William Stein ***RRF: Proposed Research***

(206) 419-0925 wstein@gmail.com http://wstein.org

1 Introduction and Rationale

Most of the PI's (principal investigator's) research has been in number theory, which is an area of pure mathematics. The PI embarked on a new and risky research path at Harvard in 2005, when he started the Sage mathematical software project, which has greatly grown since he was recruited by UW in 2006. There are now over 150 Sage developers, around 2,000 downloads of Sage per month, and an average of over 1,500 unique visitors to the `http://sagemath.org` website each day. This group of users and developers is guided by a single clear vision for Sage:

> *Create a viable free open source alternative to the commercial systems Maple, Mathematica, Matlab and Magma.*

Sage can be used to study general and advanced, pure and applied mathematics. This includes a huge range of mathematics, including algebra, calculus, elementary to very advanced number theory, cryptography, numerical computation, commutative algebra, group theory, combinatorics, graph theory, exact linear algebra and much more. Sage combines various software packages and seamlessly integrates their functionality into a common experience. It is well suited for education, studying and research. The interface to Sage is a notebook in a web-browser or the command-line. Inside the Sage notebook (see Figure 1), you can create embedded graphics, beautifully typeset mathematical expressions, add and delete input, and share your work across the network.

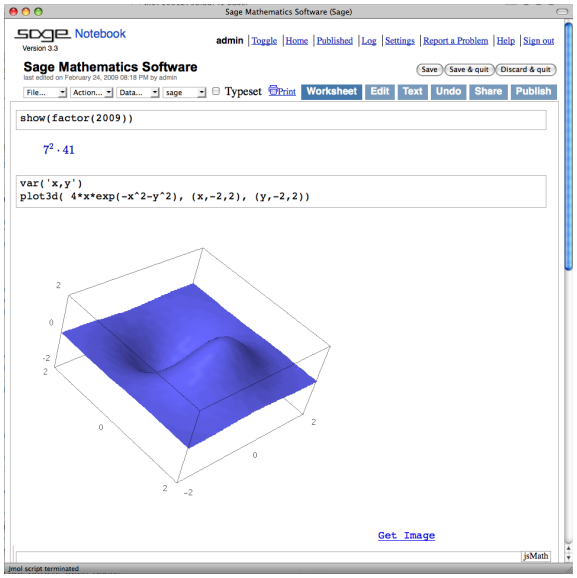

Figure 1: A Screenshot Showing Sage

(206) 419-0925 wstein@gmail.com http://wstein.org

Instead of students and researchers having to pay to buy expensive commercial mathematical software, they now have the option to use Sage for free. At many institutions, purchasing computer software—especially mathematical software—is a significant burden, and Sage has helped address this problem. Moreover, because Sage is free, it is available to many more undergraduates, high school students, and non-mathematicians.

A central technical advantage of Sage over existing mathematical software is that Sage uses the mainstream Python programming language. Because Sage is built on Python, modern exception handling and name spaces are inherent. Sage can be easily extended with new user-defined data types, and since Python is such a widely used programming language, there already exists a huge variety of extension modules. In addition, Sage uses the Cython compiler for writing extremely fast code. In contrast, any other major mathematics software invented its own language, often a slow interpreted language, so that any extension requires a new implementation. And one must not forget that computer based research also involves non-mathematical programming, e.g., create a web server to distribute the results of computations, or connect to an online database or web page and parse the results. By using Python, all this is easily done in Sage.

2 Objectives

The specific goals of this RRF proposal are:

1. **Symbolic Calculus:** Improve Sage's symbolic calculus functionality to make Sage a much better tool for undergraduate education and applied mathematics.

2. **The Sage Notebook:** Make the Sage notebook server more scalable and robust for collaboration and educational applications.

2.1 Symbolic Calculus

Sage has sophisticated symbolic manipulation capabilities, which make it very useful for teaching calculus and doing the kind of symbolic manipulation that forms the cornerstone of classical mathematics. Much of this functionality was initially implemented in Sage by the UW *undergraduate* Bobby Moretti, who devoted nearly a year of hard work to this problem in 2006. Moretti's implementation is a "reference implementation" in that it establishes how the Sage functionality for symbolic manipulation should work, and uses an old program called Maxima behind the scenes to make it actually work. But this functionality is slow, in some cases, very, very slow, especially compared to Maple and Mathematica.

In August 2008, the PI spent nearly every waking moment during two intense weeks modifying the Ginac C++ library so that it could do arithmetic with Python

(206) 419-0925 wstein@gmail.com http://wstein.org

Figure 2: Interactive Image Compression in the Sage Notebook

objects and integrate well with the Sage library. The resulting Python library is called Pynac and is extremely fast and robust. Burcin Erocal, a graduate student at the Research Institute for Symbolic Computation in Linz, Austria, has subsequently worked part time on filling in additional functionality needed to make this library the default library for symbolic manipulation in Sage.

The PI proposes to spend a full six weeks working very hard to push through full Sage/Pynac integration, thus completely replacing the current reference implementation of symbolic manipulation in Sage. This will dramatically improve the speed of Sage for symbolic calculus.

Specific goal: *Finish implementing functionality so that symbolic computation in Sage uses Pynac instead of the current slow Maxima-based system. This includes extensive support for tight integration between Pynac and the rest of Sage and a native implementation of the solve command in Pynac.*

2.2 The Sage Notebook

The Sage notebook is an AJAX application, like Gmail or Google Maps. It provides an interactive web-based worksheet in which one can enter arbitrary Sage commands, see beautifully typeset output, create 2-D and 3-D graphics, publish

worksheets, and collaborate with other users.

The PI and several UW undergraduates together developed the basic implementation of the current version of the Sage notebook during an extremely intense three-week coding session in Summer 2007. This coding work was motivated by UW's SIMUW program, which is an intense summer mathematics program for about 25 high school students. In 2007, the PI taught a two-week SIMUW course using the Sage notebook on the Riemann Hypothesis. In 2008 he taught another 2-week SIMUW course on Quantitative Finance using the notebook's new interactive controls feature (see Figure 2).

The Sage notebook is a much beloved "killer application" of Sage:

> In my opinion, SAGE's notebook is the real killer feature, which I don't recall to have seen in any other (commercial or not) software. I mean, this is the only scientific program that I've found, allowing such an easy collaborative job within local networks.
>
> – Maurizio, the Sage mailing list.

Professors at dozens of universities around the world are getting excited about how they can leverage the Sage notebook in their teaching.

> With some colleagues in our University (Lyon, France) we have built a project around Sage for undergraduate students... *And the University has decided to support this project.* Good news.
>
> Now we will be facing the problem to build a Sage configuration which will work for say 200 students at the same time (students will use the notebook), and prepare professors for Sage teaching. There are 'some' technical problems to solve...
>
> – T. Dumont, the Sage mailing list.

The Sage notebook presently does not robustly scale to more than about 25 users at the same time, no matter how good the hosting hardware is. The PI proposes to spend six weeks (supported by RRF) focused entirely on vastly improving the robustness and scalability of the notebook. The PI is well positioned to succeed at this project, since he is intimately familiar with all aspects of the notebook codebase.

Specific goal: *Improve the notebook so that it will robustly handle up to 200 simultaneous users when running on a single high-end server, as demonstrated by a robust automated test suite. Implement management tools so administrators can manage the notebook load and better balance resources.*

(206) 419-0925 wstein@gmail.com http://wstein.org

3 Procedure

The PI has secured funding from the NSF, PIMS, and the Center for Communications Research to run several Sage Days workshops each year. Much of the planning and collaboration on this project will occur during those *extremely intense* workshops. Also, the PI will collaborate with Burcin Erocal about symbolic computation, and with numerous graduate and undergraduate students at UW who will participate in this project.

All Sage developers receive accounts on the high-end NSF-funded `sage.math` compute cluster in the UW mathematics department, which greatly facilitates collaboration.

All code that comes out of this project will be made freely available as part of Sage and will be peer reviewed as part of the standard peer review process that *all* new code that gets included with Sage goes through.

4 Time Schedule

The PI intends to work full time for three months on this project.

1. Weeks 1–6: Symbolic Calculus
2. Weeks 7–12: The Sage Notebook

5 Need for RRF Support

Though the PI has obtained funding for the Sage project from Microsoft, Google, the National Science Foundation, prize money, private donations, and contracts with industry for Sage development, this has all been limited seed money. The more Sage improves and grows in quality to equal and exceed the commercial offerings, the *easier* it becomes to obtain further outside funding and for Sage to be useful in potentially thousands of educational and commercial applications.

The goals of the current proposal are to greatly enhance Sage's level of *technical credibility* in two areas that will have a huge direct impact: Symbolic Calculus and the web-based Sage Notebook. This project thus directly addresses the mission of the Royalty Research Fund by providing a unique opportunity to increase the PI's competitiveness for subsequent funding.

These two proposed goals particularly make Sage attractive to the *educational market*, a huge area of potential funding that Sage has not had success in yet, probably because these two areas were not developed enough to be easy and powerful enough for many educators. Improving the notebook and symbolic calculus are unique projects that make Sage much, much more attractive for educational funding.