Some new GP features

A tutorial

B. Allombert

IMB
CNRS/Université de Bordeaux

15/01/2018

Maps between finite fields

To define an embedding of $\mathbb{F}_{3^5}$ in $\mathbb{F}_{3^{10}}$:

```
? a = ffgen([3,5],'a);
? b = ffgen([3,10],'b);
? m = ffembed(a, b);
```

The embedding *m* is a map which can be applied with `ffmap`:

```
? A = ffmap(m, a)
%4 = b^9+b^7+b^6+2*b
? minpoly(A) == minpoly(a)
%5 = 1
? A.mod == b.mod
%6 = 1
```

Reverse maps between finite fields

The reverse (partial) map *rm* from $\mathbb{F}_{3^{10}}$ to $\mathbb{F}_{3^5}$ can computed with

```
? rm = ffinvmap(m);
? ffmap(rm,b^9+b^7+b^6+2*b)
%8 = a
? ffmap(rm,b)
%9 = []
```

Extension of finite fields

```
? P = x^2+b*x+1; polisirreducible(P)
%10 = 1
? [r,n] = ffextend(b, P, 'c);
? r
%11 = c^19+c^18+c^16+2*c^14+c^13+2*c^12+c^11+2*c^9+
? Pc = ffmap(n, P)
%12 = x^2+(2*c^17+...+c+1)*x+1
? subst(Pc, x, r)
%13 = 0
? ffgen(r)
%14 = c
```

Composition

```
? rn = ffinvmap(n);
? nm = ffcompomap(n,m);
? ffmap(n,ffmap(m,a)) == ffmap(nm, a)
%17 = 1
? ffcompomap(rm, rn) == ffinvmap(nm)
%18 = 1
```

Frobenius map

```
? a = ffgen([3,5],'a);
? f = fffrobenius(a);
? ffmap(f,a) == a^3
%21 = 1
? g = fffrobenius(a, 5);
? ffmap(g,a) == a
%23 = 1
? h = fffrobenius(a, 2);
? h == ffcompomap(f,f)
%25 = 1
```

forfactored

`forfactored` allows to loop over the integers with their factorization.

```
? forfactored(n=1,6,print(n))
% [1,matrix(0,2)]
% [2,Mat([2,1])]
% [3,Mat([3,1])]
% [4,Mat([2,2])]
% [5,Mat([5,1])]
% [6,[2,1;3,1]]
```

```
? sum(i=2^30,2^30+10^6,eulerphi(i))
%2 = 653061452371896
? ##
  ***   last result computed in 3,728 ms.
? my(s=0);forfactored(n=2^30,2^30+10^6, \
            s+=eulerphi(n));s
%3 = 653061452371896
? ##
  ***   last result computed in 481 ms.
```

The function only works for upper bounds less than 2^{64}.

fordivfactored

`fordivfactored`: as `fordiv`, but also gives the factorization.

```
? fordivfactored(4!,d,print(d))
% [1,matrix(0,2)]
% [2,Mat([2,1])]
% [3,Mat([3,1])]
% [4,Mat([2,2])]
% [6,[2,1;3,1]]
% [8,Mat([2,3])]
% [12,[2,2;3,1]]
% [24,[2,3;3,1]]
```

```
? my(s=0);fordivfactored(2^128-1,d,\
          s+=moebius(d)^2);s
%4 = 512
  ***   last result computed in 0 ms.
? ##
? my(s=0);fordiv(2^128-1,d,s+=moebius(d)^2);s
%5 = 512
  ***   last result computed in 400 ms.
? ##
```

forsquarefree

`forsquarefree`: as `forfactored`, but only return squarefree numbers.

```
? forsquarefree(n=1,10,print(n))
% [1,matrix(0,2)]
% [2,Mat([2,1])]
% [3,Mat([3,1])]
% [5,Mat([5,1])]
% [6,[2,1;3,1]]
% [7,Mat([7,1])]
% [10,[2,1;5,1]]
```

forperm

Loops over permutation

```
? forperm(3, p, print(p))
% Vecsmall([1, 2, 3])
% Vecsmall([1, 3, 2])
% Vecsmall([2, 1, 3])
% Vecsmall([2, 3, 1])
% Vecsmall([3, 1, 2])
% Vecsmall([3, 2, 1])
```

forsubset

Loops over subsets

```
? forsubset(3,s,print(s))
% Vecsmall([])
% Vecsmall([1])
% Vecsmall([2])
% Vecsmall([3])
% Vecsmall([1,2])
% Vecsmall([1,3])
% Vecsmall([2,3])
% Vecsmall([1,2,3])
```

To list subsets of a given size do:

```
? forsubset([4,2],s,print(s))
% Vecsmall([1,2])
% Vecsmall([1,3])
% Vecsmall([1,4])
% Vecsmall([2,3])
% Vecsmall([2,4])
% Vecsmall([3,4])
```

matxxxmod

The usual linear algebra functions can be called with `Mod(,n)` coefficients, but they assume that n is prime.

The new functions `matdetmod`, `matinvmod`, `matimagemod`, `matkermod` perform linear algebra modulo n with n composite.

matdetmod

```
? n=36;
? M1 = [21,7,7;23,2,7;20,21,31];
? matdet(Mod(M1,n))
*** matdet: impossible inverse in Fl_inv:Mod(21,36)
? matdetmod(M1,n)
%3 = 5
? matdet(M1)%n
%4 = 5
? M2 = matrix(400,400,i,j,if(j<i,1,i==j,prime(i+2)%
n,0));
? matdet(M2)%n;
time = 3,472 ms.
? matdetmod(M2,n);
time = 44 ms.
```

matinvmod

```
? Mod(M1,n)^(-1)
  *** _^_: impossible inverse in Fl_inv: Mod(21, 36
? matinvmod(M1,n)
%8 =
[19 22  7]
[15 23 10]
[31 19  5]
? M1^(-1)%n == %
%9 = 1
? N = M2^(-1)%n;
time = 8,945 ms.
? N == matinvmod(M2,n)
time = 44 ms.
%11 = 1
```

matimagemod and matkermod

The functions `matimagemod` and `matinvmod` return their result in Howell form, a normal form for subgroups of $(\mathbb{Z}/n\mathbb{Z})^d$.

```
? M3 = [20,7,31;30,25,9;20,21,31];
? matimagemod(M3,n)
%13 =
[6 2 5]
[0 2 1]
[0 0 1]
? matkermod(M3,n)
%14 =
[18  6]
[ 0 18]
[ 0  6]
```

ellratpoints

PARI/GP now include a port of Michael Stoll `ratpoints` program to find rational points of small height on hyperelliptic curves. For the curve $y^2 = x^3 - 25x + 1$:

```
? E=ellinit([-25,1]);
? ellratpoints(E,10)
%9 = [[-5,1],[-5,-1],[-3,7],[-3,-7],[-1,5],[-1,-5],
```

gives all points on E with numerator and denominator of the abscissa bounded by 10. It is also possible to give differents bound for the numerator and the denominator:

```
? ellratpoints(E,[10^6,1])
%10 = [[-5,1],[-5,-1],[-3,7],[-3,-7],[-1,5],[-1,-5]
```

gives the integrals points with abscissa less than 10^6.

hyperellratpoints

For the curve $y^2 = x^6 + x + 1$:

```
? hyperellratpoints(x^6+x+1,100) \\ y^2 = x^6+x+1
%11 = [[-1,1],[-1,-1],[0,1],[0,-1],
%      [19/20,13109/8000],[19/20,-13109/8000]]
? (19/20)^6+(19/20)+1-(13109/8000)^2
%12 = 0
? hyperellratpoints(x^5-5*x^3+4*x+1,[1000,1])
%13 = [[-2,1],[-2,-1],[-1,1],[-1,-1],[0,1],[0,-1],[
```

ellbsd

Let E be an elliptic curve over a number field of rank r The function `ellbsd` gives c such that $L_E^{(r)}(1)/r! = cRS$ where R is the elliptic regulator and S the conjectured cardinal of the Tate-Shafarevich group.

```
? E   = ellinit([0,1,1,-2,0]);
? N   = ellglobalred(E)[1]
%2 = 389
? tor = elltors(E) \\ trivial
%3 = [1,[],[]]
? lfunorderzero(E)
%4 = 2
? G = ellgenerators(E); \\ with elldata
? G = [[-1,1],[0,0]];  \\ without elldata
```

ellbsd

We check the BSD conjecture for E.

```
? tam = elltamagawa(E)
%40 = 2
? reg = matdet(ellheightmatrix(E,G));
? bsd = (E.omega[1]*tam)*reg
%42 = 0.75931650028842677023019260789472201908
? ellbsd(E)*reg
%43 = 0.75931650028842677023019260789472201908
? L1  = lfun(E,1,2)/2!
%44 = 0.75931650028842677023019260789472201908
```

Elliptic curves over number fields

The following functions now works over number fields: `ellheight`, `ellrootno`, `ellisomat`, `ellbsd`. See <https://pari.math.u-bordeaux.fr/Events/PARI2017b/talks/elliptic.pdf>

Infinite sums of rational functions

```
? F=1/(1+x^2);sumnumrat(F,0)
%12 = 2.0766740474685811741340507947500004904
? (Pi*I*cotan(I*Pi)+1)/2
%13 = 2.0766740474685811741340507947500004904
? sumaltrat(F,0)
%14 = 0.6360145274910665814751182918360187779 2
? (Pi/cos(I*Pi)+1)/2
%15 = 0.6355074756997091739432804154856896129 8
? F=1+1/x^2;prodnumrat(F,1)
%16 = 3.6760779103749777206956974920282606665
? exp(sumpos(a=1,log(1+1/a^2)))
%17 = 3.6760779103749777206956974920282606664
```

Sums of rational functions over prime numbers

```
? prodeulerrat(1+p^-2)
%9 = 1.5198177546350665716581919481459145836
? zeta(2)/zeta(4)
%10 = 1.5198177546350665716581919481459145836
? sumeulerrat(1/p^2)
%11 = 0.45224742004106549850654336483224793418
```

zetahurwitz

Hurwitz zeta function $\zeta(s,x) = \sum_{n\geq 0}(n+x)^{-s}$. We must have Re(s) > 1 and x real positive. Note that $\zeta(s,1) = \zeta(s)$ We also define $\zeta(1,x) = -\psi(x)$, where ψ is the digamma function.

```
? zetahurwitz(Pi,Pi)
%1 = 0.056155444497585099925180502385781494486
? zetahurwitz(2,1) - zeta(2)
%2 = -2.3509887016445750l6 E-38
? zetahurwitz(1,1) - Euler
%3 = 2.6448622893501468 93 E-38
```

lfuntwist

lfuntwist allows to twist an L function by a Dirichlet character. The conductors need to be coprime.

```
? E=ellinit([0,-1,1,-10,-20]);
? L=lfuntwist(E,Mod(2,5));
? lfunan(E,10)
%3 = [1,-2,-1,2,1,2,-2,0,-2,-2]
? lfunan(Mod(2,5),10)
%4 = [1,I,-I,-1,0,1,I,-I,-1,0]
? lfunan(L,10)
%5 = [1,-2*I,I,-2,0,2,-2*I,0,2,0]
```

lfuntwist

```
? nf=nfinit(polcyclo(5,'a));
? E2=ellinit(E[1..5],nf);
? localbitprec(64); lfun(E2,2)
%8 = 1.0543811873412420765
? L2=lfuntwist(E,Mod(4,5));
? lfun(E,2)*lfun(L2,2)*norm(lfun(L,2))
%10 = 1.0543811873410821651289745964738865962
```

Permutation

```
? p=Vecsmall([4,5,1,2,3]);
? permorder(p)
%6 = 5
? permsign(p)
%7 = 1
? p^5
%8 = Vecsmall([1,2,3,4,5])
```

galoischartable

See Wednesday talk

```
? G=galoisinit(x^6+108);
? [T,o]=galoischartable(G);
? T~
%8 =
% [1   1   1]
% [1   1 -1]
% [2 -1   0]
```

binomial

```
? binomial(7)
%7 = [1,7,21,35,35,21,7,1]
? V1=vector(5001,i,binomial(5000,i-1));
? ##
  ***   last result computed in 1,520 ms.
? V=binomial(5000);
? ##
  ***   last result computed in 8 ms.
? V==V1
%12 = 1
```

elliptic curve primality proof

```
? isprime(3^500+196)
%13 = 1
? ##
  ***   last result computed in 1,451 ms.
? C=ecpp(3^500+196);
? ##
  ***   last result computed in 917 ms.
? ecppisvalid(C)
%15 = 1
```

See Jared talk

modular symbols and p-adic L-functions

New functions msdim, mslattice, mspetersson, mspolygon, ellweilcurve, ellpadicregulator Ask Karim.

znchar

New functions: znchar, znchargalois, charpow
znchardecompose, zncharconductor, znchartoprimitive
znchargauss
Ask Karim.

SVG support

See `https://pari.math.u-bordeaux.fr/gpexp.html`.

Miscellaneous

```
? vecprod([2,3,5,7])
%1 = 210
? zeta(1/2+x+O(x^4))
%2 = -1.46035451-3.92264614*x-8.00417851*x^2-16.000
? laurentseries(x->zeta(x),4)
%3 = -0.500000000-0.918938533*x-1.00317823*x^2-1.00
? exponent(4.^111)
%4 = 222
? exponent(Pi^2)
%5 = 3
? printp([a,b;c,d])
% [a b]
% [c d]
? bestapprnf(ellj((1+sqrt(-23))/2),quadhilbert(-23)
%7 = Mod(-1084125*x^2+1904875*x-1437500,x^3-x^2+1)
```