

# RStudio screen

The screenshot displays the RStudio application window. The top menu bar includes File, Edit, Code, View, Plots, Session, Project, Build, Tools, and Help. Below the menu is a toolbar with icons for file operations and a search bar labeled 'Go to file/function'. The main interface is divided into four panes:

- Console:** Shows the R version (3.0.0), copyright information, and the results of several R commands. The commands include `getwd()`, `5*5`, `matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2)`, and `matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2, byrow=TRUE)`. The output shows the current directory and the dimensions and values of the created matrices.
- Workspace:** Lists active objects in the environment. It shows two objects, 'A' and 'B', both of type '4x2 double matrix'.
- History:** Shows a list of commands used so far.
- Files:** Displays the file structure of the current workspace. It shows a folder named 'MyData' containing a file named '.Rhistory'.

The **console** is where you can type commands and see output

The **workspace** tab shows all the active objects (see next slide). The **history** tab shows a list of commands used so far.

The **files** tab shows all the files and folders in your default workspace as if you were on a PC/Mac window. The **plots** tab will show all your graphs. The **packages** tab will list a series of packages or add-ons needed to run certain processes. For additional info see the **help** tab

# Workspace tab (1)

The workspace tab stores any object, value, function or anything you create during your R session. In the example below, if you click on the dotted squares you can see the data on a screen to the left.

The screenshot shows the RStudio interface. The top pane displays the source editor with the following R code:

```
1 getwd()
2 setwd("H:/MyData/RFiles")
3 getwd()
4 5*5
5 A <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2)
6 A
7 B <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2, byrow=TRUE)
8 B
```

The bottom pane shows the workspace tab with the following data:

Object	Class
A	4x2 double matrix
B	4x2 double matrix
house.pets	3 obs. of 4 variables
feed	character[3]
pets	character[3]
run	numeric[3]
weight	numeric[3]

On the left side of the bottom pane, a data preview for matrix B is shown:

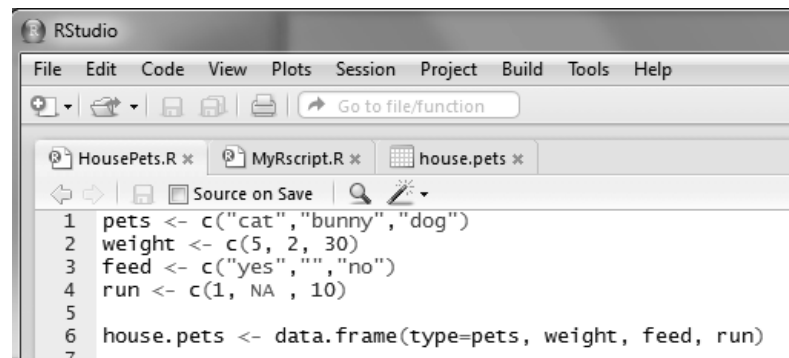
	V1	V2
1	1	2
2	3	4
3	5	6
4	7	8

Arrows indicate the relationship between the code, the workspace, and the data preview. One arrow points from the code editor to the workspace tab, and another points from the workspace tab to the data preview. A third arrow points from the text 'Showing here matrix B. To see matrix A click on the respective tab.' to the 'B' tab in the workspace.

Showing here matrix B. To see matrix A click on the respective tab.

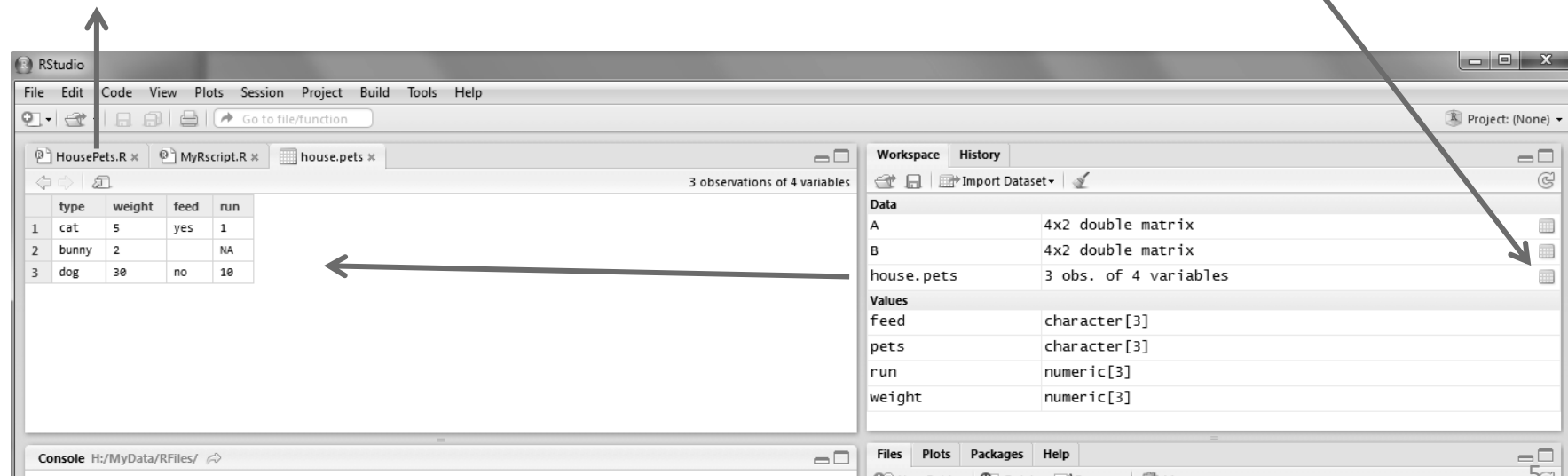
# Workspace tab (2)

Here is another example on how the workspace looks like when more objects are added. Notice that the data frame `house.pets` is formed from different individual values or vectors.



```
1 pets <- c("cat", "bunny", "dog")
2 weight <- c(5, 2, 30)
3 feed <- c("yes", "", "no")
4 run <- c(1, NA, 10)
5
6 house.pets <- data.frame(type=pets, weight, feed, run)
7
```

Click on the dotted square to look at the dataset in a spreadsheet form.



The RStudio interface shows the `house.pets` data frame in a spreadsheet view. The spreadsheet displays 3 observations of 4 variables:

	type	weight	feed	run
1	cat	5	yes	1
2	bunny	2		NA
3	dog	30	no	10

The Workspace tab on the right shows the following objects:

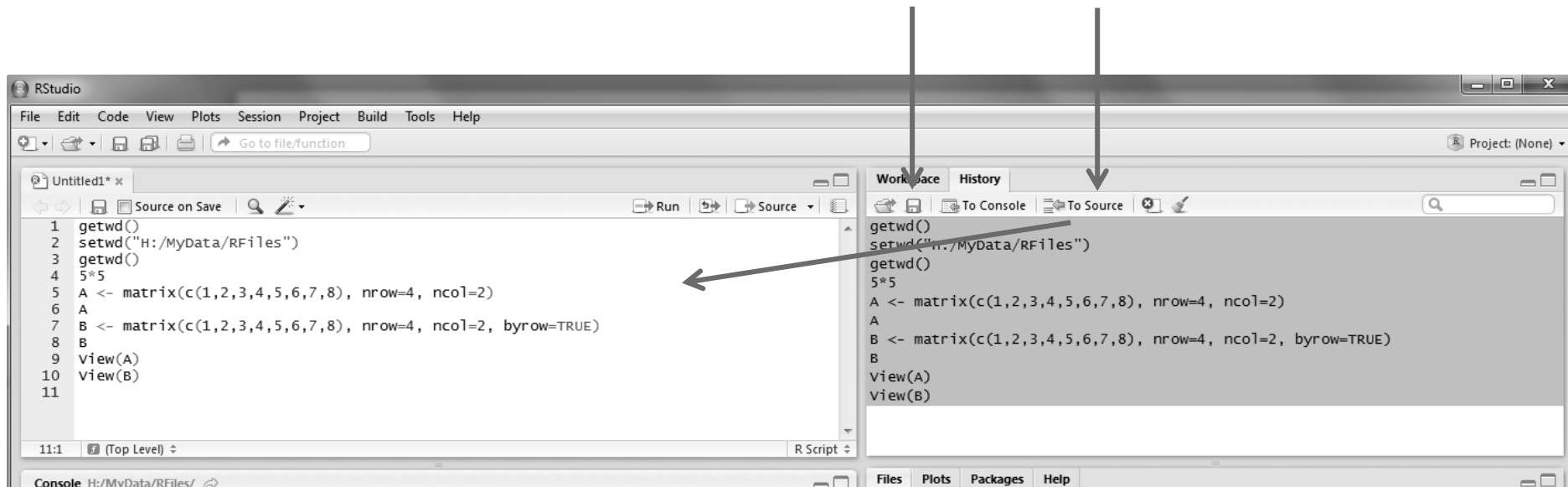
Object	Type
A	4x2 double matrix
B	4x2 double matrix
house.pets	3 obs. of 4 variables
feed	character[3]
pets	character[3]
run	numeric[3]
weight	numeric[3]

Arrows indicate the flow of information: one arrow points from the code editor to the spreadsheet view, and another points from the text instruction to the dotted square icon in the Workspace tab.

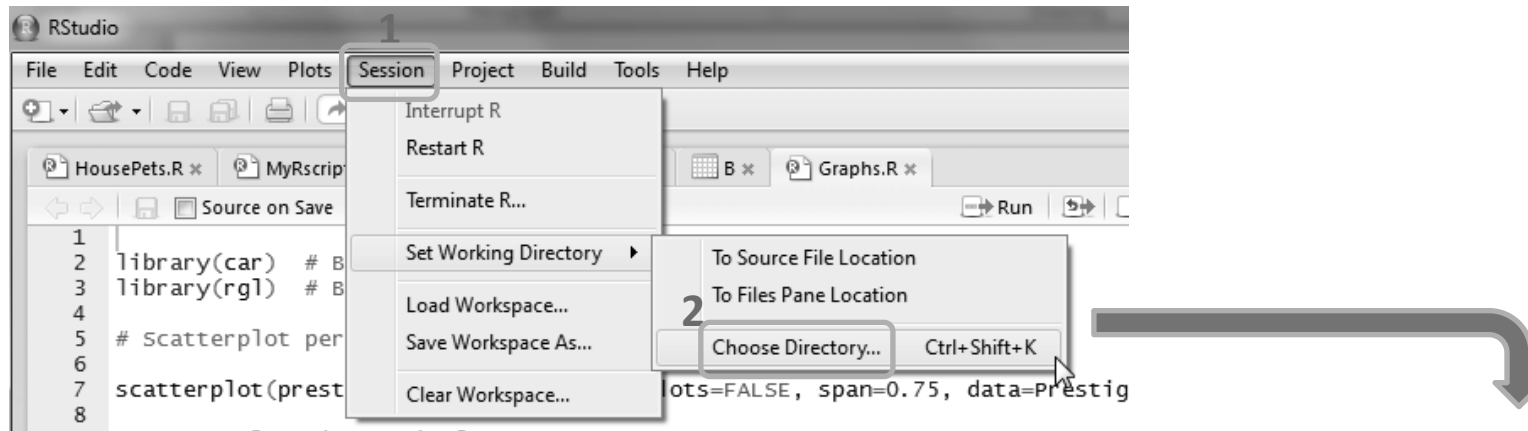
# History tab

The history tab keeps a record of all previous commands. It helps when testing and running processes. Here you can either **save** the whole list or you can **select** the commands you want and send them to an R script to keep track of your work.

In this example, we select all and click on the “To Source” icon, a window on the left will open with the list of commands. Make sure to save the ‘untitled1’ file as an \*.R script.



# Changing the working directory



If you have different projects you can change the working directory for that session, see above. Or you can type:

```
# Shows the working directory (wd)
```

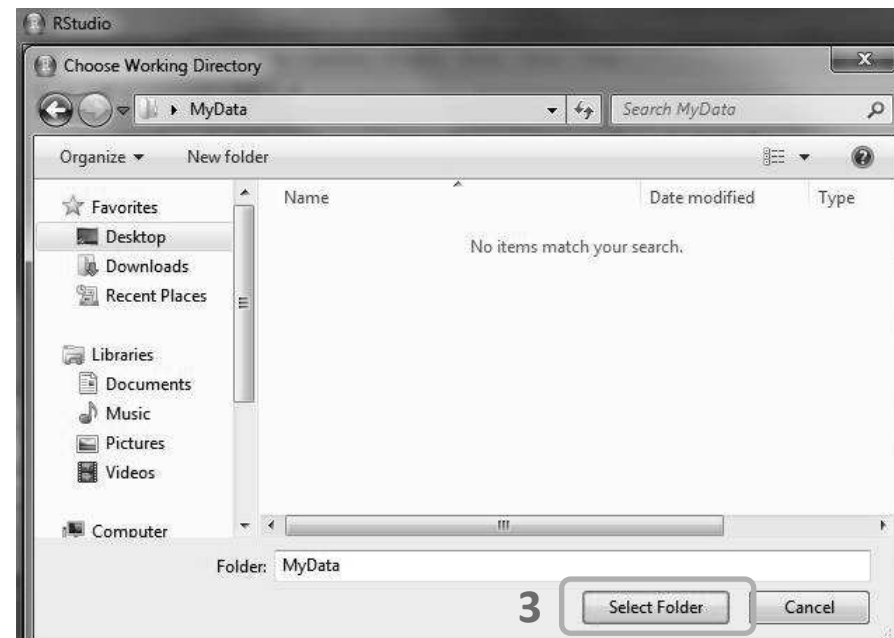
```
getwd()
```

```
# Changes the wd
```

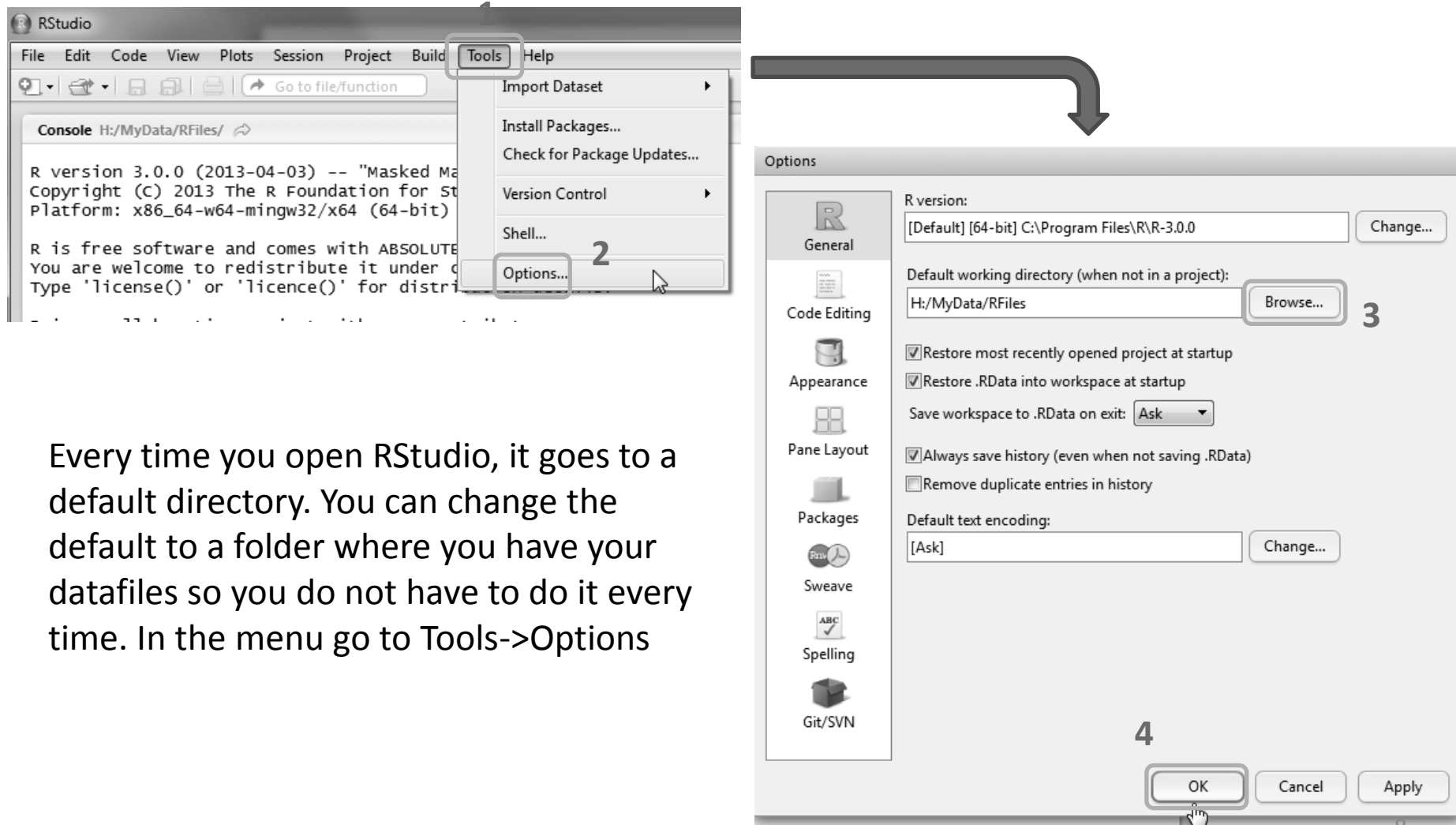
```
setwd("C:/myfolder/data")
```

More info see the following document:

<http://dss.princeton.edu/training/RStata.pdf>



# Setting a default working directory



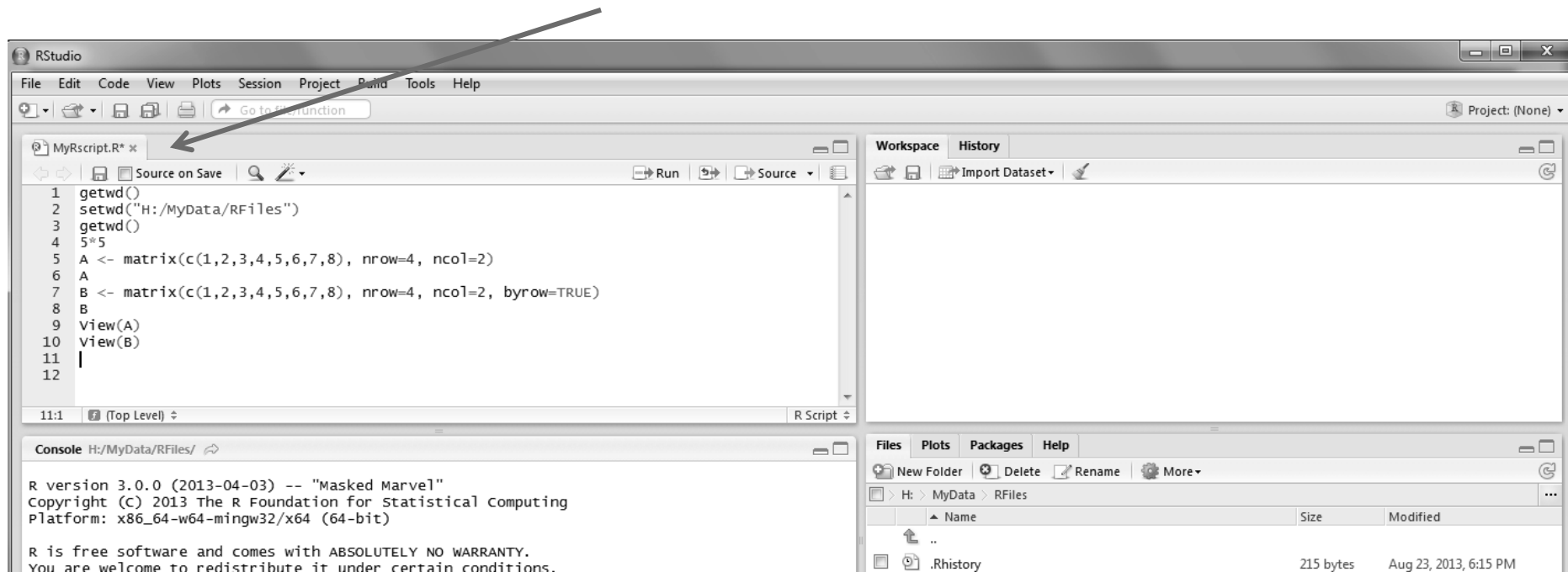
Every time you open RStudio, it goes to a default directory. You can change the default to a folder where you have your datafiles so you do not have to do it every time. In the menu go to Tools->Options

# R script (1)

The usual Rstudio screen has four windows:

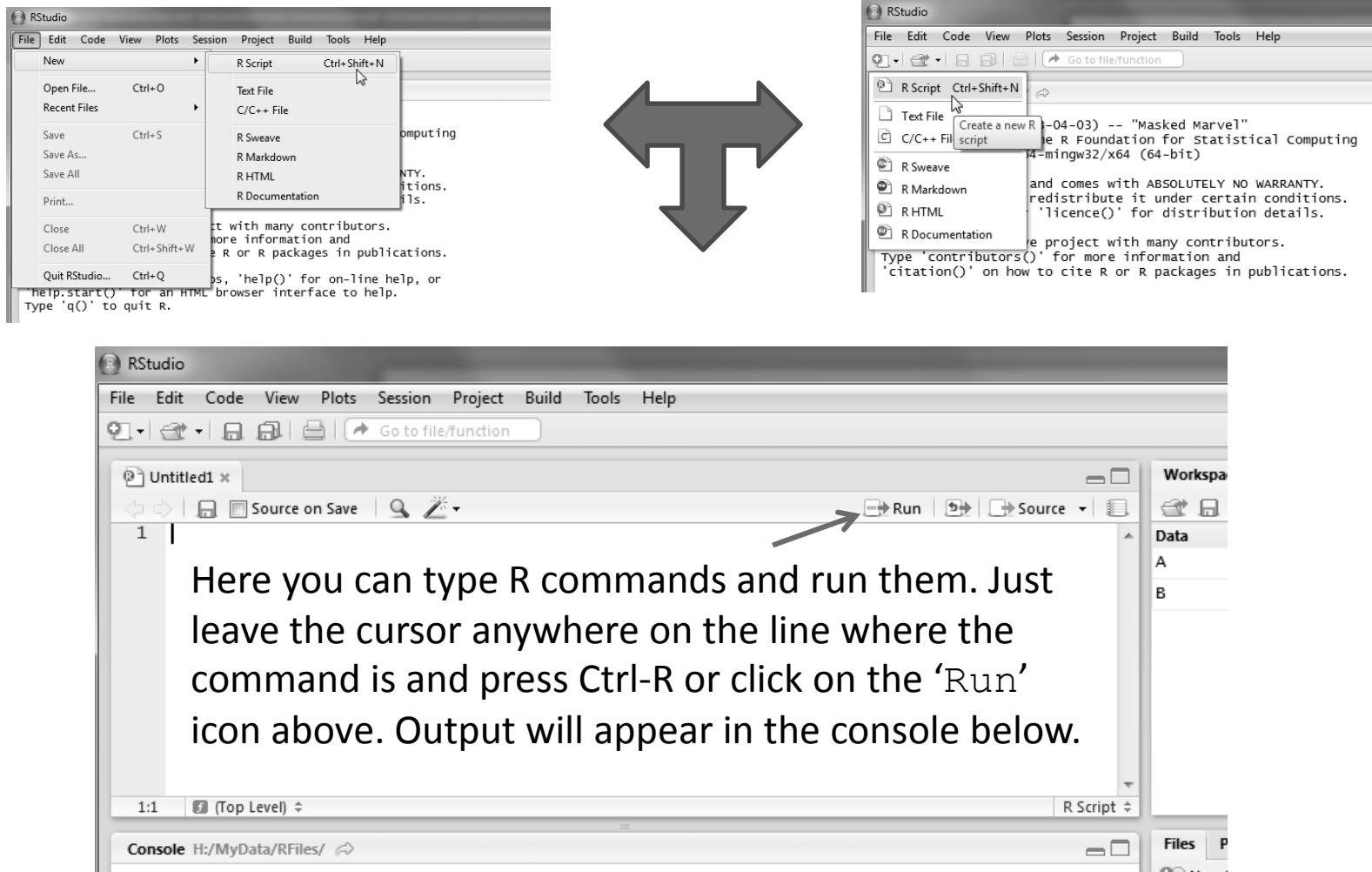
1. Console.
2. Workspace and history.
3. Files, plots, packages and help.
4. The R script(s) and data view.

The R script is where you keep a record of your work. For Stata users this would be like the do-file, for SPSS users is like the syntax and for SAS users the SAS program.



# R script (2)

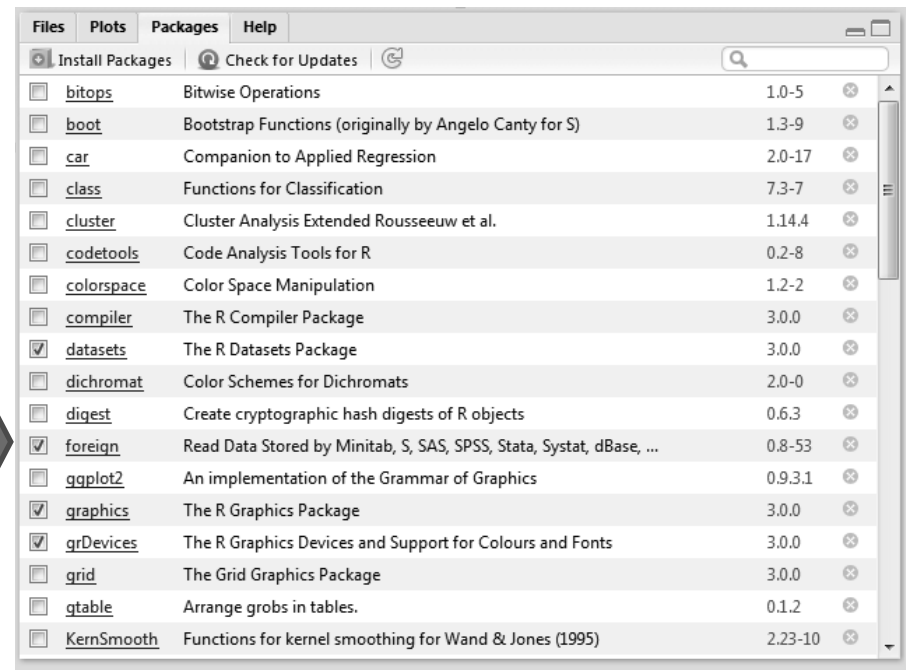
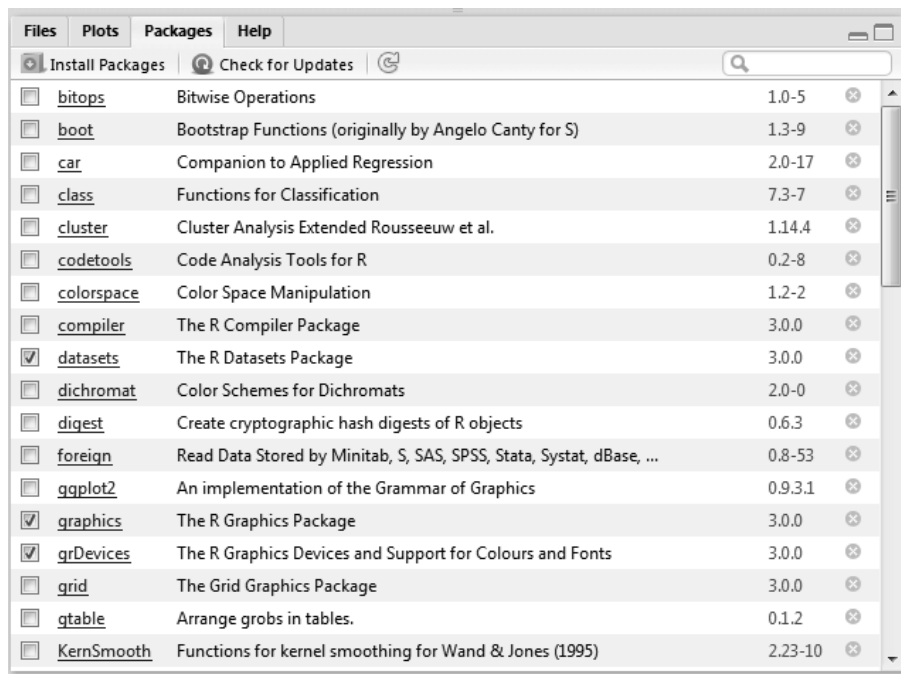
To create a new R script you can either go to `File -> New -> R Script`, or click on the icon with the “+” sign and select “R Script”, or simply press `Ctrl+Shift+N`. Make sure to save the script.





# Packages tab

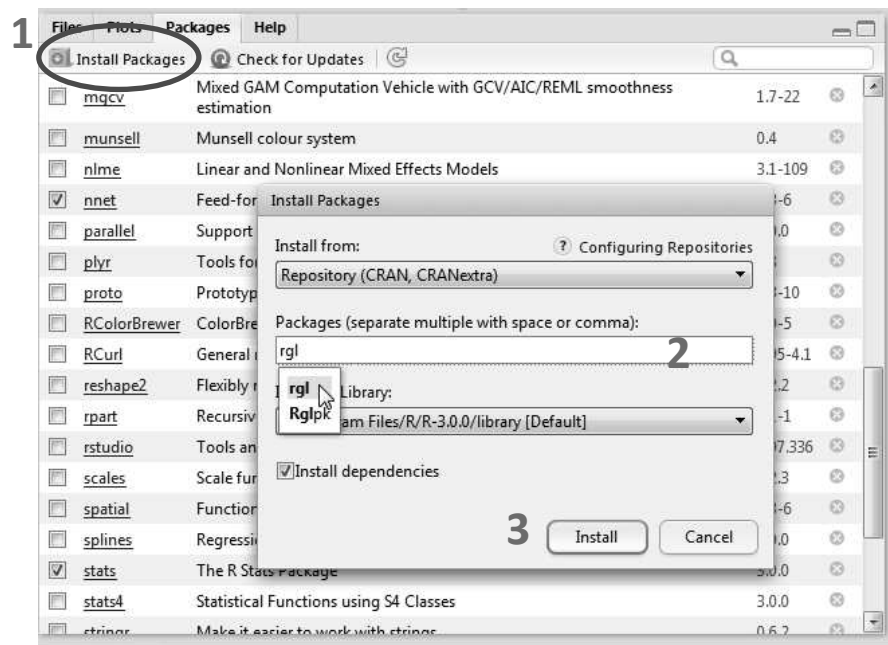
The package tab shows the list of add-ons included in the installation of RStudio. If checked, the package is loaded into R, if not, any command related to that package won't work, you will need select it. You can also install other add-ons by clicking on the 'Install Packages' icon. Another way to activate a package is by typing, for example, `library(foreign)`. This will automatically check the `--foreign` package (it helps bring data from proprietary formats like Stata, SAS or SPSS).



# Installing a package

<input type="checkbox"/>	<u>R</u> Curl	General network (HTTP/FTP/...) client interface for R	1.95-4.1	✕
<input type="checkbox"/>	<u>reshape2</u>	Flexibly reshape data: a reboot of the reshape package.	1.2.2	✕
<input type="checkbox"/>	<u>r</u> part	Recursive Partitioning	4.1-1	✕

**Before**



We are going to install the package – `rgl` (useful to plot 3D images). It does not come with the original R install.

Click on “Install Packages”, write the name in the pop-up window and click on “Install”.

**After**

<input type="checkbox"/>	<u>R</u> Curl	General network (HTTP/FTP/...) client interface for R	1.95-4.1	✕
<input type="checkbox"/>	<u>reshape2</u>	Flexibly reshape data: a reboot of the reshape package.	1.2.2	✕
<input type="checkbox"/>	<u>rgl</u>	3D visualization device system (OpenGL)	0.93.952	✕
<input type="checkbox"/>	<u>r</u> part	Recursive Partitioning	4.1-1 12	✕