version: 1.0

Orb : Reference

Damian Heard, March 3, 2007

Contents

This work was partially supported by grants from the Australian Research Council and an Australian Postgraduate Award.

1. About Orb

Orb is a computer program that can find hyperbolic structures on a large class of hyperbolic 3-orbifolds and 3-manifolds. It can start with a projection of a graph embedded in the 3-sphere, and produce and simplify a triangulation with some prescribed subgraph as part of the 1-skeleton and the remainder of the graph drilled out. It enables computation of hyperbolic structures on knot complements, graph complements and orbifolds whose underlying space is the 3-sphere minus a finite number of points.

1.1. **Where is it from?** Orb was written by Damian Heard as part of his PhD thesis at the University of University of Melbourne, supervised by Craig Hodgson. You can download a copy at the Orb home page `www.ms.unimelb.edu.au/~snap/orb.html`.

The code was created by modifying the kernel of the program SnapPea written by Jeff Weeks. The user interface was written in Qt 3.3.4, with portion of the code provided by Morwen Thistlethwaite.

1.2. **Licence.** Orb is licensed under the terms of the GNU General Public License. It can be freely distributed for noncommercial purposes.

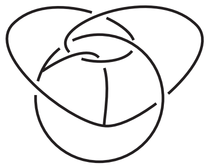
1.3. **System requirements.** Orb was developed on Linux and Mac OS X. A pre-compiled version of Orb, which should run on Mac OS X, is available at the Orb home page. Linux users need to download the source code and follow the instructions to compile.

2. Overview

Let Q be an n-orbifold and let $x \in Q$. Since n-orbifolds are locally modelled on $\mathbb{R}^n$ modulo finite subgroups G of $O(n)$, x has a neighbourhood which is a cone on a spherical $(n-1)$-orbifold $\mathbb{S}^{n-1}/G$. This gives us an extremely convenient way of describing 3-orbifolds:
Theorem: *Let Q be an orientable 3-orbifold. Then the underlying space X_Q is an orientable 3-manifold and the singular set consists of edges of order $k \geq 2$ and vertices where 3 edges meet. At a vertex, the three edges have orders p, q, r such that $1/p+1/q+1/r > 1$ (corresponding to the cone points on a compact orientable spherical 2-orbifold). Conversely, every such labelled graph in an orientable 3-manifold describes an orientable 3-orbifold.*

Let M be a closed 3-manifold and $\Gamma \subset M$ a labelled graph satisfying the conditions of Theorem 1. Then we can define a 3-orbifold Q by the pair (M, Γ). Orb allows the user to enter and study 3-orbifolds of the type (S^3, Γ) by drawing graphs projections as described in section 3 below.

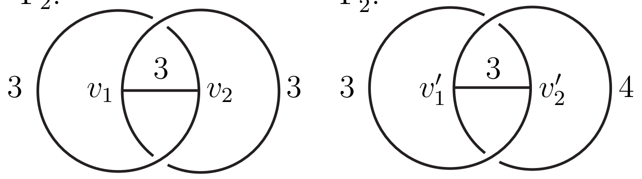
Γ_1:



Example 1: When the graph $\Gamma_1 \subset \mathbb{S}^3$ is labelled 2 it satisfies the conditions of the theorem. Orb can be used to place a hyperbolic structure on $Q = (\mathbb{S}^3, \Gamma_1)$. The hyperbolic volume of Q is approximately 0.1178.

For orientable hyperbolic 3-orbifolds with finite volume the scope of the labelling on Γ can be increased to allow for vertices coned on closed orientable Euclidean 2-orbifolds. A hyperbolic structure can be placed on Q by removing any such vertex from M, creating a cusp. Orb can also allow vertices coned on closed orientable hyperbolic 2-orbifolds. In this case Orb slices a neighbourhood of the vertex off, creating a 3-orbifold with (totally) geodesic boundary.

Γ_2: Γ_2':

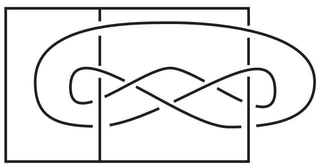


Example 2: The vertices v_1 and v_2 of Γ_2 have neighbourhoods which are cones on the Euclidean 2-orbifold $\mathbb{S}^2(3,3,3)$. In this case we abuse notation and let $Q = (S^3, \Gamma_2)$ denote the orbifold with underlying space $X_Q = \mathbb{S}^3 - \{v_1, v_2\}$ and singular locus $\Sigma(Q) = \Gamma_2 - \{v_1, v_2\}$ (so the v_i become cusps). The hyperbolic volume of Q is 0.6766....

Since a neighbourhood of v_1' is a cone on the hyperbolic 2-orbifold $\mathbb{S}^2(3,4,4)$, Orb slices off a neighbourhood of v_1' producing an orbifold, denoted $Q' = (S^3, \Gamma_2')$, with hyperbolic volume 1.2687....

Orb can also be used place to hyperbolic structures on knot and graph complements in $\mathbb{S}^3$. In this case, the edges of Γ are labelled ∞ to indicate that they will be *drilled out*, and there is a torus cusp or geodesic boundary surface corresponding to each connected component of Γ.

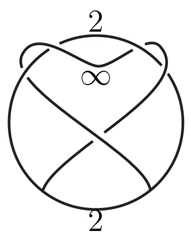
Γ_3:



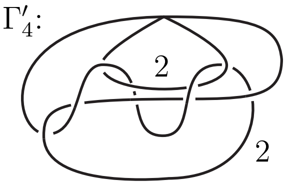
Example 3: If each of the two components of Γ_3 is labelled ∞ then $Q = (S^3, \Gamma)$ denotes the resulting handlebody with a knot drilled out. When Orb computes a hyperbolic structure on Q the boundary of the handlebody is made totally geodesic and the knot becomes a torus cusp. The hyperbolic volume of Q is 12.046....

Orb also allows the user to drill out specific edges on a graph, not just whole components. In some cases this provides a more natural way of representing orbifolds.

Γ_4:



Γ_4':



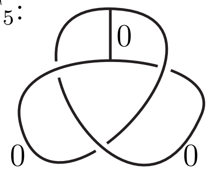
Example 4: Drilling out the arc labelled ∞ in Γ_4 produces a cusp with a $\mathbb{S}^2(2, 2, 2, 2)$ cross-section. You can enter the same orbifold into Orb using Γ_4'. To see this, turn the projection of Γ_4 into Γ_4' by retracting the edge labelled ∞ and rearranging the result using Reidemeister moves.

Interestingly, the orbifold $Q = (\mathbb{S}^3, \Gamma_4) \cong (\mathbb{S}^3, \Gamma_4')$ has the figure 8 knot as a 2-fold branch covering. You can prove this by unwrapping Γ_4 around the circle labelled 2. Using the projection Γ_4' to deduce this would be much more difficult. According to Orb , $\mathrm{vol}(Q) = 1.0149...$ which is consistent with the figure 8 knot having hyperbolic volume 2.02988....

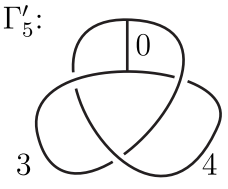
Note that Γ_4' has a degree 4 vertex. Orb accepts any graph projections with vertex degrees ≥ 3.

Finally, Orb can also be used to study so called "pared manifolds" by requiring that edges labelled 0 have parabolic meridians.

Γ_5:



Γ_5':



Example 5: An orbifold denoted $Q = (S^3, \Gamma_5)$ is produced by taking the complement of Γ_5 in $\mathbb{S}^3$ and finding a hyperbolic structure such that the meridian of each of its edges is parabolic. So Q is a manifold

with three annulus cusps (one for each edge of Γ_5) and two geodesic 3-punctured spheres as boundary components. According to Orb , Q has hyperbolic volume 5.33349....

We can also label individual edges 0. The orbifold $Q' = (\mathbb{S}^3, \Gamma_5')$ has one annulus cusp, two geodesic boundary components of the form $\mathbb{D}^2(3,4)$ and hyperbolic volume 3.2045....

3. Entering graph projections

To draw your own graph projections in Orb click the *pencil* on the toolbar. A blank canvas will appear. Use the bottom right corner to enlarge this if desired. To draw a graph (left) click once on each successive vertex. There are two ways to stop:

(1) Single click on a previous vertex.
(2) Put in the required vertices and then right[1] click anywhere else on the canvas.

You can:

- Add an edge to an existing graph by clicking on one of the vertices the adding new vertices as required.
- Switch crossings by clicking on them
- Move vertices by right[1] clicking on vertex and then dragging the vertex by holding down the mouse.
- Select a single line segment by right[1] clicking.
- Select an arc of the graph (or circle component) by (left) clicking.
- Click again to deselect.
- Kill an edge being created by right[1] clicking.

After the projection is complete you can select any edges or components you wish to drill and then click the *drill* button. Alternatively, any edge drawn while the drill button is toggled is automatically drilled. Drilled edges are coloured black.

After you are satisfied with the projection, click the *magnifying glass*[2]. Orb will triangulate the associated 3-orbifold and open a console window for you to study it.

In the console you can use the colour coding of the edges of the graph projection to label the graph's edges. The labelling on the projection is stored in the table of the top left corner of the console window. When the console window first opens, your graph projection will be hidden.

[1]If you only have a one button mouse you can perfrom a right click by holding down CTRL and then clicking.

[2]By default, Orb tries to minimize the number of tetrahedra used in each triangulation. In some cases an unsimplified triangulation is desirable. For instance, when the graph projection entered is planar the initial triangulation produced will be particularly nice, i.e. $Q = (\mathbb{S}^3, \Gamma)$ is a doubled polyhedron. To tell Orb not to simplify its triangulations, go to the menu attached to the magnifying glass and turn simplification off.

Click the *eye* in the bottom left corner of the console window to re-examine the projection. Clicking the *eye* a second time restores the console to it original state.

For graphs with a large number of edges the colour coding is not sufficient to produce a labelling. In this case you will need to use the index Orb assigns to each of the coloured edges. To determine the index on an edge of the graph, hold the cursor above it and the index will appear in the area above the graph projection. Alternatively, if you click on a coloured edge in the graph projection its label will be selected in the table.

You do not have the option of labelling the black edges - these are automatically labelled ∞ and drilled out.

Each coloured (non-black) edge on the graph becomes an edge in the triangulation of the orbifold. We call such an edge a coloured edge in the triangulation.

3.1. **Saving.** Each console window in Orb has its own *save* button located in the bottom left corner.

3.2. **Loading files.** To load a '.orb' file click on the *folder* on the toolbar and select the file. There are example '.orb' files in the `./examples/` directory. You can also import SnapPea triangulation files by clicking the appropriate button on the toolbar.

4. Using the console

In the console window you can attempt to place a hyperbolic structure on the orbifold (or manifold) you are studying. At each point Orb will have a triangulation for the orbifold in question. When you press the *update* button Orb attempts to solve the gluing equations for that triangulation using Newton's method. There are several possible outcomes:

- *Geometric* - A solution was found using geometric generalized hyperbolic tetrahedra. This is the best possible outcome.
- *Partially Flat* - A solution was found using geometric generalized hyperbolic tetrahedra and some flat generalized hyperbolic tetrahedra.
- *Nongeometric* - A solution was found but it requires nongeometric tetrahedra. Orb can not guarantee this corresponds to a genuine hyperbolic structure.
- *Flat* - A solution was found using all flat tetrahedra.
- *Other* - Newton's method found an invalid solution.
- *Step Failed* - Newton's method got 'stuck' trying to take a step between iterations. Typically this occurs when Newton's method has tried to move outside the region of realizable generalized tetrahedra.
- *No solution* - Newton's method did not converge to a solution.

If you are unsatisfied with the outcome of Newton's method there are two alternatives for finding better solutions:

(1) At times the triangulation Orb finds is not conducive to Newton's method converging. If you suspect Orb has found a 'bad' triangulation you may wish to retriangulate. You may have to retriangulate several times before a more suitable triangulation is found.
(2) You may wish to alter the *iteration style* Newton's method uses.

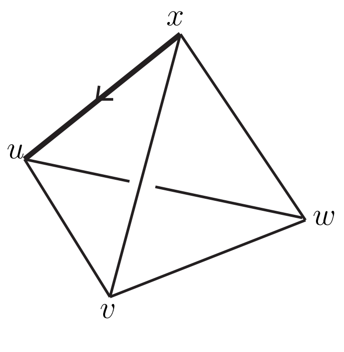
4.1. **Iteration styles.** Orb uses Newton's method to find hyperbolic structures. There are two iteration styles Newton's method can use:

- *Manual* - Orb uses the previous solution as the starting value for Newton's method. The user should be able to use this option to 'creep up' on the solution they are interested in. They can do this by creating a sequence of orbifolds (by relabelling the coloured edges) which converge to the desired orbifold.
- *Auto* - Orb automatically constructs a sequence of orbifolds to feed into Newton's method which is designed to converge to the desired orbifold.

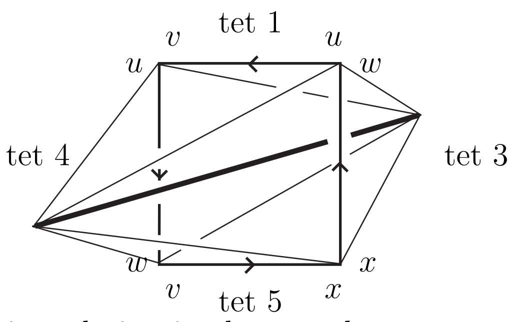
4.2. **Triangulations - <u>*tri*</u>.** Orb prints and saves its triangulations using a format adapted from Casson's Geo.

To describe the notation for triangulations, consider an orbifold obtained by gluing n tetrahedra numbered $1, \dots n$. Label the vertices of each tetrahedron u, v, w and x so that alphabetical order corresponds to positive orientation. This labelling also gives us a way of referencing the faces of each tetrahedron, e.g: face u lies opposite vertex u.

An oriented edge of a tetrahedron can be specified by an integer and two distinct letters from $\{u, v, w, x\}$. For instance, if the tetrahedron below is numbered 1 then the oriented edge pictured is $1xu$.



In the notation for a triangulation, a row contains the link of an unoriented edge in the triangulation, along with additional information indicating whether of not the edge is coloured. The edge link shown below can be described by $1uv$ $4uw$ $5vx$ $3xw$.



To examine a triangulation in the console press ***tri***. Below is an example of what you might see.

```
1  0  1.000  1vu  2xv  1vw  2wv  2vu  1xv
2  2  2.000  1wu  2ux  2xw  1xw
3  3  3.000  1xu
4  1  3.000  2wu
```

Each row will read

$$i_e \mid c_e \mid l_e \mid \overrightarrow{e_1}\overrightarrow{e_2}\ldots\overrightarrow{e_n},$$

where:

- i_e is the index on e;
- c_e is the colour index on e;
- l_e is the label on e;
- $\overrightarrow{e}_1\overrightarrow{e}_2\ldots\overrightarrow{e}_n$ are the oriented edges that make up the edge link around e.

If e is not coloured then $c_e = 0$ and $l_e = 1$. You can change the label on a coloured edge by using the table in the top left corner of the console.

There are additional output options provided through the ***tri*** menu. If you select *angle errors* the output will read

$$i_e \mid c_e \mid l_e \mid \delta_e \mid \overrightarrow{e_1}\overrightarrow{e_2}\ldots\overrightarrow{e_n},$$

where δ_e is the difference between the cone angle specified by the gluing equations and the actual cone angle at that edge.

If you select *verbose* mode from the ***tri*** menu then the above lines become

$$i_e \mid c_e \mid l_e \mid v_e^1 \mid v_e^2 \mid \overrightarrow{e_1}\overrightarrow{e_2}\ldots\overrightarrow{e_n},$$

where v_e^1, v_e^2 are the indices[4] of the vertices at each end of e. In addition, verbose mode will also print the information necessary to recreate the structure below the triangulation. Each line of this output will read

$$i_e \mid \beta_e \mid \alpha_{v_e^1} \mid \alpha_{v_e^2} \mid \theta_1\theta_2\ldots\theta_n,$$

where:

- β_e is the structure parameter for e;
- $\alpha_{v_e^1}$, $\alpha_{v_e^2}$ are the structure parameters for the vertices at the two ends of e;
- θ_s is the dihedral angle incident to e opposite $\overrightarrow{e_s}$.

[4]A finite vertex will have a negative index.

The parameters used by Orb to find hyperbolic structures are inner products of vertices when the (generalized) hyperbolic tetrahedra are lifted to Minkowski space. Please refer to Damian Heard's thesis for the details.

The ***tri*** menu also provides the actions *canonize*, *randomize* and *simplify* which can be used to search for different triangulations. There is also a *remove finite vertices* action which removes or minimizes the number of finite vertices in the triangulation.

4.3. **Tetrahedra - *tet*.** Push the ***tet*** button to examine the tetrahedra in the triangulation. By default ***tet*** prints the hyperbolic volume of each tetrahedron. Use the ***tet*** menu to print the *dihedral angles*, *edge lengths*[4], gluing maps and *vertex Gram matrices* of each of the tetrahedra. Below is an sample of some ***tet*** output[5].

```
tet  1                Volume:       0.798

        uv    uw    ux    vw    vx    wx
da :  1.570 1.047 1.047 1.047 0.628 1.047
el :  2.348 0.413 1.241 1.230 1.720 0.357
nbr:  2 (wvux) 7 (uxwv) 6 (wvux) 7 (uxwv)
grm:
       0.730 -1.222 -0.861 -1.006
      -1.222  0.000 -1.289 -1.138
      -0.861 -1.289 -0.138 -1.482
      -1.006 -1.138 -1.482 -0.566
```

This allows the user to examine various properties of tetrahedron 1, such as:

(1) The volume of tetrahedron 1 is approximately 0.798.
(2) The dihedral angle *between faces v and x* is 0.628.
(3) The vertex Gram matrix gives inner products of tetrahedron vertices lifted to Minkowski spaces.We can use the signs of the diagonal of the vertex Gram matrix to see that u is hyperinfinite, v is ideal and both w and x are finite.
(4) The length of the edge *between faces w and x* is 0.357. This length measures the distance between the hyperplane defined by u and the prescribed horosphere at v.
(5) Face u of tetrahedron 1 is glued to tetrahedron 2 via gluing map $uvwx \mapsto wvux$.

[4]The term 'edge lengths' is used loosely. In the case where there are ideal vertices, values are signed lengths obtained after slicing across some prescribed horospheres defined by the vertex parameters.

[5]Note that precision has been manually altered for ease of display.

4.4. **Symmetry Groups - *sym*.** This command prints the symmetry group of the orbifold. By default it also prints the chirality of the orbifold. The *sym* menu allows more detail to be printed. If you select *vertex action* (resp. *colour edge action*) then the permutation each symmetry induces on each of the vertices (resp. coloured edges) is printed, together with permutation a ‘p’ or a ‘r’ to indicate whether the symmetry is orientation preserving or reversing.

4.5. **Graph Types - *gt*.** This command allows the user to understand the boundary components and singular locus of the orbifold. If T denotes a triangulation for an orbifold Q we define a graph $G(Q)$ as follows:

- Denote the *2-orbifold* in the link of vertex v of T by Q_v. A vertex v is *trivial* if $Q_v = \mathbb{S}^2$.
- Each *non-trivial* vertex v of T has a corresponding vertex v' in $G(Q)$.
- $G(Q)$ has an edge joining v'_1 and v'_2 for each coloured edge in T joining v_1 and v_2.

The *gt* command gives the user enough information to reconstruct the graph $G(Q)$. The default option is to print the adjacency matrix of the graph. The user can get additional information on the Q_v orbifolds by going to the *gt* menu and selecting the *list* option. Some example output is provided below.

```
1  0 -1.000  1  1
2  2  0.000  2  2  3  3
```

This will read

$$i_v|\chi(X_{Q_v})|\chi(Q_v)|c_{e_1}c_{e_2}\dots c_{e_n},$$

where:

- i_v is the index for vertex v;
- X_{Q_v} is the underlying space of Q_v;
- $c_{e_1}c_{e_2}\dots c_{e_n}$ are the indices of the coloured edges incident to v.

In this example the three coloured edges here labelled 2 so $S_{v_1} = \mathbb{T}(2,2)$ and $S_{v_2} = \mathbb{S}^2(2,2,2,2)$.

4.6. **Fundamental Group - *fg*.** This command prints the fundamental group. The following output is produced when examining the Whitehead link:

```
Generators: {a,b}
Relations : aBAAABBAbaaabb
Parabolics:
Merdians  : AB
            BAA
Longitudes: Abaaab
            aaabba
```

Peripheral words for the meridians and longitudes on the torus cusps are also listed. When studying pared manifolds the meridian of each annulus cusp is listed under *Parabolics*.

The default option is that group presentations should be simplified with as few generators as possible. These settings can be changed under the `fg` menu. The menu also allows the user to print the homology group.

4.7. **Matrix Generators - `mg`.** This command prints matrix generators for hyperbolic orbifolds. Use the `mg` menu to choose between the $O(3,1)$ and $PSL(2,\mathbb{C})$ formats.

4.8. **Length Spectrum - `ls`.** This command allows the user to feed matrix generators into SnapPea's `length_spectrum()` function. Use the `ls` menu to modify the tile radius, the cut off length and the other variables for `length_spectrum()`. See SnapPea for a more detailed explanation of the parameters.

4.9. **Short cuts.**

Command	Quick key[6]
`tri`	Space
`tet`	t
`sym`	s
`gt`	g
`fg`	f
`mg`	m
`ls`	l

5. Other actions

There are more useful commands available to the user in the *actions* menu. This can be viewed by clicking the *wand* in the lower left corner of the console window.

5.1. **Covers.** You can use Orb to construct branched coverings of orbifolds. This does not require a hyperbolic structure to be present on the orbifold. Click the *build covers* command on the actions menu and select the desired number of sheets in the cover. The table will list the covers available. The first column will contain one of the following four characters:

(1) 'r' indicating a regular cover;
(2) 'i' indicating an irregular cover;
(3) 'c' indicating a cyclic cover;

To examine a cover in more detail, select it and press *build*.

[6]Note: Majority of menu commands also have short cut keys. These are indicated on the menus.

5.2. **Dehn Filling.** This option lets you Dehn filling torus cusps in orientable orbifolds. Dehn filling curves are specified as integer multiples of the meridians and longitudes on each cusp. When the curve specified is of the form $(\alpha p, \alpha q)$, where p and q are coprime, the curve (p, q) is filled and the core of the Dehn filling disk becomes a coloured edge labelled α.

5.3. **Export to SnapPea.** When you have manifold with torus and Klein bottle cusps you can use this command to export to SnapPea.

5.4. **Finding triangulations.** At times you may have to do a lot of randomization to find a good triangulation. The *find triangulation* command automates this operation. Use the slider to select the number of randomizations you wish to perform and then click *tabulate*.

The table will list all the combinatorially distinct triangulations found. The first two columns in the table list the number of tetrahedra and edges in each triangulation. The third column describes the *coloured edge orders*. For example: "0,1,2,0,3,0,0,0" indicates there is one edge of order two, two edges of order three and three edges order five. Edges of order greater than eight are ignored in this field. The fourth column has a similar field for non-coloured edges.

You may also tell Orb to try and compute structures on each of the triangulations. This may take along time if Orb finds a large number of triangulations. Information on the structures found is included in the fifth and sixth columns of the table.

5.5. **Graph pruning.** Use the 'Prune graph' command to retriangulate the underlying space so that specified coloured edges are not necessarily part of the 1-skeleton. Choose the edges you wish to remove by labelling them 1 and then click 'Prune graph'.

5.6. **Graph drilling.** Use the 'Drill graph' command to drill out specified coloured edges, and retriangulate so that these edges become ideal vertices. Choose the edges you wish to drill by labelling them 0 and the click 'Drill graph'.

5.7. **Undo.** Use undo to revert to the structure found when you last pressed the 'Update' button.

6. Comparing triangulations

The *Compare* window uses canonical cell decompositions to determine if two orbifolds are isomorphic. Make sure the two orbifolds are loaded in Orb and select one of the orbifolds from the left list and the other from the right list and click *compare*.

If Orb fails to compute a canonical cell decomposition then you can use random matching. Under this method Orb randomize the triangulations to see if it can match them up. Use the slider to select the level of randomization you require. If the slider is all the way to the left then Orb attempts to match the current triangulations.

7. Clipboard

The clipboard is a built-in text editor which can be used to store and save text. It can also be used to load triangulations. For example, you can load the figure eight knot into Orb by entering

```
  1  0  1.000  1vu  2uw  1ux  2xv  1xw  2vu
  2  0  1.000  1wu  2wx  1xv  2xu  1vw  2vw
```

into the clipboard and click the *Export* button. Hyperbolic structures can be read in with triangulations if entered in Orb format[7] (as given in the *verbose* mode from the triangulation menu):

```
SolutionType geometric_solution

 1  0  1.000  1vu  2uw  1ux  2xv  1xw  2vu
 2  0  1.000  1wu  2wx  1xv  2xu  1vw  2vw

 1 -23.094 0.000 0.000 1.047 1.047 1.047 1.047 1.047 1.047
 2 -23.094 0.000 0.000 1.047 1.047 1.047 1.047 1.047 1.047
```

8. More software

If you enjoyed using Orb, the following software may interest you:

- Jeff Weeks' SnapPea, a powerful tool for creating and studying hyperbolic 3-manifolds.
- Oliver Goodman's Snap, for computing exact hyperbolic structures and arithmetic invariants of hyperbolic 3-manifolds.
- Andrew Casson's Geo and Cusp, for placing geometric structures on closed and cusped 3-manifolds.
- Bruno Martelli's ographs, for computing hyperbolic structures on 3-manifolds with totally geodesic boundary.

Thanks to Nathan Dunfield, these programs and many others are available at `www.computop.org`.

9. Reference

Damain Heard, *Computation of hyperbolic structures on 3-dimensional orbifolds*, PhD thesis, University of Melbourne, December 2005, `www.ms.unimelb.edu.au/~snap/orb.html`.

[7]Note that precision has been manually altered for ease of display.