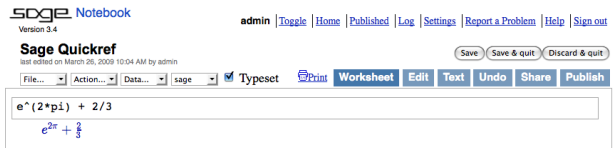
Краткое Руководство по Sage

Уильям Стайн (основано на работе П. Джипсен)

Notebook/Блокнот



Вычислить ячейку: ⟨shift-enter⟩

Вычислить, создав новую ячейку: ⟨alt-enter⟩

Разбить ячейку: ⟨control-;⟩

Соединить ячейки: ⟨control-backspace⟩

Вставить математическую ячейку: щёлкнуть мышью синюю линию между ячейками

Вставить ячейку с текстом/HTML: Shift+щёлкнуть мышью синюю линию между ячейками

Удалить ячейку: удалить содержимое, затем нажать Backspace

Командная строка

Автодополнение *команды*: *com*⟨tab⟩

Показать список команд, содержащих "bar": **bar**?

Показать документацию: *команда*?⟨tab⟩

Показать исходный код: *команда*??⟨tab⟩

Показать методы для объекта `a`: `a.`⟨tab⟩ (показать больше: `dir(a)`)

Показать скрытые методы для объекта: `a`: `a._`⟨tab⟩

Полнотекстовый поиск в документах:

`search_doc("`*string или regexp*`")`

Поиск в исходном коде:

`search_src("`*string или regexp*`")`

Предыдущий вывод данных: `_ is`

Числа

Целые: $\mathbf{Z}$ = `ZZ` напр. `-2 -1 0 1 10^100`

Рациональные: $\mathbf{Q}$ = `QQ` напр. `1/2 1/1000 314/100 -2/1`

Реальные: $\mathbf{R}$ ≈ `RR` напр. `.5 0.001 3.14 1.23e10000`

Комплексные: $\mathbf{C}$ ≈ `CC` напр. `CC(1,1) CC(2.5,-3)`

Двойная точность: `RDF` and `CDF` напр. `CDF(2.1,3)`

Mod n: $\mathbf{Z}/n\mathbf{Z}$ = `Zmod` напр. `Mod(2,3) Zmod(3)(2)`

Конечные поля: $\mathbf{F}_q$ = `GF` напр. `GF(3)(2) GF(9,"a").0`

Полиномы: $R[x,y]$ напр. `S.<x,y>=QQ[] x+2*y^3`

Ряды: $R[[t]]$ напр. `S.<t>=QQ[[]] 1/2+2*t+O(t^2)`

p-адические числа: $\mathbf{Z}_p$ ≈`Zp`, $\mathbf{Q}_p$ ≈`Qp` напр. `2+3*5+O(5^2)`

Алгебраическое замыкание: $\overline{\mathbf{Q}}$ = `QQbar` напр. `QQbar(2^(1/5))`

Интервальная арифметика: `RIF` напр. `sage: RIF((1,1.00001))`

Числовое поле: `R.<x>=QQ[]; K. <a> = NumberField(x^3+x+1)`

Арифметика

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

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

Суммы: $\sum_{i=k}^{n} f(i)$ = `sum(f(i) for i in (k..n))`

Произведения: $\prod_{i=k}^{n} f(i)$ = `prod(f(i) for i in (k..n))`

Константы и функции

Константы: π = `pi` e = `e` i = `i` ∞ = `oo`

ϕ = `golden_ratio` γ = `euler_gamma`

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

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

Функции Python: `def f(x): return x^2`

Интерактивные функции

Вставить `@interact` перед функцией (var определяют аргументы):

```
@interact
def f(n=[0..4], s=(1..5), c=Color("red")):
  var("x");show(plot(sin(n+x^s),-pi,pi,color=c))
```

Символические выражения

Задать новые символические переменные:

```
var("t u v y z")
```

Символическая функция: напр. $f(x) = x^2$

```
f(x)=x^2
```

Отношения: `f==g f<=g f>=g f<g f>g`

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

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

`factor(...) expand(...) (...).simplify_...`

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

Математический анализ

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

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

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

`diff` = `дифференцировать` = `производная`

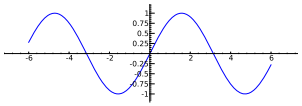
$\int f(x)dx$ = `integral(f(x),x)`

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

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

полином Тейлора, порядка n в точке a: `taylor(f(x),x,`a`,`n`)`

2D графика



Линия: `line([(`x_1`,`y_1`),...,(`x_n`,`y_n`)], `*опции*`)`

Многоугольник: `polygon([(`x_1`,`y_1`),...,(`x_n`,`y_n`)],` *опции*`)`

Окружность: `circle((`x`,`y`),`r`,` *опции*`)`

Текст: `text("txt",(`x`,`y`),` *опции*`)`

Опции: `plot.options`, `thickness=`*число* (*толщина линии в пикселях*), `rgbcolor=(`r`,`g`,`b`)` (*цвет*), `hue=`h (*оттенок*), причем $0 \le r,b,g,h \le 1$

Показать графику: `show(`*графика*`,` *опции*`)`

Задать размер: `figsize=[w,h]`

Задать пропорции: `aspect_ratio=`*число*

График функции: `plot(f(`x`),(`$x, x_{\min}, x_{\max}$`),` *опции*`)`

Параметрический график:

`parametric_plot((f(`t`),g(`t`)),(`$t, t_{\min}, t_{\max}$`),` *опции*`)`

График в полярных координатах:

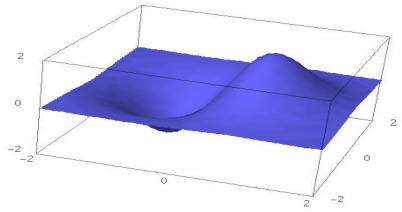
`polar_plot(f(`t`),(`$t, t_{\min}, t_{\max}$`),` *опции*`)`

Наложение графических элементов:

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

Анимировать графику (с задержкой, напр. 20 секунд): `animate(`*список, опции*`).show(delay=20)`

3D графика



3D линия: `line3d([(`x_1`,`y_1`,`z_1`),...,(`x_n`,`y_n`,`z_n`)],` *опции*)

Сфера: `sphere((`x`,`y`,`z`),` *размер, опции*)

3D текст: `text3d("текст",` (x,y,z)`,`*размер, опции*)

Тетраэдр: `tetrahedron((`x`,`y`,`z`),` *размер, опции*)

Куб: `cube((`x`,`y`,`z`),`*размер, опции*)

Октаэдр: `octahedron((`x`,`y`,`z`),` *размер, опции*)

Додекаэдр: `dodecahedron((`x`,`y`,`z`),` *размер, опции*)

Икосаэдр: `icosahedron((`x`,` *размер, опции*)

График: `plot3d(f(`x,y`),(`$x, x_\mathrm{b}, x_\mathrm{e}$`),` $(y, y_\mathrm{b}, y_\mathrm{e})$`,` *опции*)

Параметрический график:

`parametric_plot3d((f,g,h),(`$t, t_\mathrm{b}, t_\mathrm{e}$`),` *опции*)

`parametric_plot3d((f(`u,v`),g(`u,v`),h(`u,v`)),`
$(u, u_\mathrm{b}, u_\mathrm{e})$`,(`$v, v_\mathrm{b}, v_\mathrm{e}$`),` *опции*)

Опции: `aspect_ratio=`$[1,1,1]$ (*пропорции*), `color="red"` (*цвет*), `opacity=0.5` (*прозрачность*), `figsize=6` (*размер*), `viewer="tachyon"` (*способ отображения*)

Дискретная математика

$\lfloor x \rfloor$ = `floor(x)` $\lceil x \rceil$ = `ceil(x)`

Остаток n от деления на k = `n%k` $k|n$ iff `n%k==0`

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

$\phi(n)$ = `euler_phi(`n`)`

Строка: напр. `s = "Hello"` = `"Hel"+'lo'`

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

Списки: напр. `[1,"Hello",x] = []+[1,"Hello"]+[x]`

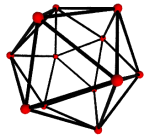
Кортежи: напр. `(1,"Hello",x)` (неизменяемый)

Множества: напр. $\{1,2,1,a\}$ = `Set([1,2,1,"a"])` $(= \{1,2,a\})$

Генераторы списков $\approx$ система обозначения множеств, напр.

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

Теория графов



Граф: `G = Graph({0:[1,2,3], 2:[4]})`

Ориентированный граф: `DiGraph(`*словарь*`)`

Семейства графов: `graphs.`⟨tab⟩

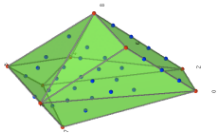
Инварианты: `G.chromatic_polynomial()`, `G.is_planar()`

Кратчайший путь: `G.shortest_path()`

Визуализация: `G.plot()`, `G.plot3d()`

Автоморфизмы: `G.automorphism_group()`, `G1.is_isomorphic(G2)`, `G1.is_subgraph(G2)`

Комбинаторика



Целочисленные последовательности: `sloane_find(`*list*`)`, `sloane.`⟨tab⟩

Разбиения: `P=Partitions(`*список*`)` `P.count()`

Сочетания: `C=Combinations(`*список*`)` `C.list()`

Декартово произведение: `CartesianProduct(P,C)`

Табло: `Tableau([[1,2,3],[4,5]])`

Слова: `W=Words("abc"); W("aabca")`

Частично упорядоченные множества: `Poset([[1,2],[4],[3],[4],[]])`

Системы корней: `RootSystem(["A",3])`

Кристаллы: `CrystalOfTableaux(["A",3], shape=[3,2])`

Целочисленные выпуклые многогранники: `A=random_matrix(ZZ,3,6,x=7)`

`L=LatticePolytope(A)` `L.npoints()` `L.plot3d()`

Матричная алгебра

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

$\begin{pmatrix}1&2\\3&4\end{pmatrix}$ = `matrix(QQ,[[1,2],[3,4]], sparse=False)`

$\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}$ = `matrix(QQ,2,3,[1,2,3, 4,5,6])`

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

Av = `A*v` A^{-1} = `A^-1` A^t = `A.transpose()`

Решить $Ax = v$: `A\v` or `A.solve_right(v)`

Решить $xA = v$: `A.solve_left(v)`

Матрица приведенного ступенчатого вида по строкам: `A.echelon_form()`

Ранг и дефект матрицы: `A.rank()` `A.nullity()`

Хессенбергова форма матрицы: `A.hessenberg_form()`

Характеристиечский полином: `A.charpoly()`

С: `A.eigenvalues()`

Собственные векторы: `A.eigenvectors_right()` (также левый)

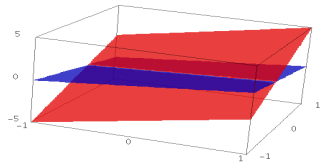
Процесс Грама-Шмидта: `A.gram_schmidt()`

Визуализировать: `A.plot()`

LLL алгоритм: `matrix(ZZ,...).LLL()`

Эрмитова матрица: `matrix(ZZ,...).hermite_form()`

Линейная алгебра



Векторное пространство K^n = `K^n` e.g. `QQ^3` `RR^2` `CC^4`

Линейная оболочка: `span(vectors,` *поле*)

например, `span([[1,2,3], [2,3,5]], QQ)`

Ядро: `A.right_kernel()` (также левое)

Сумма и пересечение: `V + W` and `V.intersection(W)`

Базис: `V.basis()`

Базисная матрица: `V.basis_matrix()`

Ограничить матрицу на подпространство: `A.restrict(V)`

Разложение вектора по базису: `V.coordinates(`*вектор*`)`

Вычислительная математика

Пакеты: `import numpy, scipy, cvxopt`

Минимизация: `var("x y z")`

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

Теория чисел

Простые числа: `prime_range(n,m)`, `is_prime`, `next_prime`

Разложение на простые множители: `factor(n)`, `qsieve(n)`, `ecm.factor(n)`

Символ Кронекера: $\left(\frac{a}{b}\right)$ = `kronecker_symbol(`a`,`b`)`

Цепные дроби: `continued_fraction(x)`

Числа Бернулли: `bernoulli(n)`, `bernoulli_mod_p(p)`

Эллиптические кривые:

`EllipticCurve([`a_1, a_2, a_3, a_4, a_6`])`

Характеры Дирихле: `DirichletGroup(`N`)`

Модулярные формы: `ModularForms(`*уровень*, *знак*`)`

Модулярные символы: `ModularSymbols(`*уровень*`,` *вес*`,` *знак*`)`

Модули Брандта: `BrandtModule(`*уровень*`,` *вес*`)`

Модулярные Абелевы многообразия: `J0(`N`)`, `J1(`N`)`

Теория групп

Группы перестановок: `G = PermutationGroup([[(1,2,3),(4,5)],[(3,4)]])`

Симметрические группы: `SymmetricGroup(`n`)`

Знакопеременные группы: `AlternatingGroup(`n`)`

Абелевы группы: `AbelianGroup([3,15])`

Матричные группы: `GL, SL, Sp, SU, GU, SO, GO`

Функции: подгруппа Силова `G.sylow_subgroup(p)`, таблица характеров `G.character_table()`, нормальная подгруппа `G.normal_subgroups()`, граф Кэли `G.cayley_graph()`

Некоммутативные кольца

Кватернионы: `Q.<i,j,k> = QuaternionAlgebra(a,b)`

Свободная алгебра: `R.<a,b,c> = FreeAlgebra(QQ, 3)`

Модули Python

Загрузить модуль: `import` *название_модуля*

Автодополнение для названия модуля: *название_модуля*⟨tab⟩

Помощь: `help(`*название_модуля*`)`

Профилирование и отладка

Показать время выполнения команды: `time` *команда*:

Замерить точное время выполнения команды: `timeit(`*команда*`)`

Процессорное время: `t = cputime(); cputime(t)`

Реальное время (выполнения команды): `t = walltime(); walltime(t)`

Включить интерактивный отладчик (только в режиме командной строки): `%pdb`

Профилировать команду (только в режиме командной строки): `%prun` *команда*