Scientific Research Publishing

Journal of Software Engineering and Applications, 2023, 16, 521-529
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Improving Performance of Computer Algebra Systems

Kostas Zotos, Irena Atanassova

Department of Informatics, Faculty of Science and Mathematics, South-West University "NeofitRilski", Blagoevgrad, Bulgaria
Email: zwtos@yahoo.com, irena.atanassova@gmail.com

How to cite this paper: Zotos, K. and Atanassova, I. (2023) Improving Performance of Computer Algebra Systems. *Journal of Software Engineering and Applications*, **16**, 521-529.
https://doi.org/10.4236/jsea.2023.1610026

Received: August 3, 2023
Accepted: October 8, 2023
Published: October 11, 2023

Abstract

Computer Algebra Systems have been extensively used in higher education. The reasons are many e.g., visualize mathematical problems, correlate real-world problems on a conceptual level, are flexible, simple to use, accessible from anywhere, etc. However, there is still room for improvement. Computer algebra system (CAS) optimization is the set of best practices and techniques to keep the CAS running optimally. Best practices are related to how to carry out a mathematical task or configure your system. In this paper, we are going to examine these techniques. The documentation sheets of CASs are the source of data that we used to compare them and examine their characteristics. The research results reveal that there are many tips that we can follow to accelerate performance.

Keywords

Mathematical Software, Computer Algebra Systems, CAS, Improving Performance, CAS Performance, CAS Optimization, MATLAB, Maple, Mathematica

1. Introduction

The field of Mathematics is found in many disciplines such as Biology, Physics, Economics, Chemistry and many more. Computer technology has completely affected many scientific fields, facilitating people's daily lives. In this light, the field of Mathematics could not remain unaffected by the rise of technology. In recent decades, a variety of mathematical software have been created, which improve the learning experience of the understanding and familiarity of the interested users.

It is difficult to compare CASs. There are few studies in the literature that dare to compare them. This happens because each one specializes in a field of Mathematics and has a different philosophical approach. So, it would be a mistake to say which of them is the best or what is wrong or right. Therefore, the aim of the paper could not be to find the best CAS but to present techniques to improve the existing ones.

Different devices, platforms, and web browsers may have distinctive necessities, limitations, and features that affect how CAS runs. Huge scale numerical calculations can put overwhelming requests of computer memory and processing power. Some CASs analysts believe that do not have to optimize CASs performance because computer hardware is progressing so quickly. On the other hand, software engineers and researchers are ordinarily less tolerant of poorly designed software. In technical markets, the general philosophy is the gains of performance to rise continuously. In this paper, we will not criticize these opinions which are all partly correct, but we are going to combine them and present some tips and techniques to improve the performance of CASs.

The rest of the paper is organized as follows: Section 2 provides a general overview of Computer Algebra Systems. In Section 3, we are going to examine some tips for better performance. Next (Section 4), the results are presented and discussed whereas in the final section (Section 5) some conclusions are drawn.

2. Basic Characteristics of Famous Computer Algebra Systems

Computer Algebra Systems are software packages, which are used in manipulation of mathematical formulas. The primary goal of a Computer Algebra System is to automate tedious and sometimes difficult algebraic manipulation tasks. There are many publications that show the positive impact of CAS on students [1]. Today, they exist over fifty Computer Algebra Systems on the market. The most famous are Mathematica, Maple, MATLAB, Maxima, SageMath, SymPy, GNU Octave, Magma and MathStudio. In this paper, we will focus on MATLAB, Mathematica and Maple. The documentation sheets of these three CASs are the source of data that we used to compare them and examine their characteristics. So, the basic features of them are the following:

- **MAPLE-Maplesoft**

MAPLE is a modern, interactive mathematical software package for symbolic and numerical calculations, as well as for graphing, used in university courses as well as in research and other applications. It started as a research project at the University of Waterloo in Canada about 30 years ago, and today it has evolved into the MapleSoft company [2] and has established itself as one of the best software packages for symbolic computing. Maple is based on a kernel, written in C, which provides the Maple language.

- **Wolfram Mathematica**

Mathematica is a computing package with a lot of capabilities in almost all areas of Mathematics (e.g. Algebra, Set Theory, Analysis, Statistics, etc.) [3]. It first appeared in the late 80s as a command execution kernel that could be adapted to any operating system (e.g. Unix, MacOS, Windows, etc.). This common Kernel still exists today (improved and enriched), while its connection with the user is made through a Notebook interface.

Despite the advantages of having a common core, unfortunately there are also some disadvantages, such as quite slow processing speed (compared to pure programming languages), program instability and increased memory requirements. The low speed is mainly seen when the repeated execution of a series of commands (e.g. Loops) is requested and is mainly due to the fact that Mathematica uses an interpreter and not a compiler like classical programming languages (e.g. C, Pascal, Visual basic).

- **MATLAB-MathWorks**

It is a modern integrated mathematical package used extensively in academia and industry. It is an interactive program for numerical calculations and graphing, but it also provides programming capability, which makes it a useful tool for all sciences. Unlike the Maple and Mathematica software, MATLAB in its initial versions did not perform symbolic calculations. In its newer versions, the package includes toolkits that allow symbolic computations.

As its name suggests, MATLAB is specially designed for matrix calculations, such as solving linear systems, finding eigenvalues and eigenvectors, inverting a quadratic matrix, etc. In addition, this mathematical package is equipped with many options for graphics and programs written in its own programming language to solve other problems such as finding the roots of a non-linear equation, solving non-linear systems, solving initial value problems with ordinary differential equations, etc. Finally, there are tools for creating custom graphical user interfaces and interactive tools for iterative exploration, design, and problem-solving [4].

According to Shacham and Cutlip (1998), the comparison of mathematical software packages ensures the selection of a system, which will fully meet the needs of each user. However, in order to achieve the comparison, these software packages should be investigated with a comparative eye based on specific objective criteria [5]. These criteria are summarized as follows:

- Numerical performance
- User friendliness
- Technical support

Mathematica, MATLAB and Maple have a simple and user-friendly interface and a huge support network of their users. This is reinforced by the fact that they have help centers integrated into the software, as well as on the websites of each company. Having discussion groups, documentation, manuals, and videos can solve any user's question.

MATLAB is between 9 to 11 times slower than the best C++ executable. Mathematica is only about three times slower than C++ (after a considerable rewriting of the code). Julia has the best performance (approximately 2.70 times slower than the best C++ executable) [6].

MATLAB is more focused on numerical computing, while Wolfram Mathematica is more focused on symbolic computation. Both are powerful tools, but they are designed for different types of tasks. Maple is best for users who want a good tool basically for Mathematics and Engineering. Maple isn't supported as widely as other mathematical software (e.g. MATLAB and Mathematica). Community support is more active and richer in Mathematica and MATLAB rather than Maple.

In the following **Table 1**, we can see some of the basic characteristics of Maple, Mathematica and MATLAB. From this table we can understand why these three CASs are on the top and how difficult is to pick out only one. There are many other features (not so important) that for reasons of brevity and space are not listed in the following **Table 1**.

3. Tips for Better CAS Performance

Computational optimization means that a system, whether software or hardware, runs as fast and stable as possible. Computer Algebra System optimization is the set of best practices and techniques to keep the CAS running optimally. This section provides some tips to help you handle CAS in a way that accelerate performance. Most of this material is general and applies to all CASs, but there are tips which apply only to specific CASs. So, if you want to have better performance follow these points:

- **Pre-allocate when appropriate.** Pre-allocation ensures that matrix elements will be stored in contiguous locations in memory and therefore incurs the cost of memory allocation just once [5].
- **Place independent operations outside loops**, use vector operations instead of loops. Some CASs use processor-optimized libraries for matrix and vector computations and therefore you can accelerate performance by vectorizing code [7]. When you can't vectorize something and need to use a loop don't forget to pre-allocate. Most uses of loops can be replaced by calls to faster (built-in) procedures that perform an iteration [8].
- **Make the mathematical operations simpler.** For example, the partial correlation coefficient can be computed via regression models, or via the simple correlation matrix. The second one is much faster [9].
- **Use profiling to measure memory and time complexity** of your program and analyze the frequency/duration of function calls. With a profiler, you can easily determine which functions use a significant amount of time or which of them are called most [10].
- **Avoid overloading built-ins functions.** In some CASs there is a warning that informs you about the existence of a built-in function.
- Sometimes using a functional style (that avoids difficult to optimize side-effects) instead of an imperative style gives higher efficiency.
- **Create new variables if data type changes.** Avoid code that generates variables, it's better to load them from files, you should avoid using "data as code".

Table 1. Basic characteristics of Maple, Mathematica and MATLAB.

Parameters	Maple	Mathematica	MATLAB
User-friendly	Yes	Yes	Yes
Provides efficient and accurate solutions to complex problems	Yes	Yes	Yes
Code generation in other programming languages	Java, Perl, C#, Fortran, C, Python, Visual Basic and Python.	C	C and C++
Support of 2D image processing.	Yes	Yes	Yes
Support of 3D image processing.	Needs access to OpenGL library to draw 3-D plots.	Yes	Yes
Capability of editing documents during a computation	No	Yes	Yes
RAM requirements	Needs a good specification in RAM.	Needs high RAM	Needs a good specification in RAM.
Used for	Computations in Engineering, Quantum Chemistry, Physics and Advanced Math	Computations in Mathematics, Engineering, Chemistry, Physics, Biology, Finance and many other fields	Computations in Mathematics, Engineering, Chemistry, Physics, Biology, Finance and many other fields
Cost	High (lower prices for students)	High (lower prices for students)	Some packages are available free to use.
Best for	Users who want a good tool for Mathematics and Engineering	Users who want help in neural networking, modelling the data and visualizing simulations.	Users who want to analyze data and model them. It's best to detect fraudulent activities by analyzing data.
Chat-powered code writing	No	You can compose code by describing the task in words.	No
Autocompletion	Yes	Yes	Yes
Code reformatting	No	Yes	No
Syntax help for	Missing arguments, scoping conflicts.	Missing arguments, bracket matching, excess arguments, scoping conflicts.	Missing arguments, bracket matching, excess arguments.

- **Avoid global variables.** Choose local functions over nested functions.
- **Use functions instead of scripts.**
- Write clean code, do not create garbage (allocated storage that's not re-

quired) because this increases the work of the garbage collector and therefore the time to complete the work.

- **Use modular programming.** Separate program functions into independent pieces.
- **Use short-circuits operators** (for example & and ||), it's more efficient [11].
- Use sparse arrays when appropriate and numeric arrays when possible.
- You can speed-up applications performing parallel computing. An increasing amount of CAS operations will automatically parallelize over local cores.
- Some CASs, like MATLAB, is designed to solve problems numerically, not symbolically. Therefore, it's better for a symbolic computation to select the appropriate tool, for example Mathematica, which is faster in symbolic computations. Keep in mind that compiled languages are less prone to runtime failures than interpreted languages due to type system adherence [12].
- In matrices, it is faster to scan down columns than over rows. Column-major order implies that elements along a column are sequential in RAM while elements along a row are separated. Scanning down columns promotes cache effectiveness [13].
- Use the advanced IDEs of CASs because they offer you the capability to easily edit and navigate code in an integrated workgroup environment with a specialized editor (code folding, bracket highlighting, syntax coloring, error reporting, command completion etc.) [14].
- **Listen code analyzer suggestions**.
- Avoid unnecessary I/O. This means avoiding unnecessary reads and writes of files or data.
- **Organize and avoid losing data.** Use a unique folder for every project to keep all related files together. Use comments (especially header comments). Avoid using dangerous commands like *clear all* (MATLAB) in a script. Beware of CAS crash.
- **Automate as much as you can.** Use the appropriate tools, don't spend time to write functions that already exist in your CAS. You don't need to reinvent the wheel. Test early and test often with automation. According to a 2020-21 World Quality Report, automation testing tools can save time and minimize human errors [15].
- **If your computer doesn't meet the basic requirements (in RAM, processing power etc.) of desktop CAS version use online version.** With these online platforms you can make mathematical computations even when your computer doesn't meet the required standards of desktop version. With a simple PC you can perform any mathematical computation directly in your browser and collaborate with other CAS users by giving them the appropriate privileges (read, write, and run) that you want and share files directly. You can store files in your CAS Cloud Drive and synchronize your important math files with the Cloud Drive. Moreover, you could access the latest version of CAS without any downloads, installation, or maintenance and with all

the latest features available.

By applying some of the previous techniques the measurements show a spectacular improvement in performance. For example, according to Mathworks [16] preallocation and vectorization can speed up code by several orders of magnitude. In Maple, the new algorithms preallocate memory and run in place, which makes them faster (approximately twice comparing to previous edition) when multiple extensions are present [17]. The use of Mathematica built-in functions can make the code run sometimes4 times faster [18] and the use of efficient built-in data structures (such as packed arrays or sparse arrays that can be used in many more situations than it may appear from their stated main purpose) gives us better execution time. All these are some examples, there are many other experiments that show us in detail how efficient is to follow the tips that are presented above.

4. Results

Computer Algebra Systems (CAS) have become increasingly popular for students and teachers. The reasons are numerous e.g., are more flexible, simple to use, accessible from anywhere etc. However, as with any instructive software, they too have a few impediments that we ought to know and optimize them.

Computer algebra system optimization is the set of best practices and techniques to keep the CAS running optimally. Different devices, platforms, and web browsers may have distinctive necessities, limitations, and features that affect how CAS runs. It's very difficult to optimize CAS performance in the same level for every environment. There are many factors such as the processing power, memory and the operating system that significantly determine the performance. Most of the techniques referred in this paper are general and applies to all CASs, but there are tips which apply only to specific CASs. The basic of them are summarized in the following lines:

- It's a very good practice to run profiler on your code and improve performance by changing lines of code or functions that use a significant amount of time. This advice is maybe the most important of all.
- Listen code analyzer suggestions.
- Make sure that your problem and the goals are clear in order to select the appropriate mathematical tool.
- Automate as much as you can.
- Make the mathematical operations simpler. Place independent operations outside loops and use vector operations instead of loops.
- Use modular programming (separate program functions into independent pieces).
- Organize and use the advanced IDEs (Integrated Development Environments) of CASs.
- Use the appropriate built-in function and method to solve a problem. Avoid overloading built-ins functions. Don't spend time to write functions that al-

ready exists.

- Create new variables if data type changes. Avoid code that generate variables. Avoid global variables. Choose local functions over nested functions. Use functions instead of scripts.
- Use sparse arrays when appropriate and numeric arrays when possible.
- Pre-allocate when appropriate.
- Use online version of CASs if your computer doesn't meet the basic desktop version requirements (in RAM, processing power etc.). All you need is a Web-Browser and Internet access.

Computer Algebra Systems have become increasingly popular for students and teachers. What tool we can use depends on what we want to solve. MATLAB is more focused on numerical computing, while Wolfram Mathematica is more focused on symbolic computation. Both are powerful tools, but they are designed for different types of tasks. MATLAB is more data-oriented than Mathematica. Maple isn't supported as widely as other mathematical software (MATLAB and Mathematica) and new users are not so familiar with it.

5. Conclusion

The world of CAS development is always evolving, with new technologies and tools. New versions of CASs have greatly improved performance and efficiency in many ways which include: Faster computation speeds for many basic operations, technical improvements for drawing graphs, integration of new technologies resulting in faster compile times and many other things. However, there is still room for improvement. Optimizing performance of CASs in any way is a wise investment. On the other hand, online Computer Algebra platforms are on the rise because all you need is a simple PC or mobile phone with Internet access. They provide a variety of tools that help you easily find a solution for any math problem. We hope that in the future all these innovations will continue and will be even more. The only thing that is certain is that CASs is going to play a very important role in every field in future.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Weigand, H.G. (2017) What is or What Might be the Benefit of Using Computer Algebra Systems in the Learning and Teaching of Calculus? In: Faggiano, E., Ferrara, F. and Montone, A., Eds., *Innovation and Technology Enhancing Mathematics Education*, *Mathematics Education in the Digital Era*, Vol. 9, Springer, Cham, 161-193. https://doi.org/10.1007/978-3-319-61488-5_8

[2] https://www.maplesoft.com/

[3] https://www.wolfram.com/mathematica/

[4] https://www.mathworks.com/products/matlab.html

[5] Shacham, M. and Cutlip, M.B. (1998) A Comparison of Six Numerical Software Packages for Educational Use in the Chemical Engineering Curriculum. *Proceedings of the* 1998 *Annual ASEE Conference*, Seattle, 28 June-1 July 1998, 1-12.

[6] http://jonathankinlay.com/2018/10/comparison-programming-languages/

[7] https://www.mathworks.com/company/newsletters/articles/accelerating-matlab-algorithms-and-applications.html

[8] https://www.maplesoft.com/support/help/maple/view.aspx?path=efficiency

[9] Tsagris, M. and Papadakis, M. (2018) Taking R to Its Limits: 70+ Tips. https://doi.org/10.7287/peerj.preprints.26605v1

[10] https://www.mathworks.com/help/matlab/matlab_prog/profiling-for-improving-performance.html

[11] https://www.mathworks.com/help/matlab/ref/shortcircuitand.html

[12] Singh, D. (2017) An Empirical Study of Programming Languages from the Point of View of Scientific Computing. *IJISET—International Journal of Innovative Science, Engineering & Technology*, **4**, 367-371.

[13] Getreuer, P. (2023) Writing Fast MATLAB Code. MATLAB Central File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/5685-writing-fast-matlab-code

[14] https://www.wolfram.com/workbench/

[15] https://www.sogeti.com/explore/reports/world-quality-report-2020/

[16] https://www.mathworks.com/products/matlab/performance.html

[17] https://www.maplesoft.com/support/help/maple/view.aspx?path=OpenMaple%2FC%2FMapleAlloc

[18] https://sudonull.com/post/96601-10-Tips-for-Writing-Quick-Code-in-Mathematica