R meets Python in the Cloud

Harald Schilly

2013-12-03

R vs. Python

R is a vertically specialized environment for statistics. It is based on the domain specific **S** language (mid 70-ties), has a lot of packages for statistics and supporting functionalities.

Python is a general purpose language, designed for beginners (clear syntax, easy to start), very popular (Class A, Tiboe Index), excellent support for extensions and also generally accepted in engineering and scientific areas.

Remarkably, the latest incarnation of S ("~"–formulas and `data.frame`) and the first version of Python appeared both in 1991.

Common Ground

R is primarily an REPL command-line interpreter, which can also run scripts. It manages its own “world” of objects.

In order to interface with R:

- data-structures: need to be passed in and out.
- functions: need to be called from the “outside” in order to change R’s internal state on the “inside”.
- plots: are created in R and need to be made available.
- speed&reliability: everything should happen fast and reliably.

RPy2

There is one famous Python library to accomplish this: **RPy2**

It is a Python module and it is compiled against R's C Code. That makes it a robust low-level interface.

R's environment for variable states, the functions, operators, libraries/-packages, and other elements are exposed as Python objects.

Exchanging data from and to R is based on converting basic data types, ordered lists, strings, and most notably NumPy Arrays.

R meets Python: RPy2: Example

```
# evaluate arbitrary code
>>> summary = rpy2.robjects.r("...") # .reval(...)

# a function reference
>>> summary = rpy2.robjects.r['summary']

# calling `seq` directly
>>> rpy2.robjects.r.seq(1,10)

# after activating NumPy Array conversion:
>>> xx = np.array([5,4,2.2,-1,-5.5])
>>> summary(xx)
```

IPython

IPython is probably the most widely used interface to Python for scientific purposes. It's excellent for rapid prototyping, exchanging ideas and results, and publishing: e. g. NBViewer renders them on-line.

Relevant here: a so called **R-"magic"-extension** exists, based on RPy2, which helps to conveniently talk to R.

IPython: Commands

- `%R` runs a line of R code: `result = %R ...`
- `%%R -i <invar> -o <outvar> ...` runs the entire cell in R while converting the given input and output variables.
- `%Rpush <var>` sends the data of a given variable to R
- `%Rpull <var>` gets the variable and Python's namespace is populated
- `%Rget <var>` is similar, only retrieves the actual data.

IPython: Example

Simple example:

```
%%R
a <- seq(10) + 5.5
print(summary(a))
```

gives

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
6.50    8.75   11.00   11.00   13.25   15.50
```

⇒ All examples shown as part of this talk are here

Sagemath Cloud

The Sagemath Cloud is a web-based online environment for all sorts of computations. It's not only running the Sage mathematical software system, but also R, LaTeX, IPython and much more. Just open a Linux terminal and explore its capabilities ☺

All projects can be *shared with collaborators*, edits in files are *synchronized* in real-time, and your work is *continuously backed-up*.

This talk and the examples are part of the Sagemath Cloud examples repository.

Links

- **RPy2**
- IPython
  - R-“magic”-extension
  - IPython Rmagic example
  - IPython Documentation Example
- Sagemath Cloud