Lecture 3. Incidence geometries and substructure (FinInG package)

Michel Lavrauw
University of Primorska

GAP Days Spring 2025, VUB

Outline

Incidence Structures

Incidence structures in FinInG

IncidenceStructure(elems, incidence_relation, type_function, type_set)

An **incidence structure** consists of

- a set of elements,
- an **incidence relation**: symmetric relation on the set of elements,
- a **type function** from the set of elements to an index set (i.e. a **set of types**),

and satisfies the following axiom:

(i) no two elements of the same type are incident.

An **incidence geometry** is an incidence structure satisfying the following axiom:

(ii) every maximal flag contains an element of each type.

Example of an Incidence Structure

First we define "points".

```
gap> pg := PG(4,2);; pg2 := PG(9,2);;
gap> GrassmannVariety(Lines(pg));
Grassmann Variety in ProjectiveSpace(9, 2)
gap> gm:=GrassmannMap(last);
Grassmann Map of <lines of ProjectiveSpace(4, 2)>
gap> points := List(Lines(pg),x->x^gm);;
  # points on the Grassmann variety
```

Next we define "lines".

```
gap> flags := Concatenation(List(Points(pg),x->List(Planes(x),y->
  FlagOfIncidenceStructure(pg,[x,y]))));; # point-plane flags
gap> prelines := List(flags,flag->ShadowOfFlag(pg,flag,2));;
  # pencils of lines
gap> lines := List(prelines,x->Span(List(x,y->y^gm)));;
  # lines on the Grasmann variety
```

Example of an Incidence Structure - continued

Next in line are the "planes".

```
gap> flags2 := Concatenation(List(Points(pg),x->List(Solids(x),y->
  FlagOfIncidenceStructure(pg,[x,y]))));; # point-solid flags
gap> flags2[1];
<a flag of ProjectiveSpace(4, 2)>
gap> Display(last);
<a flag of ProjectiveSpace(4, 2) )> with elements of types [ 1, 4 ]
respectively spanned by
[ NewVector(IsCVecRep,GF(2,1),[Z(2)^0,0*Z(2),0*Z(2),0*Z(2),0*Z(2),]),
NewMatrix(IsCMatRep,GF(2,1),5,[[ Z(2)^0, 0*Z(2), 0*Z(2), 0*Z(2), 0*Z(2)],
[ 0*Z(2), Z(2)^0, 0*Z(2), 0*Z(2), 0*Z(2) ],[ 0*Z(2), 0*Z(2), Z(2)^0, 0*Z(2), 0*
],[ 0*Z(2), 0*Z(2), 0*Z(2), Z(2)^0, 0*Z(2) ],]) ]
gap> flags2[1]!.els;
[ <a point in ProjectiveSpace(4, 2)>, <a solid in ProjectiveSpace(4, 2)> ]
gap> preplanes := List(flags2,flag->ShadowOfFlag(pg,flag,2));;
  # stars of lines
gap> planes := List(preplanes,x->Span(List(x,y->y^gm)));;
  # planes on the Grassmann variety
gap> Collected(List(planes,x->Dimension(x)));
[ [ 2, 465 ] ]
```

Example of an Incidence Structure - continued

Finally, we define two more type of elements.

```
gap> maximals1 := List(Planes(pg),x->Span(List(Lines(x),y->y^gm)));;
  # image of a field of lines: a plane on the Grassmann variety
gap> Collected(List(maximals1,x->Dimension(x)));
[ [ 2, 155 ] ]
gap> maximals2 := List(Points(pg),x->VectorSpaceToElement(pg2,List(Lines(x),y->
  # image of a 3-dim star of lines: a solid on the Grassmann variety
gap> Collected(List(maximals2,x->Dimension(x)));
[ [ 3, 31 ] ]
```

Finally, we can define our incidence structure!

```
gap> elements := Union(points,lines,planes,maximals1,maximals2);;
gap> type := x -> ProjectiveDimension(x)+1;
function( x ) ... end
gap> inc_rel := \*;
<Operation "*">
gap> inc := IncidenceStructure(elements,inc_rel,type,[1,2,3,4]);
Incidence structure of rank 4
gap> IsIncidenceGeometry(inc);
false
```

Coset Geometries

Cosets Geometries in FinInG

A **coset geometry** in FinInG is an incidence structure defined by a group G and a list L of subgroups of G:

```
CosetGeometry( G, L )
```

The subgroups in L will be the **parabolic subgroups** of the coset geometry whose rank equals the length of L.

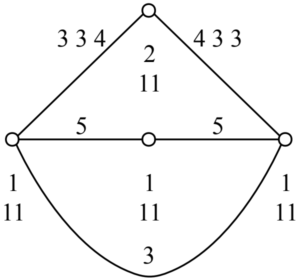
The **Borel subgroup** is equal to the stabiliser of a chamber. It corresponds to the intersection of all parabolic subgroups.

Example of a Coset Geometry

```
gap> G := PSL(2,11);
Group([ (3,11,9,7,5)(4,12,10,8,6), (1,2,8)(3,7,9)(4,10,5)(6,12,11) ])
gap> G1 := Group([ (1,2,3)(4,8,12)(5,10,9)(6,11,7),
  (1,2)(3,4)(5,12)(6,11)(7,10)(8,9) ]);;
gap> G2 := Group([ (1,2,7)(3,9,4)(5,11,10)(6,8,12),
  (1,2)(3,4)(5,12)(6,11)(7,10)(8,9) ]);;
gap> G3 := Group([ (1,2,11)(3,8,7)(4,9,5)(6,10,12),
  (1,2)(3,12)(4,11)(5,10)(6,9)(7,8) ]);;
gap> G4 := Group([ (1,2,11)(3,8,7)(4,9,5)(6,10,12),
  (1,2)(3,10)(4,9)(5,8)(6,7)(11,12) ]);;
gap> cg := CosetGeometry(G, [G1,G2,G3,G4]);
CosetGeometry( Group( [ ( 3,11, 9, 7, 5)( 4,12,10, 8, 6),
  ( 1, 2, 8)( 3, 7, 9)( 4,10, 5)( 6,12,11) ] ) )
gap> SetName(cg, "Gamma");
gap> IsIncidenceGeometry(cg);
true
gap> DrawDiagram( DiagramOfGeometry(cg), "PSL211");
```

A diagram of a coset geometry

The last command (using the graphviz package `http://www.graphviz.org`) produces a nice **diagram** of the geometry.



This works for a flag-transitive incidence geometry, see the FinInG manual.

Example of a Coset Geometry - continued

```
gap> TypesOfElementsOfIncidenceStructure(cg);
[ 1 .. 4 ]
gap> IsFlagTransitiveGeometry(cg);
true
gap> flag:=RandomFlag(cg);
<Flag of coset geometry < Gamma >>
gap> Type(flag);
[ 1, 4 ]
gap> cham:=RandomChamber(cg);
<Flag of coset geometry < Gamma >>
gap> Type(cham);
[ 1, 2, 3, 4 ]
gap> ElementsOfFlag(flag);
[ <element of type 1 of Gamma>, <element of type 4 of Gamma> ]
gap> res:=ResidueOfFlag(flag);
CosetGeometry( Group( [ ( 1, 2)( 3, 4)( 5,12)( 6,11)( 7,10)( 8, 9),
  ( 1, 7)( 2, 8)( 3,11)( 4,12)( 5, 6)( 9,10) ] ) )
gap> TypesOfElementsOfIncidenceStructure(res);
[ 1, 2 ]
gap> Rank2Parameters(res);
[ [ 2, 2, 2 ], [ 2, 3 ], [ 1, 2 ] ]
```

Rank2Parameters computes the gonality, point and line diameter of cg, etc. (see FinInG manual).

Example of a Coset Geometry - continued

```
gap> ParabolicSubgroups(cg)=[G1,G2,G3,G4];
true
gap> BorelSubgroup(cg);
Group(())
gap> AmbientGroup(cg)=G;
true
gap> ElementsOfIncidenceStructure(cg,3);
<elements of type 3 of Gamma>
gap> x:=Random(ElementsOfIncidenceStructure(cg,3));
<element of type 3 of Gamma>
gap> UnderlyingObject(x);
RightCoset(Group([ (1,2,11)(3,8,7)(4,9,5)(6,10,12), (1,2)(3,12)(4,11)(5,10)(6,9
(2,3,5,4,10)(6,11,8,9,12))
gap> y:=Random(ElementsOfIncidenceStructure(cg,1));
<element of type 1 of Gamma>
gap> IsIncident(x,y);
true
gap> ShadowOfElement(cg,y,3);
<shadow elements of type 3 in Gamma>
gap> List(last);
[ <element of type 3 of Gamma>, <element of type 3 of Gamma>,
  <element of type 3 of Gamma>, <element of type 3 of Gamma>,
  <element of type 3 of Gamma> ]
gap> x in last;
true
```

Polar Spaces

Build-in polar spaces

A **polar space** is a point-line incidence geometry, satisfying the one-or-all axiom.

polar space	standard form	characteristic p	projective dimension
hermitian polar space	$X_0^{q+1} + X_1^{q+1} + \ldots + X_n^{q+1}$	odd or even	odd or even
symplectic space	$X_0Y_1 - Y_0X_1 + \ldots + X_{n-1}Y_n - Y_{n-1}X_n$	odd or even	odd
hyperbolic quadric	$X_0X_1 + \ldots + X_{n-1}X_n$	$p \equiv 3 \pmod 4$ or $p = 2$	odd
hyperbolic quadric	$2(X_0X_1 + \ldots + X_{n-1}X_n)$	$p \equiv 1 \pmod 4$	odd
parabolic quadric	$X_0^2 + X_1 X_2 + \ldots + X_{n-1}X_n$	$p \equiv 1, 3 \pmod 8$ or $p = 2$	even
parabolic quadric	$t(X_0^2 + X_1X_2 + \ldots + X_{n-1}X_n)$, t a primitive element of $\mathrm{GF}(p)$	$p \equiv 5, 7 \pmod 8$	even
elliptic quadric	$X_0^2 + X_1^2 + X_2X_3 + \ldots + X_{n-1}X_n$	$p \equiv 3 \pmod 4$	odd
elliptic quadric	$X_0^2 + tX_1^2 + X_2X_3 + \ldots + X_{n-1}X_n$, t a primitive element of $\mathrm{GF}(p)$	$p \equiv 1 \pmod 4$	odd
elliptic quadric	$X_0^2 + X_0X_1 + dX_1^2 + X_2X_3 + \ldots + X_{n-1}X_n$, $\mathrm{Tr}(d) = 1$	even	odd

Table: finite classical polar spaces

Build-in polar spaces

A hermitian polar space

```
gap> ps := HermitianPolarSpace(4,9);
H(4, 3^2)
gap> PolarSpaceType(ps);
"hermitian"
gap> AmbientSpace(ps);
ProjectiveSpace(4, 9)
gap> EquationForPolarSpace(ps);
x_1^4+x_2^4+x_3^4+x_4^4+x_5^4
```

A orthogonal polar space

```
gap> ps := HyperbolicQuadric(5,7);
Q+(5, 7)
gap> TypesOfElementsOfIncidenceStructure(ps);
[ "point", "line", "plane" ]
gap> IsIncidenceStructure(ps);
true
gap> IsIncidenceGeometry(ps);
true
```

A symplectic polar space

```
gap> ps:=SymplecticSpace(7,3);
W(7, 3)
gap> TypesOfElementsOfIncidenceStructure(ps);
[ "point", "line", "plane", "solid" ]
gap> Rank(ps);
4
gap> rho:=PolarityOfProjectiveSpace(ps);
<polarity of PG(7, GF(3)) >
gap> ForAll(Points(ps),x->x in x^rho);
true
```

Defining a polar space from matrix

```
gap> mat := IdentityMat(4,GF(11));;
gap> Display(mat);
  1  .  .  .
  .  1  .  .
  .  .  1  .
  .  .  .  1
gap> form := BilinearFormByMatrix(mat,GF(11));
< bilinear form >
gap> ps := PolarSpace(form);
<polar space in ProjectiveSpace(3,GF(11)): x_1^2+x_2^2+x_3^2+x_4^2=0 >
gap> Rank(ps);
2
gap> ps;
Q+(3, 11): x_1^2+x_2^2+x_3^2+x_4^2=0
```

Defining a polar space from form

```
gap> r := PolynomialRing(GF(5^2),4);
GF(5^2)[x_1,x_2,x_3,x_4]
gap> poly := r.3*r.2+r.1*r.4;
x_1*x_4+x_2*x_3
gap> form := QuadraticFormByPolynomial(poly,r);
< quadratic form >
gap> ps := PolarSpace(form);
<polar space in ProjectiveSpace(3,GF(5^2)): x_1*x_4+x_2*x_3=0 >
gap> rho:=PolarityOfProjectiveSpace(ps);
<polarity of PG(3, GF(5^2)) >
gap> pg:=AmbientSpace(ps);
ProjectiveSpace(3, 25)
gap> x:=Random(Points(pg));
<a point in ProjectiveSpace(3, 25)>
gap> x^rho;
<a plane in ProjectiveSpace(3, 25)>
gap> x in x^rho;
false
gap> ForAll(Points(ps),x->x in x^rho);
true
```

Generalised Polygons

Generalised Polygons

A **generalised n-gon** is a point-line geometry whose incidence graph is bipartite of diameter n and girth $2n$.

A generalised n-gon which has at least three points on every line, must have $n \in \{2, 3, 4, 6, 8\}$.

The case $n = 2$ are complete multipartite graphs, and $n = 3$ are projective planes.

The remaining cases are called **generalised quadrangles**, **generalised hexagons**, **generalised octagons**.

FinInG provides some basic functionality to deal with generalised polygons as incidence geometries.

Generalised Polygons - First Examples

A generalised quadrangle from a symplectic polar space

```
gap> gp := SymplecticSpace(3,2);
W(3, 2)
gap> IsGeneralisedQuadrangle(gp);
true
gap> IsClassicalGQ(gp);
true
gap> IsGeneralisedPolygonRep(gp);
false
```

A generalised quadrangle from a parabolic quadric

```
gap> pq:=ParabolicQuadric(4,5);
Q(4, 5)
gap> TypesOfElementsOfIncidenceStructure(pq);
[ "point", "line" ]
gap> PolarSpaceType(pq);
"parabolic"
gap> IsGeneralisedQuadrangle(pq);
true
```

The Split Cayley Hexagon

```
gap> gp := SplitCayleyHexagon(3);
H(3)
gap> IsGeneralisedHexagon(gp);
true
gap> pg:=AmbientSpace(gp);
ProjectiveSpace(6, 3)
gap> x:=Random(Points(gp));
#I  for Split Cayley Hexagon
#I  Computing nice monomorphism...
#I  Found permutation domain...
<a point in H(3)>
gap> UnderlyingObject(x);
<immutable cvec over GF(3,1) of length 7>
gap> UnderlyingObject(Random(Lines(gp)));
<immutable cmat 2x7 over GF(3,1)>
gap> ForAll(Lines(gp),l->l in Lines(pg));
true
gap> First(Lines(pg),l->not l in Lines(gp));
<a line in ProjectiveSpace(6, 3)>
```

The Ree-Tits octagon of order [2,4] as coset geometry

```
gap> LoadPackage( "AtlasRep" );
true
gap> titsgroup:=AtlasGroup("2F4(2)'");
<permutation group of size 17971200 with 2 generators>
gap> g1:=AtlasSubgroup(titsgroup,3);
<permutation group of size 10240 with 2 generators>
gap> g2:=AtlasSubgroup(titsgroup,5);
<permutation group of size 6144 with 2 generators>
gap> conj:=ConjugacyClassSubgroups(titsgroup,g1);;
gap> # Now look for the conjugate of g1 with maximal intersection
gap> g1:=First(conj, sg -> Size(Intersection(sg,g2))=2048);
<permutation group of size 10240 with 2 generators>
gap> cg:=CosetGeometry(titsgroup,[g1,g2]);;
gap> pts := Set(ElementsOfIncidenceStructure(cg,1));;
gap> lines := Set(ElementsOfIncidenceStructure(cg,2));;
gap> gp := GeneralisedPolygonByElements(pts,lines,\*,titsgroup,
              OnCosetGeometryElement);
<generalised octagon of order [ 2, 4 ]>
```

Geometry Morphisms

The Klein Correspondence

```
gap> q:=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> ps:=AmbientGeometry(Q);
Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0
gap> TypesOfElementsOfIncidenceStructure(ps);
[ "point", "line", "plane" ]
gap> l1:=Random(Lines(PG(3,q)));
gap> l2:=First(Lines(PG(3,q)),line->Dimension(Span(l1,line))=3);
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;;
gap> IsCollinear(ps,p1,p3);
true
```

The Klein Correspondence - continued

```
gap> Lines(p1);
<shadow lines in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> Size(Lines(p1));
64
gap> pg:=AmbientSpace(ps);
ProjectiveSpace(5, 7)
gap> p1 in Points(pg);
true
gap> Size(Lines(Random(Points(pg))));
2801
gap> Planes(p1);
<shadow planes in Q+(5, 7): x_1*x_6+x_2*x_5+x_3*x_4=0>
gap> NamesOfComponents(k);
[ "fun", "prefun", "invFun", "Intertwiner" ]
gap> KnownAttributesOfObject(k);
[ "Range", "Source", "Intertwiner" ]
```

Once-in-a-lifetime opportunity

Gap - Discrete Mathematics - Beach - Finite Geometry

- Gap Days Summer 2025, 25-29 August 2025
  University of Primorska, Koper, Slovenia
  `https://www.gapdays.de/gapdays2025-summer/`

- Adriatic Coast Beach Event, 30 August-6 September 2025
  Organised by Prof. Ivo and Prof. Iva
  Rampin Lecture Hall, University La Spiaggia, Slovenia
  (talk to the locals Russ and George for more details)

- 12th PhD Summer School in Discrete Mathematics, 7-13 September 2025
  University of Primorska, Koper, Slovenia
  `https://conferences.famnit.upr.si/event/33/overview`

- Finite Geometry & Friends, 15-19 September 2025
  VUB, Brussels, Belgium
  `http://summerschool.fining.org/`