BIFX552 Midterm Exam

Randy Johnson

10/12/2017

# Instructions

This is the same package we put together as a class in early September, with a few additions. The package currently resides on the server at */home/johnson/transpose*. Clone the repository to your local machine and modify as instructed below.

For specifics on how this exam will be graded, see the rubric below.

The exam consists of three tasks worth 50 points each. Check your progress often using the R CMD CHECK transpose command, and commit chnages when appropriate (note: some things will be picked up by R CMD CHECK even though they are excluded in *.Rbuildignore* - these warnings are OK to ignore). You won't need to push any commits to the source repository, since you don't have write permissions to my directory. When you are finished, use scp to upload your modified git repository to */home/johnson/midterm/transpose\_username*, where *username* is your username. If you would prefer, I can verify that the upload was successful, since you don't have permissions to view what is in the *midterm* directory.

## transpose\_vector() (50 pts)

Your first task is to create a new function called transpose\_vector(). This should take the same input and return the same output as transpose\_loop(), but use vectorization instead of a nested for loop.

Remember:

* Your R package should contain both functions and all appropriate documentation and NAMESPACE entries.
* Make sure your function works before commiting.

## Testing (50 pts)

The second task is to add some testing checks to the package to help ensure that users get reasonable errors when they misuse your functions and to make future changes to your package less buggy.

### User checks

Our code currently assumes that x is a matrix, but there is no validation of this assumption. Add a function to the package called validate\_x that checks if the user submitted a matrix. This should be called at the beginning of both transpose\_loop and transpose\_vector.

validate\_x

* Input: a matrix, x
* Use class(x) to verify that x is a matrix.
* If class(x) is not a matrix, use stop() to throw an error with a meaningful error message.
* Output: none

Once you complete this function move to the package checks (don't forget to commit this change to the repository).

### Package checks

When running R CMD CHECK transpose, the *testing/testthat.R* script checks that everything is in order. There are currently two checks in the *testing/testthat* directory:

* Verify that transpose\_loop(x) works when x is a matrix (done).
* Verify that an error is thrown when no input is given to transpose\_loop() (done).

Add a new testthat file with two more to checks:

* Verify that transpose\_vector(x) works when x is a matrix.
* Verify that an error is thrown when x is a data.frame.

## Documentation (50 pts)

Finish filling in the next section on how to use this package. Make your edits in README.Rmd and knit the file to create an updated Word document.

# Using this package

This package was developed by the BIFX 552 class at Hood college to practice R package building and to demonstrate the potential code speedup when using comparing nested loops, vectorized code, and compiled code.

## Installation

To install this package, you currently need to navigate to the directory containing either the source code or the compiled R package and install using R CMD INSTALL. You can download the package from */home/johnson/transpose* on the BIFX server at Hood college.

## Comparison of transpose with base R

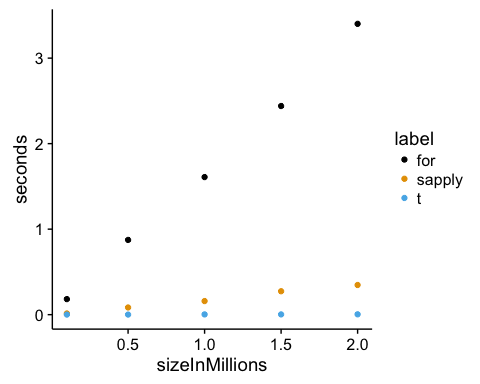
R is optimized for vector operations. For loops, especially nested for loops, are known for being inefficient. To demonstrate this inefficiency, we have created this package with two functions replecating the functionality of the transpose function, t():

* transpose\_loop() transposes a matrix using a nested for loop.
* transpose\_vector() uses vector operations to transpose the matrix.

The average speedup across the 5 sizes tested in this example is:

|  |  |  |
| --- | --- | --- |
| Method | Speedup over for | Speedup over sapply |
| sapply | 10.4 |  |
| t | 768.5 | 78.8 |

This figure shows the observed compute times for each method as the matrix size increases by 500,000 elements.



# Grading

This exam will be graded on a curve as this is the first time this exam has been administered. I anticipate that there will be about 2 A's and 3 B's, but if you all score similarly, I'm happy to give everyone A's. You may use any resources you have available to you, but you are expected to complete the exam on your own.

## Rubric

|  |  |  |
| --- | --- | --- |
| Tasks | Skill level | Points |
| Code | Appropriate comments | 10% |
|  | Appropriate indentation | 10% |
|  | Use of white space | 6% |
|  | Appropriate vectorization | 10% |
|  | File header | 4% |
|  | Runtime errors in code | -16% |
|  | Task incomplete | -20% |
|  |  | **40%** |
| Git repository | Proper structure | 10% |
|  | Commit message formatting | 6% |
|  | Commit only complete work | 6% |
|  | Clean repository history | 4% |
|  | Commit only related changes | 4% |
|  | Small commits | 4% |
|  | Appropriate use of *.gitignore* | 6% |
|  |  | **40%** |
| R Package | Installs without errors or warnings | 10% |
|  | Documentation complete | 6% |
|  | Appropriate NAMESPACE entries | 4% |
|  |  | **20%** |