SUPPLEMENT 2: INTERACT WITH PYTHON

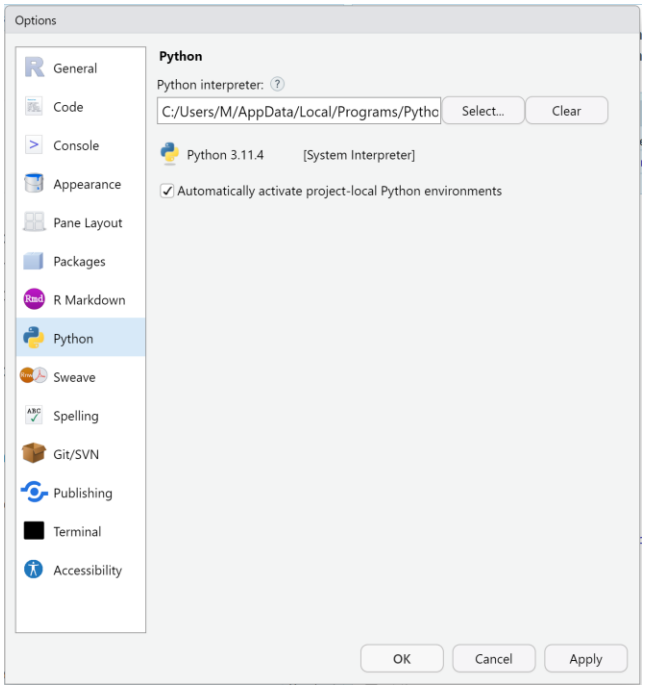
7316 - INTRODUCTION TO DATA ANALYSIS WITH R

Mickaël Buffart (mickael.buffart@hhs.se)

This short document is a supplement to the module on models. In the module outline, I explained how to use Stata models directly within R. In this document, I explain how to use Python in RStudio and make it interact with the R environment.

Python can be used natively in quarto documents, in RStudio, the same way that you use R. For this, you need to have a python interpreter installed on your computer, and set in the options of R Studio. In my case, I use the latest version of Python, but you could as well use Anaconda or Miniconda if you prefer.

Once the Python interpreter is installed in your machine, select in in RStudio, though `Tools` > `Global Options…` > `Python` > `Select`.

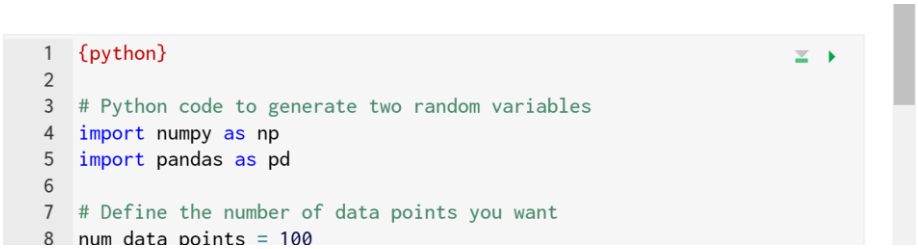


Select your Python interpreter

After this is done, restart RStudio, and start using Python in your quarto documents!

1. Running Python in RStudio

- To run a Python code in RStudio, create a new code chunk. Instead of starting it with `{r}`, simply start it with `{python}`. That's it.



Begining of a python code chunk

- Then, in the chunk, you can simply type the code in Python.

```python
# Python code to generate two random variables
import numpy as np
import pandas as pd

# Define the number of data points you want
num_data_points = 100

# Generate random values for x and y with a normal distribution
mu, sigma = 0, 1  # Mean and standard deviation
x = np.random.normal(mu, sigma, num_data_points)
y = np.random.normal(mu, sigma, num_data_points)

# Create a DataFrame
data = {'x': x, 'y': y}
df = pd.DataFrame(data)
```

- The code above should run as a Python script if Python is properly set in your environment. It generates two random variables, and save them in a dataframe called df. But this dataframe is not accessible in the r environment. It is only visible to Python.

```r
# In R, the size of df is NULL
nrow(df)
```

```
NULL
```

Note that the code above is written in R. The code chunk starts with `{r}`.

2. Getting the data in the R environment

- Let's transfer the dataframe created in python, above, into R. Thanks to the `reticulate` package, the object created in Python are accessible in R within an object called `py`.

```r
# Load reticulate
library(reticulate)

# In R, transfer df from Python to R
df <- py$df
```

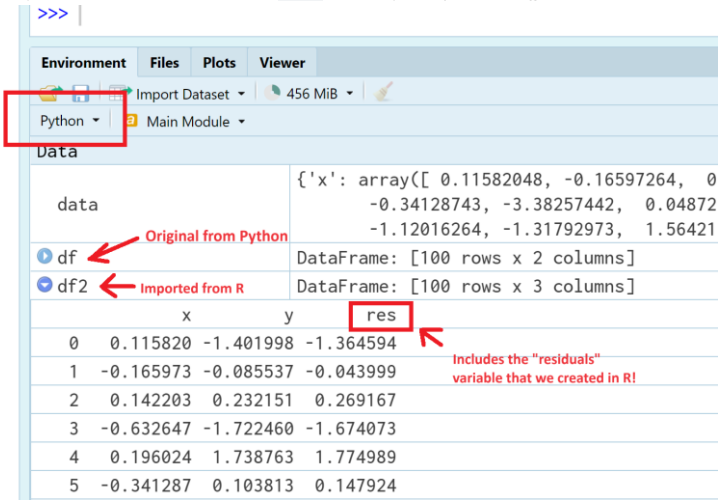
- Now, you can manipulate df in R

```
# Use the dataframe created in Python, with a model in R.
model_1 <- lm(y ~ x, data = df)

# Print the results
summary(model_1)


Call:
lm(formula = y ~ x, data = df)

Residuals:
    Min      1Q  Median      3Q     Max
-1.7409 -0.5626 -0.1567  0.3852  3.2339

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.06729    0.09540   0.705    0.482
x           -0.07700    0.08643  -0.891    0.375

Residual standard error: 0.948 on 98 degrees of freedom
Multiple R-squared:  0.008033,  Adjusted R-squared:  -0.00209
F-statistic: 0.7936 on 1 and 98 DF,  p-value: 0.3752

# Extract the residuals
df$res <- model_1$residuals
```

- In Python, it is the other way around. The objects saved in the R environment are accessible in Python through the `r.` object.

```
# In Python, transfer df from R,
#   including the residuals we generated, to Python...
df2 = r.df
```

- In your environment tab, you can see both Python and R object (if you choose Python or R in the dropdow menu). In the Python environment, you can see now the original `df` dataframe that you created, and the `df2`, that you just imported from the R environment.



df2 in Python, from R

- Then, you can manipulate df2 in python, just like any other Python object. For example, create an histogram.

```
import pandas as pd
import matplotlib.pyplot as plt

# In Python, plot a histogram of the 'res' column
plt.hist(df2['res'], bins=10, edgecolor='black')

plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram of res')
plt.show()
```

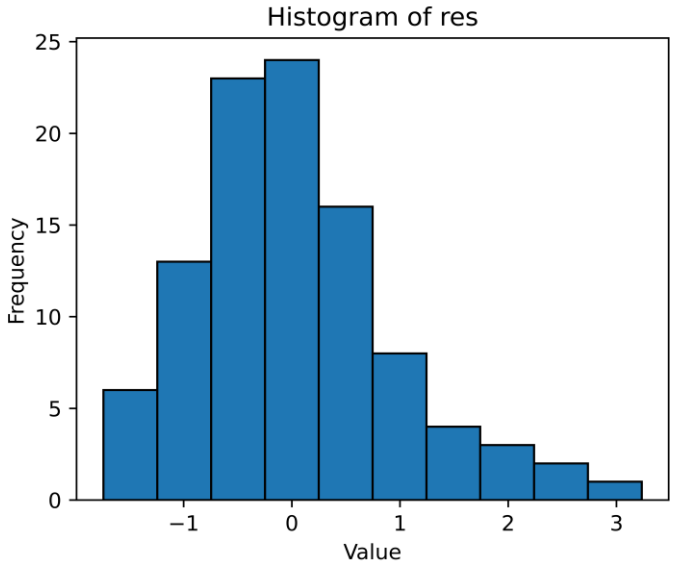


Figure 1: Histogram with Matplotlib in Python

- Note that the histogram, done with Python, now appears in the output of your quarto document.

This was a brief supplement to show how R and Python and be used together in quarto documents. To learn more about it, you can read the documentation of reticulate, and of Python in quarto.