Sage: an open-source mathematical software system

Alasdair McAndrew

VICTORIA UNIVERSITY | A NEW SCHOOL OF THOUGHT

23 June 2010 / Monash University

Outline

Outline

Since about 1990, in my teaching

Started investigating Maple and Derive, and later Matlab.

- wrote labs for calculus, discrete mathematics, cryptography, image processing
- wrote lots of Maple and Matlab procedures
- did a little research
- wrote several articles and one textbook

Since about 2006, in my teaching

Started learning about open-source software

Since about 2006, in my teaching

Started learning about open-source software

- Discovered Maxima, used it for cryptography teaching
- wrote the first version of a finite fields package (now a standard part of Maxima), and started on a z-transforms package
- also discovered Axiom, used also for cryptography teaching
- wrote several articles (and many blog posts) about them

Problems with commercial software

- get locked into expensive license agreements
- students can't take software home to play with, or carry it around on their laptops
- Beacuse software is closed source, mathematics results based on software computations can't be verified

Outline

1. Mathematics software and me
2. Enter Sage
3. Developing Sage
4. Where to from here?
5. Sage examples

Sage: my newest discovery

In 2009 I started using Sage, and I've come to believe that it represents the best possible future for mathematics teaching, learning and research.

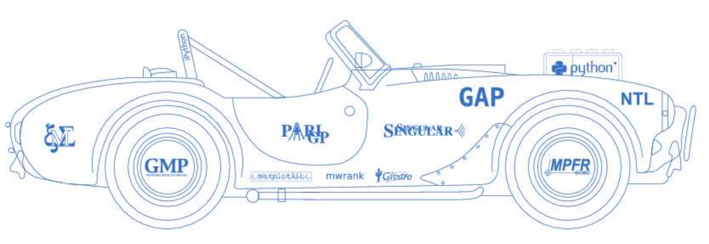
What is Sage?

Sage is a “mathematics software system” composed of free and open source software.

Its mission is to create a viable, free, open source alternative to Maple, Mathematica, Matlab and Magma.

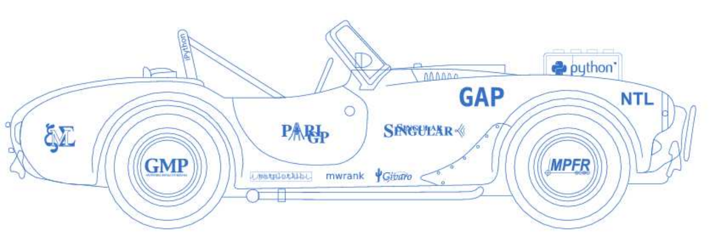
History of Sage

2005: First release (by William Stein): designed to provide a open-source alternative to Maple, Mathematica, Matlab, Magma etc
About "building the car", not "re-inventing the wheel"



Then known as SAGE: Software for Algebra and Geometry Exploration.

2005: First release (by William Stein): designed to provide a open-source alternative to Maple, Mathematica, Matlab, Magma etc
About "building the car", not "re-inventing the wheel"



Then known as SAGE: Software for Algebra and Geometry Exploration.

2010: Current release 4.4.3. Hundreds of developers world-wide, in many branches of mathematics.

What's it made of?

Sage is uses the well-known and powerful language Python for new code and to glue together many high-quality free software packages:

What's it made of?

Sage is uses the well-known and powerful language Python for new code and to glue together many high-quality free software packages:

- GAP: Groups, Algorithms, Programming
- Maxima: general purpose Computer Algebra System
- Pari/GP: Number Theory Calculator
- R: Statistical computing
- Singular: fast commutative and noncommutative algebra

What's it made of?

Sage is uses the well-known and powerful language Python for new code and to glue together many high-quality free software packages:

- GAP: Groups, Algorithms, Programming
- Maxima: general purpose Computer Algebra System
- Pari/GP: Number Theory Calculator
- R: Statistical computing
- Singular: fast commutative and noncommutative algebra

As well as:

What's it made of?

Sage is uses the well-known and powerful language Python for new code and to glue together many high-quality free software packages:

- GAP: Groups, Algorithms, Programming
- Maxima: general purpose Computer Algebra System
- Pari/GP: Number Theory Calculator
- R: Statistical computing
- Singular: fast commutative and noncommutative algebra

As well as: ATLAS, BLAS, Bzip2, Cddlib, Common Lisp, CVXOPT, Cython, mwrank, F2c, Flint, FpLLL, FreeType, G95, GD, Genus2reduction, Gfan, Givaro, GMP, GMP-ECM, GNU TLS, GSL, JsMath, IML, IPython, LAPACK, Lcalc, Libgcrypt, Libgpg-error, Linbox, M4RI, Matplotlib, Mercurial, MoinMoin Wiki, MPFI, MPFR, ECLib, NetworkX, NTL, Numpy, OpenCDK, PALP, Pexpect, PNG, PolyBoRi, PyCrypto, Python, Qd, Readline, Rpy, Scipy, Scons, SQLite, Sympow, Symmetrica, Sympy, mpmath, Tachyon, Termcap, Twisted, Weave, Zlib, ZODB

What's it made of?

Sage is uses the well-known and powerful language Python for new code and to glue together many high-quality free software packages:

- GAP: Groups, Algorithms, Programming
- Maxima: general purpose Computer Algebra System
- Pari/GP: Number Theory Calculator
- R: Statistical computing
- Singular: fast commutative and noncommutative algebra

As well as: ATLAS, BLAS, Bzip2, Cddlib, Common Lisp, CVXOPT, Cython, mwrank, F2c, Flint, FpLLL, FreeType, G95, GD, Genus2reduction, Gfan, Givaro, GMP, GMP-ECM, GNU TLS, GSL, JsMath, IML, IPython, LAPACK, Lcalc, Libgcrypt, Libgpg-error, Linbox, M4RI, Matplotlib, Mercurial, MoinMoin Wiki, MPFI, MPFR, ECLib, NetworkX, NTL, Numpy, OpenCDK, PALP, Pexpect, PNG, PolyBoRi, PyCrypto, Python, Qd, Readline, Rpy, Scipy, Scons, SQLite, Sympow, Symmetrica, Sympy, mpmath, Tachyon, Termcap, Twisted, Weave, Zlib, ZODB, and more…

Based on Python

- Mature, well-tested and well-designed programming language.
- Huge benefit from many years of continuous development.
- Lots of libraries for specific purposes, and documentation in many languages.

Open Source

From Jacob Neubüser, creator of GAP:

> *You can read Sylow's Theorem and its proof in Huppert's book in the library without even buying the book and 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 for the total time of their use. In order to protect what you pay for, you do not get the source, but only an executable, i.e. a black box. You can 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.*

Jacob Neubüser, continued:

> *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?*

Outline

1. Mathematics software and me
2. Enter Sage
3. Developing Sage
4. Where to from here?
5. Sage examples

The development model

- William Stein is still officially the lead developer.
- Each release is handled by a “release manager”.
- “Release early, release often”.
- Code is written by users, and then peer-reviewed before inclusion.
- Developers include students (school, UG & PG), academics, professionals.
- If you want more functionality—code it up and submit it!

How you can contribute

- Write code to implement functionality not currently available.
- Translate documentation.
- Write tutorials.
- Improve the website and the notebook interface.
- Teach with Sage, and write about it.
- Use Sage in your research, and write about it.

Outline

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

Visit `http://www.sagemath.org` to learn about and download Sage,

Visit `http://www.sagemath.org` to learn about and download Sage,

Visit `http://sagenb.org`

Visit `http://www.sagemath.org` to learn about and download Sage,

Visit `http://sagenb.org` to use Sage.

Visit `http://www.sagemath.org` to learn about and download Sage,

Visit `http://sagenb.org` to use Sage.

. . . and now, let's try it out. . .

Outline

Arbitrary sized integer arithmetic

```
sage: 2^1000
10715086071862673209484250490600018105614048117
05533607443750388370351051124936122493198378815
69585812759467291755314682518714528569231404359
84577574698574803934567774824230985421074605062
37114187795418215304647498358194126739876755916
55439460770629145711964776865421676604298316526
24386837205668069376
```

```
sage: factorial(250)
32328562609091077323208145520243684709948437176
73780666747942427112823747555111209488817915371
02819945092850735318943292673093171280899082279
10302790712819216765272401892647332180411862610
06832925365133678939089569935713530175040513178
76007724793306540233900616482555224881943657258
60573992226412548329822048491377217766506412768
58807153128978777672951913990844377478702589172
97325515028324178732065818848206247858265980884
88255488000000000000000000000000000000000000000
00000000000000000000000
```

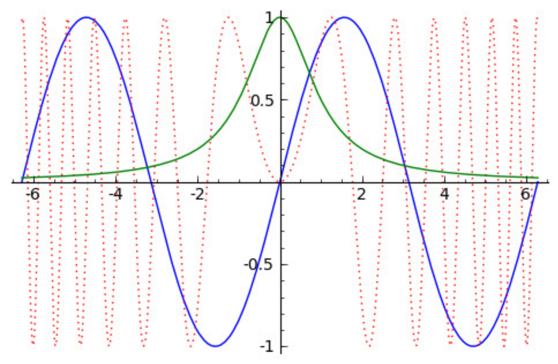
Arbitrary precision real arithmetic

```
sage: pi.n(digits=200)
3.141592653589793238462643383279502884197169399
37510582097494459230781640628620899862803482534
21170679821480865132823066470938446095505822317
25359408128481117450284102701938521105559644622
9489549303820
```

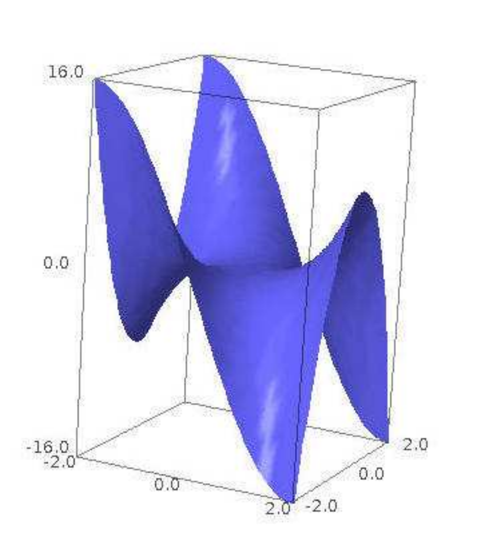
```
sage: R=RealField(500)
sage: R(exp(sqrt(163)*pi))
2.62537412640768743999999999999250072597198185 6
888793538563373369908627075374103782106479101 18
607312951181346186064504193083887949753864044 90
5728714476e17
```

Functions and plots

```
sage: A=plot(sin(x),(x,-2*pi,2*pi),color='blue')
sage: B=plot(sin(x^2),(x,-2*pi,2*pi),color='red'\
    ,linestyle=':')
sage: C=plot(1/(1+x^2),(x,-2*pi,2*pi),color='green
sage: show(A+B+C)
```



```
sage: var('x,y')
sage: pl=plot3d(x^3 - 3*x*y^2,(x,-2,2),(y,-2,2))
sage: pl.show(aspect_ratio=(1,1,0.2))
```



A bit of algebra

```
sage: s=(x^3/(x^3-1))^2;s
```

$$\frac{x^6}{(x^3-1)^2}$$

```
sage: s.partial_fraction()
```

$$\frac{4\,x+9}{9\left(x^2+x+1\right)}+\frac{4}{9\,(x-1)}+\frac{x+1}{3\left(x^2+x+1\right)^2}+\frac{1}{9\,(x-1)^2}+1$$

Polynomial factorization

Over the integers:

```
sage: var('x')
sage: f=x^6+1
sage: f.factor()
```

$(x^2+1)\cdot(x^4-x^2+1)$

Over a finite field:

```
sage: F.<x>=PolynomialRing(GF(11))
sage: f=x^6+1
sage: f.factor()
```

$(x^2+1)\cdot(x^2+5x+1)\cdot(x^2+6x+1)$

Over a finite prime power field:

```
sage: R.<a>=GF(7^2)
sage: F.<x>=PolynomialRing(R)
sage: factor(x^4+1)
```

$(x + 2a + 1) \cdot (x + 2a + 4) \cdot (x + 5a + 3) \cdot (x + 5a + 6)$

Now for some calculus

Differentiation:

```
sage: n=8
sage: f=diff((x^2-1)^n,x,n)/factorial(n)
sage: expand(f)
```

$$12870\,x^8 - 24024\,x^6 + 13860\,x^4 - 2520\,x^2 + 70$$

Integration:

```
sage: integrate(1/(1+x^3),x)
```

$$\frac{1}{3}\sqrt{3}\arctan\left(\frac{1}{3}(2\,x-1)\sqrt{3}\right) + \frac{1}{3}\log(x+1) - \frac{1}{6}\log\left(x^2-x+1\right)$$

With a twist: the Hill Cipher

```
sage: pl="SENDMEALLYOURMONEY"
sage: pll=map(lambda x:ord(x)-65,pl); pll
[18, 4, 13, 3, 12, 4, 0, 11, 11, 24, 14, 20, 17, 12, 14, 13, 4, 24]
sage: M=random_matrix(Zmod(26),3,3)
sage: gcd(M.det(),26)
1
sage: M
```

$$\begin{pmatrix} 5 & 23 & 24 \\ 19 & 24 & 8 \\ 10 & 6 & 21 \end{pmatrix}$$

```
sage: Mpl=matrix(Zmod(26),6,3,pll);Mpl
```

$$\begin{pmatrix} 18 & 4 & 13 \\ 3 & 12 & 4 \\ 0 & 11 & 11 \\ 24 & 14 & 20 \\ 17 & 12 & 14 \\ 13 & 4 & 24 \end{pmatrix}$$

```
sage: Mctl=(Mpl*M).list();Mctl
[10, 16, 9, 23, 17, 18, 7, 18, 7, 14, 20, 16, 11, 9, 18, 17, 19, 16]
sage: ctl=map(lambda x:chr(int(x)+65),Mctl);ctl
['R', 'C', 'G', 'P', 'B', 'H', 'L', 'R', 'Q',
 'U', 'G', 'C', 'Z', 'X', 'P', 'X', 'R', 'B']
sage: ct=reduce(lambda x,y:x+y,ctl);ct
'RCGPBHLRQUGCZXPXRB'
```

Some number theory

Recall the Elgamal cryptosystem: all users share a prime p with primitive root a.

1. Alice chooses $A < p$ as her private key and publishes $B = a^A \pmod{p}$ as her public key.
2. To encrypt a message $m < p$ to Alice, Bob chooses a random $k < p$ and sends the pair $(c_1, c_2) = (a^k, B^k m)$ to Alice.
3. Alice decrypts with $m = c_2/c_1^A \pmod{p}$.

Elgamal with small parameters

```
sage: p=197
sage: a=mod(primitive_root(p),p);a
2
sage: A=randint(1,p);A
121
sage: B=a^A;B
46
sage: m=100
sage: k=randint(1,p)
sage: c1,c2=a^k,B^k*m;c1,c2
(135, 132)
sage: c2/c1^A
100
```

Elgamal over a finite field

```
sage: pn=5^10
sage: F.<x>=GF(pn)
sage: a=F.multiplicative_generator();a
```
x
```
sage: A=randint(1,pn);A
```
8469825
```
sage: B=a^A;B
```
$x^9 + 4x^8 + 4x^7 + 4x^4 + x^3 + x^2 + 2$
```
sage: m=4*x^8
sage: k=randint(1,pn)
sage: c1,c2=a^k,B^k*m;c1;c2
```
$2x^9 + 4x^8 + 4x^7 + 2x^4 + 3x^3 + 3x^2 + 2x + 1$

$3x^9 + 3x^8 + 2x^7 + 2x^5 + 3x^3 + x^2 + 3x$
```
sage: c2/c1^A
```
$4x^8$

Combinatorics

```
sage: P=Permutations(20)
sage: P.cardinality()
2432902008176640000
sage: p=P.random_element();p
[17, 8, 20, 19, 4, 12, 13, 5, 11, 16, 1, 18, 15, 9, 14, 3, 6, 2, 7, 10]
sage: p.to_cycles()
[(1, 17, 6, 12, 18, 2, 8, 5, 4, 19, 7, 13, 15, 14, 9, 11), (3, 20, 10, 16)]
sage: P[10^15]
[1, 2, 5, 17, 16, 14, 15, 9, 13, 8, 6, 10, 12, 18, 20, 4, 11, 19, 3, 7]
```