

# Computing (with) affine crystallographic groups

Franz Gähler

Mathematics, University of Bielefeld

`gaehler@math.uni-bielefeld.de`

Workshop on Mathematical Crystallography

2–6 November 2011

University of the Philippines, Diliman

# Why computational crystallographic group theory?

The number of (classes of) crystallographic groups grows very rapidly with increasing dimension:

	1	2	3	4	5	6
No of Q-classes	2	10	32	227	955	7'103
No of Z-classes	2	13	73	710	6'079	85'308
No of affine classes	2	17	219	4783	222'018	28'927'915
No of Bieberbach groups	1	2	10	74	1'060	38'746

This is not feasible for printed tables!

# Why **computational** crystallographic group theory?

Printed tables are very static – once printed, it is hard to correct errors or add new information.

An electronic data base would be better, but not good enough:

- ▶ want to use those groups on the **computer**: study group actions, orbits, stabilizers, subgroup structure, etc., in **any dimension**
- ▶ want to have available all the group theoretic algorithms offered e.g. by GAP
- ▶ want to choose flexibly the **basis and origin** (setting) of the group
- ▶ better compute properties of the groups we are interested in **on the fly!**

# Space Groups

An  $n$ -dimensional space group  $S$  is a discrete subgroup of the group of Euclidean motions of  $\mathbb{R}^n$ , such that its subgroup of pure translations is a freely abelian normal subgroup with rank  $n$  and with finite index in  $S$ .

We thus have an exact sequence of groups and homomorphisms

$$0 \rightarrow T \rightarrow S \rightarrow P \rightarrow \mathbb{1}$$

$S$  acts on its translation subgroup  $T$  via the conjugation action  $t \rightarrow s * t * s^{-1}$ , which defines a representation of the point group  $P$  acting on the lattice of translations.

# Space Group Elements

Space group elements are conveniently expressed by **augmented matrices** (or affine matrices) of dimension  $n + 1$ :

$$\left( \begin{array}{c|c} M & t \\ \hline 0 & 1 \end{array} \right) \quad \text{or} \quad \left( \begin{array}{c|c} M & 0 \\ \hline t & 1 \end{array} \right)$$

The first matrix acts on column vectors  $\begin{pmatrix} x \\ 1 \end{pmatrix}$  from the left, and the second on row vectors  $(x, 1)$  from the right. They thus describe actions  $x \rightarrow Mx + t$  and  $x \rightarrow xM + t$ , respectively.

Different communities (crystallographers, mathematicians) have different conventions. One just has to make a consistent choice.

# Equivalence Classes I

Two space groups are isomorphic iff they are conjugate in the Euclidean group (affinely conjugate). This equivalence relation defines a **space group type** (or affine class). Space groups of the same type differ in the choice of **basis and origin**, with respect to which they are expressed.

As there are so many space groups, we also need a coarser classification, which only looks at the point groups. Expressed with respect to a lattice basis, the point group  $P$  consists of **integer matrices**.  $P$  can be regarded as a **finite unimodular group**, i.e. a finite subgroup of  $GL(n, \mathbb{Z})$ .

# Equivalence Classes II

Two space groups are in the same  **$\mathbb{Z}$ -class** (arithmetic class), if their point groups are conjugate subgroups of  $GL(n, \mathbb{Z})$ .

Two space groups are in the same  **$\mathbb{Q}$ -class** (geometric class), if their point groups are conjugate subgroups of  $GL(n, \mathbb{Q})$ .

For any point group  $P$ , let  $C(P)$  be the **cone of all positive definite quadratic forms** (given by symmetric matrices  $Q$ ) which are invariant under the following action of  $P$ :

$$Q \rightarrow g^{\text{tr}} \cdot Q \cdot g$$

For finite  $P$ ,  $C(P)$  is always non-empty.

# Equivalence Classes III

Let  $B(P)$  be the maximal subgroup of  $GL(n, \mathbb{Z})$  leaving the forms in  $C(P)$  invariant.  $B(P)$  is called the **Bravais group** of  $P$ . Space groups whose point groups have the same Bravais group are of the same **Bravais type**.

Roughly, the Bravais group is the symmetry of the lattice of the space group. The dimension of  $C(P)$  equals the **number of parameters** of the lattice.

# Finiteness of Matrix Groups

Let  $G$  be a unimodular group, given by a set of generating integer matrices. Is it **finite or not**? This is not obvious!

The following theorem (on the **Minkowski kernel**) may help:

Let  $h_p$  the natural homomorphism from  $G \subset GL(n, \mathbb{Z})$  to  $GL(n, \mathbb{F}_p)$ , where  $p$  is a prime number. If  $p > 2$ ,  $G$  is finite if and only if the kernel of  $h_p$  is trivial. If  $p = 2$ ,  $G$  is finite if and only if the kernel of  $h_2$  has exponent 2 (all elements of order 2, and thus abelian).

But: the Minkowski kernel is a subgroup of  $G$ , which may be infinite. . .

# Orbit-Stabilizer Algorithm

Elements  $g \in G$  act on  $h_p(G)$  via multiplication with  $h_p(g)$ . The Minkowski kernel is the stabilizer of  $\mathbb{1} \in GL(n, \mathbb{F}_p)$  under this group action. There are algorithms for this task, even if  $G$  is infinite:

- ▶ generate the orbit through  $p_0 = \mathbb{1} \in GL(n, \mathbb{F}_p)$ , by acting with the generators of  $G$  on points in the orbit
- ▶ with each point  $p_i$ , store also the element  $g_i \in G$  which maps  $p_0$  to  $p_i$
- ▶ if  $p_j = g(p_i)$  is already in the orbit, add  $g_j^{-1}gg_i$  to the generators of the stabilizer of  $p_0$

A **theorem of Schreier** says that generators of the full stabilizer of  $p_0$  are obtained in this way. This algorithm works as long as the orbit is finite.

# Minkowski-Kernel Algorithm

The Minkowski kernel algorithm works well in small dimensions, and for cyclic groups of all dimensions. For large groups in higher dimensions, the number of generators can explode, and performance degrades.

Alternative: A unimodular group  $G$  is finite if and only if there exists a positive definite quadratic form  $Q$  invariant under  $G$ :

$$g^{\text{tr}} \cdot Q \cdot g = Q \quad \forall g \in G$$

If  $G$  is finite,  $Q$  can be chosen as

$$Q = \frac{1}{|G|} \sum_{g \in G} g^{\text{tr}} \cdot g$$

# Random Finiteness Test I

We cannot perform the sum over all group elements, before we know the group is finite. But we can generate a **random sequence of elements**, and generate **approximations** to an invariant form  $Q$ .

For each element in the random sequence, we test whether it is of finite order. If not, the group is not finite. From the approximation to  $Q$ , we try to guess the real  $Q$ , by approximating its entries by the simplest rationals in the range of the last few iterations. This guessed  $Q$  is tested for invariance.

**With probability 1**, we will either find sooner or later an element of infinite order, or an invariant form  $Q$ , with which we can test the finiteness of the group, by checking positive definiteness.

# Random Finiteness Test II

This random algorithm, due to Robert Beals, works surprisingly well even for large groups in large dimensions. Its **result is exact**, only the runtime is random (but typically short).

**Rational matrix groups** first have to be converted to integer ones. If this is not possible, they are not finite.

We repeatedly add the image of random non-integer group element to  $\mathbb{Z}^n$ , and express the group with respect to the new basis. With probability 1, we will either find an **element of infinite order**, or an **invariant lattice**, with which the group is converted to an integer form.

# Support for Crystallographic Groups in GAP

The computer algebra system GAP ([www.gap-system.org](http://www.gap-system.org)) provides a wealth of group-theoretic algorithms, but does not support crystallographic groups directly.

Support for crystallographic groups has been added by **extension packages**:

- ▶ Cryst – support for **affine crystallographic groups** (space groups and subgroups thereof)
- ▶ CrystCat – **catalogue of crystallographic groups** of dimension up to 4. These groups make use of the methods provided by the Cryst package
- ▶ Carat – interface to certain routines in the CARAT package (<http://wwwb.math.rwth-aachen.de/carat/>) needed by Cryst

# CARAT

The CARAT package (<http://wwwb.math.rwth-aachen.de/carat/>) of J. Opgenorth, W. Plesken, and T. Schulz provides algorithms especially for finite unimodular groups, which prove useful for the Cryst package. GAP interface routines are provided by the Carat package, so that Cryst can directly access these CARAT routines.

For  $G$  a finite unimodular group, CARAT can determine

- ▶ the **normalizer and centralizer** of  $G$  in  $GL(n, \mathbb{Z})$
- ▶ the **Bravais group**, Bravais subgroup, and Bravais supergroup of  $G$
- ▶ list  **$\mathbb{Z}$ -class representatives** of the  $\mathbb{Q}$ -class of  $G$
- ▶ determine a conjugator between two groups in the same  $\mathbb{Z}$ -class

# Enabling Affine Crystallographic Groups in GAP

**Affine crystallographic groups**, such as space groups, are defined by a set of generating **augmented matrices**. Any set of generators is fine, and so is any choice of origin and basis, in any dimension.

In dealing with such a group  $S$ , we will make extensive use of the exact sequence

$$0 \rightarrow T \rightarrow S \rightarrow P \rightarrow \mathbb{1}$$

We first have to construct the **homomorphism  $h$  to the point group  $P$** . This homomorphism just cuts out the linear part of the augmented matrix. So, the point group is generated by all the linear parts of the generators of  $S$ .

# Computing the Translation Lattice

The translation subgroup of  $S$  is actually the **kernel of the homomorphism  $h$**  to the point group  $P$ . So, we could use a stabilizer computation like for the Minkowski kernel.

Alternatively, we can determine a **presentation** of  $P$  for the generators  $h(s_i)$ , where  $s_i$  runs over the generators of  $S$ . The defining relations of the presentation, which are words in the generators of  $P$ , evaluate to  $\mathbb{1} \in P$ . When lifted back to  $S$ , we obtain a pure translation in  $S$ .

Computing a presentation for a finite group is a standard task in GAP.

The translations obtained in this way **generate the whole translation group**.

Behind the scene, the two algorithms perform essentially the same task.

# Standard Basis of Translation Lattice

In order to check whether two lattices are equal, it is convenient to use a basis of the lattice **which is unique**, i.e., depends only on the lattice, not the original basis, with which the lattice was defined.

Note: an LLL reduced basis does not have this property!

If the lattice is spanned by the rows of an integer matrix, this is achieved by the basis given by the rows of its **Hermite normal form**.

If the matrix with the basis vectors is **rational**, one multiplies it with the least common multiple  $d$  of the denominators of all its entries, takes the Hermite normal form of the resulting integer matrix, and divides it by  $d$ .

# Membership Test

A central task is the **membership test**: determine whether a augmented matrix  $A$  is an element of a given affine crystallographic group,  $S$ .

First, we check whether the linear part  $M$  of  $A$  is contained in the point group  $P$  of  $S$ . If so, we have to **lift  $M$  to some preimage** under the homomorphism  $h$ . This is a standard task for GAP homomorphisms;  $M$  is expressed in the generators of  $P$ , representative preimages of which are known.

Finally, we check whether the so determined preimage of  $M$  **differs from  $A$  by a lattice translation**. For that, we need the translation lattice.

# Applications of Membership Test

Having a membership test opens up a lot of possibilities, for which the standard GAP methods can be used:

- ▶ determine whether one group is a subgroup of another
- ▶ determine whether two groups are equal
- ▶ determine whether two cosets are equal ( $Hg = Hg' \Leftrightarrow gg'^{-1} \in H$ )

All this works independently of the choice of generators, basis, and origin.

Internally, often a lattice basis is used, but this is not visible to the user.

A change of basis or origin is simply achieved by conjugating the group with a suitable augmented matrix.

# Orbits and Stabilizers of Space Group Actions

Having a proper membership test, the **orbit-stabilizer algorithm** can be applied to various group actions:

- ▶ enumerate cosets of a finite index subgroup
- ▶ enumerate the conjugacy class of a subgroup
- ▶ determine the normalizer and the centralizer of a subgroup
- ▶ determine orbits of points in Euclidean space, or affine subspaces of Euclidean space (modulo lattice translations)

Group actions on different kinds of objects can be flexibly defined, and the orbit-stabilizer algorithm is then ready to be used. One only must make sure that the **orbit is finite**.

# Normalizers I

Various **normalizers** of a space group  $S$  are available as well, which partly build upon the corresponding functions for point groups provided by CARAT.

The normalizer of  $S$  in the full translation group requires only the solution of a linear system of equations.

The problem here is, that there may be a subspace of **continuous translations** leaving  $S$  invariant. Such groups are not **finitely generated**, and cannot be represented as GAP groups. Only the discrete part of the translation normalizer can be used as a GAP group.

# Normalizers II

Using the translation normalizer and the normalizer of the point group in  $GL(n, \mathbb{Z})$ , the **normalizer in the affine group** can be constructed – provided the translation normalizer is discrete.

Using the affine normalizer, subgroups of a space group can be classified into affine equivalence classes.

Building upon the CARAT function returning the **conjugator between two finite unimodular groups**, a function has been implemented which returns a **conjugator between two space groups**, if they are isomorphic.

# Determining Space Groups for a Given Point Group I

Recall the exact sequence

$$0 \rightarrow T \rightarrow S \rightarrow P \rightarrow \mathbb{1}$$

Here,  $T = \mathbb{Z}^n$  and  $P \subset GL(n, \mathbb{Z})$  are given, and we seek possible solutions for  $S$ . In a sense, this is the inverse problem of determining  $T$  from  $S$ .

Every word in the generators of  $S$ , which is mapped to the identity of  $P$ , is a pure translation. This **imposes constraints** on the translational parts of the generators of  $S$ , if this translation shall be in  $\mathbb{Z}^n$ .

Specifically, every relation in the generators of  $P$  must lift to a word in the generators of  $S$ , which evaluates to a pure translation.

## Determining Space Groups for a Given Point Group II

Writing the generators of  $S$  as  $(g_i, t_i)$ , where the  $g_i$  are the generators of  $P$ , we get from each relation a homogeneous equation (with coefficients in  $\mathbb{Z}$ ) which the  $t_i$  must satisfy modulo  $\mathbb{Z}$ . Taking all defining relations of  $P$  together, we have to solve a system  $Mx = 0 \pmod{\mathbb{Z}}$ , with  $x$  the concatenation of the  $t_i$ .

This is best done by bringing  $M$  to **Smith normal form**:  $M = ADB$ , with  $A, B \in GL(n, \mathbb{Z})$  and  $D$  diagonal. We then solve  $Dy = 0 \pmod{\mathbb{Z}}$  for  $y = Bx$ , from which we obtain  $x$ .

This provides representatives of all space group types – but not all are inequivalent.

# Equivalent Space Groups

If a space group is **conjugated with a translation  $t$** , the translational part of  $(g_i, t_i)$  is changed by  $g_i t - t$ . Space groups related by such a change of origin represent the same space group type. Among the solutions of the **compatibility equations**, we have to select a set of inequivalent solutions. This set of solutions is then discrete.

Similarly, the resulting space groups may be equivalent under a change of lattice basis. As we already have chosen a fixed matrix representation of the point group, it remains to check for equivalences under conjugation with elements of the **normalizer of  $P$  in  $GL(n, \mathbb{Z})$** . Generators of this normalizer can be obtained with CARAT, and a standard orbit calculation will find the equivalences.

## Side Remark: Group Cohomology I

Solving the compatibility equations for the  $t_i$ , modulo changes induced by a shift of origin, is actually a **group cohomology** computation.

Adding a translational part to the generators of  $P$  can be regarded as a map  $P \rightarrow \mathbb{Q}^n/\mathbb{Z}^n$ . These maps form an abelian group under addition, the group of 1-cochains. The compatibility equations are the conditions for a 1-cochain to be closed (being a 1-cocycle). Finally, two 1-cocycles are equivalent, if they differ by an exact 1-cochain – one which arises through a change of origin.

So, we have classified space group types by  $H^1(P, \mathbb{Q}^n/\mathbb{Z}^n)$

(to which equivalence under change of lattice basis must still be applied).

## Side Remark: Group Cohomology II

**Group extensions** are usually classified by a **second cohomology group**.

However, we have here a short exact sequence of coefficient groups,

$$0 \rightarrow \mathbb{Z}^n \rightarrow \mathbb{Q}^n \rightarrow \mathbb{Q}^n/\mathbb{Z}^n \rightarrow 0$$

which gives via the snake lemma a long exact sequence

$$\rightarrow H^1(P, \mathbb{Q}^n) \rightarrow H^1(P, \mathbb{Q}^n/\mathbb{Z}^n) \rightarrow H^2(P, \mathbb{Z}^n) \rightarrow H^2(P, \mathbb{Q}^n) \rightarrow$$

Since  $\mathbb{Q}^n$  is divisible by the order of  $P$  (which is finite),  $H^k(P, \mathbb{Q}^n) = 0$ , and  $H^1(P, \mathbb{Q}^n/\mathbb{Z}^n) \cong H^2(P, \mathbb{Z}^n)$ .

# Computing Maximal Subgroups of Space Groups

There are infinitely many maximal subgroups, so we have to restrict their index. Maximal subgroups have  **$p$ -power index**, for some prime  $p$ .

So, we fix a prime  $p$ .

There is always the subgroup  $T_p$  of all  $p$ -fold translations. All maximal subgroups with  $p$ -power index must contain  $T_p$ .

As  $T_p$  is normal, we can look at the quotient  $S/T_p$  (which is finite), determine its maximal subgroups, and then add back the translations in  $T_p$ .

Actually, GAP returns conjugacy class representatives of maximal subgroups. The other maximal subgroups in a conjugacy class can be obtained with an orbit-stabilizer algorithm.

# Maximal Subgroups of Space Groups

Maximal subgroups of space groups have a special structure: they are either:

- ▶ **klassengleich** (class equal) – have the same point group ( $\mathbb{Q}$ -class), but a coarser translation lattice
- ▶ **translationengleich** (translation equal) – have a smaller point group, but the same translation lattice

For space groups  $S$ , the index of a subgroup  $U$  is always

$$[S : U] = [P(S) : P(U)] [T(S) : T(U)].$$

For maximal subgroups, one of the two factors is 1.

# Application: Checking the Data of IT Vol. A1

The International Tables of Crystallography comprise now a volume A1 with tables of **subgroups of index up to 4** of all plane groups and 3d space groups. These were computed by Y. Billiet, M. Aroyo and H. Wondratschek. Before publication, these tables have been thoroughly checked with GAP.

The checks directly parse the  $\text{\LaTeX}$ sources of the book, translating the data (entered with special  $\text{\LaTeX}$ macros) into GAP data structures. This makes sure that no new errors are introduced in typesetting. The tests can be run automatically after each change of the sources.

# Subgroup Table Tests Part 1

First, some tests on each **individual subgroup** were performed:

- ▶ whether it is a subgroup, and has the given index
- ▶ whether the listed transformation maps it to the preferred setting of the given space group type (and thus, whether the given space group type is correct)
- ▶ whether the listed transformation maps the generators to those of the given space group type in the preferred setting, in the same order

## Subgroup Table Tests Part 2

In a second step, all subgroups of index up to 4 were **recomputed by GAP**, and classified into conjugacy classes. It was verified that:

- ▶ every subgroup is listed exactly once
- ▶ classification into klassengleiche and translationengleiche is correct
- ▶ classification into conjugacy classes is correct

The tables include also parametrized subgroup series. These data could not be checked with GAP, as GAP does not support parametrized data (at least not without much extra work).

# Wyckoff Positions

A space group  $S$  acts naturally on Euclidean space. Each point has a stabilizer in  $S$  (**site symmetry**). Generically, this is the trivial group, but some points have a non-trivial stabilizer.

For many applications, it is interesting to classify points according to their stabilizer. Which points have the same, which have conjugate stabilizers? Where are the points with site symmetry in a given conjugacy class?

The points whose stabilizers are in a single conjugacy class form a so-called **Wyckoff position**. This is an equivalence class of points!

# Invariant Affine Subspaces

Points having the **same stabilizer** form an open subset of some **affine subspace**  $A$  (the remaining points in  $A$  have an even larger stabilizer).

A point  $y$  has the same stabilizer as  $x$ , if the translation by  $x - y$  commutes with that stabilizer. If this is the case, translations in a whole linear subspace commute.

Example: Suppose  $x$  has a mirror plane in its stabilizer. Where are the other points having the same mirror plane in their stabilizer?

We have to classify **affine subspaces in special position**, namely those which have a non-trivial, point-wise stabilizer.

# Lattices of Affine Subspaces

Stabilizers can be determined with an **orbit-stabilizer algorithm**. But there is one problem: space group orbits are **infinite!** We have to consider orbits modulo lattice translations.

Solution: consider **lattices of affine subspaces**, which are invariant under all lattice translations, and study their space group orbits. These are now finite.

In order to compare and identify affine subspace lattices, we need a **canonical representative** affine subspace, and a **canonical representation** of that representative.

# Canonical Representation

A **representative affine subspace** of a lattice of such spaces is given by a basis  $B$  spanning the space, and a representative point  $x$  in the space.

As the space is rationally oriented, we choose for  $B$  the **canonical basis** of the maximal lattice spanning the space.

Next, we choose a canonical complement to the lattice spanned by  $B$ . Let  $B'$  be the canonical basis of that complement.

As **canonical representative point**  $x$  we now choose the one in the span of  $B'$ , reduced by the vectors from  $B'$  (inside the unit cell spanned by  $B'$ ).

# Space Group Action on Affine Subspace Lattices

Having unique representatives for lattices of rational affine subspaces, we can now compare them, let space groups act on them, compute space group orbits and stabilizers.

We act on the representative space, and **reduce the image to normal form**.

But which are the ones having a **non-trivial stabilizer**?

Let  $U$  be a subgroup of the point group, and  $G$  its full preimage in the space group (containing the full lattice).

We look for affine subspace lattices invariant under  $G$ .

# Solving for the Affine Subspace

For convenience, we work with a lattice basis, so that the lattice is  $\mathbb{Z}^n$ .

Points left invariant by  $G$  modulo  $\mathbb{Z}^n$  must satisfy the equations

$$(M_i - \mathbb{1})x = -t_i \quad (\text{mod } \mathbb{Z}^n)$$

where  $(M_i, t_i)$  runs over the generators of  $G$ . So, we have to solve a inhomogeneous system  $Mx = b$  modulo  $\mathbb{Z}^n$ .

This is done as in the homogeneous case: bring  $M$  into **Smith normal form**,  $M = ADB$ , with  $A, B \in GL(n, \mathbb{Z})$ ,  $D$  diagonal, solve  $Dy = b' = A^{-1}b$  modulo  $\mathbb{Z}^n$ , and get  $x = B^{-1}y$  for all solutions.

# Solving for the Affine Subspace

**Note:** if  $d_i = 0$ , but  $b'_i \neq 0$  for some  $i$ , no solutions exist. If  $d_i = b'_i = 0$ , the whole span in that direction is a solution. Those directions span the affine subspace, and the other components determine the representative point through which the affine subspace passes.

There are several solutions in general (all parallel). For each of those affine subspace lattices, the **real stabilizer may be bigger**. It can be determined with the orbit-stabilizer algorithm.

Finally, the stabilizer of a representative subspace is determined:

if  $M_i x - t_i = x + \ell_i \quad \forall x \in A$ , then  $(M_i, t_i - \ell_i)$  fixes  $A$  point-wise.

# Algorithm: Determining Wyckoff Positions

All Wyckoff positions of a space group  $S$  can thus be obtained as follows.

- ▶ compute representatives of all subgroup conjugacy classes of the point group of  $S$ , and get their full preimage  $U$  in  $S$
- ▶ for each such  $U$ , get all affine subspace lattices invariant under  $U$
- ▶ eliminate multiple subspace lattices in the same  $S$ -orbit
- ▶ keep only the ones with orbit length  $[S : U]$ ; the other will show up again for a larger  $U$ .
- ▶ for each of the remaining ones, determine the point-wise stabilizer of a representative subspace

# Alternative Algorithm for Wyckoff Positions

For large groups, computing the full subgroup lattice is a heavy burden. Moreover, not all of these subgroups are needed.

For small dimensions, an algorithm due to Ad Thiers performs better.

We start by determining the affine subspace lattices left invariant by the full preimages of the **cyclic subgroups** of the point group.

Then we determine their orbits and their real stabilizers, and get a first set of Wyckoff positions.

The remaining Wyckoff positions, having larger stabilizers, must be **intersections** of the ones we already have.

# Intersections of Affine Subspace Lattices

The intersection of two affine subspace lattices is again a union of affine subspace lattices. To determine them, we have to solve an inhomogeneous system of equations modulo integers:

$$x_1 B_1 + r_1 = x_2 B_2 + r_2 \quad (\text{mod } \mathbb{Z})$$

By taking **all possible intersections**, all affine subspace lattices in special position can be constructed, starting from the higher-dimensional ones towards the lower-dimensional ones.

Care must be taken, that only one per orbit is constructed by intersection; the others can be obtained by the space group action.

# Color Groups

The Cryst package offers some limited support for **color groups** and color symmetry, especially **color space groups**.

The approach is a bit different from that of the Manila group here.

We do not start with a colored pattern, and determine its color group.

Rather, we start with a color group, i.e., a group  $S$  and a finite index subgroup  $U$ , identifying the latter as the **color stabilizing group**.

Colors are then associated with cosets of  $U$ . Elements of  $S$  are assigned a color, and since  $S$  acts by permutation on the cosets of  $U$ , each element of  $S$  is also assigned a **color permutation**.

# Action of Color Groups

A color group can act on a set of objects (points, lines, tiles, ...) which is invariant under  $S$ . For each orbit, the **color of one object** can be chosen; the remaining colors are determined by the action of the color group. Only **perfect colorings** are obtained in this way.

There is one subtlety: If an object has a **non-trivial stabilizer** (disregarding color), it may be assigned more than one color.

In such a case, these colors become equivalent, and may be replaced by an **equivalence class of colors**. If several orbits are involved, this must be done in a consistent way.

# References I

Some of the algorithms implemented in the Cryst package are described here:

F. Gähler, B. Eick, W. Nickel, Computing maximal subgroups and Wyckoff positions of space groups, *Acta Cryst A* **53**, 467–474 (1997).

A general reference on N-dimensional crystallography is:

R. L. E. Schwarzenberger, N-dimensional crystallography, *Research notes in mathematics*, **41**, Pitman, 1980.

The four-dimensional crystallographic groups are tabulated here:

H. Brown, R. Bülow, J. Neubüser, H. Wondratschek, H. Zassenhaus, *Crystallographic Groups of Four-dimensional Space*. John Wiley, New York 1978.

# References II

An elementary treatment of the connection between space groups and group cohomology is given here:

H. Hiller, Crystallography and Cohomology of Groups, Amer. Math. Monthly **93**, 675–779, 1986.

The algorithms implemented in CARAT are described here:

J. Opgenorth, W. Plesken, T. Schulz, Crystallographic algorithms and tables, Acta. Cryst A **54**, 517–531 (1998).

Web pages:

<http://www.gap-system.org>

<http://www.math.uni-bielefeld.de/~gaehler/gap/packages.php>

<http://wwwb.math.rwth-achen.de/carat/>