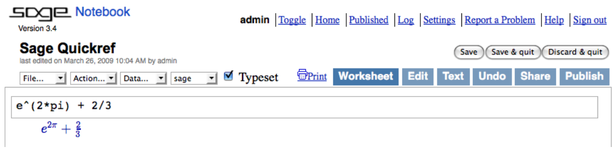
Sage Quick Reference

William Stein (based on work of P. Jipsen) (mod. by nu)

Notebook　Notebook



セルの評価: ⟨shift-enter⟩
セルを評価し新しいセルを作る: ⟨alt-enter⟩
セルの分割: ⟨control-;⟩
セルの結合: ⟨control-backspace⟩
数式セルの挿入: セルの間の青い線をクリック
Text/HTML セルの挿入: セルの間の青い線を shift-click
セルの削除: 内容を削除したあとで backspace

ORGINAL TEXT

Evaluate cell: ⟨shift-enter⟩
Evaluate cell creating new cell: ⟨alt-enter⟩
Split cell: ⟨control-;⟩
Join cells: ⟨control-backspace⟩
Insert math cell: click blue line between cells
Insert text/HTML cell: shift-click blue line between cells
Delete cell: delete content then backspace

コマンドライン　Command line

com⟨tab⟩ で *command* を補完
`*`*bar*`*?` で “*bar*” を含むコマンド名をリストアップ
command?⟨tab⟩ でドキュメントを表示
command??⟨tab⟩ でソースコードを表示
`a.`⟨tab⟩ でオブジェクト a のメソッドを表示　(`dir(a)` も)
`a._`⟨tab⟩ で a の hidden methods を表示
`search_doc("`*string or regexp*`")` ドキュメントの全文検索
`search_src("`*string or regexp*`")` ソースコードの検索
`_` は直前の出力

ORGINAL TEXT

com⟨tab⟩ complete *command*
`*`*bar*`*?` list command names containing “*bar*”
command?⟨tab⟩ shows documentation
command??⟨tab⟩ shows source code
`a.`⟨tab⟩ shows methods for object a　(more: `dir(a)`)
`a._`⟨tab⟩ shows hidden methods for object a
`search_doc("`*string or regexp*`")` fulltext search of docs
`search_src("`*string or regexp*`")` search source code
`_` is previous output

数　Numbers

整数: $\mathbb{Z}$ = `ZZ` 例 `-2 -1 0 1 10^100`
有理数: $\mathbb{Q}$ = `QQ` 例 `1/2 1/1000 314/100 -2/1`
実数: $\mathbb{R} \approx$ `RR` 例 `.5 0.001 3.14 1.23e10000`
複素数: $\mathbb{C} \approx$ `CC` 例 `CC(1,1) CC(2.5,-3)`
倍精度 (Double): `RDF` and `CDF` 例 `CDF(2.1,3)`
Mod n: $\mathbb{Z}/n\mathbb{Z}$ = `Zmod` 例 `Mod(2,3) Zmod(3)(2)`
有限体: $\mathbb{F}_q$ = `GF` 例 `GF(3)(2) GF(9,"a").0`
多項式: $R[x,y]$ 例 `S.<x,y>=QQ[] x+2*y^3`
巾級数: $R[[t]]$ 例 `S.<t>=QQ[[]] 1/2+2*t+O(t^2)`
p 進整数: $\mathbb{Z}_p \approx$`Zp`, $\mathbf{Q}_p \approx$`Qp` 例 `2+3*5+O(5^2)`
代数閉包: $\overline{\mathbb{Q}}$ = `QQbar` 例 `QQbar(2^(1/5))`
区間演算: `RIF` 例 `RIF((1,1.00001))`
数体: `R.<x>=QQ[]; K.<a>=NumberField(x^3+x+1)`

ORGINAL TEXT

Integers: $\mathbb{Z}$ = `ZZ` e.g. `-2 -1 0 1 10^100`
Rationals: $\mathbb{Q}$ = `QQ` e.g. `1/2 1/1000 314/100 -2/1`
Reals: $\mathbb{R} \approx$ `RR` e.g. `.5 0.001 3.14 1.23e10000`
Complex: $\mathbb{C} \approx$ `CC` e.g. `CC(1,1) CC(2.5,-3)`
Double precision: `RDF` and `CDF` e.g. `CDF(2.1,3)`
Mod n: $\mathbb{Z}/n\mathbb{Z}$ = `Zmod` e.g. `Mod(2,3) Zmod(3)(2)`
Finite fields: $\mathbb{F}_q$ = `GF` e.g. `GF(3)(2) GF(9,"a").0`
Polynomials: $R[x,y]$ e.g. `S.<x,y>=QQ[] x+2*y^3`
Series: $R[[t]]$ e.g. `S.<t>=QQ[[]] 1/2+2*t+O(t^2)`
p-adic numbers: $\mathbb{Z}_p \approx$`Zp`, $\mathbf{Q}_p \approx$`Qp` e.g. `2+3*5+O(5^2)`
Algebraic closure: $\overline{\mathbb{Q}}$ = `QQbar` e.g. `QQbar(2^(1/5))`
Interval arithmetic: `RIF` e.g. `RIF((1,1.00001))`
Number field: `R.<x>=QQ[]; K.<a>=NumberField(x^3+x+1)`

四則演算など　Arithmetic

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)`

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

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

ORGINAL TEXT

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)`

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

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

定数と函数　Constants and functions

定数: π = `pi`　e = `e`　i = `i`　∞ = `oo`
ϕ = `golden_ratio`　γ = `euler_gamma`
近似値: `pi.n(digits=18)` = 3.14159265358979324
函数:　`sin cos tan sec csc cot sinh cosh tanh sech csch coth log ln exp` ...
Python の関数: `def f(x): return x^2`

ORGINAL TEXT

Constants: π = `pi`　e = `e`　i = `i`　∞ = `oo`
ϕ = `golden_ratio`　γ = `euler_gamma`
Approximate: `pi.n(digits=18)` = 3.14159265358979324
Functions:　`sin cos tan sec csc cot sinh cosh tanh sech csch coth log ln exp` ...
Python function: `def f(x): return x^2`

インタラクティブな操作　Interactive functions

関数の前に `@interact` を置く (変数で controls が決まる)

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

ORGINAL TEXT

Put `@interact` before function (vars determine controls)

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

シンボリックな数式　Symbolic expressions

新しい不定元 (symbolic variables) を定義: `var("t u v y z")`
シンボリックな函数 (Symbolic function):
　例 $f(x) = x^2$　`f(x)=x^2`
関係式: `f==g f<=g f>=g f<g f>g`
$f = g$ を解く: `solve(f(x)==g(x), x)`
　`solve([f(x,y)==0, g(x,y)==0], x,y)`
`factor(...)`　`expand(...)`　`(...).simplify_...`
$x \in [a,b]$ s.t. $f(x) \approx 0$ を見付ける: `find_root(f(x), a, b)`

ORGINAL TEXT

Define new symbolic variables: `var("t u v y z")`
Symbolic function: e.g. $f(x) = x^2$　`f(x)=x^2`
Relations: `f==g f<=g f>=g f<g f>g`
Solve $f = g$: `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)` find $x \in [a,b]$ s.t. $f(x) \approx 0$

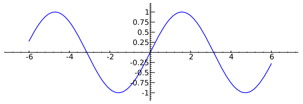
微分積分　Calculus

$\lim_{x\to a} f(x)$ = `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` = `differentiate` = `derivative`
$\int f(x)dx$ = `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)`
a に関する次数 n の Taylor 多項式: `taylor(f(x),x,`a`,`n`)`

ORIGINAL TEXT

$\lim_{x\to a} f(x)$ = 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 = differentiate = derivative
$\int f(x)dx$ = 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)
Taylor polynomial, deg n about a: taylor(f(x),x,a,n)

二次元グラフィックス 2D graphics

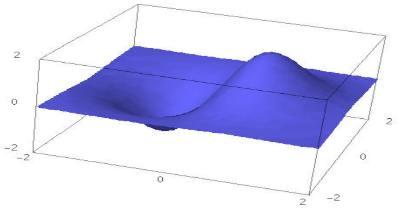


line([(x_1,y_1),...,(x_n,y_n)],*options*)
polygon([(x_1,y_1),...,(x_n,y_n)],*options*)
circle((x,y),r,*options*)
text("txt",(x,y),*options*)
options は plot.options にあるものを使用,
　例 thickness=*pixel*, rgbcolor=(r,g,b), hue=h
　ただし $0 \le r,b,g,h \le 1$
show(*graphic*, *options*)
　サイズの調整には figsize=[w,h] を使う
　縦横比を調整するには aspect_ratio=*number* を使う
plot(f(x),(x,$x_{\min}$,$x_{\max}$),*options*)
parametric_plot((f(t),g(t)),(t,$t_{\min}$,$t_{\max}$),*options*)
polar_plot(f(t),(t,$t_{\min}$,$t_{\max}$),*options*)
結合: circle((1,1),1)+line([(0,0),(2,2)])
animate(*list of graphics*, *options*).show(delay=20)

ORIGINAL TEXT

line([(x_1,y_1),...,(x_n,y_n)],*options*)
polygon([(x_1,y_1),...,(x_n,y_n)],*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 $0 \le r,b,g,h \le 1$
show(*graphic*, *options*)
　use figsize=[w,h] to adjust size
　use aspect_ratio=*number* to adjust aspect ratio
plot(f(x),(x,$x_{\min}$,$x_{\max}$),*options*)
parametric_plot((f(t),g(t)),(t,$t_{\min}$,$t_{\max}$),*options*)
polar_plot(f(t),(t,$t_{\min}$,$t_{\max}$),*options*)
combine: circle((1,1),1)+line([(0,0),(2,2)])
animate(*list of graphics*, *options*).show(delay=20)

三次元グラフィックス 3D graphics



line3d([(x_1,y_1,z_1),...,(x_n,y_n,z_n)],*options*)
sphere((x,y,z),r,*options*)
text3d("txt", (x,y,z), *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*)
plot3d(f(x,y),(x,x_b,x_e), (y,y_b,y_e),*options*)
parametric_plot3d((f,g,h),(t,t_b,t_e),*options*)
parametric_plot3d((f(u,v),g(u,v),h(u,v)),
　(u,u_b,u_e),(v,v_b,v_e),*options*)
options: aspect_ratio=[1,1,1], color="red",
　opacity=0.5, figsize=6, viewer="tachyon"

ORIGINAL TEXT

line3d([(x_1,y_1,z_1),...,(x_n,y_n,z_n)],*options*)
sphere((x,y,z),r,*options*)
text3d("txt", (x,y,z), *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*)
plot3d(f(x,y),(x,x_b,x_e), (y,y_b,y_e),*options*)
parametric_plot3d((f,g,h),(t,t_b,t_e),*options*)
parametric_plot3d((f(u,v),g(u,v),h(u,v)),
　(u,u_b,u_e),(v,v_b,v_e),*options*)
options: aspect_ratio=[1,1,1], color="red",
　opacity=0.5, figsize=6, viewer="tachyon"

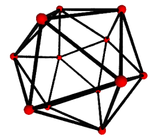
離散数学 Discrete math

$\lfloor x \rfloor$ = floor(x)　$\lceil x \rceil$ = ceil(x)
n を k で割った余り = n%k　$k|n$ iff n%k==0
$n!$ = factorial(n)　$\binom{x}{m}$ = binomial(x,m)
$\phi(n)$ = euler_phi(n)
文字列 (String): 例 s = "Hello" = "He"+'llo'
　s[0]="H"　s[-1]="o"　s[1:3]="el"　s[3:]="lo"
リスト (List): 例 [1,"Hello",x] = []+[1,"Hello"]+[x]
タプル (Tuple): 例 (1,"Hello",x)　(immutable)
集合 (Set): 例 $\{1,2,1,a\}$ = Set([1,2,1,"a"])　$(=\{1,2,a\})$
集合の内包的記法 ≈ リストの内包表記, 例
　$\{f(x)|x \in X, x > 0\}$ =Set([f(x) for x in X if x>0])

ORIGINAL TEXT

$\lfloor x \rfloor$ = floor(x)　$\lceil x \rceil$ = ceil(x)
Remainder of n divided by k = n%k　$k|n$ iff n%k==0
$n!$ = factorial(n)　$\binom{x}{m}$ = binomial(x,m)
$\phi(n)$ = euler_phi(n)
Strings: e.g. s = "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 ≈ set builder notation, e.g.
　$\{f(x)|x \in X, x > 0\}$ =Set([f(x) for x in X if x>0])

グラフ理論 Graph theory

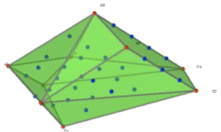


グラフ: G = Graph({0:[1,2,3], 2:[4]})
有向グラフ: DiGraph(*dictionary*)
グラフの族: graphs.⟨tab⟩
不変量: G.chromatic_polynomial(), G.is_planar()
パス: G.shortest_path()
可視化: G.plot(), G.plot3d()
自己同型: G.automorphism_group(),
　G1.is_isomorphic(G2), G1.is_subgraph(G2)

ORIGINAL TEXT

Graph: G = Graph({0:[1,2,3], 2:[4]})
Directed Graph: DiGraph(*dictionary*)
Graph families: graphs.⟨tab⟩
Invariants: G.chromatic_polynomial(), G.is_planar()
Paths: G.shortest_path()
Visualize: G.plot(), G.plot3d()
Automorphisms: G.automorphism_group(),
　G1.is_isomorphic(G2), G1.is_subgraph(G2)

組合せ論 Combinatorics



整数列: sloane_find(*list*), sloane.⟨tab⟩
分割: P=Partitions(n)　P.count()
組合せ (部分リスト): C=Combinations(*list*)　C.list()
直積: CartesianProduct(P,C)

ヤング盤 (Tableau): `Tableau([[1,2,3],[4,5]])`
ワード: `W=Words("abc"); W("aabca")`
半順序集合 (poset): `Poset([[1,2],[4],[3],[4],[]])`
ルート系: `RootSystem(["A",3])`
クリスタル: `CrystalOfTableaux(["A",3], shape=[3,2])`
格子多面体: `A=random_matrix(ZZ,3,6,x=7)`
`L=LatticePolytope(A) L.npoints() L.plot3d()`

ORGINAL TEXT

Integer sequences: `sloane_find(`*list*`)`, `sloane.`⟨tab⟩
Partitions: `P=Partitions(`n`)` `P.count()`
Combinations: `C=Combinations(`*list*`)` `C.list()`
Cartesian product: `CartesianProduct(P,C)`
Tableau: `Tableau([[1,2,3],[4,5]])`
Words: `W=Words("abc"); W("aabca")`
Posets: `Poset([[1,2],[4],[3],[4],[]])`
Root systems: `RootSystem(["A",3])`
Crystals: `CrystalOfTableaux(["A",3], shape=[3,2])`
Lattice Polytopes: `A=random_matrix(ZZ,3,6,x=7)`
`L=LatticePolytope(A) L.npoints() L.plot3d()`

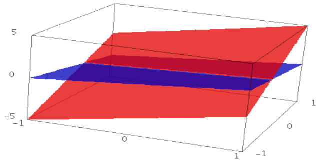
行列代数 Matrix algebra

$\begin{pmatrix}1\\2\end{pmatrix}$ = `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()`
$Ax = v$ を解く: `A\v` or `A.solve_right(v)`
$xA = v$ を解く: `A.solve_left(v)`
被約行階段行列: `A.echelon_form()`
階数と退化: `A.rank() A.nullity()`
Hessenberg 型: `A.hessenberg_form()`
特性多項式: `A.charpoly()`
固有値: `A.eigenvalues()`
固有ベクトル: `A.eigenvectors_right()` (also left)
Gram-Schmidt: `A.gram_schmidt()`
可視化: `A.plot()`
LLL reduction: `matrix(ZZ,...).LLL()`
Hermite 形式: `matrix(ZZ,...).hermite_form()`

ORGINAL TEXT

$\begin{pmatrix}1\\2\end{pmatrix}$ = `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()`
Solve $Ax = v$: `A\v` or `A.solve_right(v)`
Solve $xA = v$: `A.solve_left(v)`
Reduced row echelon form: `A.echelon_form()`
Rank and nullity: `A.rank() A.nullity()`
Hessenberg form: `A.hessenberg_form()`
Characteristic polynomial: `A.charpoly()`
Eigenvalues: `A.eigenvalues()`
Eigenvectors: `A.eigenvectors_right()` (also left)
Gram-Schmidt: `A.gram_schmidt()`
Visualize: `A.plot()`
LLL reduction: `matrix(ZZ,...).LLL()`
Hermite form: `matrix(ZZ,...).hermite_form()`

線形代数 Linear algebra



ベクトル空間 K^n = `K^n` 例 `QQ^3 RR^2 CC^4`
部分空間: `span(vectors, `*field*`)`
例 `span([[1,2,3], [2,3,5]], QQ)`
Kernel: `A.right_kernel()` (`left_` も)
和と共通部分: `V + W` と `V.intersection(W)`
基底: `V.basis()`
基底行列: `V.basis_matrix()`
行列を部分空間への制限: `A.restrict(V)`
基底を使ったベクトルの表示: `V.coordinates(`*vector*`)`

ORGINAL TEXT

Vector space K^n = `K^n` e.g. `QQ^3 RR^2 CC^4`
Subspace: `span(vectors, `*field*`)`
E.g., `span([[1,2,3], [2,3,5]], QQ)`
Kernel: `A.right_kernel()` (also left)
Sum and intersection: `V + W` and `V.intersection(W)`
Basis: `V.basis()`
Basis matrix: `V.basis_matrix()`
Restrict matrix to subspace: `A.restrict(V)`
Vector in terms of basis: `V.coordinates(`*vector*`)`

数値計算 Numerical mathematics

パッケージ: `import numpy, scipy, cvxopt`
最小化: `var("x y z")`
`minimize(x^2+x*y^3+(1-z)^2-1, [1,1,1])`

ORGINAL TEXT

Packages: `import numpy, scipy, cvxopt`
Minimization: `var("x y z")`
`minimize(x^2+x*y^3+(1-z)^2-1, [1,1,1])`

整数論 Number theory

素数: `prime_range(n,m)`, `is_prime`, `next_prime`
素因数分解: `factor(n)`, `qsieve(n)`, `ecm.factor(n)`
Kronecker symbol: $\left(\frac{a}{b}\right)$ = `kronecker_symbol(`a,b`)`
連分数: `continued_fraction(x)`
Bernoulli 数: `bernoulli(n)`, `bernoulli_mod_p(p)`
楕円曲線: `EllipticCurve(`$[a_1,a_2,a_3,a_4,a_6]$`)`
Dirichlet characters: `DirichletGroup(`N`)`
Modular forms: `ModularForms(`*level*, *weight*`)`
Modular symbols: `ModularSymbols(`*level*,*weight*,*sign*`)`
Brandt modules: `BrandtModule(`*level*, *weight*`)`
Modular abelian varieties: `J0(`N`)`, `J1(`N`)`

ORGINAL TEXT

Primes: `prime_range(n,m)`, `is_prime`, `next_prime`
Factor: `factor(n)`, `qsieve(n)`, `ecm.factor(n)`
Kronecker symbol: $\left(\frac{a}{b}\right)$ = `kronecker_symbol(`a,b`)`
Continued fractions: `continued_fraction(x)`
Bernoulli numbers: `bernoulli(n)`, `bernoulli_mod_p(p)`
Elliptic curves: `EllipticCurve(`$[a_1,a_2,a_3,a_4,a_6]$`)`
Dirichlet characters: `DirichletGroup(`N`)`
Modular forms: `ModularForms(`*level*, *weight*`)`
Modular symbols: `ModularSymbols(`*level*,*weight*,*sign*`)`
Brandt modules: `BrandtModule(`*level*, *weight*`)`
Modular abelian varieties: `J0(`N`)`, `J1(`N`)`

群論 Group theory

`G = PermutationGroup([[(1,2,3),(4,5)],[(3,4)]])`
`SymmetricGroup(`n`)`, `AlternatingGroup(`n`)`
アーベル群: `AbelianGroup([3,15])`
行列群: `GL`, `SL`, `Sp`, `SU`, `GU`, `SO`, `GO`
関数: `G.sylow_subgroup(p)`, `G.character_table()`,
`G.normal_subgroups()`, `G.cayley_graph()`

ORGINAL TEXT

`G = PermutationGroup([[(1,2,3),(4,5)],[(3,4)]])`
`SymmetricGroup(`n`)`, `AlternatingGroup(`n`)`
Abelian groups: `AbelianGroup([3,15])`
Matrix groups: `GL`, `SL`, `Sp`, `SU`, `GU`, `SO`, `GO`
Functions: `G.sylow_subgroup(p)`, `G.character_table()`,
`G.normal_subgroups()`, `G.cayley_graph()`

非可換環 Noncommutative rings

四元数: `Q.<i,j,k> = QuaternionAlgebra(a,b)`
自由代数: `R.<a,b,c> = FreeAlgebra(QQ, 3)`

ORGINAL TEXT

Quaternions: `Q.<i,j,k> = QuaternionAlgebra(a,b)`
Free algebra: `R.<a,b,c> = FreeAlgebra(QQ, 3)`

Python のモジュール Python modules

`import` *module_name*

`module_name.`⟨tab⟩ and `help(module_name)`

ORGINAL TEXT

`import` *module_name*

`module_name.`⟨tab⟩ and `help(module_name)`

解析とデバッグ Profiling and debugging

`time` *command*: timing information の表示

`timeit("command")`: accurately time command

`t = cputime(); cputime(t)`: 経過した CPU time

`t = walltime(); walltime(t)`: 経過した wall time

`%pdb`: interactive debugger を開始 (command line only)

`%prun command`: profile command (command line only)

ORGINAL TEXT

`time` *command*: show timing information

`timeit("command")`: accurately time command

`t = cputime(); cputime(t)`: elapsed CPU time

`t = walltime(); walltime(t)`: elapsed wall time

`%pdb`: turn on interactive debugger (command line only)

`%prun command`: profile command (command line only)