1. Finding strongly involutive self-dual graphs and their metric embeddings

In this section we describe the computational algorithms used to construct Reuleaux polyhedra from all strongly involutive self-dual graphs with up to 14 vertices. This software was written mostly in `C++` and can be freely downloaded (and used) from

`https://github.com/mraggi/ReuleauxPolyhedra`.

Our pipeline includes 4 steps:

(1) Generate all 3-connected planar graphs with the appropriate number of edges.
(2) Select only strongly involutive self-dual graphs.
(3) Embed the diameter graphs in $\mathbb{R}^3$.
(4) Create a Meissner or Reuleaux polyhedron from the embedding, for visualization.

1.1. **Generating 3-connected planar graphs.**

The process starts by using `plantri` [**?**], Brinkmann and McKay's wonderful software. We ask `plantri` to generate all 3-connected planar graphs with n vertices (with $n \leq 14$) and exactly $2n-2$ edges. This is by far the slowest step in the pipeline and therefore the bottleneck. This means that further optimizations in the following steps are not necessary.

1.2. **Constructing involutive isomorphisms.**

Once we have a list of all planar 3-connected graphs with the appropriate number of edges, we wish to find an bijection τ from the set of vertices $V(G)$ to the set of faces $F(G)$ that satisfies the properties of strongly involutive self-dual graphs detailed in Section **??**.

We accomplish this by posing the problem as a CSP (constraint satisfaction problem) and using the arc-consistency algorithm [**?**] in order to reduce the search space, followed by standard search. This works remarkably well in practice, and in fact takes virtually no time to refine the list generated in the previous step and find all strongly involutive self-dual graphs with up to 14 vertices. This is remarkable considering there are 23,556 planar 3-connected graphs with 14 vertices (and only 674 of them are strongly involutive self-dual).

This is how the arc consistency algorithm works in this context: Let G be a 3-connected planar graph. For each vertex $v \in V(G)$, associate a set-like data structure of candidate faces F_v, denoting "possible mappings". At first, F_v consists of all faces f with the same number of edges as the vertex degree and for which $v \notin f$ (in order to satisfy the definition of involutive self-dual graph), but we shall reduce the search space by repeatedly discarding faces which could not possibly be mapped to v. We worry about making τ a strong involution later, and focus now on simply constructing an isomorphism to the dual.

Consider all edges $uv \in E(G)$ (in CSP terms, these correspond to arcs variable $\to$ condition). Place them initially in a set-like data structure called `EdgesToProcess`. The arc-consistency part of the algorithm ends when this data structure is empty. Once an edge is removed from `EdgesToProcess` we say it is (temporarily) *consistent*.

Repeatedly consider edges uv from `EdgesToProcess` and make them consistent as follows: For each face $f \in F_u$, see if there exists a face $g \in F_v$ such that $fg \in E(G^*)$. This is in order for τ to stand a chance of being an isomorphism. If there is no such g, then remove f from F_u, as we are certain $\tau(u) \neq f$. Of course, we must now add all edges $xu \in E(G)$ to `EdgesToProcess`, as they might have stopped being consistent in the case f was needed to make xu consistent. If there is such a g, simply proceed to the next edge.

While we could further restrict the search space by performing a similar process for the property that τ must be strongly involutive, we found no further optimizations were useful for $n \leq 14$.

Once the above process finishes, we search all possible candidate τ by creating a tree of partial assignments and branching for each vertex v with every member of F_v, and pruning the tree when a partial assignment leads to inconsistencies, either because the an isomorphism or the strongly involutive properties are violated.

Given an isomorphism τ we can easily construct the diameter graph, which is helpful for the next steps.

1.3. **Finding an embedding.** Once we have a involutive self-dual graph M and its diameter graph $D(M)$, we embed it in $\mathbb{R}^3$ by choosing an appropriate objective function, for which we find minima using differential evolution [**?**, **?**].

Our software is able to numerically construct a good embedding for each involutive self-dual graph in a couple of seconds per graph. Bear in mind we want mappings with three properties: distance 1 for edges in $D(M)$, non degeneracy, and diameter 1.

In other words, we wish for the length of every edge of the diameter graph to be as close as possible to 1. Secondly, we also require that all the other pairs of vertices are somewhat separated so as to not construct degenerate or nearly degenerate examples. Finally, no pair must be at distance grater than 1 so that the overall diameter does not exceed 1.

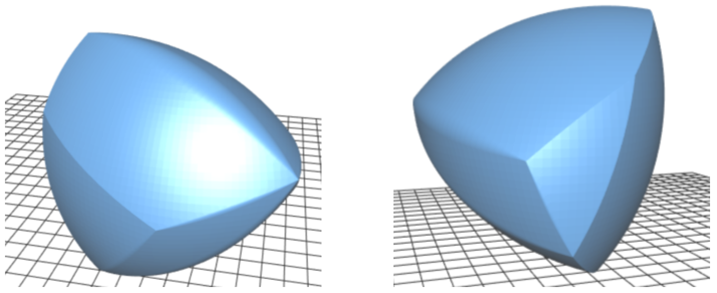

FIGURE 1. A Meissner body (from two different angles).

Let $\eta : V(M) \to \mathbb{R}^3$ be a possible metric embedding of the diameter graph $D = D(M)$. For a pair of points $a, b \in \eta(V)$, define h and w as follows:

$$h(a,b) = \begin{cases} 1 & \text{if } \operatorname{dist}(a,b) < \varepsilon, \\ 0 & \text{otherwise,} \end{cases} \quad \text{and } w(a,b) = \begin{cases} 1 & \text{if } \operatorname{dist}(a,b) > \alpha, \\ 0 & \text{otherwise,} \end{cases}$$

where $\varepsilon = 0.2$ and $\alpha = 0.95$.

With the previous considerations in mind, if $\hat{a} = \eta(a)$, the objective function we attempted to minimize is

$$J_M(\eta) = \sum_{ab \in E(D)} (\operatorname{dist}(\hat{a},\hat{b})^2 - 1)^2 + K \sum_{ab \notin E(D)} h(\hat{a},\hat{b}) + w(\hat{a},\hat{b})$$

where $K = 10$. The values of ε, α and K were chosen for practical reasons and seem to work well. Setting a higher value for ε (*e.g.* 0.3), did not yield an embedding for every graph in our software.

We stopped the algorithm when the value of J_M was less than 10^{-14}. This means, in particular, that no pair of points are too close to each other and, moreover, that the length of edges of D is almost 1, with an average error of about 0.0001.

This experimental evidence is what led us to venture Conjecture **??**.

1.4. **Visualization.** Lastly, we include code that allows us to construct a 3D model of either a Reuleaux polyhedron or a Meissner polyhedron from the embeddings found. It is a small script written in `OpenSCAD` (`https://www.openscad.org/`). For the Reuleaux polyhedron it simply constructs the intersection of the corresponding spheres. To construct a Meissner polyhedron we follow the procedure described in [**?**]. For convenience, we include premade `STL` files of one Meissner polyhedron corresponding to each involutive self-dual graph with up to 11 vertices. Figure 1 has a rendering of the Meissner body associated with the example in Figure **??**. For others, see the github site.

Instituto de Matemáticas, UNAM campus Juriquilla
Email address, L. Montejano: luis@im.unam.mx
Email address, E. Pauli: eriicpc@gmail.com

Escuela Nacional de Estudios Superiores, UNAM Campus Morelia
Email address, M. Raggi: mraggi@gmail.com

Centro de Ciencias Matemáticas, UNAM Campus Morelia
Email address, E. Roldán-Pensado: e.roldan@im.unam.mx