Lab 1: R language basics

Univariate Statistics and Methodology using R 2018-19

Welcome to Univariate!

Help with installation

The aim of this first lab is to help you get started with the R statistical programming language. So, first, let us know if you had any installation problem with:

- **R**, the programming language core
- **RStudio**, the graphical user interface for programming in R and
- **Swirls**, little interactive R lessons (not compulsory, but can be useful to practice)

The installation instructions are on Learn, but you can also check the Navarro book, Part II, chapter 3.1. If you have not installed on your laptop yet, try it now.

How to run R code

Now that R is installed, you can practice creating your first R project and writing your own first R script. A script is simply a text file, where you write some R commands that the R engine then executes for you. Again, instructions on how to run R code are on Learn and at the end of this lab sheet.

R Keyboard Shortcuts

Now it is time to practice, adding other commands to that first script!

Go the Navarro book, part II, and browse through sections 3.2-3.12 for examples of simple arithmetical commands. Put them in your script and practice completing them using the **Tab button** (your best friend…) and executing them with CTRL+Enter (Windows) or CMD+Enter (Mac). Try to practice all the other essential shortcuts too. You can find the shortcuts cheat sheets on Learn and at the end of this lab sheet. Print them and keep them somewhere handy.

R language basics

And now for some basic R. Go to the Navarro book, part II, section 4.1 to learn how to add **comments** to your script.

You can leave sections 4.2-4.5 for now, as we will not load extra packages or files today. However, they are important, so make sure you go back to these instructions during Lab 2 on data manipulation.

Now go to section 4.6 and onwards, and familiarise yourself with

1. *variables*
2. *factors*
3. *data frames*
4. *lists*
5. *formulas*
6.

Any doubt or question, raise your hand and a tutor will come to you! Enjoy

Keyboard shortcuts & tricks: **Mac OS**

Shortcuts work in most programs: Word, Excel, your browser, email editors!
Using your keyboard instead of the mouse will save you hundreds of hours

Generic shortcuts

In Windows, usually you can use the CTRL button instead of cmd ⌘

cmd ⌘ + C to COPY text **cmd ⌘ + V** to PASTE text **cmd ⌘ + X** to CUT text	**cmd ⌘ + Z** to UNDO the last action **cmd ⌘ + Y** to REDO the last action
HOME to go to the beginning of the line **cmd ⌘ + HOME** to go the beginning of the document (cmd + arrow up in Mac)	**END** to go to the end of the line **cmd ⌘ + END** to go the end of the document (cmd + arrow down in Mac)

RStudio-specific shortcuts

ESC to get back the console prompt	**Up arrow** to call back the commands you just typed at the console (and to edit them)
Tab to autocomplete function names, function parameters etc	**cmd ⌘ ^ 1** or View -> Move focus to source to get to the scripting window (upper left pane)

When your cursor is in the scripting window (upper left) you can use these shortcuts to **run code**:

- **cmd ⌘ + Enter** will run the current text line, or the lines selected
- **cmd ⌘ + SHIFT + S** will run (source) all the code in your script
- **ALT + 3** will type a hash symbol **# To insert comments in code**

If you flag "Source on save" all the code in your script (all the text you typed in the scripting area) will be run when you press: **cmd ⌘ + S** to save

cmd ⌘ + S saves a .R script if you are in RStudio, or the Word document if you are in Word, or the spreadsheet if you are in Excel, you get the idea)

cmd ⌘ + TAB to move between windows (to jump to the next program window)

Web browser short cuts

cmd ⌘ + left arrow Go back to the previous page / URL

cmd ⌘ + right arrow Go forward to the next page / URL

Keyboard shortcuts & tricks: **Windows** & some Linux OS

Shortcuts work in most programs: Word, Excel, your browser, email editors!
Using your keyboard instead of the mouse will save you hundreds of hours

Generic shortcuts

In MAC, usually you can use the cmd ⌘button instead of CTRL.

CTRL + C to COPY text **CTRL + V** to PASTE text **CTRL + X** to CUT text	**CTRL + Z** to UNDO the last action **CTRL + Y** to REDO the last action
HOME to go to the beginning of the line **CTRL + HOME** to go the beginning of the document (cmd + arrow up in Mac)	**END** to go to the end of the line **CTRL + END** to go the end of the document (cmd + arrow down in Mac)

RStudio-specific shortcuts

ESC to get back the console prompt	**Up arrow** to call back the commands you just typed at the console (and to edit them)
Tab to autocomplete function names, function parameters etc	**CTRL + 1** or View > Move focus to source to get to the scripting window (upper left pane)

When your cursor is in the scripting window (upper left) you can use these shortcuts to **run code**:

- **CTRL + Enter** will run the current text line, or the lines selected
- **CTRL + SHIFT + S** will run (source) all the code in your script

If you flag “Source on save” all the code in your script (all the text you typed in the scripting area) will be run when you press: **CTRL + S** to save

CTRL + S saves a .R script if you are in RStudio, or the Word document if you are in Word, or the spreadsheet if you are in Excel, you get the idea)

ALT + TAB to move between windows (to jump to the next program window)

Web browser short cuts

BACKSPACE Go back to the previous page / URL

CTRL + right arrow Go forward to the next page / URL

Appendix 1: How to run code in R

Adapted from: R For Dummies, by Andrie de Vries, Joris Meys

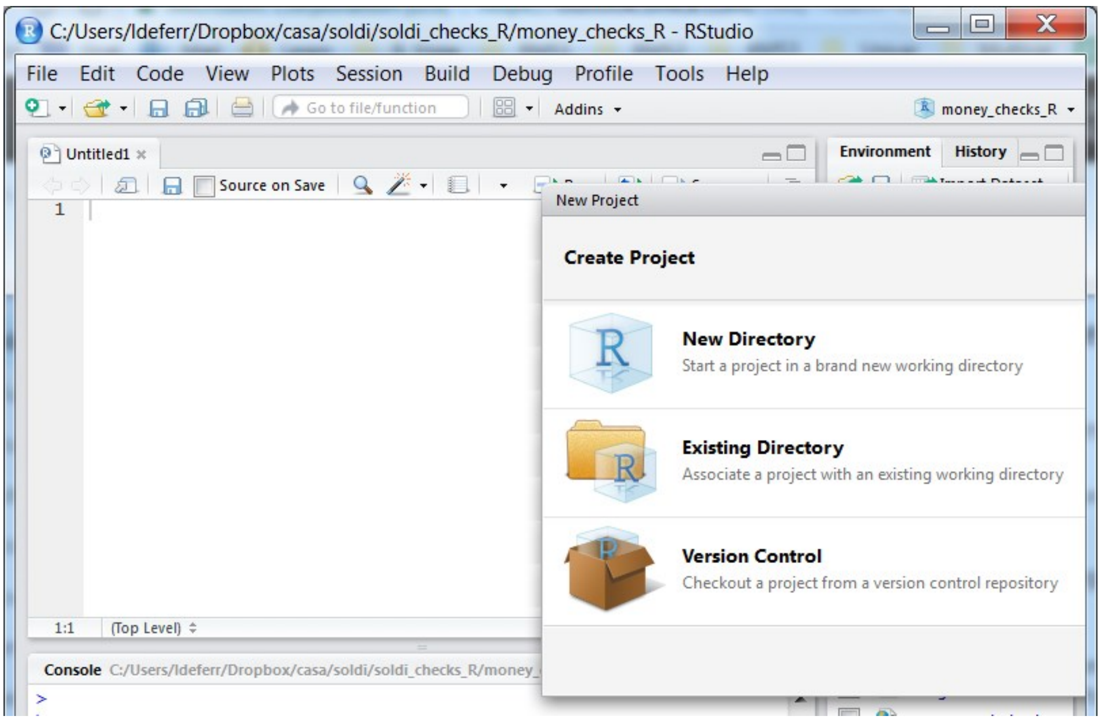
Doing statistics also means learning how to organise your files appropriately and how to work with your R code in a professional and efficient manner.

Create a new project

Create a new project for each lab or major assignment. This allows you to work in a little environment where all your preferences, packages etc. are already loaded and ready to use.

When you have more experience, you may occasionally decide to create several scripts in the same project, if those scripts share the same environment, use similar packages, or they need to "call" each other. However, for now, please create a new project for each lab.

In the RStudio top menu, click on File > New Project > New Directory > Empty Project.


C:/Users/ldeferr/Dropbox/casa/soldi/soldi_checks_R/money_checks_R - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
money_checks_R
Untitled1
Source on Save
Environment
History
1:1
(Top Level)
Console C:/Users/ldeferr/Dropbox/casa/soldi/soldi_checks_R/money_

New Project

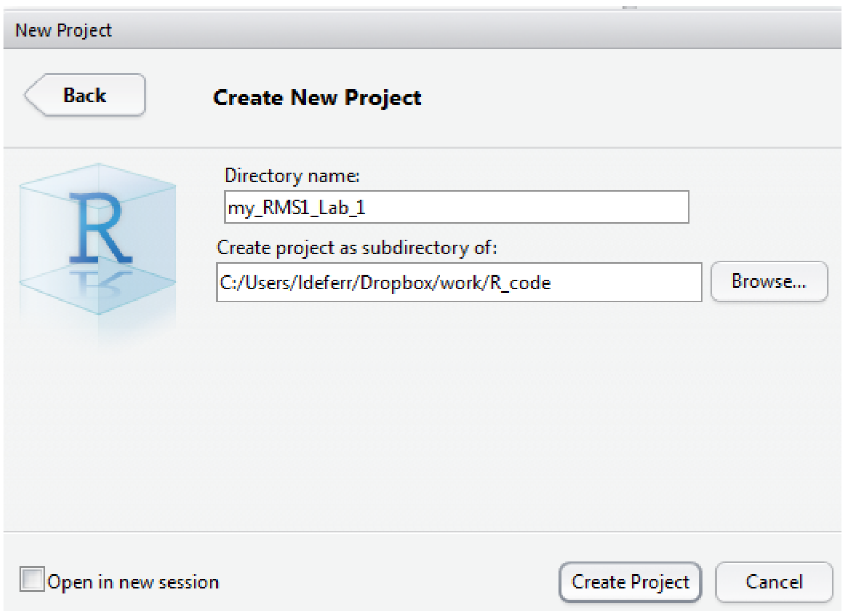
Create Project

New Directory
Start a project in a brand new working directory

Existing Directory
Associate a project with an existing working directory

Version Control
Checkout a project from a version control repository

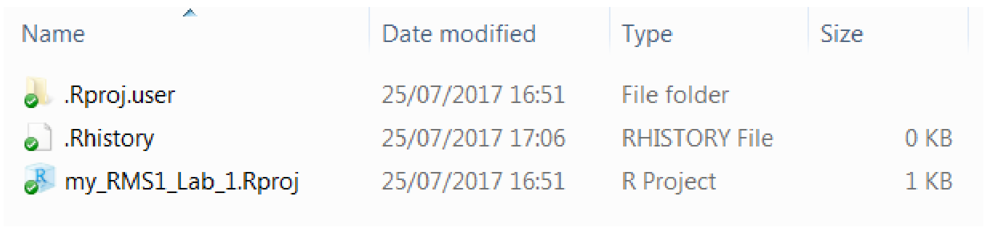
Choose a name for your project, such as *my_RMS1_Lab1* or similar. Click on Browse and pick a directory where to save the project. Something like my_uni_stuff/RMS1/labs or similar, depending on how you like to name your folders.



Directory name: my_RMS1_Lab_1

Create project as subdirectory of: C:/Users/ldeferr/Dropbox/work/R_code

Click on Create Project to create the project directory. RStudio will automatically save in that directory a few small files, such as .Rhistory, to remember about your environment and packages.



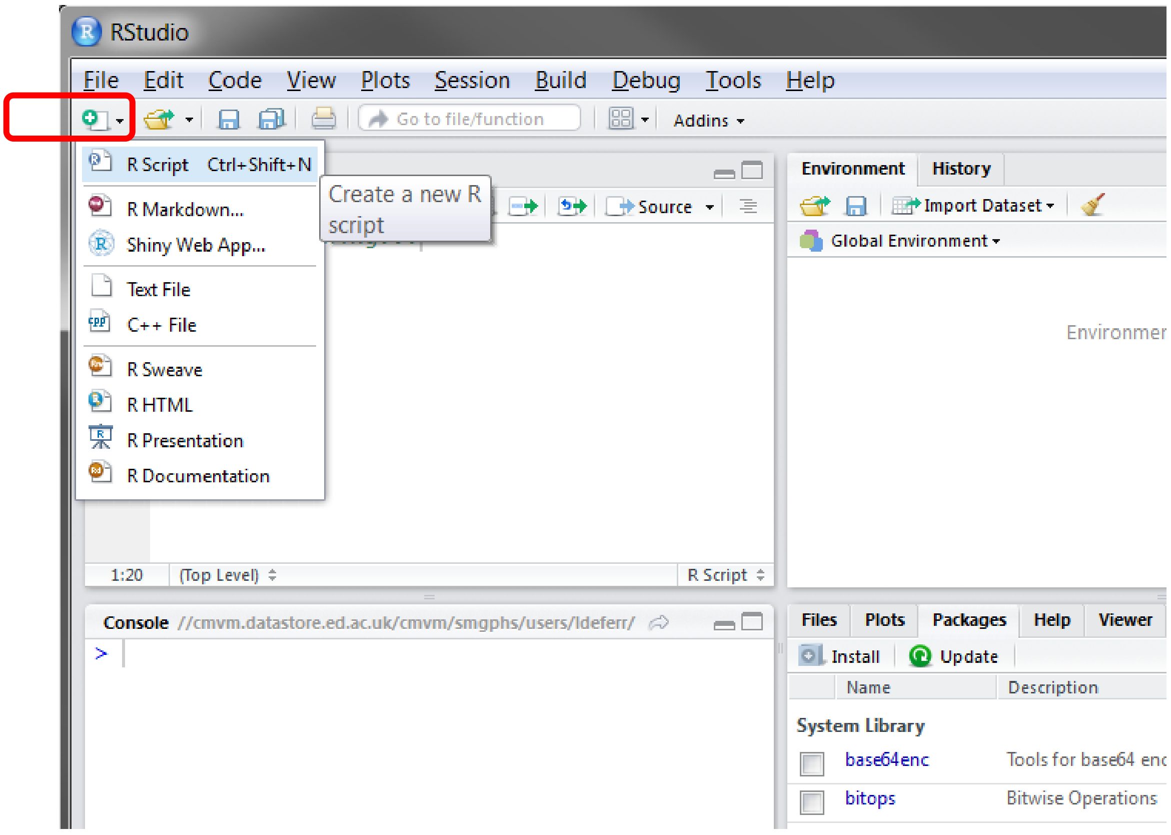
Name	Date modified	Type	Size
.Rproj.user	25/07/2017 16:51	File folder	
.Rhistory	25/07/2017 17:06	RHISTORY File	0 KB
my_RMS1_Lab_1.Rproj	25/07/2017 16:51	R Project	1 KB

If you are curious, have a look at those files before and after you save a script. Some files will look like gibberish to you, but the command history should be human-readable.

Create a new script

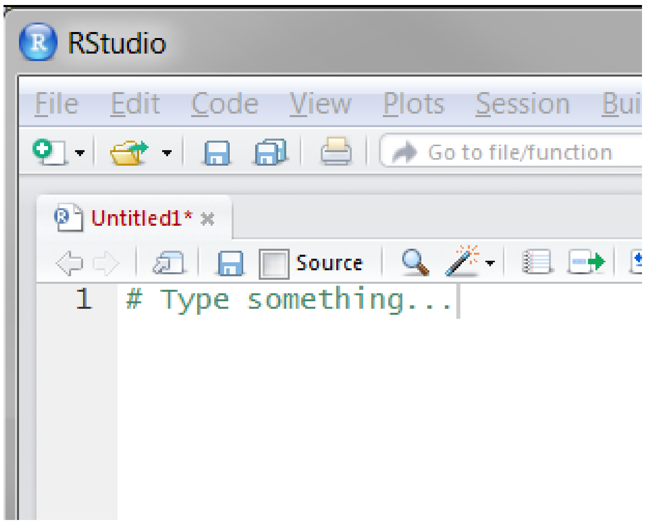
You need the upper left window of RStudio to write your first script in your first project (congratulations!). If you do not see the upper left window, click on View > Move Focus to Source.

Now click on the white icon with a green plus sign (add file) and then R Script. Or click on File > New File > R Script.

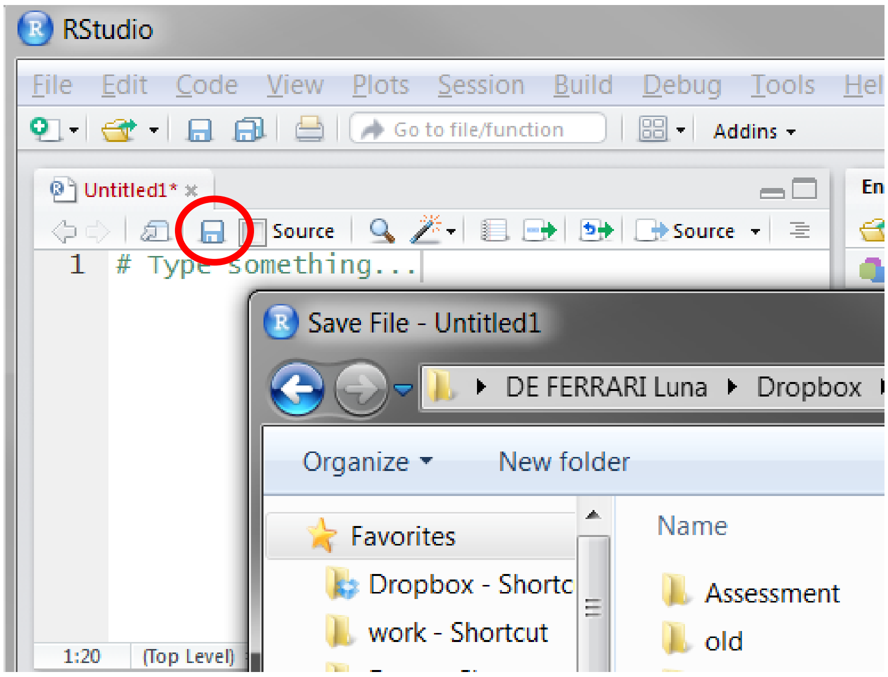


Save it with a name

This is not saved yet… you can see it because the file name is RED and with an asterisk *



Now save it with **CTRL+S** (or Cmd+S in Mac) or by pressing the Save icon which, incidentally, represents a relic few of you have ever held. A floppy disk.



Write a script command

Any text after a ***# symbol*** *is considered a* ***comment*** *by R and will not be interpreted as code (use ALT + 3 in Mac to type a #)*

Type or copy the following lines of code in the editor window:

```
# Lines starting with # are comments, not interpreted by R
# So you can write your notes here
x <- 'I love R'
y <- 'when it does what I want'

z <- paste(x, y)
# But what is now stored in the variable z? I cannot see it in the
console output... :(

# Put ( ) around an assignment to print the output to console
(z <- paste(x, y) )

# or use print() to print to console
print(z)
```

How to run your code

You have typed multiple lines of code into the source[1] editor without having each line evaluated by R. Now, when you're ready, you can send these instructions to R.

When you use RStudio, you can do this in one of four ways:

1. **Send an individual line of code from the editor to the console.** Click on the line of code you want to run, wherever you want, NO NEED TO HIGHLIGHT THE ENTIRE LINE! And then press **Ctrl+Enter** (or Cmd+Enter for Mac)

2. **Send a block of highlighted code to the console.** Select the WHOLE block of code you want to run, and then press **Ctrl+Enter** (or Cmd+Enter for Mac)

3. **Source the entire script.** In RStudio, click anywhere on the source editor and press **Ctrl+Shift+Enter** (Cmd+Shift+Enter).

Alternatively, you can click the *Source* **button**. But don't. Use your keyboard instead and you'll save hours and spare your wrists.

How to see the output

Sourced scripts behave differently from typing in the console regarding printing results. In interactive mode (console), each result is directly printed out and output to the console.

But when you source a script all in one go, output is printed only if you explicitly tell R to print. You can do so by using the `print()` function or by putting round brackets (parentheses) around the entire assignment command.

Why saving in a script? Can't I just type in the console?

- From a script, you can copy and paste examples of code into your assignment script
- You can add lines of personal notes and comments, using the hash # symbol
- With a saved script, tutors can see all your work, and help you find where a typo crept in
- After you have corrected that typo, you can re-run **in one go** the many lines of code that follow, to get your corrected results or plot
- Because that's what programmers do. After a few decades typing, they may have learned a thing or two about how to work with code.

[1] What does it mean to source? It's to tell R to perform several commands at once from a text file. Since RStudio calls the source() function to do it, R users refer to this process as sourcing a script.

To prepare your script to be sourced, you write the script in an editor window. In RStudio, the editor window is in the top-left corner of the screen.

Appendix 2: R and Swirl installation

For Research Methods and Statistics courses

`Begin at the beginning,' the King said gravely,`and go on till you come to the end: then stop.'

-- *Lewis Carroll, Alice's Adventures in Wonderland*

The world needs statistics. The Harvard Business Review defined 'data scientist' as the sexiest job of the 21st century, even if some may object to leaving the definition of sexiness to statisticians.

In short, we'll have plenty of work and we'll want to be able to reuse our analyses. Programming is worth learning in itself, as a dojo of logical thinking. But programming in R will save you time on the long run, allowing you to reuse and share your analyses.

It may even get you that job you really want, after graduation or for the summer.

Basics

R is a programming language. Like Java, Python, PHP or C. Ok, maybe not like C.

R is a bridge between you and your computer hardware. You can type at the R console something like:

mydata <- read.csv("C:/data/mydata.csv") and R will get the data out of the comma separated file on your disk. Every R instruction is just a line of text. An R computer program (a set of instructions) is just a text file, you can share it and edit it like text. But you can also execute it.

Microsoft Excel too will open a data file for you, but it involves clicking on a graphical interface, and it will not produce a set of instructions someone else -- or you at a later time -- will be able to reuse.

Also, Excel or SPSS are expensive, have only a limited set of statistical instructions and are difficult to program in. While R is free and has a mind boggling variety of statistical routines; mostly open source, which means used, reviewed and debugged by thousands of researchers worldwide.

Welcome to the R community!

R Studio & swirls in the 7 George square concourse lab

You can use R from a terminal window in most labs computers, but if you are in the 7 George square lower concourse lab, you can use the graphical interface (RStudio) to read, write and save programs.

Ensure that the lab machine is running **Windows** and not OSX. If the computer is already running OSX, reset the machine and select Windows when prompted. On Windows, the setup is easier and your packages will be installed on your M: network disk.

You should find RStudio already installed, somewhere in the "All programs" menu in Windows. Open RStudio and go to this document's section: *Try your first Swirl*. If you cannot find the right swirl course, follow the instructions in section: *Install the course swirls*. But please, do also install R and RStudio on your computer at home and your laptop for the labs.

Follow the path

A path in this context means a **file path**. A file path is a textual representation of where a file is located on your disk. Programs tend to hide paths from the user, but you are a programmer now, and you need to learn how paths work to fully control your computer and your data.

A typical path in a Windows operating system looks like this:

C:\Users\luna\work\Y2_Psychology_RMS1.zip

The disk name is **C:**

Inside the *C:* disk there is a directory (a folder) called *Users*.

Inside the directory *Users*, there is a directory called *luna* (which, in this example, is my user name). This is a series of directories (or folders) that the system created for me.

Inside the *luna* path I have then created a *work* directory to keep my files.

The **backslash** symbol \ is used to divide the directories names in Windows and form a path.

In the Mac and Linux operating systems, a path look like this:

/Users/luna/work/Y2_Psychology_RMS1.zip

Note that the **slash** symbol **/** is used, and not the backslash, to divide the directories name. There is no slash at the end of the path, because the last element is a file (a .zip file in this case). A path such as /Users/luna/work/ which ends with a slash, indicates a path to the *work* directory.

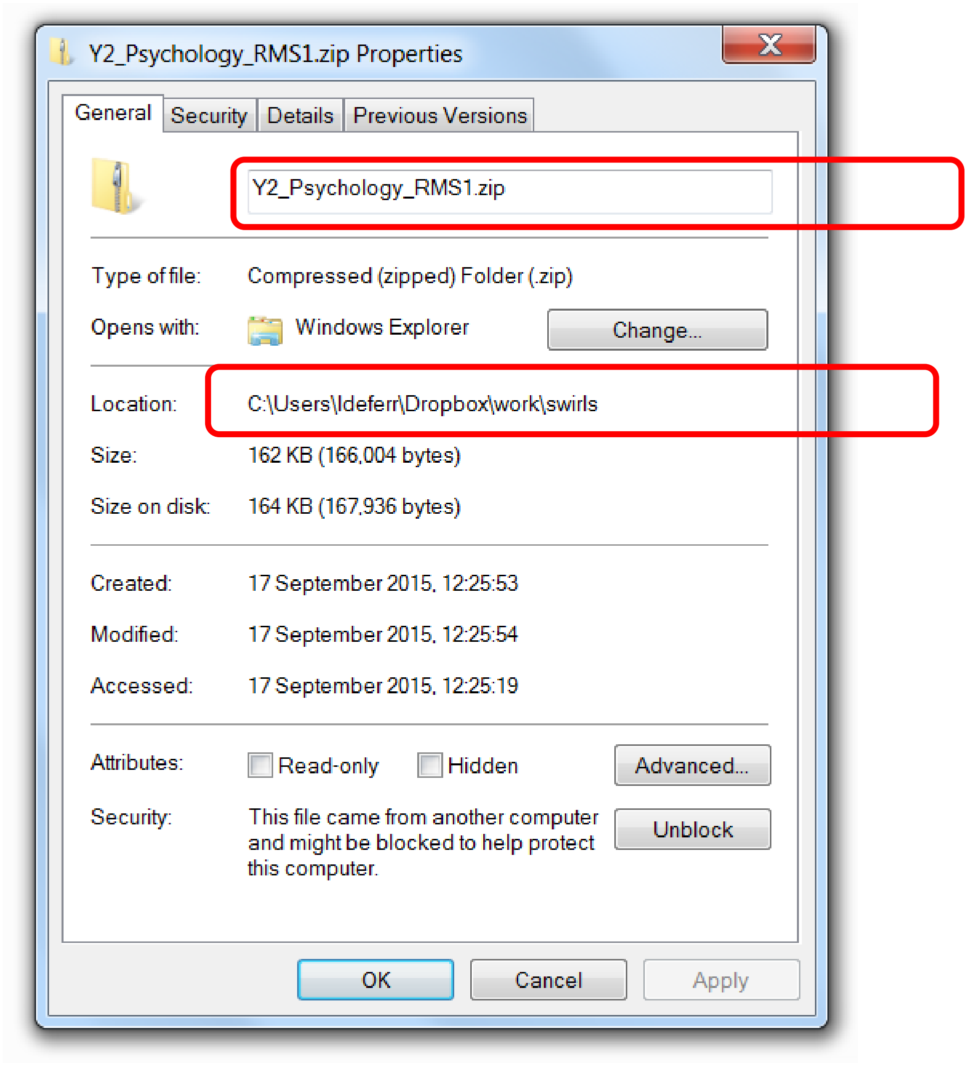
How to find a file path: Windows

Often operating systems try to hide paths from you. For example, they may save your email attachments or any file you save from your browser in a pre-set directory such as: C:\Users\luna\Downloads

You can change this in your browser or email program options.

You can find where a file really is by using the “search” box in your file manager, if you know part of the file name. Once you have found your file, you can see its path by right-clicking on the file icon and clicking on “Properties”. To compose the file path you will need to join the “Location” to the file name using a backslash.

See below:



Y2_Psychology_RMS1.zip

Type of file:	Compressed (zipped) Folder (.zip)
Opens with:	Windows Explorer
Location:	C:\Users\ldeferr\Dropbox\work\swirls
Size:	162 KB (166,004 bytes)
Size on disk:	164 KB (167,936 bytes)
Created:	17 September 2015, 12:25:53
Modified:	17 September 2015, 12:25:54
Accessed:	17 September 2015, 12:25:19
Attributes:	☐ Read-only ☐ Hidden
Security:	This file came from another computer and might be blocked to help protect this computer.

The full Windows path, in this case, will be:

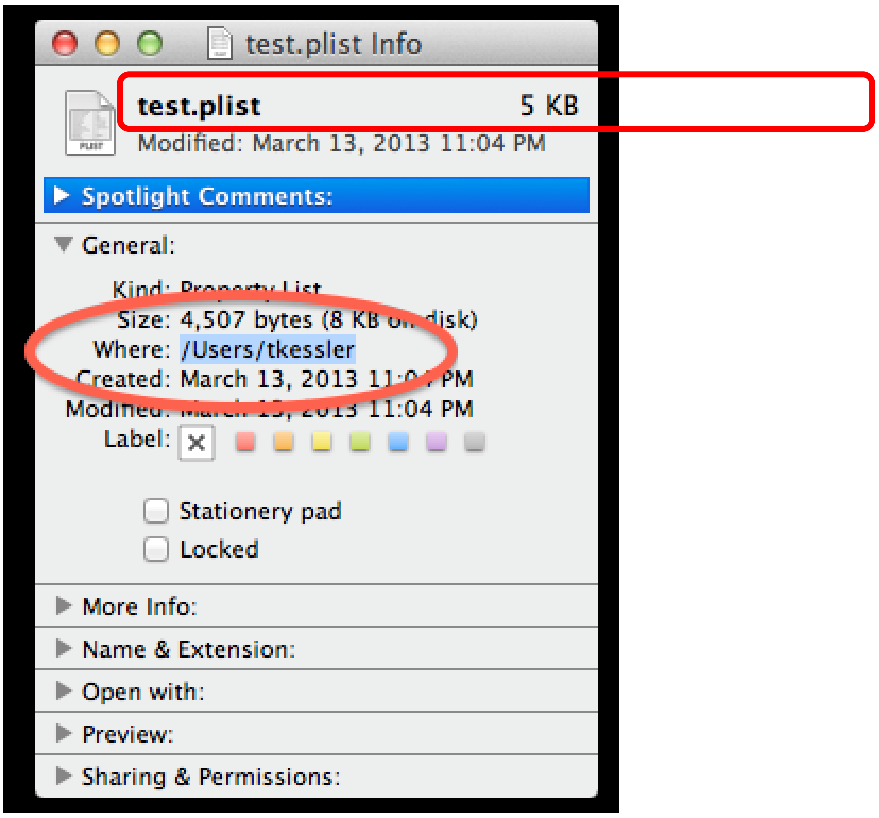
C:\Users\ldeferr\Dropbox\work\swirls\Y2_Psychology_RMS1\

However, R does not like backslashes (\ ), so you will have to re-write the path with slashes (**/**):

C:**/**Users**/**ldeferr**/**Dropbox**/**work**/**swirls**/**Y2_Psychology_RMS1**/**

How to find a file path: Mac

In some Mac releases you can right click on a file in Finder, click on Get Info > General > Where and you will find the directory path there. You will need to join it with the file name using a slash, in the example below: /Users/tkessler/test.plist



Another possibility is to open a terminal window.

You can find the terminal in Finder > Applications > Utilities > Terminal

Copy your file from the file manager or Finder (using cmd + C) and paste it into the Terminal (using cmd + V).

The path will appear in the terminal prompt, and you can highlight it and copy it from there (cmd + C)

For Mac and Linux, the similarity symbol ~ stands for the home directory.

For example: *~/Downloads* is equivalent in Mac to */Users/<your user name>/Downloads* and in Linux to */home/<your user name>/Downloads*

Install R & RStudio

If you have **brought your laptop to the lab (do!)**, or you are at home reading these lab instructions, check you have Internet connection and please install R and RStudio on your computer now:

For Windows

1. To install R go to https://cran.rstudio.com/ click on Download R for Windows, install R for the first time, Download R 3.2.1 for Windows (or similar) and click on save file.
2. Go where you have saved the R.exe file (possibly in C:\Users\<your user>\Downloads\) and double click on it.

3. Follow the installation instructions, but check in which directory R will be installed by default, to make sure you have enough free space there.
4. Once R is installed, install RStudio from https://www.rstudio.com/products/rstudio/download/ choosing RStudio [version number] - Windows Vista/7/8. At the time of writing the current version was https://download1.rstudio.org/RStudio-0.99.467.exe.
5. As before, double click on the .exe file and follow the instructions.
6. You should now have an RStudio entry in your ‘All programs’ menu.
7. Why not dragging the icon to the desktop menu bar for quicker access?
8. Open RStudio and go to Tools > Global Options > Appearance. We will be working with special characters and quotation marks, so make the Font Size at least 11 or 12 points.

For Mac

1. To install R go to https://cran.r-project.org/bin/macosx/ and follow the instructions for your operating system version.
2. To find out your Mac OS X version: from the Apple menu, choose About This Mac. The version number of OS X you're using appears directly below the words "OS X."
3. Save the .pkg file, click on it if needed, and follow the installation instructions, but check in which directory R will be installed by default, to make sure you have enough free space there.
4. Once R is installed, to install RStudio go to https://www.rstudio.com/products/rstudio/download/ and choose the current version for Mac, at the time of writing this was: RStudio 0.99.467 - Mac OS X 10.6+ (64-bit) .
5. As before, double click on the downloaded file and follow the instructions.
6. You should now have an RStudio icons in your programs list. Drag the icon to the desktop menu bar for quicker access.
7. Open RStudio and go to Tools > Global Options > Appearance. We will be working with special characters and quotation marks, so make the Font Size at least 11 or 12 points.

Check the graphics & install libraries

Type at the RStudio console:

```
demo("graphics")
```

Press enter, you should see a fancy plot appear in the “Plots” tab in RStudio. Keep pressing Enter.

For the Univariate Master course only, install the following packages:

- lme4
- psych
- reshape

```
install.packages("lme4")
install.packages("psych")
install.packages("reshape")
```

Install Swirl

Do this Once

The instructions in this section relate to installing two important packages into R and only need to be completed **once** on your machine. The next section describes what you should do each time you want to start Swirl in order to get the newest version.

Install devtools

In order to install the package we created ("Eddy") you first need to install the *devtools* package. *Devtools* allows you to install packages from sources other than those hosted on the R server.

To install *devtools* type in the RStudio console (beware on some computers it may take up to a couple of minutes!):

```
install.packages("devtools")
```

Install Eddy

Now that *devtools* is installed we can use it to install a package from our UoE Github account - type the following command:

```
> devtools::install_github("UoE-Psychology/eddy")
```

Ignore any error messages that relate to 'Rtools'. It will tell you it is required but it actually isn't for the basic package we have installed. You should see something like this if the installation was successful:

```
* installing *source* package 'eddy' ...
** R
** preparing package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
*** arch - i386
*** arch - x64
* DONE (eddy)
1
```

Run Eddy

Now that the eddy package is installed, we need to run the eddy() function which is inside that package. To do this, type the following command:

```
> eddy::eddy()
```

This will do several things – it will:

a) check your internet connection.

b) install and load the swirl package.

c) download and install the UoE-Psych swirl course which have been written specially to guide you through the statistics courses at University of Edinburgh.

If it is all successful, you should see the following message:

```
> | Hi! Type swirl() when you are ready to begin.
```

Typing `swirl()`will take you into the swirl environment.

Try your first Swirl lesson

1. Load the swirl package by typing at the console (you don't need to type > it's just to indicate you should type the command in the RStudio console window):
   ```
   > library("swirl")
   ```
2. Type **`swirl()`** at the console
3. Follow the instructions at the console and input a nickname of choice. If you have to quit, swirl will remember your name when you log in again, and will bring you back to your last swirl exercise.
4. Select course 'UoE-Psych'
5. Input a **nickname**. Keep it short. Swirl will remember the last lesson you were working on, if you want to restart there. Swirl (and R) distinguish lower from upper case, so always use: *Kate* or *kate*, not both.
6. Select 'Lesson 1 Arithmetic'. And feel free to shout 'BINGO' really loudly when you complete it (if you are in a quiet lab, you can whisper it).
7. And remember, no pirate jokes.
8. To remove your user history, such as your last location on a lesson, type: `delete_progress(user = "my user nickname")`

Material adapted from:

- ***R:*** *R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/*
- ***Swirl****: Nick Carchedi, Bill Bauer, Gina Grdina and Sean Kross (). swirl: Learn R, in R. R package version 2.3.1. http://swirlstats.com*
- ***RStudio****: RStudio Team (2016). RStudio: Integrated Development for R. RStudio, Inc., Boston, MA URL http://www.rstudio.com/*