When discussing how to speed up slow R code, my first question is what is your computer spec? It’s always surprised me that people are wondering why analysing big data is slow, yet they are using a five-year-old cheap laptop. Spending a few thousand pounds would often make their problems disappear. To quantify the impact of the CPU on analysis, was the package **benchmarkme**. The aim of this package is to provide a set of benchmarks routines and data from past runs. You can then compare your machine, with other CPUs.

The package is now on CRAN and can be installed in the usual way

# R 3.5.X only

install.packages("benchmarkme")

The benchmark\_std() function assesses numerical operations such as loops and matrix operations. This benchmark contains two main benchmarks

* benchmark\_std(): this benchmarks numerical operations such as loops and matrix operations. The benchmark comprises three separate benchmarks: prog, matrix\_fun, and matrix\_cal.
* benchmark\_io(): this benchmarks reading and writing a 5 / 50, MB CSV file.

**The benchmark\_std() function**

This benchmarks numerical operations such as loops and matrix operations. This benchmark comprises three separate benchmarks: prog, matrix\_fun, and matrix\_cal.

If you have less than 3GB of RAM (run get\_ram() to find out how much is available on your system), then you should kill any memory hungry applications, e.g. Firefox, and set runs = 1 as an argument.

To benchmark your system, use

library("benchmarkme")

## Increase runs if you have a higher spec machine

res = benchmark\_std(runs = 3)

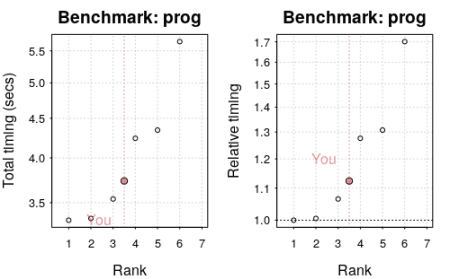
and upload your results

## You can control exactly what is uploaded. See details below.

upload\_results(res)

You can compare your results to other users via

plot(res)



**The benchmark\_io() function**

This function benchmarks reading and writing a 5MB or 50MB (if you have less than 4GB of RAM, reduce the number of runs to 1). Run the benchmark using

res\_io = benchmark\_io(runs = 3)

upload\_results(res\_io)

plot(res\_io)

By default, the files are written to a temporary directory generated

tempdir()

which depends on the value of

Sys.getenv("TMPDIR")

You can alter this to via the tmpdir argument. This is useful for comparing hard drive access to a network drive.

res\_io = benchmark\_io(tmpdir = "some\_other\_directory")

As before, you can compare your results to previous results via

plot(res\_io)

**Parallel benchmarks**

The benchmark functions above have a parallel option – just simply specify the number of cores you want to test. For example to test using four cores

res\_io = benchmark\_std(runs = 3, cores = 4)

**Previous versions of the package**

This package was started around 2015. However, multiple changes in the byte compiler over the last few years has made it very difficult to use previous results. Essentially, the detecting if and how the byte compiler was being used became near on impossible. Also, R has just “got faster”, so it doesn’t make sense to compare benchmarks between different R versions. So we have to start from scratch (I did spend a few days trying to salvage something but to no avail).

The previous data can be obtained via

data(past\_results, package = "benchmarkmeData")

**Machine specs**

The package has a few useful functions for extracting system specs:

* RAM: get\_ram()
* CPUs: get\_cpu()
* BLAS library: get\_linear\_algebra()
* Is byte compiling enabled: get\_byte\_compiler()
* General platform info: get\_platform\_info()
* R version: get\_r\_version()

**Uploaded datasets**

A summary of the uploaded datasets is available in the benchmarkmeData package

data(past\_results\_v2, package = "benchmarkmeData")

A column of this data set contains the unique identifier returned by the upload\_results() function.

**What’s uploaded**

Two objects are uploaded:

1. Your benchmarks from benchmark\_std or benchmark\_io;
2. A summary of your system information (get\_sys\_details()).

The get\_sys\_details() returns:

* Sys.info();
* get\_platform\_info();
* get\_r\_version();
* get\_ram();
* get\_cpu();
* get\_byte\_compiler();
* get\_linear\_algebra();
* installed.packages();
* Sys.getlocale();
* The benchmarkme version number;
* Unique ID – used to extract results;
* The current date.

The function Sys.info() does include the user and nodenames. In the public release of the data, this information will be removed. If you don’t wish to upload certain information, just set the corresponding argument, i.e.

upload\_results(res, args = list(sys\_info = FALSE))