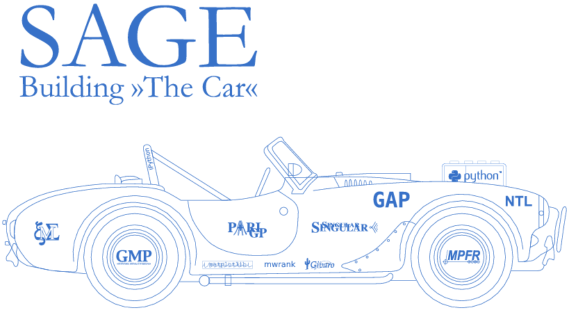
SAGE: Software for Algebra and Geometry Experimentation

William Stein

January 15, 2007, UW CSE Colloquium

http://modular.math.washington.edu/sage



»Every free computer algebra system I've tried has

The Genesis of SAGE

- **1997–1999:** (Berkeley grad student) HECKE – my free C++ program for my thesis research.
- **2000–2005:** (Harvard) I wrote $> 25,000$ lines of Magma code. I had really wanted a sufficiently powerful open source math software system with a modern language... but none existed.
- **Feb 2005:** Started SAGE (job offers with **tenure**).
- **Feb 2006: UCSD SAGE Days 1** – SAGE 1.0.
- **June 2006: UW High school** workshop – Notebook.
- **August 2006: MSRI Grad student** workshop.
- **October 2006: UW SAGE Days 2** workshop.
- **Now: SAGE-1.7**; and **SAGE 2.0** by end of month!

Welcome to SAGE!

```
rank4:~/talks/2007-01-15-sage-cse was$ sage
--------------------------------------------------------------
| SAGE Version 1.7, Release Date: 2007-01-16                 |
| Type notebook() for the GUI, and license() for information.|
--------------------------------------------------------------

sage: 2 + 2
4
sage: notebook()
**************************************************
* Open your web browser to http://localhost:8000 *
**************************************************

....
```

Is Open Source Math Software Important for Mathematical Research?

“You can read Sylow’s Theorem and its proof in Huppert’s book in the library [...] then you can use Sylow’s Theorem for the rest of your life free of charge, but for many computer algebra systems license fees have to be paid regularly [...]. You press buttons and you get answers in the same way as you get the bright pictures from your television set but you cannot control how they were made in either case.

With this situation **two of the most basic rules of conduct in mathematics are violated**: In mathematics **information is passed on free of charge** and **everything is laid open for checking**. Not applying these rules to computer algebra systems that are made for mathematical research [...] means **moving in a most undesirable direction**. Most important: Can we expect somebody to believe a result of a program that he is not allowed to see? Moreover: Do we really want to charge colleagues in Moldava several years of their salary for a computer algebra system?”

– J. Neubüser (1993) (he started GAP in 1986).

Who is Writing SAGE?

Contributors Include: Martin Albrecht, Nick Alexander, Tom Boothby, Robert Bradshaw, Iftikhar Burhanuddin, Craig Citro, Alex Clemesha, John Cremona, Didier Deshommes, David Harvey, Naqi Jaffery, David Joyner, Josh Kantor, Kiran Kedlaya, David Kirkby, Emily Kirkman, David Kohel, Jon Hanke, Bill Hart, Robert Miller, Bobby Moretti, Gregg Musiker, Andrey Novoseltsev, Bill Page, Fernando Perez, Yi Qiang, David Roe, Michael Rubinstein, Nathan Ryan, Kyle Schalm, Steven Sivek, Jaap Spies, Gonzalo Tornaria, Justin Walker, Mark Watkins, Joe Weening, Joe Wetherell, ...

- **Undergraduates**: have many **interesting** ideas; superb at researching available free software; amazing programmers.
- **Graduate students**: excellent at implementing optimized code and finding fast algorithms.
- **Faculty and computer professionals**: general direction, great writing, and quality control.

Funding: Miniscule compared to that of Magma, Maple, etc. UW Math department startup money, NSF VIGRE money, my NSF grant, the MSRI (workshops), IPAM (workshop)

Active UW Student SAGE Developers

1. **Tom Boothby** (undergrad) – The SAGE Notebook, powering algorithms, elliptic curves
2. **Robert Bradshaw** (grad) – Linear algebra, number theory
3. **Josh Kantor** (grad) – Numerical computation; 3d graphics
4. **Emily Kirkman** (undergrad) – some linear algebra; lots of work on the SAGE graph theory package.
5. **Robert Miller** (grad) – Graph theory
6. **Bobby Moretti** (undergrad) – Symbolic calculus, some group theory, packaging, chroot jails.
7. **Andrey Novoseltsev** (grad) – combinatorial geometry, polytopes
8. **Yi Qiang** (undergrad) – Distributed SAGE

SAGE Days 2 at UW: Coding Sprints...

Bobby Moretti (UW undergrad), Robert Miller (UW grad), David Harvey (Harvard grad), Joel Mohler (grad), David Joyner (USNA), Bill page (Axiom).

Upcoming SAGE-related Workshops I'm Organizing

- **Parallel Computation Workshop** at MSRI, Jan 29–Feb 2, 2007. I'm the organizing committee chair.
- **SAGE Days 3** at IPAM (in LA) Feb 17–21, 2007.

SAGE DAYS 3

February 17-21, 2007 UCLA
http://modular.math.washington.edu/sage/

- **AIM** workshop, databases in SAGE, July 30-Aug 3, 2007.

The SAGE Website

The SAGE Website

1. **Website:** `http://modular.math.washington.edu/sage`
2. **Free online SAGE notebook:** `http://sage.math.washington.edu:8100`
3. **Documentation:** Tutorial, Install Guide, Programming Guide, Reference Manual, Constructions.
4. **Targeted Platforms:** OS X, Linux, and MS Windows (Cygwin).
5. **Mailing Lists:** **sage-devel**, sage-announce, sage-forum, sage-support, and sage-uw.
6. **Wiki:** `http://sage.math.washington.edu:9001/`
7. **Bug Tracker:** `http://sage.math.washington.edu:9002/sage_trac`
8. **IRC Chatroom:** `#sage-dev` on irc.freenode.net

What is SAGE?

SAGE has **three** complementary aspects:

1. **A Free Distribution** of open source math software. 80MB source tarball that builds **easily** on OS X, Linux, and MS Windows.
2. **New Functionality** that fill in gaps in what is available elsewhere.
3. **A Unified Interface** to most math software: to Magma, Macaulay2, Singular, Maple, MATLAB, Mathematica, Axiom, etc.

What is SAGE?

1. A Free Distribution

Arithmetic	**GMP, MPFR**
Commutative algebra	**Singular** (libcf, libfactory)
Cryptography	**OpenSSL, PyOpenSSL, PyCrypto**
Group theory and combinatorics	**GAP**
Graph theory	**NetworkX**
Number Theory	**PARI, NTL**
Numerical computation	**GSL, Numpy**
Calculus, Symbolic comp	**Maxima**
Specialize dmath	many C/C++ programs...
Command Line	**IPython**
Graphical Interface	**Notebook**, **jsmath, Moin wiki**
Plotting	**Matplotlib, Tachyon, libgd**
Networking	**Twisted**
Database	**ZODB**, Python Pickles
Programming language	**Python**, **SageX** (compiled python)

The components of SAGE are all active open source projects.

What is SAGE?

2. New Functionality

Python and SageX code — **readable**:

```
algebras    edu         lfunctions  monoids         sets
categories  ext         libs        plot            structure
coding      functions   matrix      quadratic_forms tests
combinat    geometry    misc        rings
crypto      groups      modular     schemes
databases   interfaces  modules     server
```

```
Over 100,000 new lines (including docstrings):
$ cat */*.py */*/*.py */*/*/*.py */*.pyx \
                */*/*.pyx */*/*.pyx |sort |uniq | wc -l
102932
May Documentation Examples:
$ cat */*.py */*/*.py */*/*/*.py */*.pyx \
          */*/*.pyx */*/*.pyx |sort|uniq|grep "sage:" | wc -l
11899
```

Definition (Bernoulli Numbers)

The **Bernoulli numbers** B_n for $n \geq 0$ are *defined* by:

$$\frac{x}{e^x - 1} = \sum_{n=0}^{\infty} B_n \frac{x^n}{n!}. \tag{1.1}$$

Expanding the series further, we obtain the following table:

$$B_0 = 1, \quad B_1 = -\frac{1}{2}, \quad B_2 = \frac{1}{6}, \quad B_4 = -\frac{1}{30}, \quad B_6 = \frac{1}{42},$$

$$B_8 = -\frac{1}{30}, \quad B_{10} = \frac{5}{66}, \quad B_{12} = -\frac{691}{2730}, \quad B_{14} = \frac{7}{6},$$

$$B_{16} = -\frac{3617}{510}, \quad B_{18} = \frac{43867}{798}, \quad B_{20} = -\frac{174611}{330},$$

$$B_{22} = \frac{854513}{138}, \quad B_{24} = -\frac{236364091}{2730}, \quad B_{26} = \frac{8553103}{6}.$$

Relevant to computing integrals, infinite sums, and very important in number theory.

SAGE Demo: New Code (interactive help)

```
sage: bernoulli?   # one ? for help
    Return the n-th Bernoulli number, as a rational number.
    INPUT:
        n -- an integer
        algorithm:
            ’pari’  -- (default) use the PARI C library;
                       by *far* the fastest.
            ’gap’   -- use GAP
            ’gp’    -- use PARI/GP interpreter
            ’magma’ -- use MAGMA
            ’python’-- use pure Python implementation

    EXAMPLES:
        sage: bernoulli(12)
        -691/2730
        sage: bernoulli(50)
        495057205241079648212477525/66
    ...
    AUTHORS: David Joyner and William Stein
```

SAGE Demo: New Code (interactive help)

```
sage: bernoulli??          # two question marks for source code
File: ... python2.5/site-packages/sage/rings/arith.py
...
    if algorithm == ’pari’:
        x = pari(n).bernfrac()    # Use the PARI C library
        return Rational(x)
    elif algorithm == ’gap’:
        x = sage.interfaces.gap.gap(’Bernoulli(%s)’%n)
        return Rational(x)
    elif algorithm == ’magma’:
        x = sage.interfaces.magma.magma(’Bernoulli(%s)’%n)
        return Rational(x)
    elif algorithm == ’gp’:
        x = sage.interfaces.gp.gp(’bernfrac(%s)’%n)
        return Rational(x)
    elif algorithm == ’python’:
        return sage.rings.bernoulli.bernoulli_python(n)
    else:
        raise ValueError, "invalid choice of algorithm"
```

SAGE Demo: Unique New Code

```
sage: bernoulli_mod_p?
    Computes bernoulli numbers B_0, B_2, ... B_{p-3}
    modulo p.
    PERFORMANCE: Should be complexity O(p log p).
    INPUT: p -- integer, a prime
    OUTPUT: list -- the bernoulli numbers modulo p.
    EXAMPLES:
        sage: bernoulli_mod_p(37)
         [1, 31, 16, 15, 16, 4, 17, 32, 22, 31, 15,
          15, 17, 12, 29, 2, 0, 2]
    AUTHOR: David Harvey (2006-08-06)
```

This implements a famous algorithm of Buhler et al.

And there is much much more that is unique in SAGE.

What is SAGE?

3. Unified Interfaces

- SAGE **interfaces to**: Axiom, GAP, GP/PARI, Kash, Macaulay2, Magma, Maple, Mathematica, MATLAB, Maxima, Octave, Singular, etc.
- This gives SAGE a wide range of **functionality**.
- Unified **command completion and help**.

SAGE Demo: Interfaces

HOW THEY WORK: Use buffered psuedo-tty, files, and Python objects that wrap native objects. This makes it possible to wrap **all** math software that has a command line interface using similar code.

```
sage: x = gp('9+6')     # the GP/PARI math software
```

This fires up one copy of GP/PARI (if it wasn't already started) and sends the line `'sage[1] = 9+6'` to it. It also creates a Python class x with a field set to `"sage[1]"`.

```
sage: !ps ax |grep gp
16389  p5  Ss+    0:00.02 /Users/was/s/local/bin/gp --fast ...
sage: type(x)
<class 'sage.interfaces.gp.GpElement'>
sage: x
15
sage: x.name()
'sage[1]'
sage: x.factor()
[3, 1; 5, 1]
```

Overall Structure of SAGE

The Overall Structure of SAGE

- **Custom package management system** – 53 standard packages and 24 optional ones. Automated upgrades.
- **Interactive command-line** interface – IPython.
- **Graphical user interface** – via your web browser (AJAX app).
- **Fast underlying arithmetic** – built on mature C/C++ libraries (GMP, NTL, PARI, GSL). New code in C, SageX and Python.
- **Interfaces with other software** use buffered **psuedo-tty**'s.
- **Special purpose components** – **Lcalc**, **GMP-ECM** and **FlintQS** (integer factorization), **PALP** (integer polytopes), and many more.
- **Mercurial revision control system** – included standard; encourages users to be developers.

The SAGE Notebook: GUI For Mathematics Software

1. The SAGE Notebook – an **“AJAX application”** like Google maps or Gmail.
2. **Written from scratch** by me, Alex C. and Tom B.
3. Uses Python’s built-in **BaseHTTPServer** web server (we will switch to Twisted for robustness and security).
4. Works well with Firefox and Safari.
5. Client/server model which works **over network** or locally.
6. Current version is **stable and in use by many people**.
7. Try it: `http://sage.math.washington.edu:8101`

What Next? Goals for SAGE-3.0

SAGE in 2007: “Speed and Polish”

SAGE has a huge range of features, but is overall **not fast enough** to solidly compete with Magma and **not polished enough** to compete with Maple/Mathematica/MATLAB.

1. **Optimize everything** in SAGE. (Example: SAGE devs Hart and Harvey have nearly finished the first version of a new C library for vastly faster arithmetic in $\mathbb{Z}[x]$ than in any other math software in the world.)
2. **Design and implement parallel algorithms** at many different levels to take advantage of multicore and SMP computers.

Summary

1. SAGE is **free software** for research in **algebra**, **geometry**, **number theory**, **cryptography**, and **numerics**.
2. SAGE is an **environment for rigorous mathematical computation** built using Python, GAP, Maxima, Singular, PARI, etc., and provides a **unified interface** to Mathematica, Maple, Magma, MATLAB, etc.
3. There have been **several successful SAGE workshops**, and there are many active SAGE developers.
4. The **primary goal** of SAGE is to make powerful mathematical algorithms available in an integrated package with a modern interpreted language and graphical interface.