RStudio Practical

Kevin Keenan

30 April 2015

Introduction

In this practical we will set up our system, ensuring we have the latest version of both R and RStudio installed. We will also make sure that we have a number of other tools available so that we can generate documents and build packages where necessary later on in the workshop.

Installing R

R v3.2.0 can be downloaded from the CRAN website. **Make sure you choose the correct installation for your operating system**.

Installing RStudio

Once R has been successfully installed, we can download RStudio (preview release) from http://www.rstudio.com/products/rstudio/download/preview/

RStudio Demo

Installing additional tools

In some instances it may be necessary to install an package from source. To do this we will need to install a LaTeXengine and some additional tools.

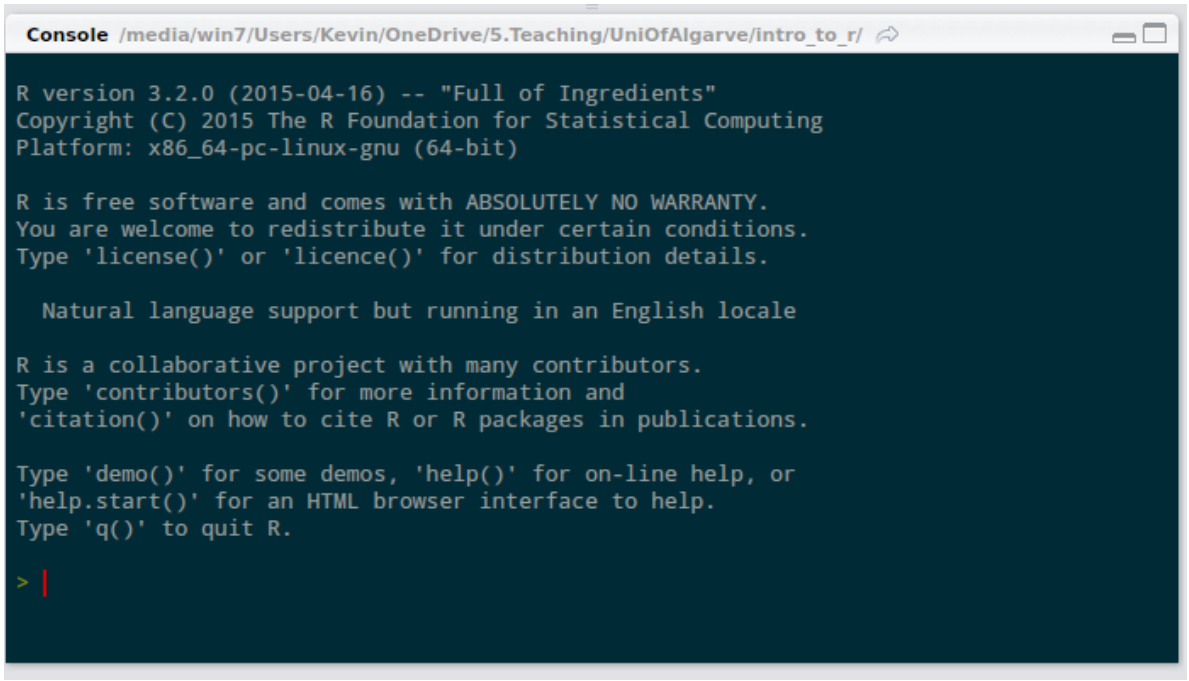
Instructions on how to install these tools on your specific operating system can be found at http://www.rstudio.com/products/rpackages/devtools/

We will also need to install the package `devtools`, which contains a number of helpful functions that we may need to use later. The code below can be used to install this package. Don't worry too much about the code for now. We will discuss package installation soon.

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

Let's make sure everything is working

1. Launch RStudio You should see the following in your console panel if R is installed correctly



This is known as the R console, and is the place that you can interactively type R code (i.e. results are returned immediately)

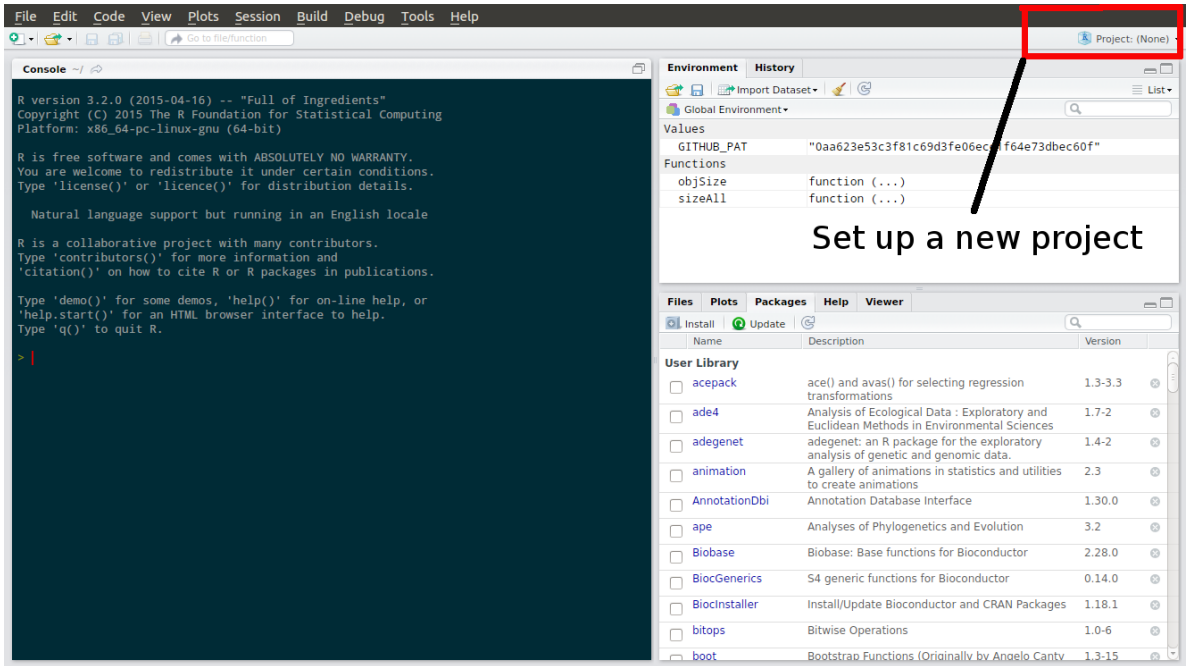
Projects in RStudio

2. Setting up an RStudio project
   - It is a very good idea to work with RStudio projects since they provide a useful way to keep your code organised and self contained.
   - Projects also make dealing with directories much more straightforward, virtually automating this process.
   - Throughout the workshop we will be setting up projects for different analysis methods.
   - The best (though not essential) style for project set up is:

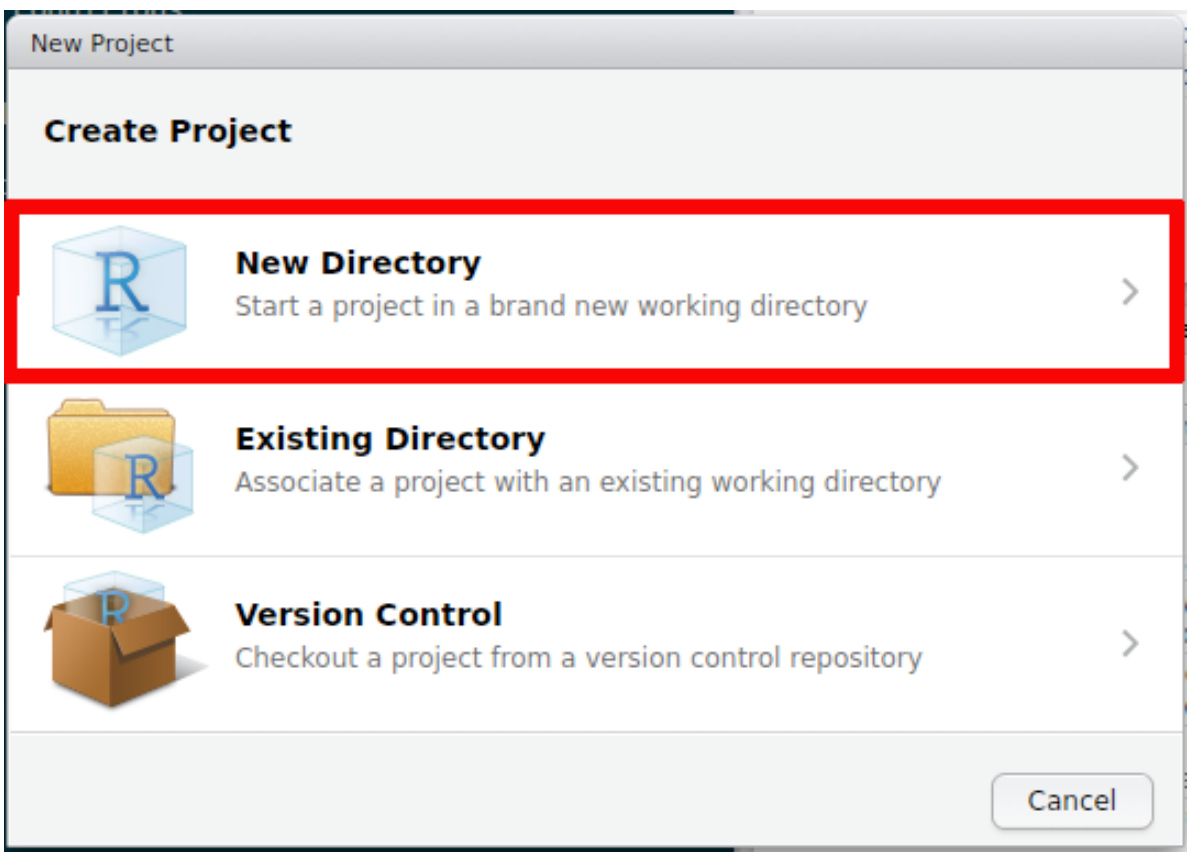
```
Working directory
                 | data
                 | code
                 | results
```

This means that we have a master folder for our project, which contains three other folders where we keep our *code*, *data* and *results*. It is also possible to include other folders, such as *manuscript* or *misc* (for junk etc.).

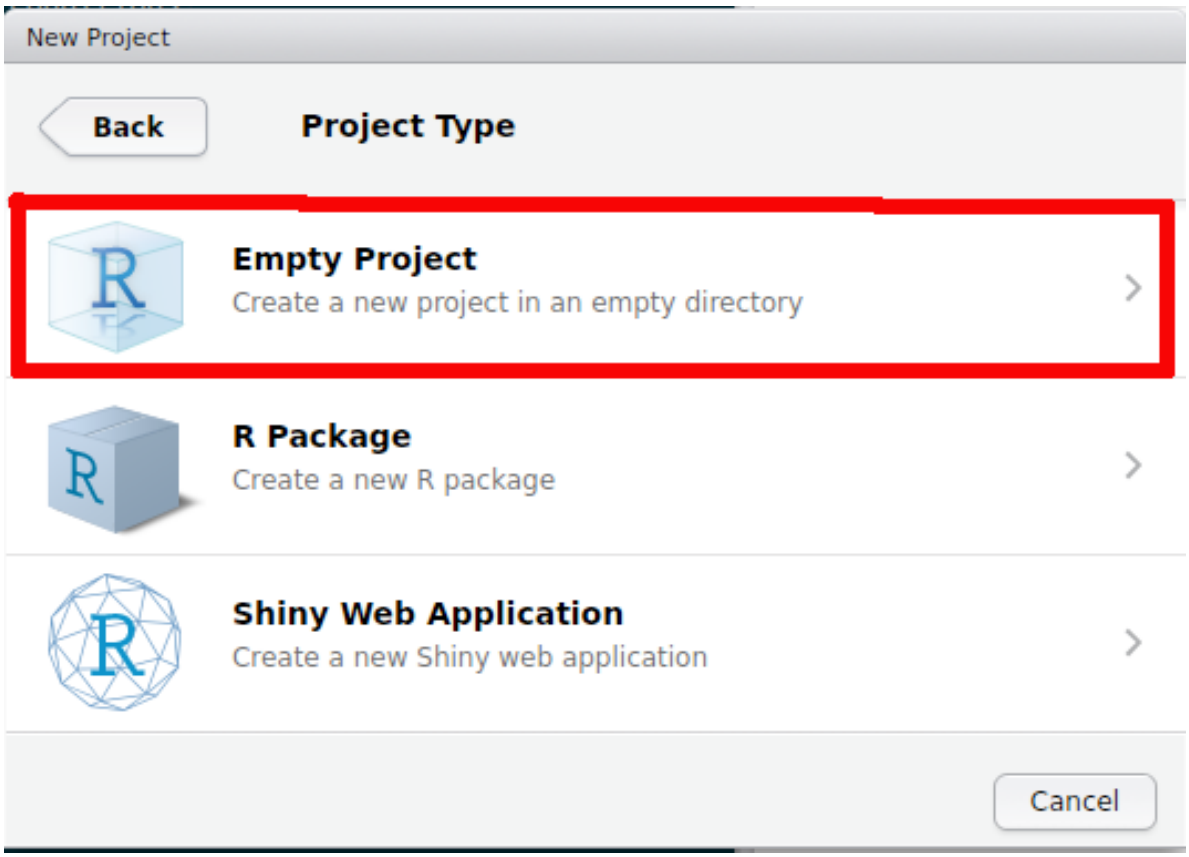
- Although we are not quite read to set up this style of project, we will begin by setting up a simpler one for this afternoon's practical
- Click on the drop-down menu highlighted in red below:



- Choose the "New Project" option
- The following dialogue window should appear:



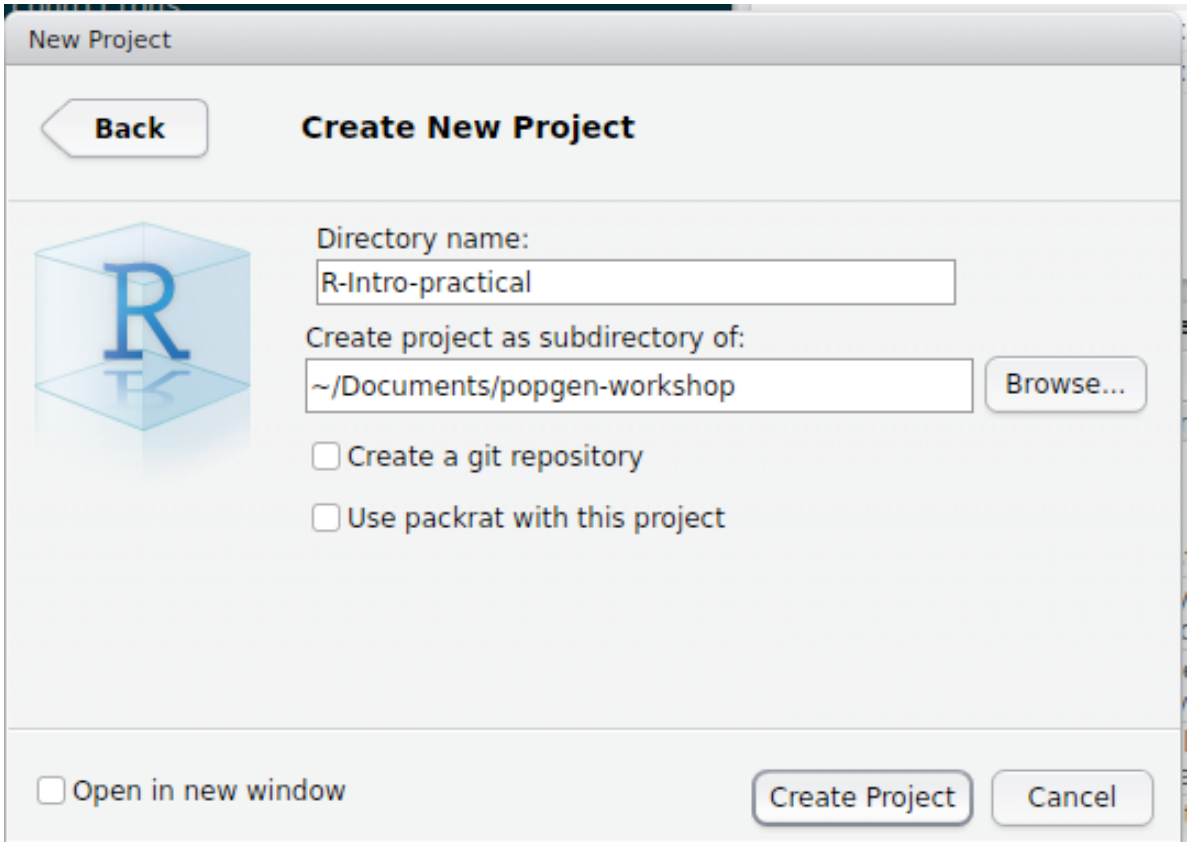
- Choose the option highlighted by the red box
- Choose the next option highlighted by the red box



- Now, using the "Browse…" button, locate a folder on your system where you would like to keep all of the projects you will generate during this workshop. For example, on my system I might have a folder like the one below:

```
/kevin/documents/popgen-workshop
```

- Once you have selected an appropriate folder, type the following name into the "Directory name:" box:



This will result in a new folder being created within your `popgen-workshop` folder. Within the `R-intro-practical` folder, a number of files may be created. We can ignore these as they will only be used by RStudio to keep your new project organised.

3. Manually creating our preferred project structure
   - As mentioned above, my recommendation for ensuring long-term consistency in analysis, especially for publications, it is a good idea to structure your project with some manually created folders.
   - In the `Files` tab within RStudio, you will see an option for *New Folder*. Using this, create three folders named **code**, **results** and **data**.
   - Now our project is set up, ready for this afternoon's practical.
4. Opening an existing project
   - Close RStudio and locate the `R-intro-practical` folder on your system.
   - Inside this folder, you will see a file named `R-intro-practical.Rproj`.
   - Double click it!

Back to the lecture for now