Using reticulate to read and write NumPy files

Dirk Eddelbuettel[1]

[1] http://dirk.eddelbuettel.com

This version was compiled on April 5, 2019

This vignette shows how to use the reticulate package to directly access the NumPy module for Python.

Motivation

The **RcppCNPy** package by Eddelbuettel and Wu (2016) provides a simple and reliable access to **NumPy** files. It does not require Python as it relies on the **cnpy** library which is connected to R with the help of **Rcpp Rcpp** (Eddelbuettel and François, 2011; Eddelbuettel, 2013; Eddelbuettel *et al.*, 2018).

Now, thanks to the **reticulate** package by Allaire *et al.* (2018), we can consider an alternative which does not require **cnpy**–but which requires Python. We can (on a correctly set up machine, how to do that is beyond the scope of this note but described in the reticulate documentation) use Python to read **NumPy** data via **reticulate**.

This short note reproduces all the examples in the primary **RcppCNPy** vignette, but using **reticulate** instead of **cnpy**.

Simple Examples

```
# load reticulate and use it to load numpy
library(reticulate)
np <- import("numpy")

# data reading
mat <- np$load("fmat.npy")
mat
#       [,1] [,2] [,3] [,4]
#  [1,]  0.0  1.1  2.2  3.3
#  [2,]  4.4  5.5  6.6  7.7
#  [3,]  8.8  9.9 11.0 12.1

vec <- np$load("fvec.npy")
vec
#  [1] 0.0 1.1 2.2 3.3 4.4
```

Integer data can be read the same way:

```
imat <- np$load("imat.npy")
imat
#       [,1] [,2] [,3] [,4]
#  [1,]    0    1    2    3
#  [2,]    4    5    6    7
#  [3,]    8    9   10   11
```

Compressed Files

The gzip Python module allows us to access compressed files.

```
# use the gzip modules for compressed data
gz <- import("gzip")
# use it to create handle to uncompressed file
mat2 <- np$load(gz$GzipFile("fmat.npy.gz","r"))
mat2
#       [,1] [,2] [,3] [,4]
#  [1,]  0.0  1.1  2.2  3.3
#  [2,]  4.4  5.5  6.6  7.7
#  [3,]  8.8  9.9 11.0 12.1
```

Saving Files

Similarly, files can be saved via reticulate access to **NumPy**.

```
tfile <- tempfile(fileext=".npy")
set.seed(42)
m <- matrix(sort(rnorm(6)), 3, 2)
m
#           [,1]     [,2]
#  [1,] -0.564698 0.404268
#  [2,] -0.106125 0.632863
#  [3,]  0.363128 1.370958
np$save(tfile, m)

m2 <- np$load(tfile)
m2
#           [,1]     [,2]
#  [1,] -0.564698 0.404268
#  [2,] -0.106125 0.632863
#  [3,]  0.363128 1.370958

all.equal(m, m2)
#  [1] TRUE
```

Savez Array Files

We can also access `savez` files.

First we save two vectors two different ways:

```
x <- seq(1, 10)
y <- sin(x)
np$savez("file1.npz", x, y)
np$savez("file2.npz", x=x, y=y)
```

We can access these files with and without names:

```
npz1 <- np$load("file1.npz")
npz1$files
#  [1] "arr_1" "arr_0"
npz1$f[["arr_0"]]
#   [1]  1  2  3  4  5  6  7  8  9 10
npz1$f[["arr_1"]]
#   [1]  0.841471  0.909297  0.141120 -0.756802
#   [5] -0.958924 -0.279415  0.656987  0.989358
#   [9]  0.412118 -0.544021
```

Ditto for the second file:

```
npz2 <- np$load("file2.npz")
npz2$files
#  [1] "y" "x"
npz2$f[["x"]]
#   [1]  1  2  3  4  5  6  7  8  9 10
npz2$f[["y"]]
#   [1]  0.841471  0.909297  0.141120 -0.756802
#   [5] -0.958924 -0.279415  0.656987  0.989358
#   [9]  0.412118 -0.544021
```

Three-dimentional Arrays

We can also import three-dimensional array from **NumPy** as the next example shows.

```
arr <- np$load("arr.npy")
arr
#  , , 1
#
#       [,1] [,2] [,3]
#  [1,]    0    2    4
#  [2,]    6    8   10
#  [3,]   12   14   16
#  [4,]   18   20   22
#
#  , , 2
#
#       [,1] [,2] [,3]
#  [1,]    1    3    5
#  [2,]    7    9   11
#  [3,]   13   15   17
#  [4,]   19   21   23
```

Summary

While the **RcppCNPy** package provides functions for the simple reading and writing of **NumPy** files, we can also use the **reticulate** package to access the **NumPy** functionality directly from R.

RcppCNPy remains an attractive option for simple, direct and lighter-weight file imports whereas **reticulate** by by Allaire *et al.* (2018) shines a more full-featured access to many more aspects of Python.

References

Allaire J, Ushey K, Tang Y (2018). *reticulate: Interface to 'Python'*. R package version 1.9, URL https://CRAN.R-project.org/package=reticulate.

Eddelbuettel D (2013). *Seamless R and C++ Integration with Rcpp*. Use R! Springer, New York. ISBN 978-1-4614-6867-7.

Eddelbuettel D, François R (2011). "Rcpp: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. URL http://www.jstatsoft.org/v40/i08/.

Eddelbuettel D, François R, Allaire J, Ushey K, Kou Q, Russel N, Chambers J, Bates D (2018). *Rcpp: Seamless R and C++ Integration*. R package version 0.12.17, URL package=Rcpp.

Eddelbuettel D, Wu W (2016). "RcppCNPy: Read-Write Support for NumPy Files in R." *Journal of Open Source Software*, **1**. URL http://dx.doi.org/10.21105/joss.00055.