CS263 Final Project: Data Science Benchmarking and Comparison of R, MATLAB, and Python

Data Science Performance Comparison and Benchmarks for Python, MATLAB, and R

Github: https://github.com/jaber-the-great/CS263_Jaber_Katie

Katherine Turnlund
dept. Computer Science
kturnlund@ucsb.edu

Jaber Daneshamooz
dept. Computer science
jaber@ucsb.edu

Abstract

We evaluated and compared R, Python, and MATLAB with 4 different benchmarks all oriented towards operations and functions heavily used in the data science field. Python greatly outperformed both MATLAB and R, with MATLAB slightly edging out R's performance. This indicates that for large tasking, Python is the best selection though significantly less accessible to the less software-central data scientist.

Index Terms

Data science, performance analysis, Python, MATLAB, R

I. Introduction

In this project, we compared the performance of data science algorithms between MATLAB, R, and Python. All of these three programming languages are interpreted languages ans widely used for data analysis; Therefore, our comparison will show which language excels and which one struggles. Since there were no good bench marking available for theses languages in data science field, we implemented some programs to test that. These programs are the ones which are frequently used in data science field.

II. Gap of knowledge

Before this project, we did not know the R language. On top of that, based on our investigation online, there doesn't seem to be a solid set of data science benchmarks that are comparable for R, MATLAB, and Python that already exist [4].

III. Overview of programming languages

A. Python

Python is a dynamically typed language and has interpreter rather than compiler. The Python byte-code has 113 opcodes(42 with arguments/operands and 71 without).

CPython is the reference implementation of the Python programming language. Written in C and Python, CPython is the default and most widely used implementation of the Python language. CPython can be

defined as both an interpreter and a compiler as it compiles Python code into bytecode before interpreting it.

CPython compiles Python code into bytecode before interpreting it makes use of a global interpreter lock (GIL) on each CPython interpreter process which means that it is single threaded.The GIL makes Python unsuitable for CPU-intensive algorithms which run across multiple cores. CPython has a fat bytecode and more work is done in handler for each bytecode [2].

B. R

The R programming language is strongly but dynamically typed. It is functional and interpreted and therefore, not compiled. R is popular amongst data scientists, because there are (free) packages with which statistical calculations (such as matrix calculations or descriptive statistics) can be performed [3].

C. MATLAB

Matlab is loosely and weakly typed as well as dynamically typed programming language. The aforementioned properties can cause problem when re-using or changing the code. Matlab is a scripted rather than compiled language and uses an interpreter. Matlab code is interpreted and optimized by JIT(Just In Time) accelerator which removes the redundant code and re-orders the commands.

One of the reasons that Matlab is slower than some of the programming languages is that it performs extra operations. For example, it checks such whether the things passed to the functions are appropriate or not. Also, sometimes Matlab performs some analysis to determine the best internal method for accomplishing the operation. This operations consume CPU cycles and make Matlab slower than its counterparts.

Matlab has a compiler as an extra toolbox for sharing. The compiler allows the client to share the code without the need of buying MATLAB for destination. Using Matlab compiler, you can deliver the Matlab program as a standalone application, web-app, docker image, excel add-in etc [1].

IV. CODE SAMPLES

A. Criteria and Variation of codes

We implemented different programs for data science benchmark over 3 different programming languages. In some of our tests, we considered java too which would a base line for comparing these 3 languages with the ones (eg Java, C, C++) that are not widely used for data science projects.

Since data science is the focus of this project, we implemented the algorithms which are the fundamental blocks of data analysis. Data cleaning is the initial step of every data analysis project. After preparing data for analysis, the data scientist perform different operations on data. Array slicing, matrix multiplication and performing basic statistical analysis are the most common operations. Beside implementing these algorithms, we also implemented linear regression which is one of the most famous machine learning algorithm.

1) Data cleaning: Our data cleaning code reads the salary a years of experience from a CSV file, and converts them to proper type (string to numeric).

2) Matrix Multiplication: The matrix multiplication code multiplies two $n * n$ matrixes. Although in many used cases of data science we multiply a vector matrix by an $m * n$ matrix, our experiment is still valid. That is because from another point of view, our code is doing vector to matrix multiplication several times.

3) Array slicing: Array slicing is used to take specific columns and/or rows are the data set. The data scientists usually take the whole data from a CSV file and need to separate the dependent and independent variables(slicing by column). Also, in many situations, the analysts need do divide the data set into training and test sets. For this purpose, they use array slicing. We implemented a simple array slicing code to compare the performance of different languages.

4) Basic stats: In many situations, the basic statistical analysis is enough and the data scientists do not need to utilize heavy ML or deep learning algorithms to learn from data. For basic statistical analysis, we calculated the average of the input dataset.

5) Linear Regression: We implemented linear regression algorithm on our dataset which finds the relation between salary and years of experience. We used skLearn for Python, Apache for java and Stats for R.

V. Benchmark

A. Strategy

We used the same system for running all of the benchmarks and made sure that the system has similar background workload while running all of the tests. Run each of the toy programs 100 times on the same system.

B. statistics

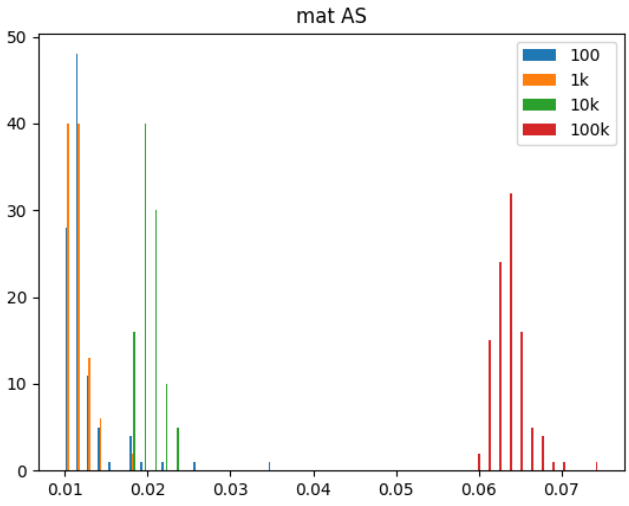

Fig. 1. Matlab Array Slicing

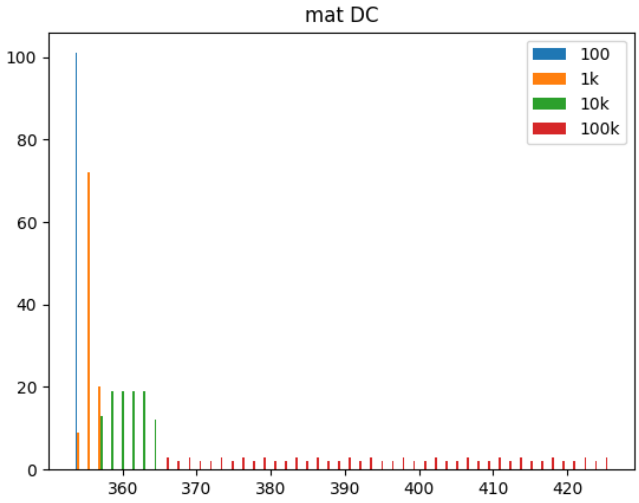

Fig. 2. Matlab Data Cleaning

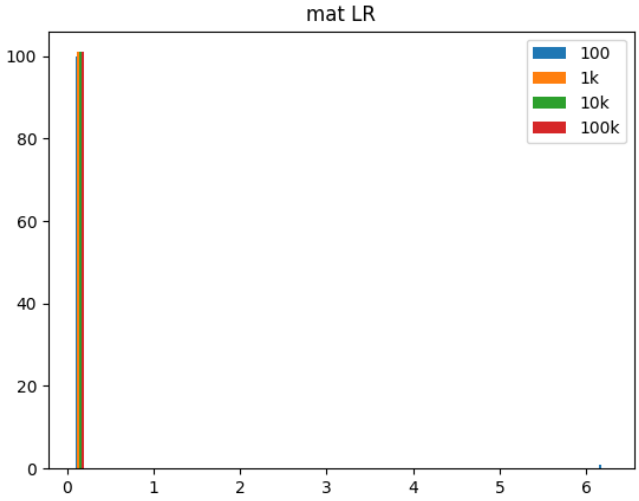

Fig. 3. Matlab Linear Regression

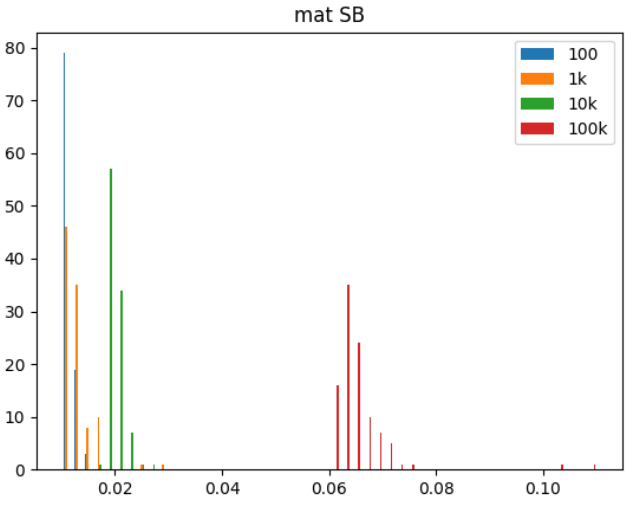

Fig. 4. Matlab Basic Stats

1) Matlab:

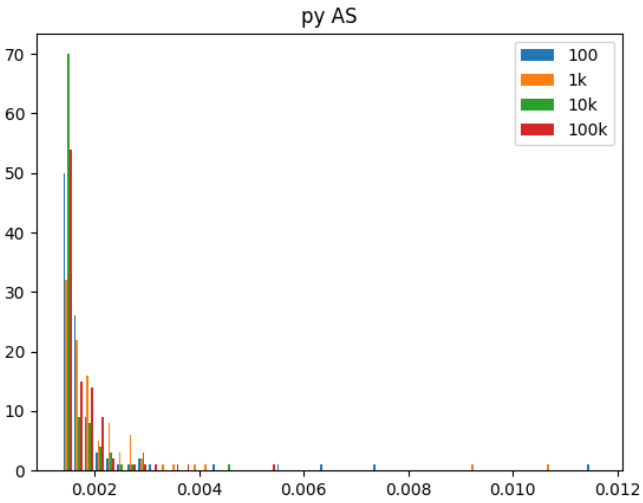

Fig. 5. Python Array Slicing

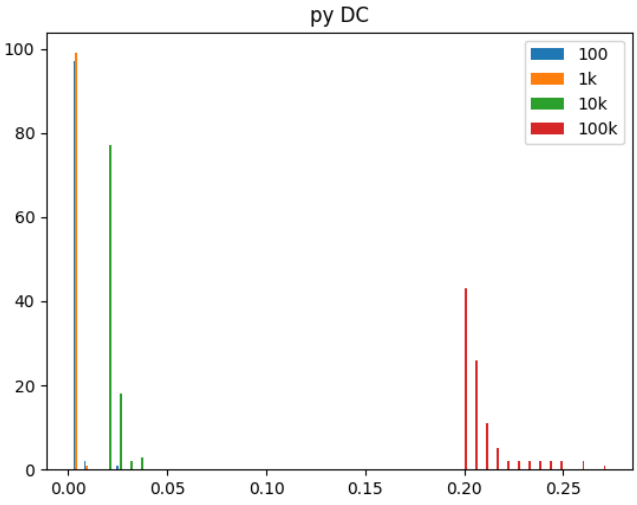

Fig. 6. Python Data Cleaning

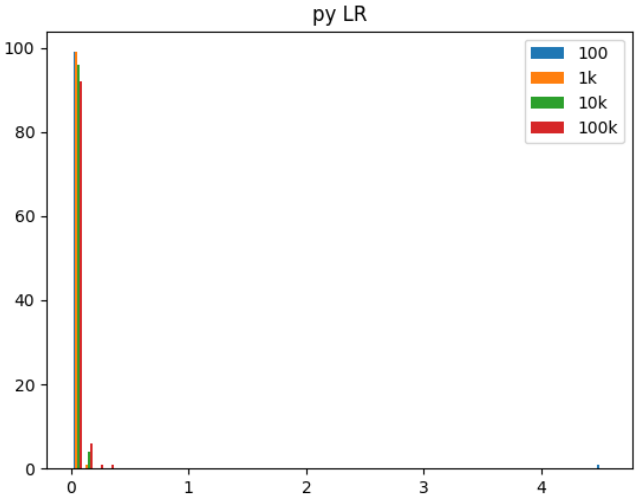

Fig. 7. Python Linear Regression

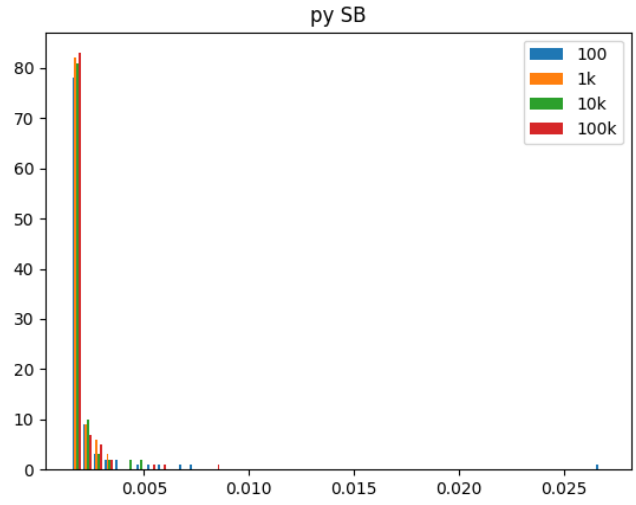

Fig. 8. Python Basic Stats

2) Python:

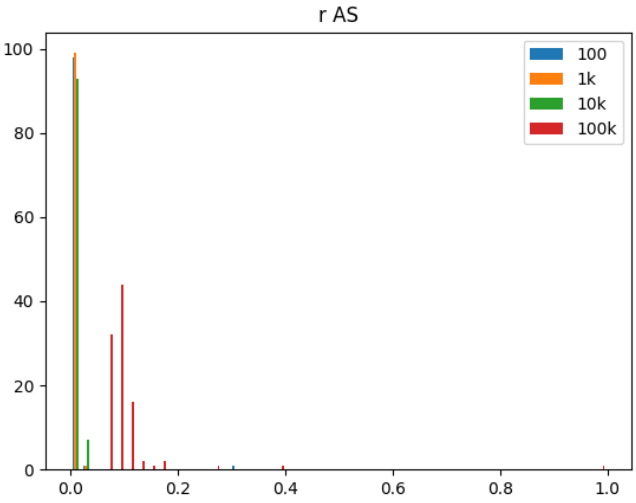

Fig. 9. R Array Slicing

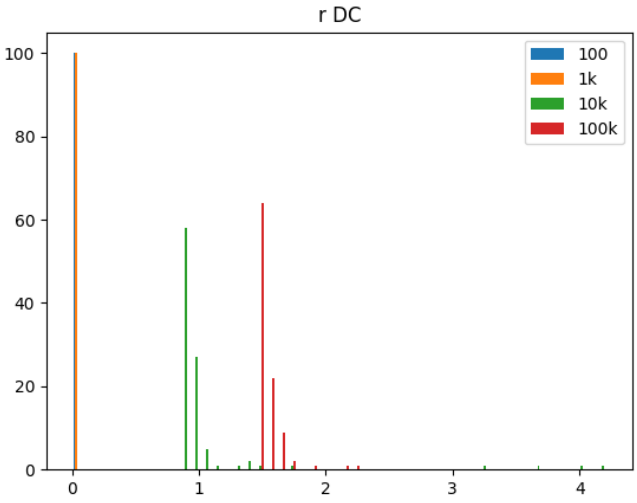

Fig. 10. R Data Cleaning

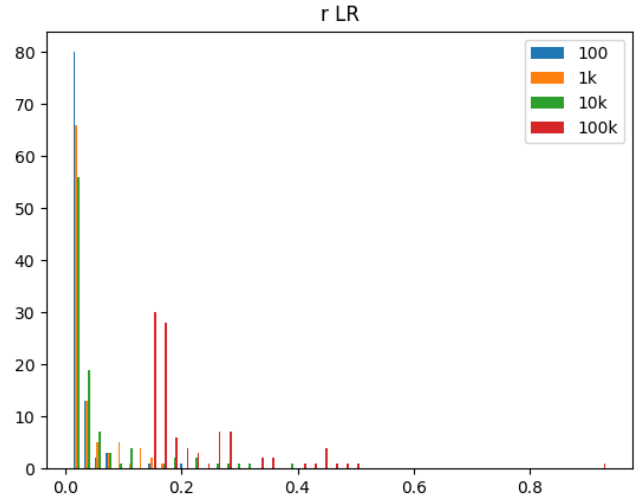

Fig. 11. R Linear Regression

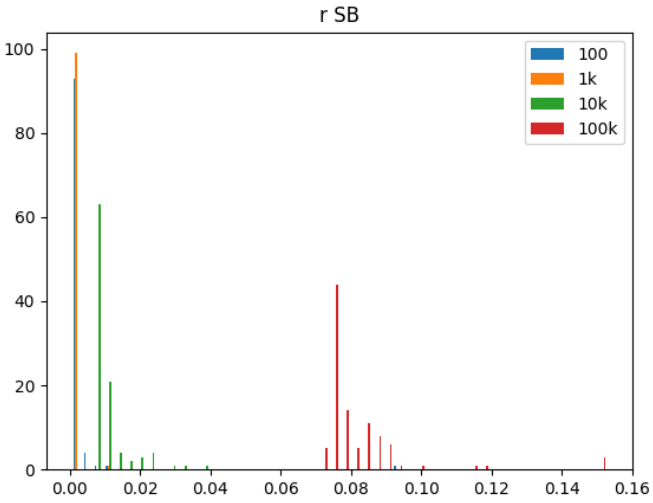

Fig. 12. R Basic Stats

3) R:

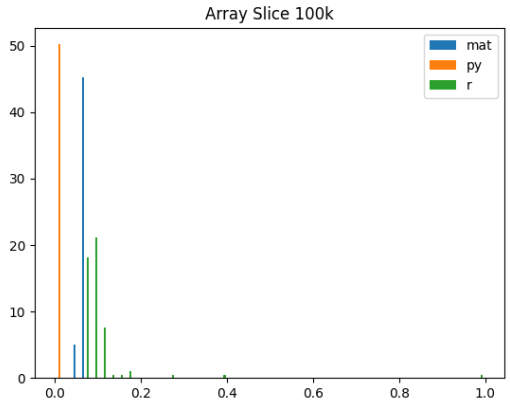

Fig. 13. Array Slicing Comparison

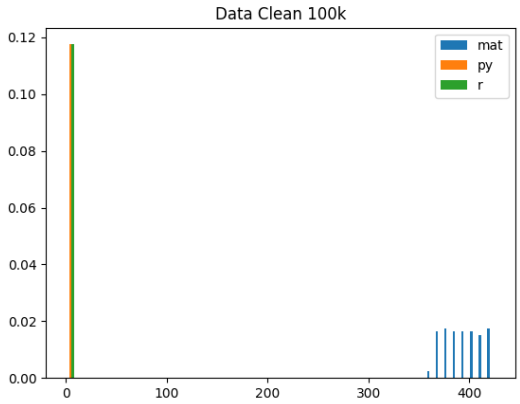

Fig. 14. Data Cleaning Comparison

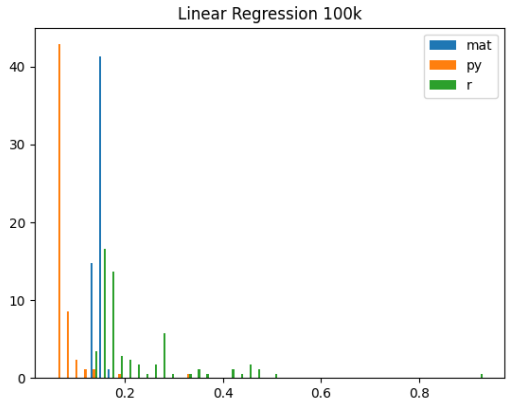

Fig. 15. Linear Regression Comparison

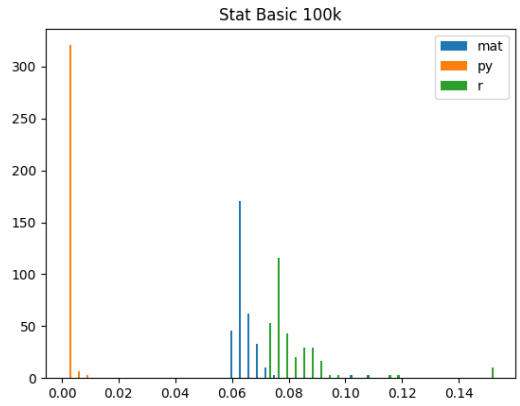

Fig. 16. Basic Stats Comparison

4) Comparison:

VI. Comparison of user experience

MATLAB is easiest out-of-the-box solutions for non-software focused data scientists. It has a very good documentation and users can perform the same task by writing fewer lines of code. The developer also made sure that the build in functions are designed in an efficient way. MATLAB is trickier for more custom solutions since it has lots of black-boxing. The thing about MATLAB is that unlike Python, you don't need to get your hands dirty of installing packages(eg Numpy, Pandas). The package installation is sometimes a nightmare for the people who do not have expertise in computer science but want to analyse their data.

It is so easy to implement data science algorithm and plot diagrams in R. You get the similar experience of using Jupyter notebook and Spyder. Not having a good and easy to use documentation is the downside of R.

Python is the most versatile the best for more custom functions but most involved to get off the ground. Plotting is more difficult and requires more package knowledge than other two

VII. Challenges and Accomplishments

The first challenge we encountered was that while Python does exist, R and MATLAB are not languages that exist on the "Benchmarksgame" website, so unfortunately we were not able to use that as a baseline comparison between the three languages. One of the main challenges in this project was benchmark equality across languages. In some part of the codes, we avoided the conventional way of doing a task in that language to make it more similar to the other implementations. Selecting useful and non-biased benchmarks and running benchmarks under the same condition was also challenging. We were also face the problem of narrowing down the entire field of data science into 5 benchmarks which cover most of the frequently used operations. MATLAB licensing problems was another issue we had.

In this project, we learned to code with R programming language. We realized that comparing to the other two programming languages, Python is very fast.

VIII. Future works

In the future, we'd like to expand into researching the various packages available for both Python and R, as they each have several packages that accomplish similar things in the data science sphere. We'd also like to spend more time optimizing R, as it is the language we taught ourselves for this research project. Hopefully with more experience, we can better optimize our benchmarks in R and improve its performance. Finally, as a stretch goal (or maybe as an aside), we'd like to investigate the memory leak that was found in the MATLAB data cleaning benchmark. Judging by the forum posts for the MATLAB documentation, this seems to be not only a problem others have encountered, but an open issue within MATLAB.

References

[1] MATLAB Compiler

[2] CPython

[3] R Documentation

[4] Benchmarksgame