PARI-GP Reference Card

(PARI-GP version 2.6.1)

Note: optional arguments are surrounded by braces {}.
To start the calculator, type its name in the terminal: `gp`
To exit `gp`, type `quit`, `\q`, or `<C-D>` at prompt.

Help

- describe function — `?`*function*
- extended description — `??`*keyword*
- list of relevant help topics — `???`*pattern*

Input/Output

- previous result, the result before — `%`, `%``, `%```, etc.
- n-th result since startup — `%`n
- separate multiple statements on line — `;`
- extend statement on additional lines — `\`
- extend statements on several lines — `{`seq_1`;` seq_2`;}`
- comment — `/* ... */`
- one-line comment, rest of line ignored — `\\ ...`

Metacommands & Defaults

- set default d to *val* — `default`$(\{d\}, \{val\}, \{flag\})$
- toggle timer on/off — `#`
- print time for last result — `##`
- print defaults — `\d`
- set debug level to n — `\g` n
- set memory debug level to n — `\gm` n
- set output mode (raw=0, default=1) — `\o` n
- set n significant digits — `\p` n
- set n terms in series — `\ps` n
- quit GP — `\q`
- print the list of PARI types — `\t`
- print the list of user-defined functions — `\u`
- read file into GP — `\r` *filename*

Debugger / break loop

- get out of break loop — `break` or `<C-D>`
- go up n frames — `dbg_up`$(\{n\})$
- examine object o — `dbg_x`(o)

PARI Types & Input Formats

- `t_INT`/`t_REAL`. Integers, Reals — $\pm n$, $\pm n.ddd$
- `t_INTMOD`. Integers modulo m — `Mod`(n, m)
- `t_FRAC`. Rational Numbers — n/m
- `t_FFELT`. Elt in finite field F_q — `ffgen(q)`
- `t_COMPLEX`. Complex Numbers — $x + y *$ `I`
- `t_PADIC`. p-adic Numbers — $x +$ `O`$(p$`^`$k)$
- `t_QUAD`. Quadratic Numbers — $x + y *$ `quadgen`(D)
- `t_POLMOD`. Polynomials modulo g — `Mod`(f, g)
- `t_POL`. Polynomials — $a * x$`^`$n + \cdots + b$
- `t_SER`. Power Series — $f +$ `O`$(x$`^`$k)$
- `t_QFI`/`t_QFR`. Imag/Real bin. quad. forms — `Qfb`$(a, b, c, \{d\})$
- `t_RFRAC`. Rational Functions — f/g
- `t_VEC`/`t_COL`. Row/Column Vectors — `[`x, y, z`]`, `[`x, y, z`]~`
- `t_MAT`. Matrices — `[`x, y`;`z, t`;`u, v`]`
- `t_LIST`. Lists — `List([`x, y, z`])`
- `t_STR`. Strings — `"abc"`

Reserved Variable Names

- $\pi = 3.14\ldots$, $\gamma = 0.57\ldots$, $C = 0.91\ldots$ — `Pi`, `Euler`, `Catalan`
- square root of -1 — `I`
- big-oh notation — `O`

Information about an Object

- PARI type of object x — `type`(x)
- length of x / size of x in memory — `#`x, `sizebyte`(x)
- real or p-adic precision of x — `precision`(x), `padicprec`

Operators

- basic operations — `+`, `-` , `*`, `/`, `^`
- `i=i+1`, `i=i-1`, `i=i*j`, ... — `i++`, `i--`, `i*=j`,...
- euclidean quotient, remainder — x`\/`y, x`\`y, x`%`y, `divrem`(x, y)
- shift x left or right n bits — x`<<`n, x`>>`n or `shift`$(x, \pm n)$
- comparison operators — `<=`, `<`, `>=`, `>`, `==`, `!=`, `===`, `lex`, `cmp`
- boolean operators (or, and, not) — `||`, `&&`, `!`
- bit operations — `bitand`, `bitneg`, `bitor`, `bitxor`
- sign of $x = -1, 0, 1$ — `sign`(x)
- maximum/minimum of x and y — `max`, `min`(x, y)
- integer or real factorial of x — x`!` or `factorial`(x)
- derivative of f w.r.t. x — f`'`
- apply differential operator — `diffop`
- restore x as a formal variable — x`='`x
- simultaneous assignment $x \leftarrow v_1$, $y \leftarrow v_2$ — `[x,y] = v`

Select Components

- n-th component of x — `component`(x, n)
- n-th component of vector/list x — x`[`n`]`
- components $a, a+1, \ldots, b$ of vector x — x`[`a`..`b`]`
- (m, n)-th component of matrix x — x`[`m, n`]`
- row m or column n of matrix x — x`[`m`,]`, x`[,`n`]`
- numerator/denominator of x — `numerator`(x), `denominator`

Conversions

- to vector, matrix, set, list, string — `Col`/`Vec`,`Mat`,`Set`,`List`,`Str`
- create PARI object (x mod y) — `Mod`(x, y)
- make x a polynomial of v — `Pol`$(x, \{v\})$
- as `Pol`/`Vec`, starting with constant term — `Polrev`, `Vecrev`
- make x a power series of v — `Ser`$(x, \{v\})$
- string from bytes / from format+args — `Strchr`, `Strprintf`
- convert x to simplest possible type — `simplify`(x)
- object x with precision n — `precision`(x, n)

Conjugates and Lifts

- conjugate of a number x — `conj`(x)
- conjugate vector of algebraic number x — `conjvec`(x)
- norm of x, product with conjugate — `norm`(x)
- square of L^2 norm of vector x — `norml2`(x)
- lift of x from Mods — `lift`, `centerlift`(x)

Lists, Sets & Sorting

- sort x by k-th component — `vecsort`$(x, \{k\}, \{fl = 0\})$
- min. m of x ($m = x[i]$), max. — `vecmin`$(x, \{\&i\})$, `vecmax`
- does y belong to x, sorted wrt. f — `vecsearch`$(x, y, \{f\})$

Sets (= row vector of strings with strictly increasing entries)

- intersection of sets x and y — `setintersect`(x, y)
- set of elements in x not belonging to y — `setminus`(x, y)
- union of sets x and y — `setunion`(x, y)
- does y belong to the set x — `setsearch`$(x, y, \{flag\})$
- is x a set ? — `setisset`(x)

Lists. create empty list: $L =$ List()

- append x to list L — `listput`$(L, x, \{i\})$
- remove i-th component from list L — `listpop`$(L, \{i\})$
- insert x in list L at position i — `listinsert`(L, x, i)
- sort the list L in place — `listsort`$(L, \{flag\})$

Programming

Functions and closures

`fun(vars) = my(local vars);` *seq*
`fun = (vars) -> my(local vars);` *seq*

Control Statements (X: formal parameter in expression *seq*)

- eval. *seq* for $a \le X \le b$ — `for`$(X = a, b, seq)$
- eval. *seq* for X dividing n — `fordiv`(n, X, seq)
- eval. *seq* for primes $a \le X \le b$ — `forprime`$(X = a, b, seq)$
- eval. *seq* for $a \le X \le b$ stepping s — `forstep`$(X = a, b, s, seq)$
- multivariable `for` — `forvec`$(X = v, seq)$
- loop over partitions of n — `forpart`$(p = n\ seq)$
- loop over vectors v, $q(v) \le B$, $q > 0$ — `forqfvec`(v, q, b, seq)
- loop over subgrps H of abelian grp G — `forsubgroup`$(H = G)$
- evaluate *seq* until $a \ne 0$ — `until`(a, seq)
- while $a \ne 0$, evaluate *seq* — `while`(a, seq)
- exit n innermost enclosing loops — `break`$(\{n\})$
- start new iteration of n-th enclosing loop — `next`$(\{n\})$
- return x from current subroutine — `return`$(\{x\})$
- raise an exception — `error()`
- if $a \ne 0$, evaluate seq_1, else seq_2 — `if`$(a, \{seq_1\}, \{seq_2\})$
- try seq_1, evaluate seq_2 on error — `iferr`(seq_1, E, seq_2)
- select from v according to f — `select`(f, v)
- apply f to all entries in v — `apply`(f, v)

Input/Output

- print with/without `\n`, TeX format — `print`, `print1`, `printtex`
- formatted printing — `printf()`
- write *args* to file — `write`, `write1`, `writetex`$(file, args)$
- write x in binary format — `writebin`$(file, x)$
- read file into GP — `read`$(\{file\})$
- read file, return as vector of lines — `readvec`$(\{file\})$
- read a string from keyboard — `input()`

Interface with User and System

- allocates a new stack of s bytes — `allocatemem`$(\{s\})$
- alias *old* to *new* — `alias`(new, old)
- install function from library — `install`$(f, code, \{gpf\}, \{lib\})$
- execute system command a — `system`(a)
- as above, feed result to GP — `extern`(a)
- as above, return GP string — `externstr`(a)
- get `$VAR` from environment — `getenv("VAR")`
- measure time in ms. — `gettime()`
- timeout command after s seconds — `alarm`$(s, expr)$

Iterations, Sums & Products

- numerical integration — `intnum`$(X = a, b, expr, \{flag\})$
- sum *expr* over divisors of n — `sumdiv`$(n, X, expr)$
- `sumdiv`, with *expr* multiplicative — `sumdivmult`$(n, X, expr)$
- sum $X = a$ to $X = b$, initialized at x — `sum`$(X = a, b, expr, \{x\})$
- sum of series *expr* — `suminf`$(X = a, expr)$
- sum of alternating/positive series — `sumalt`, `sumpos`
- sum of series using `intnum` — `sumnum`
- product $a \le X \le b$, initialized at x — `prod`$(X = a, b, expr, \{x\})$
- product over primes $a \le X \le b$ — `prodeuler`$(X = a, b, expr)$
- infinite product $a \le X \le \infty$ — `prodinf`$(X = a, expr)$
- real root of *expr* between a and b — `solve`$(X = a, b, expr)$

Random Numbers

- random integer/prime in $[0, N[$ — `random`(N), `randomprime`
- get/set random seed — `getrand`, `setrand`(s)

Vectors & Matrices

dimensions of matrix x	`matsize(x)`
concatenation of x and y	`concat(x, {y})`
extract components of x	`vecextract(x, y, {z})`
transpose of vector or matrix x	`mattranspose(x)` or `x~`
adjoint of the matrix x	`matadjoint(x)`
eigenvectors/values of matrix x	`mateigen(x)`
characteristic/minimal polynomial of x	`charpoly(x)`, `minpoly`
trace/determinant of matrix x	`trace(x)`, `matdet`
Frobenius form of x	`matfrobenius(x)`
QR decomposition	`matqr(x)`

Constructors & Special Matrices

row vec. of *expr* eval'ed at $1 \le i \le n$	`vector(n, {i}, {expr})`
col. vec. of *expr* eval'ed at $1 \le i \le n$	`vectorv(n, {i}, {expr})`
matrix $1 \le i \le m$, $1 \le j \le n$	`matrix(m, n, {i}, {j}, {expr})`
define matrix by blocks	`matconcat(B)`
diagonal matrix with diagonal x	`matdiagonal(x)`
$n \times n$ identity matrix	`matid(n)`
Hessenberg form of square matrix x	`mathess(x)`
$n \times n$ Hilbert matrix $H_{ij} = (i+j-1)^{-1}$	`mathilbert(n)`
companion matrix to polynomial x	`matcompanion(x)`
Sylvester matrix of x	`polsylvestermatrix(x)`

Gaussian elimination

kernel of matrix x	`matker(x, {flag})`
intersection of column spaces of x and y	`matintersect(x, y)`
solve $M * X = B$ (M invertible)	`matsolve(M, B)`
as solve, modulo D (col. vector)	`matsolvemod(M, D, B)`
one sol of $M * X = B$	`matinverseimage(M, B)`
basis for image of matrix x	`matimage(x)`
supplement columns of x to get basis	`matsupplement(x)`
rows, cols to extract invertible matrix	`matindexrank(x)`
rank of the matrix x	`matrank(x)`

Lattices & Quadratic Forms

upper triangular Hermite Normal Form	`mathnf(x)`
HNF of x where d is a multiple of $\det(x)$	`mathnfmod(x, d)`
elementary divisors of x	`matsnf(x)`
LLL-algorithm applied to columns of x	`qflll(x, {flag})`
like `qflll`, x is Gram matrix of lattice	`qflllgram(x, {flag})`
LLL-reduced basis for kernel of x	`matkerint(x)`
$\mathbf{Z}$-lattice $\longleftrightarrow$ $\mathbf{Q}$-vector space	`matrixqz(x, p)`
signature of quad form ${}^t y * x * y$	`qfsign(x)`
decomp into squares of ${}^t y * x * y$	`qfgaussred(x)`
eigenvals/eigenvecs for real symmetric x	`qfjacobi(x)`
find up to m sols of ${}^t y * x * y \le b$	`qfminim(x, b, m)`
perfection rank of x	`qfperfection(x)`
v, $v[i]$:=number of sols of ${}^t y * x * y = i$	`qfrep(x, B, {flag})`
automorphism group of q	`qfauto(q)`
find isomorphism between q and Q	`qfisom(q, Q)`

Formal & p-adic Series

truncate power series or p-adic number	`truncate(x)`
valuation of x at p	`valuation(x, p)`

Dirichlet and Power Series

Taylor expansion around 0 of f w.r.t. x	`taylor(f, x)`
$\sum a_k b_k t^k$ from $\sum a_k t^k$ and $\sum b_k t^k$	`serconvol(a, b)`
$f = \sum a_k t^k$ from $\sum (a_k/k!) t^k$	`serlaplace(f)`
reverse power series F so $F(f(x)) = x$	`serreverse(f)`
Dirichlet series multiplication / division	`dirmul`, `dirdiv(x, y)`
Dirichlet Euler product (b terms)	`direuler(p=a, b, expr)`

PARI-GP Reference Card

(PARI-GP version 2.6.1)

Polynomials & Rational Functions

degree of f	`poldegree(f)`
coeff. of degree n of f, leading coeff.	`polcoeff(f, n)`, `pollead`
gcd of coefficients of f	`content(f)`
replace x by y	`subst(f, x, y)`
evaluate f replacing vars by their value	`eval(f)`
replace polynomial expr. $T(x)$ by y in f	`substpol(f, T, y)`
replace $x_1, \ldots, x_n$ by $y_1, \ldots, y_n$ in f	`substvec(f, x, y)`
discriminant of polynomial f	`poldisc(f)`
resultant $R = \mathrm{Res}_v(f, g)$	`polresultant(f, g, {v})`
$[u, v, R]$, $xu + yv = \mathrm{Res}_v(f, g)$	`polresultantext(x, y, {v})`
derivative of f w.r.t. x	`deriv(f, {x})`
formal integral of f w.r.t. x	`intformal(f, {x})`
formal sum of f w.r.t. x	`sumformal(f, {x})`
reciprocal poly $x^{\deg f} f(1/x)$	`polrecip(f)`
interpol. pol. eval. at a	`polinterpolate(X, {Y}, {a}, {&e})`
initialize t for Thue equation solver	`thueinit(f)`
solve Thue equation $f(x, y) = a$	`thue(t, a, {sol})`

Roots and Factorization

number of real roots of f, $a < x \le b$	`polsturm(f, {a}, {b})`
complex roots of f	`polroots(f)`
symmetric powers of roots of f up to n	`polsym(f, n)`
factor f	`factor(f, {lim})`
factor f mod p / roots	`factormod(f, p)`, `polrootsmod`
factor f over $\mathbf{F}_{p^a}$ / roots	`factorff(f, p, a)`, `polrootsff`
factor f over $\mathbf{Q}_p$ / roots	`factorpadic(f, p, r)`, `polrootspadic`
find irreducible $T \in \mathbf{F}_p[x]$, $\deg T = n$	`ffinit(p, n, {x})`
$\#\{$monic irred. $T \in \mathbf{F}_q[x], \deg T = n\}$	`ffnbirred(q, n)`
p-adic root of f cong. to a mod p	`padicappr(f, a)`
Newton polygon of f for prime p	`newtonpoly(f, p)`
extensions of $\mathbf{Q}_p$ of degree N	`padicfields(p, N)`

Special Polynomials

n-th cyclotomic polynomial in var. v	`polcyclo(n, {v})`
d-th degree subfield of $\mathbf{Q}(\zeta_n)$	`polsubcyclo(n, d, {v})`
P_n, T_n/U_n, H_n	`pollegendre`, `polchebyshev`, `polhermite`

Transcendental and p-adic Functions

real, imaginary part of x	`real(x)`, `imag(x)`
absolute value, argument of x	`abs(x)`, `arg(x)`
square/nth root of x	`sqrt(x)`, `sqrtn(x, n, {&z})`
trig functions	`sin`, `cos`, `tan`, `cotan`
inverse trig functions	`asin`, `acos`, `atan`
hyperbolic functions	`sinh`, `cosh`, `tanh`
inverse hyperbolic functions	`asinh`, `acosh`, `atanh`
exponential / natural log of x	`exp`, `log`
Euler Γ function, $\log \Gamma$, Γ'/Γ	`gamma`, `lngamma`, `psi`
incomplete gamma function ($y = \Gamma(s)$)	`incgam(s, x, {y})`
exponential integral $\int_x^\infty e^{-t}/t\, dt$	`eint1(x)`
error function $2/\sqrt{\pi} \int_x^\infty e^{-t^2} dt$	`erfc(x)`
dilogarithm of x	`dilog(x)`
m-th polylogarithm of x	`polylog(m, x, {flag})`
U-confluent hypergeometric function	`hyperu(a, b, u)`
Bessel $J_n(x)$, $J_{n+1/2}(x)$	`besselj(n, x)`, `besseljh(n, x)`
Bessel I_ν, K_ν, H^1_ν, H^2_ν, N_ν	`(bessel)i, k, h1, h2, n`
Lambert W: x s.t. $xe^x = y$	`lambertw(y)`
Teichmuller character of p-adic x	`teichmuller(x)`

Elementary Arithmetic Functions

vector of binary digits of $\lvert x \rvert$	`binary(x)`
bit number n of integer x	`bittest(x, n)`
Hamming weight of integer x	`hammingweight(x)`
ceiling/floor/fractional part	`ceil`, `floor`, `frac`
round x to nearest integer	`round(x, {&e})`
truncate x	`truncate(x, {&e})`
gcd/LCM of x and y	`gcd(x, y)`, `lcm(x, y)`
gcd of entries of a vector/matrix	`content(x)`

Primes and Factorization

add primes in v to prime table	`addprimes(v)`
Chebyshev $\pi(x)$, n-th prime p_n	`primepi(x)`, `prime(n)`
vector of first n primes	`primes(n)`
smallest prime $\ge x$	`nextprime(x)`
largest prime $\le x$	`precprime(x)`
factorization of x	`factor(x, {lim})`
$n = df^2$, d squarefree/fundamental	`core(n, {fl})`, `coredisc`
recover x from its factorization	`factorback(f, {e})`

Divisors

number of prime divisors $\omega(n)$ / $\Omega(n)$	`omega(n)`, `bigomega`
divisors of n / number of divisors $\tau(n)$	`divisors(n)`, `numdiv`
sum of (k-th powers of) divisors of n	`sigma(n, {k})`

Special Functions and Numbers

binomial coefficient $\binom{x}{y}$	`binomial(x, y)`
Bernoulli number B_n as real/rational	`bernreal(n)`, `bernfrac`
Bernoulli polynomial $B_n(x)$	`bernpol(n, {x})`
n-th Fibonacci number	`fibonacci(n)`
Stirling numbers $s(n, k)$ and $S(n, k)$	`stirling(n, k, {flag})`
number of partitions of n	`numbpart(n)`
Möbius μ-function	`moebius(x)`
Hilbert symbol of x and y (at p)	`hilbert(x, y, {p})`
Kronecker-Legendre symbol $(\frac{x}{y})$	`kronecker(x, y)`
Dedekind sum $s(h, k)$	`sumdedekind(h, k)`

Multiplicative groups $(\mathbf{Z}/N\mathbf{Z})^*$, $\mathbf{F}_q^*$

Euler ϕ-function	`eulerphi(x)`
multiplicative order of x (divides o)	`znorder(x, {o})`, `fforder`
primitive root mod q / `x.mod`	`znprimroot(q)`, `ffprimroot(x)`
structure of $(\mathbf{Z}/n\mathbf{Z})^*$	`znstar(n)`
discrete logarithm of x in base g	`znlog(x, g, {o})`, `fflog`

Miscellaneous

integer square / n-th root of x	`sqrtint(x)`, `sqrtnint(x, n)`
solve $z \equiv x$ and $z \equiv y$	`chinese(x, y)`
minimal u, v so $xu + yv = \gcd(x, y)$	`gcdext(x, y)`
continued fraction of x	`contfrac(x, {b}, {lmax})`
last convergent of continued fraction x	`contfracpnqn(x)`
rational approximation to x	`bestappr(x, k)`, `bestapprPade`

True-False Tests

is x the disc. of a quadratic field?	`isfundamental(x)`
is x a prime?	`isprime(x)`
is x a strong pseudo-prime?	`ispseudoprime(x)`
is x square-free?	`issquarefree(x)`
is x a square?	`issquare(x, {&n})`
is x a perfect power?	`ispower(x, {k}, {&n})`
is *pol* irreducible?	`polisirreducible(pol)`

Send comments and corrections to ⟨Karim.Belabas@math.u-bordeaux.fr⟩

PARI-GP Reference Card (2)

(PARI-GP version 2.6.1)

Elliptic Curves

Elliptic curve initially given by 5-tuple $v =$`[`a_1, a_2, a_3, a_4, a_6`]`.

Initialize *ell* struct	`E = ellinit(`$v, \{Domain\}$`)`

Points are `[x,y]`, the origin is `[0]`. Struct members accessed as `E.`*member*:

- All domains: `E.a1,a2,a3,a4,a6`, `b2,b4,b6,b8`, `c4,c6`, `disc`, `j`
- E defined over $\mathbf{R}$ or $\mathbf{C}$

x-coords. of points of order 2	`E.roots`
periods / quasi-periods	`E.omega,E.eta`
volume of complex lattice	`E.area`

- E defined over $\mathbf{Q}_p$

residual characteristic	`E.p`
If $\|j\|_p > 1$: Tate's $[u^2, u, q, [a, b]]$	`E.tate`

- E defined over $\mathbf{F}_q$

characteristic	`E.p`
$\#E(\mathbf{F}_q)$/cyclic structure/generators	`E.no`, `E.cyc`, `E.gen`

- E defined over $\mathbf{Q}$

generators of $E(\mathbf{Q})$ (require `elldata`)	`E.gen`
$[a_1, a_2, a_3, a_4, a_6]$ from j-invariant	`ellfromj(`j`)`
change curve E using $v =$`[`u, r, s, t`]`	`ellchangecurve(`E, v`)`
change point z using $v =$`[`u, r, s, t`]`	`ellchangepoint(`z, v`)`
add points $P + Q$ / $P - Q$	`elladd(`E, P, Q`)`, `ellsub`
negate point	`ellneg(`E, P`)`
compute $n \cdot z$	`ellmul(`E, z, n`)`
n-division polynomial $f_n(x)$	`elldivpol(`$E, n, \{x\}$`)`
check if z is on E	`ellisoncurve(`E, z`)`
order of torsion point z	`ellorder(`E, z`)`
y-coordinates of point(s) for x	`ellordinate(`E, x`)`
point $[\wp(z), \wp'(z)]$ corresp. to z	`ellztopoint(`E, z`)`
complex z such that $p = [\wp(z), \wp'(z)]$	`ellpointtoz(`E, p`)`

Curves over finite fields, Pairings

random point on E	`random(`E`)`
$\#E(\mathbf{F}_q)$	`ellcard(`E`)`
structure $\mathbf{Z}/d_1\mathbf{Z} \times \mathbf{Z}/d_2\mathbf{Z}$ of $E(\mathbf{F}_q)$	`ellgroup(`E`)`
Weil pairing of m-torsion pts x, y	`ellweilpairing(`E, x, y, m`)`
Tate pairing of x, y; x m-torsion	`elltatepairing(`E, x, y, m`)`
Discrete log, find n s.t. $P = [n]Q$	`elllog(`$E, P, Q, \{ord\}$`)`

Curves over Q and the L-function

canonical bilinear form taken at z_1, z_2	`ellbil(`E, z_1, z_2`)`
canonical height of z	`ellheight(`$E, z, \{flag\}$`)`
height regulator matrix for pts in x	`ellheightmatrix(`E, x`)`
cond, min mod, Tamagawa num `[`N, v, c`]`	`ellglobalred(`E`)`
reduction of $y^2 + Qy = P$ (genus 2)	`genus2red(`$Q, P, \{p\}$`)`
Kodaira type of p-fiber of E	`elllocalred(`E, p`)`
minimal model of $E/\mathbf{Q}$	`ellminimalmodel(`$E, \{$`&`$v\}$`)`
p-th coeff a_p of L-function, p prime	`ellap(`E, p`)`
k-th coeff a_k of L-function	`ellak(`E, k`)`
vector of first n a_k's in L-function	`ellan(`E, n`)`
$L(E, s)$	`elllseries(`E, s`)`
$L^{(r)}(E, 1)$	`ellL1(`E, r`)`
return a Heegner point on E of rank 1	`ellheegner(`E`)`
order of vanishing at 1	`ellanalyticrank(`$E, \{eps\}$`)`
root number for $L(E, .)$ at p	`ellrootno(`$E, \{p\}$`)`
torsion subgroup with generators	`elltors(`E`)`
modular parametrization of E	`elltaniyama(`E`)`

Elldata package, Cremona's database:

db code $\leftrightarrow$ [*conductor*, *class*, *index*]	`ellconvertname(`s`)`
generators of Mordell-Weil group	`ellgenerators(`E`)`
look up E in database	`ellidentify(`E`)`
all curves matching criterion	`ellsearch(`N`)`
loop over curves with cond. from a to b	`forell(`E, a, b, seq`)`

Elliptic & Modular Functions

$w = [\omega_1, \omega_2]$ or *ell* struct (`E.omega`), $\tau = \omega_1/\omega_2$.

arithmetic-geometric mean	`agm(`x, y`)`
elliptic j-function $1/q + 744 + \cdots$	`ellj(`x`)`
Weierstrass $\sigma/\wp/\zeta$ function	`ellsigma(`w, z`)`, `ellwp`, `ellzeta`
periods/quasi-periods	`ellperiods(`$E, \{flag\}$`)`, `elleta(`w`)`
$(2i\pi/\omega_2)^k E_k(\tau)$	`elleisnum(`$w, k, \{flag\}$`)`
modified Dedekind η func. $\prod(1 - q^n)$	`eta(`$x, \{flag\}$`)`
Jacobi sine theta function	`theta(`q, z`)`
k-th derivative at z=0 of `theta(`q, z`)`	`thetanullk(`q, k`)`
Weber's f functions	`weber(`$x, \{flag\}$`)`
Riemann's zeta $\zeta(s) = \sum n^{-s}$	`zeta(`s`)`

Binary Quadratic Forms

create $ax^2 + bxy + cy^2$ (distance d)	`Qfb(`$a, b, c, \{d\}$`)`
reduce x ($s = \sqrt{D}$, $l = \lfloor s \rfloor$)	`qfbred(`$x, \{flag\}, \{D\}, \{l\}, \{s\}$`)`
composition of forms	x`*`y or `qfbnucomp(`x, y, l`)`
n-th power of form	x`^`n or `qfbnupow(`x, n`)`
composition without reduction	`qfbcompraw(`x, y`)`
n-th power without reduction	`qfbpowraw(`x, n`)`
prime form of disc. x above prime p	`qfbprimeform(`x, p`)`
class number of disc. x	`qfbclassno(`x`)`
Hurwitz class number of disc. x	`qfbhclassno(`x`)`
Solve $Q(x, y) = p$ in integers, p prime	`qfbsolve(`Q, p`)`

Quadratic Fields

quadratic number $\omega = \sqrt{x}$ or $(1 + \sqrt{x})/2$	`quadgen(`x`)`
minimal polynomial of ω	`quadpoly(`x`)`
discriminant of $\mathbf{Q}(\sqrt{D})$	`quaddisc(`x`)`
regulator of real quadratic field	`quadregulator(`x`)`
fundamental unit in real $\mathbf{Q}(x)$	`quadunit(`x`)`
class group of $\mathbf{Q}(\sqrt{D})$	`quadclassunit(`$D, \{flag\}, \{t\}$`)`
Hilbert class field of $\mathbf{Q}(\sqrt{D})$	`quadhilbert(`$D, \{flag\}$`)`
ray class field modulo f of $\mathbf{Q}(\sqrt{D})$	`quadray(`$D, f, \{flag\}$`)`

General Number Fields: Initializations

A number field K is given by a monic irreducible $f \in \mathbf{Z}[X]$.

init number field structure *nf*	`nfinit(`$f, \{flag\}$`)`

nf members:

polynomial defining *nf*, $f(\theta) = 0$	*nf*`.pol`
number of real/complex places	*nf*`.r1/r2/sign`
discriminant of *nf*	*nf*`.disc`
T_2 matrix	*nf*`.t2`
vector of roots of f	*nf*`.roots`
integral basis of $\mathbf{Z}_K$ as powers of θ	*nf*`.zk`
different	*nf*`.diff`
codifferent	*nf*`.codiff`
index	*nf*`.index`
recompute *nf* using current precision	`nfnewprec(`*nf*`)`
init relative *rnf* given by $g = 0$ over K	`rnfinit(`nf, g`)`
init*bnf* structure	`bnfinit(`$f, \{flag\}$`)`

bnf members: same as *nf*, plus

underlying *nf*	*bnf*`.nf`
classgroup	*bnf*`.clgp`
regulator	*bnf*`.reg`
fundamental units	*bnf*`.fu`
torsion units	*bnf*`.tu`
compute a *bnf* from small *bnf*	`bnfinit(`*sbnf*`)`
add S-class group and units, yield *bnf* s	`bnfsunit(`nf, S`)`
init class field structure *bnr*	`bnrinit(`$bnf, m, \{flag\}$`)`

bnr members: same as *bnf*, plus

underlying *bnf*	*bnr*`.bnf`
big ideal structure	*bnr*`.bid`
modulus	*bnr*`.mod`
structure of $(\mathbf{Z}_K/m)^*$	*bnr*`.zkst`

Basic Number Field Arithmetic (nf)

Elements are `t_INT`, `t_FRAC`, `t_POL`, `t_POLMOD`, or `t_COL` (on integral basis *nf*`.zk`). Basic operations (prefix `nfelt`): (`nfelt`)`add`, `mul`, `pow`, `div`, `diveuc`, `mod`, `divrem`, `val`, `trace`, `norm`

express x on integer basis	`nfalgtobasis(`nf, x`)`
express element x as a polmod	`nfbasistoalg(`nf, x`)`
reverse polmod $a = A(X)$ mod $T(X)$	`modreverse(`a`)`
integral basis of field def. by $f = 0$	`nfbasis(`f`)`
field discriminant of field $f = 0$	`nfdisc(`f`)`
smallest poly defining $f = 0$ (slow)	`polredabs(`$f, \{flag\}$`)`
small poly defining $f = 0$ (fast)	`polredbest(`$f, \{flag\}$`)`
are fields $f = 0$ and $g = 0$ isomorphic?	`nfisisom(`f, g`)`
is field $f = 0$ a subfield of $g = 0$?	`nfisincl(`f, g`)`
compositum of $f = 0$, $g = 0$	`polcompositum(`$f, g, \{flag\}$`)`
subfields (of degree d) of *nf*	`nfsubfields(`$nf, \{d\}$`)`
roots of unity in *nf*	`nfrootsof1(`*nf*`)`
roots of g belonging to *nf*	`nfroots(`$\{nf\}, g$`)`
factor g in *nf*	`nffactor(`nf, g`)`
factor g mod prime *pr* in *nf*	`nffactormod(`nf, g, pr`)`
conjugates of a root θ of *nf*	`nfgaloisconj(`$nf, \{flag\}$`)`
apply Galois automorphism s to x	`nfgaloisapply(`nf, s, x`)`
quadratic Hilbert symbol (at p)	`nfhilbert(`$nf, a, b, \{p\}$`)`

Linear and algebraic relations

poly of degree $\le k$ with root $x \in \mathbf{C}$	`algdep(`x, k`)`
alg. dep. with pol. coeffs for series s	`seralgdep(`s, x, y`)`
small linear rel. on coords of vector x	`lindep(`x`)`

Dedekind Zeta Function ζ_K, Hecke L series

ζ_K as Dirichlet series, $N(I) < b$	`dirzetak(`nf, b`)`
init *nfz* for field $f = 0$	`zetakinit(`f`)`
compute $\zeta_K(s)$	`zetak(`$nfz, s, \{flag\}$`)`
Artin root number of K	`bnrrootnumber(`$bnr, chi, \{flag\}$`)`
$L(1, \chi)$, for all χ trivial on H	`bnrL1(`$bnr, \{H\}, \{flag\}$`)`

Class Groups & Units (bnf, bnr)

$a_1, \{a_2\}, \{a_3\}$ usually *bnr*, *subgp* or *bnf*, *module*, $\{subgp\}$

remove GRH assumption from *bnf*	`bnfcertify(`*bnf*`)`
expo. of ideal x on class gp	`bnfisprincipal(`$bnf, x, \{flag\}$`)`
expo. of ideal x on ray class gp	`bnrisprincipal(`$bnr, x, \{flag\}$`)`
expo. of x on fund. units	`bnfisunit(`bnf, x`)`
as above for S-units	`bnfissunit(`$bnfs, x$`)`
signs of real embeddings of *bnf*`.fu`	`bnfsignunit(`*bnf*`)`
narrow class group	`bnfnarrow(`*bnf*`)`

PARI-GP Reference Card (2)

(PARI-GP version 2.6.1)

Class Field Theory

ray class number for mod. m	bnrclassno(bnf, m)
discriminant of class field ext	bnrdisc(a1, {a2}, {a3})
ray class numbers, l list of mods	bnrclassnolist(bnf, l)
discriminants of class fields	bnrdisclist(bnf, l, {arch}, {flag})
decode output from bnrdisclist	bnfdecodemodule(nf, fa)
is modulus the conductor?	bnrisconductor(a1, {a2}, {a3})
conductor of character chi	bnrconductorofchar(bnr, chi)
conductor of extension	bnrconductor(a1, {a2}, {a3}, {flag})
conductor of extension def. by g	rnfconductor(bnf, g)
Artin group of ext. def'd by g	rnfnormgroup(bnr, g)
subgroups of bnr, index <= b	subgrouplist(bnr, b, {flag})
rel. eq. for class field def'd by sub	rnfkummer(bnr, sub, {d})
same, using Stark units (real field)	bnrstark(bnr, sub, {flag})

Ideals: elements, primes, or matrix of generators in HNF

is id an ideal in nf ?	nfisideal(nf, id)
is x principal in bnf ?	bnfisprincipal(bnf, x)
give $[a, b]$, s.t. $a\mathbf{Z}_K + b\mathbf{Z}_K = x$	idealtwoelt(nf, x, {a})
put ideal a ($a\mathbf{Z}_K + b\mathbf{Z}_K$) in HNF form	idealhnf(nf, a, {b})
norm of ideal x	idealnorm(nf, x)
minimum of ideal x (direction v)	idealmin(nf, x, v)
LLL-reduce the ideal x (direction v)	idealred(nf, x, {v})

Ideal Operations

add ideals x and y	idealadd(nf, x, y)
multiply ideals x and y	idealmul(nf, x, y, {flag})
intersection of ideals x and y	idealintersect(nf, x, y, {flag})
n-th power of ideal x	idealpow(nf, x, n, {flag})
inverse of ideal x	idealinv(nf, x)
divide ideal x by y	idealdiv(nf, x, y, {flag})
Find $(a, b) \in x \times y$, $a + b = 1$	idealaddtoone(nf, x, {y})
coprime integral A, B such that $x = A/B$	idealnumden(nf, x)

Primes and Multiplicative Structure

factor ideal x in nf	idealfactor(nf, x)
expand ideal factorization in nf	idealfactorback(nf, f, e)
decomposition of prime p in nf	idealprimedec(nf, p)
valuation of x at prime ideal pr	idealval(nf, x, pr)
weak approximation theorem in nf	idealchinese(nf, x, y)
give bid =structure of $(\mathbf{Z}_K/id)^*$	idealstar(nf, id, {flag})
discrete log of x in $(\mathbf{Z}_K/bid)^*$	ideallog(nf, x, bid)
idealstar of all ideals of norm $\le b$	ideallist(nf, b, {flag})
add Archimedean places	ideallistarch(nf, b, {ar}, {flag})
init prmod structure	nfmodprinit(nf, pr)
kernel of matrix M in $(\mathbf{Z}_K/pr)^*$	nfkermodpr(nf, M, prmod)
solve $Mx = B$ in $(\mathbf{Z}_K/pr)^*$	nfsolvemodpr(nf, M, B, prmod)

Galois theory over Q

Galois group of field $\mathbf{Q}[x]/(f)$	polgalois(f)
initializes a Galois group structure G	galoisinit(pol, {den})
action of p in nfgaloisconj form	galoispermtopol(G, {p})
identify as abstract group	galoisidentify(G)
export a group for GAP/MAGMA	galoisexport(G, {flag})
subgroups of the Galois group G	galoissubgroups(G)
is subgroup H normal?	galoisisnormal(G, H)
subfields from subgroups	galoissubfields(G, {flag}, {v})
fixed field	galoisfixedfield(G, perm, {flag}, {v})
Frobenius at maximal ideal P	idealfrobenius(nf, G, P)
ramification groups at P	idealramgroups(nf, G, P)
is G abelian?	galoisisabelian(G, {flag})
abelian number fields/$\mathbf{Q}$	galoissubcyclo(N,H,{flag},{v})
query the galpol package	galoisgetpol(a,b,{s})

Relative Number Fields (rnf)

Extension L/K is defined by $T \in K[x]$.

absolute equation of L	rnfequation(nf, T, {flag})
is L/K abelian?	rnfisabelian(nf, T)
relative nfalgtobasis	rnfalgtobasis(rnf, x)
relative nfbasistoalg	rnfbasistoalg(rnf, x)
relative idealhnf	rnfidealhnf(rnf, x)
relative idealmul	rnfidealmul(rnf, x, y)
relative idealtwoelt	rnfidealtwoelt(rnf, x)

Lifts and Push-downs

absolute → relative repres. for x	rnfeltabstorel(rnf, x)
relative → absolute repres. for x	rnfeltreltoabs(rnf, x)
lift x to the relative field	rnfeltup(rnf, x)
push x down to the base field	rnfeltdown(rnf, x)

idem for x ideal: (rnfideal)reltoabs, abstorel, up, down

Norms

absolute norm of ideal x	rnfidealnormabs(rnf, x)
relative norm of ideal x	rnfidealnormrel(rnf, x)
solutions of $N_{K/\mathbf{Q}}(y) = x \in \mathbf{Z}$	bnfisintnorm(bnf, x)
is $x \in \mathbf{Q}$ a norm from K?	bnfisnorm(bnf, x, {flag})
initialize T for norm eq. solver	rnfisnorminit(K, pol, {flag})
is $a \in K$ a norm from L?	rnfisnorm(T, a, {flag})

Maximal order $\mathbf{Z}_L$ as a $\mathbf{Z}_K$-module

relative polred	rnfpolred(nf, T)
relative polredabs	rnfpolredabs(nf, T)
characteristic poly. of a mod T	rnfcharpoly(nf, T, a, {v})
relative Dedekind criterion, prime pr	rnfdedekind(nf, T, pr)
discriminant of relative extension	rnfdisc(nf, T)
pseudo-basis of $\mathbf{Z}_L$	rnfpseudobasis(nf, T)

General $\mathbf{Z}_K$-modules: $M = [\text{matrix}, \text{vec. of ideals}] \subset L$

relative HNF / SNF	nfhnf(nf, M), nfsnf
reduced basis for M	rnflllgram(nf, T, M)
determinant of pseudo-matrix M	rnfdet(nf, M)
Steinitz class of M	rnfsteinitz(nf, M)
$\mathbf{Z}_K$-basis of M if $\mathbf{Z}_K$-free, or 0	rnfhnfbasis(bnf, M)
n-basis of M, or $(n + 1)$-generating set	rnfbasis(bnf, M)
is M a free $\mathbf{Z}_K$-module?	rnfisfree(bnf, M)

Graphic Functions

crude graph of $expr$ between a and b	plot(X = a, b, expr)

High-resolution plot (immediate plot)

plot $expr$ between a and b	ploth(X = a, b, expr, {flag}, {n})
plot points given by lists lx, ly	plothraw(lx, ly, {flag})
terminal dimensions	plothsizes()

Rectwindow functions

init window w, with size x,y	plotinit(w, x, y)
erase window w	plotkill(w)
copy w to w_2 with offset (dx, dy)	plotcopy(w, w2, dx, dy)
clips contents of w	plotclip(w)
scale coordinates in w	plotscale(w, x1, x2, y1, y2)
ploth in w	plotrecth(w, X = a, b, expr, {flag}, {n})
plothraw in w	plotrecthraw(w, data, {flag})
draw window w_1 at (x_1, y_1), ...	plotdraw([[w1, x1, y1], ...])

Low-level Rectwindow Functions

set current drawing color in w to c	plotcolor(w, c)
current position of cursor in w	plotcursor(w)
write s at cursor's position	plotstring(w, s)
move cursor to (x, y)	plotmove(w, x, y)
move cursor to $(x + dx, y + dy)$	plotrmove(w, dx, dy)
draw a box to (x_2, y_2)	plotbox(w, x2, y2)
draw a box to $(x + dx, y + dy)$	plotrbox(w, dx, dy)
draw polygon	plotlines(w, lx, ly, {flag})
draw points	plotpoints(w, lx, ly)
draw line to $(x + dx, y + dy)$	plotrline(w, dx, dy)
draw point $(x + dx, y + dy)$	plotrpoint(w, dx, dy)
draw point $(x + dx, y + dy)$	plotrpoint(w, dx, dy)

Postscript Functions

as ploth	psploth(X = a, b, expr, {flag}, {n})
as plothraw	psplothraw(lx, ly, {flag})
as plotdraw	psdraw([[w1, x1, y1], ...])

Based on an earlier version by Joseph H. Silverman

Send comments and corrections to ⟨Karim.Belabas@math.u-bordeaux.fr⟩