Exploring the Clifford torus

Miles Cochran-Branson
IHRTLUHC

Friday, October 1

We will explore the variety V defined by

$$S^1 \times S^1 \subset \mathbb{R}^4. \tag{1}$$

Implicit Defining Equations

To begin, we examine the implicit equations that describe this object. Let's first consider the circle S^1 which is defined by the set

$$S^1 = \{(x_1, x_2) : x_1, x_2 \in \mathbb{R} \,|\, x_1^2 + x_2^2 = 1\}. \tag{2}$$

In order to describe our variety, we then take the cartesian product of S^1 with itself and find that

$$S^1 \times S^1 = \left\{(x_1, x_2, x_3, x_4) : x_1, x_2, x_3, x_4 \in \mathbb{R} \,|\, x_1^2 + x_2^2 = 1 \,\text{and}\, x_3^2 + x_4^2 = 1\right\}.$$

Thus, we see that our variety is defined by the equations

$$\begin{aligned} x_1^2 + x_2^2 &= \frac{1}{2} \\ x_3^2 + x_4^2 &= \frac{1}{2} \end{aligned} \tag{3}$$

where we have introduced the normalization $x_1^2 + x_2^2 + x_3^2 + x_4^2 = 1$.

Parametrization

Let's now parametrize (3) using trigonometric functions. We first note that

$$S^1 = \{(\cos(\theta), \sin(\theta)) : \theta \in \mathbb{R} \,|\, 0 \leq \theta < 2\pi\}. \tag{4}$$

It then follows that

$$S^1 \times S^1 = \{(\cos(\theta), \sin(\theta), \cos(\phi), \sin(\phi)) : \theta, \phi \in \mathbb{R} \,|\, 0 \leq \theta < 2\pi, 0 \leq \phi < 2\pi\}.$$

We thus have the parametrized map $f : \mathbb{R}^2 \to \mathbb{R}^4$ defined by

$$(\theta, \phi) \mapsto \left(\frac{1}{\sqrt{2}} \cos(\theta), \frac{1}{\sqrt{2}} \sin(\theta), \frac{1}{\sqrt{2}} \cos(\phi), \frac{1}{\sqrt{2}} \sin(\phi) \right) \tag{5}$$

where we have normalized such that the radius of the sphere $\mathbb{S}^3$ is one as above. It is also useful to note here, that this normalization is introduced as our variety V lives within the sphere $\mathbb{S}^3 \subset \mathbb{R}^4$. Thus, we see that we're looking at an object $V \subset \mathbb{S}^3 \subset \mathbb{R}^4$.

Computing and Visualization

Using `julia` we can compute one thousand points that lie on our curve using the following code:

```
using Plots

n = 1000

#Define vectors for input
theta = (2 * pi) .* rand(n)
phi = (2 * pi) .* rand(n)

#Convert to cartesian coordinates
x1 = [(1 / sqrt(2)) * cos(u) for u in theta]
x2 = [(1 / sqrt(2)) * sin(u) for u in theta]
x3 = [(1 / sqrt(2)) * cos(v) for v in phi]
x4 = [(1 / sqrt(2)) * sin(v) for v in phi]
```

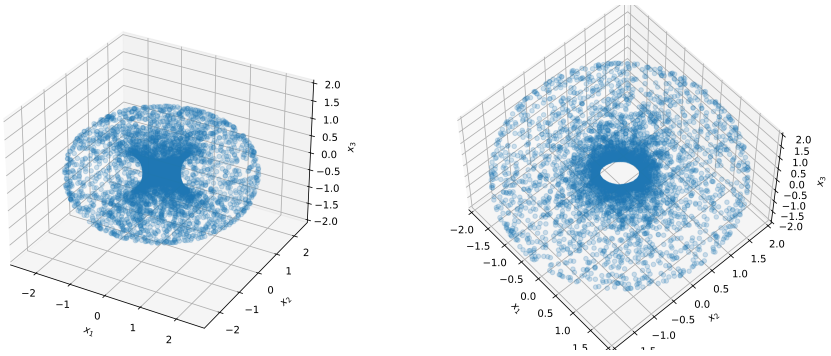
The output of this can then be turned into a `.txt` file using the `DataFrames` and `CSV` packages in `julia`. This file, which I have called `data_values.txt`, is included in my project submission.

We are now tasked with visualizing this object. We will use a stereographic projection so that we can visualize our variety in three dimensions while maintaining as much information as possible. If we project our calculated points from above, which we know lie in $\mathbb{S}^3$, from the three-sphere down into three dimensional space, we can visualize our object. This is done simply through the projection

$$(x_1, x_2, x_3, x_4) \mapsto \left(\frac{x_1}{1 - x_4}, \frac{x_2}{1 - x_4}, \frac{x_3}{1 - x_4} \right). \tag{6}$$

Now we can finally plot our object in three dimensions as shown below in Figure 1.

Of course, this only looks at our object from one point of view. We can rotate the torus in $\mathbb{R}^4$ via the rotation matrix



(a) (b)

Figure 1: Stereographically projected Clifford torus into three dimensions. (a) shows a side view, while (b) shows a view from the top both of the same object.

$$R(\eta) = \begin{pmatrix} \cos(\eta) & -\sin(\eta) & 0 & 0 \\ \sin(\eta) & \cos(\eta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{7}$$

Rotations of our variety executed from $0 \leq \eta < 2\pi$ can be projected into $\mathbb{R}^3$ and then visualized through an animation which is included as an attachment to my problem submission.

Tangent and Normal Spaces

Let's now find a basis for both the tangent and the normal space of V. In order to find a basis for the tangent space, we compute the derivative of the image of our map, i.e. we compute the derivative of (5) with respect to both θ and ϕ. We subsequently find that

$$(\theta, \phi) \mapsto \frac{1}{\sqrt{2}} \begin{pmatrix} -\sin(\theta) & 0 \\ \cos(\theta) & 0 \\ 0 & -\sin(\phi) \\ 0 & \cos(\phi) \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \tag{8}$$

where $v_1, v_2 \in \mathbb{R}$ are arbitrary scalers and note that the columns of the above matrix span the tangent space. Thus these columns form a basis of the tangent space.

We now consider the normal space and note that a basis for this space can be found by taking the derivate of the image of V given in (3). This is given by

$$(x_1, x_2, x_3, x_4) \mapsto \begin{pmatrix} 2x_1 & 2x_2 & 2x_3 & 2x_4 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} \tag{9}$$

where, again, $v_1, v_2, v_3, v_4 \in \mathbb{R}$ are arbitrary scalers. We then note that the columns of the above matrix span the normal space which lies in $\mathbb{R}^4$ and thus, the rows of the above matrix form a basis for the normal space.

Volume

In order to compute the volume of our variety, we consider the Gram Matrix which is defined by

$$(G_{u_1,\ldots,u_k})_{i,j} = u_i \cdot u_j. \tag{10}$$

In this case we're considering $k = 2$, thus

$$G_{u,v} = \begin{pmatrix} u \cdot u & u \cdot v \\ v \cdot u & v \cdot v \end{pmatrix}. \tag{11}$$

The vectors v and u are just the basis vectors we found for the tangent space, i.e.

$$\begin{aligned} u &= \frac{1}{\sqrt{2}}(-\sin(\theta), \cos(\theta), 0, 0) \\ v &= \frac{1}{\sqrt{2}}(0, 0, -\sin(\phi), \cos(\phi)) \end{aligned} \tag{12}$$

and we see that

$$G_{u,v} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \cdot I. \tag{13}$$

Okay—now from some theorem (which I just found on Wikipedia), we know that

$$Vol_2(V) = \sqrt{\det(G)}. \tag{14}$$

It's easy to compute the determinant of G, namely,

$$\begin{vmatrix} 1/\sqrt{2} & 0 \\ 0 & 1/\sqrt{2} \end{vmatrix} = \frac{1}{2} \tag{15}$$

and we see that our volume scale factor from parameter space to $\mathbb{R}^4$ is simply

$$Vol_2(V) = \frac{1}{\sqrt{2}}. \tag{16}$$

To find the volume of the whole object, we need to integrate over our parameter space. We note that this becomes trivial as $Vol_2(V)$ is constant! Thus, our volume is given by

$$\begin{aligned} Vol &= \iint 1 \cdot \sqrt{\det G} \cdot dV \\ &= \int_0^{2\pi}\int_0^{2\pi} 1 \cdot \frac{1}{\sqrt{2}}\, d\theta d\phi \\ &\boxed{= \frac{4\pi^2}{\sqrt{2}}} \end{aligned} \tag{17}$$

Distance to a point

Finally, we are tasked with finding the point $p' \in V$ that is closest to the point $(10, 10, 8, 7) \in \mathbb{R}^4$. We can do this computationally by using the distance formula to find the point closest to our object. More specifically, we have to find some $p' = (x_1, x_2, x_3, x_4)$ such that the distance d to the point $p = (10, 10, 8, 7)$ is minimized, where d is given by

$$d = \sqrt{(x_1 - 10)^2 + (x_2 - 10)^2 + (x_3 - 8)^2 + (x_4 - 7)^2}. \tag{18}$$

This is done by throwing many random points onto our surface to get a range of points from which to sample. We then run through all of these points and find which distance is the smallest. We have thrown 1×10^7 values for both θ and ϕ and then have found the minimum distance and it's corresponding point using the following function written in `julia`.

```julia
function find_min(n, p)

    #Define vectors for input
    theta = (2 * pi) .* rand(n)
    phi = (2 * pi) .* rand(n)

    #Convert to cartesian coordinates
    x1 = [(1 / sqrt(2)) * cos(u) for u in theta]
    x2 = [(1 / sqrt(2)) * sin(u) for u in theta]
    x3 = [(1 / sqrt(2)) * cos(v) for v in phi]
    x4 = [(1 / sqrt(2)) * sin(v) for v in phi]

    #define distance and minimum point
    d = Inf64
    min_point = Vector{Float64}(undef, 4)

    #find minimum distance
    for i in 1:n
    d_temp = sqrt((x1[i] - p[1])^2 + (x2[i] - p[2])^2
                    + (x3[i] - p[3])^2 + (x4[i] - p[4])^2)
        if (d_temp < d)
            d = d_temp
            min_point = [x1[i], x2[i], x3[i], x4[i]]
```

```
        end;
    end;

    return min_point, d

end;
```

Running this function ten times, we get a spread of values for the minimum distance and the point corresponding to this distance. We then append these to a `.CSV` file, read this into a data frame, and can then compute the standard deviation and our final value for the minimum point. The code for this is shown below

```
#Report average values
min_distance_avg = sum(df[!, 5]) / length(df[!, 5])
min_distance_std = std(df[!, 5])

#Initialize values for minimizer
d_min = df[1, 5]
min_point = [df[1,1], df[1, 2], df[1, 3], df[1, 4]]

#Find absolute minimum
for i in 2:length(df[!, 1])
    d_temp = df[i, 5]
    if d_temp < d_min
        global d_min = d_temp
        global min_point = [df[i,1], df[i, 2], df[i, 3], df[i, 4]]
    end;
end;
```

We find that our minimum distance is given by

$$d = 16.70229639 \pm 0.00000004. \tag{19}$$

and we see that the point closest to p is

$$\boxed{p' = (0.49998, 0.50001, 0.53215, 0.46594)} \tag{20}$$