Analysis of CSV File Reading in R

Quang Nguyen Robert Tedesco

Loyola University Chicago

STAT 407 Fall 2021

Abstract

This manuscript is a statistical investigation into the impact of different factors on the time elapsed when importing CSV files into the `R` statistical computing language. We designed a 3^3 factorial experiment with 20 replicates where our variables of interest were computer model, CSV file size, and `R` package/function. Our main findings are: 1) the `rio` package resulted in significantly lower mean CSV reading times compared to the functions from the packages `readr` and `data.table`; 2) the Lenovo device had significantly lower mean reading time compared to the MacBook and Dell laptops; and 3) the average importing time differ for the three CSV files of sizes 0.49 GB, 1.22 GB, and 1.96 GB that we considered. Furthermore, a follow-up ANCOVA showed that `readr` is the most efficient package for loading CSV files into R on a MacBook device.

1 Introduction

Data importing is undoubtedly a crucial part of a modern statistics and data science workflow, as it is the first step that paves the path for important stages such as data wrangling, visualizing, and modeling. With modern computers, loading data may seem like a straightforward task for a statistician. However, as the volume of data increases, a new set of challenges are imposed related to computing time and efficiency for data importing. A researcher may have to load a large data file multiple times a week in order to perform statistical analyses, which could potentially cost them valuable time. Identifying the fastest computing techniques is a necessity in the computational problem of working with big data.

The `R` programming language (R Core Team, 2021) is well-known among statisticians and data scientists as a powerful tool for working with data. In `R`, a popular method used by traditional statisticians to load a comma-separated values (CSV) file into the environment is `read.csv()`, which is a built-in base function. However, there are some drawbacks with this method, when working with large data sets. As demonstrated by Matthews (2021), it took minutes to import a CSV file of about two-gigabyte containing over a million rows of NFL tracking data with the historical `read.csv()` function in base `R`. Fortunately, in recent years, `R` developers have come up with better ways to read in data files. As mentioned by Gillespie & Lovelace (2021) in their "Efficient R programming" book, there are three modern functions/packages that `R` users should take advantage of in order to improve file-reading performance.

The first function that `R` programmers should consider is `import()`, which comes from the `rio` package (Chan et al., 2021). As described by its creators in the package documentation, `rio` is "a swiss-army knife for data I/O." The function `import()` in `rio` essentially simplifies the data importing process since as a function by itself, `import()` has the flexibility of reading in multiple file extensions, such as `.json`, `.xls`, `.xlsx`, in addition to the common `.csv`. The data is stored as a `data.frame` object after being loaded into `R` in using `import()`.

Another `R` package for importing data that has gotten a lot of attention recently is `readr` (Wickham & Hester, 2021), which is also part of the popular `tidyverse` collection of packages (Wickham et al., 2019) for modern data science. Within `readr`, there is a `read_csv()` function for getting CSV files into `R`, alongside other functions of the `read_*()` family designed for other file extensions. A notable aspect of the functions

in `readr` in general and `read_csv()` in particular is the data is stored as a `tibble` (a "modern" `data.frame`-typed object) in the `R` environment once it is read in.

The third and final package for efficient file reading in `R` is `data.table` (Dowle & Srinivasan, 2021), which contains the `fread()` function for data importing. As an individual function, `fread()` is also capable of loading a variety of file extensions, similar to `rio::import()`. As for object type, the `fread()` function imports and returns a data object of classes `data.table` and `data.frame`.

In this paper, we attempt to design a three-factor factorial experiment with replicates and perform analyses to compare the speed of the three CSV importing algorithms in `R` as mentioned in previous paragraphs. In particular, the independent variables we are interested in examining are 1) the `R` package/function; 2) the file size; and 3) the computer device. The paper is outlined as follows. We first describe our experiment and data generating process in Section 2. We then spend the next two section, 3 and 4, on our analyses and results of this experiment. Lastly, in Section 5, we give a quick summary of our main findings and discuss possible future work related to this project.

2 Experiment

The goal of this analysis is to determine the effects of three factors, `R` package, file size, and computer device, on the reading time for CSV files measured in seconds. For this reason, we decided to design our experiment with a 3^3 factorial experiment (Montgomery, 2012) in mind. Furthermore, we chose to include $n = 20$ replicates for each combination of our factors. Our chosen design implies that the total number of observations for our experiment is 540.

The three levels for each of our factors are described as follows. For our first factor, R functions and their associated packages, we considered the three package-function combinations as introduced in Section 1. This includes `rio::import()`, `data.table::fread()`, and `readr::read_csv()`. As for our second variable, file size, we first utilized `R` to generate and export three CSV files of different sizes consisting of columns of randomly-draw samples from a standard normal distribution. First, the largest file consists of 1000000 rows and 100 columns and has a size of 0.49 GB. Next, we have a mid-sized file of 1.22 GB on disk containing 1000000 cases and 62 variables. Finally, our smallest CSV file has a row-by-column dimension of 252000 by 100 and takes up 1.96 GB of storage. Last but

not least, we considered three different laptop devices for this experiment. In particular, we have a MacBook Air 2020 with an M1 processor, a Lenovo Legion 2019 with a Ryzen 7 4800H processor, and a Dell XPS 13 2020 with an Intel i7 processor.

Our data was generated by considering the 3^3 possible combinations of `R` package, file size, and computer. We obtained the twenty replicates for each combination according to the order of a random sample of 1 through 3^3. For each combination of reading function, file size, and device, we utilized the `R` function `Sys.time()` to collect the time it took to load the CSV file into our statistical software.

3 Analysis of Variance

In this section, we present the analysis of variance for our designed experiment. First, in order to gain a better understanding of our data, we used interaction plots (see Figure 1) to observe that reading time varies for combinations of each factor. As for file size, importing time seemed to increase significantly for all three packages when working with the largest data file (~1.96GB). `readr` seems to take longer than the other two packages considered when working with large data files. Interestingly, The Lenovo machine appeared to be faster than the MacBook and Dell devices for all file sizes. Lastly, the interaction of computer and package appears to be significant, as the MacBook seemed to read in data quicker than the other computers when using the `readr` package.

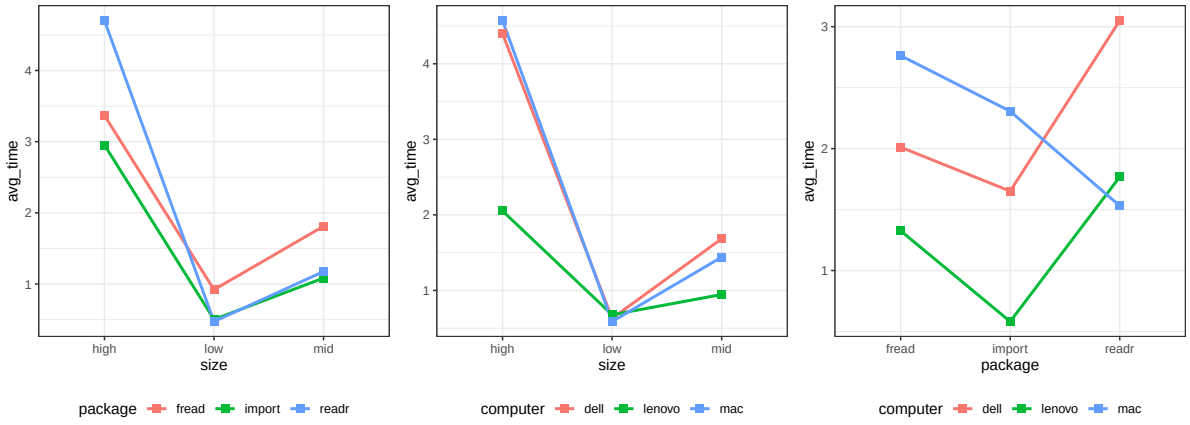

Figure 1: Interaction plots between the variables size, computer, and package

After gaining an understanding of our data, the next step was to fit the model

corresponding to our experiment. The model for a three-level factorial experiment with three factors is given by

$$y_{ijkl} = \mu + \tau_i + \beta_j + \gamma_k + (\tau\beta)_{ij} + (\tau\gamma)_{ik} + (\beta\gamma)_{jk} + (\tau\beta\gamma)_{ijk} + \epsilon_{ijkl}, \tag{1}$$

where $i, j, k = 1, 2, 3$ and $l = 1, 2, ..., 20$, since we have $n = 20$ replicates.

The response variable, y_{ijkl}, indicates the speed time in seconds corresponding to each run conducted in the experiment. Here μ represents the overall mean reading time, and τ_i, β_j, and γ_k are the main effects of three factors file size, computer device, and `R` function/package. There are also 3 two-factor and 1 three-factor interaction effects in our model. In addition, this model includes an error component ϵ_{ijkl}, and the assumptions for the residuals are that they are normally distributed and the variance is constant across the groups.

Using this model, we can conduct an ANOVA to see whether the levels in each factor significantly differ in the mean data loading time. In addition, we can test for significance of each of the specified interaction effects. The model produced significant results (see Table 1), as all the terms in the ANOVA fit appear to have an effect on the response variable, data importing time. However, the residual assumptions are not met for this model, as illustrated by Figure 2 and Table 2. In particular, a normal probability plot followed by a Shapiro-Wilk test indicated a significant departure from normality for our model residuals. Moreover, a residuals versus fitted values plot and a Levene Test showed a violation of the equality of variances condition.

Table 1: ANOVA for the three-factor factorial ANOVA model.

term	df	sumsq	meansq	statistic	p.value
computer	2	118.616	59.308	12.802	0.000
size	2	908.985	454.493	98.103	0.000
package	2	38.497	19.249	4.155	0.016
computer:size	4	136.355	34.089	7.358	0.000
computer:package	4	114.576	28.644	6.183	0.000
size:package	4	87.988	21.997	4.748	0.001
computer:size:package	8	144.365	18.046	3.895	0.000
Residuals	513	2376.621	4.633	NA	NA

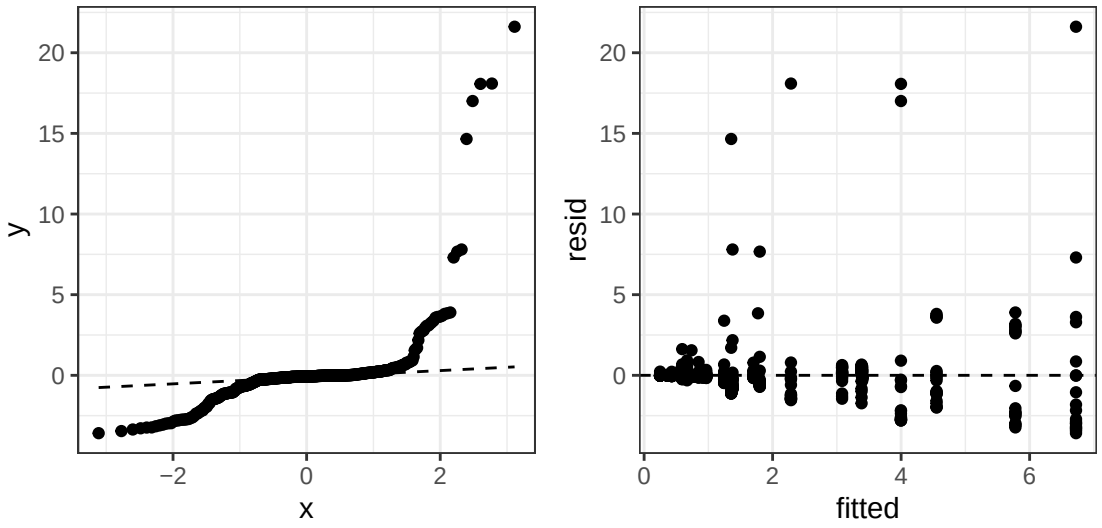

Figure 2: Residuals plots

Table 2: Shapiro-Wilk and Levene tests for the three-factor factorial ANOVA model.

statistic	p.value	method
0.439	0.000	Shapiro-Wilk normality test
3.941	0.000	Levene test for homogeneity of variance

As a result of violations of normality and homoscedasticity, we then considered a couple of data transformation methods, namely, a natural log transformation and a Box-Cox power transformation (Box & Cox, 1964) as possible ways to resolve the issues of our data. However, transforming the response variable, file reading time, did not improve the fit of this model and our assumptions are still not met. The ANOVA table, visual summary, and tests for normality and constant variance for these two methods can be found in the Appendix.

This then led us to look into a nonparametric alternative to the 3^3 factorial model. We ended up using a permutation test for a three-way design, which does not assume a normal distribution for our data. This is implemented via the function `perm.fact.test()` in the `asbio R` package (Aho, 2021). Table 3 shows the computer output for the permutation test for our three-factor factorial experiment. Since each factor and all interaction effects are significant, a key takeaway here is that statisticians have a lot to keep in mind as they work with big data. Certain machines may excel with certain packages and what package is best for reading data depends on the size of the file at hand.

Table 3: Permutation test for the three-factor factorial ANOVA model.

Term	InitialF	DF	pval
size	98.103	2	0.000
computer	12.802	2	0.000
package	4.155	2	0.012
size:computer	7.358	4	0.000
size:package	4.748	4	0.001
computer:package	6.183	4	0.000
size:computer:package	3.895	8	0.000
Residual	NA	513	NA

We then performed a pairwise permutation test to determine which pairs of levels in each of our three factors, file size, computer device, and function/package, differ significantly in terms of average file reading time. This was implemented using the `pairwisePermutationTest()` function in the `rcompanion` `R` package. At 10% significance level, we observed that the two pairs of functions `data.table::fread()` - `rio::import()` and `rio::import()` - `readr::read_csv()` have different mean file importing time. As for computer devices, Dell and MacBook don't differ in data loading time, but they each have different mean reading times compared to Lenovo. Lastly, there is a difference in file importing time for all pairs of file sizes.

Table 4: Pairwise comparisons using permutation tests with FDR adjustment for the three-factor factorial ANOVA model.

factor	comparison	statistic	p.value	p.adjust
size	high - low = 0	9.536	0.0000	0.0000
size	high - mid = 0	7.316	0.0000	0.0000
size	low - mid = 0	-4.424	0.0000	0.0000
computer	dell - lenovo = 0	3.3	0.0010	0.0015
computer	dell - mac = 0	0.1327	0.8945	0.8945
computer	lenovo - mac = 0	-3.762	0.0002	0.0005
package	fread - import = 0	2.256	0.0241	0.0532
package	fread - readr = 0	-0.254	0.7995	0.7995
package	import - readr = 0	-2.103	0.0355	0.0532

4 Additional Analysis

One particularly interesting observation we had after performing ANOVA is that `readr::read_csv()` is the fastest method when running on the MacBook device, but on the other hand, the slowest for the other two computers. This trend difference is depicted on the third plot of Figure 1. Thus, we are interested in performing a quick simulation study to see if there is a function/package effect on the file reading time for the MacBook device only. In addition to the three data files with sizes 0.49, 1.22, 1.96 GB used in the previous experiment, we simulated seven additional datasets with sizes 1.51, 1.73, 2.16, 2.36, 2.61, 2.94, 3.32 GB. After that, we used each of the three methods (`read_csv()`, `import()`, `fread()`) to import the data files and record the time it took to read in for the MacBook. Figure 3 is a scatterplot of time elapsed and file size, where the points and lines are color coded by CSV reading method. The figure reveals that `readr::read_csv()` is clearly the fastest data loading technique on the MacBook device.

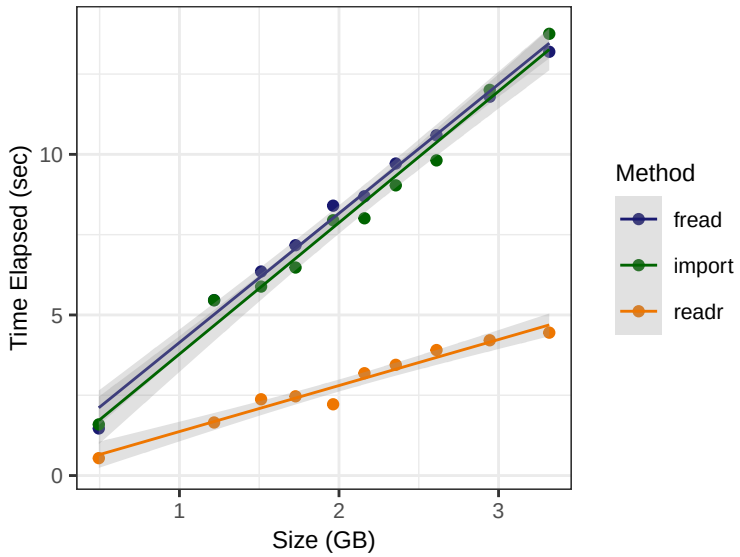

Figure 3: Scatterplot of time elapsed and file size, color coded by reading algorithm.

We then conducted an analysis of covariance (ANCOVA) with time elapsed as the response variable, the file reading function as the categorical independent variable, and the file size as the continuous covariate. The ANCOVA model is given by

$$y_{ij} = \mu + \tau_i + \beta(x_{ij} - \bar{x}_{..}) + \epsilon_{ij}, \tag{2}$$

where τ_i is the effect of the ith treatment, and β_j is a parameter associated with the jth subject.

We were interested in using ANCOVA to investigate the effect of the package/function on the time elapsed for importing CSV data. Table 5 suggests that the mean reading time differs among the CSV reading functions in `R`, after accounting for the effect of file size. Furthermore, both normality and constant variance conditions are met for this model, as illustrated by a Shapiro-Wilk test for normality and a Levene Test for homogeneity of variance (see Table 6).

Table 5: ANCOVA table for the analysis of reading time for different methods on MacBook device.

Term	SS	df	F	p-value
(Intercept)	10.5154	1	8.4657	0.0073
package	187.3580	2	75.4188	0.0000
size	192.4662	1	154.9502	0.0000
Residuals	32.2950	26	NA	NA

Table 6: Shapiro-Wilk and Levene tests for ANCOVA model.

statistic	p.value	method
0.973	0.629	Shapiro-Wilk normality test
2.363	0.113	Levene test for homogeneity of variance

We can then proceed to perform multiple comparisons to see which algorithms are different from each other. It is confirmed by Table 7 that the mean CSV file reading time on the MacBook device for `readr::read_csv()` is different (faster) than the mean time elapsed for the other two techniques, and the other two also don't differ in terms of average importing time.

Table 7: Pairwise comparisons with Tukey contrasts following ANCOVA.

contrast	estimate	se	statistic	adj p-value
import - fread	-0.2873	0.4984	-0.5764	0.8338
readr - fread	-5.4391	0.4984	-10.9127	0.0000
readr - import	-5.1518	0.4984	-10.3362	0.0000

5 Conclusion and Discussion

In this paper, we examined the effects of three factors file size, computer device, and importing function on the reading time for CSV files in the `R` statistical programming language. While we could not conduct an ANOVA on our three-level factorial experiment due to violations of normality and homoscedasticity, we were still able to find results through a nonparametric permutation test. The results of our permutation test for the corresponding factorial design indicate that at least one level for all of the factors in our experiment (`R` package/function, file size, and computer) significantly differ in average reading speed time. In particular, the `rio` package resulted in significantly different mean CSV reading times compared to the packages `readr` and `data.table`. As for laptop device, the Lenovo machine had significantly lower mean importing times compared to the MacBook and Dell laptops. Lastly, the average time elapsed for loading CSV files into `R` differ across the three file sizes of 0.49 GB, 1.22 GB, and 1.96 GB. An additional analysis using ANCOVA suggests that `readr` is the most time-efficient package for getting CSV files into R on a MacBook Air with an M1 processor.

In the wake of these results, we realize that there are numerous factors that we control for in future studies. While we ran this experiment with all three laptops on battery power, we hypothesize that there is a significant difference between file speeds for a plugged-in machine versus a device on battery power. Another variable that may be significant is the size of memory within the user's `R` environment. In addition, we could consider loading data in different environments, such as RStudio Desktop, RStudio Cloud, Google Colab Notebook, and Terminal/Command line. We could also look into the reading time for other data file formats, such as tab-separated values (`.tsv`), Excel spreadsheets (`.xlsx`), or JavaScript Object Notation (`.json`). We also wonder if there is a significant difference in the speed for importing other forms of data, such as images, raw text, or audio data. Future work may also include implementing a similar design with more than twenty replicates in order to improve the fit of the model and proceed with a parametric approach. Overall, this paper provides insight into the first step of the problem of working with big data: data importing. Moving forward, more discussion between computer scientists and statisticians will allow us to more quickly fill computational pitfalls that arise when working with high volume of data.

Supplementary Material

All materials related to this manuscript are publicly available on GitHub at https://github.com/qntkhvn/read_speed.

References

Aho, K. (2021). *Asbio: A collection of statistical tools for biologists*. https://CRAN.R-project.org/package=asbio

Box, G. E. P., & Cox, D. R. (1964). An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, *26*(2), 211–252. http://www.jstor.org/stable/2984418

Chan, C., Chan, G. C., Leeper, T. J., & Becker, J. (2021). *Rio: A swiss-army knife for data file i/o*. https://CRAN.R-project.org/package=rio

Dowle, M., & Srinivasan, A. (2021). *Data.table: Extension of 'data.frame'*. https://CRAN.R-project.org/package=data.table

Gillespie, C., & Lovelace, R. (2021). *Efficient r programming*. https://csgillespie.github.io/efficientR

Matthews, G. J. (2021). *Old dog (statistician) learns new trick (read_csv)*. https://youtu.be/E5KJkooW4RY.

Montgomery, D. C. (2012). *Design and analysis of experiments, 8th edition*. John Wiley & Sons.

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wickham, H., & Hester, J. (2021). *Readr: Read rectangular text data*. https://CRAN.R-project.org/package=readr

Appendix

Table 8: ANOVA table for the three-factor factorial ANOVA model with a natural log transformation to the response.

term	df	sumsq	meansq	statistic	p.value
computer	2	53.742	26.871	142.520	0.000
size	2	266.521	133.261	706.793	0.000
package	2	5.239	2.619	13.892	0.000
computer:size	4	11.403	2.851	15.119	0.000
computer:package	4	25.138	6.285	33.332	0.000
size:package	4	6.295	1.574	8.347	0.000
computer:size:package	8	2.717	0.340	1.801	0.074
Residuals	513	96.722	0.189	NA	NA

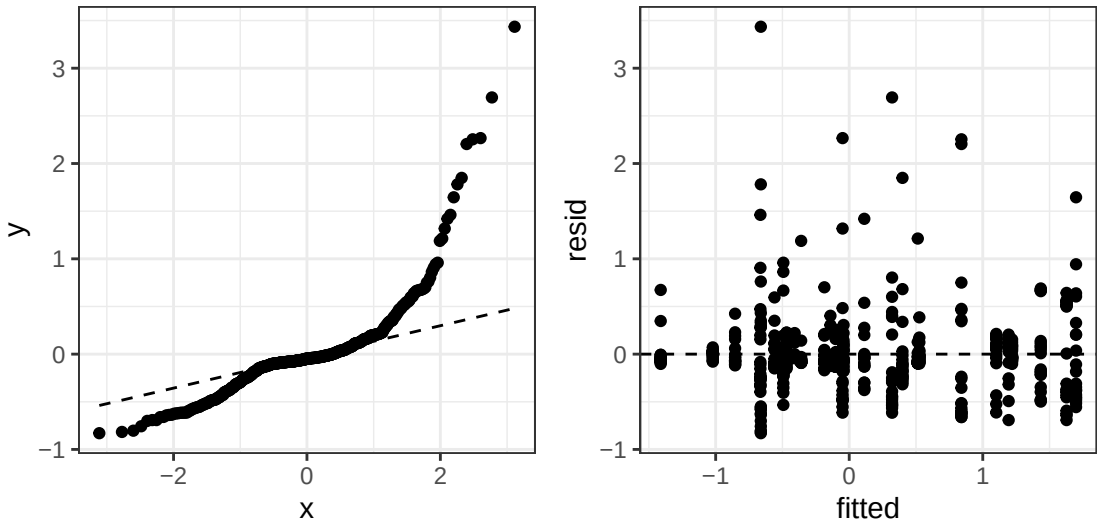

Figure 4: Residuals plots for the three-factor factorial ANOVA model with a natural log transformation to the response.

Table 9: Shapiro-Wilk and Levene tests for the three-factor factorial ANOVA model with a natural log transformation to the response.

statistic	p.value	method
0.758	0.000	Shapiro-Wilk normality test
5.150	0.000	Levene test for homogeneity of variance

Table 10: ANOVA table for the three-factor factorial ANOVA model with a Box-Cox transformation to the response.

term	df	sumsq	meansq	statistic	p.value
computer	2	3.353	1.677	191.039	0.00
size	2	15.889	7.945	905.282	0.00
package	2	0.328	0.164	18.675	0.00
computer:size	4	0.293	0.073	8.349	0.00
computer:package	4	1.616	0.404	46.048	0.00
size:package	4	0.303	0.076	8.625	0.00
computer:size:package	8	0.161	0.020	2.300	0.02
Residuals	513	4.502	0.009	NA	NA

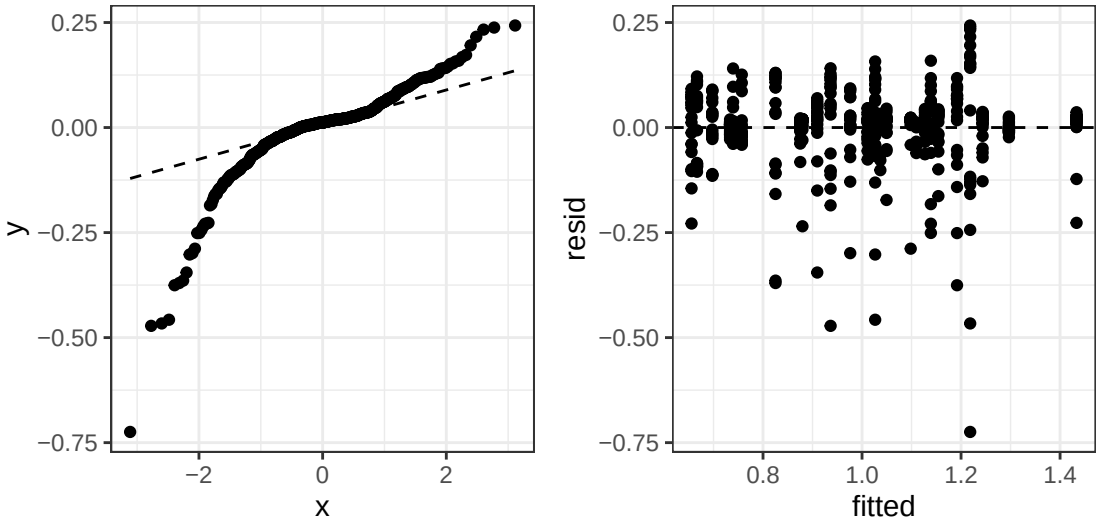

Figure 5: Residuals plots for the three-factor factorial ANOVA model with a Box-Cox transformation to the response.

Table 11: Shapiro-Wilk and Levene tests for the three-factor factorial ANOVA model with a Box-Cox transformation to the response.

statistic	p.value	method
0.808	0.000	Shapiro-Wilk normality test
6.139	0.000	Levene test for homogeneity of variance

Individual Contributions

Together we brainstormed and decided the topic, data collection process, wrote code for analyses (EDA, model fits, tests) and wrote the report. Rob wrote the abstract, experiment, analysis, and discussion sections of the report. Quang edited those sections, wrote the introduction and additional analysis sections, and formatted the report (figures, tables, citations).