Original Article

Sage Math: An Open-Source Software Platform for Mathematics Education and Research

Swati Balkrishna Jadhav[1] Vaishali Navnath Gawade[2] Nirmala Hemant Kenjale[3]
[1](M.Sc. B.Ed. Mathematics)
[2](M.A. B. Ed)
[3](M.Sc. B.Ed. Mathematics) Lecturer (KBP Polytechnic Satara)

Manuscript ID: IJERSD-2025-010612

ISSN: 3067-2325

Volume 1

Issue 6

Pp. 52-56

December 2025

Submitted: 10 Nov. 2025

Revised: 25 Nov. 2025

Accepted: 15 Dec 2025

Published: 31 Dec 2025

Correspondence Address:
Swati Balkrishna Jadhav
(M.Sc. B.Ed. Mathematics)
Email: jadhavsb01@gmail.com

Quick Response Code:

Web. https://rlgjaar.com

DOI: 10.5281/zenodo.18276921

DOI Link:
https://doi.org/10.5281/zenodo.18276921

Abstract:

Open-source mathematical software has transformed the landscape of teaching, learning, and research in mathematics by providing powerful, accessible, and cost-effective computational tools. Sage Math, one of the most advanced open-source computer algebra systems (CAS), integrates a wide range of established libraries including NumPy, SciPy, Matplotlib, Sym Py, Maxima, GAP, FLINT, and R into a unified, Python-based environment. This paper presents an overview of Sage Math's design principles, core features, and system architecture, emphasizing its versatility and flexibility through numerous configuration options tailored to diverse academic and research needs. By illustrating typical workflows in numerical methods, linear algebra, calculus, mathematical modelling, number theory, graph theory, and related fields, the paper demonstrates how Sage Math supports symbolic computation, numerical analysis, and visualization within a single platform. Strengths, limitations, and practical considerations for adoption are discussed to guide educators and researchers in effectively integrating Sage Math into their work. The paper also highlights the role of the Sage Math community, its open-source distribution model, and its potential as a robust, affordable alternative to commercial mathematical software. Overall, Sage Math serves as a valuable tool for enhancing conceptual understanding, improving computational efficiency, and promoting accessible mathematical exploration.

Keywords: *Sage Math, Open-source CAS, Numerical computation, Symbolic computation, Computational efficiency, Mathematics education, Mathematical research.*

Introduction:

Sage Math open-source software initially developed by Prof. Willan stein (Formerly at the University of Washington) and released in 2005 in Harvard University. Unlike most of the other CAS, Sage Math does not have its own specialized programing language. The mainstream Programing language of Sage Math is Python. Built upon the Python programming language, Sage Math integrates the functionality of numerous existing open-source packages and libraries, This creates a unified and powerful environment. Its open-source nature promotes transparency, collaboration, and community-driven development, allowing users to inspect, modify, and extend its codebase. Sage Math is available for various operating systems and can also be accessed and used interactively through cloud-based platforms. This allows users to access a vast scope of mathematical functionality covering algebra, calculus, number theory, graph theory, numerical analysis, statistics, and more without needing to learn multiple interfaces or pay licensing fees.

Objectives of the Paper:

1. **To examine the role of SageMath as an open-source alternative to commercial mathematical software** used in teaching, learning, and research.
2. **To analyze the features and capabilities of Sage Math** that support symbolic, numerical, and computational problem-solving in various mathematical domains.
3. **To demonstrate how Sage Math enhances conceptual understanding** through interactive and visual approaches to mathematical concepts.
4. **To evaluate the effectiveness of Sage Math in improving computational efficiency** for tasks commonly performed in mathematics education and research.
5. **To present illustrative examples and use cases of Sage Math** that highlight its practical applications in mathematics calculations and problem-solving.
6. **To discuss the benefits and potential challenges of integrating Sage Math** into mathematics curricula at different levels of education.

General Configuration during Installation (from source):

Configuring Sage Math depends on the specific aspect being addressed, as Sage Math offers various configuration options for different use cases.

When building Sage Math from source, the ./configure script is used to check for system packages and dependencies. This script determines which packages Sage can use from your system instead of building them itself, potentially saving significant build time. The output of ./configure provides information about missing or unsuitable packages and suggests system packages to install

How to cite this article:
Jadhav, S. B., Gawade, V. N., & Kenjale, N. H. (2025). Sage Math: An Open-Source Software Platform for Mathematics Education and Research. International Journal of Engineering Research for Sustainable Development, 1(6), 52–56. https://doi.org/10.5281/zenodo.18276921

Jupyter Kernel Configuration:

Sage Math is commonly installed with its own Jupyter kernel, allowing users to run Sage Math code directly within Jupyter Notebook. The availability of this kernel can be checked using jupyter kernelspec list. If Sage Math was installed in editable mode (for example, using `pip install -e`), the kernel may not appear automatically; reinstalling Sage Math in standard, non-editable mode usually resolves this. Additionally, an existing Jupyter environment can be manually linked to a Sage Math installation to register the Sage kernel as one of the selectable options.

Developer-Oriented Setup:

For contributors working on Sage Math's development, configuration involves preparing a suitable development workflow. This includes cloning the Sage Math repository, creating a personal fork, and configuring Git remotes such as origin and upstream to manage contributions effectively.

Configuration Using `sage_conf`:

The `sage_conf` package provides Sage Math with essential configuration details during installation and execution. It can be installed via `pip install -v sage_conf`. This package is responsible for identifying the appropriate directory structure for both the build and installation phases of the Sage Math system.

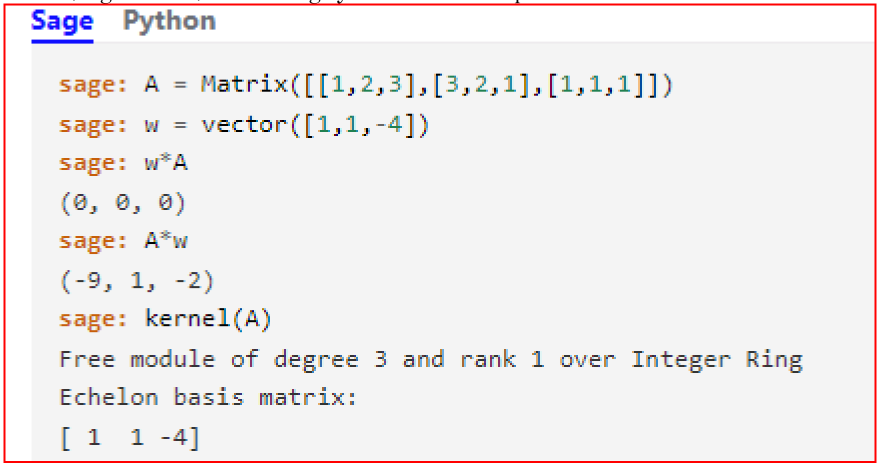
SageTeX Integration:

SageTeX, which enables SageMath computations within LaTeX documents, typically integrates automatically once both SageMath and TeX Live are installed. After installation, SageTeX commands become readily available for use in LaTeX environments, allowing dynamic inclusion of mathematical results.

Applications of SageMath (with Examples):

Linear Algebra:

Research highlights SageMath's capability in handling a wide variety of linear algebra tasks such as matrix construction, matrix computations, determinants, eigenvalues, and solving systems of linear equations.

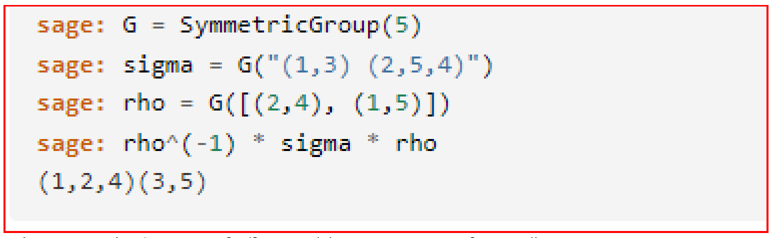


```
sage: A = Matrix([[1,2,3],[3,2,1],[1,1,1]])
sage: w = vector([1,1,-4])
sage: w*A
(0, 0, 0)
sage: A*w
(-9, 1, -2)
sage: kernel(A)
Free module of degree 3 and rank 1 over Integer Ring
Echelon basis matrix:
[ 1  1 -4]
```

Fig.1: Example of Matrix in SageMath (https://www.sagemath.org/)

Group Theory:

A significant portion of SageMath's functionality in group theory comes from its interface with GAP. This allows groups to be defined and explored in several formats, enabling users to perform operations, test properties, and work with algebraic structures.

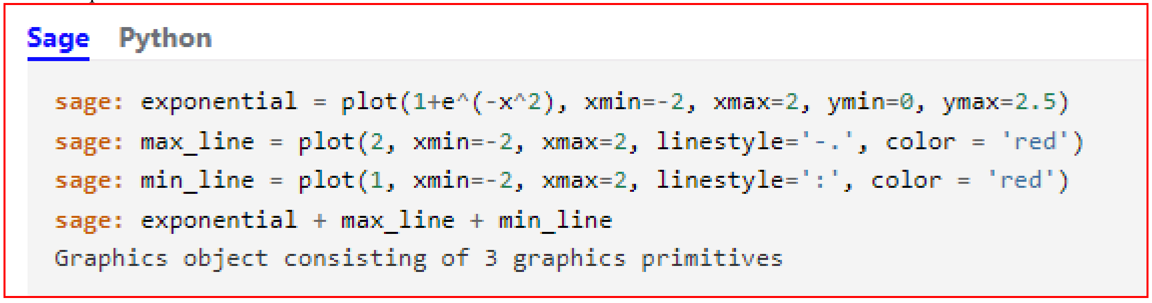


```
sage: G = SymmetricGroup(5)
sage: sigma = G("(1,3) (2,5,4)")
sage: rho = G([(2,4), (1,5)])
sage: rho^(-1) * sigma * rho
(1,2,4)(3,5)
```

Fig. 2: Example of Symmetric group in SageMath (https://www.sagemath.org/)

2D and 3D Plotting:

SageMath can create detailed two-dimensional and three-dimensional visualizations. In both Jupyter Notebook and command-line interfaces, plots are displayed interactively using the open-source ThreeJS renderer, enabling zooming, rotation, and exploration of graphical output.



```
sage: exponential = plot(1+e^(-x^2), xmin=-2, xmax=2, ymin=0, ymax=2.5)
sage: max_line = plot(2, xmin=-2, xmax=2, linestyle='-.', color = 'red')
sage: min_line = plot(1, xmin=-2, xmax=2, linestyle=':', color = 'red')
sage: exponential + max_line + min_line
Graphics object consisting of 3 graphics primitives
```

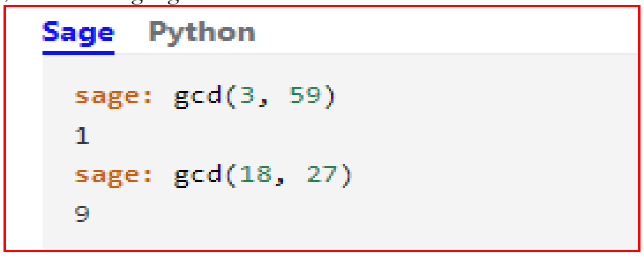
Fig..3: Example of 2D plotting in SageMath (https://www.sagemath.org/)

Graph Theory:

SageMath includes comprehensive tools for building and analyzing graphs. Users can construct graphs, search for paths and cycles, compute invariants, and explore network structures.

Number Theory:

With strong support for number-theoretic operations, SageMath is widely used for tasks such as primality testing, integer factorization, modular arithmetic, and handling algebraic number fields.

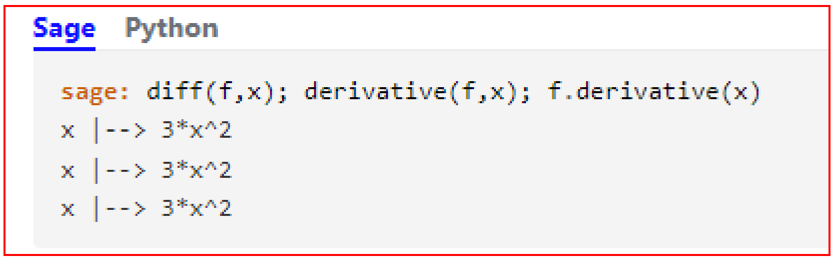


```
sage: gcd(3, 59)
1
sage: gcd(18, 27)
9
```

Fig.4: Example of GCD of two numbers in SageMath (https://www.sagemath.org/)

Calculus:

SageMath supports a broad spectrum of calculus operations, including solving equations, differentiation, integration, Taylor expansions, and Laplace transforms.



```
sage: diff(f,x); derivative(f,x); f.derivative(x)
x |--> 3*x^2
x |--> 3*x^2
x |--> 3*x^2
```

Fig.5: Example of Derivative in SageMath (https://www.sagemath.org/)

Numerical Methods:

Studies demonstrate how SageMath can be used to teach and implement common numerical techniques such as Runge Kutta methods, Gaussian elimination, Newton's method, and Gauss–Seidel iteration offering an open-source alternative to costly proprietary software.

Mathematical Modeling:

Researchers employ SageMath to solve mathematical models involving differential equations, optimization problems, double integrals, and various continuous or discrete systems.

Signal Processing:

SageMath can perform signal-processing tasks like Fourier transforms, FFT, and digital filtering. These computational tools help students better understand theoretical concepts through simulations.

Educational Use:

Several studies highlight SageMath as an effective tool in mathematics education, identifying common student errors and suggesting ways to enhance learning. It is widely applicable in undergraduate mathematics courses, engineering computation labs, online learning environments, and even high school level algebra and calculus instruction.

Research Applications:

Researchers in mathematics and related fields use SageMath for: Algorithm implementation in linear algebra Computational experiments High-dimensional data analysis Prototype development

Example workflows (brief):

Below are short illustrative workflows showing how Sage can be used in practice.

Quick symbolic/numeric example (conceptual):

Define a polynomial, factor it symbolically, compute roots numerically, and plot the polynomial with labeled roots all within the same Sage session. This seamless transition from symbolic to numeric to visualization is core strength.

Reproducible documents: Using Jupyter notebooks or SageTeX, one can embed Sage computations directly in lecture notes or papers so that figures and numeric results are generated from the same code that appears in the document improving reproducibility.

Web-based Sharing:

SageMathCell and hosted services (e.g., CoCalc) let users share short notebooks or interactive exercises without requiring recipients to install Sage locally

Strengths, Limitations, and Practical Advice:

Strengths:

SageMath, as an open-source mathematical software system, offers several key strengths:

Comprehensive Mathematical Capabilities:

SageMath integrates a vast array of functionalities spanning almost all areas of mathematics, including symbolic calculus, numerical analysis, linear algebra, abstract algebra, combinatorics, graph theory, cryptography, and more. It serves as a unified interface for numerous specialized mathematical software packages, leveraging their individual strengths.

Open-Source and Free:

Being open-source, SageMath is free to download, install, and use for anyone, whether for personal, educational, or commercial purposes. This eliminates licensing costs and promotes accessibility for students, researchers, and professionals globally.

Python-Based:

SageMath uses Python as its primary programming and interface language. This offers significant advantages:

Ease of Learning: Python is a widely-used and well-designed language, making SageMath relatively accessible for users already familiar with Python or those looking to learn a versatile programming language.

Extensibility: Users can leverage the extensive Python ecosystem and integrate numerous Python libraries into their SageMath workflows.

Scripting and Automation: Python's capabilities enable users to write complex scripts, automate tasks, and develop custom functionalities within SageMath.

Interactive and User-Friendly Interfaces:

SageMath provides multiple ways for users to interact with the system:

Web-based Notebook Interface (Jupyter): This offers a dynamic and interactive environment for exploring mathematical concepts, performing calculations, visualizing results, and sharing work.

Command-line Interface: For users who prefer a traditional terminal-based interaction.

Python Library: Allowing integration into other Python projects.

Transparency and Verifiability:

The open-source nature of SageMath means its source code is publicly available for examination. This promotes transparency, allows users to understand how calculations are performed, and facilitates bug identification and correction.

Community and Development:

SageMath fosters an active community of developers and users, encouraging collaboration, contributions to the codebase, and readily available support through forums and documentation.

Leveraging Existing Software Instead of reinventing functionalities, SageMath integrates and interfaces with established open-source and even proprietary mathematical software packages (when available), combining their power under a single umbrella. This approach ensures access to highly optimized and rigorously tested algorithms.

Limitations:

SageMath, while a powerful and comprehensive open-source mathematics software system, does have certain limitations:

Windows Support Challenges:

Natively compiling and running SageMath on Windows can be complex due to challenges with low-level C and FORTRAN libraries. While solutions like Windows Subsystem for Linux (WSL) or virtualization (e.g., VirtualBox) are available, they introduce an additional layer of complexity for Windows users.

No Stabilization Releases:

The SageMath development cycle, with its emphasis on frequent beta and release candidates, does not typically include dedicated stabilization releases. While critical bug fixes are prioritized, users should be aware that development releases, though generally reliable, may not offer the same level of long-term stability as a dedicated stable release cycle.

Learning Curve for Advanced Features:

While SageMath's Python-based interface is generally accessible, mastering its full range of advanced features, particularly for specialized mathematical domains, can require a significant learning investment.

Dependency on External Packages:

SageMath integrates numerous other open-source software packages. While this provides extensive functionality, it also introduces a dependency on the maintenance and compatibility of these external components.

No Claims of Bug-Free Software:

Like any complex software project, SageMath developers do not claim it to be entirely free of bugs. While rigorous testing is in place, users should be aware that encountering bugs is possible, and the open-source nature means community involvement in reporting and resolving them is important.

Advice:

Use cloud platforms (CoCalc,SageMathCell) for teaching to avoid installation issues.For research, use containers or version control to ensure long-term reproducibility.

Conclusion:

By automating computational steps, Sage enables deeper understanding of fundamental concepts and encourages experimental learning. As an open-source platform, SageMath provides a cost-effective alternative to proprietary software while maintaining high computational reliability and transparency.

The applications discussed in this paper demonstrate how SageMath can enhance mathematical education, support complex problem solving, and enable computational research. Given its growing popularity and community support, SageMath is likely to become a standard tool in modern mathematical practice.

SageMath provides a powerful, community-driven environment that lowers barriers to computational mathematics by combining many open libraries under a common, Pythonic interface. Its strengths in reproducibility, cost-free licensing, and extensibility make it well suited for teaching and research. Continued development and community engagement will be key to improving usability (installation, documentation) and expanding domain coverage. For educators and researchers wishing to adopt Sage,the official documentation, tutorials, and the SageMathCell quick-start are recommended starting points.

Acknowledgment

The authors would like to express their sincere gratitude to KBP Polytechnic, Satara, for providing a supportive academic environment and necessary facilities that enabled the preparation of this research work. The authors are thankful to the management and colleagues for their encouragement, cooperation, and constructive discussions during the development of this paper.

The authors gratefully acknowledge the global SageMath developer community for creating and continuously maintaining a powerful open-source mathematical platform that significantly contributes to mathematics education and research. Appreciation is also extended to the contributors of integrated open-source libraries such as NumPy, SciPy, SymPy, GAP, and related tools, which form the backbone of SageMath's computational ecosystem.

Finally, the authors sincerely thank family members and well-wishers for their constant motivation, patience, and moral support throughout the course of this academic work.

Financial support and sponsorship Nil.

Conflicts of interest

The authors declare that there are no conflicts of interest regarding the of this paper

References:

1. "Leveraging SageMath and ChatGPT for (orthogonal) diagonalization and Singular Value Decomposition..." by Chien-Pang Kuan, Ming-Hao Chiang, and Yung-Sheng Huang: Presented by AIMS Press, this paper explores SageMath's role in

assisting undergraduate students with computational tasks in linear algebra, confirming calculations, and handling complex problems.

2. "Sage math for education and research" by P. Szabó: A paper available on ResearchGate, which discusses SageMath's applications in education and research, including an example of using it for the Goldbach conjecture.
3. "Computational Mathematics with SageMath" by Paul Zimmermann et al.: A book published by the Society for Industrial and Applied Mathematics (SIAM), it provides a comprehensive overview of the SageMath system, its components, and capabilities.
4. SageMath Official Website. https://www.sagemath.org/
5. NumPy and SciPy Documentation.
6. Sage Tutorial (PDF). SageMath documentation. doc.sagemath.org