PARI-GP Reference Card

(PARI-GP version 2.3.0)

Note: optional arguments are surrounded by braces {}.

Starting & Stopping GP

to enter GP, just type its name:	`gp`
to exit GP, type	`\q` or `quit`

Help

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

Input/Output & Defaults

output previous line, the lines before	`%`, `%‘`, `%‘‘`, etc.
output from line n	`%`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	`\\ ...`
set default d to *val*	`default(`$\{d\}, \{val\}, flag$`)`
mimic behaviour of GP 1.39	`default(compatible,3)`

Metacommands

toggle timer on/off	`#`
print time for last result	`##`
print %n in raw format	`\a` n
print %n in pretty format	`\b` n
print defaults	`\d`
set debug level to n	`\g` n
set memory debug level to n	`\gm` n
enable/disable logfile	`\l` {*filename*}
print %n in pretty matrix format	`\m`
set output mode (raw, default, prettyprint)	`\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*
write %n to file	`\w` n *filename*

GP Within Emacs

to enter GP from within Emacs:	`M-x gp`, `C-u M-x gp`
word completion	⟨TAB⟩
help menu window	`M-\c`
describe function	`M-?`
display TEX'd PARI manual	`M-x gpman`
set prompt string	`M-\p`
break line at column 100, insert \	`M-\\`
PARI metacommand *letter*	`M-\`*letter*

Reserved Variable Names

$\pi = 3.14159\cdots$	`Pi`
Euler's constant $= .57721\cdots$	`Euler`
square root of -1	`I`
big-oh notation	`O`

PARI Types & Input Formats

`t_INT`. Integers	$\pm n$
`t_REAL`. Real Numbers	$\pm n.ddd$
`t_INTMOD`. Integers modulo m	`Mod(`n, m`)`
`t_FRAC`. Rational Numbers	n/m
`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	`"aaa"`

Standard 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, n`)`
comparison operators	`<=`, `<`, `>=`, `>`, `==`, `!=`
boolean operators (or, and, not)	`||`, `&&`, `!`
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`'`

Conversions

Change Objects

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 above, starting with constant term	`Polrev(`$x, \{v\}$`)`
make x a power series of v	`Ser(`$x, \{v\}$`)`
PARI type of object x	`type(`$x, \{t\}$`)`
object x with precision n	`prec(`$x, \{n\}$`)`
evaluate f replacing vars by their value	`eval(`f`)`

Select Pieces of an Object

length of x	`#`x or `length(`x`)`
n-th component of x	`component(`x, n`)`
n-th component of vector/list x	x`[`n`]`
(m, n)-th component of matrix x	x`[`m, n`]`
row m or column n of matrix x	x`[`m`,]`, x`[,`n`]`
numerator of x	`numerator(`x`)`
lowest denominator of x	`denominator(`x`)`

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`)`

Random Numbers

random integer between 0 and $N-1$	`random(`$\{N\}$`)`
get random seed	`getrand()`
set random seed to s	`setrand(`s`)`

Lists, Sets & Sorting

sort x by kth component	`vecsort(`$x, \{k\}, \{fl = 0\}$`)`

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`)`
look if y belongs to the set x	`setsearch(`$x, y, flag$`)`

Lists

create empty list of maximal length n	`listcreate(`n`)`
delete all components of list l	`listkill(`l`)`
append x to list l	`listput(`$l, x, \{i\}$`)`
insert x in list l at position i	`listinsert(`l, x, i`)`
sort the list l	`listsort(`$l, flag$`)`

Programming & User Functions

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`)`
if $a \ne 0$, evaluate seq_1, else seq_2	`if(`$a, \{seq_1\}, \{seq_2\}$`)`
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 nth enclosing loop	`next(`$\{n\}$`)`
return x from current subroutine	`return(`x`)`
error recovery (try seq_1)	`trap(`$\{err\}, \{seq_2\}, \{seq_1\}$`)`

Input/Output

prettyprint args with/without newline	`printp()`, `printp1()`
print args with/without newline	`print()`, `print1()`
read a string from keyboard	`input()`
reorder priority of variables x, y, z	`reorder(`$\{[x, y, z]\}$`)`
output *args* in TEX format	`printtex(`*args*`)`
write *args* to file	`write`, `write1`, `writetex(`*file*, *args*`)`
read file into GP	`read(`$\{file\}$`)`

Interface with User and System

allocates a new stack of s bytes	`allocatemem(`$\{s\}$`)`
execute system command a	`system(`a`)`
as above, feed result to GP	`extern(`a`)`
install function from library	`install(`$f, code, \{gpf\}, \{lib\}$`)`
alias *old* to *new*	`alias(`*new*, *old*`)`
new name of function f in GP 2.0	`whatnow(`f`)`

User Defined Functions

`name(formal vars) = local(local vars);` *seq*

`struct.member =` *seq*

kill value of variable or function x	`kill(`x`)`
declare global variables	`global(`$x, \ldots$`)`

Iterations, Sums & Products

numerical integration	`intnum(`X `=` $a, b, expr, flag$`)`
sum *expr* over divisors of n	`sumdiv(`$n, X, expr$`)`
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`
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 `=` $a, b, expr$`)`

PARI-GP Reference Card

(PARI-GP version 2.3.0)

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 of matrix x	`mateigen(x)`
characteristic polynomial of x	`charpoly(x,{v},flag)`
minimal polynomial of x	`minpoly(x,{v})`
trace of matrix x	`trace(x)`

Constructors & Special Matrices

row vec. of *expr* eval'ed at $1 \leq i \leq n$	`vector(n,{i},{expr})`
col. vec. of *expr* eval'ed at $1 \leq i \leq n$	`vectorv(n,{i},{expr})`
matrix $1 \leq i \leq m$, $1 \leq j \leq n$	`matrix(m,n,{i},{j},{expr})`
diagonal matrix whose diag. is 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)`
$n \times n$ Pascal triangle $P_{ij} = \binom{i}{j}$	`matpascal(n-1)`
companion matrix to polynomial x	`matcompanion(x)`

Gaussian elimination

determinant of matrix x	`matdet(x,flag)`
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)`
find up to m sols of ${}^t y * x * y \leq b$	`qfminim(x,b,m)`
v, $v[i]$:=number of sols of ${}^t y * x * y = i$	`qfrep(x,B,flag)`
eigenvals/eigenvecs for real symmetric x	`qfjacobi(x)`

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(x,y)`
$f = \sum a_k * t^k$ 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)`

p-adic Functions

Teichmuller character of x	`teichmuller(x)`
Newton polygon of f for prime p	`newtonpoly(f,p)`

Polynomials & Rational Functions

degree of f	`poldegree(f)`
coefficient of degree n of f	`polcoeff(f,n)`
round coeffs of f to nearest integer	`round(f,{&e})`
gcd of coefficients of f	`content(f)`
replace x by y in f	`subst(f,x,y)`
discriminant of polynomial f	`poldisc(f)`
resultant of f and g	`polresultant(f,g,flag)`
as above, give $[u,v,d]$, $xu + yv = d$	`bezoutres(x,y)`
derivative of f w.r.t. x	`deriv(f,x)`
formal integral of f w.r.t. x	`intformal(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, $a < x \leq b$	`polsturm(f,{a},{b})`
complex roots of f	`polroots(f)`
symmetric powers of roots of f up to n	`polsym(f,n)`
roots of f mod p	`polrootsmod(f,p,flag)`
factor f	`factor(f,{lim})`
factorization of f mod p	`factormod(f,p,flag)`
factorization of f over $\mathbf{F}_{p^a}$	`factorff(f,p,a)`
p-adic fact. of f to prec. r	`factorpadic(f,p,r,flag)`
p-adic roots of f to prec. r	`polrootspadic(f,p,r)`
p-adic root of f cong. to a mod p	`padicappr(f,a)`
Newton polygon of f for prime p	`newtonpoly(f,p)`

Special Polynomials

nth cyclotomic polynomial in var. v	`polcyclo(n,{v})`
d-th degree subfield of $\mathbf{Q}(\zeta_n)$	`polsubcyclo(n,d,{v})`
n-th Legendre polynomial	`pollegendre(n)`
n-th Tchebicheff polynomial	`poltchebi(n)`
Zagier's polynomial of index n,m	`polzagier(n,m)`

Transcendental 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 of x	`exp(x)`
natural log of x	`ln(x)` or `log(x)`
gamma function $\Gamma(x) = \int_0^\infty e^{-t} t^{x-1} dt$	`gamma(x)`
logarithm of gamma function	`lngamma(x)`
$\psi(x) = \Gamma'(x)/\Gamma(x)$	`psi(x)`
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)`
mth polylogarithm of x	`polylog(m,x,flag)`
U-confluent hypergeometric function	`hyperu(a,b,u)`
J-Bessel function $J_{n+1/2}(x)$	`besseljh(n,x)`
K-Bessel function of index nu	`besselk(nu,x)`

Elementary Arithmetic Functions

vector of binary digits of $\lvert x \rvert$	`binary(x)`
give bit number n of integer x	`bittest(x,n)`
ceiling of x	`ceil(x)`
floor of x	`floor(x)`
fractional part of x	`frac(x)`
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 the prime table	`addprimes(v)`
the nth prime	`prime(n)`
vector of first n primes	`primes(n)`
smallest prime $\geq x$	`nextprime(x)`
largest prime $\leq x$	`precprime(x)`
factorization of x	`factor(x,{lim})`
reconstruct x from its factorization	`factorback(fa,{nf})`

Divisors

number of distinct prime divisors	`omega(x)`
number of prime divisors with mult	`bigomega(x)`
number of divisors of x	`numdiv(x)`
row vector of divisors of x	`divisors(x)`
sum of (k-th powers of) divisors of x	`sigma(x,{k})`

Special Functions and Numbers

binomial coefficient $\binom{x}{y}$	`binomial(x,y)`
Bernoulli number B_n as real	`bernreal(n)`
Bernoulli vector $B_0, B_2, \ldots, B_{2n}$	`bernvec(n)`
nth Fibonacci number	`fibonacci(n)`
number of partitions of n	`numbpart(n)`
Euler ϕ-function	`eulerphi(x)`
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)`

Miscellaneous

integer or real factorial of x	`x!` or `fact(x)`
integer square root of x	`sqrtint(x)`
solve $z \equiv x$ and $z \equiv y$	`chinese(x,y)`
minimal u, v so $xu + yv = \gcd(x,y)$	`bezout(x,y)`
multiplicative order of x (intmod) (¡=o)	`znorder(x,{o})`
primitive root mod prime power q	`znprimroot(q)`
structure of $(\mathbf{Z}/n\mathbf{Z})^*$	`znstar(n)`
continued fraction of x	`contfrac(x,{b},{lmax})`
last convergent of continued fraction x	`contfracpnqn(x)`
best rational approximation to x	`bestappr(x,k)`

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?	`Z_issquare(x,{&n})`
is *pol* irreducible?	`polisirreducible(pol)`

Send comments and corrections to ⟨Karim.BELABAS@math.u-psud.fr⟩

PARI-GP Reference Card (2)

(PARI-GP version 2.3.0)

Elliptic Curves

Elliptic curve initially given by 5-tuple $E = [a_1, a_2, a_3, a_4, a_6]$.
Points are [x,y], the origin is [0].

Initialize elliptic struct. *ell*, i.e create	`ellinit`$(E, flag)$

$a_1, a_2, a_3, a_4, a_6, b_2, b_4, b_6, b_8, c_4, c_6, disc, j$. This data can be recovered by typing *ell*`.a1`,…,*ell*`.j`. If *fl* omitted, also

E defined over $\mathbf{R}$	
x-coords. of points of order 2	*ell*`.roots`
real and complex periods	*ell*`.omega`
associated quasi-periods	*ell*`.eta`
volume of complex lattice	*ell*`.area`
E defined over $\mathbf{Q}_p$, $\lvert j\rvert_p > 1$	
x-coord. of unit 2 torsion point	*ell*`.roots`
Tate's $[u^2, u, q]$	*ell*`.tate`
Mestre's w	*ell*`.w`
change curve E using $v = [u, r, s, t]$	`ellchangecurve`(ell, v)
change point z using $v = [u, r, s, t]$	`ellchangepoint`(z, v)
cond, min mod, Tamagawa num $[N, v, c]$	`ellglobalred`(ell)
Kodaira type of p fiber of E	`elllocalred`(ell, p)
add points $z_1 + z_2$	`elladd`(ell, z_1, z_2)
subtract points $z_1 - z_2$	`ellsub`(ell, z_1, z_2)
compute $n \cdot z$	`ellpow`(ell, z, n)
check if z is on E	`ellisoncurve`(ell, z)
order of torsion point z	`ellorder`(ell, z)
torsion subgroup with generators	`elltors`(ell)
y-coordinates of point(s) for x	`ellordinate`(ell, x)
canonical bilinear form taken at z_1, z_2	`ellbil`(ell, z_1, z_2)
canonical height of z	`ellheight`$(ell, z, flag)$
height regulator matrix for pts in x	`ellheightmatrix`(ell, x)
pth coeff a_p of L-function, p prime	`ellap`(ell, p)
kth coeff a_k of L-function	`ellak`(ell, k)
vector of first n a_k's in L-function	`ellan`(ell, n)
$L(E, s)$, set $A \approx 1$	`elllseries`$(ell, s, \{A\})$
root number for $L(E, .)$ at p	`ellrootno`$(ell, \{p\})$
modular parametrization of E	`elltaniyama`(ell)
point $[\wp(z), \wp'(z)]$ corresp. to z	`ellztopoint`(ell, z)
complex z such that $p = [\wp(z), \wp'(z)]$	`ellpointtoz`(ell, p)

Elliptic & Modular Functions

arithmetic-geometric mean	`agm`(x, y)
elliptic j-function $1/q + 744 + \cdots$	`ellj`(x)
Weierstrass σ function	`ellsigma`$(ell, z, flag)$
Weierstrass $\wp$ function	`ellwp`$(ell, \{z\}, flag)$
Weierstrass ζ function	`ellzeta`(ell, z)
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)

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, w_2, dx, dy)
scale coordinates in w	`plotscale`(w, x_1, x_2, y_1, y_2)
`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([[`w_1, x_1, y_1`],`…`])`
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, x_2, y_2)
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)
Postscript Functions	
as `ploth`	`psploth`$(X$`=`$a, b, expr, flag, \{n\})$
as `plothraw`	`psplothraw`$(lx, ly, flag)$
as `plotdraw`	`psdraw([[`w_1, x_1, y_1`],`…`])`

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)

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`, *nf*`.r2`
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`
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`
$[tu, fu]$	*bnf*`.tufu`
compute a *bnf* from small *bnf*	`bnfmake`$(sbnf)$
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`
structure of $(\mathbf{Z}_K/m)^*$	*bnr*`.zkst`

Simple Arithmetic Invariants (nf)

Elements are rational numbers, polynomials, polmods, or column vectors (on integral basis *nf*`.zk`).

integral basis of field def. by $f=0$	`nfbasis(f)`
field discriminant of field $f=0$	`nfdisc(f)`
reverse polmod $a=A(X) \bmod T(X)$	`modreverse(a)`
Galois group of field $f=0$, $\deg f \le 11$	`polgalois(f)`
smallest poly defining $f=0$	`polredabs(f, flag)`
small polys defining subfields of $f=0$	`polred(f, flag, {p})`
small polys defining suborders of $f=0$	`polredord(f)`
poly of degree $\le k$ with root $x \in \mathbf{C}$	`algdep(x, k)`
small linear rel. on coords of vector x	`lindep(x)`
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)`
basic element operations (prefix `nfelt`):	
`(nfelt)mul`, `pow`, `div`, `diveuc`, `mod`, `divrem`, `val`	
express x on integer basis	`nfalgtobasis(nf, x)`
express element x as a polmod	`nfbasistoalg(nf, x)`
quadratic Hilbert symbol (at p)	`nfhilbert(nf, a, b, {p})`
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)`
number of roots of unity in *nf*	`nfrootsof1(nf)`
conjugates of a root θ of *nf*	`nfgaloisconj(nf, flag)`
apply Galois automorphism s to x	`nfgaloisapply(nf, s, x)`
subfields (of degree d) of *nf*	`nfsubfields(nf, {d})`

Dedekind Zeta Function ζ_K

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)`

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)`
fundamental units of *bnf*	`bnfunit(bnf)`
signs of real embeddings of *bnf*`.fu`	`bnfsignunit(bnf)`

Class Field Theory

ray class group structure for mod. m	`bnrclass(bnf, m, flag)`
ray class number for mod. m	`bnrclassno(bnf, m)`
discriminant of class field ext	`bnrdisc(a_1, {a_2}, {a_3})`
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(a_1, {a_2}, {a_3})`
conductor of character *chi*	`bnrconductorofchar(bnr, chi)`
conductor of extension	`bnrconductor(a_1, {a_2}, {a_3}, 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)`

PARI-GP Reference Card (2)

(PARI-GP version 2.3.0)

Ideals

Ideals are 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)`
principal ideal generated by x	`idealprincipal(nf, x)`
principal idele generated by x	`ideleprincipal(nf, 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})`

Primes and Multiplicative Structure

factor ideal x in *nf*	`idealfactor(nf, x)`
recover x from its factorization in *nf*	`factorback(x, nf)`
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

initializes a Galois group structure	`galoisinit(pol, {den})`
action of p in nfgaloisconj form	`galoispermtopol(G, {p})`
identifies as abstract group	`galoisidentify(G)`
exports a group for GAP or MAGMA	`galoisexport(G, flag)`
subgroups of the Galois group G	`galoissubgroups(G)`
subfields from subgroups of G	`galoissubfields(G, flag, {v})`
fixed field	`galoisfixedfield(G, perm, flag, {v})`
is G abelian?	`galoisisabelian(G, flag)`
abelian number fields	`galoissubcyclo(N,H,flag,{v})`

Relative Number Fields (rnf)

Extension L/K is defined by $g \in K[x]$. We have *order* $\subset L$.

absolute equation of L	`rnfequation(nf, g, flag)`
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 $\to$ relative repres. for x	`rnfeltabstorel(rnf, x)`
relative $\to$ 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`	

Projective $\mathbf{Z}_K$-modules, maximal order

relative `polred`	`rnfpolred(nf, g)`
relative `polredabs`	`rnfpolredabs(nf, g)`
characteristic poly. of a mod g	`rnfcharpoly(nf, g, a, {v})`
relative Dedekind criterion, prime *pr*	`rnfdedekind(nf, g, pr)`
discriminant of relative extension	`rnfdisc(nf, g)`
pseudo-basis of $\mathbf{Z}_L$	`rnfpseudobasis(nf, g)`
relative HNF basis of *order*	`rnfhnfbasis(bnf, order)`
reduced basis for *order*	`rnflllgram(nf, g, order)`
determinant of pseudo-matrix A	`rnfdet(nf, A)`
Steinitz class of *order*	`rnfsteinitz(nf, order)`
is *order* a free $\mathbf{Z}_K$-module?	`rnfisfree(bnf, order)`
true basis of *order*, if it is free	`rnfbasis(bnf, order)`

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)`

Send comments and corrections to ⟨Karim.BELABAS@math.u-psud.fr⟩