Sage Quick Reference: Calculus

William Stein

Sage Version 3.4

http://wiki.sagemath.org/quickref

Builtin constants and functions

Constants: π = `pi` e = `e` i = `I` = `i`

∞ = `oo` = `infinity` NaN=`NaN` log(2) =`log2`

ϕ = `golden_ratio` γ = `euler_gamma`

0.915 $\approx$ `catalan` 2.685 $\approx$ `khinchin`

0.660 $\approx$ `twinprime` 0.261 $\approx$ `merten` 1.902 $\approx$ `brun`

Approximate: `pi.n(digits=18)` = 3.14159265358979324

Builtin functions: `sin cos tan sec csc cot sinh cosh tanh sech csch coth log ln exp` ...

Defining symbolic expressions

Create symbolic variables:

`var("t u theta")` or `var("t,u,theta")`

Use `*` for multiplication and `^` for exponentiation:

$2x^5 + \sqrt{2}$ = `2*x^5 + sqrt(2)`

Typeset: `show(2*theta^5 + sqrt(2))` $\longrightarrow 2\theta^5 + \sqrt{2}$

Symbolic functions

Symbolic function (can integrate, differentiate, etc.):

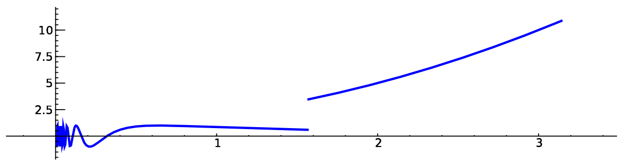
`f(a,b,theta) = a + b*theta^2`

Also, a "formal" function of theta:

`f = function('f',theta)`

Piecewise symbolic functions:

`Piecewise([[(0,pi/2),sin(1/x)],[(pi/2,pi),x^2+1]])`



Python functions

Defining:

```
def f(a, b, theta=1):
    c = a + b*theta^2
    return c
```

Inline functions:

```
    f = lambda a, b, theta = 1: a + b*theta^2
```

Simplifying and expanding

Below f must be symbolic (so **not** a Python function):

Simplify: `f.simplify_exp()`, `f.simplify_full()`, `f.simplify_log()`, `f.simplify_radical()`, `f.simplify_rational()`, `f.simplify_trig()`

Expand: `f.expand()`, `f.expand_rational()`

Equations

Relations: $f = g$: `f == g`, $f \neq g$: `f != g`,
$f \leq g$: `f <= g`, $f \geq g$: `f >= g`,
$f < g$: `f < g`, $f > g$: `f > g`

Solve $f = g$: `solve(f == g, x)`, and `solve([f == 0, g == 0], x,y)`

`solve([x^2+y^2==1, (x-1)^2+y^2==1],x,y)`

Solutions:

`S = solve(x^2+x+1==0, x, solution_dict=True)`

`S[0]["x"]` `S[1]["x"]` are the solutions

Exact roots: `(x^3+2*x+1).roots(x)`

Real roots: `(x^3+2*x+1).roots(x,ring=RR)`

Complex roots: `(x^3+2*x+1).roots(x,ring=CC)`

Factorization

Factored form: `(x^3-y^3).factor()`

List of (factor, exponent) pairs:

`(x^3-y^3).factor_list()`

Limits

$\lim_{x\to a} f(x)$ = `limit(f(x), x=a)`

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

$\lim_{x\to a^+} f(x)$ = `limit(f(x), x=a, dir='plus')`

`limit(1/x, x=0, dir='plus')`

$\lim_{x\to a^-} f(x)$ = `limit(f(x), x=a, dir='minus')`

`limit(1/x, x=0, dir='minus')`

Derivatives

$\frac{d}{dx}(f(x))$ = `diff(f(x),x)` = `f.diff(x)`

$\frac{\partial}{\partial x}(f(x,y))$ = `diff(f(x,y),x)`

`diff` = `differentiate` = `derivative`

`diff(x*y + sin(x^2) + e^(-x), x)`

Integrals

$\int f(x)dx$ = in

`integral(`

$\int_a^b f(x)dx$ = i

`integral(`

$\int_a^b f(x)dx \approx$ n

`numerical`

`assume(...)`:

`assume(x>`

Taylor and p

Taylor polyno

`taylor(f,x,a`

`taylor(sc`

Partial fractio

`(x^2/(x+1)^3`

Numerical r

Numerical roo

`(x^2 - 2)`

Maximize: fin

`f.find_m`

Minimize: finc

`f.find_m`

Minimization:

`minimize(`

Multivariabl

Gradient: `f.g`

`(x^2+y^2`

Hessian: `f.he`

`(x^2+y^2`

Jacobian matr

`jacobian`

Summing in

Not yet imple

`s = 'sum (1/`

`SR(sage.calc`