```
gap> # A gentle introduction to FinInG
gap> # webpage: <http://cage.ugent.be/fining>
gap> # 1. Importing geometric objects into your GAP session
gap> # 2. Performing your computations with FinInG
gap> # 3. Extracting the information
gap> # Examples:
gap> # First we load FinInG:
gap> LoadPackage("fining");
------------------------------------------------------------------------
-
Loading 'Forms' 1.2.3 (26/10/2015)
by John Bamberg (http://school.maths.uwa.edu.au/~bamberg/)
   Jan De Beule (http://www.debeule.eu)
For help, type: ?Forms
------------------------------------------------------------------------
-
------------------------------------------------------------------------
---------
Loading  orb 4.7.6 (Methods to enumerate orbits)
by Juergen Mueller (http://www.math.rwth-aachen.de/
~Juergen.Mueller),
   Max Neunhöffer (http://www-groups.mcs.st-and.ac.uk/~neunhoef),
and
   Felix Noeske (http://www.math.rwth-aachen.de/~Felix.Noeske).
Homepage: https://gap-packages.github.io/orb
------------------------------------------------------------------------
---------
------------------------------------------------------------------------
---------
Loading  cvec 2.5.5 (Compact vectors over finite fields)
by Max Neunhöffer (http://www-groups.mcs.st-and.ac.uk/~neunhoef).
Homepage: https://gap-packages.github.io/cvec
------------------------------------------------------------------------
---------
------------------------------------------------------------------------
---------
Loading  genss 1.6.4 (Generic Schreier-Sims)
by Max Neunhöffer (http://www-groups.mcs.st-and.ac.uk/~neunhoef) and
   Felix Noeske (http://www.math.rwth-aachen.de/~Felix.Noeske).
Homepage: https://gap-packages.github.io/genss
------------------------------------------------------------------------
---------
------------------------------------------------------------------------
---------
Loading  GRAPE 4.7 (GRaph Algorithms using PErmutation groups)
by Leonard H. Soicher (http://www.maths.qmul.ac.uk/~leonard/).
Homepage: http://www.maths.qmul.ac.uk/~leonard/grape/
------------------------------------------------------------------------
---------
loading: geometry, liegeometry, group, projectivespace,
correlations, polarspace/morphisms, enumerators, diagram, varieties,
affinespace/affinegroup, gpolygons, orbits+stabilisers
------------------------------------------------------------------------
----------
```

```
        _______________       ________      _________
_______________
        ___  ____/__(_)_____________  _/________  ____/   __<  /__|
__  /
        __  /_   __  /__  __ \__  / __  __ \  / __     __  / ___/_
<
        _  __/   _  / _  / / /_/ /  _  / / / /_/ /     _  /
______/ /
        /_/      /_/  /_/ /_//___/  /_/ /_/\____/      /_/_(_)____/
------------------------------------------------------------------
----------
Loading  FinInG 1.3.3 (Finite Incidence Geometry)
by John Bamberg (http://school.maths.uwa.edu.au/~bamberg/)
   Anton Betten (http://www.math.colostate.edu/~betten)
   Jan De Beule (http://www.debeule.eu)
   Philippe Cara (http://homepages.vub.ac.be/~pcara)
   Michel Lavrauw (http://cage.ugent.be/~ml)
   Max Neunhoeffer (http://www-groups.mcs.st-and.ac.uk/~neunhoef/)
For help, type: ?FinInG
------------------------------------------------------------------
-
true
gap>
gap>
gap> # 1. An arc (inspired by the talk of Simeon Ball)
gap> q:=9;
9
gap> pg:=PG(4,q);
ProjectiveSpace(4, 9)
gap> eta:=First(GF(q),x->x^4=-One(GF(q)));
Z(3^2)
gap> vecs:=List(GF(q),t->[1,t,t^2+eta*t^6,t^3,t^4]*One(GF(q)));
[ [ Z(3)^0, 0*Z(3), 0*Z(3), 0*Z(3), 0*Z(3) ],
  [ Z(3)^0, Z(3^2), Z(3^2)^5, Z(3^2)^3, Z(3) ],
  [ Z(3)^0, Z(3^2)^5, Z(3^2)^5, Z(3^2)^7, Z(3) ],
  [ Z(3)^0, Z(3)^0, Z(3^2)^2, Z(3)^0, Z(3)^0 ],
  [ Z(3)^0, Z(3^2)^2, Z(3^2)^6, Z(3^2)^6, Z(3)^0 ],
  [ Z(3)^0, Z(3^2)^3, Z(3^2), Z(3^2), Z(3) ],
  [ Z(3)^0, Z(3), Z(3^2)^2, Z(3), Z(3)^0 ],
  [ Z(3)^0, Z(3^2)^7, Z(3^2), Z(3^2)^5, Z(3) ],
  [ Z(3)^0, Z(3^2)^6, Z(3^2)^6, Z(3^2)^2, Z(3)^0 ] ]
gap> vecs:=Union([[0,0,0,0,1]*One(GF(q))],vecs);
[ [ 0*Z(3), 0*Z(3), 0*Z(3), 0*Z(3), Z(3)^0 ],
  [ Z(3)^0, 0*Z(3), 0*Z(3), 0*Z(3), 0*Z(3) ],
  [ Z(3)^0, Z(3)^0, Z(3^2)^2, Z(3)^0, Z(3)^0 ],
  [ Z(3)^0, Z(3), Z(3^2)^2, Z(3), Z(3)^0 ],
  [ Z(3)^0, Z(3^2), Z(3^2)^5, Z(3^2)^3, Z(3) ],
  [ Z(3)^0, Z(3^2)^2, Z(3^2)^6, Z(3^2)^6, Z(3)^0 ],
  [ Z(3)^0, Z(3^2)^3, Z(3^2), Z(3^2), Z(3) ],
  [ Z(3)^0, Z(3^2)^5, Z(3^2)^5, Z(3^2)^7, Z(3) ],
  [ Z(3)^0, Z(3^2)^6, Z(3^2)^6, Z(3^2)^2, Z(3)^0 ],
  [ Z(3)^0, Z(3^2)^7, Z(3^2), Z(3^2)^5, Z(3) ] ]
gap> Garc:=List(vecs,v->VectorSpaceToElement(pg,v));
[ <a point in ProjectiveSpace(4, 9)>, <a point in ProjectiveSpace(4,
```

```
9)>,
  <a point in ProjectiveSpace(4, 9)>, <a point in ProjectiveSpace(4,
9)>,
  <a point in ProjectiveSpace(4, 9)>, <a point in ProjectiveSpace(4,
9)>,
  <a point in ProjectiveSpace(4, 9)>, <a point in ProjectiveSpace(4,
9)>,
  <a point in ProjectiveSpace(4, 9)>, <a point in ProjectiveSpace(4,
9)> ]
gap> G:=ProjectivityGroup(pg);
The FinInG projectivity group PGL(5,9)
gap> H:=FiningSetwiseStabiliser(G,AsSet(Garc));
#I  Computing adjusted stabilizer chain...
<projective collineation group with 6 generators>
gap> StructureDescription(H);
"A6"
gap>
gap> # let's check that this is an arc
gap> hyps:=Hyperplanes(pg);
<solids of ProjectiveSpace(4, 9)>
gap> ints:=[];
[  ]
gap> for h in hyps do
> Add(ints,Size(Filtered(Garc,x->x in h)));
> od;
gap> AsSet(ints);
[ 0, 1, 2, 3, 4 ]
gap> Collected(ints);
[ [ 0, 2286 ], [ 1, 2770 ], [ 2, 1755 ], [ 3, 360 ], [ 4, 210 ] ]
gap>
gap> # 2. A cubic surface and the 27 lines on it (inspired by the
talk of Anton Betten)
gap> q:=7;
7
gap> pg:=PG(3,q);
ProjectiveSpace(3, 7)
gap> r:=PolynomialRing(GF(q),4);
GF(7)[x_1,x_2,x_3,x_4]
gap> f:=r.1^2*r.4+r.2^2*r.3+3*r.2*r.3^2+r.1*r.4^2+3*r.1*r.2*r.3+2*r.
1*r.2*r.4+3*r.2*r.3*r.4;
x_1^2*x_4+Z(7)*x_1*x_2*x_3+Z(7)^2*x_1*x_2*x_4+x_1*x_4^2+x_2^2*x_3+Z(
7)*x_2*x_3\
^2+Z(7)*x_2*x_3*x_4
gap> Csur:=AlgebraicVariety(pg,[f]);
Projective Variety in ProjectiveSpace(3, 7)
gap> pts:=AsList(Points(Csur));;
gap> lines:=Filtered(Lines(pg),l->Size(Filtered(Points(l),x->x in
Csur))=q+1);
[ <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
```

```
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)>, <a line in ProjectiveSpace(3,
7)>,
  <a line in ProjectiveSpace(3, 7)> ]
gap> Size(lines);
27
gap>
gap>
gap> # 3. Klein quadric (a "classic")
gap> q:=7;
7
gap> k := KleinCorrespondence( q );
<geometry morphism from <lines of ProjectiveSpace(3, 7)> to <points
of Q+(5,
7): x_1*x_6+x_2*x_5+x_3*x_4=0>>
gap> Q:=Range(k);
<points of Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> ps:=AmbientGeometry(Q);
Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0
gap> PolarSpaceType(ps);
"hyperbolic"
gap> EquationForPolarSpace(ps);
x_1*x_6+x_2*x_5+x_3*x_4
gap> TypesOfElementsOfIncidenceStructure(ps);
[ "point", "line", "plane" ]
gap> l1:=Random(Lines(PG(3,q)));
<a line in ProjectiveSpace(3, 7)>
gap> l2:=First(Lines(PG(3,q)),line->Dimension(Span(l1,line))=3);
<a line in ProjectiveSpace(3, 7)>
gap> p1:=l1^k;p2:=l2^k;
<a point in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
<a point in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> IsCollinear(ps,p1,p2);
false
gap> l3:=First(Lines(PG(3,q)),line->Dimension(Span(l1,line))=2);
<a line in ProjectiveSpace(3, 7)>
gap> p3:=l3^k;
```

```
<a point in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> IsCollinear(ps,p1,p3);
true
gap> Lines(p1);
<shadow lines in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> Planes(p1);
<shadow planes in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> IsomorphismPolarSpaces(ps,HyperbolicQuadric(5,q));
<geometry morphism from <Elements of Q+(5,
7): x_1*x_6+x_2*x_5+x_3*x_4=0> to <Elements of Q+(5, 7)>>
gap> CollineationGroup(HyperbolicQuadric(5,q));
PGammaO+(6,7)
gap>
gap>
gap> # 4. A Cameron-Liebler line class (inspired by the talk given
by Francesco Pavese)
gap>
gap> q:=7;
7
gap> nonzerosquares:=List(Filtered(GF(q),y->not y=Zero(GF(q))),x-
>x^2);
[ Z(7)^0, Z(7)^2, Z(7)^4, Z(7)^0, Z(7)^2, Z(7)^4 ]
gap> pg:=PG(3,q);
ProjectiveSpace(3, 7)
gap> lines:=Lines(pg);
<lines of ProjectiveSpace(3, 7)>
gap> planes:=Planes(pg);
<planes of ProjectiveSpace(3, 7)>
gap> r:=PolynomialRing(GF(q),4);
GF(7)[x_1,x_2,x_3,x_4]
gap> w:=PrimitiveElement(GF(q));
Z(7)
gap> f:=r.1^2-w*r.2^2+r.3*r.4;
x_1^2+Z(7)^4*x_2^2+x_3*x_4
gap> var:=QuadraticVariety(pg,f);
Quadratic Variety in ProjectiveSpace(3, 7)
gap> form:=QuadraticForm(var);
< quadratic form >
gap> ps:=PolarSpace(var);
<polar space in ProjectiveSpace(3,GF(7)):
x_1^2+Z(7)^4*x_2^2+x_3*x_4=0 >
gap> Display(ps);
<polar space of rank 1 in PG(3, 7)>
Non-singular elliptic quadratic form
Gram Matrix:
 1 . . .
 . 4 . .
 . . . 1
 . . . .
Polynomial: x_1^2+Z(7)^4*x_2^2+x_3*x_4

Witt Index: 1
Elliptic bilinear form
Gram Matrix:
```

```
 2 . . .
 . 1 . .
 . . . 1
 . . 1 .
Witt Index: 1
gap> eq:=EllipticQuadric(3,q);
Q-(3, 7)
gap> map:=IsomorphismPolarSpaces(eq,ps);
<geometry morphism from <Elements of Q-(3, 7)> to <Elements of Q-(3,
7): x_1^2+Z(7)^4*x_2^2+x_3*x_4=0>>
gap> int:=Intertwiner(map);
MappingByFunction( PDeltaO-(4,7), <projective collineation group of
size
235200 with 3 generators>, function( y ) ... end, function( x ) ...
end )
gap> G:=Image(int);
<projective collineation group of size 235200 with 3 generators>
gap> H:=CommutatorSubgroup(G,G);
<projective collineation group>
gap> ptorbs:=FiningOrbits(H,Points(pg));
43%..87%..100%..[ <closed orbit, 175 points>, <closed orbit, 175
points>,
  <closed orbit, 50 points> ]
gap> squarepts:=Filtered(ptorbs,o-
>EvaluateForm(form,Coordinates(o[1])) in nonzerosquares)[1];
<closed orbit, 175 points>
gap> lineorbs:=FiningOrbits(H,Lines(pg));
42%..50%..92%..100%..[ <closed orbit, 1225 points>, <closed orbit,
200 points>,
  <closed orbit, 1225 points>, <closed orbit, 200 points> ]
gap> l1:=VectorSpaceToElement(pg,[[1,0,0,0],[0,0,1,0]]*One(GF(q)));
#tangent
<a line in ProjectiveSpace(3, 7)>
gap> o1:=FiningOrbit(H,l1);
<closed orbit, 200 points>
gap> l2:=VectorSpaceToElement(pg,[[1,0,0,0],[0,1,0,0]]*One(GF(q)));
# external
<a line in ProjectiveSpace(3, 7)>
gap> o2:=FiningOrbit(H,l2);
<closed orbit, 1225 points>
gap> bd:=Union(o1,o2);;
gap> Size(bd);
1425
gap>
gap> CameronLieblerLineClassFunction:=function(pg,set)
> local a,b,l,m,lines,compl;
> a:=AsSet(List(set,l->Size(Filtered(set,m-
>Dimension(Span(m,l))=2))));
> lines:=Lines(pg);
> compl:=Filtered(lines,l->not l in set);
> b:=AsSet(List(compl,l->Size(Filtered(set,m-
>Dimension(Span(m,l))=2))));
> if Size(a)=1 and Size(b)=1 then
> return [true,[a[1],b[1]]];
```

```
> else return false;
> fi;
> end;
function( pg, set ) ... end
gap>
gap> CameronLieblerLineClassFunction(pg,bd);
[ true, [ 248, 200 ] ]
gap>
gap>
```