

Sage Quick Reference (Basic Math)

Peter Jipsen, version 1.1

latest version at wiki.sagemath.org/quickref

GNU Free Document License, extend for your own use

Aim: map standard math notation to Sage commands

Notebook (and commandline)

Evaluate cell: `<shift-enter>`

`com<tab>` tries to complete *command*

`command?<tab>` shows documentation

`command??<tab>` shows source

`a.<tab>` shows all methods for object **a** (more: `dir(a)`)

`search_doc('string or regexp')` shows links to docs

`search_src('string or regexp')` shows links to source

`lprint()` toggle LaTeX output mode

`version()` print version of Sage

Insert cell: click on blue line between cells

Delete cell: delete content then backspace

Numerical types

Integers: $\mathbb{Z} = \mathbb{ZZ}$ e.g. -2 -1 0 1 10^{100}

Rationals: $\mathbb{Q} = \mathbb{QQ}$ e.g. 1/2 1/1000 314/100 -42

Decimals: $\mathbb{R} \approx \mathbb{RR}$ e.g. .5 0.001 3.14 -42.

Complex: $\mathbb{C} \approx \mathbb{CC}$ e.g. 1+i 2.5-3*i

Basic constants and functions

Constants: $\pi = \text{pi}$ $e = \text{e}$ $i = \text{i}$ $\infty = \text{oo}$

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

Functions: `sin cos tan sec csc cot sinh cosh tanh`

`sech csch coth log ln exp`

$ab = \mathbf{a}*\mathbf{b}$ $\frac{a}{b} = \mathbf{a}/\mathbf{b}$ $a^b = \mathbf{a}^{\mathbf{b}}$ $\sqrt{x} = \text{sqrt}(x)$

$\sqrt[n]{x} = \mathbf{x}^{(1/n)}$ $|x| = \text{abs}(x)$ $\log_b(x) = \text{log}(x,b)$

Symbolic variables: e.g. `t,u,v,y,z = var('t u v y z')`

Define function: e.g. $f(x) = x^2$ `f(x)=x^2`

or `f=lambda x: x^2` or `def f(x): return x^2`

Operations on expressions

`factor(...)` `expand(...)` `(...).simplify...`

Symbolic equations: `f(x)==g(x)`

`_` is previous output

`_+a` `_-a` `_*a` `_/a` manipulates equation

Solve $f(x) = g(x)$: `solve(f(x)==g(x),x)`

`solve([f(x,y)==0, g(x,y)==0], x,y)`

`find_root(f(x), a, b)` find $x \in [a,b]$ s.t. $f(x) \approx 0$

$\sum_{i=k}^n f(i) = \text{sum}([f(i) \text{ for } i \text{ in } [k..n]])$

$\prod_{i=k}^n f(i) = \text{prod}([f(i) \text{ for } i \text{ in } [k..n]])$

Calculus

$\lim_{x \rightarrow a} f(x) = \text{limit}(f(x), x=a)$

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

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

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

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

`diff = differentiate = derivative`

$\int f(x)dx = \text{integral}(f(x), x)$

`integral = integrate`

$\int_a^b f(x)dx = \text{integral}(f(x), x,a,b)$

Taylor polynomial, deg n about a : `taylor(f(x),x,a,n)`

2d graphics

`line([(x1,y1),...,(xn,yn)], options)`

`polygon([(x1,y1),...,(xn,yn)], options)`

`circle((x,y),r, options)`

`text("txt", (x,y), options)`

options as in `plot.options`, e.g. `thickness=pixel`,

`rgbcolor=(r,g,b)`, `hue=h` where $0 \leq r, b, g, h \leq 1$

use option `figsize=[w,h]` to adjust aspect ratio

`plot(f(x), xmin, xmax, options)`

`parametric_plot((f(t),g(t)), tmin, tmax, options)`

`polar_plot(f(t), tmin, tmax, options)`

combine graphs: `circle((1,1),1)+line([(0,0),(2,2)])`

`animate(list of graphics objects, options).show(delay=20)`

3d graphics

`line3d([(x1,y1,z1),...,(xn,yn,zn)], options)`

`sphere((x,y,z),r, options)`

`tetrahedron((x,y,z),size, options)`

`cube((x,y,z),size, options)`

`octahedron((x,y,z),size, options)`

`dodecahedron((x,y,z),size, options)`

`icosahedron((x,y,z),size, options)`

options e.g. `aspect_ratio=[1,1,1]` `color='red'` `opacity`

`plot3d(f(x,y), [xb,xe], [yb,ye], options)`

add option `plot_points=[m,n]` or use `plot3d.adaptive`

`parametric_plot3d((f(t),g(t),h(t)), [tb,te], options)`

`parametric_plot3d((f(u,v),g(u,v),h(u,v)),`

`[ub,ue], [vb,ve], options)`

use `+` to combine graphics objects

Discrete math

`[x] = floor(x)` `[x] = ceil(x)`

Remainder of n divided by $k = \mathbf{n}\%k$ $k|n$ iff $\mathbf{n}\%k==0$

$n! = \text{factorial}(n)$ $\binom{x}{m} = \text{binomial}(x,m)$

$\phi = \text{golden_ratio}$ $\phi(n) = \text{euler_phi}(n)$

Strings: e.g. `s = 'Hello' = "Hello" = ""+"He"+'llo'`

`s[0]='H'` `s[-1]='o'` `s[1:3]='el'` `s[3:]='lo'`

Lists: e.g. `[1,'Hello',x] = []+[1,'Hello']+x`

Tuples: e.g. `(1,'Hello',x)` (immutable)

Sets: e.g. `{1,2,1,a} = Set([1,2,1,'a'])` ($= \{1,2,a\}$)

List comprehension $\approx$ set builder notation, e.g.

$\{f(x) : x \in X, x > 0\} = \text{Set}([f(x) \text{ for } x \text{ in } X \text{ if } x > 0])$

Linear algebra

$\begin{pmatrix} 1 \\ 2 \end{pmatrix} = \text{vector}([1,2])$

$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \text{matrix}([[1,2],[3,4]])$

$\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = \text{det}(\text{matrix}([[1,2],[3,4]]))$

$Av = \mathbf{A}*v$ $A^{-1} = \mathbf{A}^{-1}$ $A^t = \mathbf{A.transpose}()$

methods: `nrows()` `ncols()` `nullity()` `rank()` `trace()...`

Sage modules and packages

`from module_name import *` (many preloaded)

e.g. `calculus coding combinat crypto functions`

`games geometry graphs groups logic matrix`

`numerical plot probability rings sets stats`

`sage.module_name.all.<tab>` shows exported commands

Std packages: Maxima GP/PARI GAP Singular R Shell ...

Opt packages: Biopython Fricas(Axiom) Gnuplot Kash ...

`%package_name` then use package command syntax

`time command` to show timing information