field: title
Warning in .HTMLsearch(query) : Unrecognized search field: keyword
Warning in .HTMLsearch(query) : Unrecognized search field: alias
Warning in .HTMLsearch(query) : Unrecognized search field: title
Warning in .HTMLsearch(query) : Unrecognized search field: keyword
Warning in .HTMLsearch(query) : Unrecognized search field: alias
> library("pastecs", lib.loc="~/R/win-library/3.3")
Loading required package: boot

Attaching package: 'boot'

The following object is masked _by_ '.GlobalEnv':
  motor

The following object is masked from 'package:car':
  logit

The following object is masked from 'package:survival':
  aml

The following object is masked from 'package:lattice':
  melanoma
```

Help Documentation:

The 'Help' tab shows the `stat.desc` function from the `pastecs` package. A green circle highlights the package name `pastecs` in the list of packages. A green arrow points from this circle to the package entry in the 'Details' section of the help page, which is also circled in green.

Help Page Content:

`library("pastecs", lib.loc="~/R/win-library/3.3")`

stat.desc (pastecs)

Description

Compute a table giving various descriptive statistics about the series in a data frame or in a single/multiple time series

Usage

```
stat.desc(x, basic=TRUE, desc=TRUE, norm=FALSE, p=0.95)
```

Arguments

`x` a data frame or a time series

`basic` do we have to return basic statistics (by default, it is TRUE)? These are: the number of values (nbr.val), the number of null values (nbr.null), the number of missing values (nbr.na), the minimal value (min), the maximal value (max), the range (range, that is, max-min) and the sum of all non-

Descriptive Statistics in RStudio: Example

- These results may be cut-and-pasted to a report in Word or similar word processing tool

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Shows R code for generating descriptive statistics. A red oval highlights the text "Note that the width of the console is made wider to view the results better." pointing to the console area.
- Console:** Displays the command history and the resulting summary statistics for the "Internet.Users" dataset. The output includes various summary statistics like mean, median, and standard deviation for different variables.
- Environment:** Shows the global environment with objects like "Internet.Users", "I.U.summary", and "stat.desc".
- Help:** Shows the documentation for the "stat.desc" function, which is used to calculate descriptive statistics for time series data.
- Output:** Shows the detailed summary statistics for the "Internet.Users" dataset.

Key code in the editor:

```
161     else print('no spikes or more than one spike')
162   }
163   invisible(list(heights = heights, xbr = xbr))
164 }
165 else {
166   return(list(heights = heights, xbr = xbr, counts=counts))
167 }
168 }
169
170
171 }
```

Key output in the console:

```
Country.Name Country.Code IntUsrs2000 IntUsrs2001 IntUsrs2002
nbr.val      NA        224.000000  225.000000  227.000000
nbr.null     NA        11.000000   8.000000   5.000000
nbr.na       NA        20.000000   19.000000  17.000000
min          NA        0.00000000  0.00000000  0.00000000
max          NA        52.00000000 64.00000000 79.100000
range         NA        52.00000000 64.00000000 79.100000
sum          NA        1908.20000000 2362.90000000 3050.300000
median        NA        2.30000000  2.90000000  4.60000000
mean          NA        8.51875000 10.50177800 13.437445
SE.mean        NA        0.8712977 1.014985 1.203396
CI.mean        NA        1.7170305 2.000140 2.371312
var           NA        170.0517545 231.793658 328.732884
std.dev        NA        13.0403894 15.224771 18.130992
coef.var       NA        1.5307867 1.449733 1.349289
IntUsrs2003  IntUsrs2004  IntUsrs2005  IntUsrs2006  IntUsrs2007
```

Key output in the Environment pane:

```
I.U.summary <- stat.desc(Internet...  
print(I.U.summary)  
print(I.U.summary[, -(1:2)])  
stat.desc(Internet.Users)
```

Key help output in the Help pane:

```
stat.desc(x, y, mout=NULL, xmin=min(x), n=NULL, frequency=N  
delta=1/frequency, basic=FALSE, desc=FALSE, norm=F  
per=FALSE, p0.95)  
## S3 method for class "stat.slide"  
print(x, ...)  
## S3 method for class "stat.slide"  
plot(x, stat="mean", col=c(1, 2), lty=1, pch=1, lty1=1, lty2=1,  
lwd=1, main=paste("Sliding statistics", ...))  
## S3 method for class "stat.slide"  
lines(x, stat="mean", col=3, lty=1, ...)
```

Key arguments in the Help pane:

- x: a vector with time data for `stat.slide()`, or a `stat.slide` object for the methods
- y: a vector with observation at corresponding times

Graphs in RStudio

- Can use any R function that plots, and the plot will appear in the Plot pane. Successive plots will be saved, and can be navigated using the arrow buttons

The screenshot shows the RStudio interface with the following components:

- Editor pane:** Displays R code related to histogram plotting. A green circle highlights the "To navigate" button in the toolbar.
- Environment pane:** Shows the current environment with objects like `Internet.Users`, `ls()`, `summary(Internet.Users)`, `library("Rcmdr")`, `stat.desc(Internet.Users)`, and `IU_SUMMARY`.
- Plot pane:** Displays a histogram titled "Histogram of Internet.Users\$IntUsrs2012". The x-axis is labeled "Internet.Users\$IntUsrs2012" and ranges from 0 to 100. The y-axis is labeled "Frequency" and ranges from 0 to 40. The histogram bars are shaded with diagonal lines.

Graphs in RStudio

- A better program would be to write a loop.
- Start editing a R Script file as shown below.

The screenshot shows the RStudio interface with the following details:

- File Menu:** New File, New Project..., Open File..., Recent Files, Open Project..., Recent Projects, Save, Save As..., Save with Encoding..., Save All, Knit, Compile Notebook..., Print..., Close, Close All, Close Project, Quit RStudio...
Ctrl+O Ctrl+S Ctrl+Alt+S Ctrl+Shift+K Ctrl+P Ctrl+W Ctrl+Shift+W
- Code Editor:** An R Script window containing the following code:

```
han one spike')
xbr = xbr))
heights = heights, xbr = xbr,counts=count
ls()
summary(Internet.Users)
library("Rcmdr", lib.loc="~/R/win-library/3.3")
ls()
library("pastecs", lib.loc="~/R/win-library/3.3")
stat.desc(Internet.Users)
IU.summary <- stat.desc(Internet.Users)
print(IU.summary)
print(IU.summary[,-(1:2)])
stat.desc(Internet.Users)
```
- Console:** Shows the history of R commands entered, including attempts to plot non-numeric data and an error message about density being TRUE.

```
> hist(Internet.Users$IntUsrs2005,density=TRUE)
> hist(Internet.Users$IntUsrs200r,density=TRUE)
Error in hist.default(Internet.Users$IntUsrs200r, density = TR
UE) :
  'x' must be numeric
> hist(Internet.Users$IntUsrs2006,density=TRUE)
> hist(Internet.Users$IntUsrs2007,density=TRUE)
> hist(Internet.Users$IntUsrs2008,density=TRUE)
> hist(Internet.Users$IntUsrs2009,density=TRUE)
> hist(Internet.Users$IntUsrs2010,density=TRUE)
> hist(Internet.Users$IntUsrs2011,density=TRUE)
> hist(Internet.Users$IntUsrs2012,density=TRUE)
> hist(Internet.Users$IntUsrs2013,density=TRUE)
> hist(Internet.Users$IntUsrs2014,density=TRUE)
>
```
- Plots:** A histogram titled "Histogram of Internet.Users\$IntUsrs2012". The x-axis is labeled "Internet.Users\$IntUsrs2012" and ranges from 0 to 100. The y-axis is labeled "Frequency" and ranges from 0 to 40. The histogram bars are filled with diagonal hatching.
- Environment:** Shows the current environment variables.
- Help:** Provides access to R documentation and help functions.

Graphs in RStudio

- Here's a simple loop to plot all histograms, with a trick to ensure the correct title for each histogram

The screenshot shows the RStudio interface with the following components:

- Script Editor:** Displays R code for generating histograms. A green oval highlights the "Run" button at the top of the editor.
- Console:** Shows the output of the R code, including the attached datasets and the generated histogram titles.
- Plots:** Displays a histogram titled "Worldwide Internet Users in Year 2011". The x-axis is labeled "Internet Users[, i + 3]" and the y-axis is labeled "Frequency".

```
1 attach(Internet.Users)
2 for (i in 0:14) {
3   hist(Internet.Users[,i+3],density=TRUE,
4   main=paste(c("Worldwide Internet Users in year"),2000+i))
5 }
```

```
Internet.Users, Internet.Users.2010, Internet.Users.2011,
Internet.Users.2012, Internet.Users.2013, Internet.Users.2014

> ls()
[1] "AnovaModel.1"    "AnovaModel.2"    "AnovaModel.3"
[4] "AnovaModel.4"    "data1"          "ex6p2"
[7] "ex6p2df"         "GumbelSamples"  "Hematoma1"
[10] "Hematoma2"       "Hematoma3"     "i"
[13] "Internet.Users"  "IU.summary"    "motor"
[16] "sweetcorn"        "sweetcorn2"    "sweetcorn3"
[19] "t.test.summary"  "tmp"           "tmp1"
[22] "trt"              "wide"          "x"
[25] "x1"              "x2"           "x3"
[28] "x4"              "y"             "ybar"

> |
```

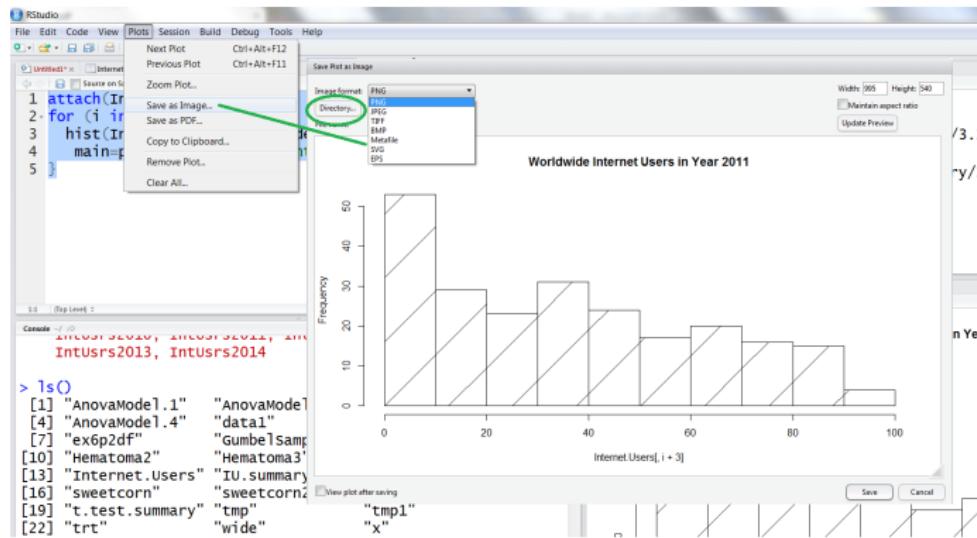
Worldwide Internet Users in Year 2011

A histogram showing the frequency distribution of Internet users in the year 2011. The x-axis is labeled "Internet Users[, i + 3]" and ranges from 0 to 100. The y-axis is labeled "Frequency" and ranges from 0 to 50. The histogram has 15 bins. The title of the plot is "Worldwide Internet Users in Year 2011".

| Bin Range (Internet Users) | Frequency |
|----------------------------|-----------|
| 0 - 10 | ~48 |
| 10 - 20 | ~28 |
| 20 - 30 | ~25 |
| 30 - 40 | ~30 |
| 40 - 50 | ~25 |
| 50 - 60 | ~18 |
| 60 - 70 | ~22 |
| 70 - 80 | ~15 |
| 80 - 90 | ~18 |
| 90 - 100 | ~5 |

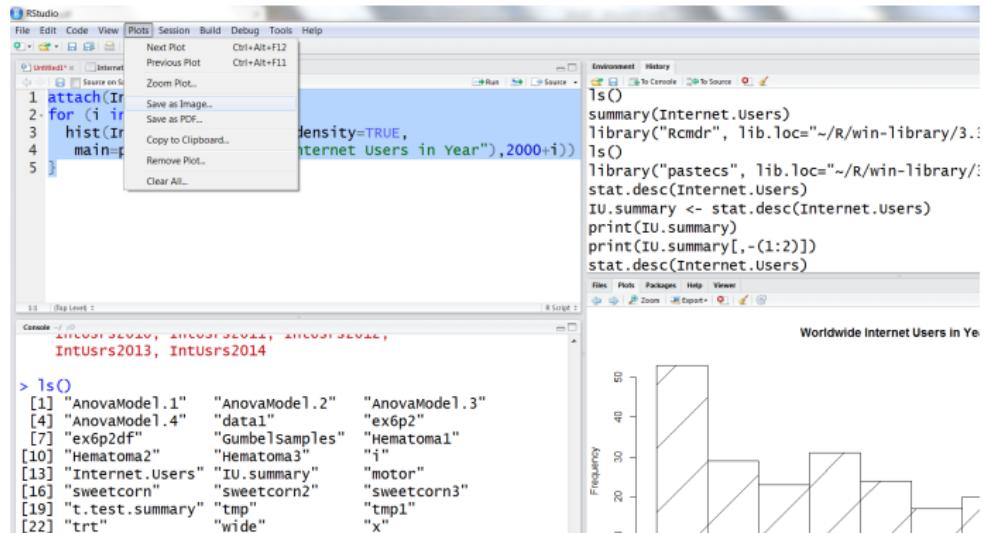
Graphs in RStudio

- Plots can be saved to the local directory: Plots -> Save as Image...



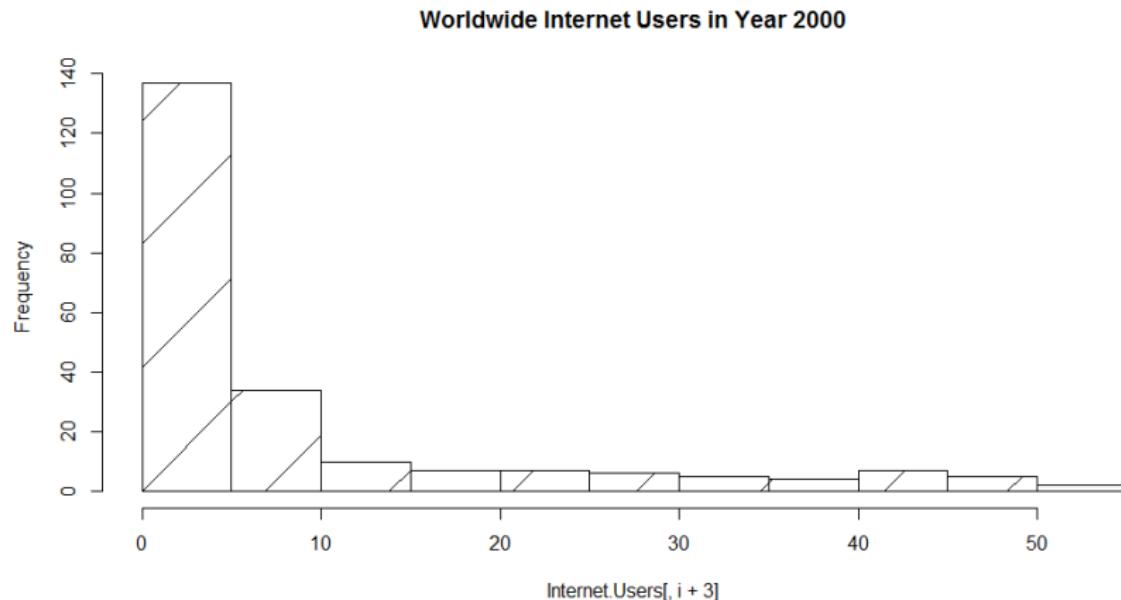
Graphs in RStudio

- Also note other options in the Plots menu:



Graphs in RStudio

- Plot is saved as a `filename.png` file and inserted into this tex document:



Scatterplot

RStudio

File Edit Code View Plots Session Build Debug Tools Help

Internet.Users

Untitled1.R

Internet.Users

Untitled2.R

Source on Save Run Source

```
1 attach(Internet.Users)
2 for (i in 0:14) {
3   hist(Internet.Users[,i+3],density=TRUE,
4   main=paste(c("Worldwide Internet Users in Year"),2000+i))
5 }
6
7 plot(Internet.Users[,3],Internet.Users[,17],xlab="IntUsers2000",
8 ylab="IntUsers2014")
9 title(main="World Wide Internet Usage 2014 vs 2000")
```

Console

```
[19] "t.test.summary" "tmp"           "tmp1"
[22] "trt"            "wide"          "x"
[25] "x1"             "x2"           "x3"
[28] "x4"             "y"            "ybar"
>
>
>
>
>
> plot(Internet.Users[,3],Internet.Users[,17])
> plot(Internet.Users[,3],Internet.Users[,17],xlab="IntUsers2000",
+ ylab="IntUsers2014")
> plot(Internet.Users[,3],Internet.Users[,17],xlab="IntUsers2000",
+ ylab="IntUsers2014")
> title(main="World Wide Internet Usage 2014 vs 2000")
>
```

Environment History

```
ls()
summary(Internet.Users)
library("Rcmdr", lib.loc="~/R/win-library/3.3")
ls()
library("pastecs", lib.loc="~/R/win-library/3.3")
stat.desc(Internet.Users)
IU.summary <- stat.desc(Internet.Users)
print(IU.summary)
print(IU.summary[,-(1:2)])
stat.desc(Internet.Users)
```

Files Plots Packages Help Viewer

Zoom Export

World Wide Internet Usage 2014 vs 2000

IntUsers2014

IntUsers2000

Side by Side Boxplots

- Suppose we want to compare the shapes of the distributions of Internet users over time.
- We need to reorganize the data before we can do this
- We can use the stack() function for this purpose

```
#   Reshaping the Internet Users Data
#   for plotting side-by-side boxplots
Internet.Users.Long <-
  stack(Internet.Users,
    measured=paste0(c("IntUsrs") ,2000:2014))
```

- Other functions for such reorganization of data: melt(), cast(), data.table()
- Change the setting from Grid to List in the Environment pane. Click on the Internet.Users.Long data.frame to bring it to the Script window Or use view()

Side by Side Boxplots

RStudio

File Edit Code View Plots Session Build Debug Tools Help

Untitled1* Internet.Users Internet.Users.Long

values ind

| | |
|----|---|
| 1 | IntUsr2000 |
| 2 | 0.1 IntUsr2000 |
| 3 | 0.5 IntUsr2000 |
| 4 | 10.5 IntUsr2000 |
| 5 | 6.5 IntUsr2000 |
| 6 | 7.0 IntUsr2000 |
| 7 | 1.3 IntUsr2000 |
| 8 | 15.4 IntUsr2000 |
| 9 | 48.9 IntUsr2000 |
| 10 | 144 "34" "161" "139" "186" "11" "185" "102" |
| 11 | "86" "137" |
| 12 | "388" "278" "346" "430" "383" "255" "429" "405" |
| 13 | "478" "330" |
| 14 | "574" "632" "673" "541" "554" "522" "674" "625" |
| 15 | "600" "590" |
| 16 | "818" "917" "876" "785" "798" "834" "918" "935" |
| 17 | "869" "766" |
| 18 | "1062" "1161" "1029" "1120" "1078" "1042" "1113" "1162" |
| 19 | "1040" "1010" |
| 20 | "1273" "1306" "1357" "1364" "1405" "1550" |

Console -> `uu= y`

`[1] "144" "34" "161" "139" "186" "11" "185" "102"
"86" "137"
[11] "388" "278" "346" "430" "383" "255" "429" "405"
"478" "330"
[21] "574" "632" "673" "541" "554" "522" "674" "625"
"600" "590"
[31] "818" "917" "876" "785" "798" "834" "918" "935"
"869" "766"
[41] "1062" "1161" "1029" "1120" "1078" "1042" "1113" "1162"
"1040" "1010"
[51] "1273" "1306" "1357" "1364" "1405" "1550"`

Rcmdr> `Boxplot(values~ind, data=Internet.Users.Long, id.method="none")`
> `Boxplot(values~ind, data=Internet.Users.Long, id.method="none")`
> `attach(Internet.Users.Long)`
> `View(Internet.Users.Long)`
>

Environment History Import Dataset Global Environment

Hematoma1 14 obs. of 12 variables
Hematoma2 42 obs. of 8 variables
Hematoma3 84 obs. of 5 variables
Internet.Users 244 obs. of 17 variables
Internet.Users.L... 3660 obs. of 2 variables
IU.summary 14 obs. of 17 variables
motor 30 obs. of 2 variables
Sweetcorn 12 obs. of 5 variables
Sweetcorn3 60 obs. of 2 variables

Files Plots Packages Help Viewer

Zoom Export

values

int

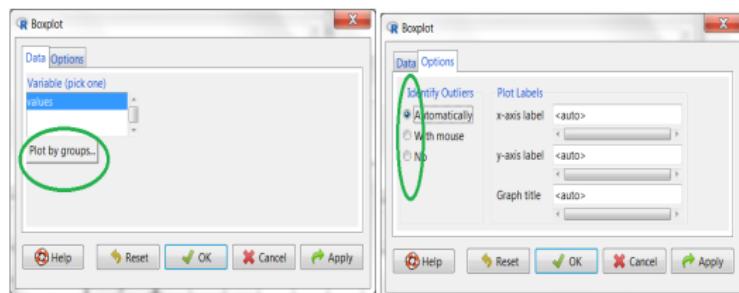
IntUsr2000 IntUsr2002 IntUsr2004 IntUsr2006 IntUsr2008 IntUsr2010 IntUsr2012 IntUsr2014

Side by Side Boxplots in Rcmdr

- Choose Internet.Usrs.Long to active data set in Rcmdr:

Data → Active Data set → Select active data set

- Use Graphs → Boxplot.... to open the dialog box for boxplot
- Click on Plot by groups and select the variable Ind as the grouping variable.



So far....

- Introduced R, RStudio, Rcmdr
- Took a brief look at their capabilities
- Compared and contrasts their capabilities
- Saw examples of how Rcmdr and RStudio interact
- Saw examples of using Rcmdr from RStudio
- Saw examples of combining them to accomplish a specific task
- We will now discuss a very comprehensive package called `rattle`

Practice Excercise

- Descriptive Statistics
- Distributions
- Univariate statistical methods

Toxics Release Inventory Data

TRI: Industries meeting certain reporting requirements self-report annual amount of designated chemicals. Main variables in the data set are as follows.

- TRI Facility Identification (TRI_FACILITY_ID), Industry sector it belongs to, Parent company
- The name of the chemical or chemical category, carcinogenicity status
- Latitude, Longitude
- Total release to all media (Unit of reporting is *pounds per year*)

Toxics Release Inventory Data

This data can be downloaded from the United States Environmental Protection Agency website by following the steps below.

1. Navigate to TRI website:
<https://www.epa.gov/toxics-release-inventory-tri-program>.
2. Click "Get TRI Data" in the middle of the page.
3. Near the bottom of the page under "More Information to Help You Use TRI Data", click on "Basic Data Files".
4. Now you should be at TRI Basic Data Files: Calendar Years 1987 - 2016 (<https://www.epa.gov/toxics-release-inventory-tri-program/tri-basic-data-files-calendar-years-1987-2016>).

Toxics Release Inventory Data

1. Perform all data summary and analysis steps shown in the previous slides for the TRI data set
2. Perform the following steps for the total release column
 - ▶ Compute the five number summary
 - ▶ Compute the percentages of data points in the intervals mean $+$ SD, mean $+$ 2SD. What are these percentages expected to be and how close are they.
 - ▶ Repeat part b for a variety of transformations of the salary data. What transformation of the data provides a good fit of the data to a Gaussian assumption on the data?