USE PYTHON TO GENERATE PDF REPORT

Ahmad Ehyaei

03 OKTOBER, 2021

Reticulate

The `MPIThems` templates aren't just for R programmers; due to the reticulate package, Python programmers may use them as well. The `reticulate` package offers a Python engine for R Markdown, allowing for simple interchange between Python and R chunks. for install reticulte use cran repository as below:

```
install.packages("reticulate")
```

Choose Python version

The first step to running Python code is to choose the Python engine or environment. There are two ways to choose a Python environment:

1. By default, reticulate uses the system Python found on your OS. To find the path, run the R command `Sys.which("python")` on the console. Alternatively, to use your installed Python, add your Python path with the command `use_python()` to set the default engine.

```
library(reticulate)
use_python("/usr/local/bin/python")
```

2. If you have an Anaconda or Miniconda, you may use the use `use_condaenv()` or use `use_virtualenv()` to run Python code in any of these environments.

```
use_condaenv("Python3.8", required = TRUE) # Name of environment
```

The Conda environment `r-reticulate` will be created when the `reticulate` package is installed. To see the list of your conda environment, run in `conda list env` bash or R command `retculate::conda_list()`. Run the `py_config()` command to ensure that reticulate is utilizing your updated conda env.

Install Python Package

The `reticulate` env has mininl Python pacakges. In order to install a new package, use `py_install()` command. The packages will be installed by default in `r-reticulate` env. For more information you can see reticulate's vignette Installing Python Packages

```
py_install("pandas")
```

Check Python working

To ensure that everything is working properly, try writing a basic Python code in RMarkdown. Set the Python command in the Python chunk `{python}` for this.

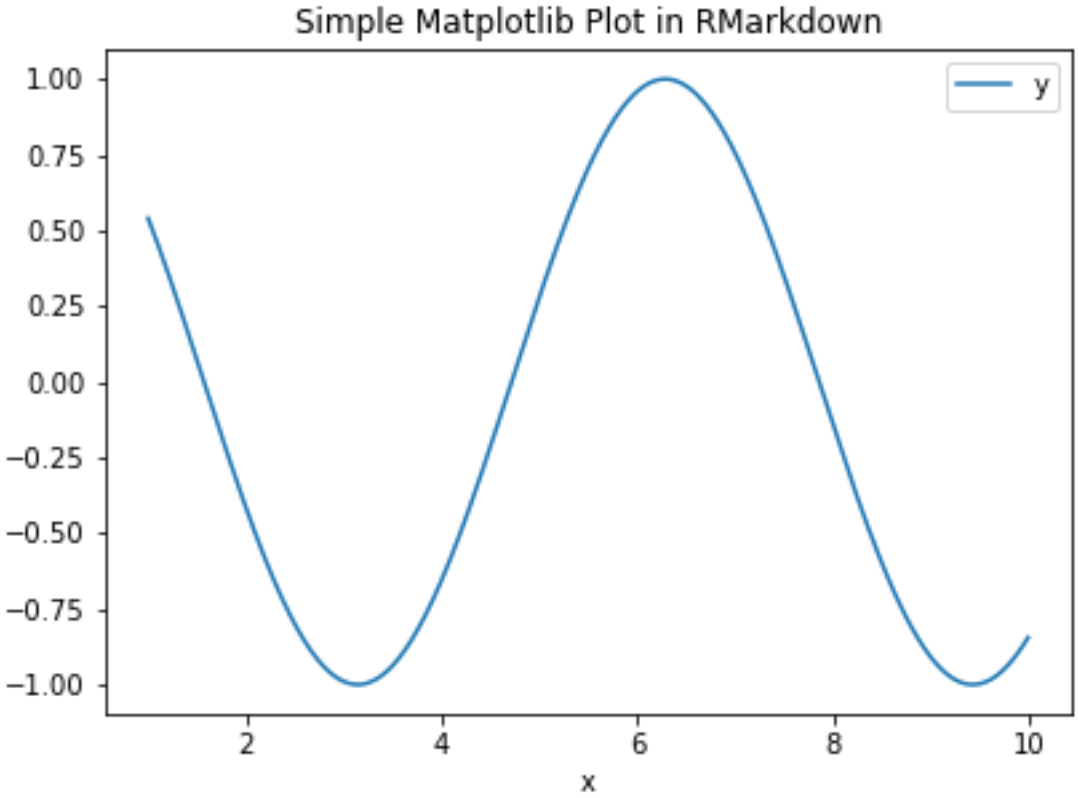
````
'''{python}
import numpy as np
import pandas as pd
np.arange(1, 10)
'''
````

array([1, 2, 3, 4, 5, 6, 7, 8, 9])

````
For the convenience of Python coding in RMarkdown, it is recommended that you create a shortcut to generate t
To create shortcut go to `Tools -> Modify Keyboard Shortcuts` and search `python` word to find `Insert Chunk
Then click on it and set a suitable shortcut for it. `Ctrl + Alt + P` sounds a good option.

In the following example, we use the `pandas` package to construct a data frame and `matplolib` to plot the d
````

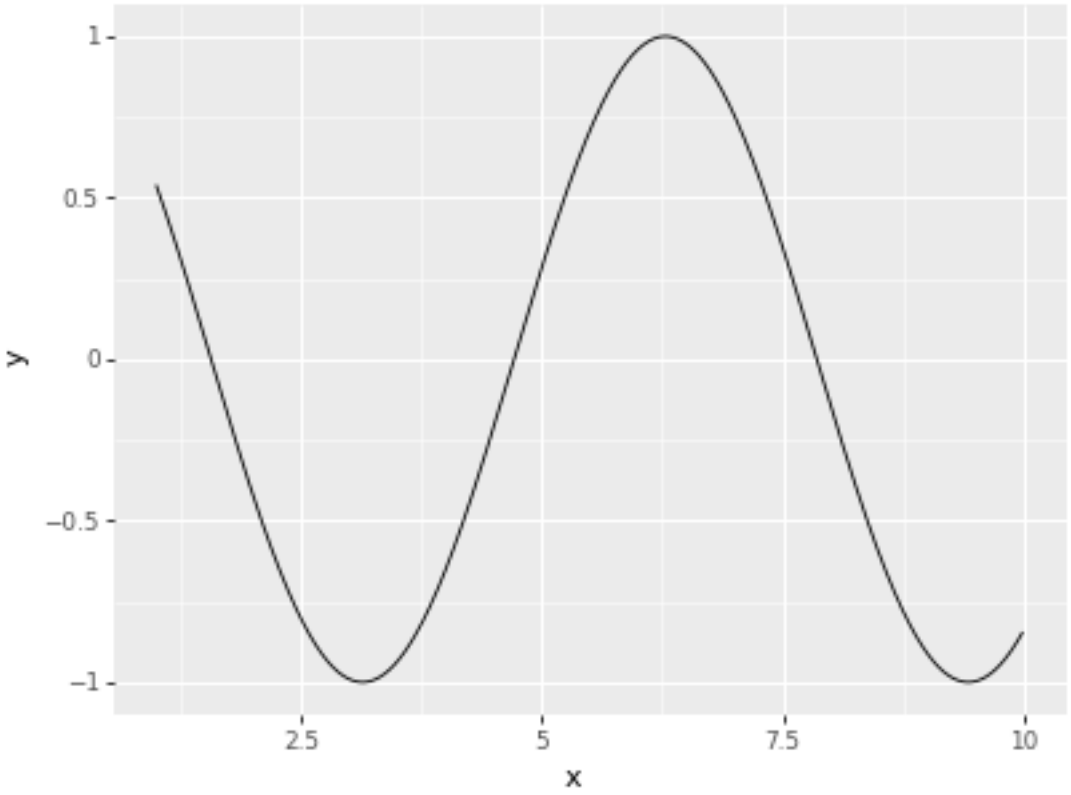
````
```python
import matplotlib as plt
df = pd.DataFrame(data = {"x":np.arange(1,10,.01)})
df = df.assign(y=np.cos(df["x"]))
df.plot(x="x", y = "y", title = "Simple Matplotlib Plot in RMarkdown")
````
````



Fortunately, Package `ggplot2` has also been implemented for Python. Below is an example of drawing a ggplot in Python

```
from plotnine.data import economics
from plotnine import ggplot, aes, geom_line
ggplot(df) + aes(x="x", y="y") + geom_line()
```

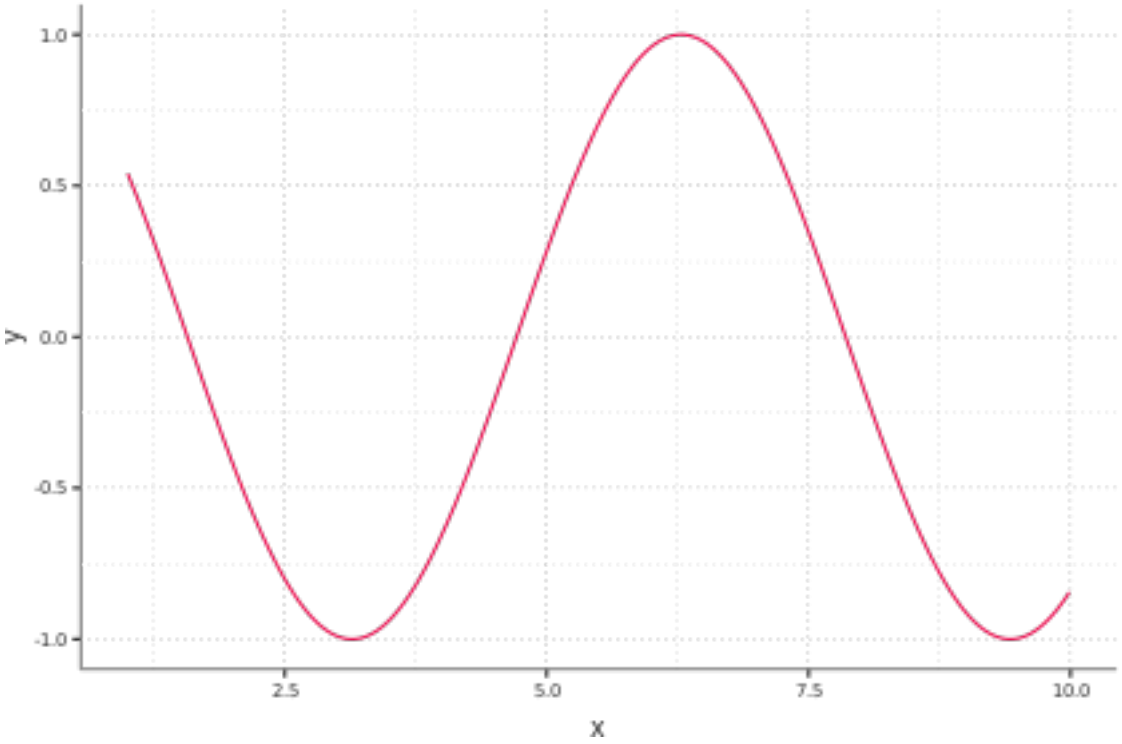
```
<ggplot: (-9223633304618956698)>
```



Calling Python from R

All objects created within Python chunks are available to R by calling the `py$object`. In the below example, we plot the previously Python-produced data with ggplot in R.

```
ggplot(py$df, aes(x=x, y=y)) +
  geom_line()
```


1.0
0.5
0.0
-0.5
-1.0
y
2.5
5.0
7.5
10.0
X