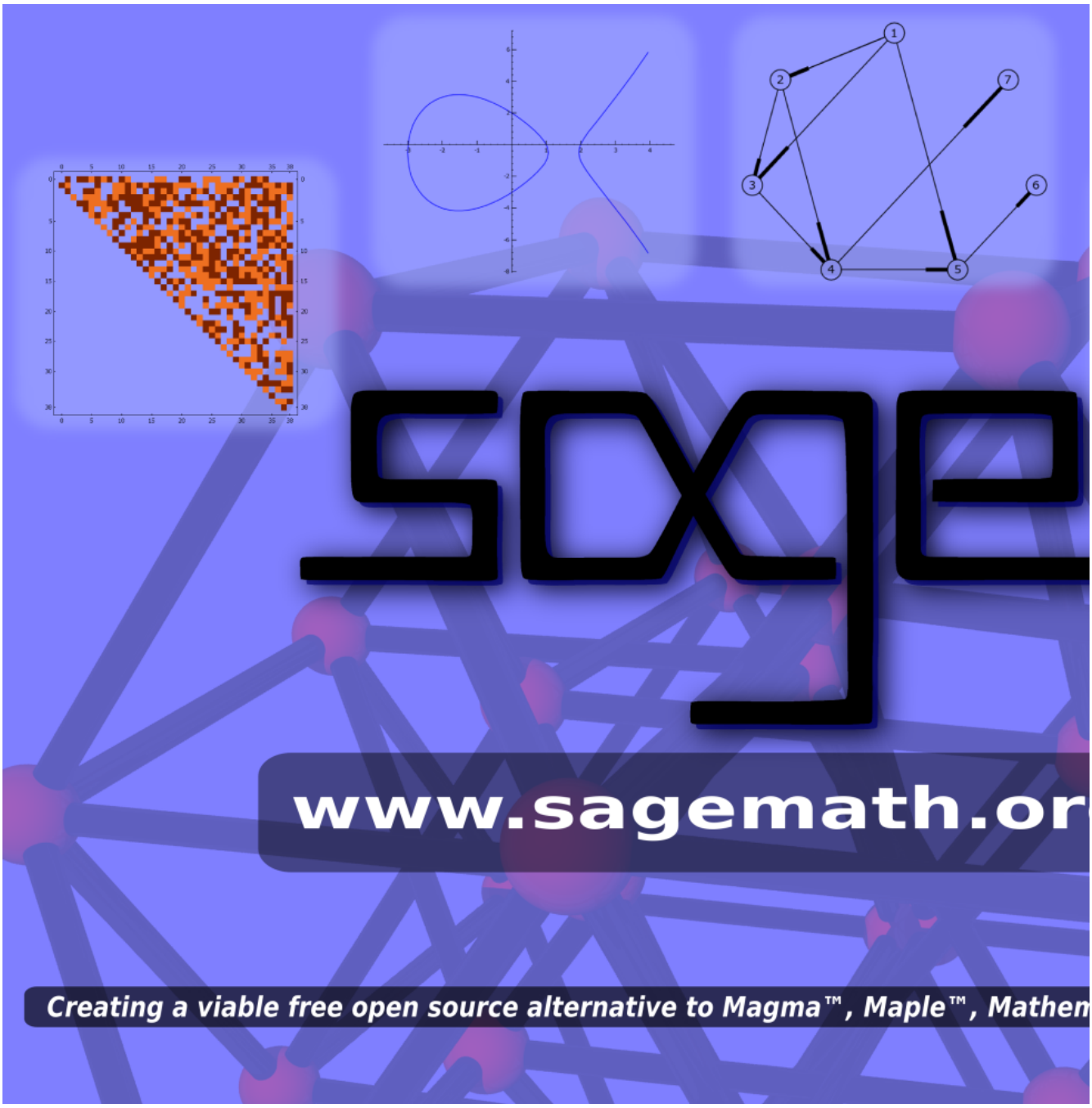
Introducción a SAGE V2



Sage es software libre

- Es gratis para el profesor (licencia académica de Matlab: 500€).
- Es gratis para los alumnos mientras estudian (licencia de estudiante de Matlab: 90).
- Es gratis para los alumnos cuando dejan de estudiar (licencia comercial de Matlab: 2000€, toolboxes aparte).
- Siempre será **gratis** (al contrario que MUPAD, por ejemplo).

- Puedes usarlo de cualquier forma (PC, cliente-servidor, supercomputadoras)
- Es posible **revisar el código**. Si por ejemplo, comparamos la velocidad de un algoritmo frente a otro distinto, toda la información está a la vista de potenciales referees.
- Los **errores** se discuten públicamente y se corrigen lo antes posible. La versión corregida está disponible para todo el mundo tan pronto como esté disponible (para nosotros fueron 24 horas).
- Podemos **adaptar Sage a nuestras necesidades** (ej: traducciones, live dvd con Sage, FEMhub)
- Podemos desarrollar nuestro código dentro de Sage, usando todas sus capacidades (ej: sage combinat)

¿Quién hace Sage?

- William Stein comienza el proyecto en 2005 (http://sagemath.blogspot.com/2009/12/mathematical-software-and-me-very.html).
- En 2006, se realiza el primer encuentro Sage Days, y se distribuye la primera versión de Sage.
- En 2007, Sage gana el primer premio de los Trophees du Libre.
- Desarrollado por cualquiera que quiera colaborar, en cualquier parte del mundo: profesores de matemáticas, estudiantes, ingenieros...
- **Financiación** por parte de universidades y empresas, becas para trabajar en Sage...

El proceso de desarrollo

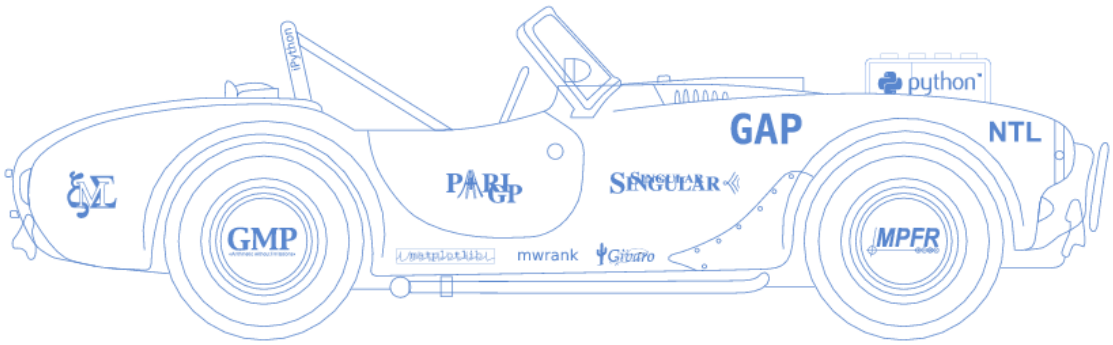
- Todo el código (desde 2007) debe ser **revisado** por otra persona distina del autor.
- El código nuevo debe estar cubierto por **tests automáticos**. De esta forma se evita que un cambio en una parte del programa estropee otra parte distinta.
- La documentación también se comprueba automáticamente, para asegurar que los ejemplos sigan funcionando.
- Todo esto es muy importante, porque Sage es joven, y necesita cambiar si tiene que ser útil a una comunidad heterogénea.

¿Dónde está la comunidad de desarrolladores?

- **Sage days**: reuniones periódicas donde se habla de Sage en general, y a menudo se busca algún propósito específico.
- Listas de correo:
  - **sage-devel**: http://groups.google.com/group/sage-devel (1220 usuarios el 09/07/2010)
  - **sage-support**: http://groups.google.com/group/sage-support (1792 usuarios el 09/07/2010)
  - **sage-edu, sage-notebook, sage-algebra, ...**
- IRC: `#sage-devel` en freenode
- Rastreador de errores: http://trac.sagemath.org/sage_trac/

¿Cómo es Sage por dentro?

Sage usa el lenguaje **python** para integrar librerías de matemáticas existentes en un entorno unificado. También se escribe *código nuevo* en python, y en **cython** cuando se necesita más velocidad.



»Every free computer algebra system I've tried has reinvented many times the wheel without being able to build the car.«

Paquetes de Matemáticas incluídos en SAGE

Álgebra	GAP, Maxima, Singular
Geometría Algebraica	Singular
Aritmética de Precision Arbitraria	GMP, MPFR, MPFI, NTL
Geometría Aritmética	PARI/GP, NTL, mwrank, ecm
Cálculo	Maxima, SymPy, GiNaC
Combinatoria	Symmetrica, Sage-Combinat
Álgebra Linear	ATLAS, BLAS, LAPACK, NumPy, LinBox, IML, GSL
Teoría de Grafos	NetworkX
Teoría de Grupos	GAP
Cálculo Numérico	GSL, SciPy, NumPy, ATLAS
Teoría de Números	PARI/GP, FLINT, NTL
Estadística	R, SciPy

Otros paquetes incluídos en SAGE

Línea de Comandos	IPython
Bases de datos	ZODB, Python Pickles, SQLite
Interfaces gráficas	Sage Notebook, jsmath, jQuery
Gráficos	Matplotlib, Tachyon3d, GD, Jmol
Lenguaje de programación	Python, Cython
Networking	Twisted
Etcétera	Paquetes estándar

Usa python en vez de inventar su propio lenguaje

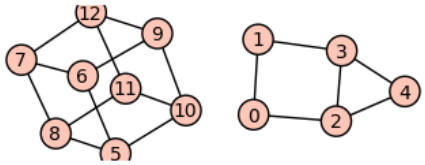
Usar un lenguaje popular y sofisticado tiene muchas ventajas:

- Programación orientada a objetos, especificación de listas por comprensión, generadores, excepciones...
- Python también es Open Source: la comunidad propone y discute la evolución del lenguaje. Esto garantiza que el lenguaje mejora sin dejar de ser útil para tareas de todo tipo.

- **Enorme cantidad de librerías** implementando servidores web, conexión con bases de datos, compiladores...
- Muchos paquetes científicos escriben interfaces con python independientemente de Sage.
- Estamos enseñando a los alumnos un lenguaje que podrán usar en un trabajo.

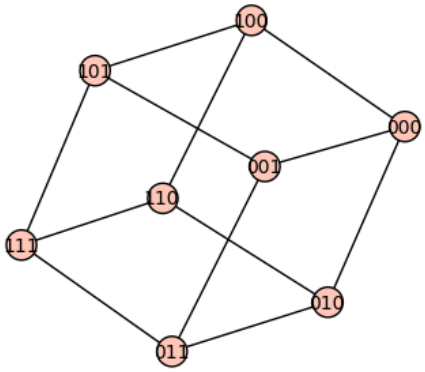
```
#La indentación marca el principio y el final de los bloques
n = 17
while n != 1:
    print n,
    if n%2 == 0:
        n /= 2
    else:
        n = 3*n + 1
```

```
#python es dinámico: no necesitamos saber el tipo de datos
#al escribir el código
def suma(a,b):
    return a + b

print suma(1,1)
var('x')
show( suma(sin(x), e^x) )
show( suma(graphs.HouseGraph(), graphs.CubeGraph(3)) )
```




```
#python es orientado a objetos
#Cada dato tiene un tipo de datos, con sus propios métodos

g = graphs.CubeGraph(3)
print type(g)
show(g,layout='spring')

M = g.adjacency_matrix()
print type(M)
show(M)

p = M.characteristic_polynomial()
print type(p)
show(p)
```

```
<class 'sage.graphs.graph.Graph'>
```




```
<type 'sage.matrix.matrix_integer_dense.Matrix_integer_dense'>
```

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

```
<type
'sage.rings.polynomial.polynomial_integer_dense_flint.Polynomial_int\
eger_dense_flint'>
```

$$x^8 - 12x^6 + 30x^4 - 28x^2 + 9$$

```
#list comprehension

#todos los primos menores que 100 que son de la forma z^2+1
print [p for p in prime_range(100) if (p-1).is_square()]

#el numero de lados de cada CubeGraph para k<10
#(número de aristas de un cubo k-dimensional)
print [graphs.CubeGraph(k).num_edges() for k in range(2,10) ]
```

```
[2, 5, 17, 37]
[4, 12, 32, 80, 192, 448, 1024, 2304]
```

```
#ejemplo con "generator expression"
#suma de los inversos de los primos menores que 100
#nota: es un numero racional
sum(1/p for p in prime_range(100) )
```

```
#hay algun numero primo de la forma 2k+1 con k entre A y B?
A = 1000
B = 2000
any(is_prime(2*k+1) for k in srange(A, B))

#Al usar un generador, no se emplea tiempo en comprobar si todos
#los números entre A y B son primos.
#La ejecución se detiene cuando se encuentra el primer número primo
```

```
#Jerarquía de errores
A = matrix(QQ,[[1,1,0],[2,2,0]])
v = vector(QQ,[1,0])
A.solve_right(v)
```

```
A = matrix(QQ,[[2,0],[1,3]])
v = vector(CDF,[i,0])
A.solve_right(v)
```

```
A = matrix(QQ,[[2,0],[1,3]])
v = vector(CDF,[1,0])
try:
    print A.solve_right(v)
except TypeError:
    print A.base_extend(CDF).solve_right(v)
```

```
A = matrix(QQ,[[1,1,0],[2,2,0]])
v = vector(QQ,[1,0])
try:
    print A.solve_right(v)
except TypeError:
    print A.base_extend(CDF).solve_right(v)
```

Documentación en varios formatos útiles

- El tabulador [Tab] sugiere formas de completar un comando.
- Las sugerencias se limitan a las variables miembro y métodos propios de cada variable o tipo de datos
- Escribe el comando seguido de una interrogación para ver la ayuda (**`comando?`**)
- Dos interrogaciones para ver el código fuente (**`comando??`**)
- El link de ayuda ("Help") en lo alto del navegador muestra ayuda compilada a html y pdf (requiere ejecutar `sage --docbuild all pdf`)

```

```

```
VectorSpace?
```

```
is_power_of_two??
```

```

```

La separación de cliente y servidor ofrece muchas posibilidades

- Una sala de ordenadores anticuados puede volver a la vida con un servidor potente.
- Acceso a capacidad de cómputo desde cualquier lugar (incluso desde tu teléfono móvil).
- Una forma cómoda de mostrar tu trabajo, a tus colegas o al mundo entero.
- Instalar Sage en un "supercomputador", y ejecutar código remotamente es tan cómodo como hacerlo en tu propio ordenador.

```

```

```

```

Crea interacciones en el navegador que se pueden usar sin saber programar

```
#No es necesario entender este código para usar el resultado

@interact
def mi_taylor(f = sin(x),
              interval = range_slider(0,10,default=(0,1)),
              a = (0,10),
```

```
             orden = slider(0,10,1,default = 1)
             ):
    f_taylor = f.taylor(x,a,orden)
    x0, x1 = interval
    x0 = min(x0,a)
    x1 = max(x1,a)
    show(f)
    show(f_taylor)
    (plot(f,x,x0,x1) +
     plot(f_taylor,x,x0,x1,color='red') +
     point2d((a,f(x=a)), pointsize= 30, color='green')
    ).show()
```

```
#Otro ejemplo, usando el nuevo editor de grafos interactivo

g = graphs.HouseGraph()
graph_editor(g);
```

```
#Siguen cálculos relacionados con el número cromático y el polinomio cromático

numero_cromatico = g.chromatic_number()
K = numero_cromatico + 2
print 'Número cromático: %d'%numero_cromatico
#H = g.coloring(hex_colors=True, algorithm="MILP")
H = g.coloring(hex_colors=True)
p = g.chromatic_polynomial()
g.plot(vertex_colors=H).show()
vs = [p(x=j) for j in range(K)]
print 'Polinomio cromático'
show(p)
(point([(j,p(x=j)) for j in range(K+1)], pointsize=30, color='red') +
 plot(p,0,K)).show()
for j in range(K+1):
    print j,':',p(x=j)
```

```

```

```

```

```

```

Una forma rápida de instalar mucho software científico

Instalando Sage instalas octave, maxima, singular, GAP ...

sage -maxima

```
;;; Loading #P"/home/moriarty/sage/local/lib/ecl/defsystem.fas"
;;; Loading #P"/home/moriarty/sage/local/lib/ecl/cmp.fas"
;;; Loading #P"/home/moriarty/sage/local/lib/ecl/sysfun.lsp"
Maxima 5.20.1 http://maxima.sourceforge.net
using Lisp ECL 10.2.1
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1)
```

sage -R

```
R version 2.10.1 (2009-12-14)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
```

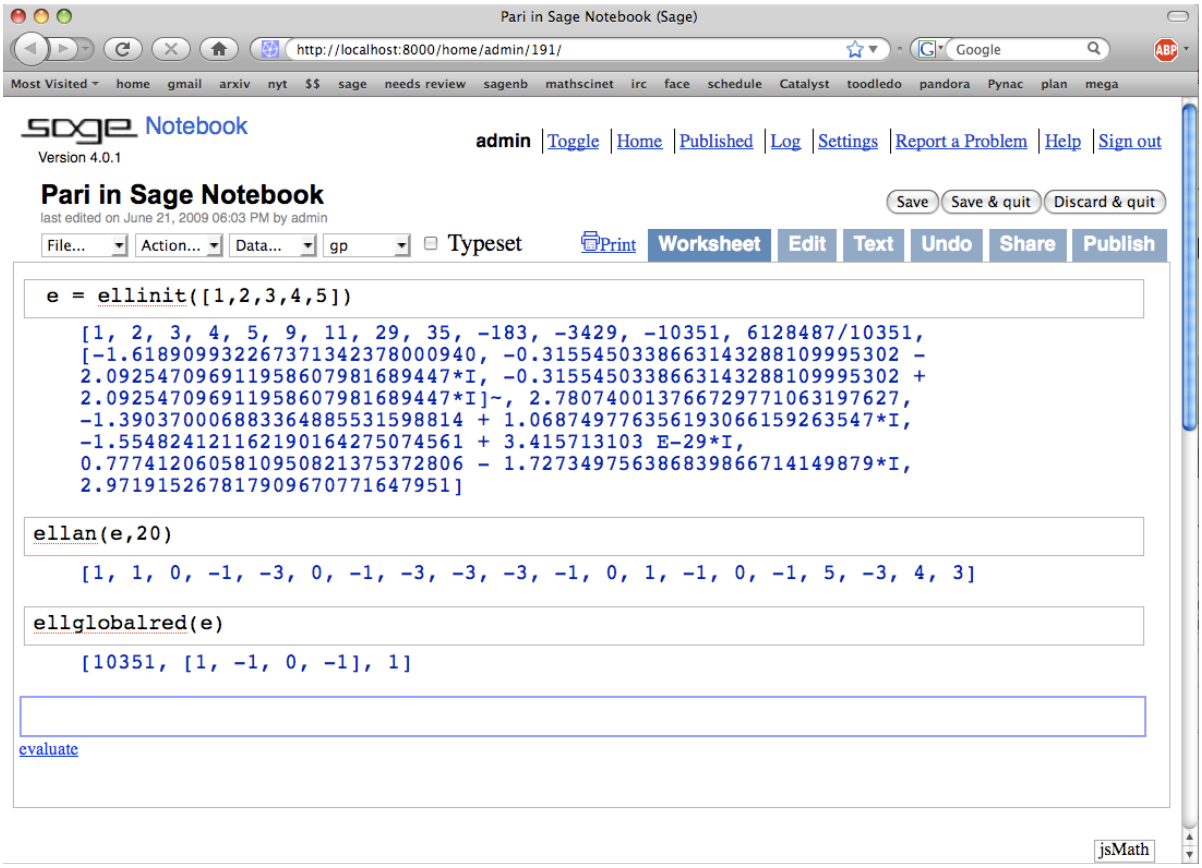
```
R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

>
```

etcetera...

Interacción via web con otros programas

Puedes otros programas desde el notebook.

PARI+GP seleccionando gp en el desplegable de lenguaje:

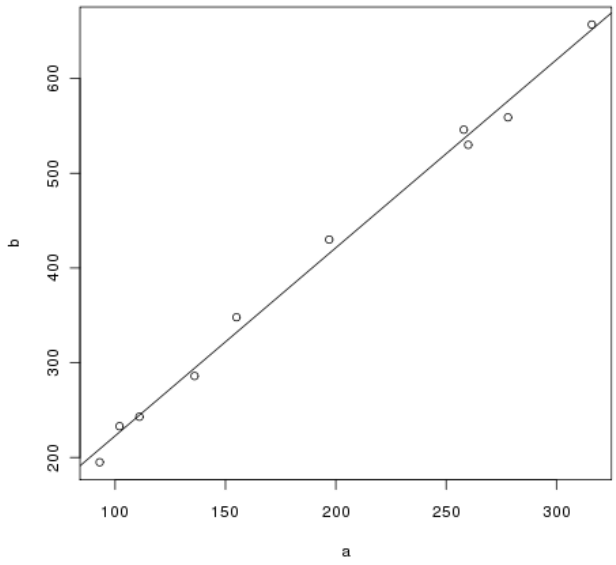



```
%r
#Escribiendo %r al principio del cuadro de comandos,
#el código que sigue es de R, no de sage

a <- c(260, 316, 111, 93, 136, 278, 258, 155, 197, 102)
b <- c(530, 657, 243, 195, 286, 559, 546, 348, 430, 233)
stem(a)

#Para guardar las imagenes en el notebook, usamos los
```

```
#comandos de R necesarios para guardar la imagen
png('rplot.png')
plot(a,b)
abline(lm(b ~ a))
dev.off()
```

```
The decimal point is 2 digit(s) to the right of the |

0 | 9
1 | 014
1 | 6
2 | 0
2 | 668
3 | 2
```

```
null device
          1
```



Combinar software muy diverso de forma armoniosa

- Por supuesto, lo más interesante es llamarlos a todo este software usando python, y no tener que aprender lenguaje distintos.
- Este objetivo todavía no se ha terminado. Algunas librerías están muy bien integradas (Singular, PARI...), otras no del todo (scipy, matplotlib...), otras más bien no (R, octave...)

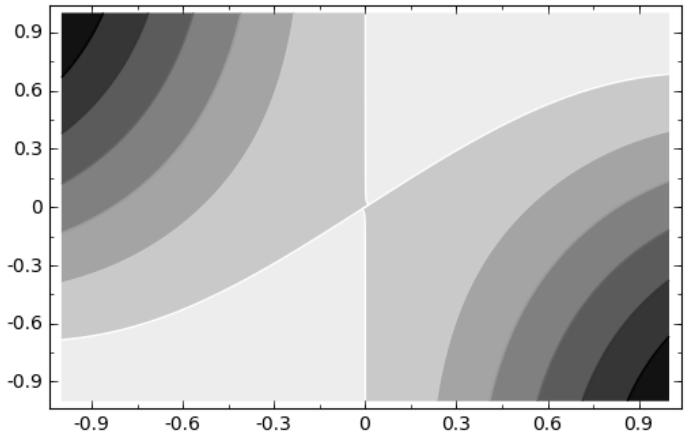
```
var('x y')
f = sin(y)*x + e^(-x^2)

#Usa maxima
assume(x != 0)
a = f.integrate(y,0,x^2).integrate(x,0,1)
show(a)
```

$$\frac{1}{4}\left(\sqrt{\pi}e\mathrm{erf}\left(1\right)-2\,e\sin\left(1\right)+2\,e-2\right)e^{\left(-1\right)}$$

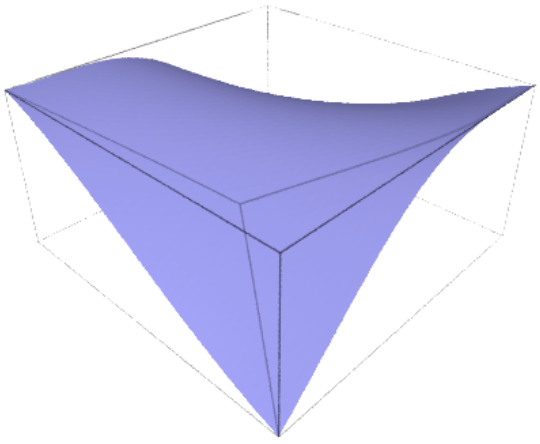
```
#Usa PARI
n(a)
```

```
0.268736853416544
```

```
#Usa matplotlib
show( contour_plot(f,(x,-1,1),(y,-1,1)) )
```




```
#Usa tachyon
show( plot3d(f,(x,-1,1),(y,-1,1)), viewer='tachyon' )
```



```
#scipy no está completamente integrado
#antes de poder usar alguna funcionalidad muy interesante
#hay que importar scipy

from scipy.integrate import dblquad
dblquad(lambda y0,x0: sin(y0)*x0 + e^(-x0^2), 0, 1, lambda x0:0, lambda x0: x0^2 )
```

```
(0.26873685341654413, 2.9835784222836294e-15)
```

```

```

Mejora tu código sin apenas cambiar de lenguaje

- python es un lenguaje dinámico, y eso es muy cómodo, pero el precio a pagar es la eficiencia

- **cython** permite compilar código con sintaxis idéntica a la de python
- Para acelerar los cálculos, es necesario declarar los tipos de las variables
- La optimización se hace de manera progresiva, no tienes que reescribir tu código para hacerlo más eficiente: "first make it right, then make it fast"

```
def mysum(N):
    s = int(0)
    for k in range(1,N):
        s += k
    return s
```

```
time mysum(10^7)
```

```
49999995000000
Time: CPU 1.31 s, Wall: 1.32 s
```

```
%cython
def mysum_cython1(N):
    s = 0
    for k in range(N):
        s += k
    return s
```

__home_mor...10_code_sage4_spyx.c __home_mor...code_sage4_spyx.html

```
time mysum_cython1(10^7)
```

```
49999995000000
Time: CPU 0.96 s, Wall: 0.96 s
```

```
%cython
def mysum_cython2(N):
    cdef int k
    cdef long long s = 0
    for k in range(N):
        s += k
    return s
```

__home_mor...10_code_sage7_spyx.c __home_mor...code_sage7_spyx.html

```
time mysum_cython2(10^7)
```

```
49999995000000L
Time: CPU 0.01 s, Wall: 0.02 s
```

Aprovecha la infraestructura para desarrollar tu propio código

- Herramientas de empaquetado, control de versiones, rastreador de errores, documentación...
- Publicidad, y más potenciales usuarios, al integrarse en un sistema que pueden usar para otras cosas
- python es un lenguaje muy claro, y tienes cython si necesitas eficiencia

Por ejemplo, **sage-combinat** (antes mupad-combinat):

```
P = Partitions(8)
print P.random_element()
print P.cardinality()
print list(P)
```

```
[6, 1, 1]
22
[[8], [7, 1], [6, 2], [6, 1, 1], [5, 3], [5, 2, 1], [5, 1, 1, 1],
[4, 4], [4, 3, 1], [4, 2, 2], [4, 2, 1, 1], [4, 1, 1, 1, 1], [3, 3,
2], [3, 3, 1, 1], [3, 2, 2, 1], [3, 2, 1, 1, 1], [3, 1, 1, 1, 1, 1],
[2, 2, 2, 2], [2, 2, 2, 1, 1], [2, 2, 1, 1, 1, 1], [2, 1, 1, 1, 1,
1, 1], [1, 1, 1, 1, 1, 1, 1, 1]]
```

```
list(Subsets(range(8),2))
```

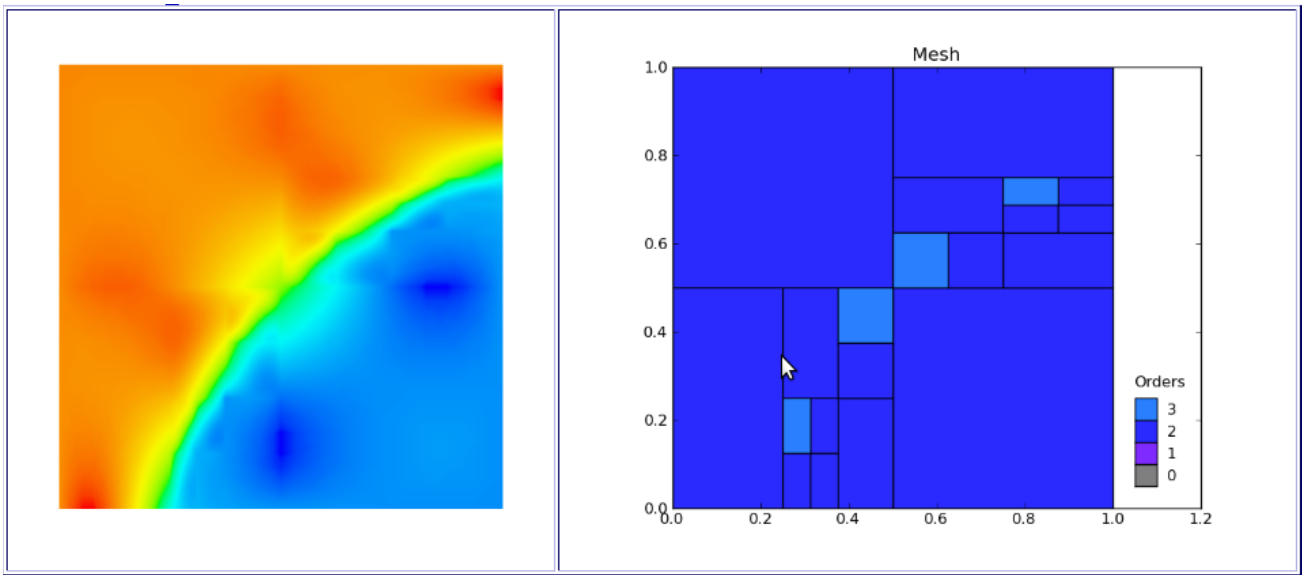
```
[{0, 1}, {0, 2}, {0, 3}, {0, 4}, {0, 5}, {0, 6}, {0, 7}, {1, 2}, {1,
3}, {1, 4}, {1, 5}, {1, 6}, {1, 7}, {2, 3}, {2, 4}, {2, 5}, {2, 6},
{2, 7}, {3, 4}, {3, 5}, {3, 6}, {3, 7}, {4, 5}, {4, 6}, {4, 7}, {5,
6}, {5, 7}, {6, 7}]
```

```
list(CartesianProduct(range(4),Combinations(range(5),2)))
```

```
    [[0, [0, 1]], [0, [0, 2]], [0, [0, 3]], [0, [0, 4]], [0, [1, 2]],
    [0, [1, 3]], [0, [1, 4]], [0, [2, 3]], [0, [2, 4]], [0, [3, 4]], [1,
    [0, 1]], [1, [0, 2]], [1, [0, 3]], [1, [0, 4]], [1, [1, 2]], [1, [1,
    3]], [1, [1, 4]], [1, [2, 3]], [1, [2, 4]], [1, [3, 4]], [2, [0,
    1]], [2, [0, 2]], [2, [0, 3]], [2, [0, 4]], [2, [1, 2]], [2, [1,
    3]], [2, [1, 4]], [2, [2, 3]], [2, [2, 4]], [2, [3, 4]], [3, [0,
    1]], [3, [0, 2]], [3, [0, 3]], [3, [0, 4]], [3, [1, 2]], [3, [1,
    3]], [3, [1, 4]], [3, [2, 3]], [3, [2, 4]], [3, [3, 4]]]
```

Modificar Sage

Si Sage no es exactamente lo que necesitas, siempre puedes adaptarlo a tus necesidades: **FEMhub** usa python y el notebook de Sage para compartir y comparar código de elementos finitos.



Ejemplos

Seguimos con ejemplos hasta que nos echen. Por favor interrumpid con comentarios, preguntas, sugerencias...

Animación de un haz de cónicas

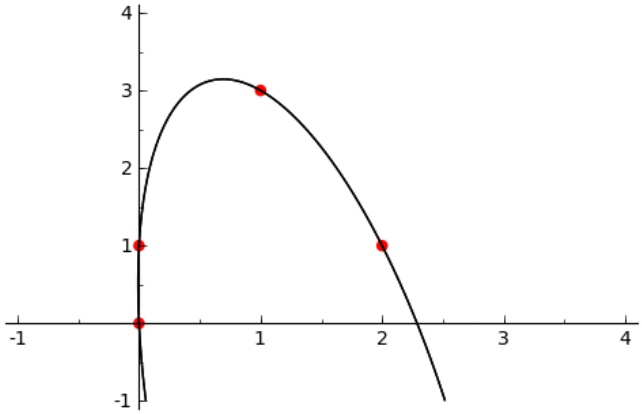
Calculamos el haz de cónicas que pasa por 4 puntos y lo animamos

```
puntos = [(0,0),(0,1),(1,3),(2,1)]
K = len(puntos)

var('x y')
coefs = matrix(QQ, K, 6)
for j in range(K):
    x0, y0 = puntos[j]
    coefs[j,0] = x0^2    #Primera columna (coef. de a)
    coefs[j,1] = y0^2    #Segunda columna (coef. de b)
    coefs[j,2] = x0*y0   #Tercera columna (coef. de c)
    coefs[j,3] = x0      #Cuarta columna (coef. de d)
    coefs[j,4] = y0      #Quinta columna (coef. de e)
    coefs[j,5] = 1       #Sexta columna (coef. de f)

K = coefs.right_kernel()
v1 = K.basis()[0]
v2 = K.basis()[1]
show(K.basis())

graficas = []
for t in srange(0,2*pi,0.3):
    c1, c2 = sin(t), cos(t)
    a,b,c,d,e,f = c1*v1 + c2*v2
    curva = a*x^2 + b*y^2 + c*x*y + d*x + e*y + f
    graficas.append( point2d(puntos,color=(1,0,0),pointsize=30) + implicit_plot(curva,(x,-1,4),
(y,-1,4)))
a = animate(graficas)
```

$$\left[\left(1, 0, \frac{1}{2}, -\frac{5}{2}, 0, 0\right), (0, 1, -3, 3, -1, 0)\right]$$

```
a.show(delay=10)
```

```
CPU time: 0.02 s,  Wall time: 16.77 s
```



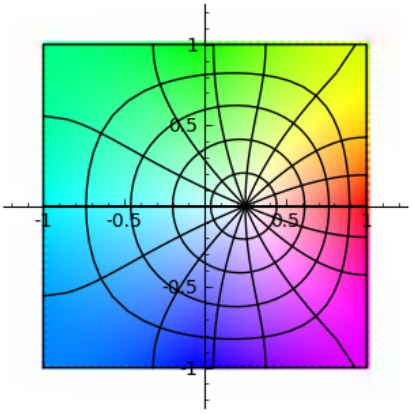

```
len(graficas)
```

```
21
```

Teorema de la aplicación conforme de Riemann

Recientemente se ha incluído en Sage código para calcular de forma numérica la aplicación conforme desde un dominio cualquiera al disco unidad.

```
ps = polygon_spline([(-1, -1), (1, -1), (1, 1), (-1, 1)])
f = lambda t: ps.value(t)
fprime = lambda t: ps.derivative(t)
m = Riemann_Map([f], [fprime], 0.25, ncorners=4)
m.plot_colored() + m.plot_spiderweb()
```




```
CPU time: 7.33 s,  Wall time: 8.27 s
```

Álgebra lineal exacta y numérica

```
M = random_matrix(RDF, 4)
show(M)
```

$$\begin{pmatrix} -0.291647816622 & 0.0377918507121 & 0.0917671354307 & -0.0235900476959 \\ -0.602198567982 & 0.0783619762552 & -0.264880646608 & -0.348248886156 \\ -0.389075286771 & -0.85338987633 & 0.530704256258 & -0.628525959516 \\ 0.344858365103 & -0.263796249278 & -0.658633204488 & 0.345325735213 \end{pmatrix}$$

```
M.SVD()
```

```
(
[  -0.134084916117   -0.258829327908   0.0486349662263
-0.955334106135]
[  -0.234011204077   -0.591581422429    0.736268778111
0.230604560626]
[  -0.866345327119    0.485723314869    0.116197818851
-0.00408686510948]
[   0.420363849321    0.589161829249    0.664861634728
-0.184774402468],

[ 1.36963996781            0.0            0.0            0.0]
[           0.0 0.776522557379            0.0            0.0]
[           0.0            0.0  0.67266901112            0.0]
[           0.0            0.0            0.0 0.149035802349],

[  0.483387128729   0.574266564117  -0.406575721994
0.520817606654]
[  0.441747496756  -0.806247240506  -0.319646902628
0.229456736501]
[ -0.501561218549 0.00345171004483  -0.842603897476
-0.196069124811]
[  0.565360395621   0.142026005402  -0.150135644948
-0.798533358708]
)
```

```
M.LU()
```

```
(
[0.0 1.0 0.0 0.0]  [             1.0              0.0
0.0               0.0]
[0.0 0.0 1.0 0.0]  [   0.646091351685              1.0
0.0               0.0]
[0.0 0.0 0.0 1.0]  [  -0.572665534989     0.24216424361
1.0               0.0]
[1.0 0.0 0.0 0.0], [    0.48430506502 0.000176159257535
-0.224350253858              1.0],

[-0.602198567982 0.0783619762552 -0.264880646608 -0.348248886156]
[            0.0 -0.904018871489   0.70184135126 -0.403525365936]
[            0.0             0.0 -0.980282101648  0.243615015532]
[            0.0             0.0             0.0  0.199794827064]
)
```

```
cc = [[2412, -7452, -846, 504, -6822], [-1008, 22662, 4058, -1818, 17398], [21240, 6606, 5726, 342,
8674], [5346, 80496, 10330, -594, 72902], [-1683, -9837, -3202, 1152, -5888]]
```

```
M = matrix(QQ, [[2412, -7452, -846, 504, -6822],
                [-1008, 22662, 4058, -1818, 17398],
                [21240, 6606, 5726, 342, 8674],
                [5346, 80496, 10330, -594, 72902],
                [-1683, -9837, -3202, 1152, -5888]])
```

```
M.jordan_form()
```

```
[6948    1|   0    0    0]
[   0 6948|   0    0    0]
[---------+--------------]
[   0    0|3474    1    0]
[   0    0|   0 3474    1]
[   0    0|   0    0 3474]
```

```

```

Geometría algebraica

Al principio, Sage fue desarrollado sobre todo por investigadores en Álgebra, y sus capacidades en este campo son muy extensas.

```
P.<x,y,z> = ProjectiveSpace(QQ,2)
X = P.subscheme([x*z^2, y^2*z, x*y^2]); X
```

```
Closed subscheme of Projective Space of dimension 2 over Rational
Field defined by:
  x*z^2,
  y^2*z,
  x*y^2
```

```
X.irreducible_components()
```

```
[
Closed subscheme of Projective Space of dimension 2 over Rational
Field defined by:
  z,
  y,
Closed subscheme of Projective Space of dimension 2 over Rational
Field defined by:
  z,
  x,
Closed subscheme of Projective Space of dimension 2 over Rational
Field defined by:
  y,
  x
]
```

Recursos

- Ejemplos usando interact!!: http://wiki.sagemath.org/interact
- Servidor público de Sage: http://www.sagenb.org/home/pub/
- Servidor público de FEMhub: http://lab.femhub.org/pub/
- Servidor público con ejemplos de estadística: http://sage.math.canterbury.ac.nz/pub/