Sage Quick Reference: Calculus

William Stein (modified by nu)

Sage Version 3.4

http://wiki.sagemath.org/quickref

組込み定数と函数　Builtin constants and functions

定数: π=`pi`　e=`e`　i=`I`=`i`

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

ϕ=`golden_ratio`　γ=`euler_gamma`

0.915≈`catalan`　2.685≈`khinchin`　0.660≈`twinprime`

0.261≈`merten`　1.902≈`brun`

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

組込み函数: `sin cos tan sec csc cot sinh cosh tanh sech csch coth log ln exp` ...

. . . . ORGINAL TEXT

Constants: π=`pi`　e=`e`　i=`I`=`i`

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

ϕ=`golden_ratio`　γ=`euler_gamma`

0.915≈`catalan`　2.685≈`khinchin`　0.660≈`twinprime`

0.261≈`merten`　1.902≈`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` ...

シンボリックな数式の定義　Defining symbolic expressions

不定元 (symbolic variable) の生成:

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

かけ算は `*`, 冪乗は `^`: $2x^5+\sqrt{2}$ =`2*x^5 + sqrt(2)`

タイプセット: `show(2*theta^5 + sqrt(2))` $\longrightarrow 2\theta^5+\sqrt{2}$

. . . . ORGINAL TEXT

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))` $\longrightarrow 2\theta^5+\sqrt{2}$

シンボリックな函数　Symbolic functions

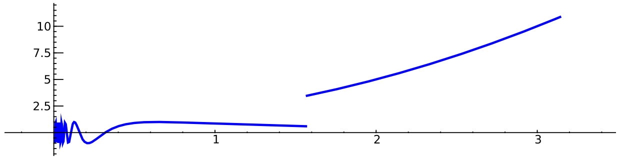
シンボリックな函数 (Symbolic function) (微分や積分ができる):

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

theta の “形式的な” 函数: `f = function('f',theta)`

区分的なシンボリックな函数:

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



. . . . ORGINAL TEXT

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

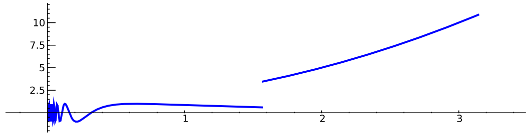
`f(a,b,theta) = a + b*theta^2`

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 の関数　Python functions

定義:

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

インライン関数:

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

. . . . ORGINAL TEXT

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
```

簡単化と展開　Simplifying and expanding

以下の f は, シンボリックな函数でなければならない (Python の関数では**ない**):

簡単化: `f.simplify_exp()` `f.simplify_full()`
`f.simplify_log()` `f.simplify_radical()`
`f.simplify_rational()` `f.simplify_trig()`

展開: `f.expand()` `f.expand_rational()`

. . . . ORGINAL TEXT

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()`

等式　Equations

関係式: $f=g$: `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`

$f=g$ を解く: `solve(f == g, x)` とか
`solve([f == 0, g == 0], x,y)`

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

解: `S = solve(x^2+x+1==0, x, solution_dict=True)`
`S[0]["x"]` `S[1]["x"]` are the solutions

厳密解: `(x^3+2*x+1).roots(x)`

実数解: `(x^3+2*x+1).roots(x,ring=RR)`

複素数解: `(x^3+2*x+1).roots(x,ring=CC)`

. . . . ORGINAL TEXT

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)`

因数分解　Factorization

因数分解: `(x^3-y^3).factor()`

(因数, 巾) というペアのリスト: `(x^3-y^3).factor_list()`

. . . . ORGINAL TEXT

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

List of (factor, exponent) pairs: `(x^3-y^3).factor_list()`

極限　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')`

. . . . ORGINAL TEXT

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

微分　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)`

. . . . ORGINAL TEXT

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

積分 Integrals

$\int f(x)dx =$ `integral(f,x) = f.integrate(x)`

```
integral(x*cos(x^2), x)
```

$\int_a^b f(x)dx =$ `integral(f,x,a,b)`

```
integral(x*cos(x^2), x, 0, sqrt(pi))
```

$\int_a^b f(x)dx \approx$ `numerical_integral(f(x),a,b)[0]`

```
numerical_integral(x*cos(x^2),0,1)[0]
```

`assume(...)`: 積分の際に質問されたら使う.

```
assume(x>0)
```

ORGINAL TEXT

$\int f(x)dx =$ `integral(f,x) = f.integrate(x)`

```
integral(x*cos(x^2), x)
```

$\int_a^b f(x)dx =$ `integral(f,x,a,b)`

```
integral(x*cos(x^2), x, 0, sqrt(pi))
```

$\int_a^b f(x)dx \approx$ `numerical_integral(f(x),a,b)[0]`

```
numerical_integral(x*cos(x^2),0,1)[0]
```

`assume(...)`: use if integration asks a question

```
assume(x>0)
```

テイラー展開と部分分数展開 Taylor and partial fraction expansion

a に関する次数 n のテイラー多項式:

`taylor(f,x,a,n)`$\approx c_0 + c_1(x-a) + \cdots + c_n(x-a)^n$

```
taylor(sqrt(x+1), x, 0, 5)
```

部分分数展開: `(x^2/(x+1)^3).partial_fraction()`

ORGINAL TEXT

Taylor polynomial, deg n about a:

`taylor(f,x,a,n)`$\approx c_0 + c_1(x-a) + \cdots + c_n(x-a)^n$

```
taylor(sqrt(x+1), x, 0, 5)
```

Partial fraction: `(x^2/(x+1)^3).partial_fraction()`

数値解と最適化 Numerical roots and optimization

数値解: `f.find_root(a, b, x)`

```
(x^2 - 2).find_root(1,2,x)
```

最大化: $f(x_0) = m$ が極大となる (m, x_0) を探す

`f.find_maximum_on_interval(a, b, x)`

最小化: $f(x_0) = m$ が極小となる (m, x_0) を探す

`f.find_minimum_on_interval(a, b, x)`

最小化: `minimize(f,`*start_point*`)`

```
minimize(x^2+x*y^3+(1-z)^2-1, [1,1,1])
```

ORGINAL TEXT

Numerical root: `f.find_root(a, b, x)`

```
(x^2 - 2).find_root(1,2,x)
```

Maximize: find (m, x_0) with $f(x_0) = m$ maximal

`f.find_maximum_on_interval(a, b, x)`

Minimize: find (m, x_0) with $f(x_0) = m$ minimal

`f.find_minimum_on_interval(a, b, x)`

Minimization: `minimize(f,`*start_point*`)`

```
minimize(x^2+x*y^3+(1-z)^2-1, [1,1,1])
```

多変数函数 Multivariable calculus

勾配 (Gradient): `f.gradient()` or `f.gradient(`*vars*`)`

```
(x^2+y^2).gradient([x,y])
```

ヘッセ行列 (Hessian): `f.hessian()`

```
(x^2+y^2).hessian()
```

ヤコビ行列: `jacobian(f,`*vars*`)`

```
jacobian(x^2 - 2*x*y, (x,y))
```

ORGINAL TEXT

Gradient: `f.gradient()` or `f.gradient(`*vars*`)`

```
(x^2+y^2).gradient([x,y])
```

Hessian: `f.hessian()`

```
(x^2+y^2).hessian()
```

Jacobian matrix: `jacobian(f,`*vars*`)`

```
jacobian(x^2 - 2*x*y, (x,y))
```

無限級数 Summing infinite series

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

まだ実装されていないが, Maxima を使うことが出来る:

```
s = 'sum (1/n^2,n,1,inf), simpsum'
SR(sage.calculus.calculus.maxima(s))
```

$\longrightarrow \pi^2/6$

ORGINAL TEXT

Not yet implemented, but you can use Maxima:

```
s = 'sum (1/n^2,n,1,inf), simpsum'
SR(sage.calculus.calculus.maxima(s))
```

$\longrightarrow \pi^2/6$