Synergy Effects

A Sage Introduction – http://www.sagemath.org

Martin Albrecht (M.R.Albrecht@rhul.ac.uk),
updated and edited by Paul Leopardi (paul.leopardi@anu.edu.au)

2008-10-23

Outline

1. Introduction

2. Features

3. Development model

4. Getting Started

Outline

1 Introduction

2 Features

3 Development model

4 Getting Started

Why? I

"Neubüser's Law"

"You can read [some] Theorem and its proof in [some] book in the library [. . .] then you can use [that] 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).

Why? II

Neil Sloane

```
From: N. J. A. Sloane <XXXX@XXXXXXXXXXX.XXX.XXX>
Date: 8 Nov 2007 06:28
Subject: Re: dumb question about installing pari-gp with fink

I would like to thank everyone who responded to
my question about installing PARI on an iMAC.

The consensus was that it would be simplest to install sage,
which includes PARI and many other things.

I tried this and it worked!

Thanks!

Neil

(It is such a shock when things actually work!!)
```

Mission Statement

Mess with the Best

Provide an open source, high-quality, and free viable alternative to **Magma, Mathematica, Maple** and **MATLAB**.

In other words: create a unified mathematics software package for algebra, calculus, elementary to very advanced number theory, cryptography, numerical computation, commutative algebra, group theory, combinatorics, graph theory, exact linear algebra and more.

To achieve this do not reinvent the wheel but **reuse** as much **existing building blocks** as possible and make sure the result is **rigorously tested**, **easy to modify** by the end user and **very well documented**.

Also create a **helpful environment** for users to get help (mailinglists, irc-channel, meetings, coding sprints).

What is Sage?

Sage is a mathematics software package developed by a **worldwide** community of developers.

1. a **distribution** of the best free, open-source mathematics software available that is easy to compile or install from binaries.
2. an **interface** to most free and commercial mathematics software packages (e.g. Magma, Mathematica)
3. a **huge new library**, which uniformly covers the widest area of **functionality**, including several new implementations not yet found elsewhere.

Welcome to Sage:

```
----------------------------------------------------------------------
| SAGE Version 3.1.2, Release Date: 2008-09-19                       |
| Type notebook() for the GUI, and license() for information.        |
----------------------------------------------------------------------

sage: 2+3
5
```

Outline

1. Introduction
2. Features
3. Development model
4. Getting Started

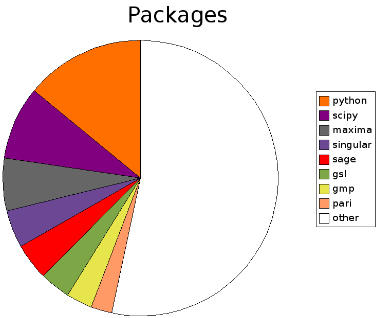
A Distribution I

Numerical Computation	**GSL, Numpy, Scipy**
Symbolic Computation	**Maxima, Sympy**
Arithmetic	**GMP, MPFR, Givaro, MPFI, Qd**
Linear Algebra	**LinBox, M4RI, IML, fpLLL, ATLAS, BLAS, LAPACK**
Optimization	**CVXOPT**
Statistics	**R**

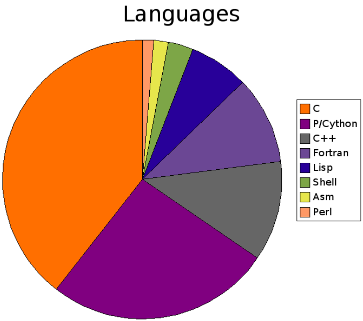
Commutative Algebra	**SINGULAR, libSINGULAR, PolyBoRi**
Integer Factorization	**FlintQS, GMP-ECM**
Group Theory	**GAP**
Combinatorics	**Symmetrica**
Graph Theory	**NetworkX**
Number Theory	**PARI, NTL, Flint, mwrank**
Lattice Polytopes	**PALP**
Cryptosystems	**GnuTLS, PyCrypto, OpenCDK, Libgcrypt, Libgpg-error**

User Interface	Sage Notebook, **jsmath, Moin wiki, IPython**
Programming Languages	**Python, Cython, Common Lisp, F2c, G95, Rpy, Weave**
Graphics	**Matplotlib, Tachyon, GD, libgd, Java3d, PNG**
Networking	**Twisted, GNU TLS**
Databases	**ZODB, SQLite**, Python Pickles
Revision Control	**Mercurial**

A Distribution II



Packages



Languages

Overall more than 4.5 million lines of source code and estimated several hundred person-years.

A Unified Interface

- SAGE **interfaces to**: Axiom, GAP, GP/PARI, Kash, Macaulay2, Magma, Maple, Mathematica, MATLAB, Maxima, MuPad, Octave, Singular, etc.

- This gives SAGE a wide range of **functionality**.

- Unified **command completion and help**.

Example:

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

This fires up one copy of GP/PARI and sends the line `'sage[1] = 9+6'` to it

```
sage: x, x.name()
15, 'sage[1]'
sage: x.factor()
[3, 1; 5, 1]
```

The Interface is a General Purpose Language: Python

`http://www.python.org` and `http://www.cython.org`

- easy for you to define **your own data types** and methods on it (bitstreams, ciphers, rings, whatever).
- very clean language that results in **easy to read code**.
- a **huge number of libraries**: statistics, networking, databases, bioinformatics, physics, video games, 3d graphics, numerical computation (Scipy), and serious “pure” mathematics (via Sage)
- easy to **use existing C/C++ libraries** from Python.
- Cython – an almost Python compiler.

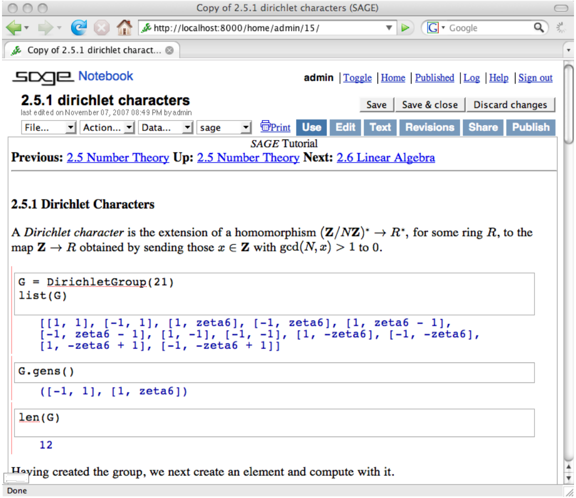
```
On 11/1/07, Jack <XXXX.XXXXXXXXX@XXXXX.XXX> wrote:
> Is it possible to fetch information from a database (say, PostgreSQL)
> with sage?  If so, how is it done?  If not, is there  a workaround
> that people use?

Using Psycopg2 is the most popular way of using PostgreSQL from python,

see:
http://www.initd.org/tracker/psycopg/wiki/PsycopgTwo
...
```

Web-based Notebook Interface

public notebooks available at `http://sage.milnix.org`



Copy of 2.5.1 dirichlet characters (SAGE)
http://localhost:8000/home/admin/15/
Google
Copy of 2.5.1 dirichlet charact...
Notebook
admin | Toggle | Home | Published | Log | Help | Sign out
Save
Save & close
Discard changes
File...
Action...
Data...
sage
Print
Use
Edit
Text
Revisions
Share
Publish
Done

2.5.1 dirichlet characters

last edited on November 07, 2007 08:49 PM by admin

SAGE Tutorial

Previous: 2.5 Number Theory **Up:** 2.5 Number Theory **Next:** 2.6 Linear Algebra

2.5.1 Dirichlet Characters

A *Dirichlet character* is the extension of a homomorphism $(\mathbf{Z}/N\mathbf{Z})^* \to R^*$, for some ring R, to the map $\mathbf{Z} \to R$ obtained by sending those $x \in \mathbf{Z}$ with $\gcd(N, x) > 1$ to 0.

```
G = DirichletGroup(21)
list(G)
```

```
    [[1, 1], [-1, 1], [1, zeta6], [-1, zeta6], [1, zeta6 - 1],
    [-1, zeta6 - 1], [1, -1], [-1, -1], [1, -zeta6], [-1, -zeta6],
    [1, -zeta6 + 1], [-1, -zeta6 + 1]]
```

```
G.gens()
```

```
    ([-1, 1], [1, zeta6])
```

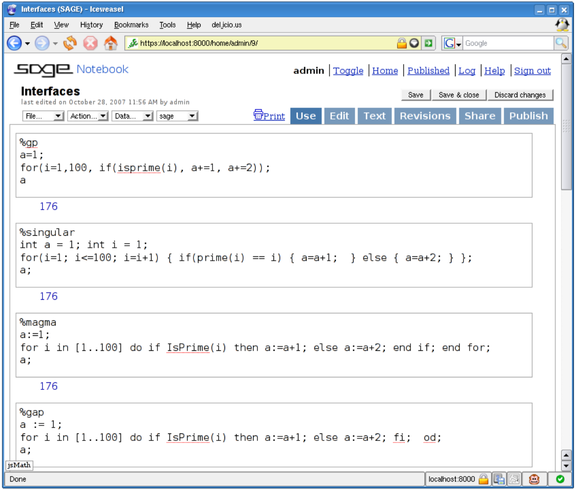
```
len(G)
```

```
    12
```

Having created the group, we next create an element and compute with it.

- graphical user interface
- 2d plotting
- $\LaTeX$ typesetting
- web service (**AJAX**, SSL) inspired by Google Docs
- worksheet sharing
- worksheet up/download

GUI to Many Mathematics Packages



Interfaces

last edited on October 28, 2007 11:56 AM by admin

```
%gp
a=1;
for(i=1,100, if(isprime(i), a+=1, a+=2));
a
```

```
    176
```

```
%singular
int a = 1; int i = 1;
for(i=1; i<=100; i=i+1) { if(prime(i) == i) { a=a+1;  } else { a=a+2; } };
a;
```

```
    176
```

```
%magma
a:=1;
for i in [1..100] do if IsPrime(i) then a:=a+1; else a:=a+2; end if; end for;
a;
```

```
    176
```

```
%gap
a := 1;
for i in [1..100] do if IsPrime(i) then a:=a+1; else a:=a+2; fi;  od;
a;
```

Examples:

- Pari
- Maxima
- Singular
- Gap
- Mathematica
- Maple
- ...

SAGE: Lots of new code

Unique lines of code and docstrings:

```
$ cat *.py */*.py ... */*/*/*.pxd |sort |uniq | wc −l
274241
```

Unique input lines of autotested examples:

```
$ cat *.py */*.py ... */*/*/*.pxd | grep ”sage:”
             | sort |uniq |wc −l
47589
```

Doctesting coverage:

```
$ sage −coverage .
...
Overall weighted coverage score:  59.9%
Total number of functions:  50521
```

Also, close **collaboration with upstream** authors:

build fixes, porting, testing on a wide range of platforms, end user support, contribution (e.g. libSingular)

Outline

1 Introduction

2 Features

3 Development model

4 Getting Started

How Did Sage Come About?

Sage is:

1. A huge amount of **extremely hard mostly volunteer work** and
2. **Refusal to acknowledge reality**, i.e., that Sage is impossible.

Seriously, how did Sage really come about?

1997–1999	(**Berkeley**) **HECKE** – C++ (modular forms)
1999–2005	(**Berkeley**, **Harvard**) William Stein wrote over 25,000 lines of **Magma** code but got frustrated with closed-source Magma, Mathematica, Maple, Matlab vs open source and Python.
Feb 2005	**Sage 0.1** released — a Python math library.
Feb 2006	Sage 1.0; **UCSD Sage Days 1**.
Oct 2006	Sage 1.4; **U Washington Sage Days 2**.
Feb 2007	Sage 2.1.4; **UCLA Sage Days 3**.
Jun 2007	Sage 2.6; **U Washington Sage Days 4**.
Oct 2007	Sage 2.8.6; **Clay Math Institute Sage Days 5**.
Nov 2007	Sage 2.8.12; **U Bristol Sage Days 6**.
Feb 2008	Sage 2.10.1; **UCLA Sage Days 7**.
Mar 2008	Sage 2.10.2; **Enthought Sage/Scipy Days 8**.
Jun 2008	Sage 3.0.2; **U Washington Sage Days 8.5**.
Aug 2008	Sage 3.1; **Simon Fraser U Vancouver Sage Days 9**.
Oct 2008	Sage 3.1.4; **LORIA Nancy Sage Days 10**.

A Selection of Contributors

More than **190 contributors** to Sage, including: William Stein, Tim Abbott, Michael Abshoff, Martin Albrecht, Nick Alexander, Jennifer Balakrishnan, Jason Bandlow, Francois Bissey, Jonathan Bober, Tom Boothby, Robert Bradshaw, Michael Brickenstein, Dan Bump, Iftikhar Burhanuddin, Ondrej Certik, Wilson Cheung, Craig Citro, Francis Clarke, Timothy Clemans, Alex Clemesha, John Cremona, Karl-Dieter Crisman, Doug Cutrell, Didier Deshommes, Dan Drake, Alexander Dreyer, Gabriel Ebner, Burcin Erocal, Gary Furnish, Alex Ghitza, Andrzej Giniewicz, Chris Gorecki, Jason Grout, Jon Hanke, Carlo Hamalainen, Marshall Hampton, Mike Hansen, Bill Hart, David Harvey, Neal Holtz, Sean Howe, Naqi Jaffery, Peter Jipsen, David Joyner, Michael Kallweit, Josh Kantor, Kiran Kedlaya, Emily Kirkman, David Kohel, Ted Kosan, Jason Martin, Robert Miller, Kate Minola, Joel Mohler, Bobby Moretti, Gregg Musiker, Pablo De Nápoli, Andrey Novoseltsev, Willem Jan Palenstijn, John Palmieri, Clement Pernet, Yi Qiang, Dorian Raymer, R. Rishikesh, David Roe, Bjarke Hammersholt Roune, Franco Saliola, Kyle Schalm, Anne Schilling, Harald Schilly, Jack Schmidt, Steven Sivek, Jaap Spies, Chris Swierczewski, Nicolas Thiery, Gonzalo Tornaria, John Voight, Justin Walker, Mark Watkins, Joe Wetherell, Carl Witty, Cristian Wuthrich, Dal S. Yu, Mike Zabrocki, Paul Zimmermann.

Communication

Discussion groups:

sage-support support requests, 784 members, ca. 470 messages per month

sage-devel development discussions, 664 members, ca. 950 messages per month

sage-newbie basic questions about Sage, 154 members, low traffic

sage-edu teaching and education discussions, 125 members, low traffic

IRC channel:

#sage-devel@irc.freenet.net very busy during bug squashes, usually at least two Sage developers around

Issue tracker:

trac `http://trac.sagemath.org`: bug and feature tracker for Sage.

The Sage Development Guidelines

1. Don't work on anything that is not on trac. All enhancement proposals, bug reports and tasks are available on `http://trac.sagemath.org`.
2. All discussions happen in the open on public mailing lists and/or on a public chat channel (IRC).
3. One release per week on average: release often, release early. Several different release managers.
4. Changes happening to the main repository can be tracked in real time online
5. Code that goes into Sage is refereed by at least one other developer
6. If code is rejected by the release maintainer every developer can appeal to a board of editors of the **Journal of Sage** and they vote on the inclusion of the patch.

Also, developer's $HOME directories on our main development server `http://sage.math.washington.edu` are world readable through a web browser.

Outline

1 Introduction

2 Features

3 Development model

4 Getting Started

Getting Started With Sage

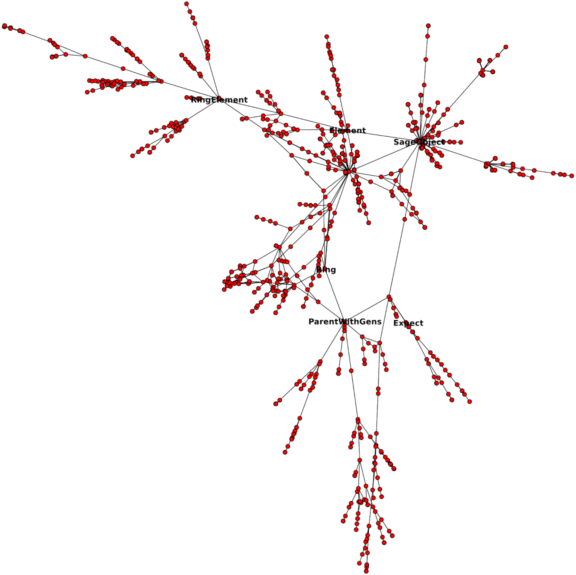
Ways to use Sage:

- download source; type 'make', wait, run (Linux, OSX, Solaris (soon))
- download binaries for a wide range of platforms
- download VMWare image (e.g. for Windows)
- try it online: `http://sage.milnix.org`

Documentation:

- Sage installation guide
- interactive tutorial
- comprehensive reference manual
- "How Do I Construct . . ." manual
- Sage programming guide
- "Sage programming for Newbies" book
- `http://wiki.sagemath.org`

Questions?



Thank You!