



R news and tutorials contributed by (573) R bloggers

- [Home](#)
- [About](#)
- [RSS](#)
- [add your blog!](#)
- [Learn R](#)
- [R jobs](#)
- [Contact us](#)

Welcome!

Follow @rbloggers { 30.9K

Here you will find daily **news and tutorials about R**, contributed by over 573 bloggers.

There are many ways to **follow us** -

[By e-mail:](#)

28725 readers

BY FEEDBURNER

[On Facebook:](#)



R blogg..
32 K mentions J'

Soyez le premier de vos amis à aimer ça.



If you are an R blogger yourself you are invited to [add your own R content feed to this site](#)

(**Non-English R** bloggers should add themselves- [here](#))

[Jobs for R-users](#)

- [Principal Data Engineer @ London](#)
- [Insights Data Analyst](#)
- [Research Assistant](#)
- [Data Scientist for facebook @ Tel Aviv](#)
- [Data Scientist for ARMUS @ California](#)

Popular Searches

- [web scraping](#)
- [heatmap](#)
- [maps](#)
- [twitter](#)
- [time series](#)

- [boxplot](#)
- [animation](#)
- [shiny](#)
- [hadoop](#)
- [how to import image file to R](#)
- [ggplot2](#)
- [ggplot](#)
- [trading](#)
- [excel](#)
- [finance](#)
- [latex](#)
- [eclipse](#)
- [googlevis](#)
- [rattle](#)
- [pca](#)
- [quantmod](#)
- [Rstudio](#)
- [sql](#)
- [knitr](#)
- [market research](#)
- [regression](#)
- [tutorial](#)
- [coplot](#)
- [rcmdr](#)
- [Map](#)

Recent Posts

- [Battle Maps Using R and Leaflet](#)
- [Red herring bites](#)
- [ShinyDevCon videos now available](#)
- [Shiny JavaScript Tutorials](#)
- [R Tools for Visual Studio 3.0 now available](#)
- [Announcing new forecastHybrid package](#)
- [RcppArmadillo 0.6.700.6.0](#)
- [Student-Run Conference in Data Science](#)
- [a Simpson paradox of sorts](#)
- [RZabbix – easy and direct communication with ‘Zabbix API’](#)
- [vtreat cross frames](#)
- [R 3.3.0 now available](#)
- [Manipulate\(d\) Regression!](#)
- [BioC 3.3: NEWS of my packages](#)
- [Need user feedback? Send it directly from R](#)

Other sites

- [Statistics of Israel](#)
- [SAS blogs](#)
- [Jobs for R-users](#)

Integrating Python and R into a Data Analysis Pipeline – Part 1

October 7, 2015

By [Mango Blogger](#)

Like Share { 8

Tweet

Share

71

(This article was first published on [Mango Solutions](#), and kindly contributed to [R-bloggers](#))

For a conference in the R language, EARL London 2015 saw a surprising number of discussions about Python. I like to think that at least some of this was to do with the fact that the day before the conference, we ran a 3-hour workshop outlining various strategies for integrating Python and R.

This is the first in a series of three blog posts that:

- outline the basic strategy for integrating Python and R;
- run through the different steps involved in this process; and
- give a real example of how and why you would want to do this.

This post kicks everything off by:

- covering the reasons why you may want to include both languages in a pipeline;
- introducing ways of running R and Python from the command line; and
- showing how you can accept inputs as arguments and write outputs to various file formats.

Why "And" not "Or"?

From a quick internet search for articles about "R Python", of the top 10 results, only 2 discuss the merits of using both R and Python rather than pitting them against each other. This is understandable; from their inception, both have had very distinctive strengths and weaknesses. Historically, though, the split has been one of educational background: statisticians have preferred the approach that R takes, whereas programmers have made Python their language of choice. However, with the growing breed of data scientists, this distinction blurs:

Data Scientist (n.): Person who is better at statistics than any software engineer and better at software engineering than any statistician. — [twitter @josh_wills](#)

With the wealth of distinct library resources provided by each language, there is a growing need for data scientists to be able to leverage their relative strengths. For example:

Python tends to outperform R in such areas as:

- **Web scraping and crawling:** though *rvest* has simplified web scraping and crawling within R, Python's *beautifulsoup* and *Scrapy* are more mature and deliver more functionality.
- **Database connections:** though R has a large number of options for connecting to databases, Python's *sqlalchemy* offers this in a single package and is widely used in production environments.

Whereas R outperforms Python in such areas as:

- **Statistical analysis options:** though Python's combination of *Scipy*, *Pandas* and *statsmodels* offer a great set of statistical analysis tools, R is built specifically around statistical analysis applications and so provides a much larger collection of such tools.
- **Interactive graphics/dashboards:** *bokeh*, *plotly* and *intuitics* have all recently extended the use of Python graphics onto web browsers, but getting an example up and running using *shiny* and *shiny dashboard* in R is faster, and often requires less code.

Further, as data science teams now have a relatively wide range of skills, the language of choice for any application may come down to prior knowledge and experience. For some applications – especially in prototyping and development – it is faster for people to use the tool that they already know.

Flat File "Air Gap" Strategy

In this series of posts we are going to consider the simplest strategy for integrating the two languages, and step through it with some examples. Using a flat file as an air gap between the two languages requires you to do the following steps.

1. Refactor your R and Python scripts to be executable from the command line and accept command line arguments.
2. Output the shared data to a common file format.
3. Execute one language from the other, passing in arguments as required.

Pros

- Simplest method, so commonly the quickest
- Can view the intermediate outputs easily
- Parsers already exist for many common file formats: CSV, JSON, YAML

Cons

- Need to agree upfront on a common schema or file format
- Can become cumbersome to manage intermediate outputs and paths if the pipeline grows.
- Reading and writing to disk can become a bottleneck if data becomes large.

Command Line Scripting

Running scripts from the command line via a Windows/Linux-like terminal environment is similar in both R and Python. The command to be run is broken down into the following parts,

```
<command_to_run> <path_to_script> <any_additional_arguments>
```

where:

- `<command>` is the executable to run (`Rscript` for R code and `Python` for Python code),
- `<path_to_script>` is the full or relative file path to the script being executed. Note that if there are any spaces in the path name, the whole file path must be enclosed in double quotes.
- `<any_additional_arguments>` This is a list of space delimited arguments parsed to the script itself. Note that these will be passed in as strings.

So for example, an R script is executed by opening up a terminal environment and running the following:

```
Rscript path/to/myscript.R arg1 arg2 arg3
```

A Few Gotchas

- For the commands `Rscript` and `Python` to be found, these executables must already be on your path. Otherwise the full path to their location on your file system must be supplied.
- Path names with spaces create problems, especially on Windows, and so must be enclosed in double quotes so they are recognised as a single file path.

Accessing Command Line Arguments in R

In the above example where `arg1`, `arg2` and `arg3` are the arguments parsed to the R script being executed, these are accessible using the `commandArgs` function.

```
## myscript.R

# Fetch command line arguments
myArgs <- commandArgs(trailingOnly = TRUE)

# myArgs is a character vector of all arguments
print(myArgs)
print(class(myArgs))
```

By setting `trailingOnly = TRUE`, the vector `myArgs` only contains arguments that you added on the command line. If left as `FALSE` (by default), there will be other arguments included in the vector, such as the path to the script that was just executed.

Accessing Command Line Arguments in Python

For a Python script executed by running the following on the command line

```
python path/to/myscript.py arg1 arg2 arg3
```

the arguments `arg1`, `arg2` and `arg3` can be accessed from within the

Python script by first importing the `sys` module. This module holds parameters and functions that are system specific, however we are only interested here in the `argv` attribute. This `argv` attribute is a list of all the arguments passed to the script currently being executed. The first element in this list is always the full file path to the script being executed.

```
# myscript.py
import sys

# Fetch command line arguments
my_args = sys.argv

# my_args is a list where the first element is the
# file executed.
print(type(my_args))
print(my_args)
```

If you only wished to keep the arguments parsed into the script, you can use list slicing to select all but the first element.

```
# Using a slice, selects all but the first element
my_args = sys.argv[1:]
```

As with the above example for R, recall that all arguments are parsed in as strings, and so will need converting to the expected types as necessary.

Writing Outputs to a File

You have a few options when sharing data between R and Python via an intermediate file. In general for flat files, CSVs are a good format for tabular data, while JSON or YAML are best if you are dealing with more unstructured data (or metadata), which could contain a variable number of fields or more nested data structures.

All these are very common [data serialisation formats](#), and parsers already exist in both languages. In R the following packages are recommended for each format:

- [readr](#) for CSV files
- [jsonlite](#) for JSON files
- [yaml](#) for YAML files

And in Python:

- [csv](#) for CSV files
- [json](#) for JSON files
- [PyYAML](#) for YAML files

The `csv` and `json` modules are part of the Python standard library, distributed with Python itself, whereas `PyYAML` will need installing separately. All R packages will also need installing in the usual way.

Summary

So passing data between R and Python (and vice-versa) can be done in a single pipeline by:

- using the command line to transfer arguments, and
- transferring data through a commonly-structured flat file.

However, in some instances, having to use a flat file as an intermediate data store can be both cumbersome and detrimental to performance. In the next post, we will look at how R and Python can directly call each other and return the output in memory.

Like Share 8 Tweet Share 71

Related

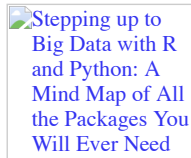


[Overview of R](#)

Workshops at London

EARL 2015

In "R bloggers"



Stepping up to Big Data with R and Python: A Mind Map of All the Packages You Will Ever Need

In "R bloggers"

Integrating Python and R Part II – Executing R from Python and Vice Versa

By Chris Musselle In a previous article we went over why you might want to integrate both R and Python into In "R bloggers"

8

Tweet

71

Like

Share

Share

To leave a comment for the author, please follow the link and comment on their blog: [Mango Solutions](#).

R-bloggers.com offers **daily e-mail updates** about R news and **tutorials** on topics such as: [Data science](#), [Big Data](#), [R jobs](#), visualization ([ggplot2](#), [Boxplots](#), [maps](#), [animation](#)), programming ([RStudio](#), [Sweave](#), [LaTeX](#), [SQL](#), [Eclipse](#), [git](#), [hadoop](#), [Web Scraping](#)) statistics ([regression](#), [PCA](#), [time series](#), [trading](#)) and more...

If you got this far, why not **subscribe for updates** from the site? Choose your flavor: [e-mail](#), [twitter](#), [RSS](#), or [facebook](#)...

Like Share

8

Tweet

Share

71

Comments are closed.

Search & Hit Enter

Recent popular posts

- [How to write the first for loop in R](#)
- [R tutorials](#)
- [R – GPU Programming for All with 'gpuR'](#)
- [ShinyDevCon videos now available](#)

Most visited articles of the week

1. [How to write the first for loop in R](#)
2. [Installing R packages](#)
3. [R 3.3.0 is released!](#)
4. [R tutorials](#)
5. [How to perform a Logistic Regression in R](#)
6. [In-depth introduction to machine learning in 15 hours of expert videos](#)
7. [Using apply, sapply, lapply in R](#)
8. [Cross-Validation for Predictive Analytics Using R](#)
9. [Computing and visualizing PCA in R](#)

Sponsors

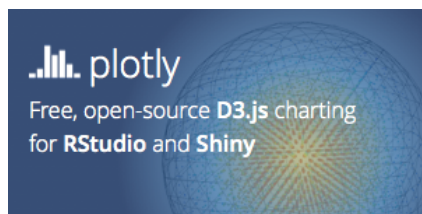




Learn R By Doing
www.DataCamp.com

Start Free Course

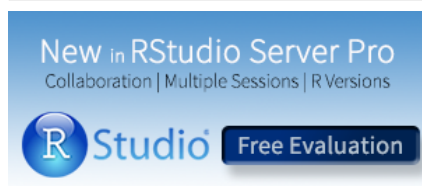
The banner features a blue background with a white R console window on the left showing code and a line plot on the right.



plotly
Free, open-source **D3.js** charting
for **RStudio** and **Shiny**

The banner has a dark blue background with a glowing, abstract geometric pattern.

[Plotly: collaborative, publication-quality graphing.](#)



New in **RStudio Server Pro**
Collaboration | Multiple Sessions | R Versions

RStudio Free Evaluation

The banner is light blue with the RStudio logo and a dark blue button.



Highland Statistics Ltd.
Statistics courses in Australia
June - August 2016

Perth, Adelaide, Melbourne,
Canberra, Sydney, Alice Springs

The banner features a photo of two koalas on the left.



Quantide: statistical consulting and training



Get hands on with R.
Start learning now!

Start Learning Now

udemy

The banner has a blue background with yellow text and a yellow button.



Werden Sie zum Expe[R]ten mit der
R-Akademie von

eoda
daten » wissen » nutzen

Beratung | Software
Training | Lösungen

The banner features the eoda logo and a green box with white text.



STATISTICS
VIEWS

Bringing Statistics Together

The banner has a white background with a bar chart icon.

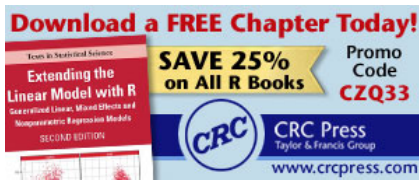


NYC DATA SCIENCE ACADEMY

Become a Data Scientist
Develop expertise in **R, Python, Hadoop & Spark**
in just 12 Weeks

Apply for Data Science Bootcamp

The banner features a group photo of students and a red button.



[Contact us](#) if you wish to help support R-bloggers, and place your banner here.

[Jobs for R users](#)

- [Principal Data Engineer @ London](#)
- [Insights Data Analyst](#)
- [Research Assistant](#)
- [Data Scientist for facebook @ Tel Aviv](#)
- [Data Scientist for ARMUS @ California](#)
- [Data Services Adjunct Librarian: Adjunct Quantitative Data Analysis Specialist](#)
- [Data Scientist @ England](#)

Search & Hit Enter

[Full list of contributing R-bloggers](#)

[R-bloggers](#) was founded by [Tal Galili](#), with gratitude to the [R](#) community.

Is powered by [WordPress](#) using a [bavotasan.com](#) design.

Copyright © 2016 **R-bloggers**. All Rights Reserved. [Terms and Conditions](#) for this website