About Scaling Topologies in AuToGraFS

Matthew A. Addicoat, Damien E. Coupry, and Thomas Heine

School of Engineering and Science, Jacobs University Bremen Campus Ring 1, 28759 Bremen, Germany

E-mail: m.addicoat@jacobs-university.de

Introduction

An inherent problem with building a molecular framework beginning with the topology is that, by definition, the topology has no inherent notion of geometry. Topologies sourced from the RCSR[1] are (those with embed type of 1, see http://rcsr.net/about.html for more detailed discussion regarding embed types) typically “maximum symmetry embeddings in which edge lengths are constrained to be equal to unity and the volume is maximized subject to that constraint”. This has a few effects when one is attempting to build a molecular framework as it is necessary to somehow (re-)create this geometric information. AuToGraFS attempts to construct this information using the method described below:

Consider the **soc** topology: Figure 1 shows one 6-connected object and one of its adjacent 4-connected objects. The black line directly connects the centre-of-mass (COM) of each object. From the description of the topology embedding above, we know that in the coordinates provided in the topology (each “atom" in a systre file indicates the position of the COM of an object), this distance is equal to unity. A distance of 1.0Å is clearly too small for an actual molecule, so we need to scale it somehow.

A reasonable guess at a scaling factor is to use the sum of the sizes of the two objects being connected, and this is what AuToGraFS does. The size of each object is defined as the distance from the COM of the object to the dummy atom, the sizes for the two objects in our **soc** topology are shown as green lines in Figure 2 below. It should be apparent that when the two COM$\rightarrow$dummy atom vectors lie along the COM$\rightarrow$COM vector, that this will be a very good guess indeed. However, the quality of the guess will degrade as the deviation between these vectors increases. It can be seen in Figure 2 that the two dummy atoms are not coincident, which they would be in the case of the COM$\rightarrow$dummy atom vectors being coincident with the COM$\rightarrow$COM vector.

A useful point to note here is that dummy atoms for the molecular building blocks supplied in AuToGraFS internal component databases are placed 0.75Å from their connecting atom. This means that a carbon-carbon bond formed with dummy atoms coincident will have a bond length of 1.5Å - which is quite a reasonable value (*cf* 1.54Å for a C-C single bond).

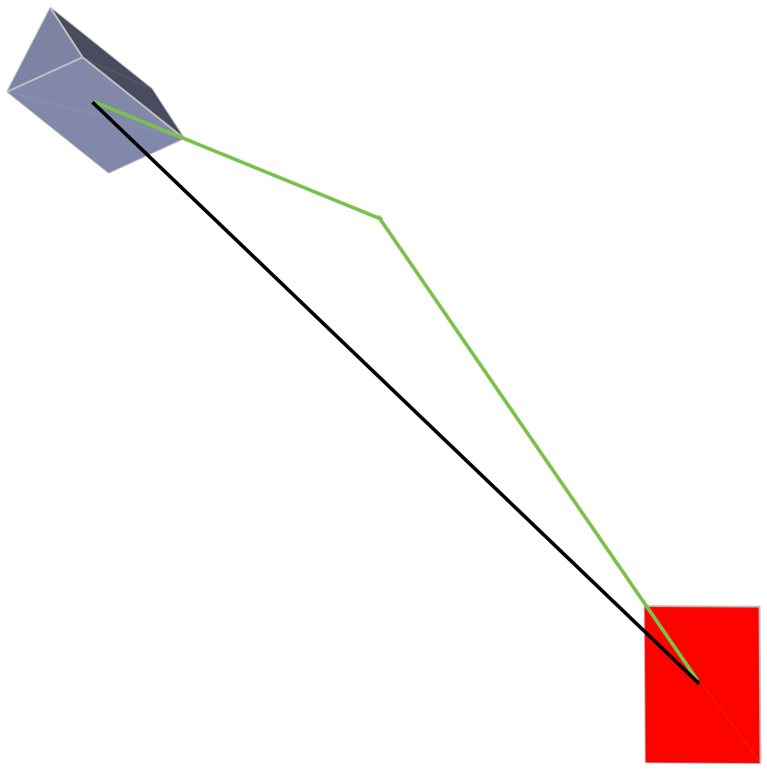

Figure 1: A single 6-connected object and adjacent 4-connected object extracted from the **soc** topology. The black line connects the COM of the two objects. The two green lines extend from the COM of each object, through a corner of the object. The point where the two green lines meet corresponds approximately to the centre of the bond that will be formed between the two objects.

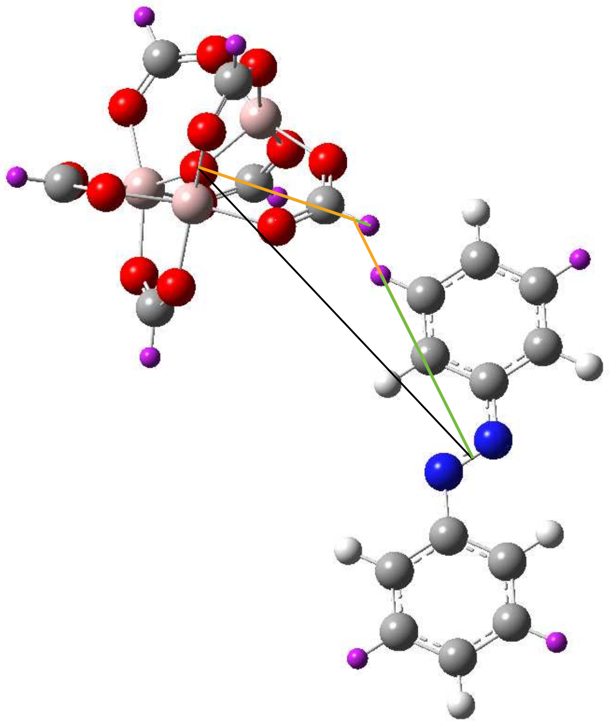

Figure 2: The same 6-connected object and adjacent 4-connected object as in Figure 1, but showing molecular coordinates. The black line connects the COM of the two objects and its length is equal to the sum of the two green lines. The orange lines follow the same vectors as the green lines and indicate where the two vectors meet.

It is, however, unfortunately clear that the geometry around newly formed bonds can be quite poor. The two carbon atoms that are being bonded in Figure 2 begin approximately 2.7Å apart, which is clearly less than optimal. It is here that the approach of maintaining the molecular connectivity from fragments (building blocks) through to the complete framework is immensely useful. Despite the poor geometry, AuToGraFS defines a bond between these two carbon atoms. Using the `noautobond` and `fix` options within GULP[2,3] when optimising the structure will preserve the bond and correct the geometry for us.

At this point, let us take stock of the geometric problems we have:

1. As described above, simply scaling the unit cell provided by the RCSR is not guaranteed to

place the two atoms that will form a new bond between two building blocks in a reasonable geometry to actually from that bond.

2. The building blocks themselves, while occupying topologically equivalent sites, are nearly always distorted from perfect molecular symmetry (e.g. the M_3O building block above does not exist in perfect D_{3h} symmetry in the framework and the linker is significantly non-planar in the actual framework. This lack of internal distortion in the initial fragments is likely to exacerbate problem 1.

3. Simple scaling of the topological geometry does not allow for the cell vectors to expand at different rates. Where the molecular framework requires such differential expansion (and it is not done), this is the major cause of problem 1.

Considering **Problem 2** in more detail, it is clear that any molecular framework based on a topology, can have *at most* the space group symmetry of that topology, but will likely have less. It is also expected that the symmetry of the resulting framework will change according the (combination) of components used to build it. The corollary of this is that a given building block is likely to distort in different ways when used in different frameworks. A clear example of this is given by building blocks containing adjacent phenyl rings (e.g. biphenyl, triphenyl and similar pyridines etc.), that may be exactly or nearly planar when used in a planar (stacked) COF, but have a significant inter-ring dihedral angle when used in a 3D framework such as an IRMOF. Attempting to create, store and index many different versions of each linker would be a combinatorial nightmare and would create the further problem of needing to translate and align such distorted building blocks appropriately. For these reasons, AuToGraFS deals with highly symmetric building blocks and, accepting that this choice reduces to some degree the quality of the resultant framework structure, then fixes that structure.

Problem 3 is highly variable in magnitude, dependent on the specific topology. Considering again the square octahedron, **soc** topology, shown in another form in Figure 3, it can be seen that the mutual constraints of fitting the purple 6-connected unit with the red 4-connected unit means that uniform expansion of the unit cell is actually desired. The corollary in this case, is that there is some maximum degree to which the side lengths of the red rectangle may differ from each other. A very long and skinny rectangle will form highly distorted bonds between building blocks (grey lines) and internal distortion of the building blocks may not be sufficient to correct for this.

A further note about the 6-connected units in this topology. These units sit on the corners of a regular cube and connect along the edges that pass through it. An octahedron defined in such a way is shown in Figure 4. Choosing any two opposite faces allows one to visualise the octahedron as a *trigonal antiprism*. Yet the M_3O building block that we typically wish to align in this position is a *trigonal prism*! AuToGraFS has several alignment routines for 6-connected objects and, as a general rule, the subtype defined in the topology is the one that is used throughout. In this particular topology, AuToGraFS transforms the trigonal antiprism into a trigonal prism.

Returning to considering **Problem 3**, and now exmaining the **chs-1** topology detailed in the text, the UHM-7 MOF[4] from which this topology was derived, has the lattice parameters $\alpha = \beta = 28.7527$, $\gamma = 35.4316$, with all angles equal to 90°. So, for the real structure, $\alpha/\gamma = 0.811$. On the other hand, in the **chs-1** topology, $\alpha = \beta = 4.14300$, $\gamma = 4.66569$, yielding $\alpha/\gamma = 0.888$. This scaling error mainly manifests itself in the bonds between triangles (as defined by the topology, as discussed in the text AuToGraFS adds in pseudolinear Si building blocks here in order to address the problem of halving a centrosymmetric building block), which are coplanar in the topology, but approximately 120° in the molecular structure, where this is a C-Si-C angle. After optimisation in GULP, using UFF4MOF[5] (more on this below), the final lattice parameters are $\alpha = \beta = 29.49$ and $\gamma = 35.93$ leading to $\alpha/\gamma = 0.821$, very close to those of the original structure (both α and γ are slightly overestimated).

All three of these problems are interrelated, and AuToGraFS was designed around the philoso-

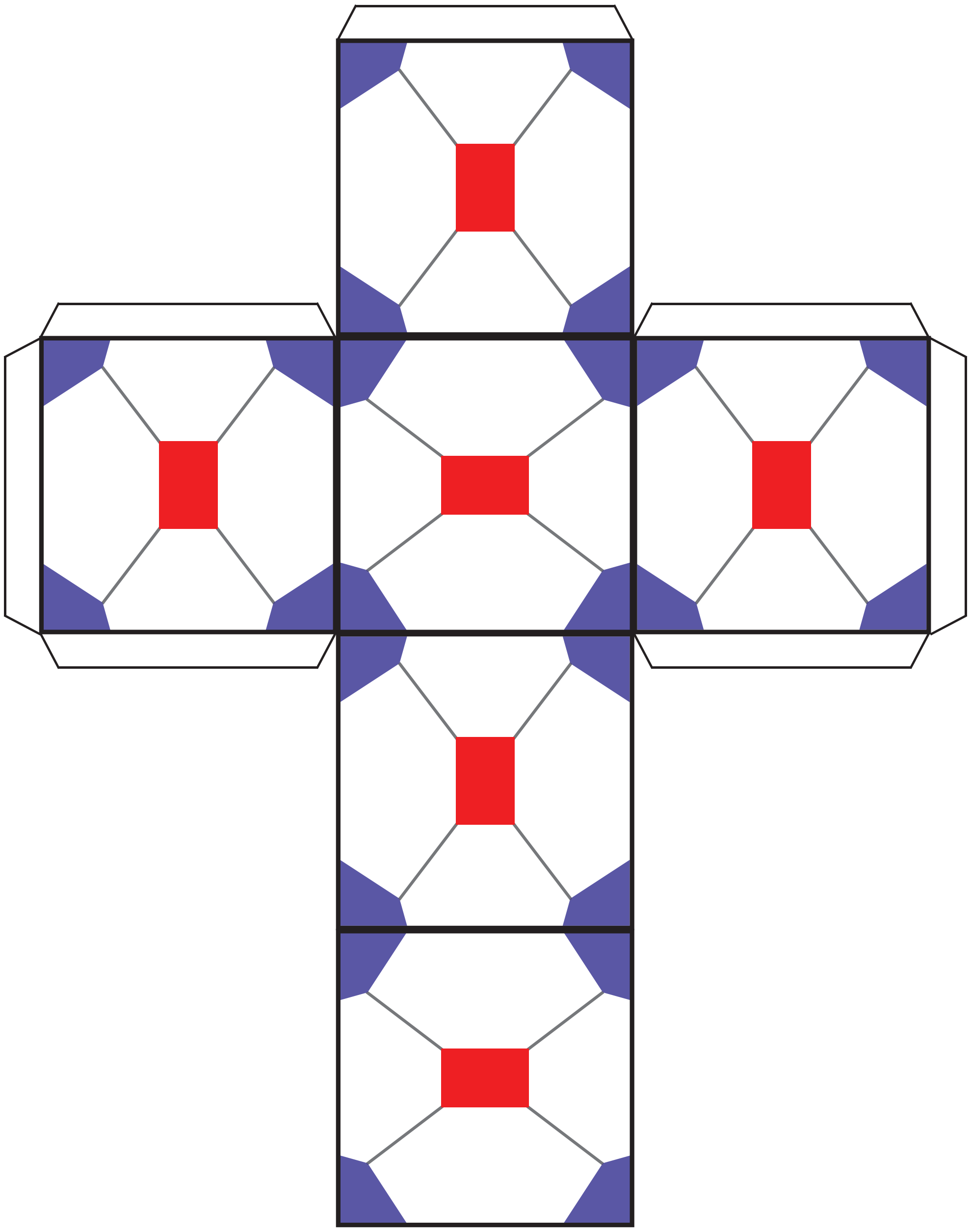

Figure 3: One cube of the **soc** topology, actually as a cube! Print this page, cut out the net, and assemble with sticky tape or glue. Decorate your desk or use it as a projectile! If you use it as a projectile, use common sense and be careful not to injure anybody. This is most important.

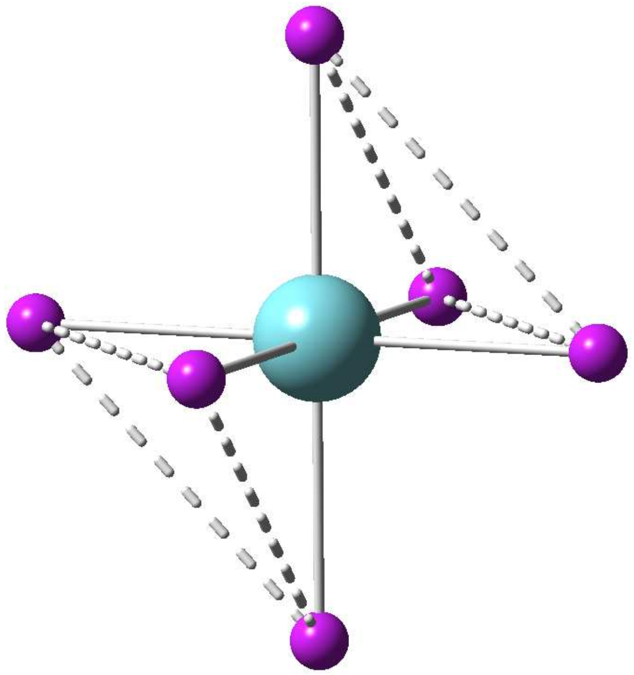

Figure 4: Octahedron showing two opposing faces (half bonds) that define a trigonal antiprism.

phy of tackling all three simultaneously. It is intended (but not mandated) that all initial structures generated by AuToGraFS are optimised using UFF4MOF. We use GULP as our force field program, it is free (as in beer) for academic users and is an incredibly powerful program with helpful features that are particularly helpful here.

GULP is capable of optimising both the atomic positions and lattice vectors simultaneously and setting a fairly high tolerance on the maximum gradient tolerance, or alternately, optimising for say, 100 steps, nets a high quality structure quite quickly. We recommend the following header line in GULP:

```
opti conp molmec noautobond cartesian fix
```

UFF4MOF was designed for this purpose and is available in recent editions of GULP and so we recommend the following line in the footer of your GULP input:

```
library uff4mof.lib
```

What one is essentially doing in this case, is using the molecular geometry information (ideal bond lengths, angles…) encoded in the force field, to correct for the lack of such information contained in the topology. Doing one global optimisation allows us to address all three geometric problems identified above simultaneously. Phrased another way: In a non-straightforward topology, the idea is not to get a perfect structure initially, but merely to generate a structure that is close enough for the force field to fix. We note that it is certainly possible for this approach to fail. Sometimes the failure will be informative - i.e. a topology simply may not work with a shorter / longer linker. Other times, the failure will simply be a failure. Recommended strategies to deal with failures are as follows:

- It is normally easier to fix bonds that are too long than too short. Scaling the topology by a factor that is too small will result in overlapping building blocks, which are difficult (though not impossible) to correct. AuToGraFS uses the position of the dummy atoms to determine the size of the building block, so a crude way to make your building block seem bigger is to place it further from its connecting atom.

- The way that each topology is used to build a molecular framework is defined in a topology-specific file, located in the `ase/topology` subdirectory and the scaling information is defined near the top of each file. This could be modified to include a % ‘fudge-factor’ (e.g. size of building blocks $\times$ 1.05), a fixed ‘fudge-factor’ (e.g. size of building blocks + 1Å) or to scale the lattice vectors non-uniformly.

Finally it is worth re-stating that AuToGraFS can output structures using any format it has available, and one is not obliged to pre-optimise using GULP (and for straightforward topologies, with connections largely defined along the edge vectors, you may not need to). If one wished to use a program that could not optimise the atomic positions and lattice parameters simultaneously, it would also be possible to do the optimisation in two steps, first optimising the atomic positions, then the cell. However, we really do recommend pre-optimising your structure using UFF4MOF and GULP.

Bibliography

(1) O'Keeffe, M.; Peskov, M. A.; Ramsden, S. J.; Yaghi, O. M. *Acc. Chem. Res.* **2008**, *41*, 1782–1789.

(2) Gale, J.; Rohl, A. *Mol. Simulat.* **2003**, *29*, 291–341.

(3) Gale, J. D. *Z. Krist.* **2005**, *220*, 552–554.

(4) Frahm, D.; Hoffmann, F.; Froba, M. *CrystEngComm* **2013**, *15*, 9429–9436.

(5) Addicoat, M. A.; Vankova, N.; Akter, I. F.; Heine, T. *J. Chem. Theory Comput.* **2014**, *10*, 880–891.