Introduction à

Marc Mezzarobba

CNRS, LIP6, Université Paris 6

Qu'est-ce que Sage ?

« *Une alternative libre viable à Magma, Maple, Mathematica et Matlab* »

« *Construire la voiture au lieu de réinventer la roue* »

Essayer Sage

http://sagemath.org/

GNU GPL

http://sagenb.org/

http://cloud.sagemath.com/

Qu'est-ce que Sage ?

1. Une distribution
2. Une bibliothèque Python
3. Un système interactif

Une distribution

```
$ wget http://mirror/sagemath/src/sage-6.3.tar.gz
  && tar xfz sage-6.3.tar.gz && cd sage-6.3 && make
$ ./sage
```

ATLAS • *boehm_gc* •  cddlib • cephes • cliquer
cvxopt •  • *ECL* • eclib • ecm • *f2c* • fplll
FLINT • GAP • gfan •  • GLPK • GSL
IML •  ipython •  LAPACK • lcalc
 LinBox •   Maxima • M4RI
MPC • MPFI •  • mpmath • networkx • NTL
 • PALP •  •  •  • Pynac
 •  •  •  • symmetrica
sympow •  tachyon • zn_poly • + *d'autres...*

\+ ∼ 60 paquets optionnels + ∼ 60 paquets expérimentaux

Une bibliothèque Python

```
sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...
```

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique
  (hors doc + tests)

Une bibliothèque Python

```
sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...
```

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique (hors doc + tests)

Une bibliothèque Python

sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique (hors doc + tests)

Une bibliothèque Python

```
sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...
```

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique (hors doc + tests)

Une bibliothèque Python

```
sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...
```

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique (hors doc + tests)

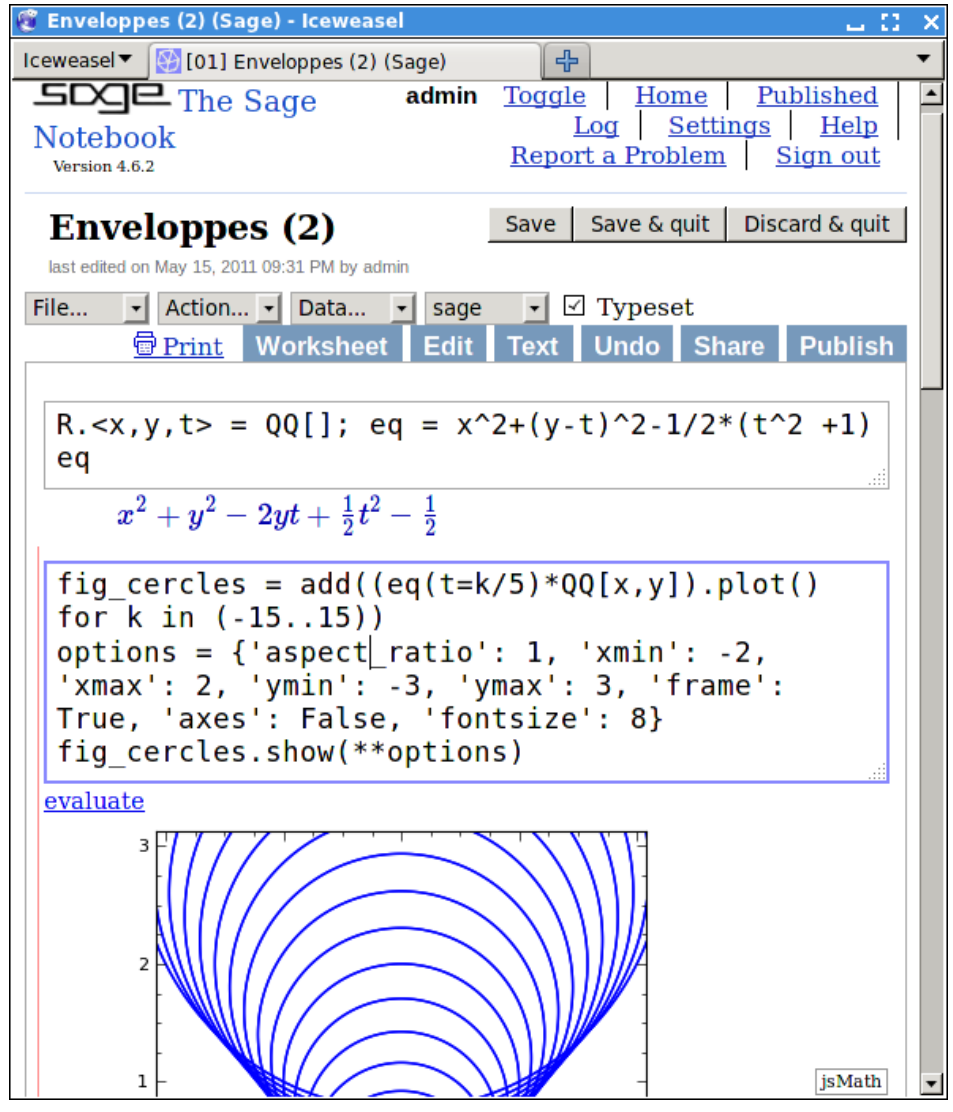
Une bibliothèque Python

```
sage.algebras
sage.calculus
sage.categories
sage.coding
sage.combinat
sage.crypto
sage.databases
sage.finance
sage.functions
sage.geometry
sage.graphs
sage.groups
sage.homology
sage.interfaces
sage.lfunctions
sage.logic
sage.matrix
sage.modular
sage.modules
sage.monoids
sage.numerical
sage.parallel
sage.plot
sage.rings
sage.sat
sage.schemes
sage.sets
sage.stats
sage.symbolic
...
```

- S'appuie sur les logiciels tiers embarqués
- $\simeq$ 600 000 lignes de code spécifique (hors doc + tests)

Un système interactif

```
~$ sage
------------------------------------------------------------------------
| Sage Version 4.6.2, Release Date: 2011-02-25                         |
| Type notebook() for the GUI, and license() for information.          |
------------------------------------------------------------------------
sage: taylor(exp(x), x, 0, 5)
1/120*x^5 + 1/24*x^4 + 1/6*x^3 + 1/2*x^2 + x + 1
sage:
sage: MatrixSpace(RR,5,3).random_element()
[-0.570390764900653  0.521446993576251 -0.950894560265950]
[-0.942431942330060  0.254122819002693  0.916721924359961]
[-0.195702504102615 -0.350489870318781 -0.214359534055980]
[ 0.487076746020482  0.461116221981387 -0.665179594662514]
[ 0.180194930460366  0.616390883848273 -0.389309976296204]
sage:
sage: import urllib2
sage: f = urllib2.urlopen("http://sagemath.org/")
sage: f.read(121)
'<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"\n"http
://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">'
sage:
```



Sage comme calculatrice

```
>>>  1+1

     2

>>>  factorial(50)

     30414093201713378043612608166064768844377641568960512000000000000

>>>  s = add(1/n for n in (1..30))

>>>  s

     9304682830147/2329089562800

>>>  s.n()

     3.99498713092039

>>>
```

Python

```
>>>  a = -2/3

>>>  type(a)

     <type 'sage.rings.rational.Rational'>

>>>  dir(a)[140:160]

     ['_singular_',
     '_singular_init_',
     '_sub_',
     '_sympy_',
     '_test_category',
     '_test_eq',
     '_test_nonzero_equal',
     '_test_not_implemented_methods',
     '_test_pickling',
     '_tester',
     'abs',
     'absolute_norm',
     'additive_order',
     'base_extend',
     'base_ring',
     'cartesian_product',
     'category',
     'ceil',
     'charpoly',
     'conjugate']

>>>  a.abs(), abs(a)

     (2/3, 2/3)

>>>  def myfact(n):
         res = 1
         for k in srange(1, n+1):
             res = res*k
         return res

>>>  myfact(50)

     30414093201713378043612608166064768844377641568960512000000000000

>>>
```

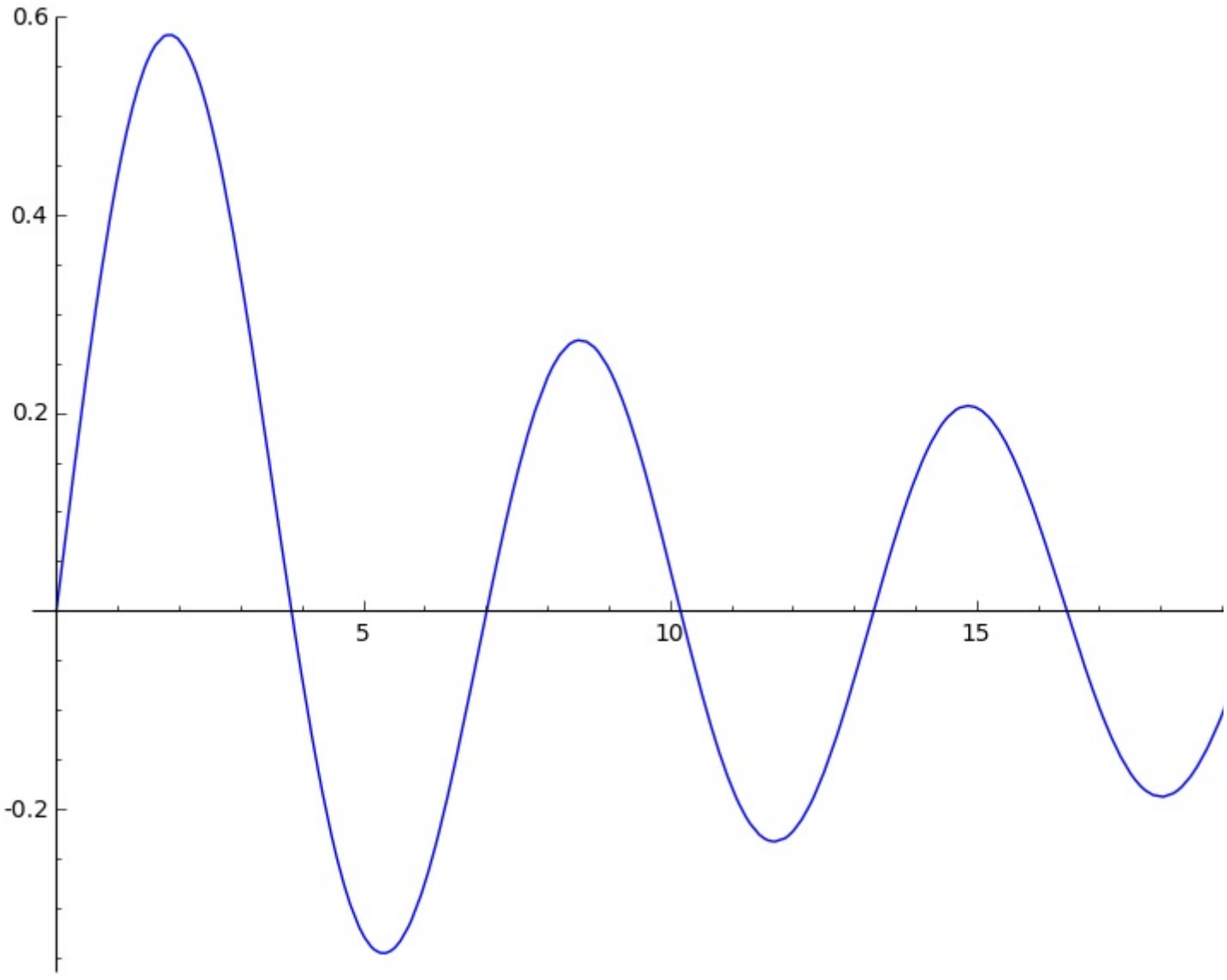
Expressions symboliques

```
>>>  x, y = var('x', 'y')
     u = cos(x)*sin(y)
     u

     cos(x)*sin(y)
```

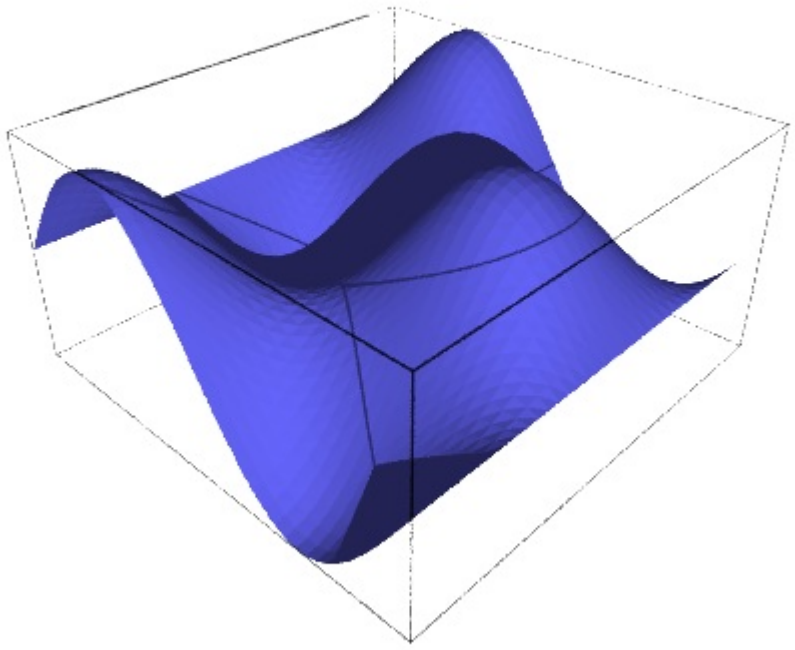
```
>>>   diff(u, x)

      -sin(x)*sin(y)

>>>   u.derivative(y)

      cos(x)*cos(y)

>>>   u.series(x, order=5)

      (sin(y)) + (-1/2*sin(y))*x^2 + (1/24*sin(y))*x^4 + Order(x^5)

>>>   u(x=1)

      cos(1)*sin(y)

>>>   u(x=sqrt(2), y=1).n(prec=1000)

      0.13122209440880167354179466394943557258151479997881813049404739139275005288731829814184371149465604288553757327376174379326766605682621211798986525637145053119092297681960434525400692467056441133781847596004378690230103677152626720456712901449325329568055173470137249434273963974796978559495221133134 5

>>>
```

Graphiques

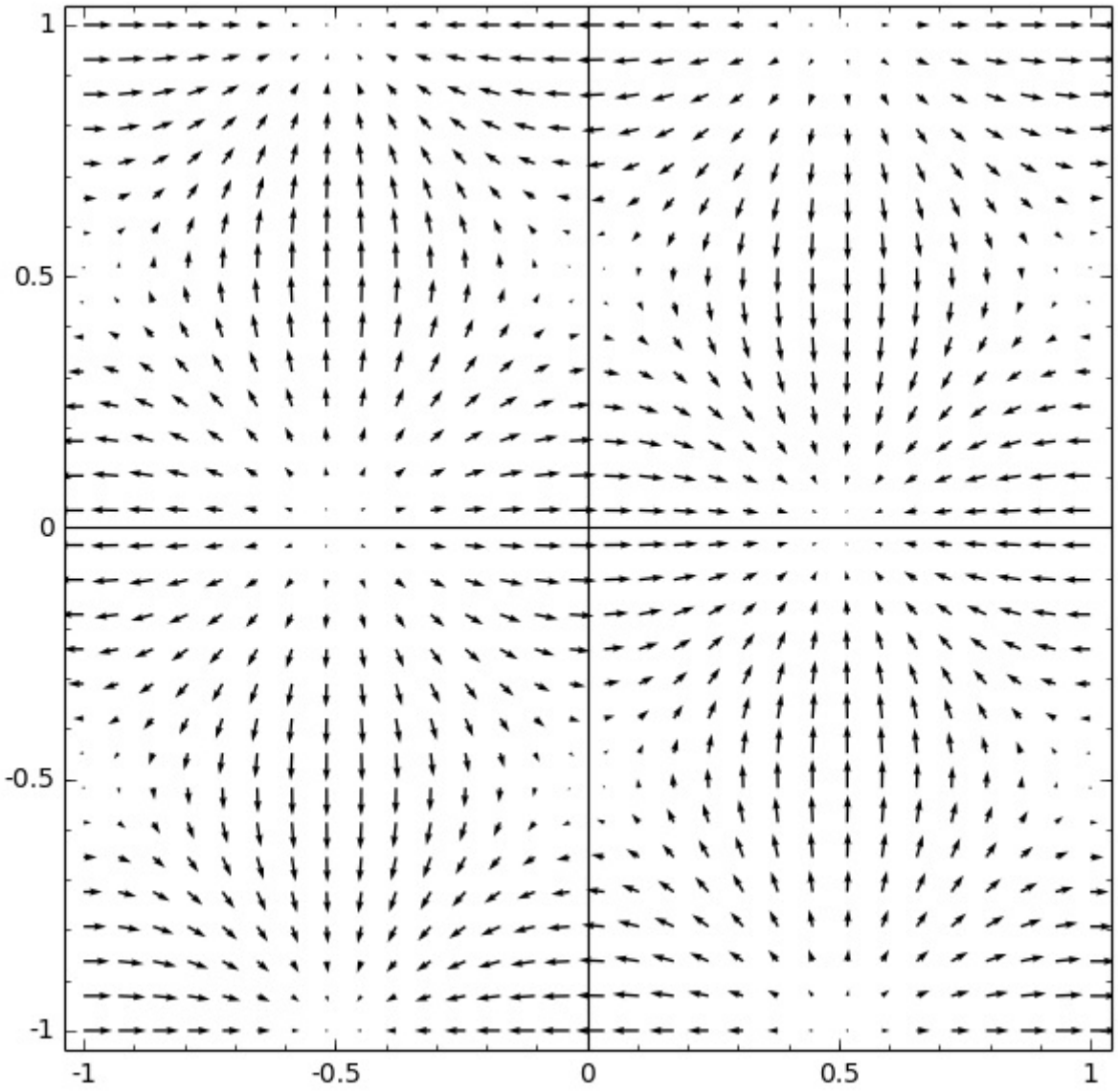
```
>>>   plot(Bessel(1,'J'), 0, 20)
```



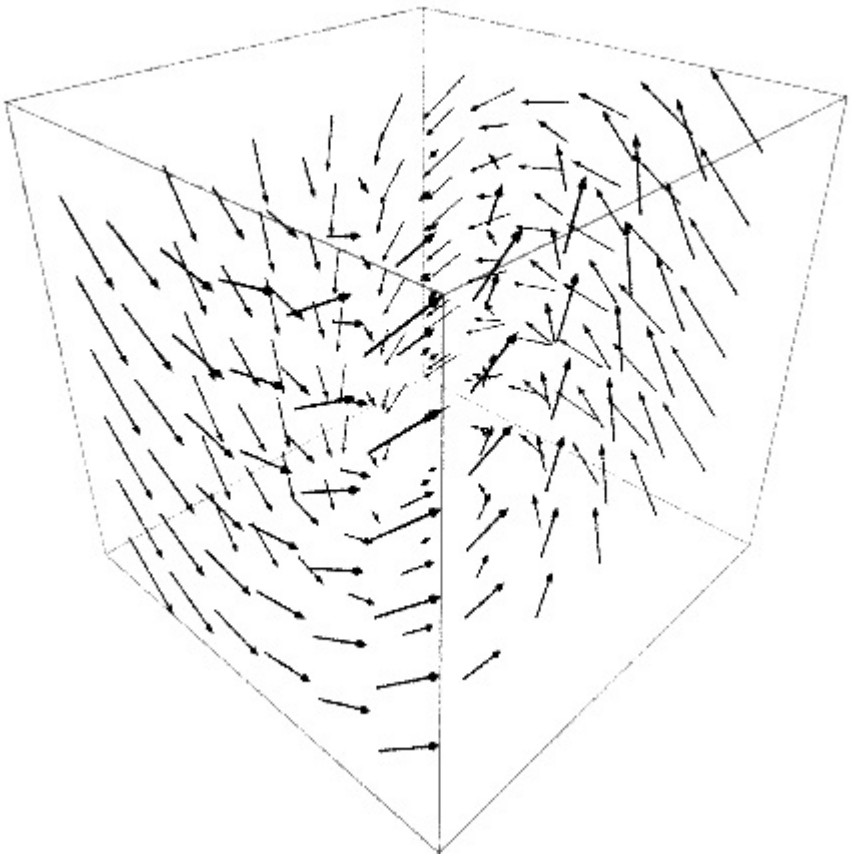
```
>>>   u, v = var('u', 'v')
      f = sin(pi*u) * cos(pi*v)
      plot3d(f, (u, -1, 1), (v, -1, 1))
```



```
>>>   plot_vector_field(f.gradient(), (u,-1,1), (v,-1,1), aspect_ratio=1,
      plot_points=30)
```



```
>>>   x,y,z=var('x y z')
      myplot = plot_vector_field3d(
      (-y, 0, x), (x,-2,2), (y,-2,2), (z,-2,2),
      colors='black', plot_points=6)
      show(myplot, viewer="tachyon")
```



```
>>>
```

Éléments et parents

```
>>>   type(1)
      <type 'sage.rings.integer.Integer'>
>>>   parent(1)
      Integer Ring
>>>   ZZ
      Integer Ring
>>>   parent(1) is ZZ
      True
>>>   parent(1/1)
      Rational Field
>>>   type(ZZ)
      <type 'sage.rings.integer_ring.IntegerRing_class'>
>>>   type(QQ)
      <class 'sage.rings.rational_field.RationalField_with_category'>
>>>   ZZ.cardinality()
      +Infinity
```

```
>>>   MyParent = QQ.cartesian_product(ZZ); MyParent

      The cartesian product of (Rational Field, Integer Ring)

>>>   MyParent.is_ring()

      True

>>>   MyParent.an_element()

      (1/2, 1)

>>>   QQ.category()

      Category of quotient fields

>>>
```

Quelques parents

```
>>>   Integers()

      Integer Ring

>>>   Rationals()

      Rational Field

>>>   R = IntegerModRing(10); R

      Ring of integers modulo 10

>>>   R(8)^2

      4

>>>   Reals()

      Real Field with 53 bits of precision

>>>   Complexes()

      Complex Field with 53 bits of precision

>>>   MatrixSpace(RDF, 2, 3)

      Full MatrixSpace of 2 by 3 dense matrices over Real Double Field

>>>   PolynomialRing(QQ, 'x')

      Univariate Polynomial Ring in x over Rational Field

>>>   PolynomialRing(QQ, 'x, y, z')

      Multivariate Polynomial Ring in x, y, z over Rational Field

>>>   MatrixSpace(PolynomialRing(ZZ, 'x'), 2)

      Full MatrixSpace of 2 by 2 dense matrices over Univariate Polynomial Ring in x over Integer Ring

>>>   _.random_element()

      [ 2*x^2 + 37*x 12*x^2 + x - 1]
      [ -x^2 + x -7*x^2 - 13*x]

>>>

>>>
```

Conversions

```
>>>   RDF

      Real Double Field

>>>   RDF(42)

      42.0
```

```
>>>   RDF(42).parent()

      Real Double Field

>>>   ZZ(1.0)

      1

>>>   ZZ(1.5)

      ---------------------------------------------------------------------------
      TypeError Traceback (most recent call last)
      <ipython-input-69-562488d24cf3> in <module>()
      ----> 1 ZZ(RealNumber('1.5'))

      /home/marc/co/sage/local/lib/python2.7/site-packages/sage/structure/parent.so in
      sage.structure.parent.Parent.__call__ (build/cythonized/sage/structure/parent.c:9603)()

      /home/marc/co/sage/local/lib/python2.7/site-packages/sage/structure/coerce_maps.so in
      sage.structure.coerce_maps.NamedConvertMap._call_
      (build/cythonized/sage/structure/coerce_maps.c:5577)()

      /home/marc/co/sage/local/lib/python2.7/site-packages/sage/rings/real_mpfr.so in
      sage.rings.real_mpfr.RealNumber._integer_ (build/cythonized/sage/rings/real_mpfr.c:15923)()

      TypeError: Attempt to coerce non-integral RealNumber to Integer

>>>
```

Coercitions

(= conversions canoniques automatiques)

```
>>>   a = 42
      a, a.parent()

      (42, Integer Ring)

>>>   b = a + 1/2
      b, b.parent()

      (85/2, Rational Field)

>>>   c = b + 1/2
      (c, c.parent())

      (43, Rational Field)

>>>   d = ZZ(c)
      (d, d.parent())

      (43, Integer Ring)

>>>   M = MatrixSpace(ZZ, 3); M

      Full MatrixSpace of 3 by 3 dense matrices over Integer Ring

>>>   obj = M.identity_matrix() + 1/2

>>>   obj

      [3/2 0 0]
      [ 0 3/2 0]
      [ 0 0 3/2]

>>>   obj.parent()

      Full MatrixSpace of 3 by 3 dense matrices over Rational Field

>>>
```