Finite fields

B. Allombert

IMB
Inria/Université de Bordeaux

16/04/2024

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 676541

Prime finite fields

To create a random prime number :

```
? p=randomprime(2^100)
%1 = 792438309994299602682608069491
```

To create an element of $\mathbb{F}_p$:

```
? a=Mod(17,p);
? a^(p-1) \\ powering
%3 = Mod(1,792438309994299602682608069491)
```

To access the components of *a* :

```
? a.mod
%4 = 792438309994299602682608069491
? lift(a) \\lift to Z
%5 = 17
```

General finite fields

To build an irreducible polynomial of degree n of $\mathbb{F}_p$, use `ffinit(p,n)`.

```
? P=ffinit(13,2)
%6 = Mod(1,13)*x^2+Mod(1,13)*x+Mod(12,13)
? polisirreducible(P)
%7 = 1
```

To build an element of $\mathbb{F}_{p^n}$ from its minimal polynomial :

```
? a=ffgen(P,'a)
%8 = a
```

The above can be abbreviated by `ffgen(p^n,'a)`.

```
? a=ffgen(13^2,'a);
```

The sign `'a` says you want the element to be printed as `a`

General finite fields

```
? b = a^2+3*a+2
%10 = 2*a+3
```

To access the components of *b* :

```
? b.pol
%11 = 2*a+3
? b.mod
%12 = a^2+a+12
? [b.p, b.f]
%13 = [13, 2]
```

To recover *a* from *b* :

```
? ffgen(b)
%14 = a
```

Operations on elements

You can use many generic functions on finite field elements.

```
? c = ffgen(3^8,'c);
? d = random(c) \\random element in the field
%16 = 2*c^6 + 2*c^5 + 2*c^3 + c^2 + 2*c + 1
? issquare(d)
%17 = 1
? trace(d) \\over F_3
%18 = Mod(2, 3)
? norm(d)
%19 = Mod(1, 3)
? minpoly(d^82)
%20 = Mod(1,3)*x^4+Mod(2,3)*x^3+Mod(1,3)*x^2+Mod(2,
```

Operations on elements

```
? factor(x^5+x^3+c)
%21 = [x + (2*c^5 + c^4 + 2*c) 1]
%     [x^2 + (c^7 + 2*c^6 + ... + c^2 + 2) 1]
%     [x^2 + (2*c^7 + c^6 + ... + 2*c^2 + 1) 1]
? R=polrootsmod(x^7+x+c)
%22 = [c^7 + 2*c^6 + c^5 + c^3 + 2*c + 2,
%      2*c^7 + c^6 + c^2 + 1]~
? subst(x^7+x+c,x,R)
%23 = [0,0]~
```

Operations related to the multiplicative structure

Warning : the field generator is not necessarily a primitive root (group generator) !

```
? fforder(c)
%24 = 1640
? z = ffprimroot(c)
%25 = 2*c^7+2*c^6+2*c^5+2*c^4+c^3+c^2+c+2
? fforder(z)
%26 = 6560
? n = fflog(c,z)
%27 = 2612
? z^n
%28 = c
```

Reminder : there are corresponding functions on rings $\mathbb{Z}/N\mathbb{Z}$: `znorder`, `znprimroot`, `znlog`.

Maps between finite fields

There is a structure for maps between finite fields.

```
? d = ffgen([3,24],'d)
%29 = d
? Mcd = ffembed(c,d); \\compute some embedding
? ffembed(d,c)
  ***   at top-level: ffembed(d,c)
  ***                 ^-------------
  *** ffembed: domain error in ffembed: d is not a
? c2 = ffmap(Mcd,c^5+c+1) \\apply the map
%32 = d^20+2*d^18+d^16+d^15+2*d^14+2*d^12+2*d^11+2*
? F = fffrobenius(d,8); \\8-th power of Frobenius
? ffmap(F, d) == d^(3^8)
%34 = 1
? ffmap(F, c2) == c2
%35 = 1
```

Extending finite fields

You can construct extensions of finite fields defined by an irreducible polynomial with `ffextend`.

```
? T = x^3+d*x+1; polisirreducible(T)
%36 = 1
? [e,Mde] = ffextend(d, T, 'e);
? e.f
%38 = 72
? fforder(e)
%39 = 1595328861543098787996 86
? ffmap(Mde, d)
%40 = 2*e^67 + e^66 + e^65 + 2*e^64 + e^59 + e^58 +
```

Composing maps

You can compute the composition of maps :
`ffcompomap`$(f, g) = f \circ g$.

```
? Mce = ffcompomap(Mde,Mcd);
? ffmap(Mce, c) == ffmap(Mde, ffmap(Mcd, c))
%42 = 1
? ffcompomap(F,Mcd) == Mcd
%43 = 1
? ffcompomap(F,F) == fffrobenius(d,16)
%44 = 1
```

Preimages

You can compute the partial inverse of a map with `ffinvmap`.

```
? Mdc = ffinvmap(Mcd);
? ffmap(Mdc, ffmap(Mcd, c^3+c+1))
%46 = c^3 + c + 1
? Mec = ffcompomap(Mdc, ffinvmap(Mde));
? ffmap(Mec, ffmap(Mce, c))
%48 = c
? ffinvmap(fffrobenius(c,3)) == fffrobenius(c,5)
%49 = 1
```

Relative extensions

```
? ffmap(Mdc, d)
%50 = []
```

This indicates that *d* is not in the field of definition of *c*.
To express *d* as an algebraic element over the field of definition of *c*, use `ffmaprel`

```
? rd = ffmaprel(Mdc, d)
%51 = Mod(d,d^3+d^2+(2*c^7+2*c^6+2*c^4+2*c^3+c+2)*d
? sd = ffmaprel(Mdc, d^4+1)
%52 = Mod((c^7+c^6+c^4+c^3+2*c+2)*d^2+(2*c^7+c^6+2*
```

Relative extensions

This allows to compute relative trace norm and minimal polynomial :

```
? trace(rd)
%53 = 2
? norm(rd)
%54 = 2*c^6+2*c^5+2*c^4+c^3+2*c
? [norm(norm(rd)), norm(d)]
%55 = [Mod(1,3), Mod(1,3)]
? minpoly(rd)
%56 = x^3+x^2+(2*c^7+2*c^6+2*c^4+2*c^3+c+2)*x+(c^6+
? minpoly(sd)
%57 = x^3+(2*c^6+2*c^5+2*c^4+c^3)*x^2+(2*c^6+c^5+2*
```

Creation from number fields

You can create finite fields as residue fields of prime ideals.

```
\begin{verbatim}
? nf = nfinit(y^8-2*y^7+9*y^6-2*y^5+38*y^4-34*y^3\
              +31*y^2-6*y+1);
? pr = idealprimedec(nf,2)[2]; [pr.e,pr.f]
%58 = [2, 2]
? modpr = nfmodprinit(nf,pr,'z); \\ compute the mor
? g = nfmodpr(nf,y,modpr)
%60 = z
? nfmodpr(nf,y^2+1,modpr)
%61 = z
? nfmodprlift(nf,g+1,modpr) \\trouve une préimage
%62 = [1, 1, 0, 0, 0, 0, 0, 0]~
```

Elliptic curves construction

An elliptic curve given from its short

$$y^2 = x^3 + a_4 x + a_6$$

or long

$$y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6$$

Weierstrass equation is defined by

```
? E=ellinit([a4,a6]);
? E=ellinit([a1,a2,a3,a4,a6]);
```

Elliptic curves over a finite field

Let u be a finite field element :

```
? u = ffgen([101,2],'u);
? E = ellinit([10,81*u+94],u);
```

(The extra `u` is to make sure the curve is defined over $\mathbb{F}_{101^2}$ and not $\mathbb{F}_{101}$).

```
? ellcard(E) \\ cardinal of E(F_q)
%69 = 10116
? P = random(E) \\ random point on E(F_q)
%70 = [19*u + 57, 34*u + 29]
? Q = random(E) \\ another random point on E(F_q)
%71 = [u + 23, 6*u + 95]
? ellisoncurve(E, P) \\ check that the point is on
%72 = 1
```

Elliptic curves over a finite field

```
? elladd(E, P, Q)   \\ P+Q in E
%73 = [20*u + 37, 98*u + 92]
? ellmul(E, P, 100) \\ 100.P in E
%74 = [12*u + 5, 71*u + 38]
? ellorder(E,P) \\order of P
%75 = 1686
```

Structure of the group $E(\mathbb{F}_q)$

```
? [d1,d2]=ellgroup(E) \\ structure of E(F_q)
%76 = [1686, 6]
```

Above $[d1, d2]$ means $\mathbb{Z}/d_1\mathbb{Z} \times \mathbb{Z}/d_2\mathbb{Z}$, with $d_2 \mid d_1$.

Pairings

```
? [G1,G2] = ellgenerators(E)
%77 = [[48*u + 68, 22*u + 10], [35*u + 85, 62*u + 1
? ellorder(E,G1)
%78 = 1686
? w = ellweilpairing(E,G1,G2,d1)
%79 = u + 1
? fforder(w)
%80 = 6
? t = elltatepairing(E,G2,G1,d2)^((101^2-1)/d2)
%81 = 100*u
? fforder(t)
%82 = 6
```

Discrete logarithms

```
? e = random(d1);
? S = ellmul(E,P,e)
%84 = [50*u + 58, 85*u + 24]
? elllog(E,S,P)
%81= 240
? e
%86 = 240
```

Twists

```
? et = elltwist(E)
%87 = [0, 0, 0, 96*u + 25, 62*u + 74]
? Et = ellinit(et);
? ellap(E)
%89 = 86
? ellap(Et)
%90 = -86
```

Isogenies

```
? P3 = ellmul(E,G1,d1/3);
? ellorder(E,P3)
%92 = 3
? [eq,iso] = ellisogeny(E,P3);
? eq
%94 = [0, 0, 0, u + 12, 8*u + 2]
? iso
%95 = [x^3+85*u*x^2+(57*u+77)*x+(80*u+59),
%      y*x^3+77*u*y*x^2+(91*u+71)*y*x+(84*u+35)*y,
%      x+93*u]
? G1q = ellisogenyapply(iso, G1)
%96 = [8*u + 98, 59*u + 37]
? Eq = ellinit(eq); ellorder(Eq, G1q)
%97 = 562
```