"Multilingual" Programming for Psychological Research

Jingmeng Cui
BSRM student, BSI
4 Sep. 2020

R is nice...

but it isn't always the best choice.

Two Disadvantages of R:

- R is mainly designed for statisticians so it lacks some packages for other uses.
- R is slow *especially when you have loops that run millions of times*.

Fortunately,

Python and C++ can help!

Advantages of Python and C++:

- Python is a multipurpose language and used by people from all disciplines.
- C++ is FAST! (but difficult to write...)

And you don't have to write your entire code with an unfamiliar language

because you can use Python and C++

within R!

Link R and Python: `reticulate`

https://rstudio.github.io/reticulate

In order to use `reticulate`, you need to first install Python and the modules you need.

Example

I need to read some SCR (skin conductance response) data, but I could only find a package in Python (`bioread`)

```r
library(reticulate)
bioread <- import("bioread") # import the module
acqfile <-  bioread$read_file('physio-5.0.1.acq')
# use read_file function from the module to read in the file

acqfile # see the info
# AcqKnowledge file (rev 132): 4 channels, 2000.0 samples/sec
```

Example (Cont.)

```r
acqdata1 <- acqfile$channels[[1]]$data # retrive the data from the first channel

str(acqdata1)
# num [1:123787(1d)] -101 -101 -101 -101 -101 ...
# acqdata1, retrived from a Python object, is now an R vector
# so you can use R to handle the remaining things
# e.g.:

mean(acqdata1)
# [1] -110.7604
```

Link R and C++: `Rcpp`

http://www.rcpp.org/

In order to use `Rcpp`, you need to first install Rtools...

... and also be able to write a C++ function.

(the example will be easy!)

Example

I need to calculate something that loops billions of times, but R is to slow for that.

Two files are needed to use Rcpp. The first one is a C++ file:

add_one_en.cpp

Example (Cont.)

```cpp
#include <Rcpp.h>
using namespace Rcpp;

// [[Rcpp::export]]
NumericVector add_one(int time) {
  // calculate a vector. the first element is 0,
  // and every element afterwards is equal to the previous element plus one
  NumericVector x(time+1);
  x(0) = 0;
  for(int i = 1; i <= time; i++){
    x(i) = x(i-1) + 1;
  }
  return x;
}
```

Example (Cont.)

And the second one is an R file:

```r
library(Rcpp)
sourceCpp("add_one_en.cpp") # source the C++ function
add_one(100000000) # run the function
```

The running time for the Rcpp function is **1.55 secs** on my computer...
... but for the R function with the same purpose, it needs **18.91 secs**!

Rcpp is more than 10 times faster for tasks like this.

Take Home Messages

- R is nice, but Python is more versatile and C++ is faster.
- You can link different languages to make your code works better!

The codes in this presentation can be found at
https://github.com/Sciurus365/RPyCpp

Some additional resources for Rcpp
http://adv-r.had.co.nz/Rcpp.html
https://teuder.github.io/rcpp4everyone_en/

The slides are powered by Marp

Thanks for your attention

Have fun in coding!