Intro to Parallel Computing in R

Kevin Lee

Department of Statistics
Western Michigan University

February 22, 2019
1. List of Useful R Packages

2. Introduction to Parallel Computing in R
List of Useful R Packages

Some of the top most downloaded R packages:

Outline

1. List of Useful R Packages

2. Introduction to Parallel Computing in R
Why Parallel Computing?

- Many statistical analysis tasks are computationally intensive.
- But at the same time, many problems are “embarrassingly parallel”.
- And often we have multiple cores in our computer!
- However, R only uses a single core.
Embarrassingly Parallel Problems

Easy to speed things up when:

- Calculating similar things many times (e.g. iterations in a loop).
- Calculations are independent of each other.
- Each calculation takes a decent amount of time.
How Does Parallel Computing Work?

**Serial Approach**
- **Start**
- For $k = 1 : N$
- Evaluate model
- **Stop**

**Parallel Approach**
- **Start**
- Evaluate model
- Evaluate model
- Evaluate model
- Evaluate model
- **Stop**

...workers...
Motivation Example: Birthday Problem

What is the probability that in a group of $n$ people at least two have the same birthday?
Key Motivation: Speeding up R
Key Motivation: Speeding up R
Repeated executions can be done manually, but it becomes quite tedious to execute repeated operations.

R comes with various looping constructs that solve this problem. The for loop is one of the more common looping constructs, but the repeat and while statements are also quite useful.

In addition, there is the family of “apply” functions, which includes apply, lapply, sapply, tapply, and others.
The `foreach` package provides a new looping construct for executing R code repeatedly.

The main reason for using the `foreach` package is that it supports parallel execution.

It can execute those repeated operations on multiple processors/cores on your computer, or on multiple nodes of a cluster.
The `doParallel` package is a “parallel backend” for the `foreach` package.

It provides a mechanism needed to execute `foreach` loops in parallel.

The `foreach` package must be used in conjunction with a package such as `doParallel` in order to execute code in parallel.
We will perform a simulation study to check the performance of bootstrap estimate of standard error of sample mean on different sample sizes.

We first generate random samples of size 10, 50, and 100 from normal distribution with mean 0 and standard deviation 2. Next we find the bootstrap estimate of standard error of sample mean for each sample size with 2000 bootstrap samples.

We repeat this 100 times and calculate the root-mean-squared error (RMSE) to compare the performance of bootstrap estimate of standard error of sample mean on different sample sizes.

$$\text{RMSE} = \sqrt{\frac{1}{R} \sum_{r=1}^{R} \left( \text{SE}(\bar{x}) - \hat{\text{SE}}_r(\bar{x}^*) \right)^2}$$
A rule of thumb: If you can wrap your task in an `apply` function or one of its variants then you can also use parallel computing!

Check how many cores your laptop or desktop has and start using parallel computing in R!
References

- L. Collado-Torres’s website
  - http://lcolladotor.github.io/

- Steve Weston’s documents.
  - Using the foreach Package
  - Getting Started with doParallel and foreach