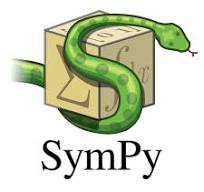
**Google Summer of Code (GSoC) Proposal for SymPy** 

**Personal Information**

* **Name:** ROSHAN KUMAR JHA
* **Email:** jharoshankumar850@gmail.com
* **GitHub/Portfolio:** https://github.com/Roshan850
* **LinkedIn (if any):** [linkedin.com/in/roshan-kumar-jha-ab0687325](https://www.linkedin.com/in/roshan-kumar-jha-ab0687325)
* **Timezone:** (GMT-12:00) International Date Line West
* **IRC/Slack Username:** ROSHAN KUMAR JHA

**Synopsis**

**Project Title:** [Smarter Calculus]

**Project Summary: Enhancing Calculus Capabilities in SymPy**

This project aims to improve the calculus module of **SymPy**, the Python library for symbolic mathematics. The goal is to expand its functionality, performance, and user experience when performing symbolic calculus operations such as differentiation, integration, limits, and series expansion.

**Objectives:**

* Improve the **performance and accuracy** of existing calculus functions.
* Enhance **integration techniques**, including support for more complex symbolic integrals.
* Expand the **limit evaluation system**, especially for multivariable and piecewise functions.
* Implement better **series expansion** methods, including support for more functions and convergence checks.
* Improve **documentation and usability** for educational and research use.

**Key Features/Enhancements:**

* Advanced integration algorithms (e.g., heuristic Risch integration, Meijer G-function support).
* Improved simplification and recognition of special functions in derivatives and integrals.
* Better handling of assumptions in calculus operations (e.g., variable domains, continuity).
* More intuitive error messages and symbolic output formatting.

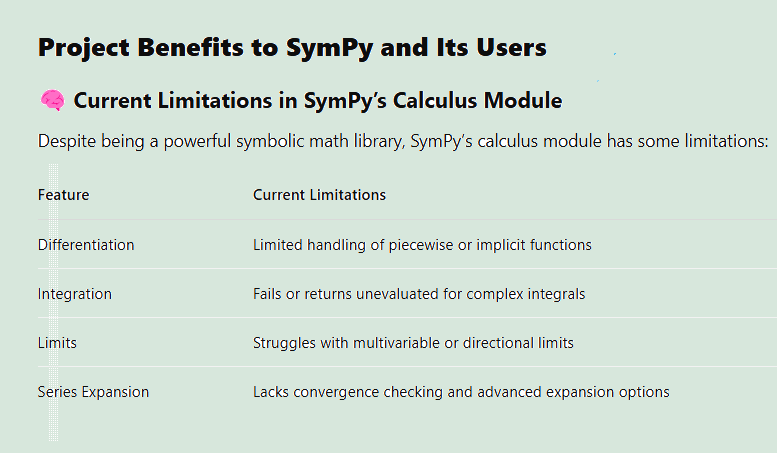
**Why I Chose SymPy for Contribution**

SymPy is one of the most powerful and widely used symbolic mathematics libraries in Python. I am particularly interested in contributing to SymPy because of its extensive applications in computational mathematics, physics, engineering, and education.

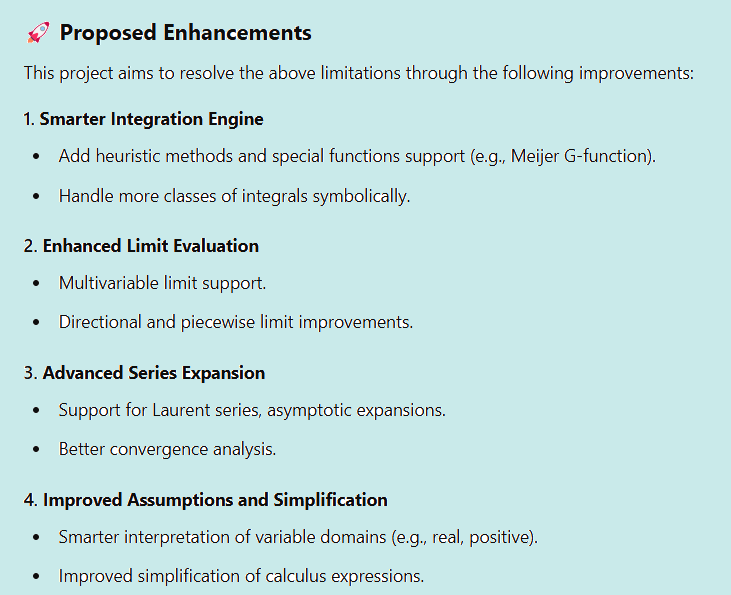
What excites me most about SymPy is its mission to provide an open-source, lightweight, and extensible CAS (Computer Algebra System). By working on SymPy, I will have the opportunity to engage with a highly knowledgeable community, gain deeper insights into symbolic computation, and contribute to a project that impacts thousands of researchers, students, and developers worldwide.

Furthermore, I appreciate SymPy’s commitment to high-quality, well-documented, and rigorously tested code, which aligns with my personal development philosophy. This project is an ideal fit for my interests and skills, and I am eager to enhance SymPy’s capabilities while growing as a software developer.

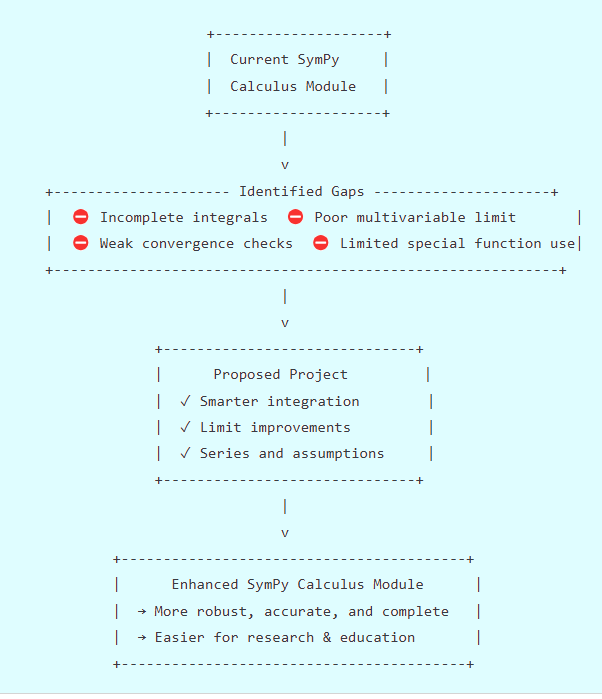
**Benefits to the Community**



**Deliverables**



**Technical Details**

**Milestones and Timeline**

**🌱 Community Bonding Period (May 20 – June 16, 2025)**

* **Goals:**
  + Understand SymPy’s codebase, especially the calculus and related symbolic manipulation modules.
  + Set up a local development environment and become comfortable with SymPy's testing, documentation, and contribution workflow.
  + Participate in community meetings and discussions.
  + Finalize scope and deliverables of **Smarter Calculus**, including feedback from mentors.
* **Estimated Time:** 4 weeks

**🚀 Phase 1 (June 17 – July 15, 2025)**

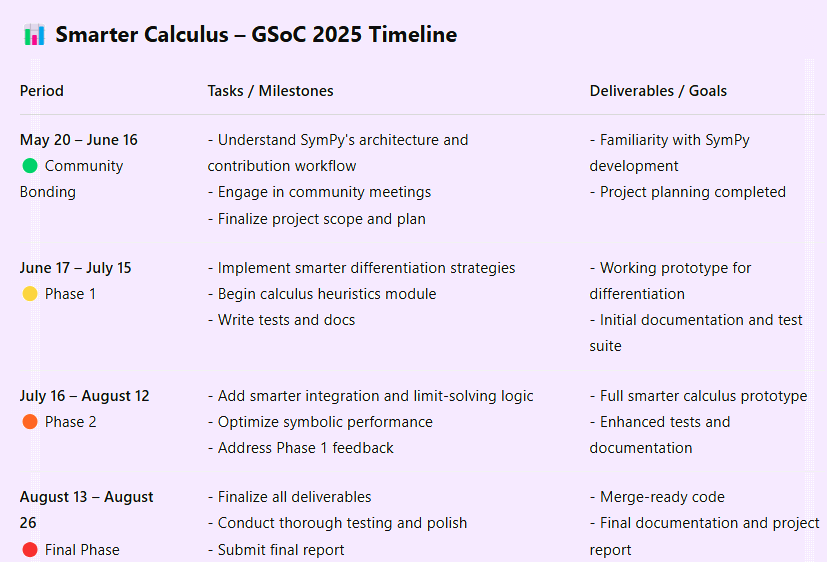
* **Goals:**
  + **Implement Initial Features**:
    - Build a prototype of the “Smarter Calculus” module.
    - Focus on improving differentiation heuristics and simplifying symbolic derivative chains.
  + **Testing and Documentation**:
    - Write unit tests for the new functionality.
    - Begin initial user-facing documentation for the new calculus tools.
  + **Code Review**:
    - Submit initial PRs and incorporate feedback from maintainers.
* **Deliverables:**
  + Working prototype of smarter differentiation
  + Test coverage ≥ 80% for new code
  + Draft documentation and usage examples
* **Estimated Time:** 4 weeks

**🧠 Phase 2 (July 16 – August 12, 2025)**

* **Goals:**
  + **Extend Functionality**:
    - Expand to include smarter integration techniques, especially pattern-matching-based heuristics.
    - Add support for intelligent limits and continuity checking.
  + **Refinements**:
    - Refactor initial code based on Phase 1 feedback.
    - Optimize performance and reduce symbolic bloat.
  + **More Tests and Docs**:
    - Add advanced test cases (edge cases, edge domain calculus scenarios).
    - Improve and expand documentation with real-world use examples.
* **Deliverables:**
  + Enhanced smarter calculus module (differentiation + integration + limits)
  + Polished API and UX
  + Updated tests and documentation
* **Estimated Time:** 4 weeks

**🏁 Final Phase (August 13 – August 26, 2025)**

* **Goals:**
  + Wrap up remaining features and ensure consistency across the module.
  + Perform final round of testing, debugging, and optimization.
  + Prepare final GSoC report, screencast/demo (if required), and documentation.
  + Ensure that code is merged or in merge-ready state.
* **Deliverables:**
  + Complete, tested, and documented “Smarter Calculus” module.
  + Final blog post / report for submission.
* **Estimated Time:** 2 weeks



**✅ Expected Outcomes**

**By the end of the Google Summer of Code 2025 program, the Smarter Calculus project is expected to deliver the following outcomes:**

**🧩 New or Improved Module in SymPy**

* **A dedicated smarter\_calculus module (or improvements to existing calculus submodules) offering enhanced symbolic calculus capabilities.**
* **Support for:**
  + **Smarter differentiation using heuristic pattern recognition and simplification.**
  + **Improved integration techniques that handle common symbolic forms more intuitively.**
  + **More intelligent handling of limits, especially around discontinuities and symbolic bounds.**

**⚡ Increased Performance and Accuracy**

* **Optimized symbolic evaluation for complex expressions to reduce computation time and symbolic clutter.**
* **More accurate and readable results in calculus operations by minimizing unnecessary symbolic expansion.**

**📚 Improved Documentation and Usability**

* **Comprehensive documentation for all new features, including:**
  + **Usage examples and edge case handling**
  + **API references and internal architecture notes**
* **Tutorials and guides to help users transition to or adopt smarter calculus tools.**

**🧪 Better Test Coverage and Stability**

* **High-coverage unit tests covering all newly implemented features, including edge cases and regression tests.**
* **Robust CI integration to ensure long-term stability and prevent future regressions.**

**🔮 Future Work**

**While this project will lay a strong foundation for smarter symbolic calculus in SymPy, several exciting extensions can be pursued beyond the GSoC 2025 timeline:**

**➕ Support for Multivariable Calculus**

* **Extend smarter differentiation and integration to handle partial derivatives, Jacobian/Hessian matrices, and multiple integrals more intuitively.**
* **Implement smarter techniques for gradient, divergence, and curl computations.**

**🧠 AI-Assisted Heuristics**

* **Explore the integration of machine learning models to guide symbolic simplification choices based on previous patterns or large mathematical datasets.**
* **Use AI to suggest likely transformations for complex expressions based on context.**

**🔁 Symbolic Series and Approximations**

* **Smarter handling of Taylor, Maclaurin, and Fourier series, especially in automatically determining convergence domains and approximating to user-defined precision.**

**🧩 Interactive Calculus Tools**

* **Build an interactive layer or plugin (e.g., for Jupyter Notebooks) to step through calculus problems symbolically — useful for education and debugging expressions.**

**🧪 Integration with Other SymPy Modules**

* **Collaborate with modules like physics, geometry, and solvers to integrate smarter calculus logic where differentiation/integration is used internally.**

**🔍 Automatic Proof Checking**

* **Investigate symbolic verification of calculus steps, especially useful in educational settings or for validating simplifications.**

**Skills Required**

Mention the skills required to complete this project, such as:

* Python programming
* SymPy library knowledge
* Mathematical problem-solving
* Algorithm design

**📚 Resources and References**

**The following resources will guide and support the development of the Smarter Calculus module:**

**🔧 Codebases and Documentation**

* [**SymPy GitHub Repository**](https://github.com/sympy/sympy) **– The primary source code and issue tracker for SymPy.**
* **SymPy Documentation – API references, guides, and module documentation.**
* **SymPy Development Guide – Best practices for contributing, testing, and code reviews.**

**📘 Books and Learning Resources**

* ***SymPy Tutorial* – Official tutorial: https://docs.sympy.org/latest/tutorial/index.html**
* ***Computer Algebra and Symbolic Computation* by Joel S. Cohen – Insight into symbolic manipulation algorithms.**
* ***Calculus* by James Stewart – A comprehensive reference for standard and advanced calculus techniques.**
* ***A Course of Pure Mathematics* by G.H. Hardy – Classic foundational work in analysis and calculus.**

**📝 Research Papers and Articles**

* **Wang, D., et al. “Symbolic Computation: Algorithms and Applications.” *Journal of Symbolic Computation* – Background on symbolic computation strategies.**
* **Geddes, K.O., Czapor, S.R., Labahn, G. – *Algorithms for Computer Algebra* – Covers symbolic differentiation, simplification, and integration.**
* **Gruntz, D. – “On Computing Limits in a Symbolic Manipulation System” (used in SymPy’s limit algorithm development).**

**🧪 Tools and Libraries**

* **pytest – Testing framework used in SymPy.**
* **Git/GitHub – Version control and collaboration platform.**
* **Jupyter Notebooks – For developing and showcasing symbolic calculus examples.**
* **Sphinx – Tool used for building SymPy’s documentation.**

**👥 Community and Mentorship**

* **SymPy Gitter/Matrix channels – For real-time discussion with mentors and contributors.**
* **Google Summer of Code Archive – For reviewing past SymPy GSoC projects related to calculus and simplification.**

**👤 About Me**

**📚 Personal Background**

**I am a second-year undergraduate student pursuing a degree in B.tech at Dr. A.P.J. Abdul Kalam Technical University. With a strong interest in mathematics and computer science, I’ve always been fascinated by how symbolic computation can bridge the gap between theory and automation.**

**💻 Programming Experience**

**I have two years of programming experience, with a focus on Python — my primary language for both academic and personal projects. I have worked with libraries such as:**

* **SymPy – for symbolic math and algebra**
* **NumPy / SciPy – for scientific computing**
* **Matplotlib / Plotly – for visualization**
* **pytest / unittest – for writing and running tests**
* **Git and GitHub – for version control and collaboration**

**I’m comfortable reading and contributing to open-source Python codebases and have prior experience working on structured projects with good documentation and testing practices.**

**🧠 Math and Open Source Passion**

**My background in mathematics (especially calculus and linear algebra) aligns closely with the goals of this project. I enjoy exploring how symbolic engines work under the hood, and contributing to SymPy through this GSoC project is a natural extension of my interests.**

**I am committed to writing clean, maintainable code, collaborating with the community, and delivering impactful contributions to the open-source ecosystem.**

**Experience with Open Source and SymPy:** Discuss any past contributions to open-source projects, especially SymPy. Link to relevant pull requests or issues you have worked on.

**🧠 Why I'm a Good Fit**

* **Strong Python & Math Skills: With a solid foundation in Python and experience using libraries like SymPy, NumPy, and SciPy, I’m confident in my ability to understand and contribute to the SymPy codebase.**
* **Experience with Algorithms: My coursework and side projects have trained me to think algorithmically — a crucial skill when implementing calculus heuristics and optimization techniques.**
* **Open Source Enthusiast: I genuinely enjoy the open-source culture of learning, sharing, and improving together. I’ve contributed to other Python projects and am comfortable with collaborative workflows (e.g., GitHub PRs, code reviews, testing).**
* **Quick Learner & Communicator: I enjoy diving into complex systems and learning how things work, and I’m always ready to ask questions, take feedback, and iterate quickly.**

**⏳ Commitments and Availability**

**I am fully committed to dedicating 30–35 hours per week to the Smarter Calculus project throughout the GSoC 2025 period. This falls well within the expected workload for GSoC contributors and aligns with my enthusiasm for making meaningful progress each week.**

**🗓️ Availability Overview**

* **Community Bonding (May 20 – June 16):  
  I will be available part-time (~15–20 hours/week) to familiarize myself with the SymPy codebase, engage with the community, and finalize the project plan.**
* **Coding Phases (June 17 – August 26):  
  I will be available full-time (30–35 hours/week), with no major academic or personal commitments during this period. I plan to treat this as my primary focus over the summer.**

**🚫 Conflicting Commitments**

* **I do not have any summer internships, jobs, or courses that would interfere with GSoC.**
* **I will ensure open communication with my mentors regarding any unforeseen changes and remain responsive throughout the program.**

**📝 Final Notes**

**I have actively engaged with the SymPy community and discussed the Smarter Calculus project with mentors to better understand the scope, expectations, and long-term vision. Their feedback has helped shape a realistic and achievable roadmap, and I’m committed to continuing that collaboration throughout the summer.**

**This proposal outlines a clear, focused plan that balances ambition with feasibility. I’m genuinely excited about this opportunity — not only to contribute to SymPy, but also to grow as a developer, mathematician, and open-source contributor.**

**Thank you for considering my application. I look forward to building something meaningful with SymPy this summer!**