SAGE: A basic overview[1]

http://www.sagemath.org/

David Joyner

Sep 11, 2007

[1] Happy birthday to Tony Gaglione (amg@usna.edu)!

Outline

The GOAL

Create the **best available software** for number theory, algebra, geometry, and numerical computation, using the best possible GPL-compatible (open source) software.

Included are: **GAP, Singular, Pari, Maxima, SciPy** and more.

The Python part of SAGE is primarily due to the mathematician **William Stein**, who heads the project and is at the University of Washington, in Seattle. A SAGE recent lecture was recorded on video:

http://www.sagemath.org/talks/2007-cse-colloquium-movie.html

Much of this talk was pasted together from various talks of William..

Open Source

1. With SAGE **everything is open source** and the system is setup to strongly encourage looking at code. In fact, arbitrary modifications and redistribution of every single line must be allowed.
2. SAGE should be well-documented and easy to install and use.
3. Give credit to authors of GPL'd packages which SAGE includes.

Open Source

An invitation to computational group theory:

"... in my view it is necessary to *avoid everything* that would separate work on the design and implementation of algorithms from other mathematical work. Rather, we have to make every possible effort to adjust to habits and to adopt rules of conduct that are common practice in other parts of mathematics. "

Joachim Neubüser

Core Components of SAGE: All Bases are Covered

Basic Arithmetic	**GMP, PARI, NTL**
Command Line	**IPython**
Commutative algebra	**Singular**
Graphical Interface	**SAGE Notebook**
Graphics	**Matplotlib, Tachyon**
Group theory and combinatorics	**GAP**
Interpreted programming language	**Python**
Networking	**Twisted**
Numerical computation	**SciPy, GSL, etc.**
Source control system	**Mercurial**
Symbolic computation, calculus	**Maxima**

To be a component of SAGE, the software must be:
free, open source, robust, high quality, and portable
Nothing else should be included in the core SAGE package.

SAGE $\neq$ The Borg

The Borg are a Star Trek race characterized by their relentless pursuit of targets for assimilation.

We are not The Borg!

From the beginning SAGE has always tried hard to promote, and improve if possible, the software it includes. For example, we always ask that **credit to be given to GAP** whenever GAP is used in SAGE.

SAGE $\neq$ The Borg

Sage contributes upstream in form of patches/improvements.

Examples: LinBox and libSingular for Singular: ports to Cygwin and/or Solaris, build fixes in general for more exotic architectures.

Also, wider exposure of "specialized" systems like lcalc or mwrank. That increased userbase translates in suggestions and bug fixes upstream ....

We are a community that **tests** other mathematical software packages.

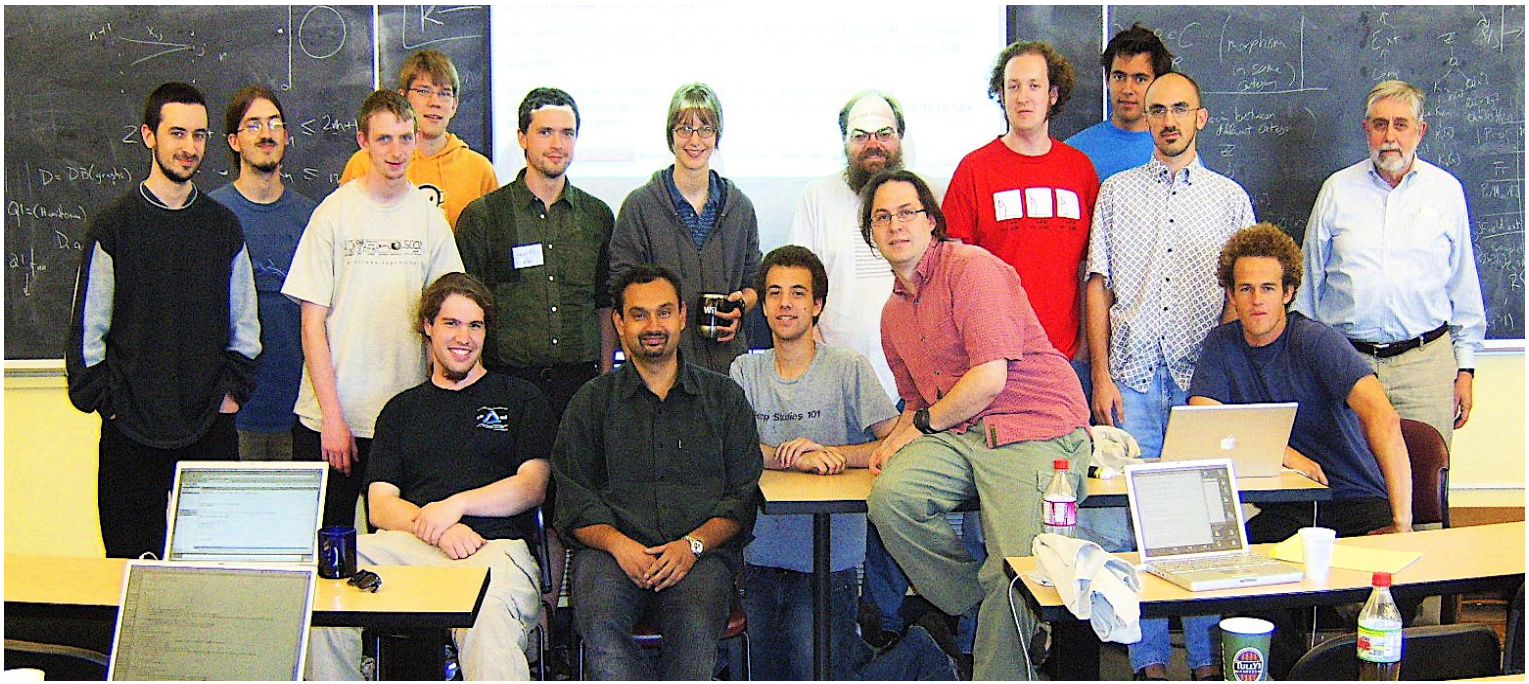
Who is Writing SAGE?

SAGE Contributors Include:

William Stein (project leader), Michael Abshoff, Martin Albrecht, Nick Alexander, Tom Boothby, Robert Bradshaw, Iftikhar Burhanuddin, Craig Citro, Timothy Clemans, John Cremona, Wilson Cheung, Alex Clemesha, Doug Cutrell, Didier Deshommes, Nathan Dunfield, Jon Hanke, Mike Hansen, Bill Hart, David Harvey, Naqi Jaffery, David Joyner, Josh Kantor, Kiran Kedlaya, David Kirkby, Emily Kirkman, David Kohel, Jason Martin. Robert Miller, Joel Mohler, Bobby Moretti, Kate Minolta, Gregg Musiker, Andrey Novoseltsev, Bill Page, Yi Qiang, Dorian Raymer, David Roe, Kyle Schalm, Steven Sivek, Jaap Spies, Gonzalo Tornaria, Michel Vandenbergh, Justin Walker, Mark Watkins, Joe Weening, Joe Wetherell, Carl Witty, Cristian Wuthrich, Gary Zablackis.

Of course, some are much more active than others. The barrier to becoming a more active SAGE developer (or becoming less active, if you were one) is very low.

Some SAGE developers



Many of the developers attended SAGE Days 4 recently in Seattle.

http://www.sagemath.org/

SAGE machines:

`http://www.sagemath.org/`

`http://www.sagenb.com/` - where you can use SAGE online.

`http://sage.math.washington.edu/` - where SAGE developers have accounts and directories (most are viewable from the www).

Technology Overview

SAGE is based on Python

Python is a powerful modern interpreted programming language.

- "Python is fast enough for our site and allows us to **produce maintainable features in record times**, with a minimum of developers," said Cuong Do, Software Architect, **YouTube.com.**
- "Google has made no secret of the fact they use Python a lot for a number of internal projects. Even knowing that, once **I was an employee, I was amazed at how much Python code there actually is in the Google source code system**.", said Guido van Rossum, **Google**, creator of Python.
- "Python plays a key role in our production pipeline. Without it a project the size of **Star Wars: Episode II** would have been very difficult to pull off. From crowd rendering to batch processing to compositing, **Python binds all things together**," said Tommy Burnette, Senior Technical Director, **Industrial Light & Magic.**

Python is...

- Easy for you to **define your own data types** and methods on it. **permutation groups, abelian groups, matrix groups, rings, whatever**
- Very clean language that results in **easy to *read* code**.
- **Easy to learn**:
  - **Free:** Dive into Python `http://www.diveintopython.org/`
  - **Free:** Python Tutorial `http://docs.python.org/tut/`
- A **huge number of libraries**: statistics, networking, databases, bioinformatic, physics, video games, 3d graphics, and serious mathematics (via SAGE)
- Very easy to **use any C/C++ libraries** from Python.
- Excellent support for **string manipulation and bit fiddling**.
- Cython – a **Python compiler** (`http://www.cython.org`).

The SAGE Command Line

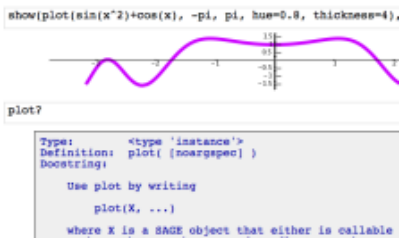
```
wdj@wooster:~/sagefiles/sage-2.8.3.rc3> ./sage
----------------------------------------------------------------------
| SAGE Version 2.8.3, Release Date: 2007-08-31                       |
| Type notebook() for the GUI, and license() for information.        |
----------------------------------------------------------------------

sage: 2^3
8
sage: gap_[TAB]
gap_console           gap_reset_workspace  gap_version
sage: gap_console()
GAP4, Version: 4.4.9 of 6-Nov-2006, x86_64-unknown-linux-gnu-gcc
gap>
```

1. Uses **IPython**, an amazing shell (history, completions, customizable, interface to pdb, etc.), created by Fernando Perez
2. Has TeXmacs and emacs command-line interfaces.

Be Curious

SAGE gives you easy access to **documentation** and **source code**.

- Type **plot?** for help on the plot command and `plot??` to see the source code.
- If **X** is anything, type **X.[tab key]** to see all commands that apply to **X**.




```
%magma
E := EllipticCurve([1,2,3,4,5]);
Rank(E);
    1

octave('rand(3)')
     0.861146 0.323013 0.0147563
     0.482702 0.888791 0.55302
     0.0686355 0.202225 0.22652
```

Part I: How SAGE **interfaces** with other computer algebra systems.

Pexpect and Pseudotty's

- **Pseudotty:** A device which appears to an application program as an ordinary terminal but which is *in fact* connected to a different process. Pseudo-ttys have a slave half and a control half.

  `gap_console()` **brings up a GAP prompt in SAGE**

- **Pexpect:** *makes Python a better tool for controlling other applications.* (`pexpect.sourceforge.net`) Pexpect is a pure Python module for spawning child applications; controlling them; and responding to expected patterns in their output.

  `gap.eval('gapcommand')` **sends** `'gapcommand'` **to GAP**

Difficulties

1. Getting subprocesses (and their children!) to quit when you quit SAGE.
2. I/O Prompts: make very obscure or embed control codes.
3. Control-C: How to break out of a computation.
4. Large I/O: use files.

How to Improve an Existing Interface

1. Systematically work through a standard tutorial for GAP but using SAGE; find something that is difficult, impossible, or *unnatural* to do. Add SAGE functionality to remedy the problem.
2. Write conversion functions ("wrappers") between SAGE objects and objects in GAP, e.g., free groups. We need far more of these!

A detailed example is in the SAGE Programming manual (using Lie algebra GAP commands).

What GAP group theory is currently wrapped is described in
`http://www.sagemath.org/doc/html/ref/node177.html`
(See also "Group theory in SAGE", with David Kohel. Also, some combinatorics and character-theoretic stuff is also wrapped.)

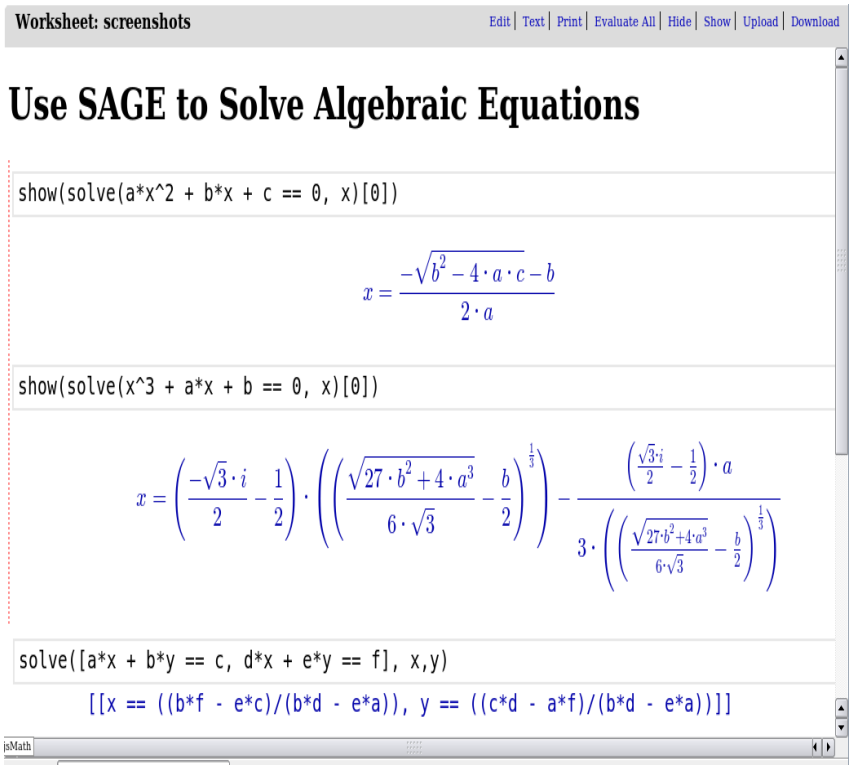
The SAGE Notebook

Interesting FACT: Most people polled **vastly prefer** using a good GUI for interacting with math software, if available.

SAGE **has one.**

1. The SAGE Notebook – An "AJAX application" like Google maps or gmail: lots of `CSS`, `Javascript`, and `XMLHTTPRequest`.
2. Written from scratch by William S., Alex C. and Tom B.
3. Uses Python's `Twisted web2` web server to provide a GUI.
4. Client/server model which works over network.
5. A very usable and robust version done.
6. `http://www.sagenb.com/`

The SAGE Notebook

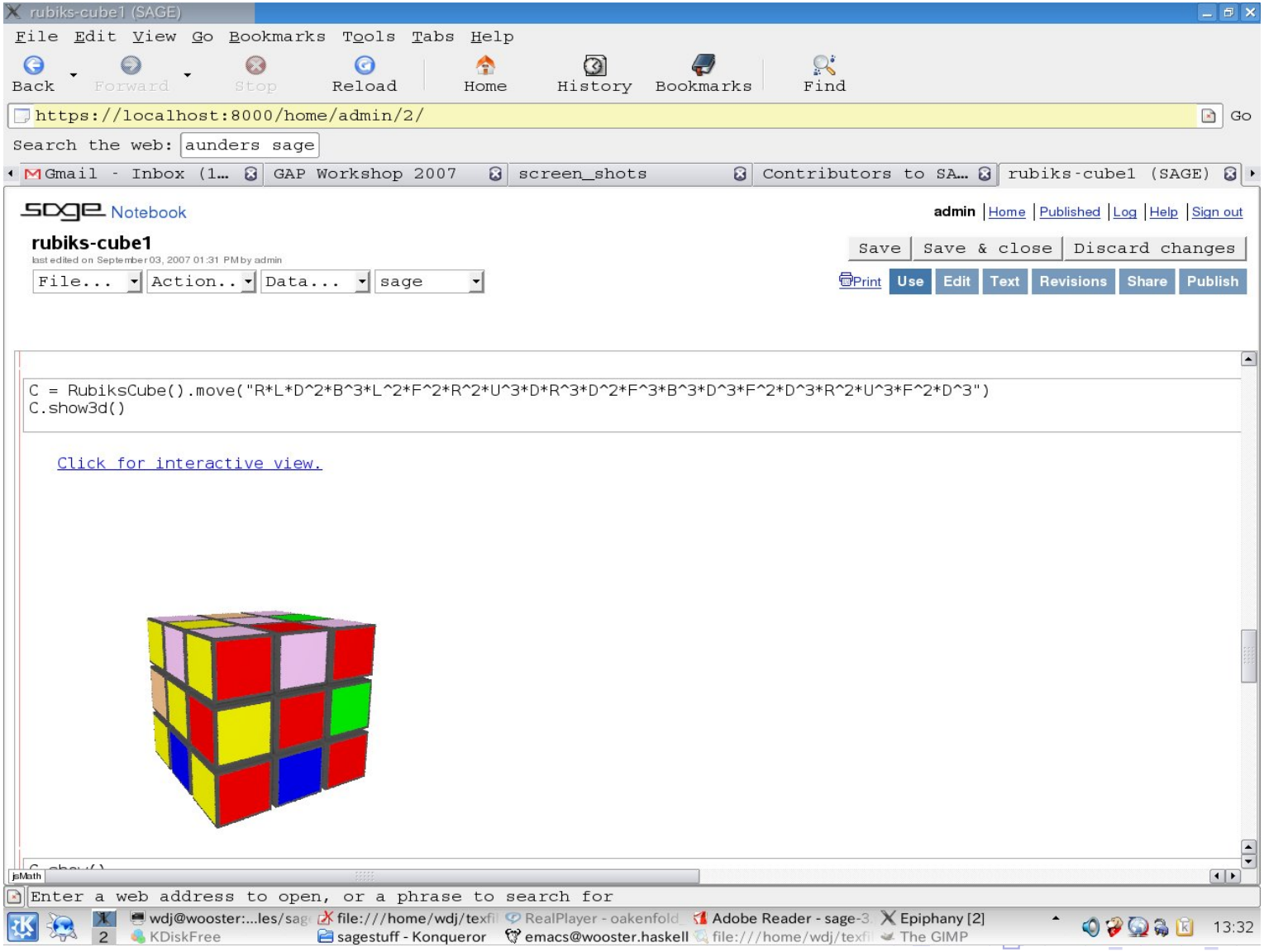


- Connect to SAGE running locally ***or elsewhere*** (via ethernet).
- Create **embedded graphics**
- **Typeset** mathematical expressions
- Add and delete input
- Start and interrupt **multiple calculations** at once.
- The notebook also works with **GAP**, Singular, latex, html, etc.!

The SAGE GUI ("notebook")

```
sage:  notebook()
Please choose a new password for the SAGE Notebook 'admin' user.
Do _not_ choose a stupid password, since anybody who could guess your password
and connect to your machine could access or delete your files.
NOTE: Only the md5 hash of the password you type is stored by SAGE.
You can change your password by typing notebook(reset=True).

Enter new password:
Retype new password:
Please login to the notebook with the username 'admin' and the above password.
Password changed for user 'admin'.
**************************************************
*                                                *
* Open your web browser to https://localhost:8000  *
*                                                *
**************************************************
There is an admin account.  If you do not remember the password,
quit the notebook and type notebook(reset=True).
2007/09/03 10:26 -0400 [-] Log opened.
2007/09/03 10:26 -0400 [-] twistd 2.5.0 (/home/wdj/sagefiles/sage-2.8.3.rc3/local/bin/python 2.5.1) starting up
2007/09/03 10:26 -0400 [-] reactor class: <class 'twisted.internet.selectreactor.SelectReactor'>
2007/09/03 10:26 -0400 [-] Loading sage_notebook/twistedconf.tac...
2007/09/03 10:26 -0400 [-] Loaded.
2007/09/03 10:26 -0400 [-] twisted.web2.channel.http.HTTPFactory starting on 8000
2007/09/03 10:26 -0400 [-] Starting factory <twisted.web2.channel.http.HTTPFactory instance at 0x124f680>
kfmclient: symbol lookup error: /usr/lib64/libXft.so.2: undefined symbol: FT_Library_SetLcdFilter


2007/09/03 10:27 -0400 [-] (Notebook cleanly saved.  Press control-C again to exit.)
```

SAGE notebook screenshot (an uploaded *.sws file)



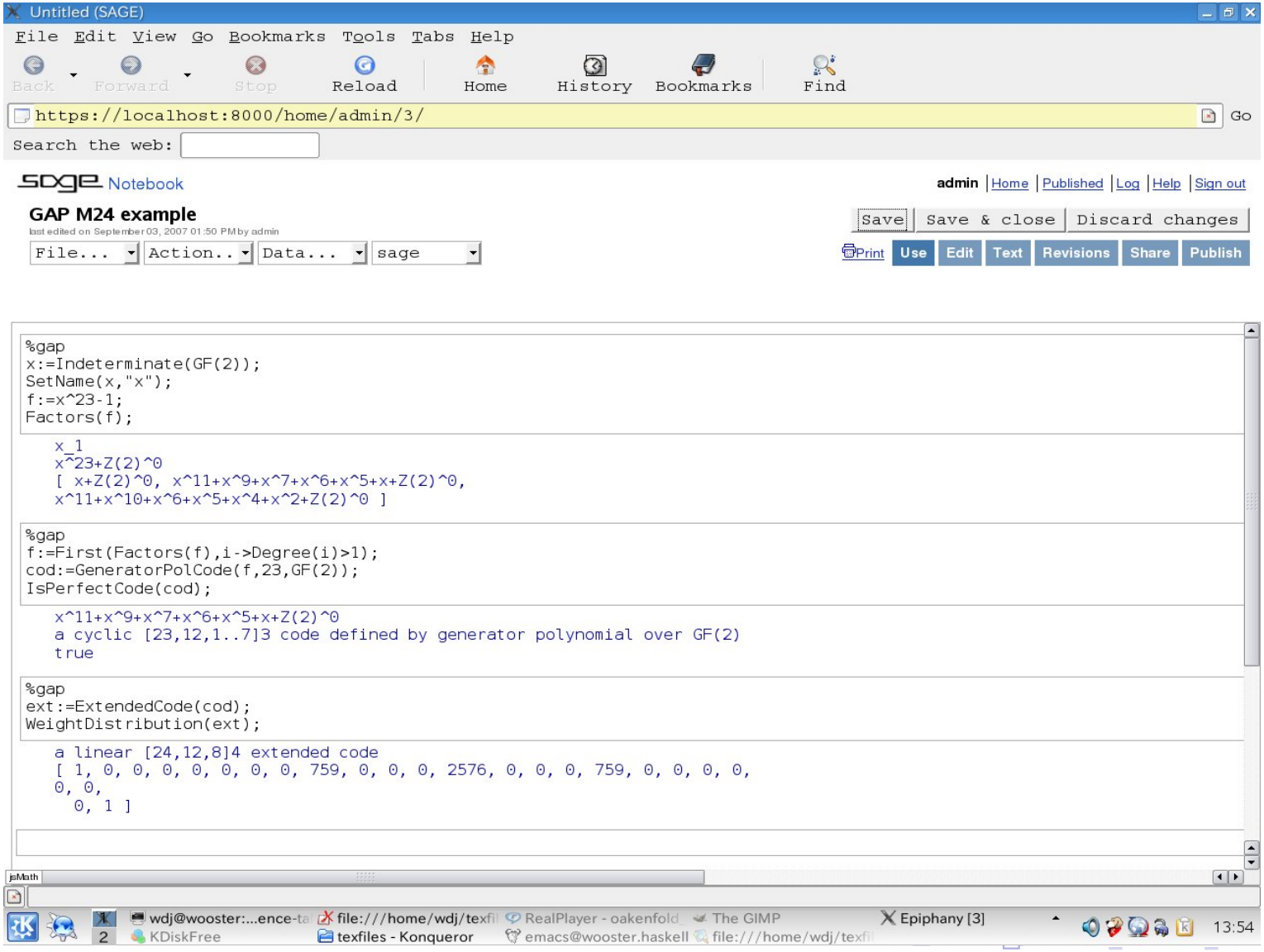
Interfaces – You can use anything from SAGE

Continue to use your favorite programs and code from within SAGE:

- SAGE includes (mostly pseudo-tty) interfaces to

  GAP, **GP/PARI**, Kash, Macaulay2, Magma, Maple, Mathematica, **Maxima**, Octave, **Singular**, etc.
- Red systems are included standard with SAGE.
- Get access to **100%** of the functionality of the other systems via interfaces. (But there is some overhead.)
- Get tab completion and online help.

SAGE notebook screenshot (a GAP session)



Cython – Fast compiled code

- If SAGE were written entirely in pure Python it would be slow.
- SAGE can be very fast, since it is partly written in **Cython** (a flavor of Pyrex), which is a Python-like language that is converted to C and compiled:
  `http://www.cosc.canterbury.ac.nz/greg.ewing/python/Pyrex/`
  `http://www.cython.org/`
- Much of SAGE's basic arithmetic types has been (re-)written this way.

SAGE multiplies two integers

Example: **SAGE creates and multiplies two integers** as follows:

1. Create two integers via a direct C-level wrapping of the GMP C library, i.e., using the functions `mpz_init` and `mpz_set`...
2. Multiply them using the GMP function `mpz_mul`.

The C code gets compiled and becomes an "extension" to the Python language. These "extensions" are potentially just as powerful as anything written in the core of Python.

```
def __mul_(Integer self, Integer other):
    cdef Integer x
    x = Integer()
    _sig_on   # so ctrl-c always works perfectly.
    mpz_mul(x.value, self.value, other.value)
    _sig_off
    return x
```

History and Status Report

Some history: SAGE 0.1 to SAGE 2.8.3

- Jan 2005: Spoke with William Stein at the Atlanta AMS National meeting for $\approx$ 15 mins. SAGE was called Manin then and only William had a copy. I convinced him to email me a copy ASAP. (This was the highlight of the meeting for me, in fact.)
- Feb 2005: SAGE *0.1*. This included Pari.
- April 2005: Created **interfaces** to Mathematica, Magma, etc.
- Oct 2005, SAGE *0.8*: GAP and Singular included as standard.
- Jan 2006: *SAGE 0.10*, Maxima and clisp included as standard.

Some history: SAGE 0.1 to SAGE 2.8.3

- Feb 2006: **SAGE Days 1** workshop, UCSD – **SAGE 1.0**
- ... lots of people join sage-devel ...
- May-July, 2006 (SAGE 1.3.*) GUI Notebook developed by William S., Alex C. and Tom B.
- Sep 30-Oct 3, 2007: **SAGE Days 5**, Cambridge (USA), funded by the Clay Math. Institute.
- Nov, 2007: **SAGE Days 6**, Heilbronn Institute.

See `http://wiki.sagemath.org/` for more details (lectures, mp3's of talks, etc.) on the various SAGE Days.

SAGE now has a *huge* range of functionality. It was the only system to do all the math-polyglot problems.

The SAGE Library (new code)

This was in **Aug 2006**:

```
...> ls --group-directories-first devel/sage/sage
algebras    coding     dsage      geometry  interfaces  matrix   modules      quadratic_forms  sets            a
calculus    combinat   ext        graphs    lfunctions  media    monoids      rings            structure       a
catalogue   crypto     functions  groups    libs        misc     plot         schemes          tests           _
categories  databases  games      gsl       logic       modular  probability  server           all_cmdline.py  v
```

The number of unique lines of code:

```
.../devel/sage/sage> cat *.py */*.py
*/*/*.py */*/*/*.py */*.pyx */*/*.pyx */*/*/*.pyx */*.pxd */*/*.pxd */
164017
```

The number of unique lines of input doctests in the source code:

```
.../devel/sage/sage> cat *.py */*.py
*/*/*.py */*/*/*.py */*.pyx */*/*.pyx */*/*/*.pyx */*.pxd */*/*.pxd
*/*/*/*.pxd |sort |uniq | grep "sage:" |wc -l
cat: */*/*/*.pyx: No such file or directory
cat: */*/*/*.pxd: No such file or directory
20768
```

What is included in SAGE

`http://www.sagemath.org/packages/standard/`

Standard (GPL compatible; easy build on Linux and OS X):
blas, cddlib, clisp, conway_polynomials, cremona_mini, cython, doc, ecm, examples, extcode, flintqs, fortran, freetype, gcl, gap (and GUAVA), genus2reduction, gfan, givaro, gmp, gnuplotpy, graphs, GSL, iml, ipython, lapack, lcalc, libpng, libgcrypt, LinBox, matplotlib, maxima, moin, mpfr, mwrank, ntl, networkx, numpy, palp, pari, pexpect, pycrypto, cylon, pyrexembed, python, readline, sage, scipy, singular, sqlite, sympow, tachyon, termcap, twisted, zlib, zodb3

These are stored as bzipped tarballs (`spkg` = `tar.bz2`).

What is included in SAGE

`http://www.sagemath.org/packages/optional/`

These may not be GPL'd but we have permission to redistribute (at least, no laywers have sent a "cease and desist" letter yet:-):

Optional Packages: `sage -i package_name`
ace, atlas, axiom4sage, biopython, database_cremona_ellcurve, database_gap, database_jones_numfield, database_kohel, database_odlyzko_zeta, database_sloane_oeis, database_stein_watkins_mini, dvipng, extra_docs, gap_packages, gd, gnuplot, hermes, kash3_linux, kash3_osx, lie, macaulay2, mayavi, moin, numarray, numpy, nzmath, openmpi, openssl, phcpack, polymake, pygtk, pyopenssl, trac

These are stored as bzipped tarballs (`spkg` = `tar.bz2`).

What is included in SAGE

Other:

- Install almost ***any Python package***: `sage -python setup.py`.
- Ubuntu: Use any standard system-wide Python code in SAGE:
```
import sys;
sys.path.append('/usr/lib/python2.5/site-packages')
```
- "Experimental" SAGE packages:
`http://www.sagemath.org/packages/experimental/`

Some Shortcomings of SAGE

1. There are currently probably **less than a thousand users** of SAGE (there are millions of Python users).
2. **Not robust enough** – sometimes Ctl-C doesn't interrupt, etc.
3. SAGE is **sometimes much slower** than Magma or Mathematica (and sometimes faster, to be fair).
4. The big problem – lack of **MONEY**.
5. SAGE is new – there are **bugs**, though there are regular "bug-squashing sessions".

If you think something is bad **you can sometimes fix it yourself**. Example, `number_of_partitions`...

Example: Number of Partitions

```
sage: list(partitions(5))
[(1, 1, 1, 1, 1), (1, 1, 1, 2), (1, 2, 2), (1, 1, 3),
 (2, 3), (1, 4), (5,)]
sage: number_of_partitions(5)
7
```

1. The beginning of the *Mathematica Tour* has an assertion that: **"Mathematica computes the number of partitions of 1 billion in a few seconds – a frontier number theory calculation".**
2. **SAGE (and Magma!) would take years** to do that. William Stein posted a question about this on sage-devel; **72 posts** among **15 people** followed.
3. Now – thanks to Jon Bober (U Mich grad student) SAGE is faster at this than any other program in the world:

   ```
   sage: time len(str(number_of_partitions(10^9)))
   CPU times: user 67.21 s, sys: 0.34 s, total: 67.56 s
   35219
   ```

   Mathematica 6.0 takes 83s, and 6.1 takes 77s.

Some Advantages of SAGE

1. SAGE is a serious general purpose CAS that uses a **mainstream programming language** (Python).
2. SAGE allows you to use GAP Maple, Mathematica, Singular, etc., all **together**.
3. SAGE has **more functionality out of the box** than any other open source mathematics software.
4. SAGE has a large, **active**, and well rounded **developer community**: `sage-devel` mailing list has **186** subscribers, working very hard on everything from highly optimized arithmetic, to high school education, to computing modular forms. It averages 25 messages a day.
5. SAGE **development is done in the open**. You can read about why all decision are made, have input into decisions, see a list of every change anybody has made, etc. This is **totally different than the situation with Magma and Mathematica.**

Web addresses

http://www.sagemath.org/

http://www.sagenb.com/

http://sage.math.washington.edu/**home/wdj/**

http://wiki.sagemath.org/