R & Python Installation Guide for Financial Modeling

Matthew Son

Last Updated : 2022-06-29

Contents

1 Download R & RStudio

Let's download R and RStudio for our IDE[1]. R (RStudio) is powerful software for programming and statistical analysis, and it integrates with Python seamlessly. This setting will help us to transit smoothly from Python to R or vise versa when we need to do regression analysis on R in the later part of the course. In my research experience, RStudio is one of the most convenient and intuitive IDE even for Python workflows.

This setup method might be a bit technical, but I'll provide much details so that everyone can follow.

1. Let's navigate to the R CRAN webpage to download R client. Please download most recent R client that matches your OS.

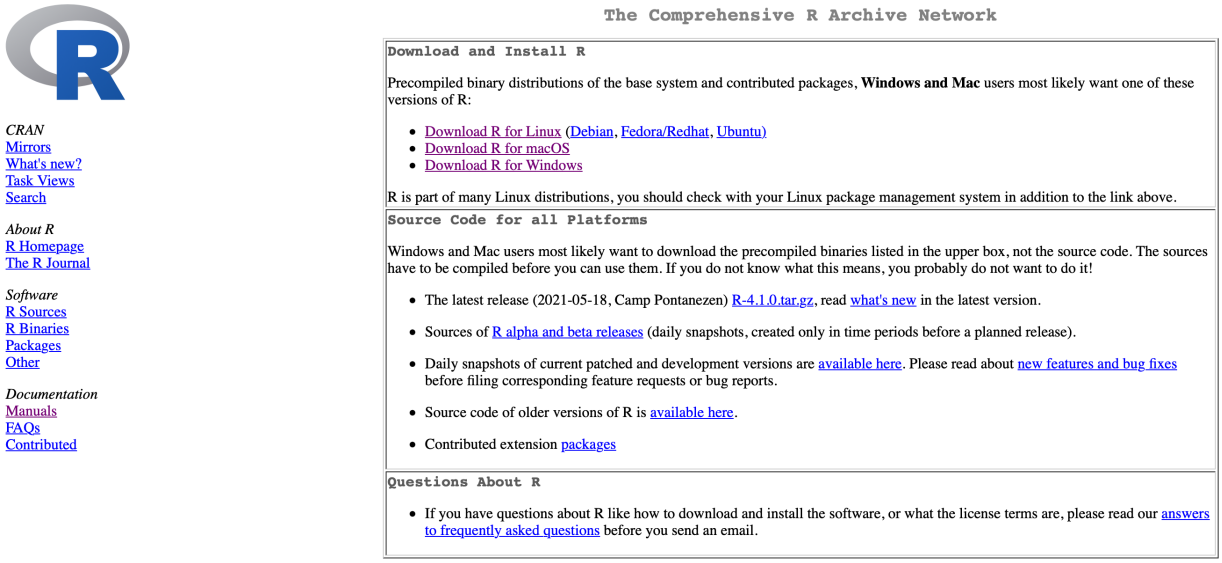

Figure 1: R CRAN Homepage

[1]IDE stands for Integrated Development Environment, which gives us nice user interface and convenient features to work on programming. RStudio is and IDE built for R, but it also integrates Python as well. Other famous Python IDEs include :VSCode, PyCharm, Atom, Spyder and Jupyter Notebook. I'll be introducing how to install/launch Jupyter Notebook (Lab) later in the course for those who prefer Jupyter setup.

- For Windows, choose **'base'** subdirectories to download R.

R for Windows

Subdirectories:

base	Binaries for base distribution. This is what you want to **install R for the first time**.
contrib	Binaries of contributed CRAN packages (for R >= 3.4.x).
old contrib	Binaries of contributed CRAN packages for outdated versions of R (for R < 3.4.x).
Rtools	Tools to build R and R packages. 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 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.

Figure 2: Choose base for Windows

- For Mac, choose R-4.x.x.pkg file to download package for R installation. If your Mac is with Apple silicon (M1,M2, etc), choose R-4.x.x.-arm64.pkg instead.

R for macOS

This directory contains binaries for a base distribution and packages to run on macOS. Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the old directory.

Note: Although we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

Package binaries for R versions older than 3.2.0 are only available from the CRAN archive so users of such versions should adjust the CRAN mirror setting (`https://cran-archive.r-project.org`) accordingly.

R 4.2.0 "Vigorous Calisthenics" released on 2022/04/22

Please check the integrity of the downloaded package by checking the signature:
`pkgutil --check-signature R-4.2.0.pkg`
in the *Terminal* application. If Apple tools are not avaiable you can check the SHA1 checksum of the downloaded image:
`openssl sha1 R-4.2.0.pkg`

Latest release:

R-4.2.0.pkg (notarized and signed) SHA1-hash: 2a90fb8629e44f72f9d89d6a9bac9b71564587d7 (ca. 90MB) for Intel Macs	**R 4.2.0** binary for macOS 10.13 (**High Sierra**) and higher, **Intel 64-bit** build, signed and notarized package. Contains R 4.2.0 framework, R.app GUI 1.78 in 64-bit for Intel Macs, Tcl/Tk 8.6.6 X11 libraries and Texinfo 6.7. The latter two components are optional and can be ommitted when choosing "custom install", they are only needed if you want to use the `tcltk` R package or build package documentation from sources. Note: the use of X11 (including `tcltk`) requires XQuartz to be installed (version 2.7.11 or later) since it is no longer part of macOS. Always re-install XQuartz when upgrading your macOS to a new major version. This release supports Intel Macs, but it is also known to work using Rosetta2 on M1-based Macs. For native Apple silicon arm64 binary see below. **Important:** this release uses Xcode 12.4 and GNU Fortran 8.2. If you wish to compile R packages from sources, you may need to download GNU Fortran 8.2 - see the tools directory.
R-4.2.0-arm64.pkg (notarized and signed) SHA1-hash: ada2602d245164d316967d24f5482b58e2dfddff (ca. 89MB) for M1 Macs only!	**R 4.2.0** binary for macOS 11 (**Big Sur**) and higher, **Apple silicon arm64** build, signed and notarized package. Contains R 4.2.0 framework, R.app GUI 1.78 for Apple silicon Macs (M1 and higher), Tcl/Tk 8.6.12 X11 libraries and Texinfo 6.8. **Important: this version does NOT work on older Intel-based Macs.** Note: the use of X11 (including `tcltk`) requires XQuartz (version 2.8.1 or later). Always re-install XQuartz when upgrading your macOS to a new major version. This release uses Xcode 13.1 and experimental GNU Fortran 12 arm64 fork. If you wish to compile R packages which contain Fortran code, you may need to download GNU Fortran for arm64 from https://mac.R-project.org/tools. Any external libraries and tools are expected to live in `/opt/R/arm64` to not conflict with Intel-based software and this build will not use `/usr/local` to avoid such conflicts (see the tools page for more details).

Figure 3: Choose Intel(X64) or Apple Silicon(Arm64)

2. Next, navigate to RStudioIDE Stable Build to get the stable version of RStudio IDE. Download installers that matches your OS from `RStudio Desktop` section.

RStudio Desktop 1.4.1717 - Release Notes

1. Install R. RStudio requires R 3.0.1+.

2. Download RStudio Desktop. Recommended for your system:

DOWNLOAD RSTUDIO FOR MAC
1.4.1717 | 203.06MB

Requires macOS 10.14+ (64-bit)

All Installers

Linux users may need to import RStudio's public code-signing key prior to installation, depending on the operating system's security policy.

RStudio requires a 64-bit operating system. If you are on a 32 bit system, you can use an older version of RStudio.

OS	Download	Size	SHA-256
Windows 10	RStudio-1.4.1717.exe	156.18 MB	71b36e64
macOS 10.14+	RStudio-1.4.1717.dmg	203.06 MB	2cf2549d
Ubuntu 18/Debian 10	rstudio-1.4.1717-amd64.deb	122.51 MB	e27b2645

Figure 4: RStudio Desktop

2 Installation - Windows

2.1 R & RStudio

Installation of R & RStudio is pretty straightforward. It is advised not to change any default settings because it is likely to cost you a lot of time with path settings later. Please install R first then RStudio.

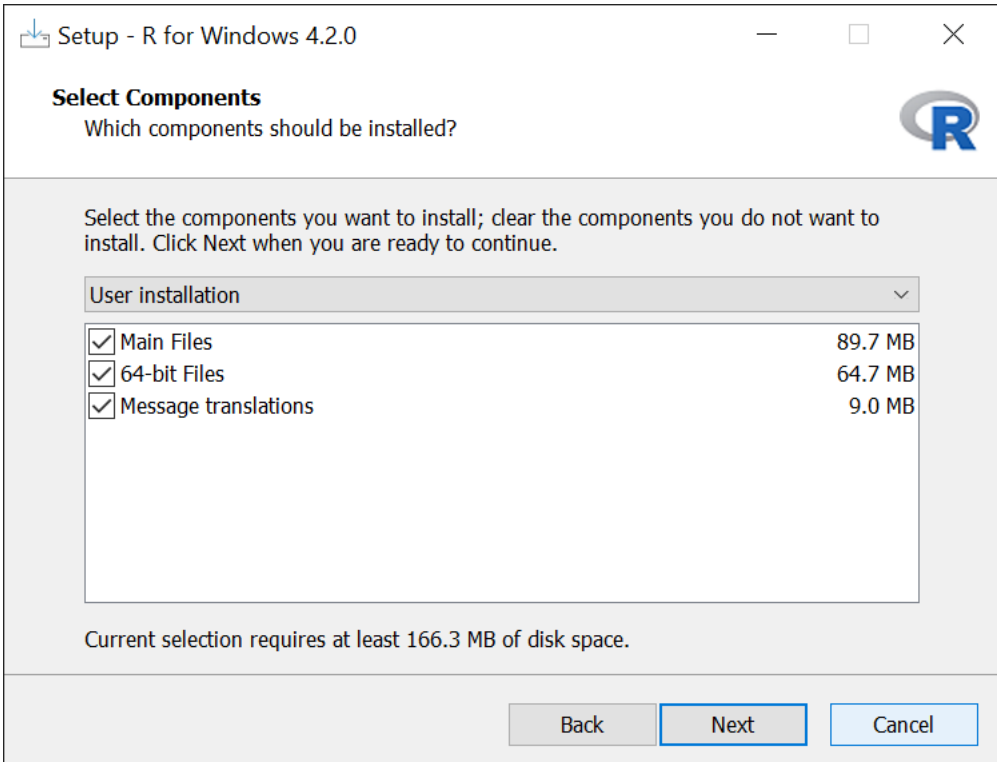

Figure 5: Windows R installation

If you have successfully installed R and Rstudio, congratulations, you will be seeing Figure 12 on your machine.

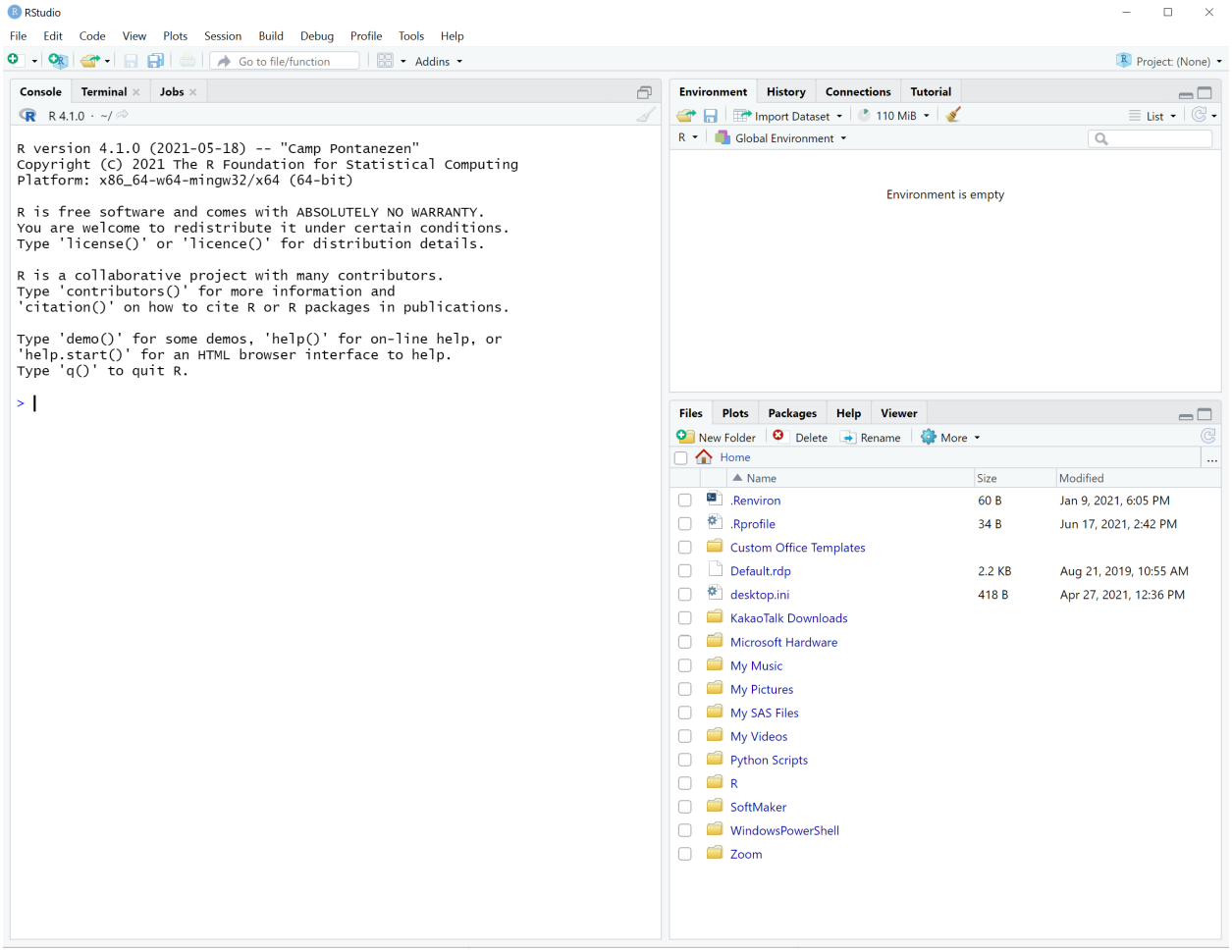

Figure 6: RStudio First Look

2.2 Python Installation

Now it is time to install python. We'll be installing python through an R Package called `reticulate`, which enables to run R and Python interchangeably with the power of RStudio IDE.

- Open a blank python script and click "run" button. It will try to install `reticulate` package on your machine.

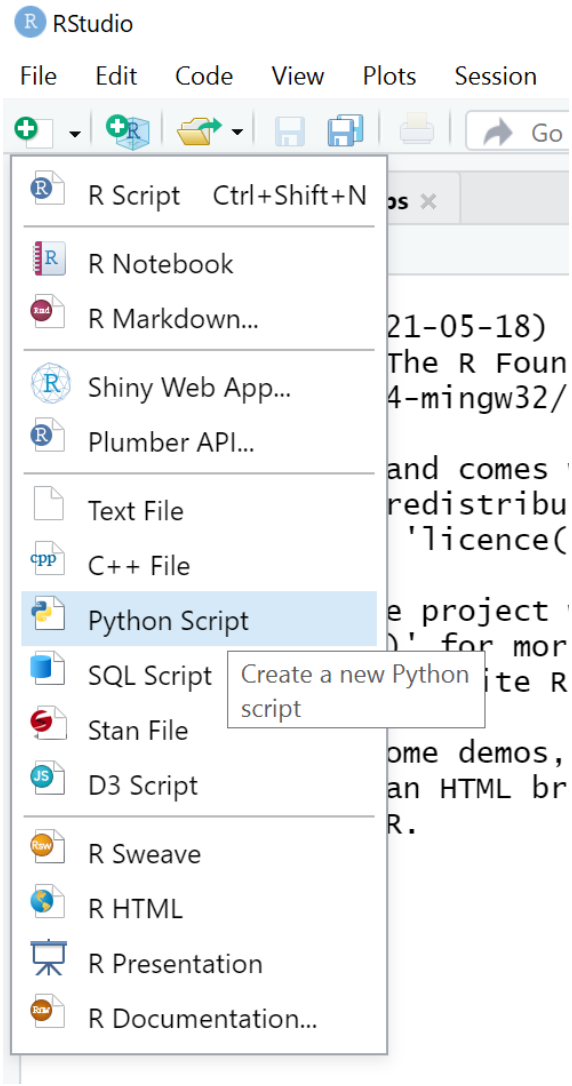

Figure 7: Create Python Script

- It will ask you to download necessary packages and will install miniconda[2] automatically. Answer yes to allow installations.
- If miniconda is not being installed automatically, then in the R Prompt type `reticulate::install_miniconda()`. Then follow all install suggestions and answer "Yes". It will download a Python conda distribution named "miniconda".

[2]This is a minimal and efficient version of Anaconda distribution. The standard Anaconda distribution comes with too much unnecessary packages.

If above installation worked without any trouble, then python installation is complete. Confirm your python installation by executing some simple commands.

- For Windows machine, you'll see `Anaconda Prompt` added in the start menu. From now on you can use this prompt (terminal) to install python packages and manage environments.

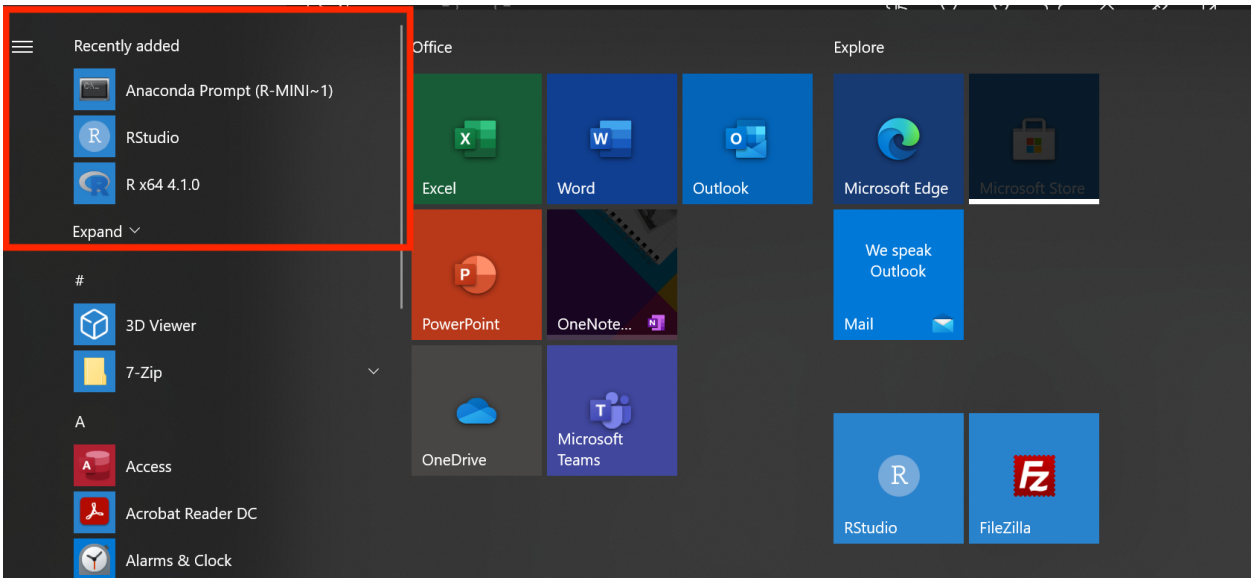

Figure 8: Windows: Anaconda Prompt generated on Start Menu

Figure 9: Windows: Miniconda Promopt

2.2.0.1 Set Python Path for Windows (Optional)

For Windows, copy paste below code on your command prompt. Then exit and re-launch RStudio.

```
echo RETICULATE_PYTHON = "%userprofile%\AppData\Local\r-miniconda\envs\r-reticulate\python.exe" >>
%userprofile%\Documents\.Renviron
```

1. Then open a python script (.py) as in figure 6 and run the empty code to activate python console.
2. Then Go to Tools > Global Options and set python interpreter for RStudio. Make sure you are choosing python from `conda environments` and choose python under `r-reticulate` environment as below.

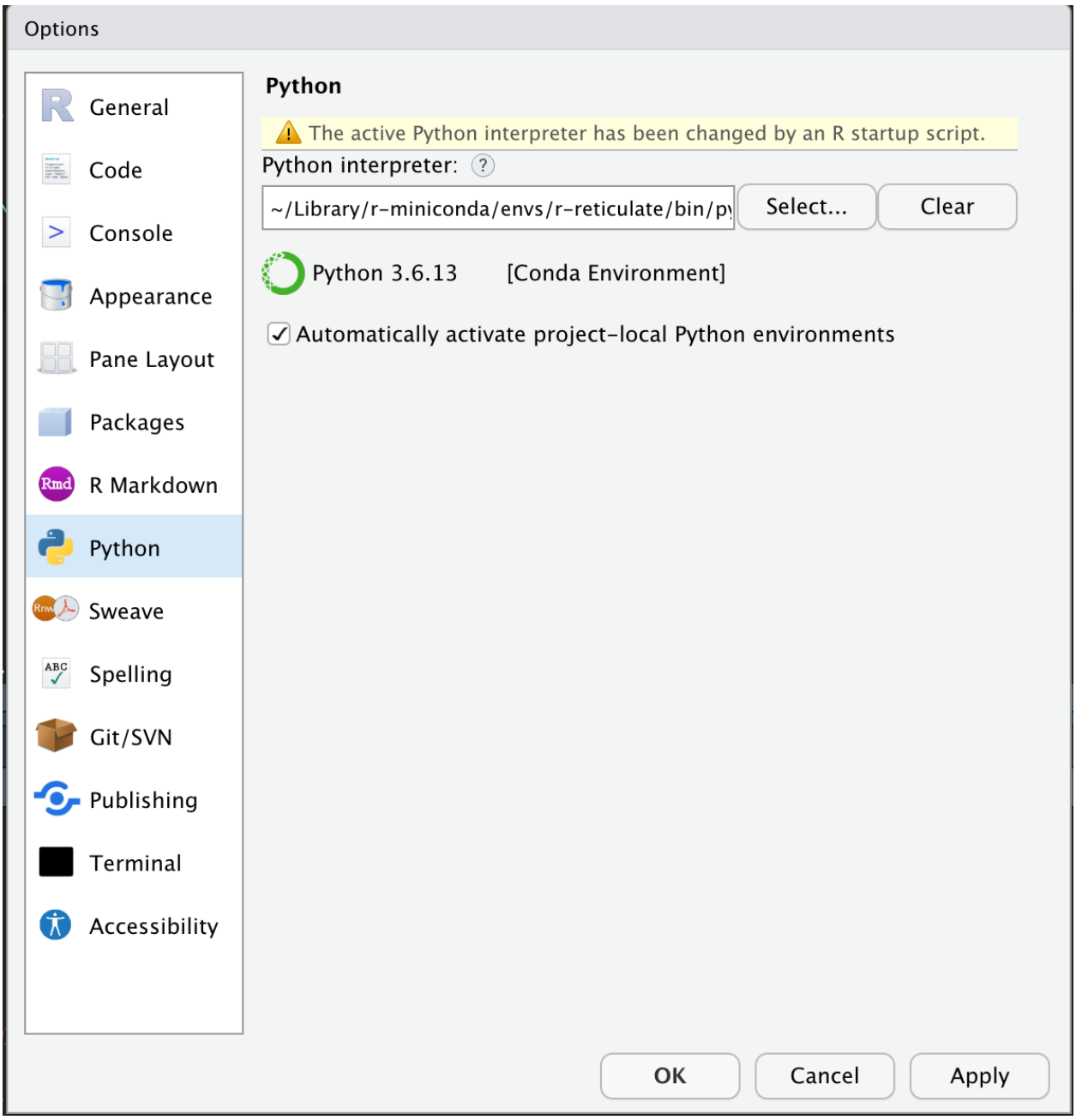

Figure 10: RStudio Python Interpreter setup

3 Installation - macOS

3.1 R & RStudio

Installation of R & RStudio is pretty straightforward. It is advised not to change any default settings because it is likely to cost you a lot of time with path settings later. Please install R first then RStudio.

- For macOS, install R package by executing `.pkg` file. For RStudio, drag and drop the RStudio application to your `applications` folder.

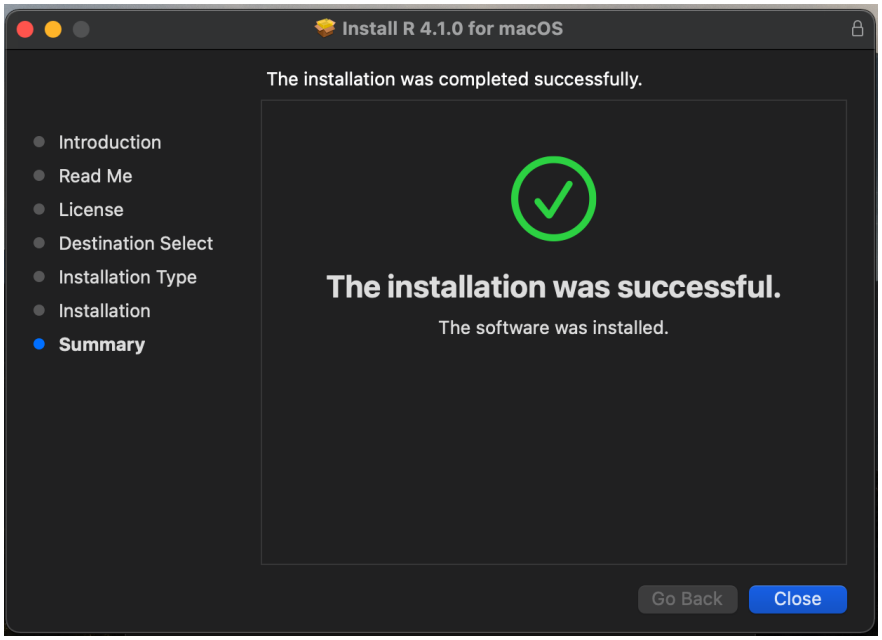

Figure 11: Mac R installation

If you have successfully installed R and Rstudio, congratulations, you will be seeing Figure 12 on your machine.

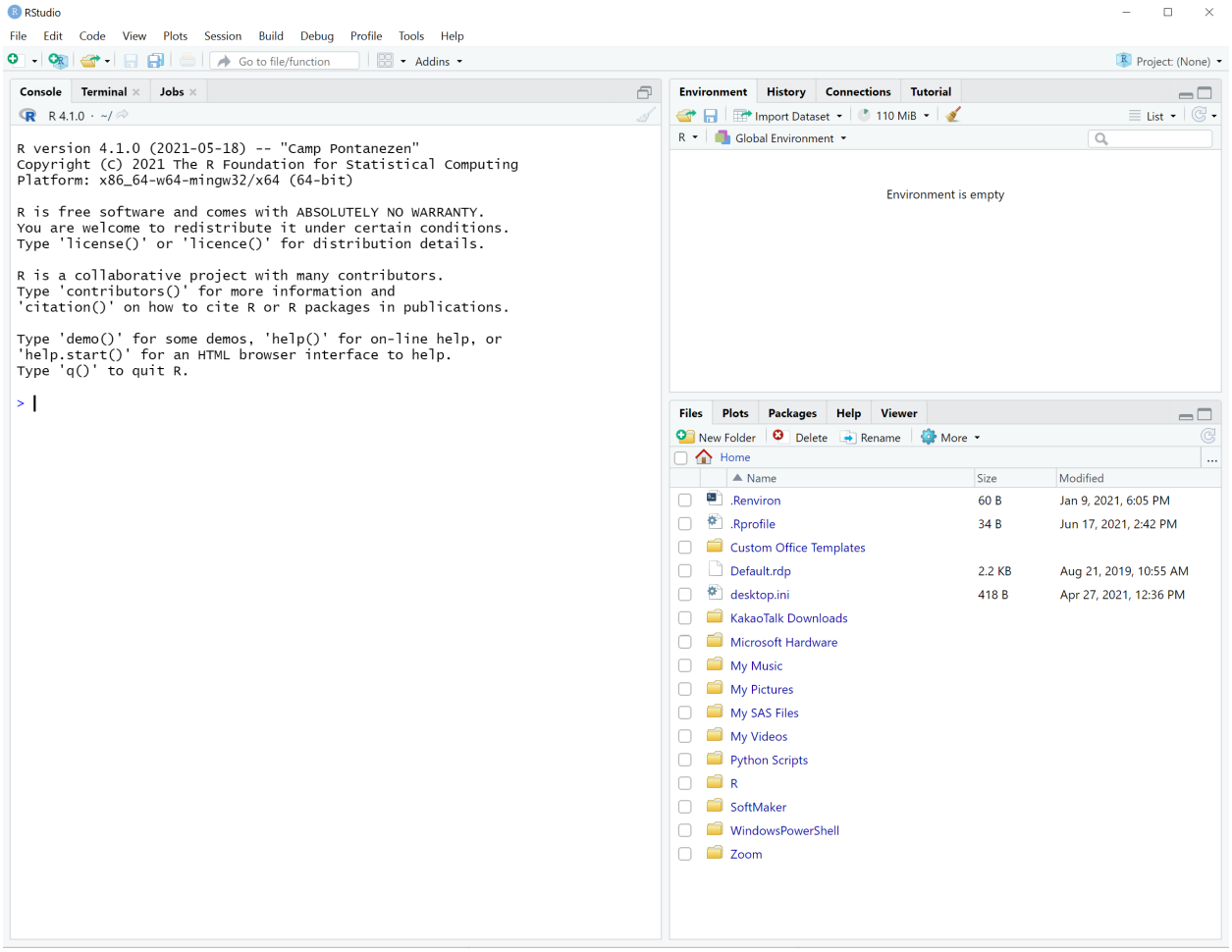

Figure 12: RStudio First Look

3.2 Python Installation

3.2.1 Apple Command Line Tools

When RStudio was first launched, it may suggest you to download Apple Command Line Tools. This tool is absolutely beneficial so go ahead and install it to your machine. It will take roughly 30 mins to fully install.

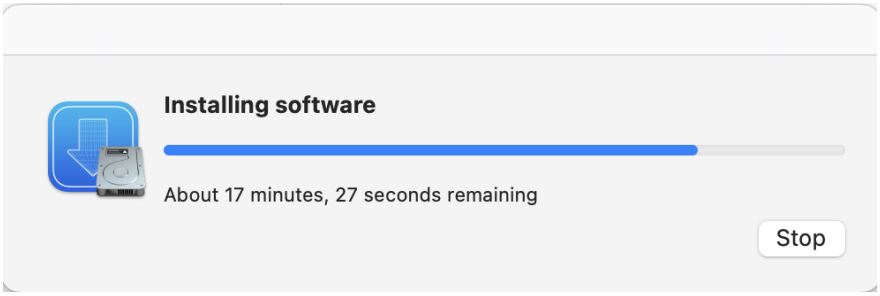

Figure 13: Apple Command Line Tools Installation

3.2.2 Python installation

Now it is time to install python. We'll be installing python through an R Package called `reticulate`, which enables to run R and Python interchangeably with the power of RStudio IDE.

- Open a blank python script and run it. It will install `reticulate` package.
- It will ask you to download necessary packages and will install miniconda[3] automatically. Answer yes to allow installations.

[3]This is a minimal and efficient version of Anaconda distribution. The standard Anaconda distribution comes with too much unnecessary packages.

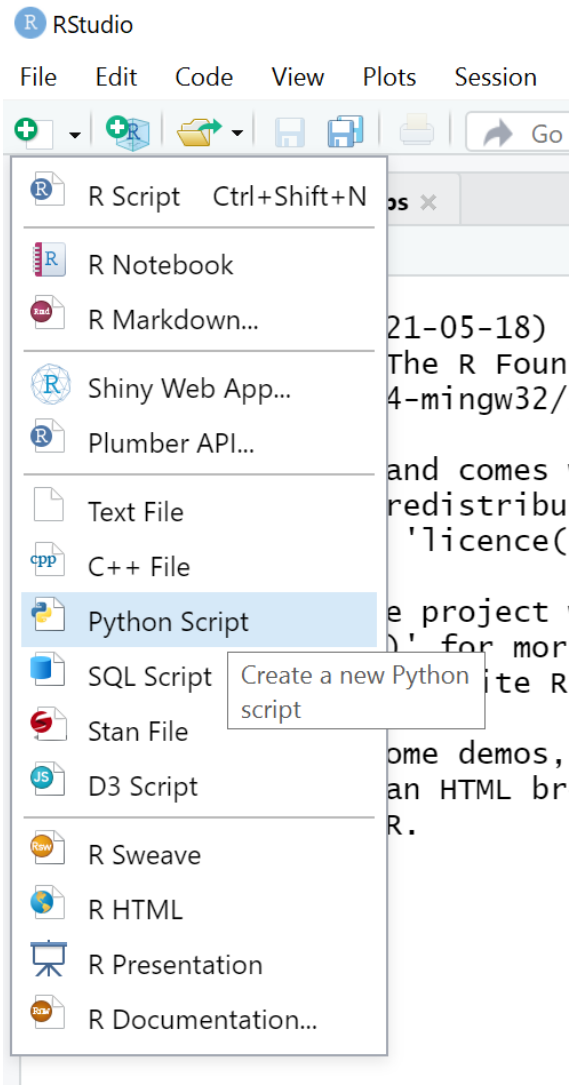

Figure 14: Create Python Script

For MacOS, they comes with built-in python3, which is installed with `Apple Command Line Tools`. We won't use this python but instead will be using miniconda python instead. Hence we need to install miniconda manually.

- Generate **R script** as shown below.

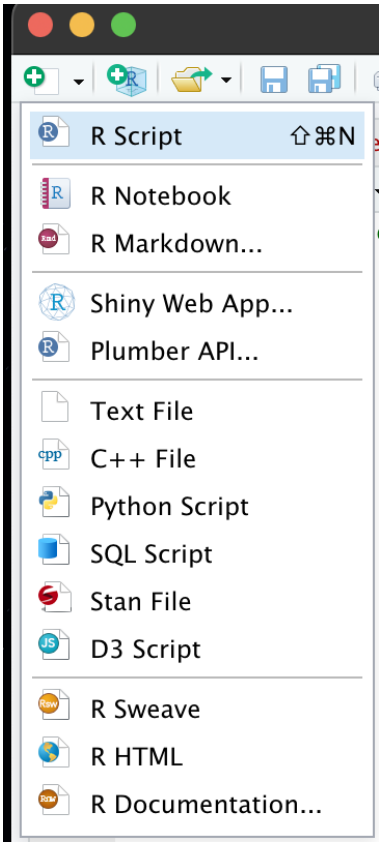

Figure 15: Create R Script

- Type commands as shown, and run it.

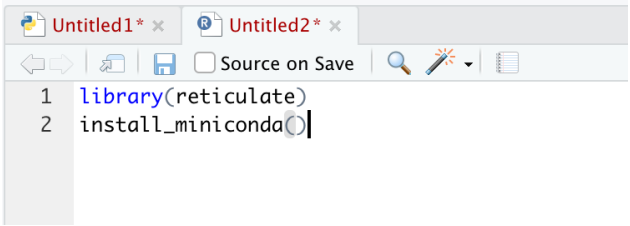

Figure 16: Run commands on R script to Install Miniconda

Unlike windows, `conda` should be activated and initialized to be used in Terminal. To achieve this, fire up your terminal and copy paste below commands line by line.

- Tip: You can find Terminal on your Launchpad, or simply use spotlight search by hitting `Cmd + Spacebar` and search Terminal.

Step 1)

```
source ~/Library/r-miniconda/bin/activate
```

- If you're using Apple Silicon Mac, then name of miniconda directory will be slightly different. Use this code instead of above if this being the case.

```
source ~/Library/r-miniconda-arm64/bin/activate # use this command instead if Apple silicon
```

Step 2)

Then, again type this command below. This will initialize conda on your `zsh`.

```
conda init zsh
```

- If you are using `bash` as your default shell, then use `conda init bash`. Your shell name is on the top of the terminal.
- If you see (base) on the left of your prompt, then your Terminal is set for conda!

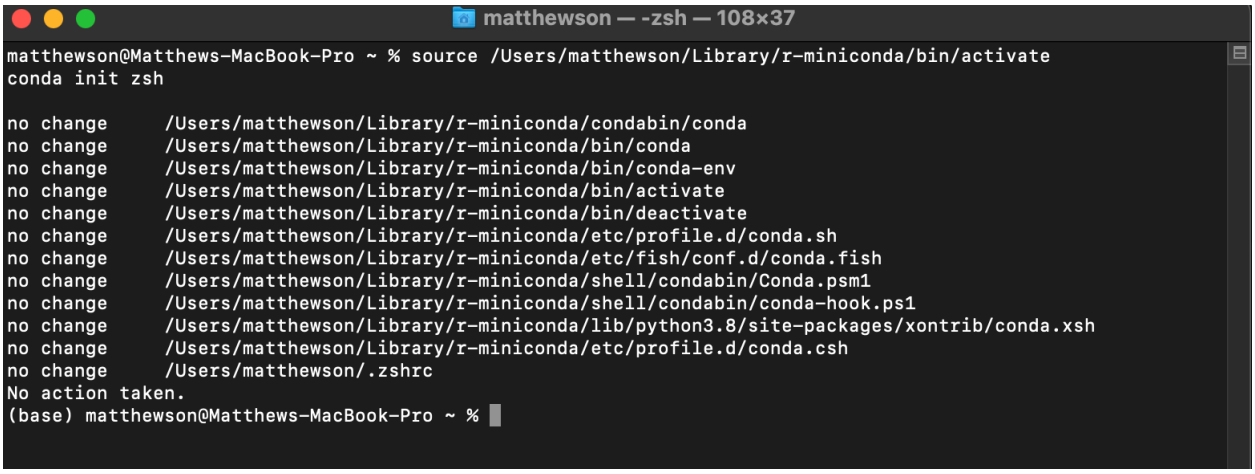

Figure 17: MacOS: Anaconda Prompt generated

3.2.2.1 Set Python Path for Mac (Optional)

Copy paste below command on your terminal. It will generate the file for settings. Then exit and re-launch RStudio.

```
echo 'RETICULATE_PYTHON="~/Library/r-miniconda/envs/r-reticulate/bin/python"' >> ~/.Renviron
```

1. Then open a python script (.py) as in figure 6 and run the empty code to activate python console.
2. Then Go to Tools > Global Options and set python interpreter for RStudio. Make sure you are choosing python from `conda environments` and choose python under `r-reticulate` environment as below.

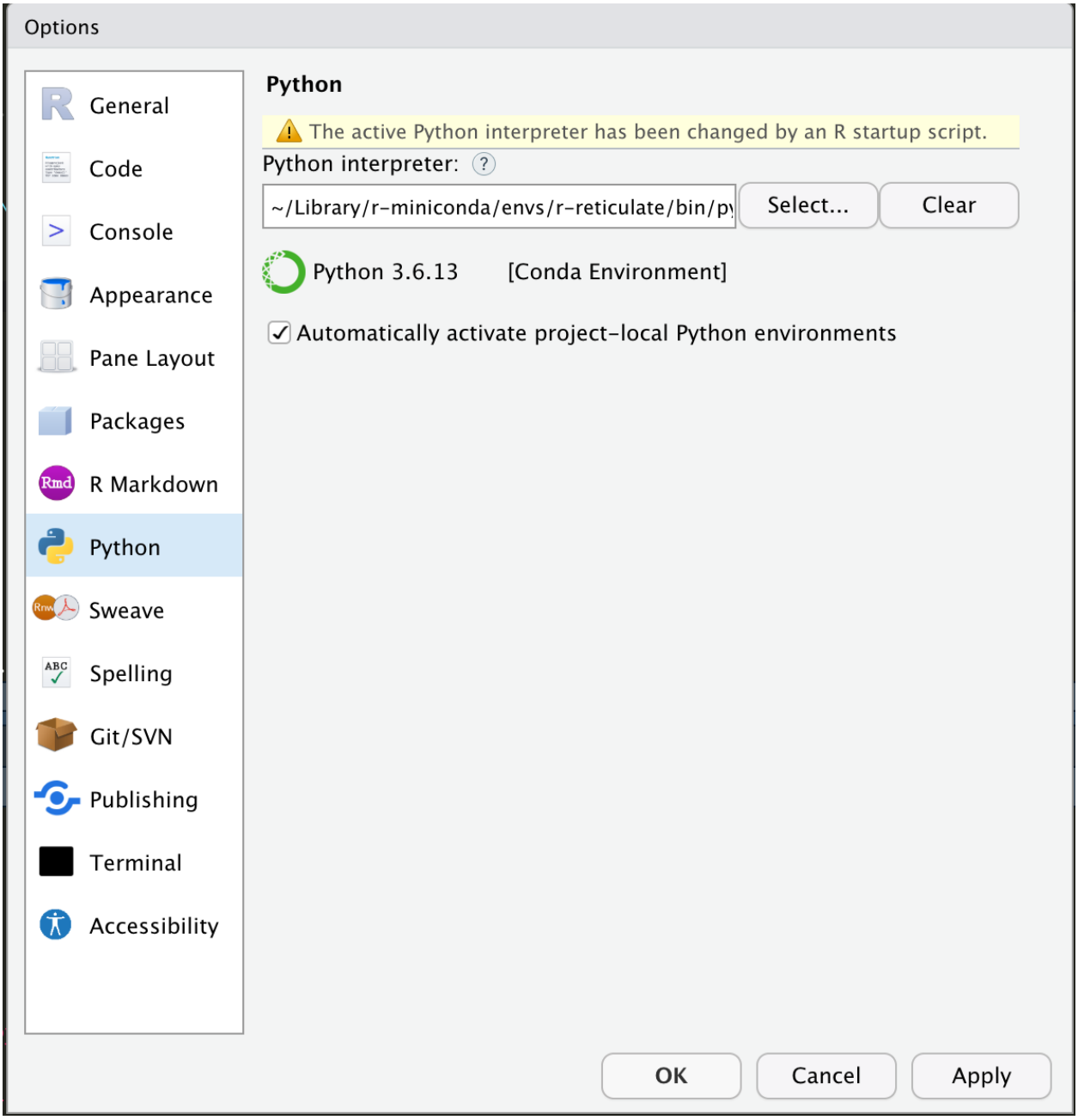

Figure 18: RStudio Python Interpreter setup

4 Troubleshooting

For some reason, python installation may not be successful. Refer to this section when above installation does not work. This section will be kept updated.

4.1 Download time limit

When internet connection is not strong and not fast enough, sometimes downloading takes more than 60 seconds. R automatically disconnects when it takes more than 60 seconds.

A quick fix is to change setting temporarily, by typing below on `R console`.

```
options(timeout=500) # setting timeout for 500 secs
```

Or to change this setting permanently, type below command on Terminal (MacOS) or Command Prompt (Windows).

1. Windows

```
echo R_DEFAULT_INTERNET_TIMEOUT=500 >> %userprofile%\Documents\.Renviron
```

2. Mac

```
echo "R_DEFAULT_INTERNET_TIMEOUT=500" >> ~/.Renviron
```

4.2 Windows

4.2.1 Error installing miniconda due to whitespace

It is reported that when username has blank space, `R` does not install on the default path. In this case, user has to specify a folder address where there is no blank space, such as `C:\r-miniconda`.

To do this, on R Prompt type this command.

```
reticulate::install_miniconda("C:\r-miniconda")
```

Then set python path for R Environment by typing below on command prompt.

```
echo RETICULATE_PYTHON = "C:\r-miniconda\envs\r-reticulate\python.exe" >>
%userprofile%\Documents\.Renviron
```

4.2.2 Manually Setting up Anaconda Prompt

If anaconda prompt is not installed on your machine, we can manually add the prompt.

- Create a shortcut (i.e. symlink) for command prompt on your desktop.
- Go to its properties and change target as

```
%windir%\System32\cmd.exe "/K" %userprofile%\AppData\Local\R-MINI~1\Scripts\activate.bat
```

- change Start in as

```
%HOMEPATH%
```

- and paste the shortcut to a folder with proper folder name located at

```
%userprofile%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs
```

4.3 MacOS

4.3.1 Reticulate path setting

Sometimes Rstudio does not locate the miniconda path automatically. In this case, let's manually set the path for Python so that our R can locate where it is installed.

4.3.2 Apple Silicon models

Currently miniconda installed by reticulate is `x86_64` architecture. Below error message will come up:

```
Error in py_initialize(config$python, config$libpython, config$pythonhome,  :
: no suitable image found.  Did find:
: mach-o, but wrong architecture
```

Instead, `miniforge` offers `arm64`(Apple Silicon) architecture, so I'm introducing this `miniforge` workaround.

1. Go to Miniforge3 webpage and download `Miniforge3-MacOSX-arm64`.

It will be downloaded in the `~/Downloads`.

2. Then type below command on your terminal to install

```
zsh ~/Downloads/Miniforge3-MacOSX-arm64.sh
```

`miniforge3` will be installed on `~/miniforge3` by default. Answer `yes` to agree and another `yes` for initiating conda.

3. Then, let's make a new virtual environment[4] called `r-reticulate` for consistency. Type below on your terminal:

```
conda create -n r-reticulate python
```

4. Next, let's set python path such that RStudio may find the path where `miniforge3` is installed. Type below command on terminal:

```
echo 'RETICULATE_PYTHON="~/miniforge3/envs/r-reticulate/bin/python"' >> ~/.Renviron
```

Now you're ready to go.

[4] simply, virtual environment is like installing a copy so that the original one is intact.

5 Installing Python packages

The virtual environment of python we will be using is `r-reticulate`, which is a python env that is connected to the RStudio. We will be installing python packages on this environment.

To achieve this, we should activate our conda environment as `r-reticulate` by using command below on `conda prompt`.

```
conda activate r-reticulate
```

then let's install python packages using command `pip install`. Below packages are what we will be using throughout the semester.

```
pip install pandas numpy scipy numpy-financial xlwings mplfinance yfinance matplotlib jupyterlab
```

6 Uninstall

6.1 MacOS R Uninstallation

Below code completely removes R, RStudio and miniconda(reticulate) installed on your computer. Use it when you need to remove and reinstall R/RStudio. You'll need to provide password on the prompt.

```
sudo rm -rf /Applications/R.app
sudo rm -rf /Applications/RStudio.app
sudo rm -rf /Library/Frameworks/R.framework
sudo rm -rf /Library/Saved Application state/org.rstudio*
sudo rm -rf ~/.config/rstudio
sudo rm /usr/local/bin/{R,Rscript}
sudo rm /private/var/db/receipts/org.R-project*
sudo rm /private/var/db/receipts/org.r-project*
sudo rm /private/var/db/receipts/org.rstudio*
rm -rf ~/Library/Application Support/R
rm -rf ~/.Renviron

conda deactivate
sudo rm -rf ~/Library/r-miniconda/
rm -rf ~/.zshrc
rm -rf ~/.bashrc
```

7 Option settings

7.1 Silence reticulate (optional)

Reticulate prints out the current version of `Python` and `reticulate` whenever python command is executed, which might be cumbersome in RMarkdown environment. In order the code to be run quietly, `.Rprofile` needs to be created to make default setup.

- For Mac, copy paste below command on your terminal will generate the settings file.

```
echo "options(reticulate.repl.quiet = T)" >> ~/.Rprofile
```

- For Windows, copy paste on your (Minconda) command prompt.

```
echo options(reticulate.repl.quiet = T) >> %userprofile%\Documents\.Rprofile
```

7.2 RStudio Workspace setup

Usually it is not recommended to load workspace in the beginning of the session, so let's turn off this feature. Uncheck `Restore .RData into workspace at startup` and set `Never` to save workspace to .RData on exit.

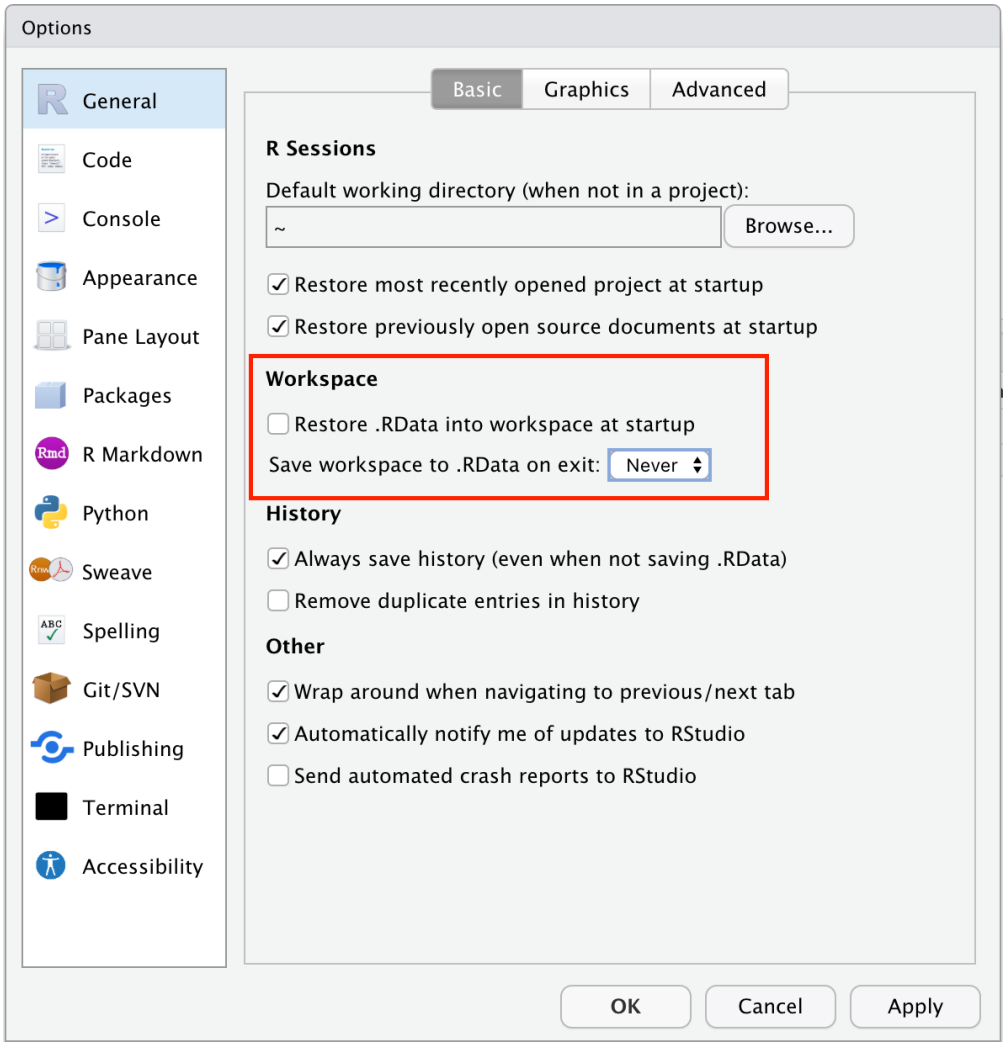

Figure 19: Workspace setup

7.3 Terminal setup for MacOS

- For MacOS, let's match our default shell (zsh) as RStudio's default as well.

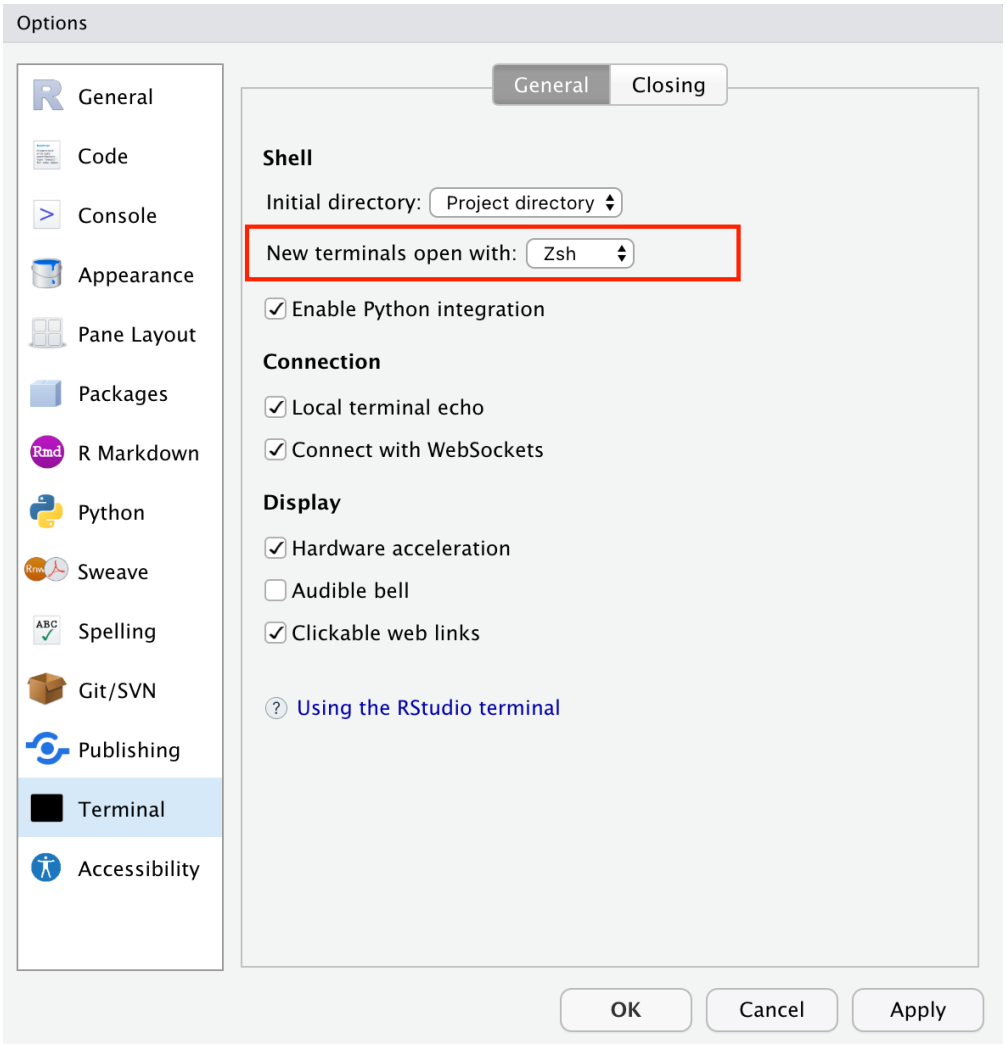

Figure 20: Change default sheel for MacOS

7.4 Excel GetFormula function setup

Throughout the course, I will use `getformula` function to explicitly show the excel function in a cell. To define a custom function, we need to add an VBA module for this.

1. Open VBA Editor

- For Windows computers, press `Alt + F11` to open VBA.
- For Mac computers, choose `Tools -> Macro -> Visual Basic Editor`

2. Add a module

- Click `Edit -> Insert Module`

3. Code Copy paste below code and get back to excel.

```
Function getformula(r As Range) As String
    Application.Volatile
    If r.HasArray Then
    getformula = "<-- " & _
        " {" & r.FormulaArray & "}"
    Else
    getformula = "<-- " & _
        " " & r.FormulaArray
    End If
End Function
```