Use R in the cloud

August 15, 2021
Software and apps

A limited guide about how to how to run R via a **cloud computing** (serverless) option. For local installation of R onto a personal computer, please see Install R.

My suggestion for BI-311 students — use Google CoLaboratory. However, do explore each option to find your favorite way to interact with R in the cloud. For example, the sandbox options are great for testing code snippets.

Installation guides quickly become outdated. This page last updated 12 August 2025 and describes working installation protocols at that time.

Quick links

1. myCompiler
2. JupyterLite
3. CoCalc by SageMATH
4. Google CoLaboratory ← Dohm's preferred choice
5. RStudio in the cloud
6. rdrr.io

Example R code snippet

Below you'll find brief introductions to five different cloud options for running R. For your convenience, the example R code I run is listed below

```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY, myX)
```

Bonus: Since Descartes and the Cartesian coordinate system, by convention we place the X variable on the horizontal axis. Confirm and, if necessary, alter the provided code with smallest number of changes to yield a scatter plot with X variable on horizontal axis and Y variable on vertical axis.

Run R on your computer (ie, local development environment or LDE)

This page is about running R in the cloud, a server-less options. For installing R and R Commander onto your own computer (= your LDE option), see Install R and Install R Commander.

Note 1: You cannot install R Commander to any cloud server option. R Commander requires access to a local graphical display; in the cloud, all interactions between you and R are accomplished via the web browser.

Run R "in the Cloud"

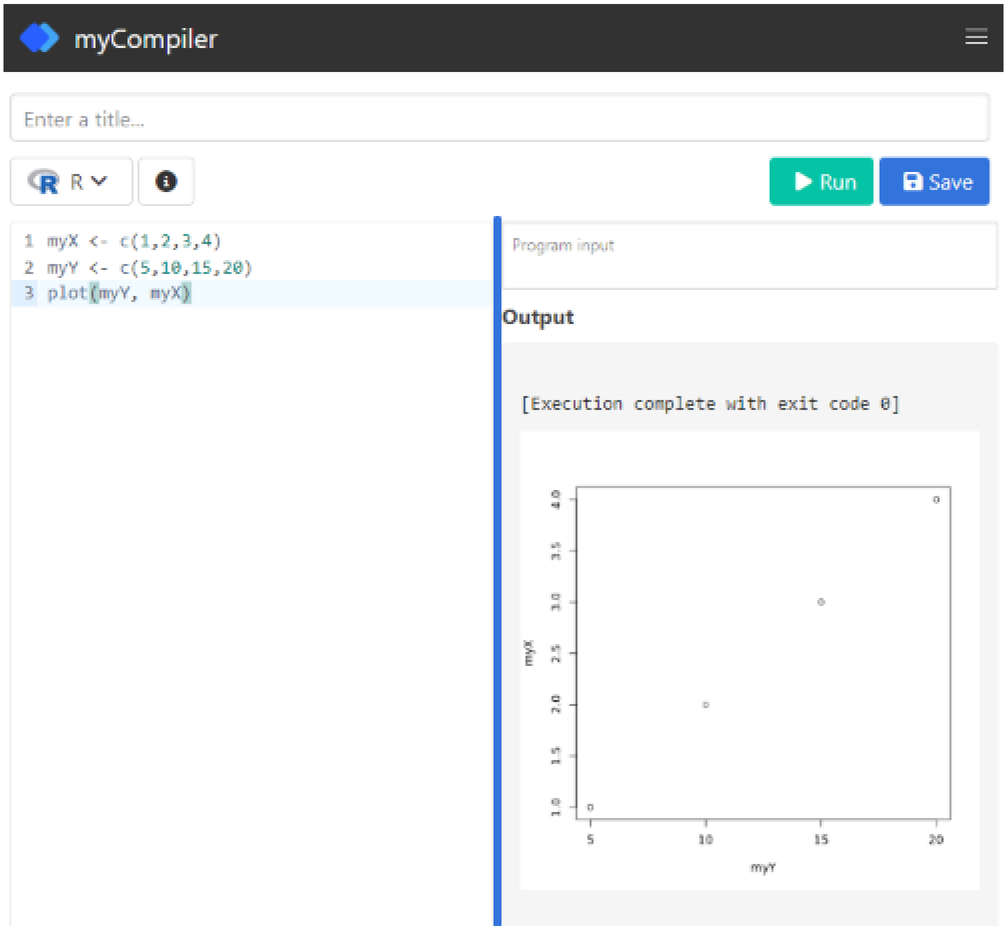
If you do not wish to install R, or, if you have a ARM-based Chromebook and, therefore cannot gracefully install R, then there are alternatives; Run R in the Cloud. I'll list five ways to run R in the cloud — run R on a server, not your own computer — for free.

Note 2: JupyterLite, Google CoLab, CoCalc, and RStudio in the Cloud offers a free version that is generally sufficient for BI-311 coursework. However, the free service has limitations, such as reduced computing resources, session timeouts, and occasional downtime. Paid CoLab Pro options are available but not required for this course. All exercises in this Workbook are designed to run on the free version.

If you have a Chromebook, or you want to run R on your tablet (iPad, Kindle, etc.), you can't typically install R to any of these devices (see Linux distros for limited exception for some Chromebooks). However, you can access R via a serverless Cloud solution.

Note 3: myCompiler (#1), JupyterLite (#2), and rdrr.io (#6) are examples of code playgrounds or sandboxes. These are great for running small amounts of code — like solving a homework problem. Playgrounds shouldn't be used for larger projects and you shouldn't expect all R packages will run in playgrounds. The other disadvantage, you probably shouldn't mount your Google drive in a code playground!

1. Run R code at Online R Compiler using **myCompiler**'s online IDE, link at https://www.mycompiler.io/online-r-compiler. Example of myCompiler and R screenshot shown in Fig 33.



```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY, myX)
```

```
[Execution complete with exit code 0]
```

Figure 33. Screenshot of myCompiler session.

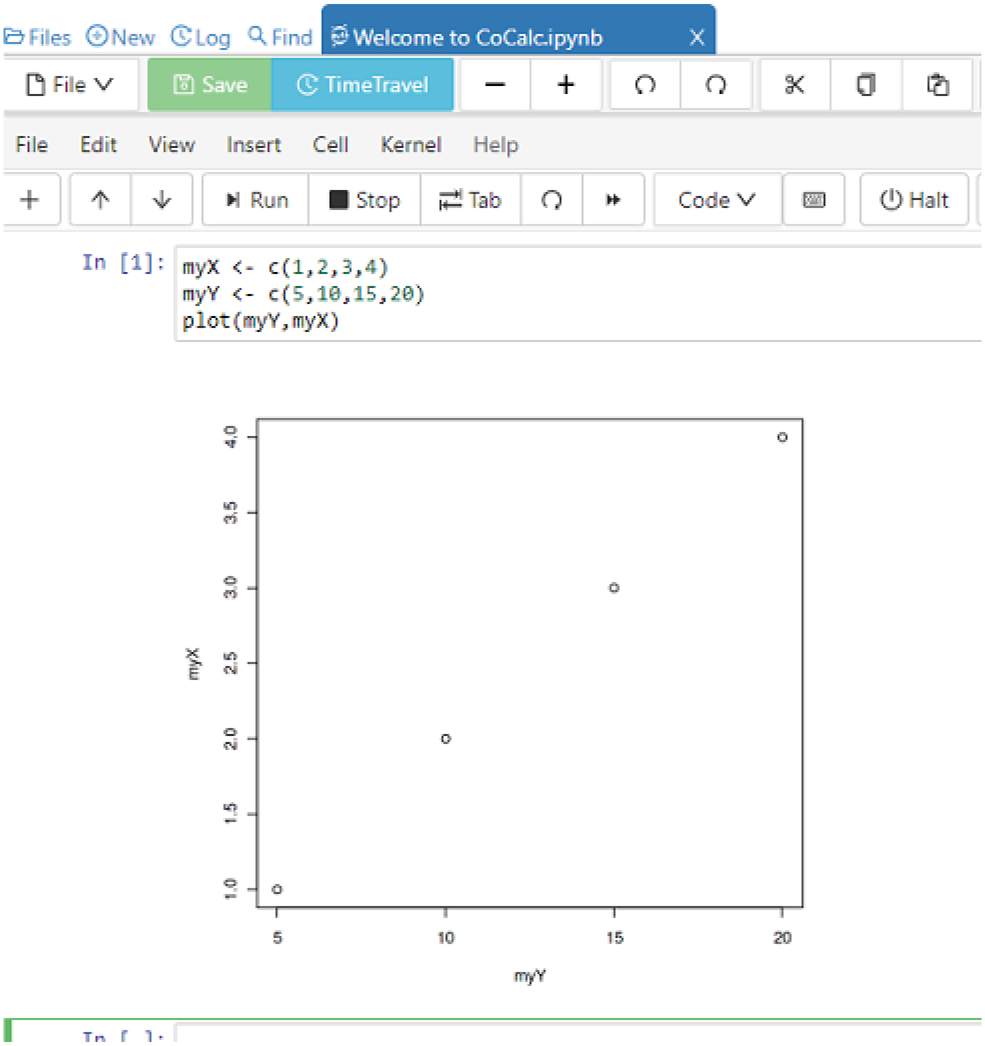
myCompiler runs dozens of programming languages. MIT's "Try It Online," seems to be tops at this, with access to hundreds of programming languages, including Pascal — which takes me back to my graduate student days.

2. JupyterLite uses WebAssembly to run Python code within the browser. It's a good option to try code snippets. Like all JupyterLab options, the default environment is Python. The home screen provide options to "launch" different coding environments and R is one of the options. In general, choose the Notebook options, although note that you can load and edit markdown files. For additional information about running R code relevant to Jupyter Notebooks see step 4a or step 4b below.

3. Run code snippets in **CoCalc** by folks at SageMath and available at https://cocalc.com/ . CoCalc uses Jupyter Notebooks, a wonderful, open-source project which supports interactive computer coding for many languages, including R and Markdown.

While CoLab is my go to, CoCalc is a really good student option — hint: I have my Systems Biology students use this option — includes SageMATH, python, GNU Octave and other software.

Create a free account (you'll then be able to save your code), or simply click "Run CoCalc Now" and check the box to agree to the terms to begin a session (Fig 34). Choose to open a new **Jupyter notebook**, then select R (system wide) from the choice of kernels.



```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY,myX)
```

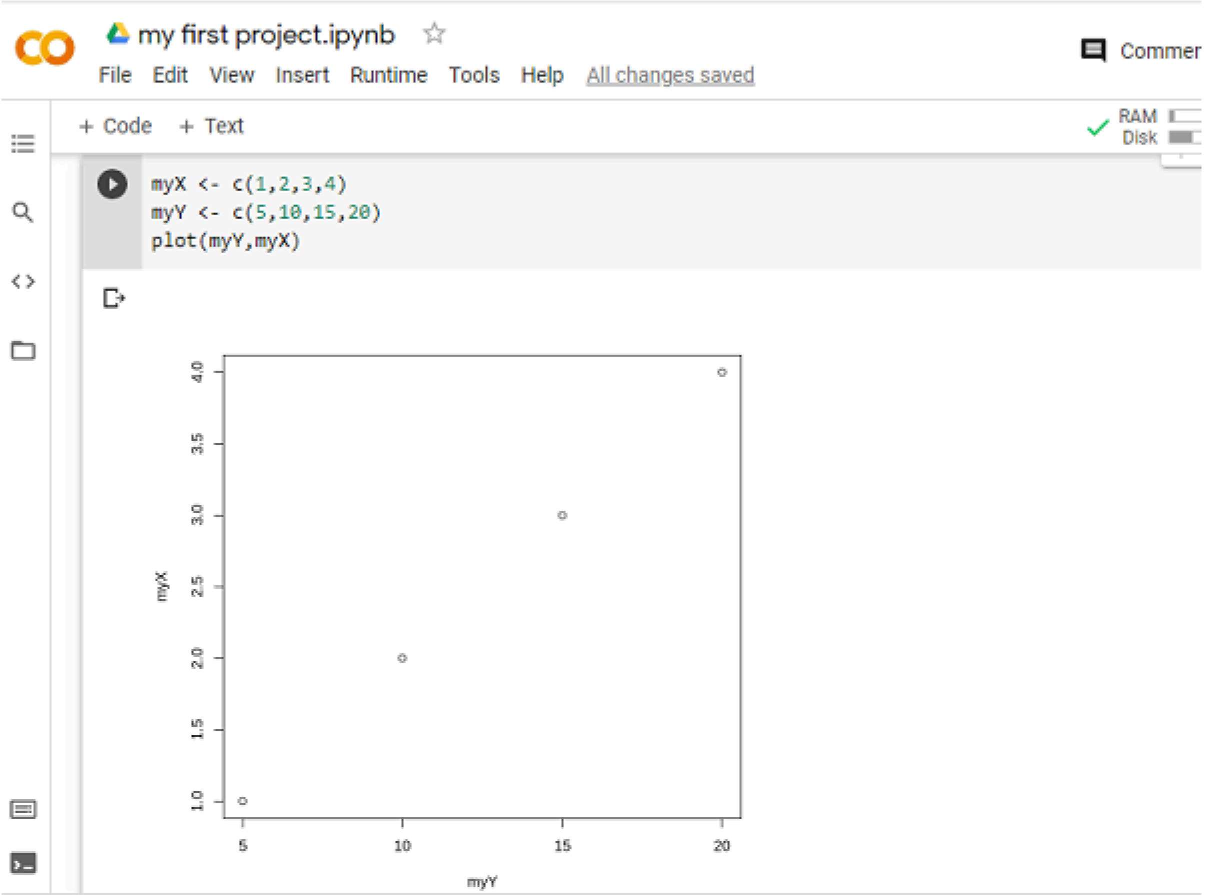
Figure 34. Screenshot of CoCalc session.

You can load files from your computer for use in CoCalc sessions. There is also a version of the software you can download to your computer.

4. My favorite option for running R in the cloud is to run R code snippets at Google **Colaboratory** (Fig 35). Like CoCalc, Colaboratory uses Jupyter Notebooks. One real advantage of choosing CoLab, there are apps to run Google Colaboratory and Jupyter Notebooks on iPad/iPhone and for Android phones are available at Apple App Store and Google Play, respectively.

Note 4: Google CoLab runs python by default. Because the runtime resets with each session, you'll need to reset the runtime to R each time, following these listed steps (4a). Alternatively, leave the python environment as is an run R commands via magic commands and `rpy2.ipython` extension (listed below at 4b).

4a. Setup CoLab for use in the cloud is straightforward, just four steps as of August 2025.



```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY,myX)
```

Figure 35. Screenshot of Google Colab session.

Step 1. Log in to your Google account (or create one if you don't already have one).

Step 2. Click the url link https://colab.research.google.com/#create=true&language=r , or try the short URL https://colab.to/r. Alternatively, if you prefer, here's a qrcode (Fig 36).

Figure 36. QR code with url to create new R notebook in CoLab.

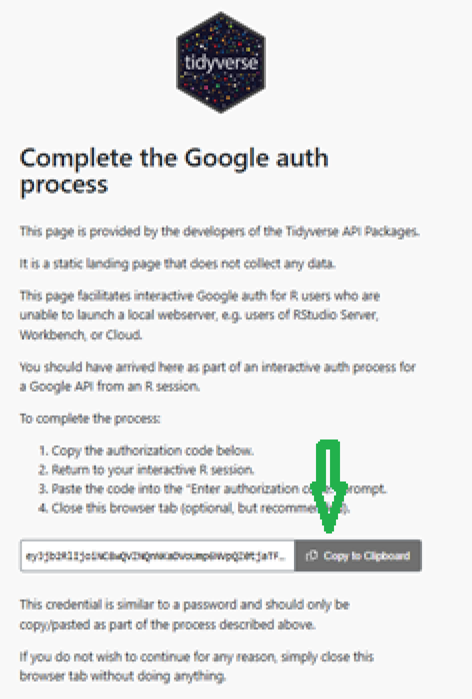
Step 3. After creating a notebook, you'll want to connect to your Google Drive to upload data files and script files, and to save files from R sessions.

```
# Install and load the functions needed to connect to Google drive
install.packages("googledrive")
```

```
library(googledrive)

# Authenticate your account
drive_auth(use_oob = TRUE)
```

The instructions call for you to click on web link and copy from the web page an authorization code (green arrow, Fig 37).



Complete the Google auth process

This page is provided by the developers of the Tidyverse API Packages.

It is a static landing page that does not collect any data.

This page facilitates interactive Google auth for R users who are unable to launch a local webserver, e.g. users of RStudio Server, Workbench, or Cloud.

You should have arrived here as part of an interactive auth process for a Google API from an R session.

To complete the process:

1. Copy the authorization code below.
2. Return to your interactive R session.
3. Paste the code into the "Enter authorization code" prompt.
4. Close this browser tab (optional, but recommended).

eyJjb2RlIjoiNCBaQVZNQnNkaDVoUmp0WpQZ0tjaTF…

This credential is similar to a password and should only be copy/pasted as part of the process described above.

If you do not wish to continue for any reason, simply close this browser tab without doing anything.

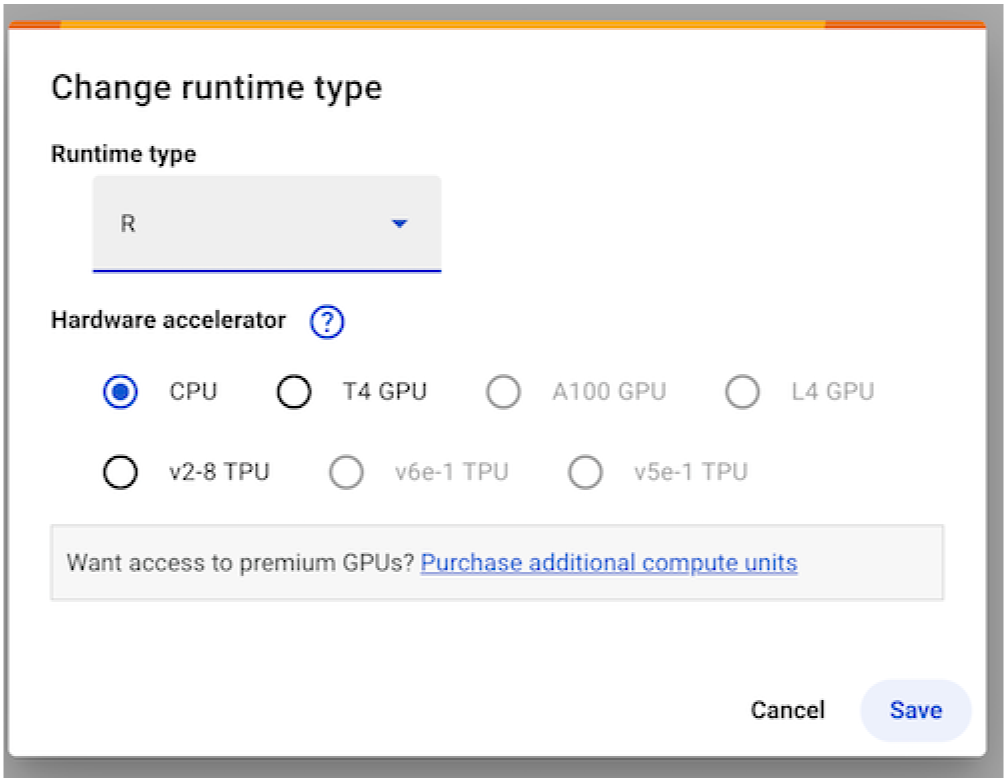
Figure 37. Screenshot of "Complete the Google auth process" page.

Copy the authentication code and paste it into the space provided on the colab notebook.

Step 4. You'll want to rename and update the file name. Note in Figure 35 that the notebook name has the `.ipynb` file extension. Because we're running R, don't forget to change from the Python file extension to the `.R` file extension.

You're ready to run your R code.

To confirm use of R as opposed to Python (the default language), from the menu bar select **Runtime → Change runtime type** and confirm R is the runtime type (Fig 38).



Change runtime type

Runtime type

R

Hardware accelerator

● CPU ○ T4 GPU ○ A100 GPU ○ L4 GPU

○ v2-8 TPU ○ v6e-1 TPU ○ v5e-1 TPU

Want access to premium GPUs? Purchase additional compute units

Figure 38. Screenshot, confirm R is the programming language in use by the Google Colab session.

After completing a session, hide code you do not want to share and print the notebook to a pdf file.

Colabs is worth the effort — you end up with a system to run R in your browser, it's free to use, and you can store/retrieve files from your Google Drive. This is my choice for Cloud computing, and it's the most generic solution. Starting in Fall 2025, we use Google CoLab in my biostatistics course at Chaminade University.

For more information about R and CoLab, see post by Ed Adityawarman, How to use R in Google Colab.

Colab Jupyter notebooks use Python by default. To run R, either use the link listed above each time you want to create a new R notebook, or add the following code snippet to your new notebook page

4b. Run R from within Python environment in your Jupyter Notebook by use of "magic"

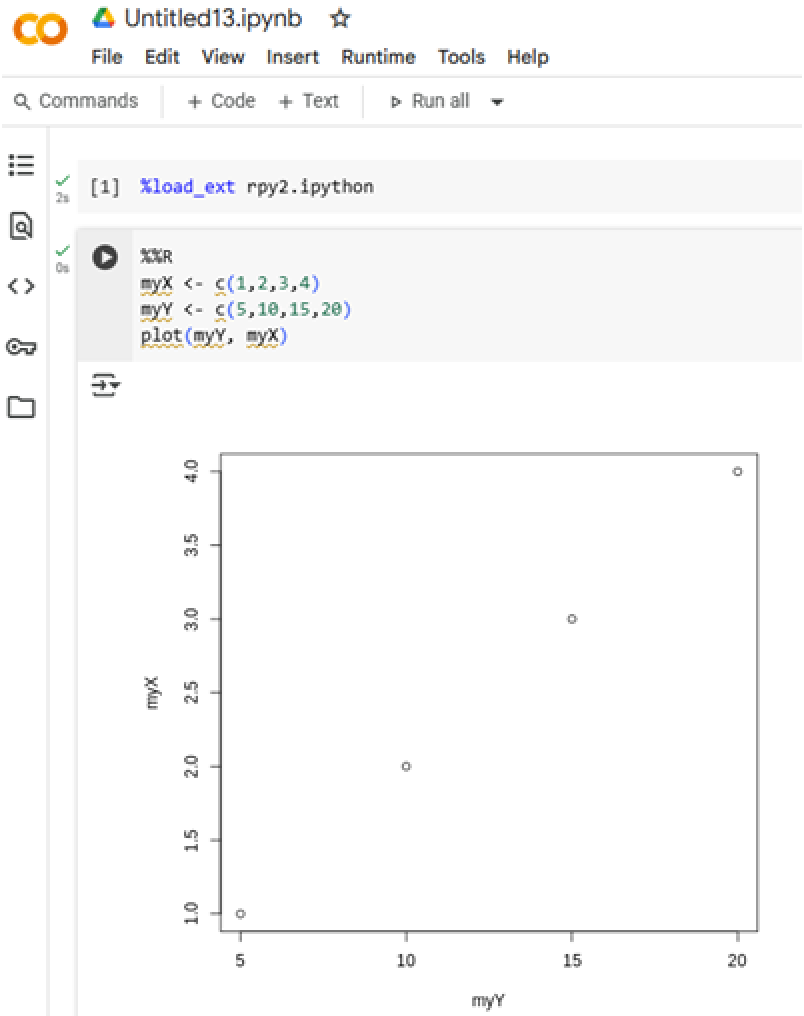
```
# activate R magic – must begin each R code with %%R
%load_ext rpy2.ipython
```

For all subsequent R code, start the section with

```
%%R
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY, myX)
```

Code and output displayed in Fig 39.

Note 5: %%R is for use with multiple lines of code. If only one line, then use %R.


Untitled13.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
4.0
3.5
3.0
2.5
2.0
1.5
1.0
myX
5
10
15
20
myY

```
[1] %load_ext rpy2.ipython
```

```
%%R
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY, myX)
```

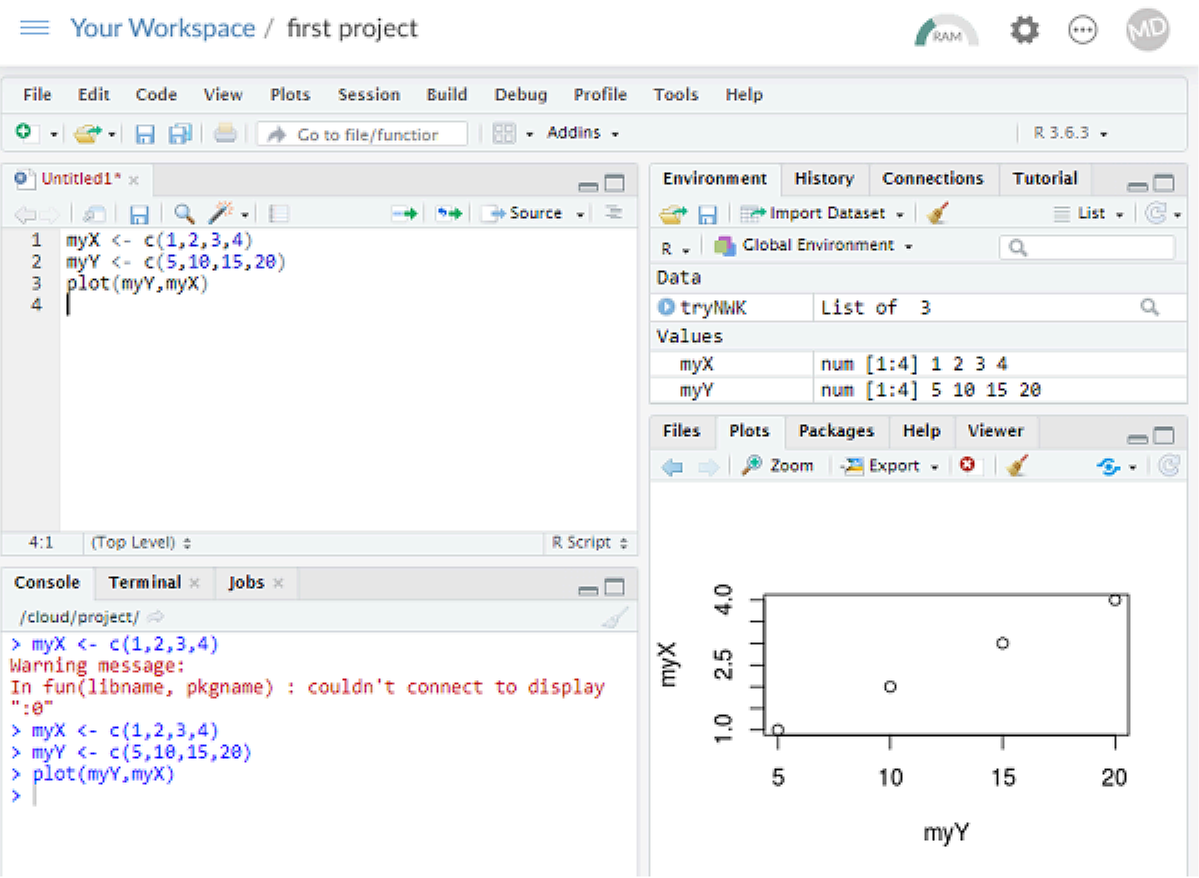
Figure 39. Screenshot magic function `rpy2` in use on CoLab, python environment.

Note 6: Jupyter Notebooks are a fantastic development in data science, particularly for collaboration. Although not a focus on my course, serious students may want to explore the Jupyter Notebook environment further (see Kluyver et al 2016). For example, you can install Jupyter onto your computer via Miniconda — **conda** is an open source package management system but then, you still would have to install R to your computer.

5. You can run **RStudio** at Posit Cloud. Registration and use is free for students. This works OK, but can be slow and it's hard to work on your own data (free plan allows 25 projects and no more than 25 hours of computing time). It does have the advantage of providing the familiar RStudio interface. No doubt, RStudio is the predominate tool used by R programmers.

Choose the free plan; Instructions to get started are at https://posit.cloud/plans. You can link to your Google Drive in cloud version of RStudio via the googledrive package (introduced above with instructions for Google Colab).

A screenshot of an RStudio cloud session is shown in Figure 40.



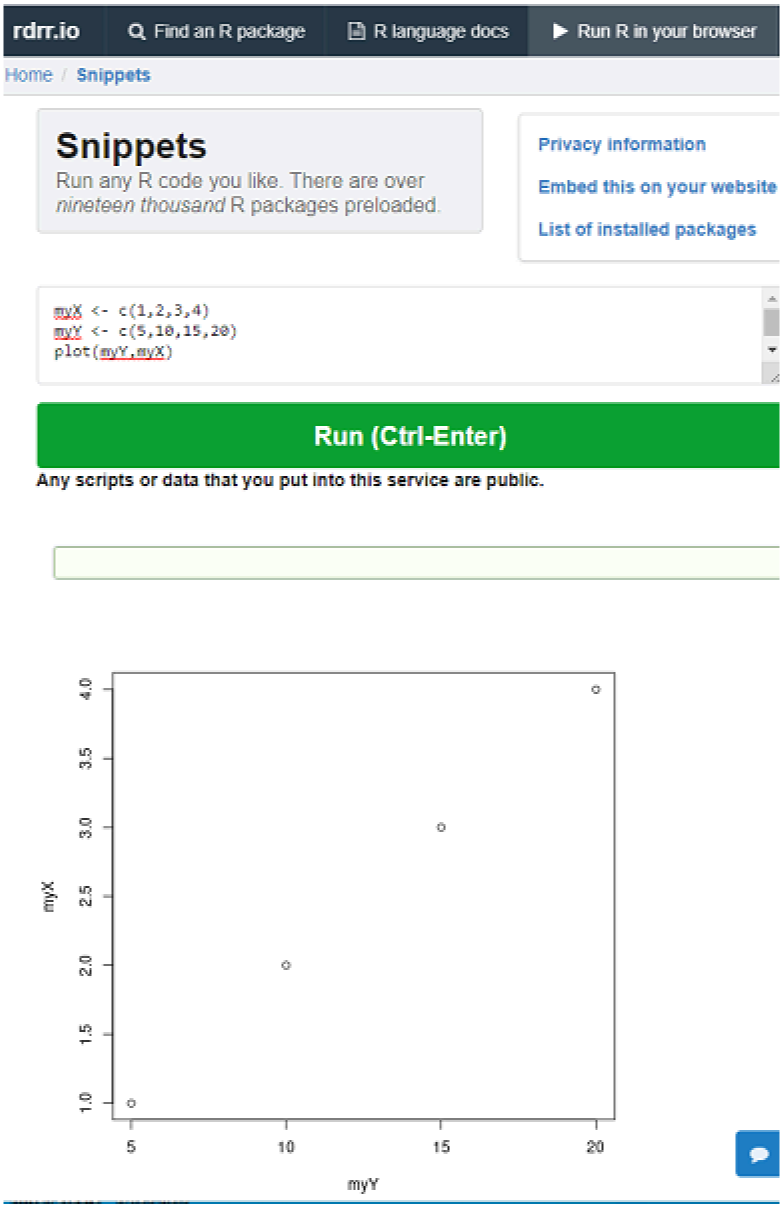
```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY,myX)
```

```
Data
tryNWK     List of  3
Values
  myX      num [1:4] 1 2 3 4
  myY      num [1:4] 5 10 15 20
```

```
> myX <- c(1,2,3,4)
Warning message:
In fun(libname, pkgname) : couldn't connect to display
":0"
> myX <- c(1,2,3,4)
> myY <- c(5,10,15,20)
> plot(myY,myX)
>
```

Figure 40. Screenshot of RStudio at Posit Cloud session.

6. For limited use, ie, you just need to run a little code to solve an assignment problem, you can run R **code snippets** in your browser at https://rdrr.io/snippets/ (Fig 41). You'll see many of my code embedded in this service so that you can run code snippets from my Chaminade University CANVAS pages.



Snippets

Run any R code you like. There are over *nineteen thousand* R packages preloaded.

```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY,myX)
```

Any scripts or data that you put into this service are public.

Figure 41. Screenshot of rdrr.io/snippets session.

References and additional resources

Kluyver, T., Ragan-Kelley, B., Pé, Rez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., n, Abdalla, S., Willing, C., & Team, J. D. (2016). *Jupyter Notebooks – a publishing format for reproducible computational workflows*. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (pp. 87–90). IOS Press. https://doi.org/10.3233/978-1-61499-649-1-87