**Non-Euclidean Geometry for SAT**

Tom Henderson

6 June 2023

University of Utah

A set of Matlab functions have been created to allow the exploration of the usefulness of setting geometric SAT into non-Euclidean geometry. Three major models are represented: (1) the Poincare half-plane (PH), (2) the Poincare disk (PD), and (3) the Beltrami-Klein disk (BK). For more information on these formulations, see Appendixes A, B and C, respectively (taken from Wikipedia).

## The Poincare Half-Plane (H)

Since the goal is to represent n-dimensional polytopes which represent the feasible region for a SAT solution, it is necessary to represent these in non-Euclidean spaces. Examples are given in 2D for illustration purposes. Consider the knowledge base with the single clause: . Then in regular Euclidean space, the feasible region (after chopping the (0,1) vertex) will be the triangle [(0,0), (1,0), (1,1)] (see Figure 1).

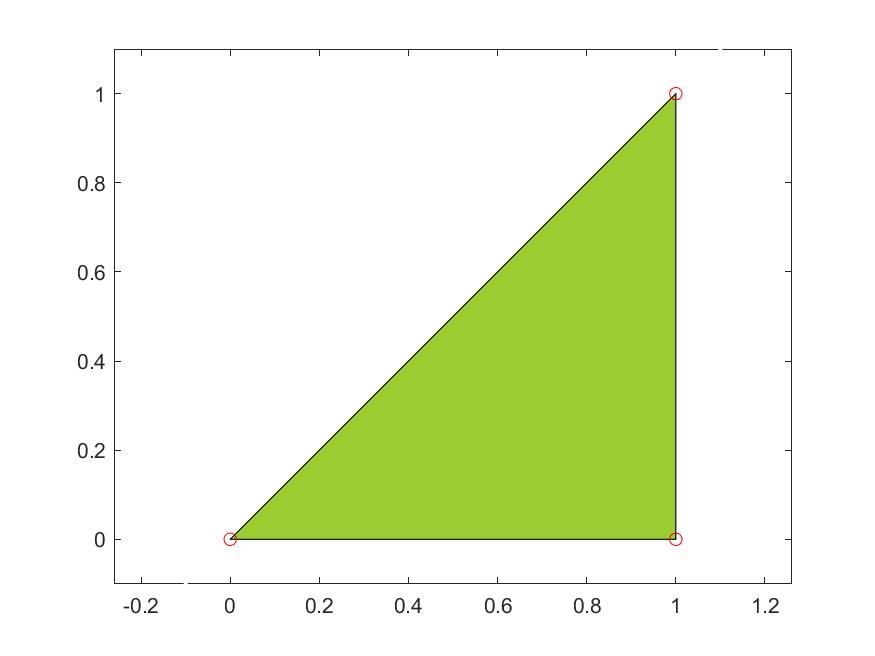


Figure 1. Feasible Region for KB with Clause .

Converting this to the Poincare half-plane model requires deciding how the unit square will be represented. The most straightforward is to use the same points: (0,0), (1,0), (1,1), (0,1); however, the points on the x-axis are not in H, and this poses some problems. Figure 2 shows how the unit square transforms into H.

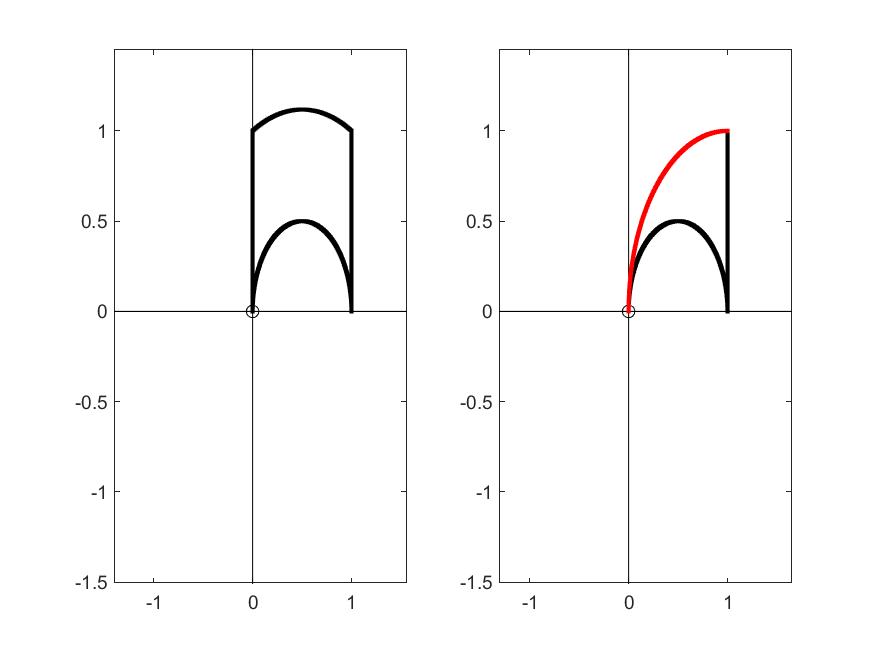


Figure 2. (left) Unit Square in Poincare Half-Plane. (right) Feasible Region for KB with Clause (cutting plane shown in red).

The left side plot is achieved as follows; first get the sides of the unit square:

seg1 = NON\_PH\_seg\_pts([0,0],[1,0]);

seg2 = NON\_PH\_seg\_pts([1,0],[1,1]);

seg3 = NON\_PH\_seg\_pts([1,1],[0,1]);

seg4 = NON\_PH\_seg\_pts([0,1],[0,0]);

Then plot them:

NON\_plot\_PH\_pts([seg1;seg2;seg3;seg4],1,'k.');

The right side is found by first finding the cutting line, then plotting the three remaining segments:

seg8 = NON\_PH\_seg\_pts([1,1],[0,0]);

NON\_plot\_PH\_pts([seg1;seg2],1,'k.');

NON\_plot\_PH\_pts(seg8,1,'r.');

Note that the area of the resulting triangle is , so that the area of the unit square in H is not 1!

## The Poincare Disk (D)

The unit square represented by [(0,0),(0.5,0),(0.5,0.5),(0,0.5)] is shown on the left side of Figure 3, while the same feasible region is shown on the right side of the figure.

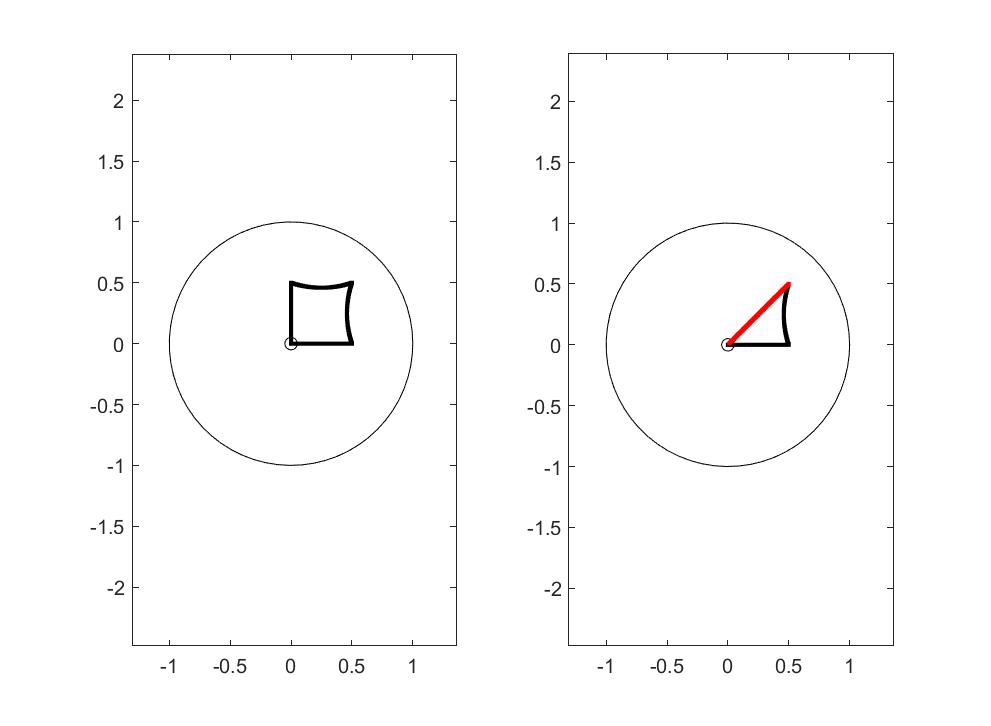


Figure 3. (left) Unit Square in Poincare Disk. (right) Feasible Region for KB with Clause (cutting plane shown in red).

This is produced as follows; for the left side:

seg1 = NON\_PD\_seg\_pts([0,0],[0.5,0]);

seg2 = NON\_PD\_seg\_pts([0.5,0],[0.5,0.5]);

seg3 = NON\_PD\_seg\_pts([0.5,0.5],[0,0.5]);

seg4 = NON\_PD\_seg\_pts([0,0.5],[0,0]);

NON\_plot\_PD\_pts([seg1;seg2;seg3;seg4],1,'k.');

The figure on the right:

seg5 = NON\_PD\_seg\_pts([0.5,0.5],[0,0]);

NON\_plot\_PD\_pts([seg1;seg2],1,'k.');

>> NON\_plot\_PD\_pts(seg5,1,'r.');

### Converting Points between Representations

Sometimes it is convenient to change representation; therefore, we have provided functions to convert as follows:

* NON\_H2D: Poincare half-plane to Poincare disk
* NON\_D2H: Poincare disk to Poincare half-plane
* NON\_H2K: Poincare half-plane to Beltrami-Klein
* NON\_K2H: Beltrami-Klein to Poincare half-plane
* NON\_D2K: Poincare disk to Beltrami-Klein
* NON\_K2D: Beltrami-Klein to Poincare disk

Note that all of these take on complex number input and produce one complex number output.

### Distance Between Points

Functions have been provided to compute the distance between points:

* NON\_norm\_PD: Poincare norm
* NON\_norm\_PH: Poincare norm
* NON\_norm\_BK: Beltrami-Klein norm

### Possible Representation of the Hypercube in n-D

A possible representation of the hypercube in n-D is to project the corners of the unit cube (centered a 0 and scaled to circumscribe the unit sphere) onto the unit hypersphere in D. Figure 4 shows this for 2D; note that the corners of the square are ideal points (see Appendix D), and not in D.

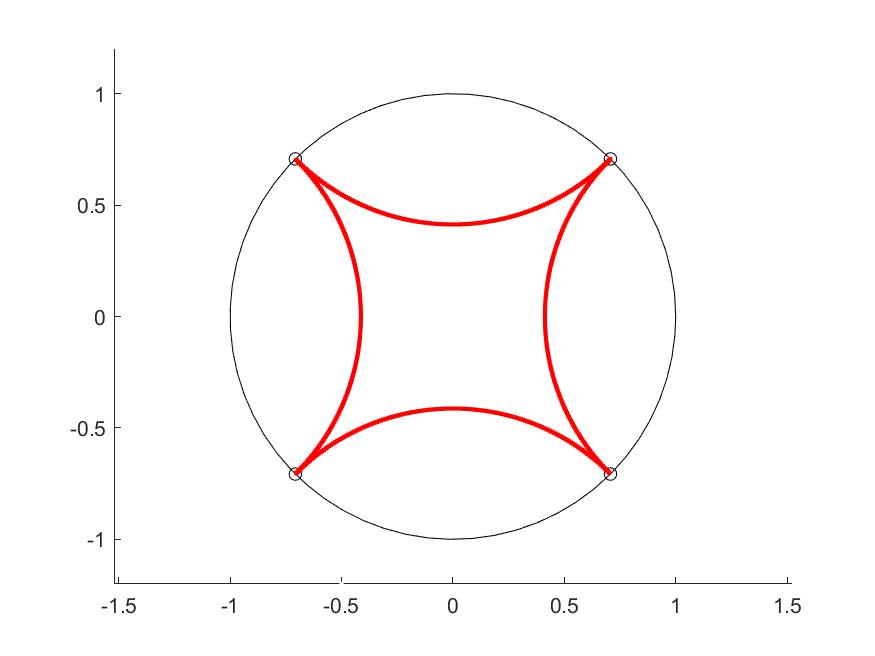


Figure 4. A projection of the Circumscribed Hypercube onto the Unit Sphere. In this case, the corners are not in D, but rather are ideal points on the circle boundary.

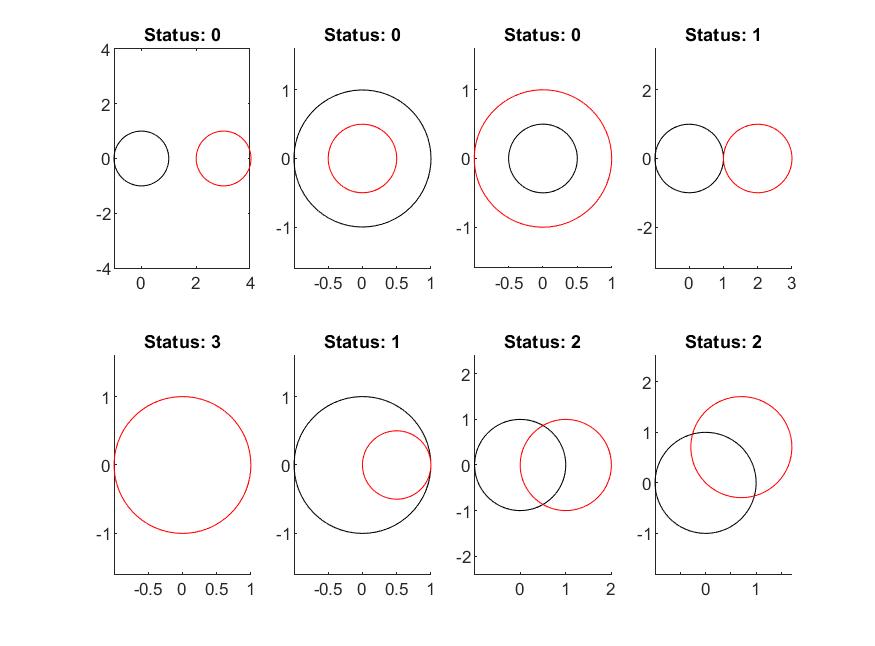
This makes a shape that is geometrically similar to the square, but note that its area is . Figure 4 is produced by:

NON\_H2circumcribedinPD; % files in PSSAT/non\_Euclidean/develop

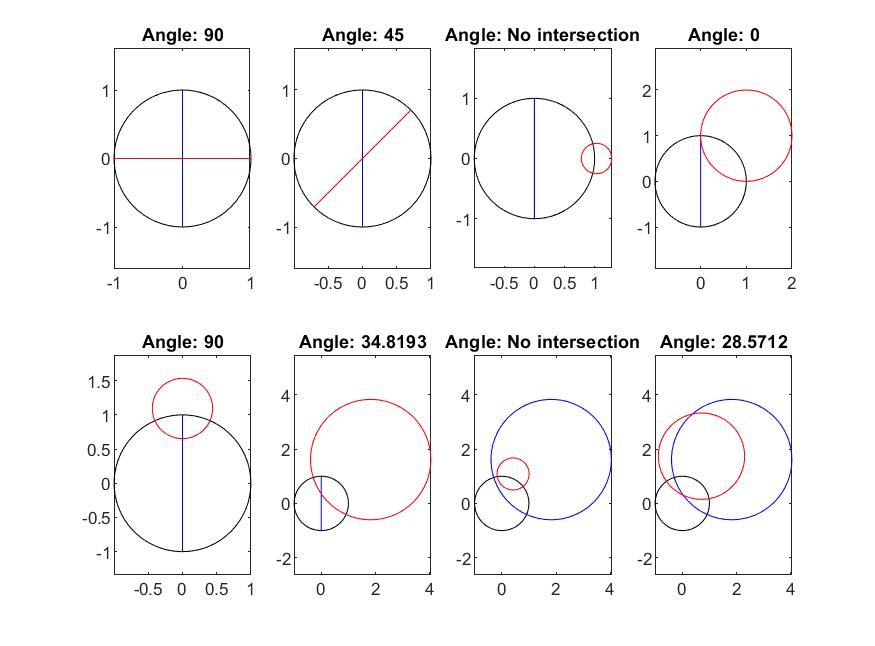
**27 June 2023**

Development over last few weeks is in PSSAT/non\_Euclidean/ with prefix NE\_ (see Non-Euclidean-Matlab-Functions.pdf and NE\_Function\_Dependencies.pdf).

Results testing Euclidean circle intersection (NE\_test\_int\_E2\_2circles):



Results testing great circle intersection angles (NE\_test\_angle\_2circles):



## Inversion:

Given a circle with center and radius , then and are *symmetrical* with respect to if:

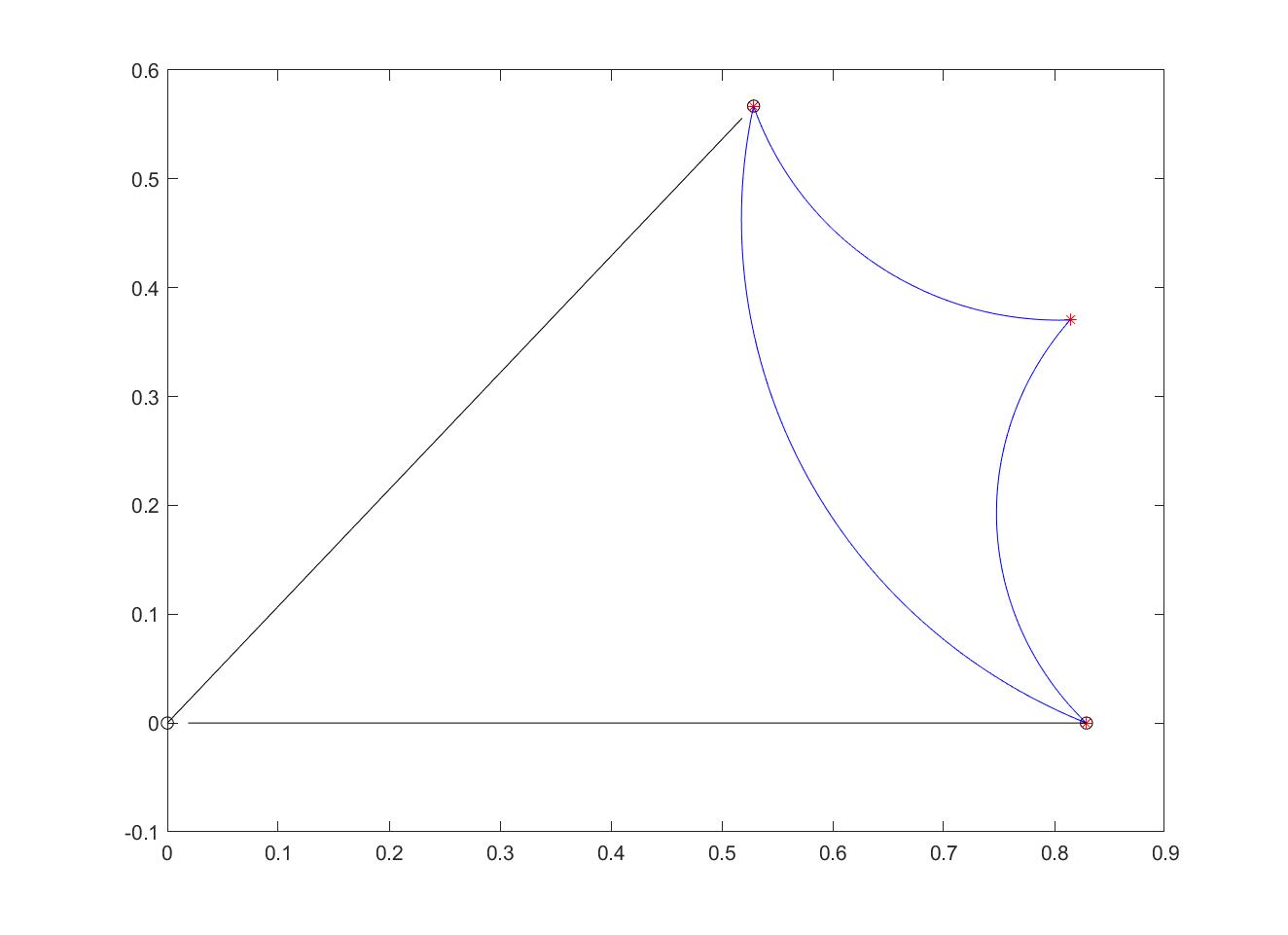
1. are collinear with outside of

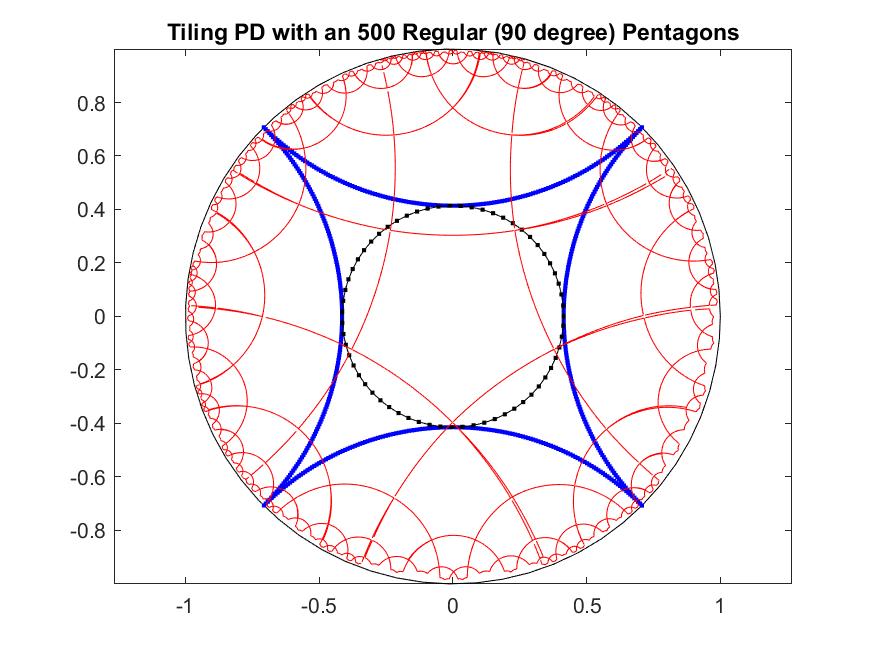
If is inside and let be perpendicular to and contains ,

then the intersection of the tangent line to at and line is .

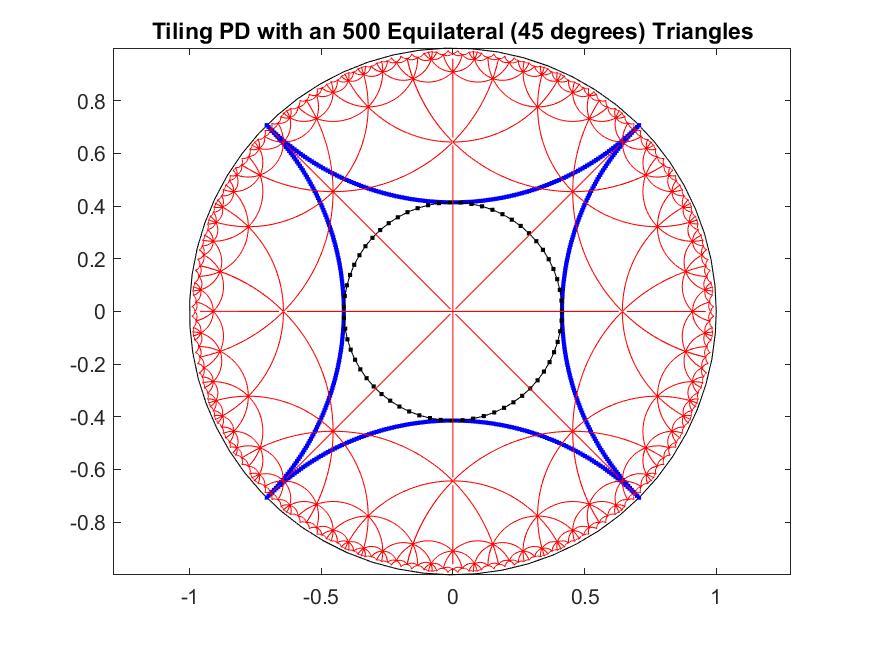
From equation (2) we have the symmetrical point:

Given the vertexes of a polygon, the polygon resulting from their inversion through one of the polygon’s sides produces a tiling of the Poincare Disk if repeated ad infinitum. Inverting the 30-40-50 triangle yields the triangle to the right in the figure below. Results of tiling PD with a regular pentagon (NE\_inversion\_experiment1):





Results of tiling PD with 45-degree angle regular triangle (NE\_inversion\_experiment2):



**10 July 2023**

The tangent angle between spheres ( and ( is given by:

Then the sphere to achieve a chop is found by making a direction vector with 0 values for coordinates with no literal in the clause, a -1 for atoms and a 1 for negated atoms. Set the radius and solve for the distance between centers using the equation above and place the center at that distance along the direction vector. Here are example 2D and 3D chops.

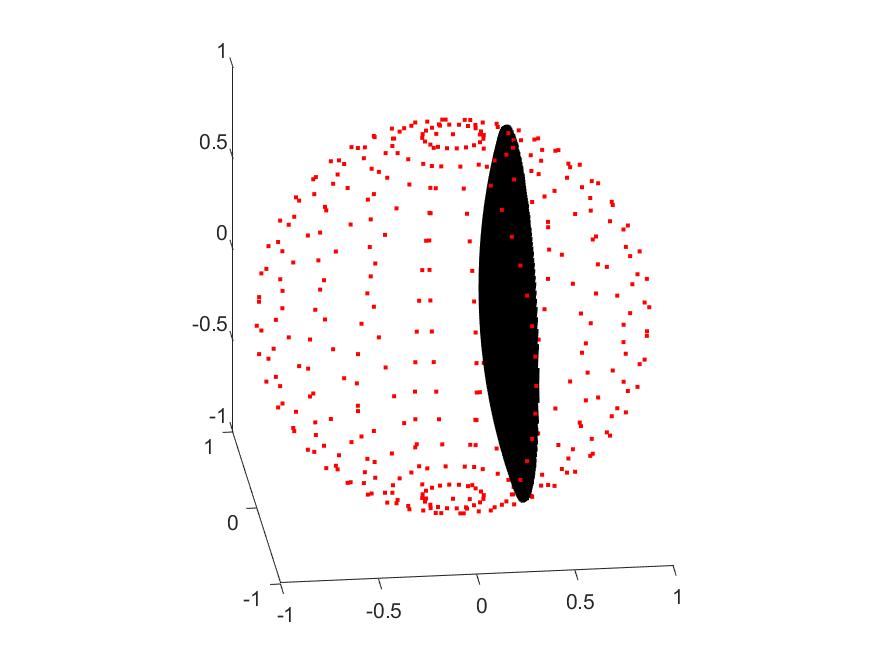
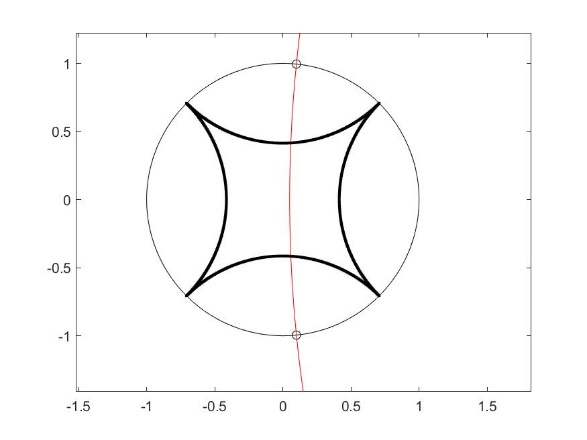
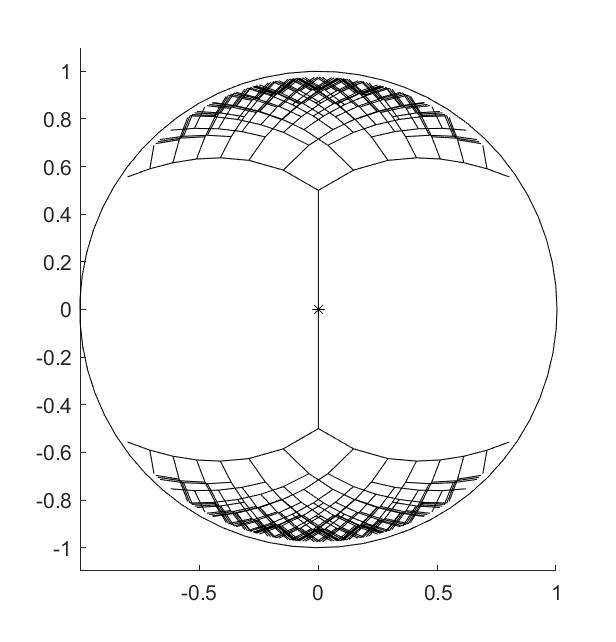


Figure. (left) cutting edge for clause ~a1. (right) Cutting face for clause ~a1.

Developments: Yesterday, we considered the mapping of the hypercube onto a binary tree in the Poincare disk:



Thinking that sub-hypercubes would map to contiguous parts of the circle boundary. It’s not the case, and therefore, loses the desired projective nature.

Also, revisited MVE; this remains a possible option in that the negative results are questionable: they may be due to numerical issues. In order to better understand this, a more careful study of ellipsoid equations is required (from <https://tcg.mae.cornell.edu/pubs/Pope_FDA_08.pdf>, “Alorithms for Ellipsoids, S. Pope, Cornell, 2008).

Let be an ellipsoid centered at with an orthogonal matrix with columns providing the axes of the ellipse and a diagonal matrix with diagonal elements, , where is the length of the semi-axis and are the eigenvalues of:

The principal axes of are the ellipsoid axes and the eigenvalues of are the . We have:

Let , where is an arbitrary orthogonal matrix; then:

Finally, if is factored as , where is lower diagonal and is orthogonal, then :

The E matrix returned by the ipm4mve code does not match any of these. Given ellipse with , (as does ) scales the volume of the hypersphere.

**11 July 2023**

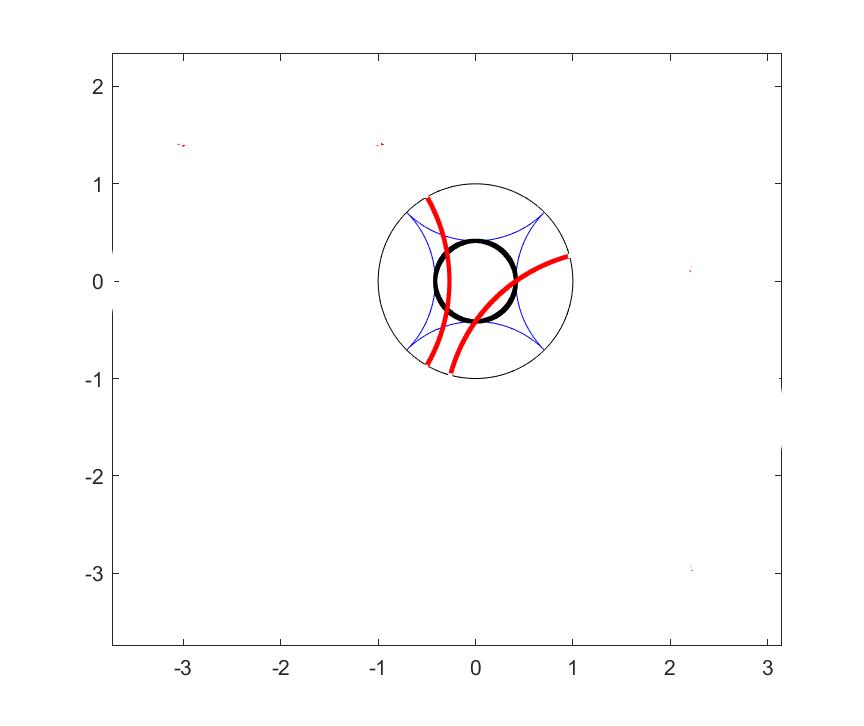
Got chops in 2D to work; equations are:

Let [+-s,…,+-s] be the coordinates of the projected hypercube corners onto ; then let be one of these vectors (say framing the x-axis). Then let be the unit vector in the x direction. Then

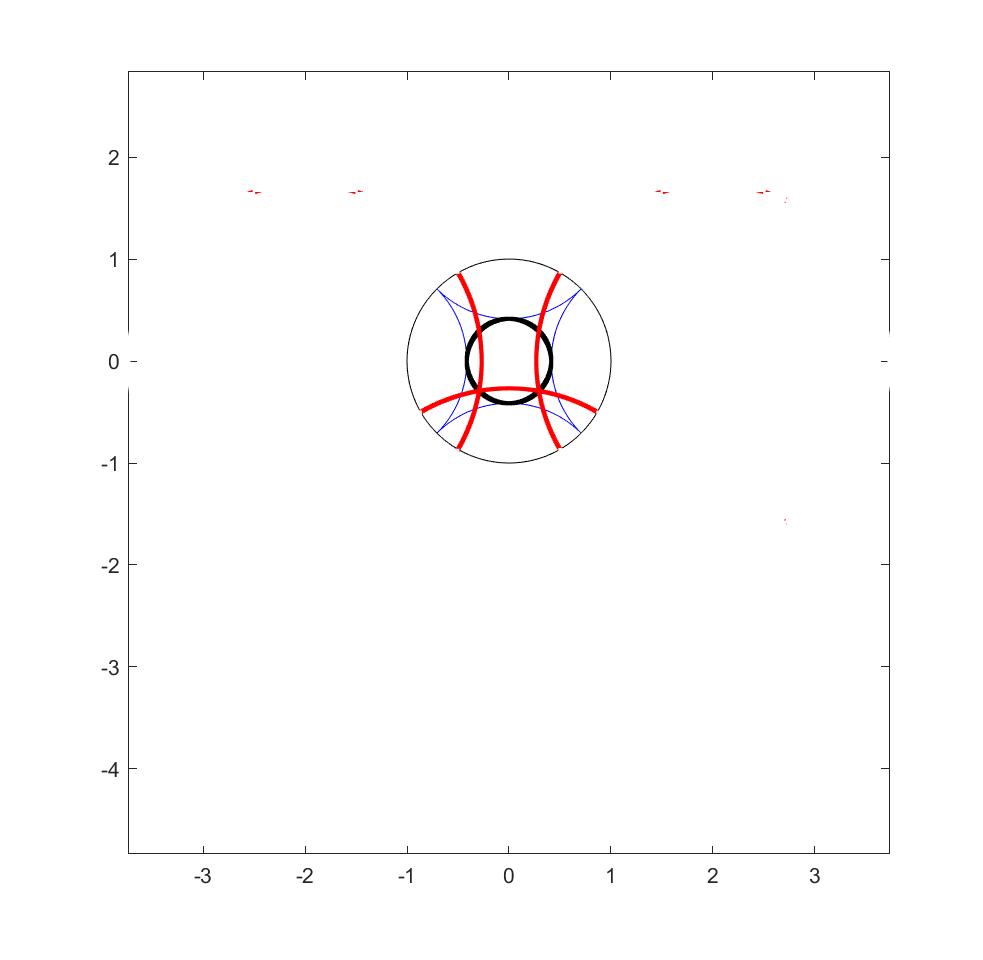
This means:

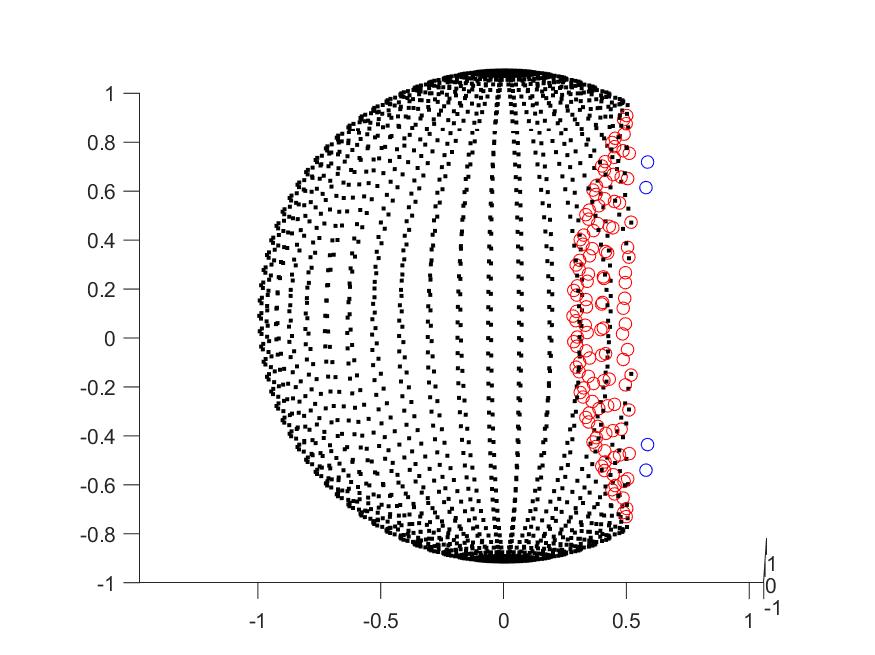
,

Modus Ponens:



Right edge, left edge and bottom edge:



3D chop for KB(1).clauses = [1] (corners are blue circles):

**21 July 2023**

How to create a triangle in the PD from 3 given angles:

Note that given a PD distance, a point that distance from the PD origin and at angle theta and expressed in Euclidean coordinates can be found as follows:

1. Convert PD distance (d) to Euclidean distance (r): r = tanh(d/2)
2. Euclidean point is r\*[cos(theta),sin(theta)]

Also, given the 3 angles of a PD triangle with no right angle, the lengths of the sides are:

and similarly for the other sides.

Then, let one vertex A = [0,0] (i.e., PD origin); let C = [Eb,0], where Eb is Euclidean distance for b; let , where is the Euclidean distance for c, and [,] is the unit vector in direction alpha. Then Angle AB-AC is alpha, AB-BC is beta, and AC-CB is gamma.

Here is a simple example of a 30-40-50 degree triangle:

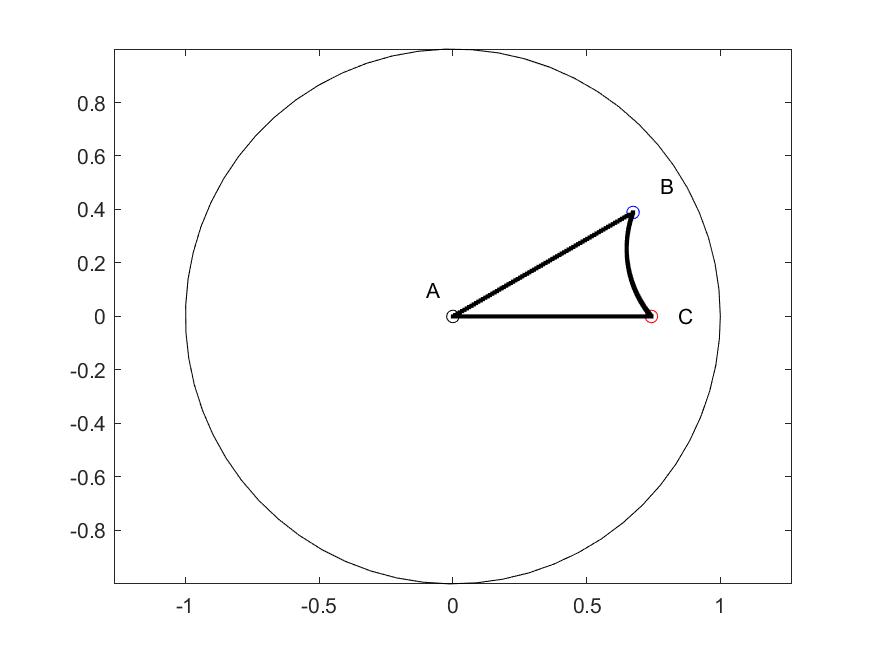


Figure 10. A 30-40-50 degree Triangle.

Matlab function is: NE\_3angles2triangle(30\*pi/180,40\*pi/180,50\*pi/180);

**31 July 2023**

Ellipse info:

Given ( parameters), the implicit coefficient representation is:

Equation of ellipse:

Given A,B,C,D,E,F, the geometric parameters are found as:

Using Foci, and , , where is the length of the major semi-axis.

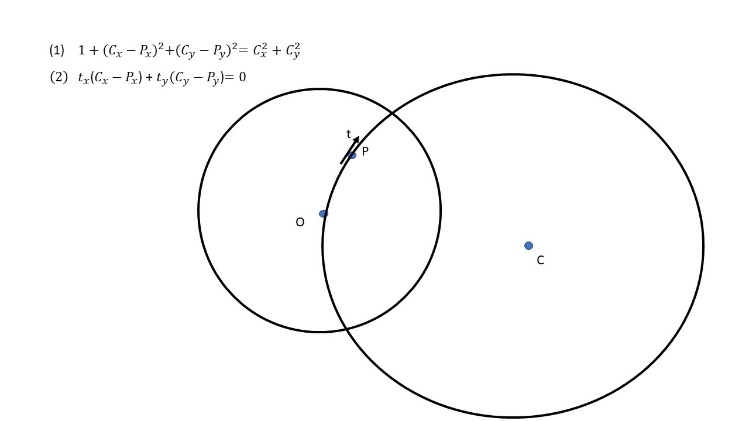
Question: Do points on PD ellipse share the above foci definition? Using PD dist for |P|.

Given geometric parameters, the affine transformation is:

, U = principal axes (for 2D, these are from ), is length of axis.

Convert between PD and Euclidean distance:

Given a point in the PD, and a tangent direction at the point, find orthogonal circle through the point.



Solution:

To find a point along a certain direction from a given point, find the tangent circle; generate sample points on the circle and find closest point to desired distance (NE\_pt\_tan\_dist.m).

Translation of point z in PD: by -b (NE\_PD\_trans.m):

zp = (z-b)./(-conj(b).\*z+1);

Rotation of point z about b by theta:

zp = exp(i\*theta)\*(z-b)./(-conj(b).\*z+1);

Clause to hypersphere algorithm:

Given clause: , convert to

:

Let

Obtain a chop point by substituting -1 for 0 in , if any:

Obtain a non-chopped neighbor by switching a non-zero chop\_pt element:

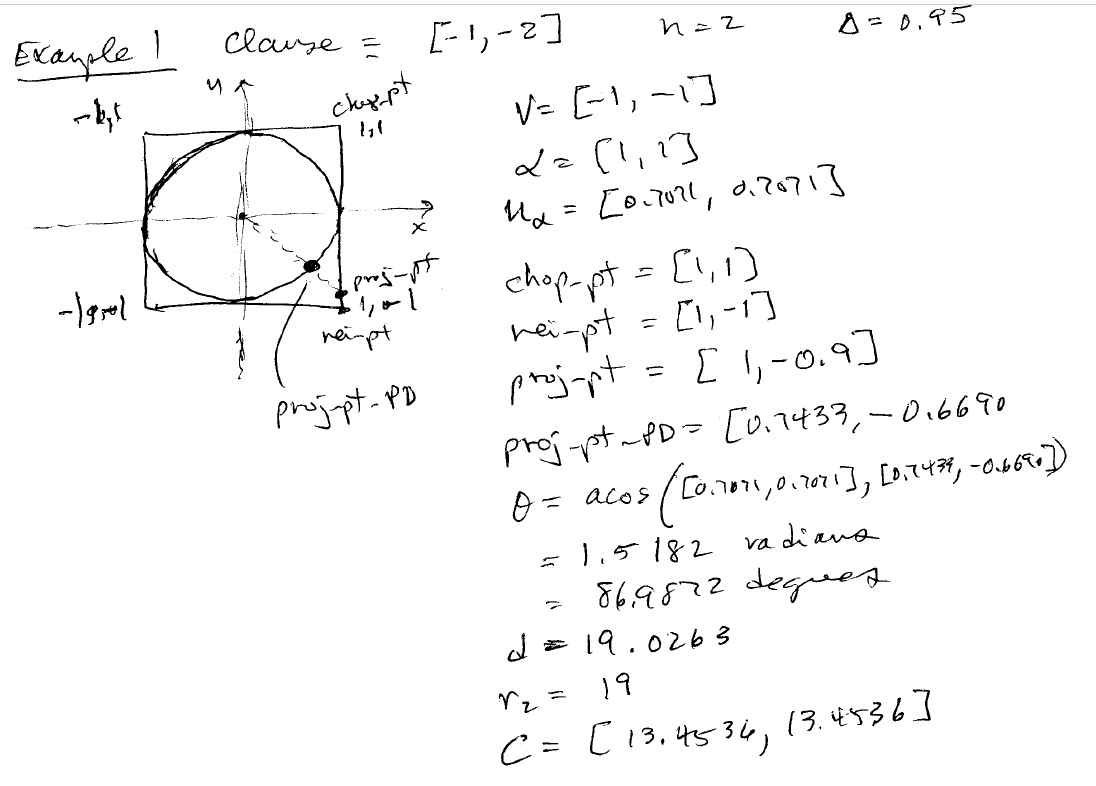
Obtain projection point by sliding along edge connecting chop\_pt to nei\_pt by percentage amount :

Get projection point on boundary of PD:

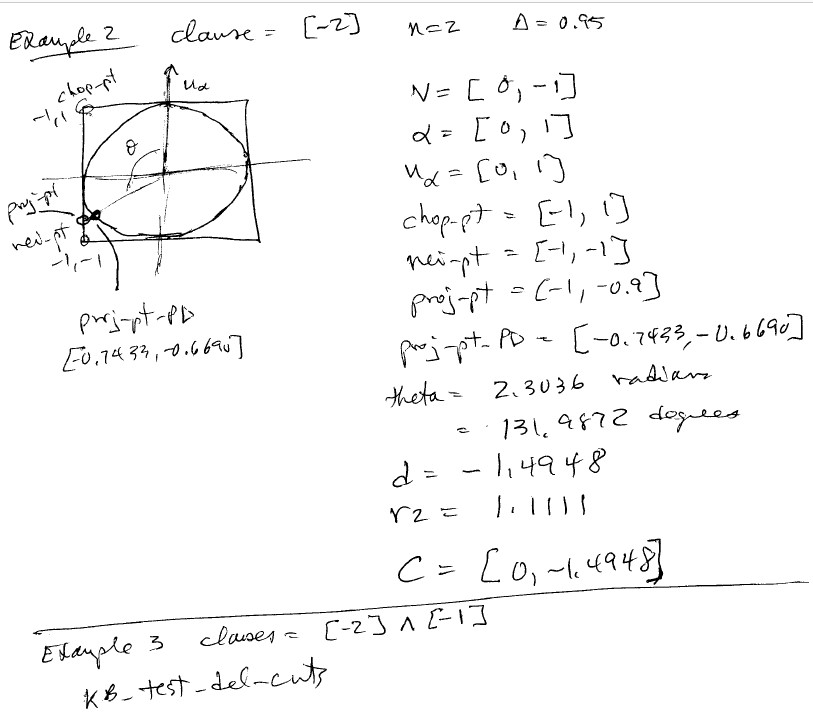
Then,

Note that the center of the hypersphere may flip sides to allow orthogonality; if so, the radius is represented with a minus sign, meaning that the feasible side of the hypersphere is its interior (usually it is the exterior of the hypersphere).

Example 1:



Example 2:



14 August 2023

Problems:

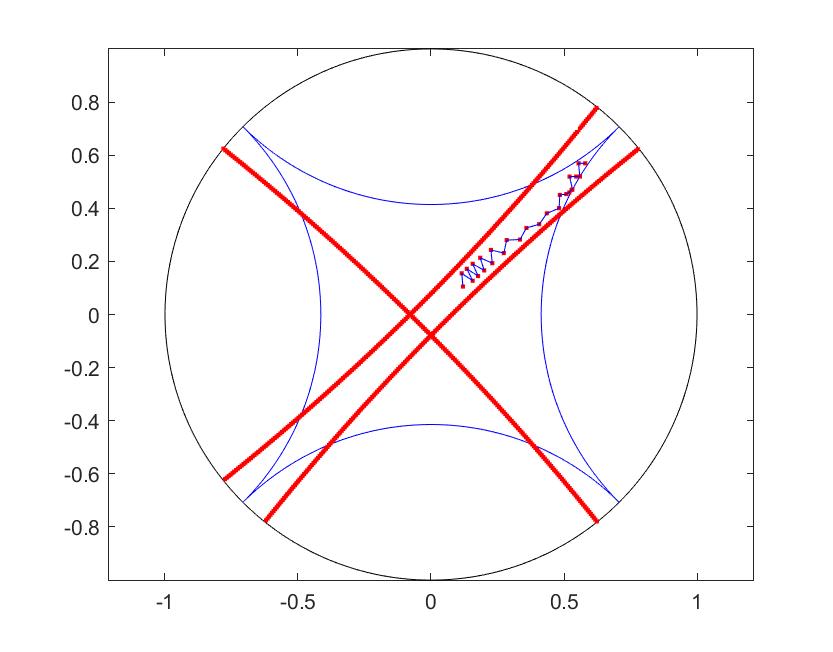
1. Find an initial feasible point (using rand search now)
2. Determine max distance of any point in feasible region of unsatisfiable NCF sentence
   1. Consider intersection of n hyperspheres (pick vertex and n-1 neighbors)

16 August 2023

Formulate Barrier Method as Force Field problem (p. 567, Boyd & Vandenberghe, 2004); for each conjunct constraint:

and projection constraint encoded as:

Here, we simply pick a point in the feasible region, form the sum of the forces, then move some amount in the sum of the forces direction (if that leaves the feasible region, then step distance is cut in half iteratively until next point is in the feasible region). Here’s a 2D example (only solution is (1,1) truth assignment):



is the distance of from the constraint circle.

The general Barrier Method is (p. 569, Boyd & Vandenberghe, 2004):

Given

repeat

1. Centering Step

Compute by minimizing subject to starting at .

1. Update
2. Stepping Criterion: quit if
3. Increase t:

**17 August 2023**

The logarithmic barrier (potential) function:

The function to be minimized:

The distance functions:

where is the center and the radius of the hypersphere.

Then a force field model is defined in terms of forces generated by the minimization function (an impulse to move in a certain direction) and the repulsive force of the constraint surfaces.

For the hypersphere setting, these are given as follows:

🡺

* 🡺

Issues:

1. Error estimate for analytic center (vs. actual atom probabilities)
2. Find , where is Euclidean distance from the origin.
3. Find initial point in feasible region (min t s.t. Ax>=(1-t)b, t>=0 initial t =0
4. Plot force field with and without impulse from minimization function
5. Methods to move
   1. Interior Point Method
   2. Bounce (follow f to wall; bounce off in normal direction; repeat)

**28 August 2023**

Intersection n-D line (l(t) = P0 + t(Q0-P0)) and hypersphere (C,r):

Move C to origin so that x(1)^2 + x(2)^2 + … + x(n)^2 = r^2

Move P0 and Q0 to origin: P = P0 – C; Q = Q0 – C

Then we have:

(P(1)+t(Q(1)-P(1))^2 + … + (P(n)+t(Q(n)-P(n))^2 – r^2 = 0

* [sum(Q(i)-P(i))^2]\*t^2 + [2sum(P(i)(Q(i)-P(i)))]t + [sum(P(i)^2)-r^2] = 0

Solve t1 and t2 from quadradic formula:

If t1 or t2 complex, no intersection

elseif t1 == t2, then 1 point intersection ip = P0 + t(Q0-P0)

else 2 points of intersection ip1 = P0 + t(Q0-P0) ip2 = P0 + t(Q0-P0)

**31 August 2023**

Maximal distance from origin of points in feasible region for CNF sentence which chops each vertex individually. See: ANZIAM J. 59(2017), 271–279, doi:10.1017/S1446181117000372, “A NOTE ON COMPUTING THE INTERSECTION OF SPHERES IN Rn,” D. S. MAIOLI)1, C. LAVOR1 and D. S. GONC¸ ALVES.

This paper shows that the intersection of n hyperspheres whose centers are affinely independent is either empty, 1 point, or 2 points.

Let I\_Dn be the feasible region for the unsatisfiable CNF sentence with n variables that chops each vertex individually; the I\_Dn is the largest feasible region of any unsatisfiable sentence.  The most distant point from the origin can be found as the intersection of n hyperspheres representing chops of the PD feaisble region:   
  1) the hypersphere chopping the vertex representing all true assignments   
  2) the hypersphere chopping the vertex representing the first n-1 variables true and the last false   
  3) the hyperspheres defining the feasible region boundary from the first n-2 axes.   
  
E.g.: 2D:   this is the intersection of the [1,1] and [1,-1] circles   
        3D:    this is the intersection of the [1,1,1] [1,1,-1] and face [1,0,0] chops.   
        ...   
        5D:    this is the intersection of the [1,1,1,1,1], [1,1,1,1,-1], [1,0,0,0,0], [0,1,0,0,0], [0,0,1,0,0] chops.

See function: NE\_D\_max.m

**1 September 2023**

Analytic Center:

Found by function: NE\_analytic center.m (uses gradient descent on sum of constraint forces (repel from constraint surfaces)

7 September 2023

1. So long as there is no 1-literal clause in the KB, then no chop removes the origin; i.e., the origin is always in the feasible region.
2. In the case of the hypercube, the max distance from the center of H\_n for any feasible region arising from an unsatisfiable KB is sqrt(n-2)/2; i.e., goes to infinity in the limit. Likewise, this distance in the Poincare Disk goes to 1 (i.e., infinity) in the limit. At the moment this distance is determined by intersecting n hyperspheres.

2 October 2023

For Euclidean with no face constraints (using CS\_compare\_face\_no\_face.m):

* Max coordinate value =
* Max distance from center =

With face constraints:

* Max coordinate value = 1
* Max distance from center =

Values for n = 3:10

0 0 0 0

0 0 0 0

1.0000 0.8660 1.0000 0.5000

1.5000 2.0000 1.0000 0.7071

2.0000 3.3541 1.0000 0.8660

2.5000 4.8990 1.0000 1.0000

3.0000 6.6144 1.0000 1.1180

3.5000 8.4853 1.0000 1.2247

4.0000 10.5000 1.0000 1.3229

4.5000 12.6491 1.0000 1.4142

**13 October 2023**

Another take:

|  |  |  |  |
| --- | --- | --- | --- |
| N | Max pt dist | Max in x-dir | Max coord val |
| 3 | 0.4005 | 0.3178 | 0.3288 |
| 4 | 0.5465 | 0.2679 | 0.3182 |
| 5 | 0.6001 | 0.2361 | 0.3030 |
| 6 | 0.6307 | 0.2134 | 0.2750 |

Which means that a max coord threshold of 0.33 can be used to make a decision.

Ideas:

1. Every solution vertex in feasible region has at least one face constraint hypersphere through it.
2. Any feasible region vertex created with only hp intersections is not a solution vertex
3. If path points get closer to only hp’s, then quit
4. If path follows an hs, then:
   1. Project the last few points onto the hs
   2. Determine the 1D geodesic curve for these points (on the hs surface)
   3. Check whether its intersection with the unit disk is a solution vertex
5. If forces are symmetric about some (basis?) axis, then move to one side along that axis.

4b Method. To determine the geodesic curve, just linearly extend the last 2 (k?) points and project back to the hs surface. Continue until the projected point is not a feasible point.

# Appendix A: Poincare Half-Plane

# 

# 

# 

# 

# 

# Appendix B: Poincare Disk

A screenshot of a computer

Description automatically generated with low confidence

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# Appendix B: Beltrami-Klein Disk

# 

# 

# 