Extreme Genera and Other Techniques of Graph Embeddings

Eion Blanchard, Esther Hunt[*]

August 11, 2016

Abstract

This paper presents a relatively quick strategy to find an upper bound on the minimum genus of circular graph, details a direction of research in electrical networks by applying Y-Δ transformations to embeddings of complete graphs, and begins to develop an algorithm for depicting a maximum embedding of a graph that is both intuitive and reliable without trial-and-error.

Contents

[*] This paper was written as a part of the University of Washington 2016 Mathematics REU under the direction of Dr. Jim Morrow.

1 Quickly Bounding the Minimum Genus of a Circular Graph

This section is motivated by seeking the minimum genus of a graph with boundary such that the embedding is circular. This is the analogue of circular planar graphs on surfaces of higher genera. We aim here to generalize the Cut Point Lemma for electrical networks. The present authors assume the reader bears familiarity with representing electrical networks as graphs with boundary, as detailed in the book that serves as the crux of the University of Washington Mathematics REU [3].

Definition 1.1. The *minimum genus* of a graph G is the minimum g such that G can be embedded in a surface of genus g. That is, G is drawn without edge crossings.

Definition 1.2. A *minimum embedding* of a graph G is an embedding of G on a surface of the minimum genus of G.

Definition 1.3. A graph G with boundary ∂V is *g-circular* if g is the minimum genus such that G has an embedding on a surface of genus g upon which a boundary circle may be drawn connecting each vertex in ∂V such that every edge of G exists on the same side of the circle and does not cross it.

Remark 1.4. Circular planarity and 0-circularity are equivalent.

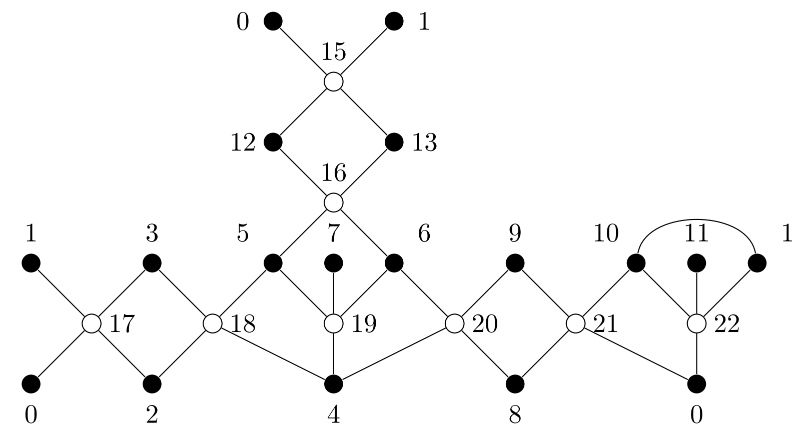
For convenience in the following technique, we now contrive and name a structure for graphs inspired by previous work in the University of Washington Mathematics REU, notably Nick Reichert [8].

Definition 1.5. Given a graph G with boundary ∂V, the *appended graph* G' is G with a new vertex appended such that new edges are drawn between the appended vertex and each boundary vertex in ∂V.

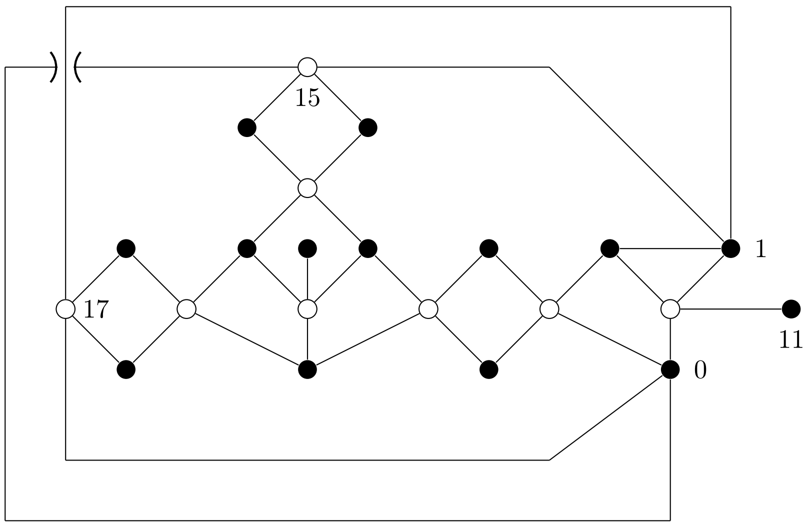
A helpful device for ensuring that a graph is circular is seeking a minimum genus embedding of the appended graph. Since rotation systems may be defined locally at each vertex, any such embedding can be manipulated to see that when the appended vertex and edges are removed, a boundary circle may be drawn cleanly around the boundary vertices according to the rotation system around the appended vertex. For a more technical explanation of rotation systems, consult Section 2.2.

1.1 3-to-1 Graph

The following work was directly inspired by seeking the minimum circular genus of the 3-to-1 graph, drawn below with labels 0 to 13 on the boundary vertices and 15 to 22 on the interior vertices. Notice the thrice-identified vertices 0 and 1 on the wings of the graph [10].



We first seek an embedding of the 3-to-1 graph. We simply identify the 0 and 1 vertices, but notice that the graph is not planar, so we add a handle that corresponds to the edge crossing.



Clearly the minimum genus of the 3-to-1 graph is 1. Now we consider the appended 3-to-1 graph, which includes an appended vertex 14 that is adjacent to each boundary vertex 0 to 13. The diagram of the appended 3-to-1 graph is cumbersome, so we observe the equivalent technique of seeking the minimum genus embedding of the original graph with the restriction that every boundary vertex is incident to some common face.

Generally, this is not possible on the original graph's minimum genus, so we must move up in genus; this motivates the next subsection. Essentially, we try to add as few handles through edges as possible so that one face touches each boundary vertex. In doing so, we exhibit an easily found upper bound on the minimum genus. The resulting number may very well be the actual minimum genus, but this procedure does not ensure this, as we will later demonstrate with the 3-to-1 graph drawn above.

1.2 Pendant-Spike Adjacency Count

First, we recall some terminology from electrical networks.

Definition 1.6. A *boundary pendant* is a boundary vertex with degree one whose only neighbor is another boundary vertex.

Definition 1.7. A *boundary spike* is a boundary vertex with degree one whose only neighbor is an interior vertex.

Notice that in an embedding, since boundary pendants and spikes are leaves in the graph, each will jut out into a face that is homeomorphically unaffected by contraction of the pendant or spike. Now we introduce terminology that will make clear the culminating theorem of this subsection. Recall that every edge in an embedding lies incident on either one or two faces.

Definition 1.8. For unique faces f and f' of a graph embedding, f' is a *face neighbor* of f if some edge lies incident to both f and f'.

Definition 1.9. For a face f in a graph embedding Π, the *pendant-spike adjacency count* $\eta(f, \Pi)$ of f is the number of face neighbors of f that contain boundary pendants or boundary spikes. We impose that f does not contribute to this count if f contains boundary pendants or spikes.

Clearly, η is not graph-invariant, but rather depends on the rotation system detailed, since the face neighbors may vary over different embeddings of the same graph on the same surface if multiple embeddings exist in a given genus.

We now seek to attain a bound for the minimum genus of a circular graph. If we consider a face f in some graph embedding Π, we construct the appended graph by placing the appended vertex on f. We must add a handle for each face neighbor of f that contributes to $\eta(f, \Pi)$.

Theorem 1.10. *Let Δg represent the increase in genus when a graph is appended as detailed by Definition 1.5 on an embedding Π. If the appended vertex is added on face f, then $\Delta g \geq \eta(f, \Pi)$.*

Proof. We notice that Δg is composed of the genus increase induced by adding handles as described to connect the face neighbors counted in η, as well as the genus increase induced by connecting any faces containing remaining boundary vertices. We notate the former count by ∂g. The latter number is nonnegative (though we aim for it to be zero in application of this theorem). So $\Delta g \geq \partial g$.

Let G be a graph and G' be its appended graph with the appended vertex on face f in the embedding Π. Let V, E, F, and g (V', E', F', and g') count the number of vertices, edges, and faces, and the genus, respectively, of G (G').

For brevity, let $n = \eta(f, \Pi)$. We index the face neighbors $f_1, f_2, \ldots, f_n$ of f. Let ∂V_i denote the number of boundary vertices incident to f_i. Let the boundary pendants and boundary spikes on f_i be counted by pV_i and sV_i respectively.

When the appended vertex is added on f and we connect the boundary vertices, we inspect the change on the face count, seeking F' as a function of F. We must add a handle to connect each face neighbor that contributes to η. For each i, we decrement the face count by adding this handle and merging f with f_i. Note here that the theorem seeks to establish that these are the only handles we need to add on any face neighbors of f.

Drawing edges from the appended vertex through the handle to each boundary vertex on the border of the face (i.e. not yet connecting the boundary pendants and spikes), we add $(\partial V_i - pV_i - sV_i) - 1$ faces since there are $\partial V_i - pV_i - sV_i$ such vertices but the first additional edge does not create a new face due to the handle. Now we connect the boundary pendants and spikes, adding back $pV_i + sV_i$. We do not add edges to boundary vertices that have been already connected if a boundary vertex is incident to more than one face neighbor. Then for each i, we add $\partial V_i - rV_i - 2$ where rV_i is the repeated boundary correction factor. So $\sum_{i=1}^{n}(\partial V_i - rV_i) = \partial V$.

Summing over the faces, we see that $\sum_{i=1}^{n}(\partial V_i - rV_i - 2) = \partial V - 2n$. So $F' = F + \partial V - 2n - 1$ since the first added edge of the scheme does not add a face (instead, it essentially renders the appended vertex into a pendant).

We then plug this formula for F' into Euler's equation. Obviously, $V' = V + 1$ since we added a vertex and $E' = E + \partial V$ since we added an edge for every boundary vertex in the original graph. By convention, $g' - g = \partial g$ since we only inspect changes on f and its face neighbors. Thus:

$$\begin{aligned} 2 - 2g' &= V' - E' + F' \\ &= (V + 1) - (E + \partial V) + (F + \partial V - 2n - 1) \\ &= (V - E + F) + (1 - \partial V + \partial V - 2n - 1) \\ &= (2 - 2g) - 2n \\ \iff 2n &= 2g' - 2g \\ \iff n &= \partial g \end{aligned}$$

So $\eta(f, \Pi) = n = \partial g$ implies $\eta(f, \Pi) = \partial g$. Thus, the change in genus required by the above procedure is indeed covered by the handles added to enter each face neighbor of interest. Accounting for any boundary vertices incident to neither f nor any of its face neighbors, then, we conclude $\eta(f, \Pi) \leq \Delta g$ as desired. □

Remark 1.11. If a graph has minimum genus g and is g'-circular, then $\Delta g \geq g' - g$ by the notation in Theorem 1.10.

After the procedure in the proof is followed, we greedily add handles on our embedding to connect the unmet boundary vertices so that we realize the genus of $g + \Delta g$. Utility of this theorem demands that we find an embedding Π that optimizes the following two goals on the selected face f:

1. Maximize the number of boundary vertices incident to f and the face neighbors of f.

2. Minimize the number of face neighbors of f that have incident boundary vertices.

Each boundary vertex not included in the former goal may require its own handle, and each face neighbor of f containing a boundary vertex in the latter goal will require a handle. The more closely that the embedding achieves these goals, the better of an upper bound $g + \Delta g$ will be for g' where the graph G is g'-circular.

Remark 1.12*.* To recap and debrief the notation used in this section:

- The graph G has minimum genus g.

- The appended graph G' has minimum genus g'.

- G is g'-circular.

- By the procedure in the proof of Theorem 1.10 to get Δg for a face f in the embedding Π of G, $g + \Delta g$ is the upper bound on g' dependent on $\eta(f, \Pi)$ from Definition 1.9.

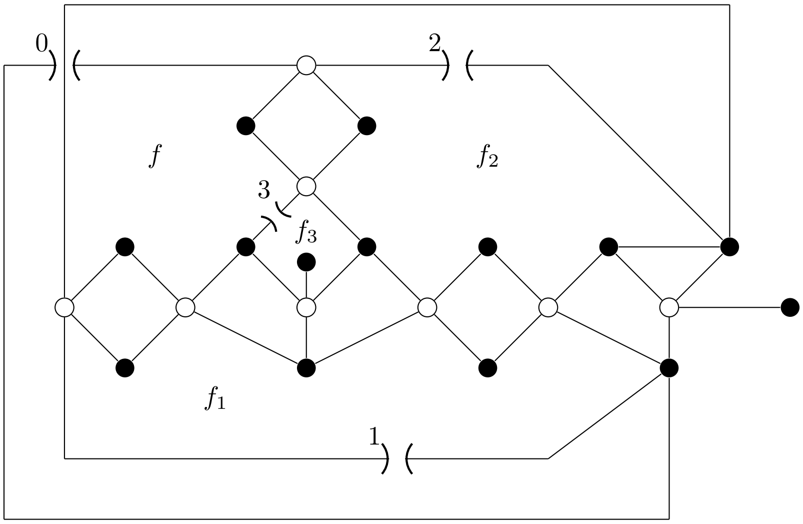
1.3 3-to-1 Graph, revisited

We employ the proof technique on the 3-to-1 graph to show that $g' \leq 4$.

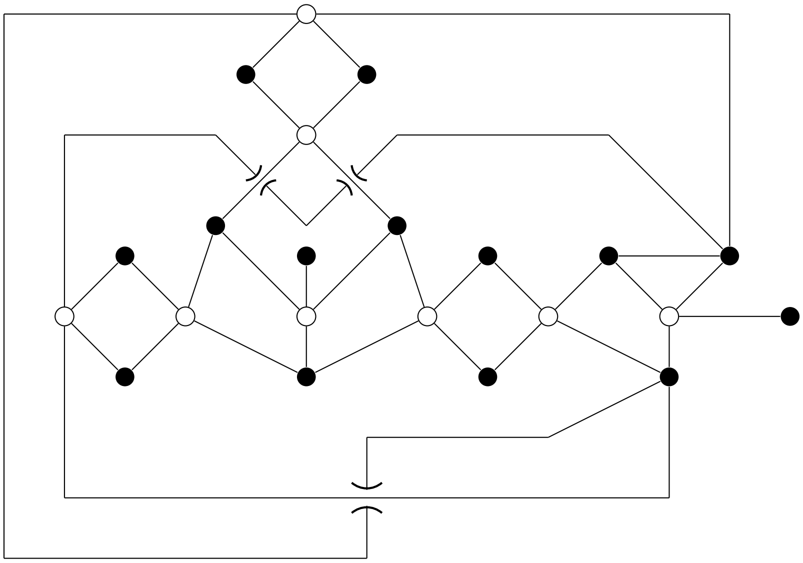
We begin the proof procedure with the embedding of the 3-to-1 graph in a surface of genus 1 as above, where the pre-existing handle corresponds to the labeled handle 0. We let f be as labelled, upon which 6 boundary vertices lay incident before we added any handles.

Note that the way the faces of this embedding are drawn with interior vertices, it is equivalent to consider f_1 and f_2 as face neighbors of f with boundary spikes/vertices. The face f_3 already meets criteria.

We add handle 1 to merge f with f_1. This face now bears 9 boundary vertices. We add handle 2 to merge $f \cup f_1$ with f_2, so the face now bears 13 boundary vertices. We add handle 3 to merge $f \cup f_1 \cup f_2$ with f_3 to complete the face now containing all 14 boundary vertices. This is equivalent to the appended 3-to-1 graph now.



We now observe that the appended 3-to-1 graph can be embedded on a surface of genus 3, as obtained by hand-drawing trials.



Observe that each boundary vertex lies incident on a common face. So where g' is the minimum genus of the appended 3-to-1 graph or where the 3-to-1 graph is g'-circular, we have shown that $g' \leq 3$.

Though the earlier proof strategy obviously did not provide the best value for the g' value we seek for the 3-to-1 graph, it did provide a quick way to get a reasonable upper bound. We proceeded from a pre-existing drawing, while procuring a more accurate value required intuition and myriad re-drawings of the graph.

We conclude this section with a conjecture that the above drawing is indeed the realization of the appended 3-to-1 graph's minimum genus.

Conjecture 1.13. *The 3-to-1 graph is 3-circular.*

This conjecture is equivalent to asserting equality in the already-established $g' \leq 3$. The present authors posit that a method of proof would be exhausting all embeddings of the appended 3-to-1 graph and showing that none exist on surfaces of genera less than 3 and that a method of disproof would be drawing an embedding (or, equivalently, defining a combinatorial rotation system) that lies on a surface of genus less than 3 (probably 2, if any).

A strategy for a counterexample would be to follow the goals set above by forcing a face with many boundary vertices and finding a 1-embedding such, then adding just one (or zero) handles to adjoin a face neighbor incident to every remaining boundary vertex, if any.

2 Triangle Faces in Complete Graphs

We begin with the following conjecture to motivate the section.

Conjecture 2.1. *Every graph that contains a 3-cycle and can be embedded with at least two faces in a surface of given genus can be embedded in that surface with a triangle face.*

This is posed toward applying a Y-Δ transformation on the triangle face in the embedding to decrement the number of faces while maintaining equivalence as an electrical network.

We begin with some standard definitions and theorems that will prop up later discussion.

Definition 2.2. A *2-cell* is a two-dimensional region that is homeomorphic to the open unit disk.

Definition 2.3. A *2-cell embedding* of a graph G is one in which each face is a 2-cell.

Definition 2.4. The *maximum genus* $\gamma_M(G)$ of a graph G is the highest genus of a surface on which G has a 2-cell embedding.

Notice the abuse of nomenclature by the following two definitions, which may be distinguished by context or the number of arguments. Generally, the terms deficiency or Xuong deficiency will be used to refer to the latter, as it is a combinatorial invariant on a given graph.

Definition 2.5. The *Xuong deficiency* $\xi(G, T)$ of a spanning tree T in a graph G counts the number of connected components in the cotree of T in G that have an odd number of edges.

Definition 2.6. The *Xuong deficiency* $\xi(G)$ of a graph G is the minimum value that $\xi(G, T)$ takes over all spanning trees T of G.

Definition 2.7. The *Betti number* $\beta(G)$ of a graph G counts the number of cotree edges of a spanning tree in a graph. This is also called the cycle rank of G and is easily computed $\beta(G) = |E| - |V| + 1$ since every spanning tree has $|V| - 1$ edges.

Theorem 2.8. *(Xuong, 1979) [13] Let G be a graph. Then*

$$\gamma_M(G) = \frac{\beta(G) - \xi(G)}{2}.$$

Definition 2.9. A graph is *upper-embeddable* if it may be embedded (on a surface) with one or two faces.

Definition 2.10. The *cotree* $G - T$ of a tree T in a graph G is the subgraph of G containing all vertices of G and every edge of G not included in T. Observe that the cotree is not necessarily connected and thus is not necessarily a tree.

To make progress toward Conjecture 2.1, we prove the following lemma regarding upper-embeddability.

Lemma 2.11. *(Xuong, 1979) [14] A graph with two disjoint spanning trees is upper-embeddable.*

Proof. Let G be a graph. Since minimizing the face count is the same as maximizing the (essential) genus of an embedding, it will suffice to show that the Xuong deficiency $\xi(G)$ of G is either 0 or 1.

Let T and T' be disjoint spanning trees of G. Since T and T' are disjoint, T' is in the cotree $G - T$ of T. T' is spanning, so $G - T$ is connected.

We observe that the Xuong deficiency on spanning trees is non-negative and maintains parity (i.e. even or odd) across all spanning trees of a given graph. Since $G - T$ is connected, then, it has at most one component with odd edges. Hence $\xi(G, T) = \xi(G)$ is either 0 or 1 as desired. □

2.1 Complete Graphs

We now inspect complete graphs. Recall that the complete graph on n vertices is notated by K_n and is characterized by each vertex being adjacent to every other vertex in the graph.

Theorem 2.12. *All complete graphs are upper-embeddable.*

Proof. We will first exhibit complete graphs K_1, K_2, and K_3. We then will employ Lemma 2.11 on the general K_n graph where n is greater than 3.

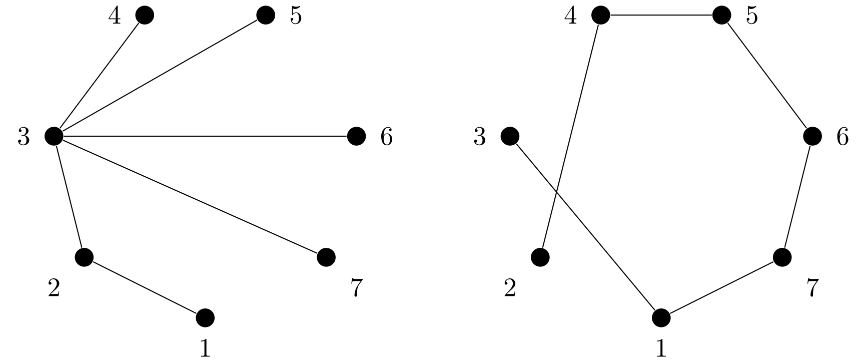
K_1 is a lone vertex and K_2 consists of a single edge connecting two vertices, so the upper-embeddability of each of these graphs is trivial. K_3 is a 3-cycle, and its on the plane contains two faces, so it is trivially upper-embeddable.

We now let $n > 3$ and consider K_n. Label the vertices $1, \ldots, n$. We will exhibit two disjoint spanning trees. Let T be first composed of the edges $(1, 2)$ and $(2, 3)$. Add edges between 3 and each of the vertices $4, \ldots, n$ in star-like fashion on 3. T is now a spanning tree. So $T = \{(1,2),(2,3),(3,4),(3,5),\ldots,(3,n)\}$. Let T' be first composed of the edges $(1,3)$ and $(2,4)$. Add edges in a path from 4 to 1 along the remaining vertices. T' is now a spanning tree. So $T' = \{(1,3),(2,4),(4,5),\ldots,(5,n),(n,1)\}$. T and T' are disjoint spanning trees of K_n, so by Lemma 2.11, K_n is upper-embeddable for $n > 3$.

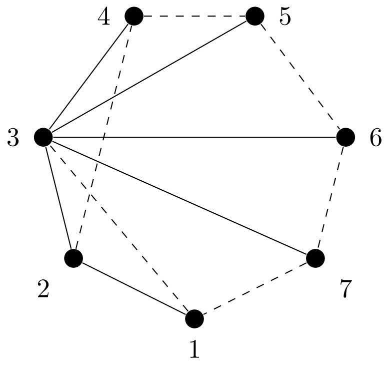
Hence, all complete graphs are upper-embeddable. □

An example of the construction used in the proof of Theorem 2.12 is given for the complete graph on 7 vertices, K_7.

Example 2.13. Consider K_7. On the left, we construct T as in the proof of Theorem 2.12; on the right, T'.



We view T in solid edges and T' in dashed edges in the same drawing to quickly verify that these spanning trees are indeed disjoint.



Hence, K_7 is upper-embeddable by Lemma 2.11.

We use Theorem 2.8 and Theorem 2.12 to construct Table 1, which demonstrates the relations between number of vertices n, Betti number $\beta(K_n)$, Xuong deficiency $\xi(K_n)$, and maximum genus $\gamma_M(K_n)$ for the first fifteen complete graphs.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\beta(K_n)$	0	0	1	3	6	10	15	21	28	36	45	55	66	78	91
$\xi(K_n)$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
$\gamma_M(K_n)$	0	0	0	1	3	5	7	10	14	18	22	27	33	39	45

Table 1: computations on K_n for $1 \leq n \leq 15$.

Observe that for complete graphs, the Betti number, Xuong deficiency, and maximum genus are governed by the following equations:

$$\begin{aligned}
\beta(K_n) &= \binom{n}{2} - n + 1 = \frac{(n-2)(n-1)}{2} \\
\xi(K_n) &= \frac{(-1)^{\left\lfloor \frac{n+1}{2} \right\rfloor} + 1}{2} \\
\gamma_M(K_n) &= \left\lfloor \frac{\binom{n}{2} - n + 1}{2} \right\rfloor = \left\lfloor \frac{(n-2)(n-1)}{4} \right\rfloor .
\end{aligned}$$

Theorem 2.14. *The complete graph on n vertices K_n such that every vertex is in the boundary is g-circular where K_{n+1} has minimum genus g.*

Proof. A proof of this theorem may employ the appended graph from Definition 1.5 and is left to the reader. □

Further work will integrate triangle faces and star-K strategies into analysis of complete graphs.

2.2 Combinatorial Rotation Systems

While it is most intuitive to represent embeddings as drawings of the graph on a two-dimensional representation of the surface, this becomes cumbersome and ambiguous as the average vertex degree increases or the genus increases, so we seek a more technical alternate. We turn to combinatorial (as opposed to coordinate, which will not be presently discussed) rotation systems.

Observe that in the drawing of a graph embedding, the characterizing qualities are not the paths of the edges between adjacent vertices, but rather the local arrangement of incident edges around each vertex. We will inspect simple graphs, so for a fixed vertex, the incident edges each connect to a unique adjacent vertex, Hence, we may encode the order for this vertex as a permutation in cycle notation of the adjacent vertices.

Definition 2.15. A *combinatorial rotation system* for a graph G is a set of cyclic permutations such that each vertex of G is assigned an ordering of its adjacent vertices. Conventionally, a rule (i.e. either *clockwise* or *counter-clockwise*) is associated with the system to make it clear on which side of an adjacent vertex the next vertex in the permutation lies.

Remark 2.16. The present authors will use the counter-clockwise rule for their combinatorial rotation systems unless otherwise noted.

Remark 2.17. Additionally, combinatorial rotation systems may be extended to include an extra dimension for each edge that denotes a twist or the lack thereof, which is necessary to define embeddings on non-orientable surfaces (characterized by their demigenus or Euler genus) such as the Klein bottle. Only orientable surfaces (characterized by their genus) are included in the current scope of this paper, so we eschew this additional parameter. Consult Alex Waldrop's paper from the University of Washington REU in 2011 for combinatorial rotation systems that incorporate twists. [12].

Definition 2.18. A *dart* is a directed edge that, with its negative, corresponds to an undirected edge in a graph. So each edge in a graph gives way to two directed edges, the darts. A dart is represented by the labels of its endpoints with its starting vertex written first (ex. the dart from vertex 3 to vertex 8 is denoted 3–8).

2.3 Deriving the Faces of an Embedding

We may construct the faces of a graph embedding according to a given combinatorial rotation system and its associated rule. From these faces, we construct the surface upon which the graph is embedded. The algorithm for finding the faces for this purpose is presented below.

We will notate a face induced by darts moving counterclockwise along the boundary as $[x–y–z–\cdots–x]$. Let a graph G have vertices labeled 1 through n.

1. We select an arbitrary unused dart $x–y$ from vertex x to vertex y.[1]

2. We initiate the face fragment $[x–y)$ and inspect the initial dart $x–y$.

3. To proceed along the dart $x–y$ on the face fragment $[x–y)$, we consult the cyclic permutation for y and take the successor of x. Vertex z is the y-successor of x where the permutation for y is $(\cdots xz\cdots)$.

4. Next, we add the dart $y–z$ to the face fragment, which becomes $[x–y–z)$.

5. We repeat steps 3 and 4 on this next dart $y–z$ unless we arrive at the initial dart $x–y$ used to construct this face (i.e. when we find some vertex of which y is the x-successor).[2]

6. Without adding the repeated initial dart, the face fragment is now a complete face $[x–y–z–\cdots–x]$. If each dart from the graph is accounted for on the complete faces, we are done. If any darts remain to be used, we repeat the process in steps 1 through 5.

[1] The present authors prefer to start with the first unused dart in the intuitive ordering of darts $1–2, \ldots, 1–n, 2–1, 2–3, \ldots$ as they exist in the graph.

[2] Note that vertices are likely to be repeated on a given face; we do not stop, but rather keep going until a dart is repeated. No dart other than the initial dart on that face will be repeated.

7. All faces are now complete, and all darts have been included. The algorithm is complete, and we have a set of faces derived from the given combinatorial rotation system, as desired.

Remark 2.19. If we seek a drawing of the graph embedding, we take the faces (which have darts as boundary) from the algorithm and piece them together such that each dart x–y is sewn into its negative dart y–x so that each vertex x is identified and each vertex y is identified. This will build a surface of the embedding's genus, which can be computed once the number of faces is known via the Poincaré formula, with edges of the graph as seams.

We will now exemplify the algorithm by constructing the faces of a graph embedding from a combinatorial rotation system.

Example 2.20. We will assemble the faces corresponding to the following combinatorial rotation system for K_5 and use the face count to compute the genus.

1: (2 5 4 3)

2: (1 4 3 5)

3: (1 4 5 2)

4: (1 5 3 2)

5: (1 2 3 4)

We begin the first face with the dart 1–2. We observe that the 2-neighbor of 1 is 4, so we look next at the dart 2–4 and continue this procedure until we return to the dart 1–2. At that point, we have the face [1–2–4–1–3–4–2–3–1].

Next, we take initial dart 1–4. This face is [1–4–5–1].

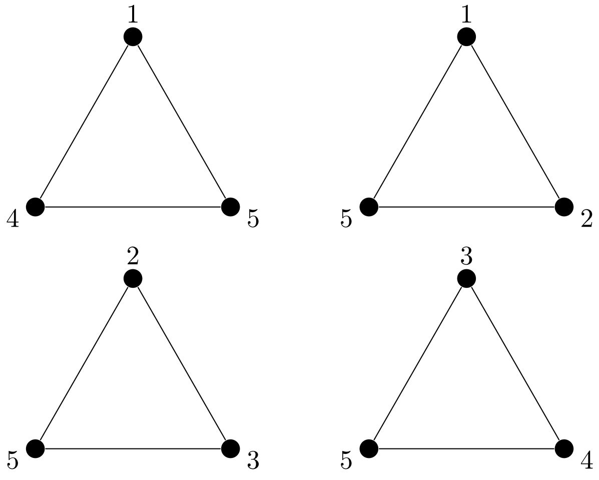
Now taking initial dart 1–5, we discover the face [1–5–2–1].

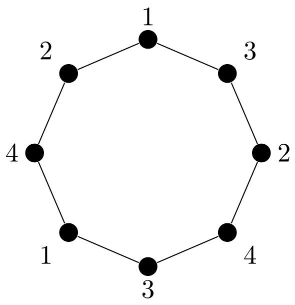
Similarly, the initial dart 2–5 gives us the face [2–5–3–2].

We lastly take the initial dart 3–5 to procure the final face [3–5–4–3].

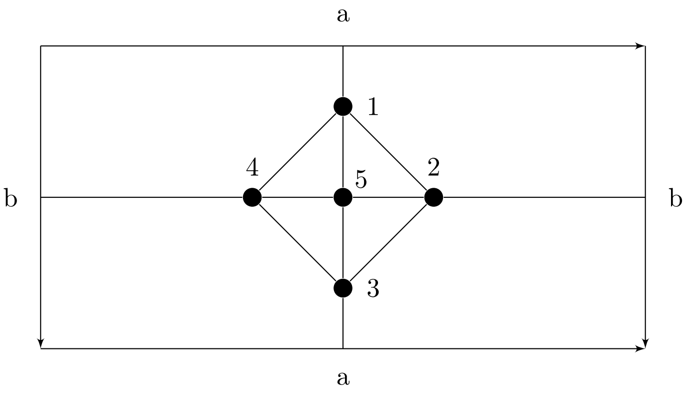
With these 5 faces assembled, we plug in the face count with vertex count of 5 and edge count of 10 into the Poincaré formula to compute that the genus of this embedding is $g = 1$.

We present the faces as polygons with darts as boundary below.





Further, we may sew the corresponding darts on the face boundaries together such that their endpoints align to build the surface of the embedding's genus, which we earlier computed to be 1 for this example. While we eschew the three-dimensional depiction, we depict the embedding in a 1-torus below by representation on a rectangle where the opposite border edges are identified.



From this, one may easily discern the faces we calculated and also envision a 1-torus in three dimensions upon which the graph is embedded as given.

Since we know K_5 is not planar and we know K_5 can be embedded in a surface of genus 1 as in Example 2.20, the minimum genus of K_5 is 1. Applying Theorem 2.14, then, K_4 is 1-circular where each vertex is in the boundary.

More illustrative examples are bountiful but should be intently sought.

3 Drawing a Maximum Embedding

We employ the same definitions for maximum genus and upper-embeddable as in Section 2. In addition, we specify our working definitions for the following common concepts–notably, while matchings are usually considered on vertices of a graph, we will define them on the edges.

Definition 3.1. A *matching* on a graph G is a set of edge pairs, called matches. A given edge may be matched to itself, but may only be in one match.

Definition 3.2. An *adjacency matching* on a graph G is a matching such that the edges in each match are adjacent in G.

Definition 3.3. A *maximum matching* on a graph G is a matching such that no edge is unmatched.

We will now expand upon a few definitions from Section 2.

Definition 3.4. Let T be a spanning tree in a graph G that realizes the Xuong deficiency of the graph as in Definitions 2.5 and 2.6 so that $\xi(G, T) = \xi(G)$. Then T is a *Xuong tree* and $G - T$ is a *Xuong cotree.*

Remark 3.5. With these terms now defined, one sees easily that the Xuong deficiency of a graph is the number of odd connected components of any Xuong cotree in the graph.

We now build up some definitions to a type of graph procedure that will be used in the algorithm.

Definition 3.6. An *edge neighbor* of an edge e in a graph G is another edge e' such that e and e' are each incident to a common vertex.

Definition 3.7. Given a graph G, the *auxiliary graph* G' is obtained by subdividing each edge e of G. If e has endpoints u and v, e has $n = \deg(u)+\deg(v)-2$ edge-neighbors, so we append $n-1$ vertices along e to subdivide e into n edges in G'.

3.1 An Algorithm for the Drawing

Now, we present the algorithm that constructs a drawing of a maximum embedding of a simple graph. Let a simple graph G on n vertices have a fixed combinatorial orientation system and counter-clockwise rule established.

1. We label vertices and edges of G.[3]

2. We impose a maximum adjacency matching M on G. First match each edge of G to an adjacent edge so that at most one edge is left unmatched. Match this edge to itself (this will only occur if the edge count of G is odd).

3. Now, we assemble the auxiliary graph G' from G. Then label each edge in G' subdivided from e in G with two juxtaposed labels as follows. The first label of each is that of the parent edge e. The second labels account for the labels of each edge-neighbor in G of the parent edge e. So if edge x has edge-neighbors y and z in G, we label the edges in G' subdivided from x as xy and xz in arbitrary order.

[3] The labels on vertices and edges of G here and later of G' serve solely to demonstrate that the matchings to follow are well-defined. In practice, we may omit such labels.

4. We establish a maximum matching M' on G' such that edge with reverse labels are matched. That is, xy is matched to yx.[4]

5. By trial-and-error[5], we delete pairs of M' from G' until G' is acyclic. At each deletion, we break as many cycles as possible–we keep track of matches, as M' remains untouched.

6. Now we expand what remains of G' into a spanning tree. Note that we will not restore pairs of edges from G', or else we create a cycle. So, we restore at most one edge from each removed match. Once a spanning tree is obtained, a minimum number of unpaired edges are left unrestored, which is ensured by the efficiency (if improved) or rigor (by exhaustion) of step 5.

7. If an edge originally in G' that was subdivided from e is no longer in G', we delete the parent edge e from G, and it becomes a cotree edge in G. If this missing edge in G' is matched, the parent edge of its match also becomes a cotree edge in G. These two cotree edges in G are thus paired cotree edges. Otherwise, a cotree edge in G is unpaired.

8. We draw a one-face embedding of G, which is easily found since G is now a spanning tree (further work will articulate taking what the present authors call 'bad' faces, corners, etc.).

9. We then restore paired cotree edges to G as follows. For paired cotree edges x and y between u and v and between v and w respectively, we restore x. We then restore y such that it exits the shared vertex v into the face created by the restoration of x to which vertex w is not incident. We add a handle to realize the restoration of y, which increases the genus of the surface upon which we are working but also brings the face count back down.

10. We then restore unpaired cotree edges in the obvious and simple way, creating new faces.[6] G is now embedded in a surface of its maximum genus.

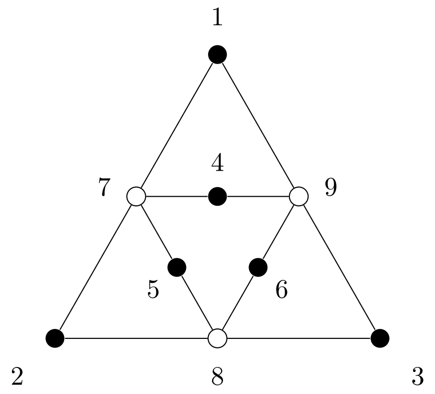
We will now exemplify the algorithm by drawing the maximum embedding of a graph.

Example 3.8. We will construct a maximum embedding of the 2-to-1 graph, drawn below with labels 1 to 6 on the boundary vertices and 7 to 9 on the interior vertices.

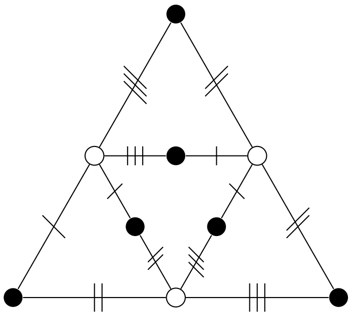
[4] By our construction and labeling of the auxiliary graph G' in step 3, this matching will always be maximum.

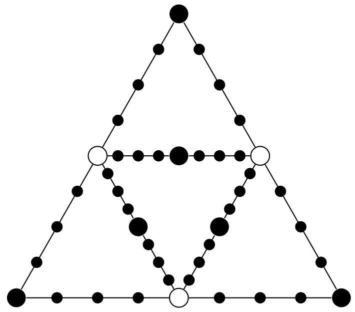
[5] This step is the target for further work, which should aim to essentially find a Xuong cotree. We suspect that in doing so, the auxiliary graph will no longer be a vestige as it currently is.

[6] Notice that if more than one unpaired cotree edge is restored, then $\xi(G) > 1$ and G is not upper-embeddable.



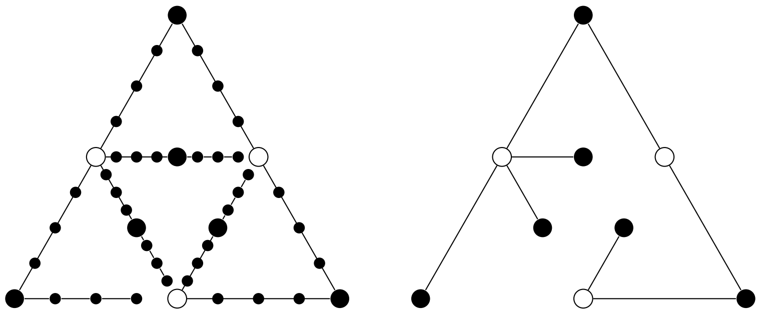
Below on the left, we denote a maximum adjacency matching (observe that notation is abused but remains unambiguous since the matched edges must be adjacent) on G.[7] Below on the right, we construct the auxiliary graph G' of G.





We now delete matched edges from G' until G' is acyclic. By the algorithm description for step 3, recall that the order of the subdivided edges of G' is arbitrary, so in each pair, we delete each edge closest to its co-deleted edge. In this way, we may still see evidence of the matches in the updated G'. For this example, we delete matched edges in G' corresponding to $(2, 8)$ and $(5, 8)$ as well as $(4, 9)$ and $(6, 9)$ in G, as shown below on the left. After these two pairs are removed, G' is a spanning tree, so we need not add back any edges. We translate this spanning tree G' into a spanning tree of G below on the right.

[7] In this example, the maximum adjacency matching was made through trial-and-error to ensure we easily find a maximum embedding. Until the algorithm presented earlier is improved, some maximum adjacency matchings do not lead to a maximum embedding (as will be shown in Example 3.9. However in this matching since the 2-to-1 graph has boundary, we notice that the common vertex of each edge match is an interior vertex, and every interior vertex is the common vertex of at least one edge match.

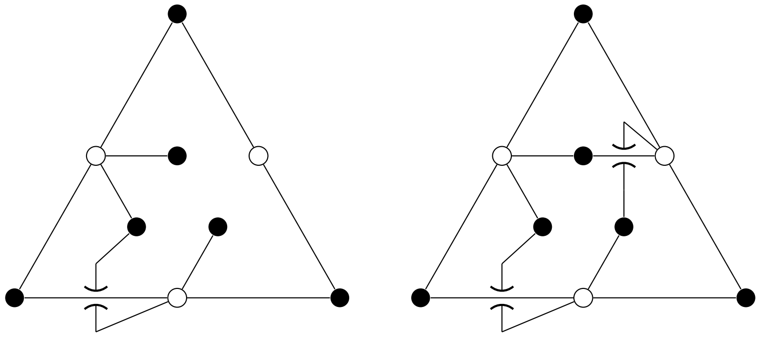


We now restore the paired cotree edges of G.

First working on the match $(2,8)$ and $(5,8)$, we restore the edge $(2,8)$ simply. This creates a face f on the outside of the graph as drawn. Then when we restore the edge $(5,8)$, it must exit vertex 8 into f. Thus we add a handle to connect the edge to vertex 5 on any corner not touching f (in this case, 8 does not lie incident to f, so we may enter on any corner). After the handle is added as drawn below on the left, f is merged with the other face in the graph, returning the face count of the embedding to 1.

Now we work on the match $(4,9)$ and $(6,9)$. We restore $(4,9)$ simply. This creates a rectangle face f' with boundary [1–7–4–9–1]. Then when we restore the edge $(6,9)$, it must exit vertex 9 into f'. Thus we add a handle to connect the edge to vertex 6 on any corner not touching f' (in this case, 6 does not lie incient to f', so we may enter on any corner). After the handle is added as drawn below on the right, f' is merged with the other face in the graph, returned the face count of the embedding to 1.

We then proceed by restoring unpaired cotree edges, but this example of G does not bear any, so we continue on.

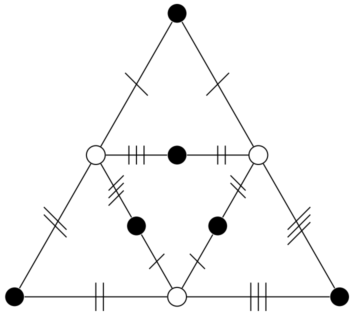


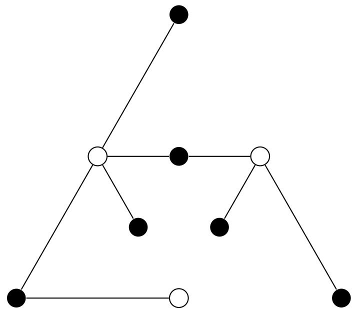
Thus, we are done. The embedding drawn above on the right is indeed a maximum embedding of the 2-to-1 graph, as desired. We notice that the embedding bears a single face, so the 2-to-1 graph is upper-embeddable by Definition 2.9.

3.2 Algorithmic Shortcomings

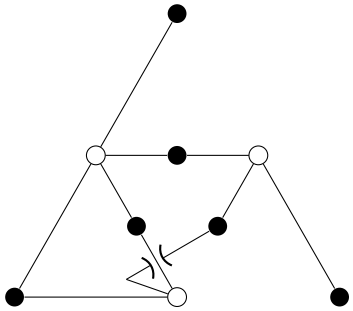
Example 3.9. We now present a bad matching on the 2-to-1 graph that under the algorithm as given, results in an embedding of genus 1. This proves that the algorithm needs adjustment, whether on the step of assigning the maximum adjacency matching, the deletion of pairs, or the expansion of the pared-down auxiliary graph to a spanning tree.

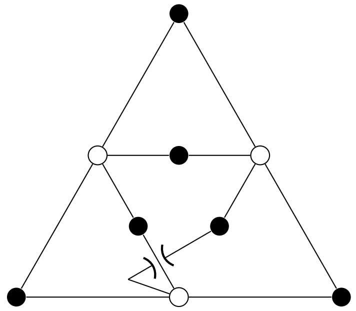
The maximum adjacency matching of G below on the left was found by accident. We abbreviate the algorithm by showing the eventual spanning tree on G below on the right.





As we construct the embedding, we restore one pair of cotree edges below on the left and two unpaired cotree edges below on the right.





This results in three faces and a genus of 1. This would lead us draw the false conclusion that the 2-to-1 graph is not upper-embeddable, but we know that it is indeed upper-embeddable, as demonstrated by Example 3.8.

Further work will flesh out the inefficient facets due to the given simplification of the cited algorithm.

4 Gauging the Circular Planarity of an Electrical Network from its Response Matrix

4.1 Motivation

Up until this point, recovering graphs that are not circular planar has taken much creativity. However there are many response matrices, regardless of the original graph they were formed from, that could correspond to a circular planar graph. In this section, we will lay out a proposed method to incorporate many non circular response matrices into the circular planar case. We begin with a known theorem from Curtis and Morrow [3].

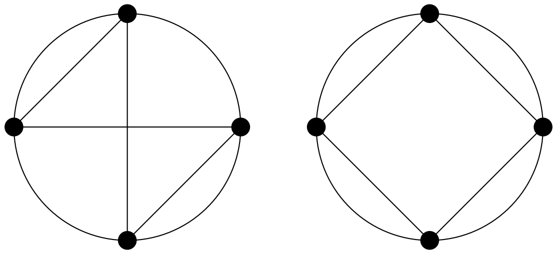
Theorem 4.1. *Suppose Λ is a matrix which satisfies the following properties.*

1. *Λ_γ is symmetric; this is, $\Lambda_\gamma(i;j) = \Lambda_\gamma(j;i)$.*
2. *The sum of the entries in each row is 0.*
3. *For each circular pair $(P;Q) = (p_1, \ldots, p_k; q_1, \ldots, q_k)$,*

$$(-1)^k \det \Lambda_\gamma(P;Q) \geq 0$$

Then there is a circular planar graph G, and there is a conductivity γ on G so that the response matrix for the resistor network (G, γ) is Λ.

What this theorem tells us is that if we have a response matrix that satisfies the proper determinental conditions, then there exists some recoverable circular planar graph that corresponds. Suppose then, you are given an arbitrary response matrix. If you run the determinental conditions and they are not satisfied, then the response matrix does not represent a circular planar graph. However, remember that the response matrix is simply a Kirchhoff matrix, where each row and column represents a boundary node of your graph. Thus, by construction, permuting rows and columns is equivalent to permuting boundary nodes. It is feasible, then, that some response matrices may not correspond to any circular planar graph, but a permuted version of the matrix does.



Permuting the boundary edges changes this graph from non-circular planar to circular planar.

4.2 The 4 Boundary Vertex Case

Clearly there is a brute force algorithm to find such a permutation, or determine that one does not exist. However we begin here an exploration of more efficient algorithms to determine if any circular planar permutation exists given an arbitrary response matrix. We now theorize a specific case in which the first two conditions from Theorem 4.1 imply the third.

Theorem 4.2. *Given a response matrix Λ_{4x4} on four boundary vertices, there exists an underlying circular planar graph G and conductivity γ on G such that the response matrix for the resistor network (G, γ) is Λ_{4x4}.*

Proof. Let Λ_{4x4} be a response matrix with boundary vertices labeled 0, 1, 2, and 3 in circular order. So Λ_{4x4} is symmetric and its row sums are zero.

Since there are four boundary vertices, we inspect circular pairs $(P_1; Q_1) = (0,1;3,2)$ and $(P_2; Q_2) = (0,3;1,2)$. We inspect three cases on the signs of the determinantal conditions.

(a) Both determinantal conditions are nonnegative.
By Theorem 4.1, there exists a circular planar graph G as required.

(b) Exactly one determinantal condition is nonnegative.
Without loss of generality, let $\det(P_1, Q_1) < 0$ and $\det(P_2, Q_2) \geq 0$. Then we have the following two inequalities.

$$\lambda_{0,3}\lambda_{1,2} - \lambda_{0,2}\lambda_{1,3} < 0$$
$$\lambda_{0,1}\lambda_{3,2} - \lambda_{0,2}\lambda_{3,1} \geq 0$$

From these, we observe the following string of inequalities.

$$\lambda_{0,3}\lambda_{2,1} = \lambda_{0,3}\lambda_{1,2} < \lambda_{0,2}\lambda_{1,3} = \lambda_{0,2}\lambda_{3,1} \leq \lambda_{0,1}\lambda_{3,2} = \lambda_{0,1}\lambda_{2,3}$$

Then each $\lambda_{0,3}\lambda_{1,2} < \lambda_{0,2}\lambda_{1,3}$ and $\lambda_{0,3}\lambda_{2,1} < \lambda_{0,1}\lambda_{2,3}$ follow.
Since they are adjacent in the boundary ordering, we now permute vertices 2 and 3 so that our new determinantal conditions are $\lambda_{0,2}\lambda_{1,3} - \lambda_{0,3}\lambda_{1,2}$ (1) and $\lambda_{0,1}\lambda_{2,3} - \lambda_{0,3}\lambda_{2,1}$ (2). By the preceeding inequalities, both determinantal conditions (1) and (2) are nonnegative. We then apply case (a).

(c) Both determinantal conditions are negative.
Let $\det(P_1, Q_1) < 0$ and $\det(P_2, Q_2) < 0$. Then we have the following two inequalities.

$$\lambda_{0,3}\lambda_{1,2} - \lambda_{0,2}\lambda_{1,3} < 0$$
$$\lambda_{0,1}\lambda_{3,2} - \lambda_{0,2}\lambda_{3,1} < 0$$

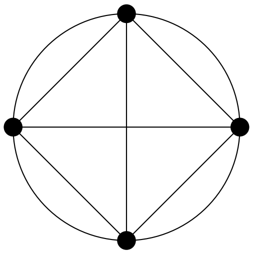
From these, we observe the following string of inequalities.

$$\lambda_{1,2}\lambda_{0,3} = \lambda_{0,3}\lambda_{1,2} < \lambda_{0,2}\lambda_{1,3} = \lambda_{1,3}\lambda_{0,2}$$

Then $\lambda_{1,2}\lambda_{0,3} < \lambda_{1,3}\lambda_{0,2}$ follows.
Since they are adjacent in the boundary ordering, we now permute vertices 0 and 1 so that our new determinantal conditions are $\lambda_{1,3}\lambda_{0,2} - \lambda_{1,2}\lambda_{0,3}$ (1) and $\lambda_{1,0}\lambda_{3,2} - \lambda_{1,2}\lambda_{3,0}$ (2). By the preceeding inequalities, determinantal condition (1) is nonnegative. If (2) is negative, we then apply case (b); else, if (2) is nonnegative, we then apply case (a).

Each case eventually reduces to case (a), which invokes Theorem 4.1. Hence, we witness the circular planar graph G in all cases of the determinantal conditions, as desired. □

So given any 4x4 response matrix, there exists a some permuted response matrix that is circular planar. Notice that this result is not intuitive. Consider the following graph with four boundary vertices that geometrically cannot be permuted to become planar.



4.3 The 5 Boundary Vertex Case

Because of the proof of Theorem 4.2, any subset of four boundary vertices can be forced as circular planar. We can use this to our advantage to "untangle" a subset of larger systems. We will start with the 5x5 case.

Theorem 4.3. *For the 5x5 case, if a response matrix is not in a circular planar order, then at least one 4x4 determinantal pair can be forced strictly positive. In other words, four boundary vertices can be forced into circular planar order.*

Proof. Assume a 5x5 response matrix is not in circular planar order. So there exists at least one determinantal condition that is negative by Theorem 4.1. This determinant will come from a subset of 4 vertices. By Theorem 4.2, we can force this subset of four vertices into circular planar order. Notice that by the proof of 4.2, both determinants will come out strictly positive. Thus for the 5x5 case, if a response matrix is not in a circular planar order, then at least one 4x4 determinantal pair can be forced strictly positive. □

What this means is that if we are attempting to find a circular planar ordering of a given response matrix, we can "lock" four of the boundary vertices in place from the very beginning. These vertices cannot be permuted with regard to each other, or else their determinantal conditions would come out negative. Once four of the vertices are locked in order, determining where to place the last vertex, in the 5x5 case, can be determined simply from the 10 determinants. Thus we arrive at the following algorithm to determine if a circular planar ordering exists of a given 5x5 response matrix exists.

```
Run determinantal conditions of the matrix
If one comes out negative
Then permute the this determinant, along with it's pair as laid out in Theorem 4.2
Begin to run determinantal conditions of the permuted matrix.
For each determinant coming from circular pair P=(a,b) and Q=(d,c)
If the determinant is positive, then the final ordering must be a, d, b, c or a, d, c, b
Elif the determinant is negative, then the final ordering must be a, c, d, b or a, c, b, d
Continue checking determinantal conditions
If a contradiction is reached of required orderings, then there is no permutation that would correspond to a circular planar graph
Else all determinants run, and algorithm ends by choosing a circular planar permutation
```

Notice that by this algorithm, not all determinants would need to be run. All determinants would check only if it was already circular planar. Current work is being done to create a program that will run this for any given 5x5 response matrix.

4.4 Extension to Larger Systems

As soon as we reach a system with 6 boundary vertices, there is a possibility of 3 connections. These would correspond to 3x3 determinantal conditions. Unfortunately, this means direct extention of the algorithm proposed for the 5x5 case is not sufficient. Even if all 2x2 determinants were satisfied by the previous algorithm, it is possible that a 3x3 determinantal condition is unsatisfied. Geometrically, this poses an interesting open problem. What does it mean geometrically if all 2x2 connections are positive, meaning no individual lines cross, but a 3x3 determinant is negative? This question may hold the key to extending the algorithm to higher cases. For an 8x8 case, there are then four connections that must be satisfied, and for a 10x10 case, five connections to be satisfied. Once geometrically understood, these may be able to speed up the algorithm for higher cases.

5 Directions for Further Research

Another direction for algorithm improvement is to question necessity of adjacency in the matching. Can matches that lead to a Xuong cotree require the edges in some match to not share a common vertex?

Look into seeking bounds on the deficiency ξ based on the maximum genus, here notated by γ_M.

Code associated with the previous section and generating combinatorial rotation systems from certain given faces (but not all) should be sought for rigor and ease of conjecture development.

References

[1] O. Biesel and J. Eaton. Notes on multiple embeddings. *University of Washington Mathematics REU*, 2005.

[2] H. Y. Cheung, L. C. Lau, and K. M. Leung. Algebraic algorithms for linear matroid parity problems. *ACM Transactions on Algorithms*, 10(3), 2014.

[3] E. B. Curtis and J. A. Morrow. *Inverse Problems on Electrical Networks.* World Scientific, 2000.

[4] L. Deming and L. Yanpei. An efficient algorithm for finding an adjacency matching of a graph. *Journal of Northern Jiaotong University*, 23(2), 1999.

[5] M. Furst, J. Gross, and L. McGeoch. Finding a maximum-genus graph imbedding. *Journal of the Association for Computing Machinery*, 35(3), 1988.

[6] H. Gabow and M. Stallmann. Efficient algorithms for graphic matroid intersection and parity. In *Automata, languages, and programming: 12th colloquium, Nafplion, Greece, July 15-19, 1985*, pages 210–220, 1985.

[7] Z. Geballe. How extra connections ruin the cut point lemma. *University of Washington Mathematics REU*, 2006.

[8] N. Reichert. Generalized circular medial graphs. *University of Washington Mathematics REU*, 2004.

[9] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination.* American Mathematical Society, 1952.

[10] C. Klumb. Discrete unsolvability for the inverse problem for electrical networks. *University of Washington Mathematics REU*, 2011.

[11] B. Mohar and C. Thomassen. *Graphs on Surfaces.* Johns Hopkins University Press, 2001.

[12] A. Waldrop. Rotation systems and cellular imbeddings. *University of Washington Mathematics REU*, 2011.

[13] N. H. Xuong. How to determine the maximum genus of a graph. *Journal of Combinatorial Theory*, B(26), 1979.

[14] N. H. Xuong. Upper-embeddable graphs and related topics. *Journal of Combinatorial Theory*, B(26), 1979.