# varona

```
5+4/3
```

```
19/3
```

```
(5+4)/3
```

```
3
```

```
cos(0);arctan(1);N(_)
```

```
1
1/4*pi
0.785398163397448
```

```
v=[3,4,-6]; v[2]
```

```
-6
```

```
3^50,3.0^50; factorial(40)
```

```
(717897987691852588770249, 7.17897987691853e23)
815915283247897734345611269596115894272000000000
```

```
N(sqrt(10),digits=50); sqrt(10).n(digits=50); N(sqrt(10),170)
```

```
3.1622776601683793319988935444327185337195551393252
3.1622776601683793319988935444327185337195551393252
3.1622776601683793319988935444327185337195551393252
```

```
var("alpha, x, y, z")
```

```
(alpha, x, y, z)
```

```
z=sqrt(7*x+y^5-sin(alpha)); show(z); latex(z)
```

$$\sqrt{y^5 + 7\,x - \sin\left(\alpha\right)}$$

```
\sqrt{y^{5} + 7 \, x - \sin\left(\alpha\right)}
```

```
(3+4*I)^10;e^(i*pi)
```

```
1476984*I - 9653287
-1
```

```
var('x'); p=(x+1)*(x-1)^2; q=expand(p); q; factor(q)
```

```
x
x^3 - x^2 - x + 1
(x - 1)^2*(x + 1)
```

```
find_root(q, 0, 3)
```

```
1.000000000082526
```

```
var("theta"); find_root(cos(theta)==sin(theta)+1/5,0,pi/2)
```

```
theta
0.64350110879328448
```

```
time is_prime(2^127-1);time factor(2^128-1)
```

```
True
Time: CPU 0.00 s, Wall: 0.00 s
3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 6728042131072
Time: CPU 0.01 s, Wall: 0.15 s
```

```
reset("a");reset()
```

```
f(x)=1/(1+x^2)
```

```
var("r"); [f(x),f(x+1),f(3),f(r)]
```

```
r
[1/(x^2 + 1), 1/((x + 1)^2 + 1), 1/10, 1/(r^2 + 1)]
```

```
diff(f(x));integrate(f(x),x)
```

```
-2*x/(x^2 + 1)^2
arctan(x)
```

```
var("x,y")
diff(sin(x^2),x,4);diff(x^2+17*y^2,y)
```

```
16*x^4*sin(x^2) - 48*x^2*cos(x^2) - 12*sin(x^2)
34*y
```

```
integral(x*sin(x^2),x); show(integrate(x/(1-x^3),x))
```

```
-1/2*cos(x^2)
```

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

```
integral(x/(1+x^2),x,0,1)
```

```
1/2*log(2)
```

```
integral(x*tan(x), x)
```

```
1/2*I*x^2 - 1/2*x*log(sin(2*x)^2 + cos(2*x)^2 + 2*cos(2*x) + 1) -
I*x*arctan2(sin(2*x), cos(2*x) + 1) + 1/2*I*polylog(2, -e^(2*I*x)
```

```
integral(x*tan(x), x,0,1)
```

```
-1/2*log(sin(2)^2 + cos(2)^2 + 2*cos(2) + 1) + 1/2*I*limit(-x^2 +
2*x*arctan(sin(2*x)/(cos(2*x) + 1)) - realpart(polylog(2,
-e^(2*I*x))), x, 0) - 1/2*I*limit(-x^2 +
2*x*arctan(sin(2*x)/(cos(2*x) + 1)) - realpart(polylog(2,
-e^(2*I*x))), x, 1) + 1/2*imagpart(-1/12*pi^2) -
1/2*imagpart(polylog(2, -e^(2*I)))
```

```
numerical_integral(x*tan(x), 0,1)
```

```
(0.42808830136517595, 4.7527348874829114e-15)
```

```
limit(sin(x)/abs(x), x=0)
```

```
und
```

```
limit(sin(x)/abs(x), x=0, dir="minus")
```

```
-1
```

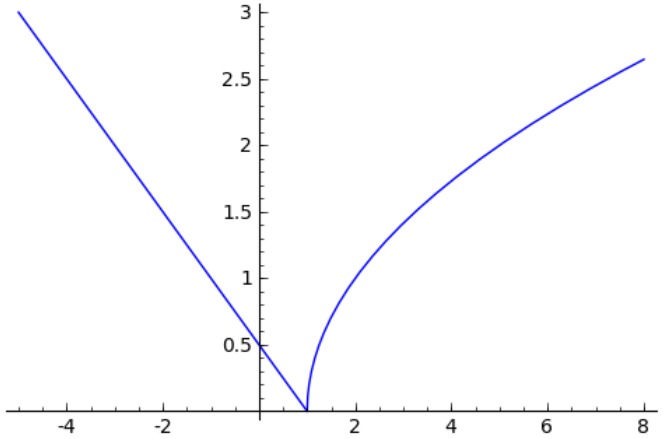
```
limit(sin(x)/abs(x), x=0, dir="plus")
```

```
1
```

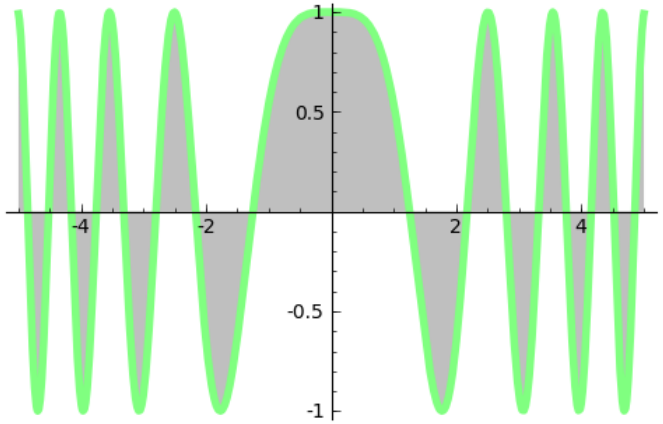
```
lim(factorial(x)*exp(x)/x^(x+1/2), x=oo)
```

```
sqrt(pi)*sqrt(2)
```

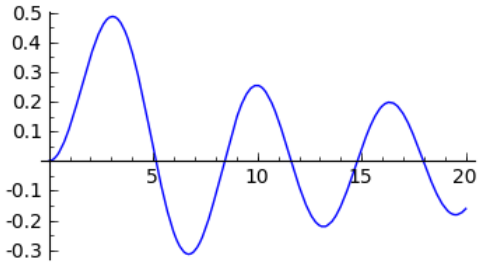
```
g = Piecewise([[(-5,1),(1-x)/2], [(1,8),sqrt(x-1)]],x);
plot(g)
```




```
plot(cos(x^2),x,-5,5,thickness=4,rgbcolor=(0.5,1,0.5),fill='axis')
```



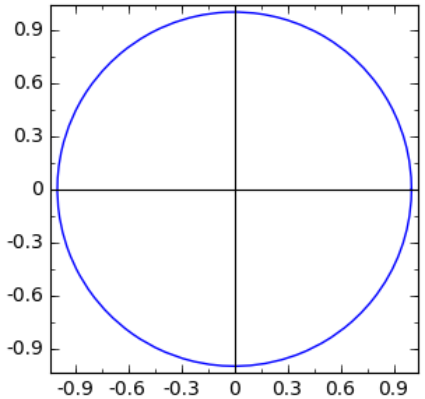

```
plot(bessel_J(2,x,"maxima"),0,20,figsize=[4,2.2])
```



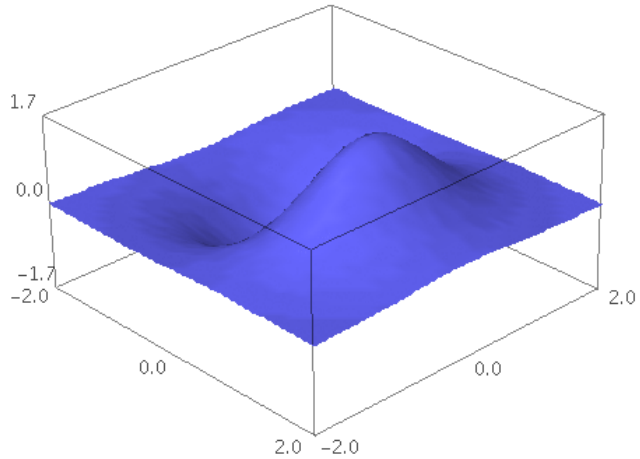

```
save(plot(sin(x)/x,-20,20),
"/Users/pepearanda/Desktop/dibujo.pdf")
```

```
automatic_names(true)
```

```
c=parametric_plot((cos(t),sin(t)), (0,2*pi))
c.show(aspect_ratio=1,frame=true,figsize=[3.4,3.4])
```

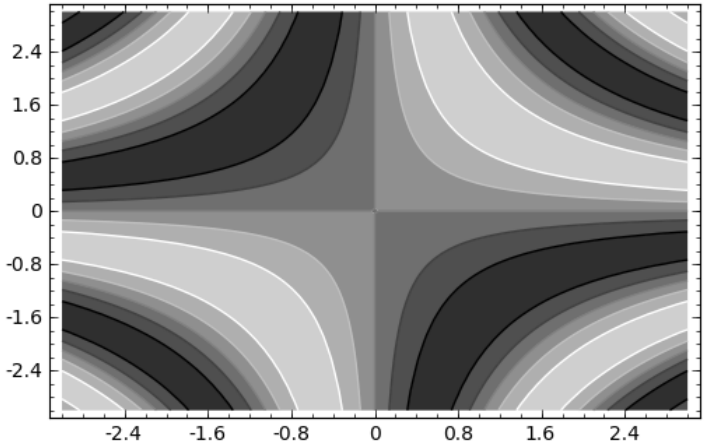



```
plot3d(4*x*exp(-x^2-y^2), (x,-2,2), (y,-2,2))
```

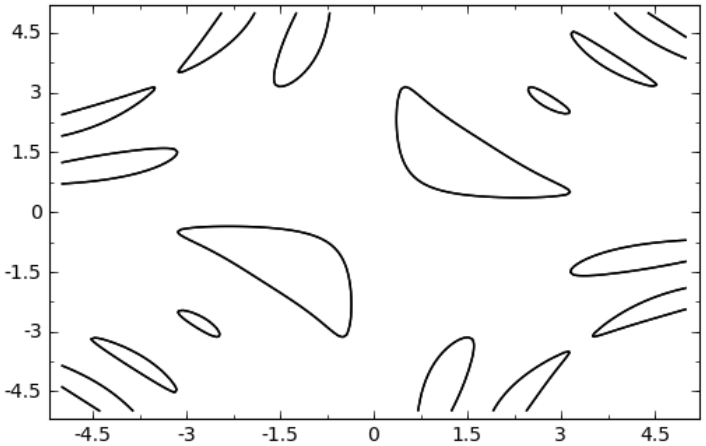



Get Image

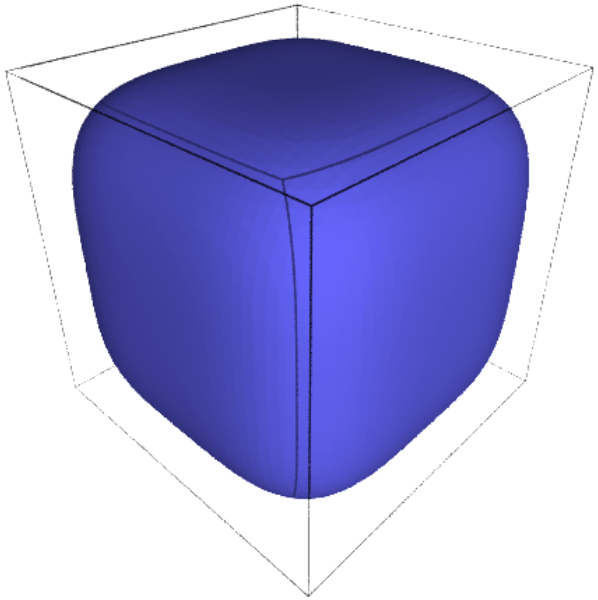
```
contour_plot(sin(x*y), (x,-3,3), (y,-3,3), contours=5,
plot_points=80)
```



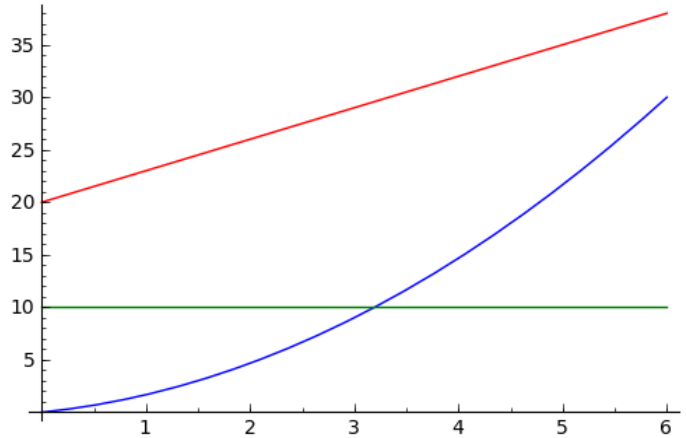

```
implicit_plot(sin(x*y) + sin(x)*sin(y) == 1, (x,-5,5), (y,-5,5))
```



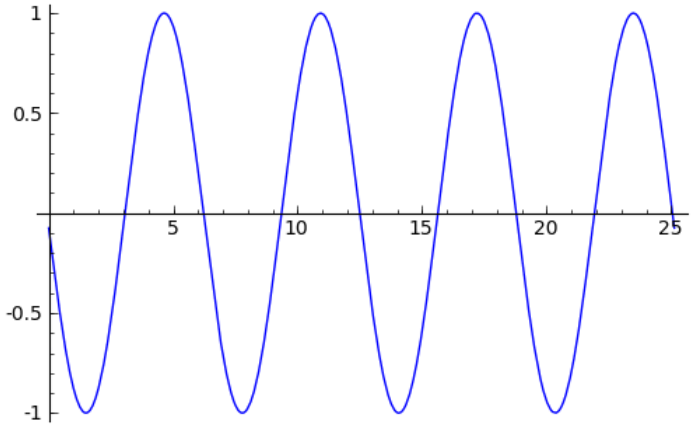

```
var('x,y,z');
implicit_plot3d(x^4+y^4+z^4==16,(x,-2,2),(y,-2,2),(z,-2,2),
viewer='tachyon')
```



```
r=plot(2*t^2/3+t,0,6);p=plot(3*t+20,0,6,rgbcolor='red')
r + p + line([(0,10),(6,10)],rgbcolor='green')
```

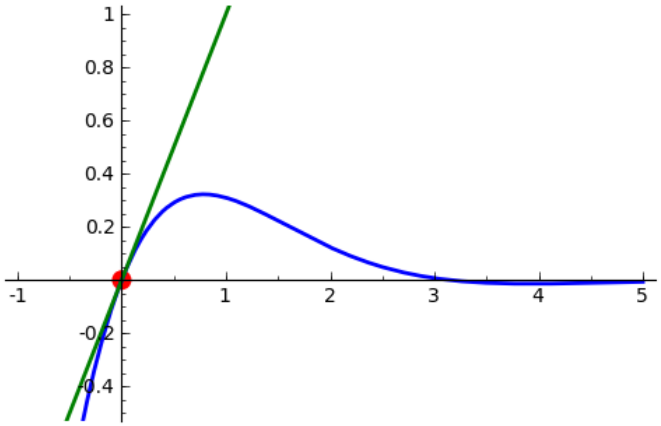



```
onda=animate([sin(x+k) for k in srange(0,10,0.5)],xmin=0,xmax=8*pi)
onda.show(delay=30, iterations=1)
```




```
f = sin(x)*e^(-x); dibujof = plot(f,-1,5, thickness=2);
punto = point((0,f(x=0)), pointsize=80, rgbcolor=(1,0,0));
@interact
def _(orden=(1..12)):
 fy = f.taylor(x,0,orden)
 dibujotaylor = plot(fy,-1, 5, color='green', thickness=2)
 show(punto + dibujof + dibujotaylor, ymin = -.5, ymax = 1)
```

orden 1




```
search_doc("animate")
```

## Search Documentation: "animate"

1. reference/genindex-A.html
2. reference/genindex-G.html
3. reference/genindex-P.html
4. reference/genindex-S.html
5. reference/genindex-all.html
6. reference/index.html
7. reference/modindex.html
8. reference/plotting.html
9. reference/sage/combinat/words/paths.html
10. reference/sage/plot/animate.html
11. reference/sage/plot/arrow.html
12. reference/sage/plot/plot.html

```
solve(x^2-2 == 0, x)
```

```
[x == -sqrt(2), x == sqrt(2)]
```

```
f=x^4+2*x^3-4*x^2-2*x+3;solve(f==0,x,multiplicities=true)
```

```
([x == -3, x == -1, x == 1], [1, 1, 2])
```

```
var('y');soluciones=solve([9*x-y==2,x^2+2*x*y+y==7],x,y)
soluciones[0][0].rhs()
```

```
-1/38*sqrt(709) - 5/38
```

```
sum(1/n^2 for n in (1..20)) # No sabe si en vez de 20 ponemos oo
```

```
17299975731542641/10838475198270720
```

```
maxima("sum(1/n^2,n,1,inf), simpsum")
```

```
%pi^2/6
```

```
A=matrix([[-4,1,0],[3,5,-2],[6,8,3]]); B=identity_matrix(3);
A; A^2*transpose(A)-5*B-(1/20)*det(A)*exp(B)
```

```
[-4  1  0]
[ 3  5 -2]
[ 6  8  3]
[ 29/4*e - 80            66           116]
[           48  29/4*e + 60            -6]
[           -2           418 29/4*e + 642]
```

```
v=vector([3,-2,8]); w=vector([-1,1,1])
v,w,v.dot_product(w); x=A\w; x
```

```
((3, -2, 8), (-1, 1, 1), 3)
(36/145, -1/145, -21/145)
```

```
H = matrix([[1/(i+j+1) for i in [0..2]] for j in [0..2]]);
H; H.inverse()
```

```
[  1 1/2 1/3]
[1/2 1/3 1/4]
[1/3 1/4 1/5]
[   9  -36   30]
[ -36  192 -180]
[  30 -180  180]
```

```
x = var("x"); y = function("y",x);
desolve(diff(y,x,2)-2*diff(y,x)-3*y == exp(x)*sin(x),y)
```

```
k1*e^(3*x) + k2*e^(-x) - 1/5*e^x*sin(x)
```

```
desolve(diff(y,x)+2*y-8==0,y,ics=[3,5]) # Cond. inicial y(3)=5
```

```
(4*e^(2*x) + e^6)*e^(-2*x)
```

```
desolvers?
```

**File:** /Applications/mates/sage/local/lib/python2.6/site-packages/sage/calculus/desolvers.py

**Type:** <type 'module'>

**Definition:** desolvers( [noargspec] )

**Docstring:**

Solving ordinary differential equations

This file contains functions useful for solving differential equations which occur commonly in a 1st semester differential equations course. For another numerical solver see `ode_solver()` function and optional package Octave.

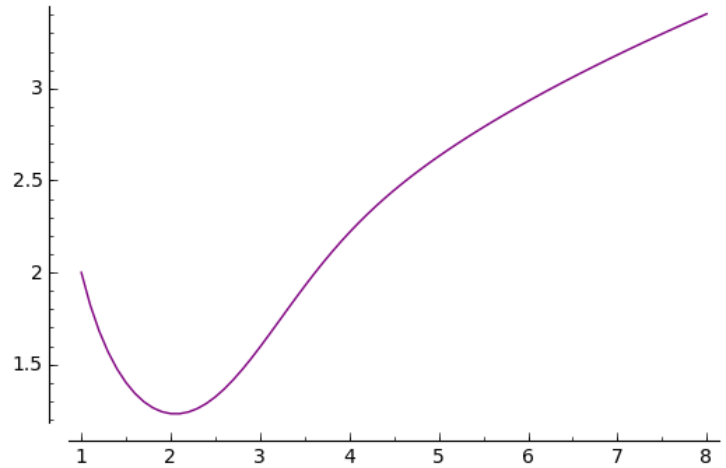
Commands:

- `desolve` - Computes the "general solution" to a 1st or 2nd order ODE via Maxima.
- `desolve_laplace` - Solves an ODE using laplace transforms via Maxima. Initials conditions are optional.
- `desolve_system` - Solves any size system of 1st order odes using Maxima. Initials conditions are optional.
- `desolve_rk4` - Solves numerically IVP for one first order equation, returns list of points or plot
- `desolve_system_rk4` - Solves numerically IVP for system of first order equations, returns list of points
- `eulers_method` - Approximate solution to a 1st order DE, presented as a table.
- `eulers_method_2x2` - Approximate solution to a 1st order system of DEs, presented as a table.
- `eulers_method_2x2_plot` - Plots the sequence of points obtained from Euler's method.

AUTHORS:

- David Joyner (3-2006) - Initial version of functions
- Marshall Hampton (7-2007) - Creation of Python module and testing
- Robert Bradshaw (10-2008) - Some interface cleanup.
- Robert Marik (10-2009) - Some bugfixes and enhancements

```
y = function('y',x);
sol=desolve_rk4(diff(y,x)+y*(y-2)==x-
3,y,ics=[1,2],step=0.1,end_points=8);
list_plot(sol, plotjoined=True, color="purple")
```




```
var("x"); sqrt(x^2); sqrt(x^4); simplify(_)
```

```
x
sqrt(x^2)
sqrt(x^4)
sqrt(x^4)
```

```
assume(x>0); simplify(sqrt(x^2))
```

```
x
```

```
sin(asin(x));asin(sin(x));simplify(_)
```

```
x
arcsin(sin(x))
arcsin(sin(x))
```

```
assume(-pi/2 <= x <= pi/2); simplify(asin(sin(x)))
```

```
x
```

```
var('k t'); assume(k, 'integer'); simplify(sin(t+2*k*pi))
```

```
(k, t)
sin(t)
```

```
find_root(x*exp(-x), 2, 100) # Chapuzas
```

```
99.999997605618816
```

```
t=-40.0 # Número real
sum([t^n/factorial(n) for n in [0..300]])
```

```
5.88116131704963
```

```
t=-40   # Número entero
N(sum([t^n/factorial(n) for n in [0..300]]))
```

```
4.24835425529159e-18
```

```
def letraDelDNI(n):
  """ Esta funcion calcula la letra de un DNI espanol """
  letras = "TRWAGMYFPDXBNJZSQVHLCKE"
  return letras[n%23]
letraDelDNI(51444857)
```

```
'K'
```

```
def f(n):
    if n <= 1: return 1
    elif n%2 == 0: return 2*f(n/2)
    else: return 3*f((n-1)/2)
f(12345678)
```

```
725594112
```

```
def is_prime_lucas_lehmer(p):
  s = Mod(4,2^p-1) # ¡Definimos s como un entero modular!
  for i in range(0, p-2):
      s = s^2 - 2
  return s==0
is_prime_lucas_lehmer(127)
```

```
True
```

```
time is_prime_lucas_lehmer(19937) # El mayor primo conocido en
1971
```

```
True
```