Summer 2016

Drawing place field diagrams of neural codes using toric ideals

Nida K. Obatake
San Jose State University

DRAWING PLACE FIELD DIAGRAMS OF NEURAL CODES USING TORIC IDEALS

A Thesis

Presented to

The Faculty of the Department of Mathematics & Statistics

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Nida K. Obatake

August 2016

The Designated Thesis Committee Approves the Thesis Titled

DRAWING PLACE FIELD DIAGRAMS OF NEURAL CODES
USING TORIC IDEALS

by

Nida K. Obatake

APPROVED FOR THE DEPARTMENT OF MATHEMATICS & STATISTICS

SAN JOSÉ STATE UNIVERSITY

August 2016

Dr. Elizabeth Gross	Department of Mathematics & Statistics
Dr. Timothy Hsu	Department of Mathematics & Statistics
Dr. Richard Kubelka	Department of Mathematics & Statistics

ABSTRACT

DRAWING PLACE FIELD DIAGRAMS OF NEURAL CODES
USING TORIC IDEALS

by Nida K. Obatake

A neural code is a collection of codewords (0-1 vectors) of a given length n; it captures the co-firing patterns of a set of neurons. A neural code is convexly realizable in dimension two if there exist n convex sets in $\mathbb{R}^2$ so that each codeword in the code corresponds to a unique intersection carved out by the convex sets. There are some methods to determine whether a neural code is convexly realizable; however, these methods do not describe how to draw a realization, that is, a place field diagram of the code. In this work, we construct toric ideals from neural codes, and we show how we can use these ideals, along with the theory of inductive piercings and Euler diagrams, to draw realizations for particular classes of codes.

DEDICATION

To my family: with your love I can accomplish anything.

ACKNOWLEDGEMENTS

This thesis would not have come to be without the love, support, and advice of many important individuals.

To Dr. Elizabeth Gross, my advisor, thank you for teaching me how to research and write. I appreciate your enthusiastic approach to mathematics, your knowledge, and your constant motivation.

I would like to thank Nora Youngs for being an outstanding collaborator and for working alongside Dr. Gross to ease me into research.

I am extremely grateful for my thesis committee members, Dr. Richard Kubelka and Dr. Tim Hsu, for their insightful comments and attention to detail in the proofreading of this thesis. Thank you both for your continual support throughout my time at SJSU. Thank you to Dr. Bem Cayco for your laughs and to Mrs. Renee Paris for accomplishing any logistical task I needed.

To Zeeshawn and Zerreen, my inspiring brother and sister: thank you for your sense of humor, and for all the laughs that keep me sane. Olive you both! To Mom and Puppa, thank you for being my parents. Your faith in me from the day I was born are the reason I am where I am today. I love you both so much. This thesis is a token of my gratitude for you.

To my husband and best friend, David, thank you. For your encouragement, patience, and sustained support, I am blessed. Thank you for believing in me. I am so thankful for you and I love you forever and after!

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

Figure

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 Motivation from Neuroscience: Place Cells and Place Fields

In the 1970s, neuroscientist John O'Keefe experimented with rats in mazes. He measured the neuronal activity of a rat as it traversed a maze and deduced that there was a special interplay between the geographic location of the rat and the firing pattern of neurons in the hippocampal region of the brain. O'Keefe termed these special neurons *place cells*, and termed the region in the space corresponding to the firing of a place cell a *place field*. O'Keefe and his team were awarded the Nobel Prize in Medicine or Physiology in 2014 for this finding, that the animal's brain creates a spatial map of its environment [OD71].

Scientists can obtain the firing patterns of place cells whilst a rat is in motion. At any particular instance, some neurons are firing while others are dormant. We can model this neuronal activity using binary strings, which we call *codewords*: a 1 indicates that a neuron is active and a 0 indicates that a neuron is dormant. The length of the string is the total number of neurons. We then collect all the codewords obtained and term this collection the *neural code* for the rat in this space.

We assume that the neurons are place cells and that neuron i fires when the animal is in the i-th region of the space modeled by an open set U_i. The collection $\{U_i\}$ is a *place field diagram* of the code and describes the arrangement of the place fields in space [CIVCY13]. If the U_i are all convex and all subsets of $\mathbb{R}^2$, we call the place field diagram convex and say that $\mathcal{C}$ is *convexly realizable in dimension two*. Figures 1.2 and 1.3 show examples of place field diagrams of two different neural

codes, the first with 2 place fields and the second with 6 place fields. These diagrams are convex realizations of the neural codes.

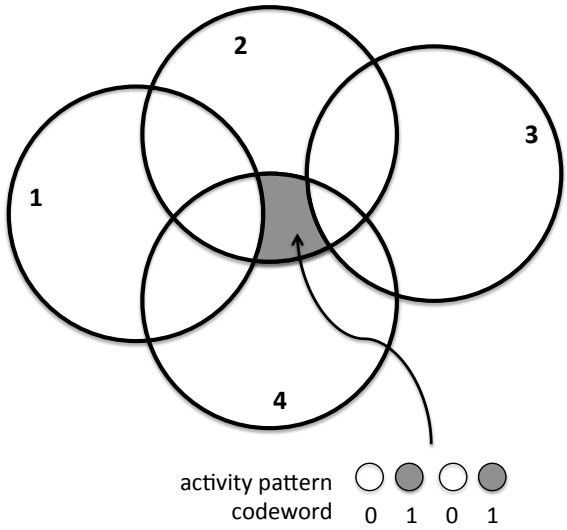


Figure 1.1: A place field diagram of a neural code on 4 neurons.

The guiding question for this thesis is the following:

Question 1.1.1. Given a neural code $\mathcal{C}$ that is convexly realizable in dimension two, how can we draw a place field diagram of $\mathcal{C}$?

It is known that not every neural code is convexly realizable: see [CY15, CIVCY13, CIM$^+$13, CGJ$^+$15, GIK, GI14]. In this thesis, we work under the assumption that certain codes are convexly realizable in dimension two [CGJ$^+$15].

Drawing place field diagrams of a neural code in dimension two with convex place fields is equivalent to drawing what are called *Euler diagrams* of the code, since a place field diagram is an example of an Euler diagram. Euler diagrams are ways of visualizing relationships among sets of data and are well-studied objects that have been studied since the 1700s [Ham60]. A Venn diagram is a common example of an Euler diagram. We formally define Euler diagrams in Section 3.1.

Algorithmically drawing Euler diagrams using convex sets is tricky, but has been studied in the field of Information Visualization [FH02], [Cho07], [RZF08], [SAA09]. Specifically, Stapleton et al. [SZHR11] developed an algorithm to draw Euler diagrams using circles for a class of codes called *inductively pierced* codes. Thus the focus of this thesis becomes the following question:

Question 1.1.2. Given a neural code $\mathcal{C}$, how do we determine if $\mathcal{C}$ is *k-inductively pierced*?

Once we have determined that a code $\mathcal{C}$ is k-inductively pierced and answered Question 1.1.2, we will have demonstrated that we can draw a place field diagram of $\mathcal{C}$ using disks, that is, that $\mathcal{C}$ is convexly realizable in dimension 2. At this point, we answer Question 1.1.1 by inputting the code into the algorithm developed by [SZHR11] and obtaining a place field diagram of $\mathcal{C}$, as desired.

This thesis is organized as follows: In Section 1.2 we formally define neural codes and place field diagrams. In Section 1.3 we provide necessary background information from algebra and topology. In Chapter 2 we define a map $\phi_{\mathcal{C}}$ corresponding to a neural code $\mathcal{C}$, which we will use to define the *toric ideal $I_{\mathcal{C}}$ of a neural code*, a computational object we will use to analyze the neural code $\mathcal{C}$. We conclude Chapter 2 by explaining immediate conclusions that can be concluded from $I_{\mathcal{C}}$. In Chapter 3 we show that drawing place field diagrams is equivalent to drawing Euler diagrams using circles. In particular, we develop language from the field of Information Visualization, applying it in the context of neural codes. Once we have set up the theory of piercings and the notion of k-inductively pierced codes, in Chapter 4 we use the toric ideal to determine whether a code is inductively pierced. We extend a result from Chapter 4 in Section 4.4 using Gröbner bases. In Chapter 5, we end the thesis by summarizing our main results and illustrating our

findings by drawing a place field diagram of a 6-neuron code.

1.2 Neural Codes and Place Field Diagrams

A neural code[1] is a form of discretized data that arises in neuroscience [CY15, CIVCY13, CIM$^+$13, CGJ$^+$15, GIK, GI14]. In particular, a *neural code* $\mathcal{C} \subseteq \{0,1\}^n$ is a set of binary vectors that record the firing activity of n neurons labeled $[n] = \{1, 2, 3, \ldots, n\}$. We will refer to an element of a neural code as a *codeword*, $\mathbf{c} = (c_1, \ldots, c_n) \in \mathcal{C}$; each codeword corresponds to a subset of firing neurons determined by $\operatorname{supp}(\mathbf{c}) \stackrel{\text{def}}{=} \{i \in [n] \mid c_i = 1\} \subseteq [n]$.

The region U_i in space X where $c_i = 1$ is the *place field* of neuron i. We will consider neural codes $\mathcal{C}$ that are place field codes. A place field code comes from a place field diagram, as defined in the following definition:

Definition 1.2.1 (Place field code [CIVCY13])**.** Let $X \subseteq \mathbb{R}^d$ (we call X the stimulus space), and let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be a collection of open sets from $\mathbb{R}^d$, where each $U_i \subseteq X$ is the place field of the i-th neuron. The *place field code* $\mathcal{C}(\mathcal{U}) \subseteq \{0,1\}^n$ is the set of all binary codewords corresponding to stimuli in X:

$$\mathcal{C}(\mathcal{U}) \stackrel{\text{def}}{=} \left\{ \mathbf{c} \in \{0,1\}^n \mid \left(\bigcap_{i \in \operatorname{supp}(\mathbf{c})} U_i \right) \setminus \left(\bigcup_{j \notin \operatorname{supp}(\mathbf{c})} U_j \right) \neq \emptyset \right\}.$$

Given a neural code, note that there always exists a $\mathcal{U}$ such that $\mathcal{C} = \mathcal{C}(\mathcal{U})$, so every code is *realizable* [CGJ$^+$15, CY15]. A code is realizable if it is a place field code, and if so, the collection $\mathcal{U}$ is called a *place field diagram* of $\mathcal{C} = \mathcal{C}(\mathcal{U})$.

As an example, consider the following code on two neurons: $\mathcal{C} = \{00, 10, 01, 11\}$. A place field diagram of $\mathcal{C}$ is illustrated in Figure 1.2. Each region in the diagram is called a *zone* and has a corresponding codeword in the

[1] Every $\mathcal{C} \subseteq \{0,1\}^n$ is a neural code, so a neural code is a binary code.

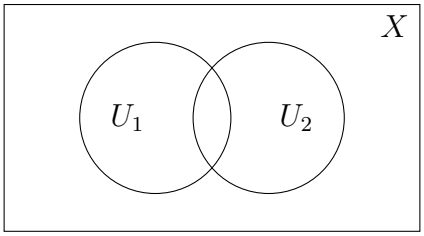


Figure 1.2: A place field diagram of $\mathcal{C} = \{00, 10, 01, 11\}$

code. For example, the region $U_1 \cap U_2$ is the zone {1,2}, which corresponds to the codeword 11. The realization of this code has 4 total zones, one zone for each codeword, including the all zeroes codeword.

Now, consider a more complicated neural code $\mathcal{C}$ on six neurons, $\mathcal{C} = \{000000, 100000, 010000, 001000, 000100, 000010, 110000, 011000, 000011, 001100, 000110, 100010, 110010, 010010, 010100, 010110, 011100\}$. It turns out this code is convexly realizable in dimension 2 and a place field diagram of $\mathcal{C}$ is pictured in Figure 1.3.

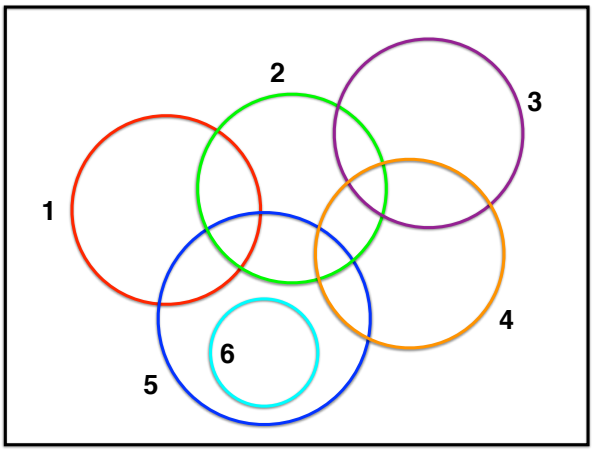


Figure 1.3: A place field diagram of a six neuron code $\mathcal{C}$

We see that in this code we have triples of place fields intersecting, such as U_1, U_2, and U_5. Notice that since U_5 covers U_6, the codeword $000001 \notin \mathcal{C}$. The realization of this code is made up of 17 zones, one for each codeword.

1.3 Fundamentals from Algebra and Topology

We will be considering place field diagrams which are realizable in $\mathbb{R}^2$. We will require that each place field U_i be an open, convex set in the plane. (In fact, as will become clear in Section 3.1, we will make the additional assumption that the U_i are disks in $\mathbb{R}^2$.) In Section 2.1, we will define a ring homomorphism $\phi_{\mathcal{C}}$. This map will be our main tool for determining whether a code is k-inductively pierced. We start with the basic definitions necessary to define $\phi_{\mathcal{C}}$.

Definition 1.3.1 ([Hun74])**.** A *commutative ring with unity* is a nonempty set R together with two binary operations (usually denoted as addition and multiplication) such that:

(1) under addition, R is an abelian group;

(2) multiplication is associative and commutative;

(3) multiplication is distributive over addition;

(4) there exists a multiplicative identity element $1 \in R$.

Some common rings are the set of all integers $\mathbb{Z}$ with the usual addition and multiplication; the set of all real numbers $\mathbb{R}$, again with the usual addition and multiplication; and the set of all integers modulo 2, $\mathbb{Z}_2 = \{\bar{0}, \bar{1}\}$, with addition and multiplication mod 2.

Recall that $\mathbb{C}$ is the set of all complex numbers. The complex numbers form a ring under addition and multiplication. The real numbers $\mathbb{R}$ and the complex

numbers $\mathbb{C}$ have an added condition which makes them fields: a *field* is a ring in which each element (except the additive identity) has a multiplicative inverse. For $\mathbb{C}$, it is certainly true that any complex number $a + bi$ (where a and b are not both 0) has a multiplicative inverse $\dfrac{a - bi}{a^2 + b^2}$.

An important example of a ring is the polynomial ring in one variable. In this ring, we can add and multiply polynomials to get other polynomials under the usual polynomial operations.

Example 1.3.2. Let K be a commutative ring, and let x be an indeterminate. Then $K[x]$ is a commutative ring called a polynomial ring whose elements are polynomials in x of the form $p = p_0 + p_1x + p_2x^2 + \ldots + p_{m-1}x^{m-1} + p_mx^m$ for $p_i \in K$.

A ring homomorphism is a map between rings which preserves the operations of the rings:

Definition 1.3.3 ([Hun74])**.** Let R and S be rings. A function $f : R \to S$ is a *homomorphism of rings* provided that for all $a, b \in R$:

$$f(a + b) = f(a) + f(b) \text{ and } f(ab) = f(a)f(b).$$

Given any morphism, it is natural to consider the kernel of the map, which is the set of all elements in the domain which map to the additive identity element of the codomain.

Definition 1.3.4 ([Hun74])**.** Let $\mathbb{K}$ be a field, and let $f : \mathbb{K}[x_1, \ldots, x_m] \to \mathbb{K}[y_1, \ldots, y_n]$ be a homomorphism of polynomial rings. The *kernel* of f is $\text{Ker}f = \{p \in \mathbb{K}[x_1, \ldots, x_m] \mid f(p) = 0\}$.

The kernel $\text{Ker}f$ of a ring homomorphism $f : R \to S$ is an ideal of the domain ring R, meaning it is a *subring* under the operations of R (it contains the additive

identity from R, is closed under the operations of addition, negation, and multiplication of R), and it is closed under left and right multiplication by elements from R.

Definition 1.3.5 (Subring of a Ring)**.** [Hun74] Let R be a ring and S a nonempty subset of R that is closed under the operations of addition and multiplication in R. If S is itself a ring under these operations then S is called a *subring* of R.

Definition 1.3.6 (Ideal of a Ring)**.** [Hun74] A subring I of a ring R is an ideal provided $r \in R$ and $x \in I \Rightarrow rx \in I$, $xr \in I$.

The kernel of a homomorphism of rings is an ideal in the domain ring, as defined by the following theorem:

Theorem 1.3.7. *[Hun74] If $f : R \to S$ is a homomorphism of rings, then the kernel of f is an ideal in R.*

We have now defined the algebraic and topological machinery necessary to introduce our main object of study, the toric ideal of a neural code $\mathcal{C}$. In Section 2.1 we will define a monomial map $\phi_{\mathcal{C}}$ for a given neural code $\mathcal{C}$. Eventually, our goal is to use the kernel of this map to help determine whether we can apply the algorithm from [SZHR11] to draw a place field diagram of the code. We will explore this concept further in Chapter 4.

CHAPTER 2

TORIC IDEALS OF NEURAL CODES

2.1 Toric Ideal of a Neural Code

Let $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$ be a neural code on n neurons and let $\mathcal{C}^* \stackrel{\text{def}}{=} \mathcal{C} \setminus \{00\ldots00\}$, i.e. $\mathcal{C}^*$ is $\mathcal{C}$ with the all zeros word removed. A code on n neurons has $|\mathcal{C}| = n$.

Definition 2.1.1 (The neural homomorphism of a neural code.)**.** Let $\mathbb{K}$ be a field, and define two polynomial rings: $\mathbb{K}[p_{\mathbf{c}} \mid \mathbf{c} \in \mathcal{C}^*]$, in which the m indeterminates are indexed by the codewords of the code $\mathcal{C}$ and are of the form $p_{\mathbf{c}}$ form $\mathbf{c} \in \mathcal{C}^*$, and $\mathbb{K}[x_i \mid i \in \{1, \ldots, n\}]$, in which the n indeterminates are indexed by the neurons of $\mathcal{C}$. Then *neural homomorphism* $\phi_{\mathcal{C}}$ of $\mathcal{C}$ is the ring homomorphism $\phi_{\mathcal{C}} : \mathbb{K}[p_{\mathbf{c}} \mid \mathbf{c} \in \mathcal{C}^*] \longrightarrow \mathbb{K}[x_i \mid i \in \{1, \ldots, n\}]$ defined by $p_{\mathbf{c}} \longmapsto \prod_{i \in \text{supp}(\mathbf{c})} x_i$.

Definition 2.1.2 (Toric ideal of a neural code)**.** The *toric ideal* of $\mathcal{C}$ is the kernel $I_{\mathcal{C}}$ of the map $\phi_{\mathcal{C}}$, so $I_{\mathcal{C}} \stackrel{\text{def}}{=} \ker(\phi_{\mathcal{C}})$.

Note that since $\phi_{\mathcal{C}}$ is a monomial map, the ideal $I_{\mathcal{C}}$ is generated by binomials (Lemma 1.1) [Stu96]. In Chapter 4, we will show that we can use these binomial generators to understand important intersection information among the place fields in a realization of the neural code.

Using the program *Macaulay2* [GS] with the `4ti2` package [tt], we are able to compute the generators of the toric ideal of a given neural code. We enter a neural code $\mathcal{C}$ as a matrix whose columns are the codewords of $\mathcal{C}$ and invoke `toricMarkov` to calculate a generating set (not necessary minimal) for the toric ideal $I_{\mathcal{C}}$.

Formally, if $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$ and $\mathcal{C} \subseteq \{0,1\}^n$ then the matrix of $\mathcal{C}$ is the $n \times m$ matrix $A_\mathcal{C}$ whose ith column is $\mathbf{c}_i^{\mathrm{T}}$, so $A_\mathcal{C} = [\mathbf{c}_1^{\mathrm{T}} \ \ \mathbf{c}_2^{\mathrm{T}} \ \ \cdots \ \ \mathbf{c}_m^{\mathrm{T}}]$.

Remark. To ensure nontrivial results, we omit the all zeroes codeword when entering the matrix of the code and computing its toric ideal.

Example 2.1.3. Let us look at examples of $\phi_\mathcal{C}$ in order to understand what the ring homomorphism looks like for different codes $\mathcal{C}$. For each example we will identify the image $\phi_\mathcal{C}(p_\mathbf{c})$ for each generator $p_\mathbf{c}$ of $\mathbb{K}[p_\mathbf{c} \mid \mathbf{c} \in \mathcal{C}^*]$.

Consider the code C2= $\{000, 100, 010, 101\}$. Images of elements under ϕ_{C_2} are elements of $\mathbb{K}[x_1, x_2, x_3]$. Then, $\phi_{\mathrm{C2}}(p_{100}) = x_1$, $\phi_{\mathrm{C2}}(p_{010}) = x_2$, and $\phi_{\mathrm{C2}})(p_{101}) = x_1x_3$.

Consider the code A12= $\{000, 100, 110, 111\}$. Then $\phi_{\mathrm{A12}}(p_{100}) = x_1$, $\phi_{\mathrm{A12}}(p_{110}) = x_1x_2$, and $\phi_{\mathrm{A12}}(p_{111}) = x_1x_2x_3$.

Let $\mathcal{C} = \{00000, 10000, 11000, 10100, 11100, 01000, 00010, 01010, 01011\}$. The images of the generators of $\mathbb{K}[p_\mathbf{c} \mid \mathbf{c} \in \mathcal{C}^*]$ are x_1, x_1x_2, x_1x_3, $x_1x_2x_3$, x_2, x_4, x_2x_4, and $x_2x_4x_5$.

The following examples illustrate the computation of the toric ideals of codes on three neurons. Appendix A.1 lists all codes on three neurons up to symmetry obtained from [CY15], and Appendix A.2 catalogs all their toric ideals. The names of these codes, e.g. A1 and A2, come from the naming of the three neuron codes from [CY15].

Example 2.1.4. Let $\mathcal{C} = \mathrm{A1} = \{000, 100, 010, 001, 110, 101, 011, 111\}$, a neural code on three neurons.

The corresponding matrix of this code (omitting the all zeroes codeword as

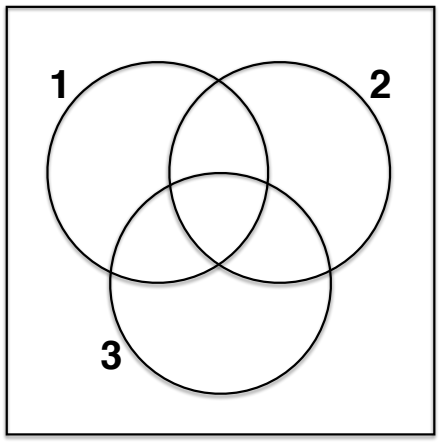


Figure 2.1: A place field diagram of the neural code A1.

noted) is

$$A_{\mathcal{C}} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

By [tt], the toric ideal of this code, I_{A1}, is generated by the following cubic and quadratics: $f_1 = p_{111} - p_{100}p_{010}p_{001}$, $f_2 = p_{110} - p_{100}p_{010}$, $f_3 = p_{101} - p_{100}p_{001}$, and $f_4 = p_{011} - p_{010}p_{001}$. To confirm that these binomials are elements of the kernel, we check the image of each of these binomials. In particular:

$$\phi_{\text{A1}}(p_{111} - p_{100}p_{010}p_{001}) = x_1x_2x_3 - x_1 \cdot x_2 \cdot x_3 = 0$$

$$\phi_{\text{A1}}(p_{110} - p_{100}p_{010}) = x_1x_2 - x_1 \cdot x_2 = 0$$

$$\phi_{\text{A1}}(p_{101} - p_{100}p_{001}) = x_1x_3 - x_1 \cdot x_3 = 0$$

$$\phi_{\text{A1}}(p_{011} - p_{010}p_{001}) = x_2x_3 - x_2 \cdot x_3 = 0.$$

The set of binomials $\{f_1,\ f_2,\ f_3, f_4\}$ is a generating set of the kernel since any element of the kernel can be written in the form $\sum_{i=1}^{4} a_i f_i$ where $a_i \in \mathbb{K}[p_{\mathbf{c}} \mid c \in \text{A1}]$. The code A1 is convexly realizable in dimension two as illustrated in Figure 2.1.

Example 2.1.5. Let $\mathcal{C} = \text{A2} = \{000, 100, 010, 110, 101, 111\}$, another neural code on three neurons. The corresponding matrix of this code is

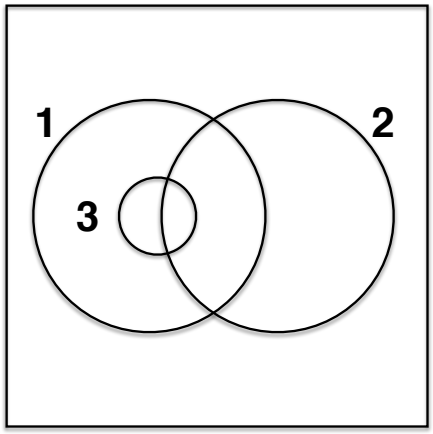


Figure 2.2: A place field diagram of the neural code A2.

$$A_{\mathcal{C}} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

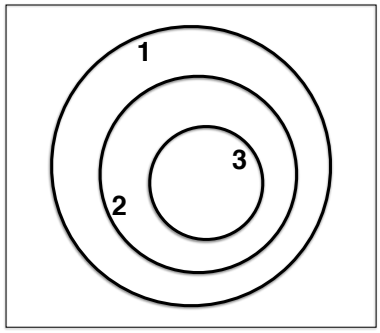
The toric ideal of this code I_{A2} is generated by the quadratics $p_{111} - p_{010}p_{101}$ and $p_{110} - p_{100}p_{010}$. This code is also convexly realizable in dimension two as a place field diagram drawn in Figure 2.2.

Note that I_{A2} has generators of degree 2. In Section 3.3 we will show how the degree of the generators will help us to decide if a code is k-inductively pierced.

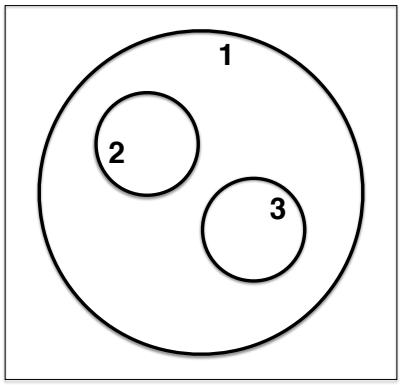
From the list of toric ideals of 3-neuron codes in Appendix A.2 we see that different neural codes have toric ideals which are generated by generators of varying degree. We ask:

Question 2.1.6. What do generators of the toric ideal tell us about place field diagrams of a neural code?

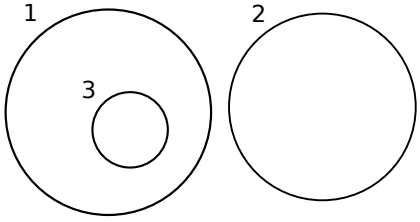
For many codes on three neurons the toric ideal is the zero ideal, meaning that $I_{\mathcal{C}} = \langle 0 \rangle$ (see Appendix A.2). For example, in Figure 2.3 we see that place field diagrams for several codes on three neurons with zero ideals have no intersecting boundary curves, i.e. if $i \neq j$, then U_i and U_j are either disjoint or nested. We have learned how to compute toric ideals of neural codes and are starting to notice that the generators of $I_{\mathcal{C}}$ seem to give some intersection information about place fields in a place field diagram of a code $\mathcal{C}$.



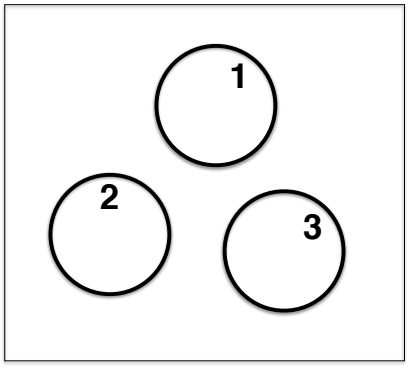
(a) A place field diagram of A12



(b) A place field diagram of B4



(c) A place field diagram of C2



(d) A place field diagram D1

Figure 2.3: Some three-neuron codes with $I_{\mathcal{C}} = \langle 0 \rangle$

For now, we develop hypergraphs of neural codes, which will help us prove statements about neural codes.

2.2 Hypergraphs of Neural Codes

Toric ideals have a nice combinatorial structure. We will exploit this structure using hypergraphs. A hypergraph is a generalization of a graph in which an edge can connect any number of vertices. Toric ideals associated to hypergraphs have been studied in [Vil01, GP12, PS14].

To visualize the information gathered from the toric ideal and to aid in proof, we introduce the notion of a hypergraph, which is a generalization of a graph in which an edge can connect any number of vertices. Toric ideals associated to hypergraphs have been studied in [Vil01, GP12, PS14].

Definition 2.2.1. [Bre13] A **hypergraph** $\mathcal{H}$ is a pair $\mathcal{H} = (V, E)$ where V is a set of elements called nodes or vertices, and E is a set of non-empty subsets of V called hyperedges or edges.

A code $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$ on n neurons can be visualized as a hypergraph $\mathcal{H}_\mathcal{C}$, with n vertices corresponding to the neurons and m hyperedges corresponding to the codewords. Each codeword $\mathbf{c} \in \mathcal{C}$ produces one edge containing all vertices v_i for $i \in \operatorname{supp}(\mathbf{c})$.

Definition 2.2.2 (Hypergraph of a neural code)**.** Given a code $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$, a neural code on n neurons, the hypergraph associated with $\mathcal{C}$ is $\mathcal{H}_\mathcal{C}$, where $V = \{1, \ldots, n\}$ and $E = \{\operatorname{supp} \mathbf{c} \mid \mathbf{c} \in \mathcal{C}\}$.

We say that an edge $\mathcal{E} \in E$ *covers* a vertex i if $i \in \mathcal{E}$. Now to exploit properties of hypergraphs, we define colorings of edges in a hypergraph, which we

will relate to the elements in the toric ideal of the code. We begin with some necessary definitions.

Definition 2.2.3 (Multiset)**.** A *multiset* is a generalization of a set that, unlike a set, allows multiple instances of the multiset's elements.

A set of edges $\mathcal{E}$ can be a multiset, so that we allow more than one copy of an edge in a set of edges. As such, we will also refer to a set of edges as an edge multiset. Now we discuss the coloring of edges in an edge multiset.

Definition 2.2.4 (Bicoloring of an edge multiset [GP12])**.** Let $\mathcal{E}$ be an edge multiset. We partition $\mathcal{E}$ into two disjoint subsets and assign one color to each of these subsets of edges, say blue and red. Then $\mathcal{E} = (R, B)$, where R is the set of red edges, and B is the set of blue edges. Such a coloring of $\mathcal{E}$ is called a *bicoloring* of $\mathcal{E}$.

Definition 2.2.5. [PS14] Let $\mathcal{E}$ be a multiset of edges in a hypergraph $\mathcal{H}$. We say that $\mathcal{E} = (R, B)$ is *balanced* with respect to a given bicoloring of $\mathcal{E}$ if for each vertex v covered by $\mathcal{E}$, the number of red edges containing v equals the number of blue edges containing v. If $\mathcal{E}$ is balanced, we call $\mathcal{E}$ a *balanced edge set* in $\mathcal{H}$.

A balanced edge set that is minimal in the sense that it does not contain any other nonempty balanced edge set is called a *primitive* balanced edge set.

Definition 2.2.6 (Primitive balanced set [GP12])**.** The balanced edge set $\mathcal{E}$ is primitive if there exists no other balanced edge set $\mathcal{E}' = (\mathcal{E}'_{\text{blue}}, \mathcal{E}'_{\text{red}})$ such that $\mathcal{E}'_{\text{blue}} \subsetneq \mathcal{E}_{\text{blue}}$ and $\mathcal{E}'_{\text{red}} \subsetneq \mathcal{E}_{\text{red}}$.

Each binomial in $I_{\mathcal{C}}$ corresponds to a balanced edge set in the hypergraph $\mathcal{H}_{\mathcal{C}}$ [PS14]. We say $f_{\mathcal{E}}$ arises from $\mathcal{E}$ if it can be written as $f_{\mathcal{E}} = \prod_{e \in \mathcal{E}_{\text{blue}}} t_e - \prod_{e' \in \mathcal{E}_{\text{red}}} t_{e'}$. Proposition 3.1 [GP12] tells us the toric ideal $I_{\mathcal{C}}$ is generated by binomials arising from primitive balanced edge sets in $\mathcal{H}_{\mathcal{C}}$.

We see how balanced edge sets in the hypergraph of a code encode the binomial generators in the toric ideal of the code.

As an example, consider the code $F1$=$\{000, 100, 010, 110\}$. The hypergraph $\mathcal{H}_{\text{F1}}$ is illustrated in Figure 2.4. By coloring the edges in the hypergraph, we see that there is a set of balanced edge sets: each of v_1 and v_2 is contained in a single blue edge and in a single red edge. The blue edge around vertex v_1 corresponds to the codeword 100, the blue edge around vertex 2 corresponds to the codeword 010, and the red edge around vertices v_1 and v_2 corresponds to the codeword 110. The binomial in the toric ideal of $F1$ can be read off the diagram as $p_{110} - p_{100}p_{010}$.

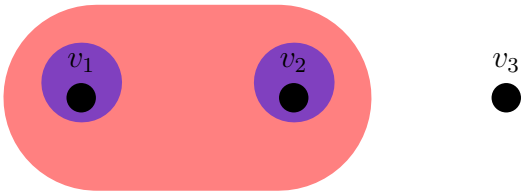


Figure 2.4: Hypergraph of neural code F1, $\mathcal{H}_{F1}$

As another example, consider the code $B1$=$\{000, 100, 010, 001, 110, 011\}$. We can visualize the information from the code using the following hypergraph, illustrated in Figure 2.5.

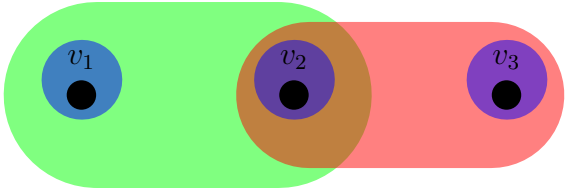


Figure 2.5: Hypergraph of neural code B1, $\mathcal{H}_{B1}$

By coloring the edges in the hypergraph, we see that there are at least two balanced edge sets: each of v_1 and v_2 is contained in a single blue edge and in a

single green edge. The blue edge around vertex v_1 corresponds to the codeword 100, the blue edge around vertex v_2 corresponds to the codeword 010, and the green edge around vertices v_1 and v_2 corresponds to the codeword 110. Additionally, each of v_2 and v_3 is contained in a single blue edge and in a single red edge. In this case, the red edge corresponds to the codeword 011. The generators of the toric ideal of $B1$ can be read off the diagram as $p_{110} - p_{100}p_{010}$ and $p_{011} - p_{010}p_{001}$. So, the generators of the toric ideal of a code are just the pairs of balanced edge sets in the hypergraph of the code.

CHAPTER 3

MOTIVATION FROM INFORMATION VISUALIZATION

Recall that our ultimate goal is to draw place field diagrams of neural codes. In this chapter, we introduce a class of Euler diagrams called k-inductively pierced diagrams. There exists a polynomial time algorithm for drawing $0, 1$, and $2-$inductively pierced diagrams developed by the authors in [SZHR11]. These inductively pierced diagrams are drawn algorithmically using circles, which are convex sets in dimension two. We study these inductively pierced diagrams and their corresponding abstract descriptions and show we can use the toric ideal of a neural code to determine if the code, and consequently its place field diagram, are inductively pierced. Once we have determined that a code is inductively pierced, we can apply the algorithm to draw a place field diagram of the code, and we will have answered our original question (Question 1.1.1).

3.1 Euler Diagrams and Abstract Descriptions

We will now discuss drawing Euler diagrams. A convexly realizable place field diagram in dimension two is an Euler diagram. One problem of interest in Information Visualization is how to visualize a collection of set-theoretic relationship data. Stapleton et al. give a partial answer to this question in [SZHR11]: we can visualize set-theoretic relationship data using an Euler diagram if the combinatorial code is k-inductively pierced.

Definition 3.1.1 (Euler diagram)**.** An *Euler diagram* d for n sets is a collection of n labeled simple, closed curves $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ in $\mathbb{R}^2$. Here, the λ_i are also called

curve labels. The interior of the region bounded by the curve λ_i is a subset U_i of $\mathbb{R}^2$, i.e. $U_i = \text{int } \lambda_i$. Denoting the boundary of U_i as ∂U_i, we have that $\lambda_i = \partial U_i$. Non-empty intersections of the sets $U_1, \ldots, U_n$ and their complements $\bar{U}_1, \ldots, \bar{U}_n$ form regions called *zones*.

Definition 3.1.2 (Well-formed)**.** An Euler diagram is said to be *well-formed* [SZHR11] if it satisfies the following conditions:

(1) Each curve label λ_i is used only once.

(2) All curves intersect generically (so curves that cross at exactly two points in the plane).

(3) A point in the plane is passed through at most two times by the curves in the diagram.

(4) Each zone is connected.

Since we will focus on well-formed Euler diagrams in this thesis and well-formedness requires each curve label to be used only once, we will use λ_i to denote both the ith curve and the label of the ith curve.

Now, we define abstract descriptions and establish a one-to-one correspondence between abstract descriptions and neural codes.

An Euler diagram can be abstracted by some collection of set data; this is termed the *abstract description* of the diagram:

Definition 3.1.3. An *abstract description* $D = (L, Z)$ of an Euler diagram d is an ordered pair specifying the curve labels L and the zones of d, $Z \subseteq \mathcal{P}(L)$, where $\mathcal{P}(L)$ is the power set of L. We will assume $\emptyset \in Z$ and if $\lambda \in L$, then there exists a $z \in Z$ such that $\lambda \in z$. We make these assumptions to avoid triviality. We will call an Euler diagram d with abstract description D a *realization* or *diagram* of D.

Now we show that a neural code $\mathcal{C}$ naturally defines an abstract description of the code $D_{\mathcal{C}}$. Let $\mathbf{c} \in \{0,1\}^n$ be a codeword and $z_{\mathbf{c}} = \{\operatorname{supp} \mathbf{c}\} \subseteq [n]$. We remind the reader that $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$. For a neural code $\mathcal{C}$, the set of curve labels is the set of all neurons, so $L_{\mathcal{C}} = [n]$, and the set of zones is the collection of the supports of the codewords in $\mathcal{C}$. Thus, the neural code $\mathcal{C}$ on n neurons corresponds naturally to the abstract description $D_{\mathcal{C}} = (L_{\mathcal{C}}, Z_{\mathcal{C}})$ where $Z_{\mathcal{C}} = \{z_{\mathbf{c}} \mid \mathbf{c} \in \mathcal{C}\}$. Drawing an Euler diagram d of the abstract description of the code $D_{\mathcal{C}}$ is equivalent to drawing a place field diagram of $\mathcal{C}$.

Remark. When working with neural codes we will require the all zeroes word to be in the code and that every neuron fires at least once, i.e. for all i from 1 to n, there exists a $\mathbf{c} \in \mathcal{C}$ such that $\mathbf{c}_i = 1$. The inclusion of the all zeroes codeword is to ensure that the empty zone is in Z, i.e., $\emptyset \in Z$. The requirement that all neurons fire at least once is to ensure that if $\lambda \in [n]$, then there exists some $z_{\mathbf{c}} \in Z_{\mathcal{C}}$ with $\lambda \in z_{\mathbf{c}}$. This is to stay consistent with the definition of an abstract description, and to maintain the one-to-one correspondence between abstract descriptions (as introduced in [SZHR11]) and neural codes.

Example 3.1.4. Consider, as an example, the code $\mathcal{C} = \mathrm{C2} = \{000, 100, 010, 101\}$. This is a three neuron code, so there are three curve labels, $\lambda_1 = 1$, $\lambda_2 = 2$, and $\lambda_1 = 3$, and $L_{\mathcal{C}} = \{1, 2, 3\}$. This code has four zones, corresponding to each of the four codewords of the code. In particular $Z_{\mathcal{C}} = \{z_{000}, z_{100}, z_{010}, z_{101}\}$. We can simplify the zone notation by indicating the curve labels which enclose each zone. As an example, $z_{101} = \{1, 3\}$. In particular, since $z_{\mathbf{c}} = \{\operatorname{supp} \mathbf{c}\}$, we can write the sets of zones as $Z_{\mathcal{C}} = \big\{\emptyset, \{1\}, \{2\}, \{1, 3\}\big\}$.

We now describe a couple of subsets of the power set $\mathcal{P}(L)$ that will be used in the definition of a k-piercing of an abstract description. Let $D = (L, Z)$ be an

abstract description. Given $\lambda \in L$, let $X_\lambda \subseteq Z$ be the set of all zones that contain λ:

$$X_\lambda = \{z \in Z \mid \lambda \in z\}.$$

We can think of X_λ as encoding the set of regions contained in U_λ. Given $z \in Z$ and a set of curve labels $\Lambda \subseteq L$ such that $z \cap \Lambda = \emptyset$, let the Λ-*cluster of* z, denoted by $Y_{z,\Lambda}$, be the set:

$$Y_{z,\Lambda} \;=\; \{z \cup \Lambda_i \mid \Lambda_i \subseteq \Lambda\}.$$

A cluster of a zone is a way to abstract the concept of topological adjacency in a diagram to a notion of relatedness in the abstract description [SZHR11].

Example 3.1.5 (An Euler diagram, its abstract description, and a cluster)**.** As an example, consider the following diagram pictured in Figure 3.1. Here the set of labels in the diagram d is $L = \{P, Q, R, S\}$ and the set of zones in d is $Z = \{\emptyset, \{P\}, \{Q\}, \{P, Q\}, \{P, R\}, \{P, S\}, \{P, Q, R\}, \{P, Q, S\}, \{P, R, S\}, \{P, Q, R, S\}\}$. Then the diagram d has abstract description:

$$D = (L, Z) = \Big(\{P, Q, R, S\}, \big\{\emptyset, \{P\}, \{Q\}, \{P, Q\}, \{P, R\}, \{P, S\}, \{P, Q, R\}, \{P, Q, S\}, \{P, R, S\}, \{P, Q, R, S\}\big\}\Big)$$

Consider the curve label $Q \in L$. We can consider X_Q, the set of all zones containing Q:

$$\begin{aligned} X_Q &= \{z \in Z \mid Q \in z\} \\ &= \big\{\{Q\}, \{P, Q\}, \{P, Q, R\}, \{P, Q, S\}, \{P, Q, R, S\}\big\}. \end{aligned}$$

Consider the zone $z = \{P\} \in Z$. Then $\Lambda = \{R, S\}$ is a set of labels disjoint from zone $z = \{P\}$, so we can consider $Y_{z,\Lambda}$, the Λ-cluster of z. In other words we

are considering the $\{R, S\}$-cluster of zone $\{P\}$, and we have that since $\{\Lambda_i \mid \Lambda_i \subseteq \{R, S\}\} = \big\{\emptyset, \{R\}, \{S\}, \{R, S\}\big\}$,

$$\begin{aligned} Y_{\{P\},\{R,S\}} &= \{\{P\} \cup \Lambda_i \mid \Lambda_i \subseteq \{R, S\}\} \\ &= \big\{\{P\} \cup \emptyset, \{P\} \cup \{R\}, \{P\} \cup \{S\}, \{P\} \cup \{R, S\}\big\} \\ &= \big\{\{P\}, \{P, R\}, \{P, S\}, \{P, R, S\}\big\}. \end{aligned}$$

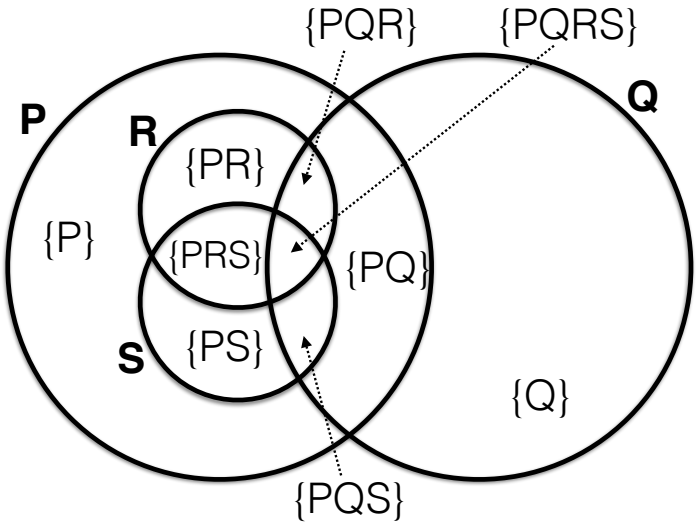


Figure 3.1: An Euler diagram of the abstract description $\Big(\{P, Q, R, S\}, \big\{\emptyset, \{P\}, \{Q\}, \{P, Q\}, \{P, R\}, \{P, S\}, \{P, Q, R\}, \{P, Q, S\}, \{P, R, S\}, \{P, Q, R, S\}\big\}\Big)$

3.2 k-piercings

Next, we define k-piercings, a notion from [SZHR11]. A k-piercing is a curve that intersects k existing curves.

Example 3.2.1 (k-piercings)**.** A piercing curve is a curve added to an existing diagram which adds a curve label and zones to the diagram. As an example consider the Euler diagram in Figure 3.2.

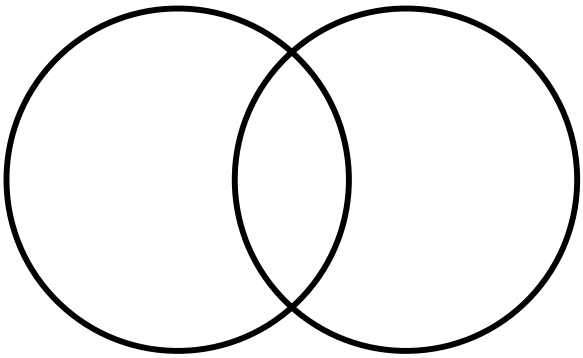


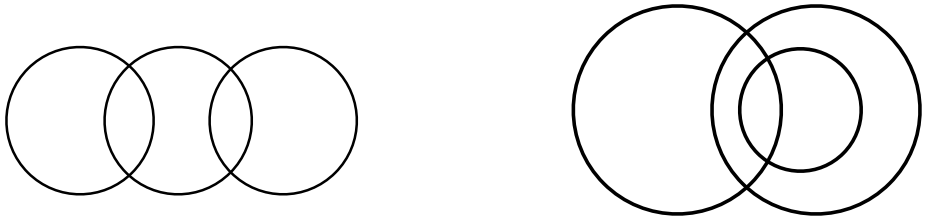
Figure 3.2: An Euler diagram d

We can add a piercing curve that does not intersect any of the existing curves, called a 0-piercing. Piercing the diagram in Figure 3.2 using 0-piercings produces one of the following diagrams as shown in Figure 3.3. In each case, one new zone is added to the diagram by the new curve.

(a) One possible 0-piercing of d (b) Another possible 0-piercing of d

Figure 3.3: 0-piercing of d from Figure 3.2

We could also add a piercing curve that intersects exactly one existing curve and adds two zones to the diagram d in Figure 3.2 as pictured in Figure 3.4.



(a) One possible 1-piercing of d (b) Another possible 1-piercing of d

Figure 3.4: 1-piercings of d from Figure 3.2

A piercing curve added to the diagram d that intersects both existing curves and leads to four new zones is a 2-piercing of d and is shown in Figure 3.5.

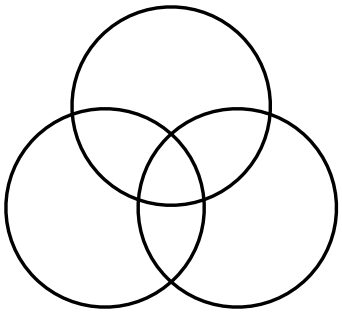


Figure 3.5: 2-piercing of d from Figure 3.2

Example 3.2.1 gives us an intuition on how we can pierce diagrams. We now explicitly define these piercings using abstract descriptions.

Definition 3.2.2 (0-piercings [SZHR11])**.** Let $D = (L, Z)$ be an abstract description. Then $\lambda \in L$ is a 0-*piercing* in D if there exists a *background zone* $z \in Z$ such that

(1) $\lambda \notin z$

(2) $X_\lambda = \{z \cup \{\lambda\}\} = Y_{z\cup\{\lambda\},\emptyset}$, and

(3) $Y_{z,\emptyset} = \{z \cup \{\lambda\}\} \subset Z$.

Example 3.2.3. As an example, consider the code $\mathcal{C} = \text{C2} = \{000, 100, 010, 101\}$. A place field diagram of $\mathcal{C}$ is pictured in Figure 3.6. Notice that curve 3 can be added to the diagram with curves 1 and 2 by a 0-piercing. In terms of the abstract

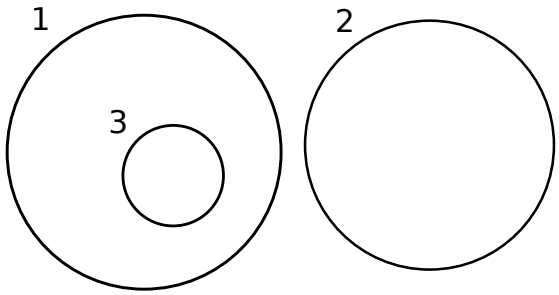


Figure 3.6: Euler diagram d of the abstract description of D_{C2}.

description, $D_{\mathcal{C}} = \Big\{\big\{1,2,3\big\}, \big\{\emptyset, \{1\}, \{2\}, \{1,3\}\big\}\Big\}$ and the curve label "3" is a 0-piercing (of the curve label "1") identified by the zone $z_{100} = \{1\}$.

In this example the set of all curve labels is $L_{\mathcal{C}} = \{1,2,3\}$ and the set of all zones in the diagram d is the set $Z_{\mathcal{C}} = \{z_{000}, z_{100}, z_{010}, z_{101}\} = \big\{\emptyset, \{1\}, \{2\}, \{1,3\}\big\}$. To show that "3" is a 0-piercing of "1", we show that the three conditions of a 0-piercing hold:

(1) $3 \notin z_{100} = \{1\}$, so the first condition holds.

(2) $X_3 = \{z \in Z_{\mathcal{C}} \mid 3 \in z\} = \{z_{101}\} = \big\{\{1,3\}\big\} = \big\{z_{100} \cup \{3\}\big\}$, i.e the set of all zones that contain 3 is exactly the $\{3\}$-cluster of z_{100}, so second condition holds.

(3) $z_{100} \cup \{3\} = \{1,3\} \in Z_{\mathcal{C}} = Y_{(z_{100}\cup\{3\}),\emptyset}$, so the third condition holds.

Hence, "3" is a 0-piercing of "1".

Note: In this case, "2" is also a 0-piercing in the abstract description $D_{\mathcal{C}}$, identified by the empty zone $z_{000} = \emptyset$.

Definition 3.2.4 (1-piercing [SZHR11])**.** Let $D = (L, Z)$ be an abstract description. Then λ_2 is a 1-piercing of λ_1 in D if there exists a zone $z \in Z$ such that

(1) $\lambda_1, \lambda_2 \notin z$

(2) $X_{\lambda_2} = Y_{z \cup \{\lambda_2\}, \{\lambda_1\}}$, and

(3) $Y_{z,\{\lambda_1\}} \subseteq Z$.

Example 3.2.5 (1-piercing)**.** As an example, consider the following diagram d pictured in Figure 3.7. The abstraction description D for this diagram has curve labels $L = \{1, 2, 3\}$ and zones $Z = \big\{\emptyset, \{1\}, \{2\}, \{1, 2\}, \{2, 3\}, \{1, 2, 3\}\big\}$. The curve

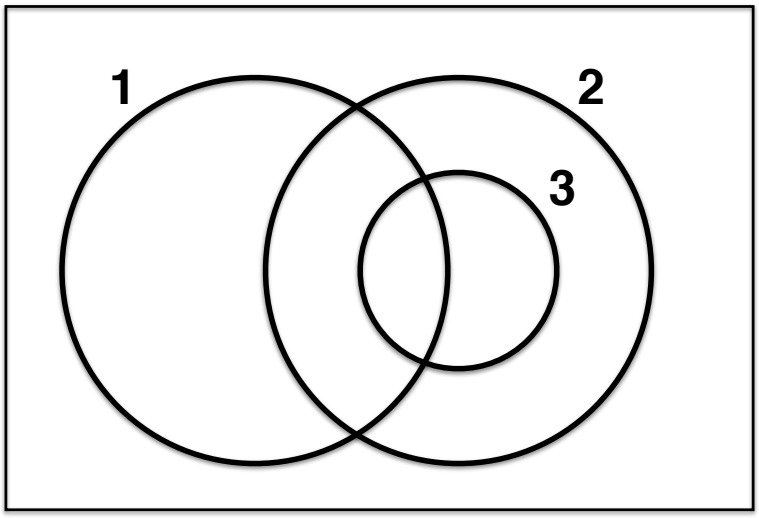


Figure 3.7: Example of a 1-piercing. The curve labeled 3 is a 1-piercing of the curve labeled 1 identified by the zone $\{2\}$

labeled 3 is a 1-piercing of the curve labeled 1 identified by the zone $\{2\} \in Z$ since

(1) $3,\ 1 \notin \{2\}$,

(2) $X_3 = \big\{z \in Z \mid 3 \in z\big\} = \big\{\{2,3\},\{1,2,3\}\big\}$ and
$Y_{\{2\}\cup\{3\},\{1\}} = \big\{(\{2\}\cup\{3\})\cup\Lambda_i \mid \Lambda_i \subseteq \{1\}\big\} =$
$\big\{(\{2\}\cup\{3\})\cup\emptyset, (\{2\}\cup\{3\})\cup\{1\}\big\} = \big\{\{2,3\},\{1,2,3\}\big\}$, so
$X_3 = Y_{\{2\}\cup\{3\},\{1\}}$, and

(3) $Y_{\{2\},\{1\}} = \big\{\{2\}\cup\Lambda_i \mid \Lambda_i \subseteq \{1\}\big\} = \big\{\{2\}\cup\emptyset, \{2\}\cup\{1\}\big\} = \big\{\{2\},\{1,2\}\big\} \subseteq Z$,
because $\{2\},\{1,2\} \in Z$.

Definition 3.2.6 (2-piercing [SZHR11])**.** Let $D = (L, Z)$ be an abstract description. Then λ_3 is a 2-piercing of λ_1 and λ_2 in D if there exists a zone $z \in Z$ such that

(1) $\lambda_1, \lambda_2, \lambda_3 \notin z$,

(2) $X_{\lambda_3} = Y_{z\cup\{\lambda_3\},\{\lambda_1,\lambda_2\}}$, and

(3) $Y_{z,\{\lambda_1,\lambda_2\}} \subseteq Z$.

In the diagram in Figure 3.8, the curve labeled 4 is a 2-piercing of the curves labeled 1 and 2 identified by the zone $\{3\}$. We check the three conditions of a 2-piercing:

(1) $1, 2, 4 \notin \{3\}$;

(2) $X_4 = \{\{3,4\},\{1,3,4\},\{2,3,4\},\{1,2,3,4\}\}$ and

$Y_{\{3\}\cup\{4\},\{1,2\}} = \big\{\,(\{3\}\cup\{4\})\cup\Lambda_i \mid \Lambda_i \subseteq \{1,2\}\big\}$,
$= \big\{\,(\{3\}\cup\{4\})\cup\emptyset, (\{3\}\cup\{4\})\cup\{1\}, (\{3\}\cup\{4\})\cup\{2\}, (\{3\}\cup\{4\})\cup\{1,2\}\big\}$
$= \big\{\{3,4\},\{1,3,4\},\{2,3,4\},\{1,2,3,4\}\big\}$, so

$X_4 = Y_{\{3\}\cup\{4\},\{1,2\}}$, and

Example 3.2.7 (2-piercing)**.**

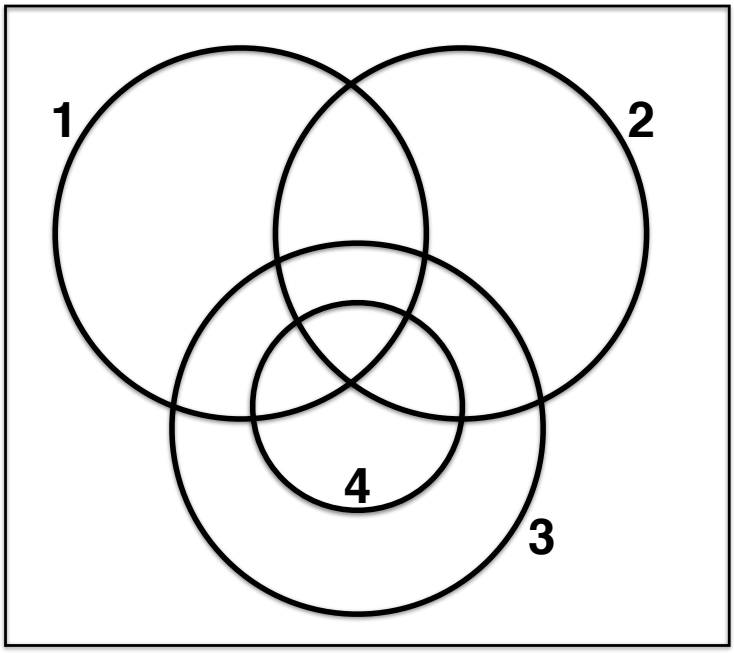


Figure 3.8: A diagram with a 2-piercing

(3) $Y_{\{3\},\{1,2\}} = \big\{\{3\} \cup \Lambda_i \mid \Lambda_i \subseteq \{1,2\}\big\}$,

$Y_{\{3\},\{1,2\}} = \big\{\{3\} \cup \emptyset, \{3\} \cup \{1\}, \{3\} \cup \{2\}, \{3\} \cup \{1,2\}\big\}$, so

$Y_{\{3\},\{1,2\}} = \big\{\{3\}, \{1,3\}, \{2,3\}, \{1,2,3\}\big\}$, and thus, $Y_{\{3\},\{1,2\}} \subseteq Z$.

Hence, the three conditions of a 2-piercing hold, that is, the curve labeled 4 is a 2-piercing of the curves labeled 1 and 2.

Example 3.2.8 (0, 1, and 2-piercings in a diagram)**.** Consider the diagram in Figure 3.9. In this diagram we can identify a 0-piercing, a 1-piercing, and a 2-piercing.

In particular, 6 is a 0-piercing, 7 is a 1-piercing of 3, and 1 is a 2-piercing of 2 and 5. The curves labeled 2, 3, 4, and 5 are not 0, 1, or 2-piercings in the big picture, but if we remove the curves labeled 1, 6, and 7, then each of the curves labeled 3 and 5 are 2-piercings of the curves labeled 2 and 4.

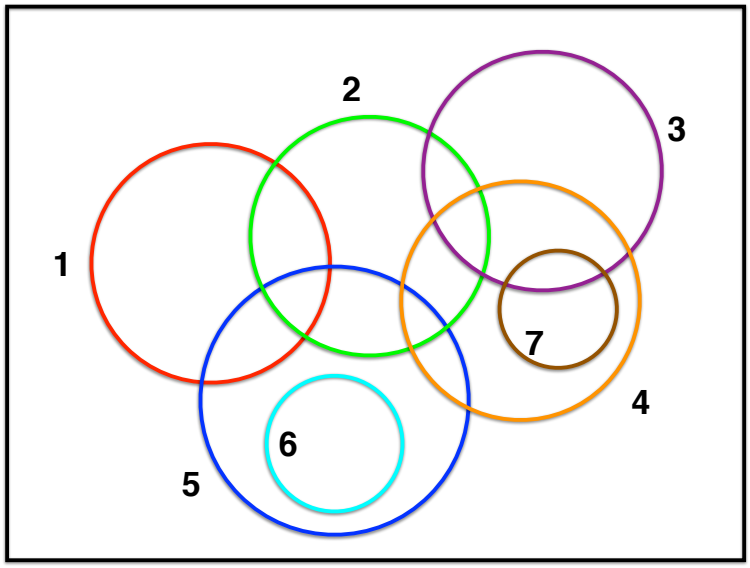


Figure 3.9: A diagram with 0-,1-, and 2-piercings

The fact that each curve in this example is a 0-, 1-, or 2-piercing (after removing some curve(s), if necessary) results in the realizability of this abstract description; the abstract description and this diagram are called inductively pierced, which will be defined in Section 3.3.

We can generalize the concept of a k-piercing for any non-negative integer k as follows:

Definition 3.2.9. [SZHR11] Let $D = (L, Z)$ be an abstract description. Let $\Lambda = \{\lambda_1, ..., \lambda_k\} \subseteq L$ be distinct curve labels. Then $\lambda_{k+1} \in L$ is a *k-piercing* of Λ in D if there exists a background zone $z \in Z$ such that

(1) $\lambda_i \notin z$ for each $i \leq k+1$

(2) $X_{\lambda_{k+1}} = Y_{z \cup \{\lambda_{k+1}\}, \Lambda}$, and

(3) $Y_{z,\Lambda} \subseteq Z$.

When the above three conditions hold, the background zone z is said to *identify* λ_{k+1} as a k-piercing of $\lambda_1, ..., \lambda_k$.

Notice for a 0-piercing, $\Lambda = \emptyset$.

In terms of drawings, we can think of a k-piercing as a curve that pierces k other curves and splits 2^k zones. These 2^k zones appear in the abstract description in the following way.

Lemma 3.2.10. *Let $D = (L, Z)$ be an abstract description. Let $\Lambda = \{\lambda_1, ..., \lambda_k\} \subseteq L$ be distinct curve labels. If $\lambda_{k+1} \in L$ is a* k-piercing *of Λ in D, then there exist exactly 2^k elements of Z that contain λ_{k+1}, i.e., $|X_{\lambda_{k+1}}| = 2^k$, corresponding to subsets of Λ.*

Proof. The statement follows from the second condition in the definition of a k-piercing. Assuming the hypotheses, by definition,

$$X_{\lambda_{k+1}} = Y_{z \cup \{\lambda_{k+1}\}, \Lambda} = \{(z \cup \{\lambda_{k+1}\}) \cup \Lambda_i \mid \Lambda_i \subseteq \Lambda\}.$$

Now, $\{\Lambda_i \mid \Lambda_i \subseteq \Lambda\} = \mathcal{P}(\Lambda)$, which means that in the cluster, the zone $z \cup \{\lambda_{k+1}\}$ is being unioned with all of the subsets of $\Lambda = \{\lambda_1, \ldots, \lambda_k\}$. Well, $\{\lambda_1, \ldots, \lambda_k\}$ is a set of cardinality k, so it has 2^k subsets. Note that by the definition of a k-piercing, $\lambda_1, \ldots, \lambda_{k+1} \notin z$, and by the underlying assumption of well-formedness, each of the λ_i are distinct. This ensures that each of the subsets are distinct, so we do in fact have 2^k of these subsets. Then $|X_{\lambda_{k+1}}| = 2^k$, as desired. □

3.3 Inductively pierced

In [SZHR11], the authors show that if D is an inductively pierced abstract description, then there exists an inductively pierced drawing d of D, which can be drawn in polynomial time. Our goal is to identify what it means for a code $\mathcal{C}$ to be

k-inductively pierced using algebra. We will explore these conditions further in Chapter 4.

In order to define what it means for an abstract description to be k-inductively pierced, we discuss the removal of piercing curves in the context of abstract descriptions.

Definition 3.3.1 (Removal of a curve)**.** Given an abstract description $D = (L, Z)$ with $\lambda \in L$, we define

$$D - \lambda = (L \setminus \{\lambda\}, Z - \lambda),$$

where $Z - \lambda \stackrel{\text{def}}{=} \{z \setminus \{\lambda\} \mid z \in Z\}$.

When $\mathcal{C}$ is a neural code, the analogue of the removal of a piercing curve from $D_{\mathcal{C}}$ is the deletion of a neuron from the code. For a code $\mathcal{C}$ on n neurons, we define

$$\mathcal{C} - \lambda = \{(c_1, \ldots, c_{\lambda-1}, \hat{c}_\lambda, c_{\lambda+1}, \ldots, c_m) \mid (c_1, \ldots, c_m) \in \mathcal{C}\},$$

where $\hat{c}_\lambda$ indicates the removal of the λth component of each codeword c_λ for all $\mathbf{c} \in \mathcal{C}$. Recall that for $\mathcal{C}$ a code on n neurons, $|\mathcal{C}| = n$. Then, $|\mathcal{C} - \lambda| = n - 1$, that is $\mathcal{C} - \lambda$ is a code on $n - 1$ neurons.

For now, we define what it means for an abstract description to be k-inductively pierced and what it means for a code to be inductively pierced.

Definition 3.3.2. An abstract description $D = (L, Z)$ is *k-inductively pierced* if D has a 0, 1, ..., or k-piercing λ and $D - \lambda$ is k-inductively pierced.

Definition 3.3.3. A code $\mathcal{C}$ is *inductively pierced* if D has a 0, 1, or 2-piercing λ and $\mathcal{C} - \lambda$ is 2-inductively pierced.

In [SZHR11], the authors explore inductively pierced diagrams. These diagrams have abstractions that are 2-inductively pierced. Thus, we will be interested in 0, 1, and 2-inductively pierced descriptions.

Definition 3.3.4 (0-inductively pierced code)**.** A code $\mathcal{C}$ is 0-inductively pierced when the abstraction description of the code $D_{\mathcal{C}}$ has a 0-piercing λ such that $D_{\mathcal{C}} - \lambda$ is 0-inductively pierced.

By the definition of a 0-piercing from Definition 3.2.2, $D_{\mathcal{C}}$ has a 0-piercing λ means that there exists a codeword z such that λ was placed inside zone z, so $z_\lambda \neq 1$ and that if $c_\lambda = 1$ then $\text{supp}(c) = \text{supp}(z) \cup \{\lambda\}$. We can thus characterize a 0-piercing as adding in exactly one codeword to a code in a very particular form.

Lemma 3.3.5. *If λ is a 0-piercing of $\mathcal{C}$, then $\mathcal{C}$ can be obtained from $\mathcal{C} - \lambda$ by adding a neuron which is always 0, and then adding a codeword whose support is equal to the support of one of the codewords of $\mathcal{C}$ along with λ.*

Proof. Suppose that λ is a 0-piercing of $\mathcal{C}$, identified by zone $z_{\mathbf{c}}$. Then

(1) $\lambda \notin z_{\mathbf{c}}$,

(2) $X_\lambda = \{z_{\mathbf{c}} \cup \{\lambda\}\}$, and

(3) $z_{\mathbf{c}} \cup \{\lambda\} \in Z_{\mathcal{C}}$.

This information directly translates into the language of neural codes as such:

(1) $\lambda \notin \text{supp}(z_{\mathbf{c}})$,

(2) $\{\mathbf{c} \in \mathcal{C} \mid \mathbf{c}_\lambda = 1\} = \{v\}$, where $\text{supp}(v) = \text{supp}(\mathbf{c}) \cup \lambda$, and

(3) $z_{\mathbf{c}} \in \mathcal{C}$.

So we have that in $\mathcal{C}$, all codewords except $z_{\mathbf{c}}$ have $c_\lambda = 0$ and that there is also exactly one codeword whose support is identical to $z_{\mathbf{c}}$ except at λ. □

By Lemma 3.3.5 we get that a 0-piercing is determined by the codeword gained in the code from the one zone the piercing curve adds to the diagram.

Figure 3.10 shows several examples of 2-inductively pierced diagrams. Notice that although the conditions for inductively pierced diagrams are fairly simple, we can draw many different diagrams using this seemingly straightforward condition.

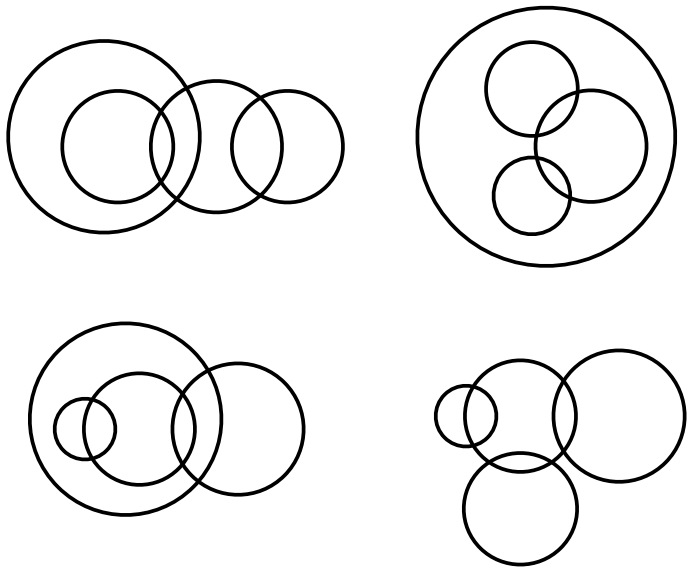


Figure 3.10: Some inductively pierced codes on four neurons

By the definition of 0-pierced, we see that in an abstract description, the curve $\lambda \in L$ is a 0-piercing for an empty set of curve labels, $\Lambda = \emptyset$ for $\Lambda \subset Z$. In other words, well-formed diagrams drawn with circles in which no curves intersect correspond to 0-inductively pierced descriptions.

Proposition 3.3.6. *An abstract description D is 0-inductively pierced if and only if all curves in any well-formed realization of D do not cross.*

Proof. ($\Rightarrow$) Suppose D is 0-inductively pierced. Assume for the sake of contradiction that there exists a well-formed realization d of D such that there exist

two curves, λ_1 and λ_2, that cross.

Since D is 0-inductively pierced, we can remove 0-piercings until λ_1 or λ_2 is a 0-piercing of the remaining curves, thus, without loss of generality, let us assume λ_1 is a 0-piercing of D. Now if λ_1 is a 0-piercing of D, then condition (2) of Definition 3.2.9 implies that there exists a zone z, with corresponding codeword z, such that the curve λ_1 is contained entirely in $\cap_{i \in Z} U_i$. Thus, we can zoom in on this crossing as illustrated in Figure 4.1.

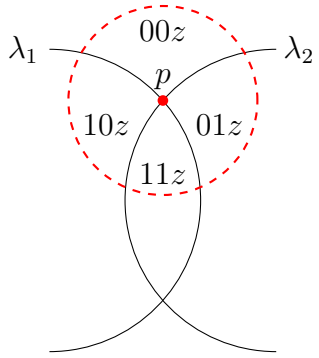


Figure 3.11: A closeup of a crossing of curves λ_1 and λ_2.

From Figure 4.1, we see that $|X_{\lambda_1}| \geq 2$. But since λ_1 is a 0-piercing, by Lemma 3.2.10, there exists exactly $2^0 = 1$ element of Z that contains λ_1, so $|X_{\lambda_1}| = 1$, a contradiction.

($\Leftarrow$) Let D be a well-formed abstract description. Suppose for any well-formed realization d of D, no two curves intersect. We will proceed by induction on n, the number of curves. The statement holds for $n = 1$, since the only code on one neuron is 0-inductively pierced. Now suppose the statement holds for $n \leq r$, and let $n = r + 1$. Since none of the curves $\lambda_1, \ldots, \lambda_n$ intersect, every pair of fields, U_i and U_j, in d are disjoint or nested. Pick one nested sequence of fields and select the minimal field with respect to set inclusion, that is, select a field U_k such that for all

$1 \leq i \leq n$ with $i \neq k$ either $U_i \cap U_k = \emptyset$ or $U_k \subset U_i$. Then λ_k is a 0-piercing of D, and by the induction hypothesis, $D - \lambda_k$ is 0-inductively pierced. Therefore, D is 0-inductively pierced. □

We see that 0-piercings correspond to curve labels in a diagram that do not intersect other curve labels. We relate this notion back to the information gained from toric ideals: if curve labels are nested or disjoint, these curve labels will not add any generators to the toric ideal. Building off of this observation, in the next chapter (Chapter 4) we show that we can relate k-piercings to the degrees of the generators in the toric ideal.

CHAPTER 4

MAIN RESULTS

4.1 Conditions for 0-piercings

Theorem 4.1.1. *Let $\mathcal{C}$ be a code on n neurons such that each neuron fires at least once, i.e. $\cup_{z\in\mathcal{C}}\ \mathrm{supp}(z) = [n]$. Let $\mathcal{C}$ be well-formed. Then, the toric ideal $I_\mathcal{C} = \langle 0\rangle$ if and only if $\mathcal{C}$ is 0-inductively pierced.*

In order to prove Theorem 4.1.1, we prove the following lemma which allows us conclude that an intersection of curves implies the existence of a nontrivial element in the toric ideal.

Lemma 4.1.2. *Let $\mathcal{C} \subseteq \{0,1\}^n$ be a neural code with abstract description $D_\mathcal{C}$. If a well-formed diagram d of $D_\mathcal{C}$ contains two curves that intersect, then the toric ideal $I_\mathcal{C}$ is nonzero.*

Proof. Assume a well-formed diagram has a crossing. Specifically, let d be a well-formed diagram of $D_\mathcal{C}$ such that two curves λ_1 and λ_2 intersect. Let q be an intersection point of λ_1 and λ_2. Since d is well-formed, there exists an open ball around p, that is contained entirely in a single zone z of $d - \lambda_1 - \lambda_2$. Thus, the following codewords must be in $\mathcal{C}$: $10z, 01z, 00z, 11z$ (as seen in Figure 4.1).

We know that for each of the codewords, when we project onto the last $n-2$ components, the vectors are identical and are exactly z, since the other relationships

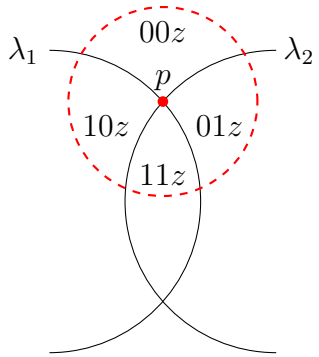


Figure 4.1: A closeup of a crossing of curves λ_1 and λ_2.

in the diagram are preserved. Then the matrix of the codewords in $\mathcal{C}$ looks like

$$\begin{pmatrix} 0 & 1 & 0 & 1 & \dots \\ 0 & 0 & 1 & 1 & \dots \\ z^T & z^T & z^T & z^T & \ddots \end{pmatrix},$$

and we have

$$\phi_{\mathcal{C}}(p_{00z}) = x^z \qquad \phi_{\mathcal{C}}(p_{10z}) = x_1 x^z$$
$$\phi_{\mathcal{C}}(p_{01z}) = x_2 x^z \qquad \phi_{\mathcal{C}}(p_{11z}) = x_1 x_2 x^z,$$

where $x^z \overset{\text{def}}{=} \prod_{i \in supp(z)} x_i$. Therefore $p_{11z}p_{00z} - p_{10z}p_{01z}$ is a nonzero element in the toric ideal $I_{\mathcal{C}}$. Indeed,

$$\phi_{\mathcal{C}}(p_{11z}p_{00z} - p_{10z}p_{01z}) = (x^z \cdot x_1 x_2 x^z - (x_1 x^z \cdot x_2 x^z) = 0.$$

Thus, $I_{\mathcal{C}}$ is nonzero as desired. □

Now we turn our attention to proving Theorem 4.1.1, which states that zero ideals are characteristic of codes that are 0-inductively pierced.

Proof of Theorem 4.1.1. Suppose the hypotheses, so $\mathcal{C}$ is a well-formed code on n neurons such that each neuron fires at least once.

($\Rightarrow$) Suppose $I_{\mathcal{C}} = \langle 0 \rangle$ and assume for the sake of contradiction that $D_{\mathcal{C}}$ is not 0-inductively pierced. Then Proposition 3.3.6 implies that it is not the case that all curves in any well-formed realization of $D_{\mathcal{C}}$ do not cross. So, in some realization d of the neural code there are at least two intersecting curves. Well then we have that a well-formed diagram d of $D_{\mathcal{C}}$ contains two curves that intersect, so by Lemma 4.1.2, the toric ideal $I_{\mathcal{C}}$ is nonzero. But this contradicts the original hypothesis that $I_{\mathcal{C}} = \langle 0 \rangle$. Thus the assumption that $D_{\mathcal{C}}$ is not inductively 0-pierced code given the hypotheses was false: hence the forward implication is true.

($\Leftarrow$) Suppose that $D_{\mathcal{C}}$ is 0-inductively pierced. We will prove by induction on the number of neurons n that $I_{\mathcal{C}} = \langle 0 \rangle$. For the base case, take $n = 1$, so $\mathcal{C} = \{0, 1\}$ is a one neuron code. Then since $I_{\mathcal{C}}$ is a binomial ideal, there is no possible way to get a non-trivial binomial generator in $I_{\mathcal{C}}$. More specifically, the map $\phi_{\mathcal{C}} : \mathbb{K}[p_1] \to \mathbb{K}[x_1]$ is one-to-one, hence the kernel is trivial. Hence $I_{\mathcal{C}} = \langle 0 \rangle$ as desired for the base case.

Now for the induction hypothesis, suppose that $I_{\mathcal{C}} = \langle 0 \rangle$ for any code $\mathcal{C}$ on $n = r$ neurons such that $D_{\mathcal{C}}$ is 0-inductively pierced. We need to show that this implies that a code on $\mathcal{C}$ on $n = r + 1$ neurons has $I_{\mathcal{C}} = \langle 0 \rangle$. By our induction hypothesis, $\mathcal{C}$ is 0-inductively pierced, so by Definition 3.3.3, $\mathcal{C}$ has a 0-piercing λ and $\mathcal{C} - \lambda$ is 0-inductively pierced. Recall that $\mathcal{C} - \lambda$ is a code on $|\mathcal{C}| - 1$ neurons, so here, $\mathcal{C} - \lambda$ is a code on $(r + 1) - 1 = r$ neurons. Then by the induction hypothesis, $I_{\mathcal{C}-\lambda} = \langle 0 \rangle$. We will now use the hypergraphs of $\mathcal{C}$ and $\mathcal{C} - \lambda$, $H_{\mathcal{C}}$ and $H_{\mathcal{C}-\lambda}$ respectively, to show that $I_{\mathcal{C}} = \langle 0 \rangle$.

Recall from Lemma 3.3.5 that a 0-piercing entails adding one very specific codeword. Then we obtain the hypergraph $H_{\mathcal{C}}$ from the hypergraph $H_{\mathcal{C}-\lambda}$ by adding a vertex λ and a single hyperedge $z \cup \{\lambda\}$ containing λ. Now recall from Section 2.1 that balanced edge sets correspond to binomials in the toric ideal. Adding only one

hyperedge that contains λ will not add any balanced edge sets to $H_{\mathcal{C}}$ that are not already in $H_{\mathcal{C}-\lambda}$; λ is in only one codeword, so it cannot be contained in any balanced edge sets. Hence the balanced edge sets of $H_{\mathcal{C}}$ are exactly the balanced edge sets of $H_{\mathcal{C}-\lambda}$. But $I_{\mathcal{C}-\lambda} = \langle 0 \rangle$, so there are no non-trivial binomials in the toric ideal of $\mathcal{C} - \lambda$. Thus there are no balanced edge sets in $H_{\mathcal{C}-\lambda}$, and consequently no balanced edge sets in $H_{\mathcal{C}}$. Therefore $I_{\mathcal{C}} = \langle 0 \rangle$, as desired. □

4.2 Conditions for 1-piercings

We have seen that the toric ideal of a code can determine whether the code is 0-inductively pierced. Now we investigate the relationship between the toric ideal of a code and 1-inductively pierced codes. We start by considering examples of some 1-inductively pierced codes. Recall from Definition 3.3.3 that this means that there exists a 0-piercing or 1-piercing λ such that $\mathcal{C} - \lambda$ is 1-inductively pierced.

Example 4.2.1. Consider again the neural code A2= $\{000, 100, 010, 110, 101, 111\}$. A place field diagram of A2 is pictured in Figure 4.2. The toric ideal of A2, $I_{\text{A2}} = \langle p_{111} - p_{010}p_{101},\ p_{110} - p_{100}p_{010} \rangle$, is generated by quadratics.

While our goal is to understand the realization of a code by understanding its toric ideal, we note that Lemma 4.1.2 and its proof allow us to understand some things quickly about a toric ideal of a code simply by noticing motifs in place field diagrams of 1-inductively pierced codes.

Since a curve is a 1-piercing is a curve that intersects one other curve in exactly 2 points, we conclude that there is a quadratic binomial in $I_{\mathcal{C}}$ for every two fields that intersect transversally as in Figure 4.3. We can generalize Lemma 4.1.2: the toric ideal of the code associated to a chain of n fields contains $n - 1$ pairwise intersections as illustrated in Figure 4.4 contains $n - 1$ quadratic binomials.

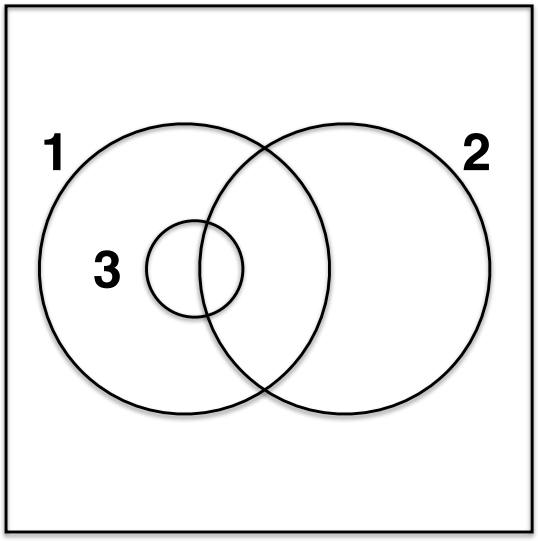


Figure 4.2: A place field diagram of the 1-inductively pierced code A2

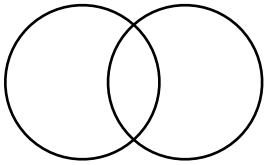


Figure 4.3: Two fields that intersect transversally lead to a quadratic binomial in the toric ideal.

Theorem 4.2.2. *Let $\mathcal{C}$ be well-formed. If $\mathcal{C}$ is* 1*-inductively pierced then the toric ideal $I_{\mathcal{C}}$ is generated by quadratics or $I_{\mathcal{C}} = \langle 0 \rangle$.*

The proof of Theorem 4.2.2 requires a deeper discussion of hypergraphs and is proved in detail in [GOY].

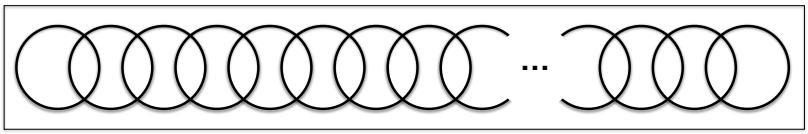


Figure 4.4: A chain of n fields intersecting transversally.

4.3 Observations on 2-piercings

The code A1 = {000, 100, 010, 001, 110, 101, 011, 111} has toric ideal $\langle p_{111} - p_{100}p_{010}p_{001},\ p_{110} - p_{100}p_{010},\ p_{101} - p_{100}p_{001},\ p_{011} - p_{010}p_{001}\rangle$. This code has a cubic generator in its toric ideal. We notice that the place field diagram of A1 shows a triple intersection of place fields, which corresponds to a 2-piercing.

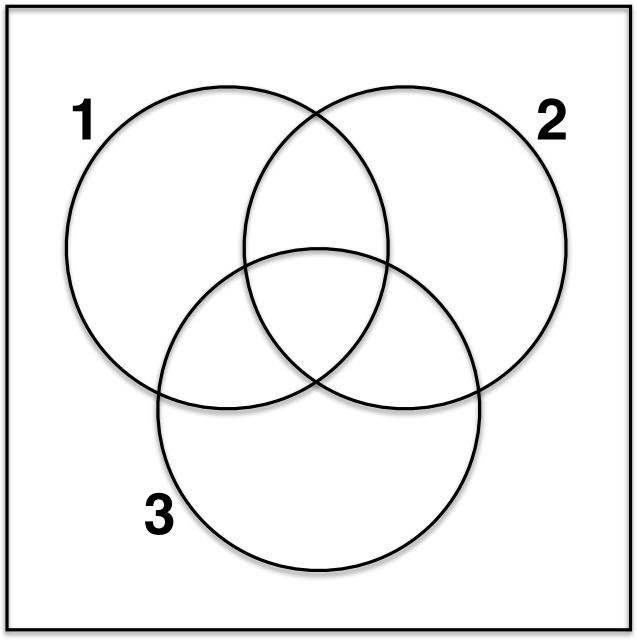


Figure 4.5: A place field diagram for A1

The cubic generator in the toric ideal is pointing out this triple intersection of place fields. The cubic generator is a signature of a 2-piercing in the diagram, and

in fact we can show that this is true in general for all well-formed neural codes as follows.

Proposition 4.3.1. *Let $\mathcal{C}$ be a well-formed neural code on n neurons. If there is a 2-piercing in a diagram, then the toric ideal $I_{\mathcal{C}}$ contains a cubic binomial, in particular, a binomial of the form $p_{111z} - p_{100z}p_{010z}p_{001z}$ or $p_{111z}p_{000\ldots0}p_{000\ldots0} - p_{100z}p_{010z}p_{001z}$.*

Proof. Let $\mathcal{C}$ be a well-formed neural code on n neurons with a 2-piercing. Since we have a 2-piercing, by Definition 3.2.9, the abstract description has at least three curve labels. Without loss of generality, relabel the curve labels so that 1 is a 2-piercing of {2,3}. Then this piercing is identified by a zone, z, such that this 2-piercing is contained in this zone z. Let d be a well-formed diagram of $D_{\mathcal{C}}$ such that λ_1 is a 2-piercing of λ_2 and λ_3. We zoom in on this 2-piercing and we have the following arrangement in a place field diagram d of $\mathcal{C}$ as seen in Figure 4.6

Let p and q be the intersection points of λ_1 and λ_2 (these points exist since λ_3 is a by definition, λ_3 is a 2-piercing of $\{\lambda_1, \lambda_2\}$ implies that λ_1 and λ_2 intersect, and since $\mathcal{C}$ is well-formed, all curves intersect generically). Thus, the following codewords must be in $\mathcal{C}$: $000z,\ 100z,\ 010z,\ 001z,\ 111z$. We know that for each of the codewords, when we project onto the last $n-3$ components, the vectors are identical and are exactly z, since the other relationships in the diagram are preserved. Then the matrix of the codewords in $\mathcal{C}$ looks like

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & \ldots \\ 0 & 0 & 1 & 0 & 1 & \ldots \\ 0 & 0 & 0 & 1 & 1 & \ldots \\ z^T & z^T & z^T & z^T & z^T & \ddots \end{pmatrix},$$

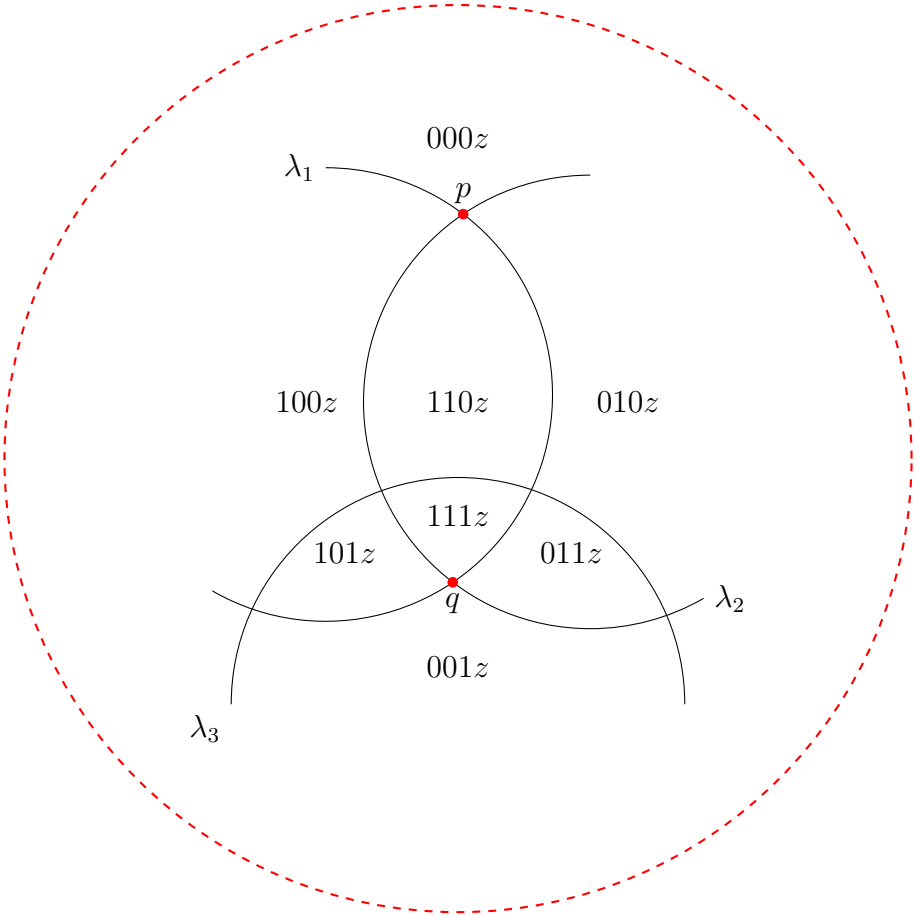


Figure 4.6: A closeup on a 2-piercing.

and we have

$$\phi_{\mathcal{C}}(p_{100z}) = x_1x^z \qquad \phi_{\mathcal{C}}(p_{010z}) = x_2x^z$$

$$\phi_{\mathcal{C}}(p_{001z}) = x_3x^z \qquad \phi_{\mathcal{C}}(p_{111z}) = x_1x_2x_3x^z.$$

Therefore, in the case where z is the empty zone, $p_{111z} - p_{100z}p_{010z}p_{001z}$ is a generator in the toric ideal $I_{\mathcal{C}}$. Indeed,

$$\phi_{\mathcal{C}}(p_{111z} - p_{100z}p_{010z}p_{001z}) = (x_1x_2x_3) - (x_1 \cdot x_2 \cdot x_3) = 0.$$

On the other hand, if z is not the empty zone, then $\prod_{i \in \mathrm{supp}(z)} x_i$ is not 1. In this

case, $p_{111z}p_{000z}^2 - p_{100z}p_{010z}p_{001z}$ is a generator in the toric ideal $I_{\mathcal{C}}$. Indeed,

$$\phi_{\mathcal{C}}(p_{111z}p_{000z}^2 - p_{100z}p_{010z}p_{001z}) = \left(x_1x_2x_3 \prod_{i\in\text{supp}(z)} x_i \cdot \left(\prod_{i\in\text{supp}(z)} x_i \right)^2 \right) - \left(x_1 \prod_{i\in\text{supp}(z)} x_i \cdot x_2 \prod_{i\in\text{supp}(z)} x_i \cdot x_3 \prod_{i\in\text{supp}(z)} x_i \right) = 0.$$

□

4.4 Modifying Term Order Using Gröbner Bases

The code A1 is inductively pierced, but its toric ideal contains the cubic $p_{111} - p_{100}p_{010}p_{001}$; in fact, this cubic is in the generating set of I_{A1} that we gave in Example 2.1.4. Recall that from Proposition 4.3.1 we can expect to see a cubic in the toric ideal of a code when there is a a 2-piercing in the diagram. Notice that in the A1 example, the cubic $p_{111} - p_{100}p_{010}p_{001}$ can be written in terms of quadratics. In particular,

$$p_{111} - p_{100}p_{010}p_{001} \;=\; (p_{111} - p_{110}p_{001}) + \; p_{001}(p_{110} - p_{100}p_{010}).$$

Note that each of the quadratics on the right hand side of the equation are in I_{A1} since $\phi_{\text{A1}}(p_{111} - p_{110}p_{001}) = x_1x_2x_3 - x_2x_3 \cdot x_1 = 0$ and $\phi_{\text{A1}}(p_{110} - p_{100}p_{010}) = x_1x_2 - x_1 \cdot x_2 = 0$. Thus, we can give a generating set of the toric ideal of A1 that is generated only by quadratics:

$$I_{\text{A1}} = \langle p_{110} - p_{100}p_{010},\; p_{101} - p_{100}p_{001},\; p_{011} - p_{010}p_{001},\; p_{111} - p_{110}p_{001} \rangle.$$

Since cubics are signatures of 2-piercings, we would like to know if there are certain term orders on the monomials that can identify this generator as an element of the reduced Gröbner basis. Informally, a set $\{g_1, \ldots, g_t\} \subset I$ is a Gröbner basis of I if and only if the leading term of any element of I is divisible by the leading term

of some g_i [CLO07]. The hope here is that we can use a specific term order and the Gröbner basis of the toric ideal to prove the converse of Theorem 4.2.2. Formally, we conjecture the following:

Conjecture 4.4.1. *Let $\mathcal{C}$ be a well-formed code. There exists a term order such that a code is 1-inductively pierced if and only if the Gröbner basis of $I_{\mathcal{C}}$ contains only binomials of degree 2 or less.*

We begin by formally defining monomial term orders and Gröbner bases.

Definition 4.4.2 (Monomial term order [CLO07])**.** A *monomial ordering* $>$ on a set of monomials x^a for $a \in \mathbb{Z}^n_{\geq 0}$, satisfies the following:

(1) The ordering $>$ is a well-ordering on $\mathbb{Z}^n_{\geq 0}$, so every nonempty subset of $\mathbb{Z}^n_{\geq 0}$ has a smallest element under $>$.) This implies that $>$ is a total (or linear) ordering on $\mathbb{Z}^n_{\geq 0}$. (So for every pair of monomials x^a and x^b exactly one of $x^a > x^b$, $x^a = x^b$, or $x^b > x^a$ is true).

(2) If $a > b$ and $c \in \mathbb{Z}^n_{\geq 0}$, then $a + c > b + c$,

Under a monomial term order, each polynomial has a *leading term.* We denote the leading term of a polynomial f by $\mathrm{LT}(f)$.

Definition 4.4.3 (Gröbner basis of an ideal [Ver14])**.** A set of polynomials G is a *Gröbner basis* for an ideal I if $I = \langle G \rangle$ and the leading terms of G generate the ideal of leading terms of the polynomials in I, i.e.: $\langle \mathrm{LT}(G) \rangle = \langle \mathrm{LT}(I) \rangle$.

Definition 4.4.4 ([CLO07])**.** A *reduced Gröbner basis* for a polynomial ideal I is a Gröbner basis G for I such that:

(1) The leading coefficient of p is 1 for all $p \in G$.

(2) For all $p \in G$, no monomial of p lies in $\langle \mathrm{LT}(G - \{p\}) \rangle$.

Furthermore, by Proposition 6 in [CLO07], for any non-zero ideal I and a given monomial ordering, I has a unique reduced Gröbner basis.

Now that we have the definitions necessary to understand term order and a Gröbner basis of an ideal, we show that we can determine that a code on three neurons is k-inductively pierced by determining the degrees of elements of the reduced Gröbner basis with respect to a particular term order.

We investigated Conjecture 4.4.1 using the package `gfanInterface.m2` [Jen] in `Macaulay2`, and we were able to find a term order that worked to accomplish this for codes on $n = 3$ neurons as described in Proposition 4.4.5.

To define our a monomial order, we use a *weighted graded reverse lexicographic order*: Let $deg_w(x^a) = a_1w_1 + a_2w_2 + \ldots + a_nw_n$ with weights given by a weight vector w. Then $x^a < x^b$ if and only if $deg_w(x^a) < deg_w(x^b)$ or $deg_w(x^a) = deg_w(x^b)$ and there exists $1 \leq i \leq n$ such that $a_n = b_n, \ \ldots \ , a_{i+1} = b_{i+1}$, $a_i > b_i$ [Dev15]. That is, we compute the dot product of the weight vector and the exponents of the monomials, and order the monomials under the prescribed weighted reverse lexicographic order.

Proposition 4.4.5. *A well-formed neural code $\mathcal{C}$ on 3 neurons is 1-inductively pierced if and only if the Gröbner basis of $I_{\mathcal{C}}$ with respect to the weighted graded reverse lexicographic order with the weight vector $[0, 0, 0, 1, 1, 1, 0]$ contains only binomials of degree 2 or less.*

Proof. Using the weighted graded reverse lexicographic order with weight vector $[0, 0, 0, 1, 1, 1, 0]$ we computed the reduced Gröbner bases of the toric ideals of each well-formed neural code up to symmetry. We found that only the 0 and 1-inductively pierced codes had reduced Gröbner bases with maximum degree two.

For the A1 case, this ordering weights $p_1,\ p_2,\ p_3$ (corresponding to single neurons firing i.e., codewords 100, 010, and 001) and p_7 (corresponding to triple intersections of neurons firing, so codeword 111) less than $p_4,\ p_5,\ p_6$ (pairwise intersections of place fields, i.e., with codewords 110, 101, 011) and pulls out the cubic generator $(p_1p_2p_3 - p_7)$ as a generator of the toric ideal of $I_{\mathcal{C}}$. See Appendix A.3 for the computational proof. □

CHAPTER 5

SUMMARY

5.1 Using toric ideals to determine k-inductively pierced codes

Our motivating question for this thesis was to determine how to draw the realization of a place field diagram for a neural code assuming we know *a priori* that it is convexly realizable in dimension two. We explained that because of existing work on drawing Euler diagrams done by [SZHR11], an algorithm for drawing such a realization already exists for data sets that are inductively pierced. Then our new question became how to determine whether a neural code is inductively pierced. We showed that once we have determined that a code is inductively pierced (using its toric ideal), we may apply the algorithm for automatically drawing Euler diagrams using circles developed by Stapleton et al. to determine a place field diagram for the neural code, as desired. The following theorem summarizes our the results from this thesis and [GOY].

Main Theorem ([GOY]). *Let $\mathcal{C}$ be well-formed.*

- *The neural code $\mathcal{C}$ is 0-inductively pierced if and only if $I_{\mathcal{C}} = \langle 0 \rangle$.*
- *If the neural code $\mathcal{C}$ is 0- and 1-inductively pierced then $I_{\mathcal{C}}$ is $\langle 0 \rangle$ or generated by quadratics.*
- *If there is a 2-piercing in a diagram of the neural code $\mathcal{C}$, then the toric ideal $I_{\mathcal{C}}$ contains a cubic binomial of a particular form.*

We have completely classified 0-inductively pierced codes and we conjecture that we can completely classify 1-inductively pierced codes. Currently this work is still in progress, but we hope to completely classify k-inductively pierced codes using their toric ideals.

Once we have a full classification, we can immediately understand whether a code is inductively pierced (so, 0-, 1-, or 2-inductively pierced). Then, if a code is convexly realizable in two dimensions, we can draw a realization of the place fields using convex sets by taking the following steps:

- Given a neural code $\mathcal{C}$, compute its toric ideal $I_{\mathcal{C}}$.
- Use toric ideal $I_{\mathcal{C}}$ to determine if $\mathcal{C}$ is 0, 1-, or 2-inductively pierced.
- If the code is 0-, 1, or 2-inductively pierced, draw a place field diagram of the code by the existing algorithm in [SZHR11] that draws Euler diagrams with circles. The algorithm is implemented and available at http://www.eulerdiagrams.org/inductivecircles.html.

Recall the code $\mathcal{C} = \{000000, 100000, 010000, 001000, 000100, 000010, 110000, 011000, 000011, 001100, 000110, 100010, 110010, 010010, 010100, 010110, 011100\}$ from Section 1.2. Using `Macaulay2` we compute the toric ideal of $I_{\mathcal{C}}$, as illustrated in Appendix A.4. In this case,

$$\begin{aligned} I_{\mathcal{C}} = \langle & p_{000100}p_{000010} - p_{000110}, p_{001000}p_{000100} - p_{001100}, p_{010000}p_{000010} - p_{010010}, \\ & p_{010000}p_{000100} - p_{010100}, p_{010000}p_{000100}p_{000010} - p_{010110}, p_{010000}p_{001000} - p_{011000}, \\ & p_{010000}p_{001000}p_{000100} - p_{011100}, p_{100000}p_{000010} - p_{100010}, p_{100000}p_{010000} - p_{110000}, \\ & p_{100000}p_{010000}p_{000010} - p_{110010}\rangle, \end{aligned}$$

and $I_{\mathcal{C}}$ has Gröbner basis

$$\{p_{001000}p_{010110} - p_{000010}p_{011100},\ p_{000100}p_{110010} - p_{100000}p_{010110},$$
$$p_{010000}p_{000100}p_{000010} - p_{010110},\ p_{100000}p_{010000}p_{000010} - p_{110010},$$
$$p_{010000}p_{001000}p_{000100} - p_{011100},\ p_{010100} - p_{010000}p_{000100},\ p_{010010} - p_{010000}p_{000010},$$
$$p_{100010} - p_{100000}p_{000010},\ p_{000110} - p_{000100}p_{000010},\ p_{001100} - p_{001000}p_{000100},$$
$$p_{011000} - p_{010000}p_{001000},\ p_{110000} - p_{100000}p_{010000}\}.$$

From these computations we see that a Gröbner basis of $I_{\mathcal{C}}$ is generated by generators of degree at most 3. Although we cannot determine it automatically yet, it turns out that $\mathcal{C}$ is inductively pierced. So, we can draw a place field diagram by entering the code in the `CirclesMain` program [SZHR11]. The following shows the input and output of the program. Note that to input the code in this program we rename each codeword by its support, omitting commas and braces. Figure 5.1 shows a place field diagram of $\mathcal{C}$.

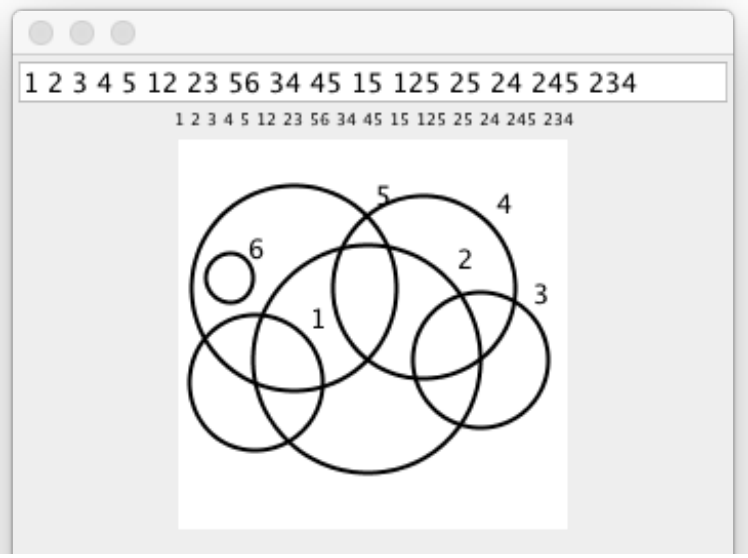


Figure 5.1: A place field diagram of a six-neuron code

BIBLIOGRAPHY

[Ber] C. Berge, *Hypergraphs : combinatorics of finite sets*, vol. 45, Elsevier Science Publishers B.V.

[Bre13] A. Bretto, *Hypergraph theory: an introduction*, Springer, 2013.

[CGJ+15] C. Curto, E. Gross, J. Jeffries, K. Morrison, M. Omar, Z. Rosen, A. Shiu, and N. Youngs, *What makes a neural code convex?*, arXiv:1508.00150 (2015), Submitted to *Bulletin of Mathematical Biology*.

[Cho07] S. Chow, *Generating and drawing area-proportional euler and venn diagrams*, Ph.D. thesis, University of Victoria, 2007.

[CIM+13] C. Curto, V. Itskov, K. Morrison, Z. Roth, and J. L. Walker, *Combinatorial neural codes from a mathematical coding theory perspective*, *Neural Comput.* **25** (2013), no. 7, 1891–1925.

[CIVCY13] C. Curto, V. Itskov, A. Veliz-Cuba, and N. Youngs, *The neural ring: an algebraic tool for analyzing the intrinsic structure of neural codes.*, *Bull. Math. Biol.* **75** (2013), no. 9, 1571–1611.

[CLO07] D. Cox, J. Little, and D. O'Shea, *Ideals, varieties, and algorithms: An introduction to computational algebraic geometry and commutative algebra*, Springer, 2007.

[CY15] Carina Curto and Nora Youngs, *Neural ring homomorphisms and maps between neural codes*, arXiv:1511.00255 (2015).

[Dev15] The Sage Developers, *Sagemath, the Sage Mathematics Software System (Version 7.1)*, 2015, `http://www.sagemath.org`.

[FH02] J. Flower and J. Howse, *Generating euler diagrams*, Proceedings of 2nd International Conference on the Theory and Application of Diagrams, Springer, 2002, pp. 61–75.

[GI14] C. Giusti and V. Itskov, *A no-go theorem for one-layer feedforward networks*, *Neural Comput.* **26** (2014), no. 11, 2527–2540.

[GIK] C. Giusti, V. Itsko, and W. Kronholm, *On convex codes and intersection violators*.

[GOY] E. Gross, N. Obatake, and N. Youngs, *Neural ideals and stimulus space visualization.*

[GP12] E. Gross and S. Petrović, *Combinatorial degree bound for toric ideals of hypergraphs*, arXiv:1206.2512 (2012).

[GS] D. R. Grayson and M. Stillman, *Macaulay2, a software system for research in algebraic geometry.*

[Ham60] Sir W. Hamilton, *Lectures on metaphysics and logic*, William Blackwood and Sons, Edinburgh and London., 1860, Edited by Henry Longueville, Mansel and John Veitch.

[Hun74] Thomas W. Hungerford, *Algebra*, vol. 73, Springer-Verlag New York, 1974.

[HW59] D. H. Hubel and T. N. Wiesel, *Receptive fields of single neurons in the cat's striate cortex*, *J. Physiol.* **148** (1959), no. 3, 574–591.

[Jen] A. N. Jensen, *Gfan, a software system for gröbner fans and tropical varieties*, Available at http://home.imf.au.dk/jensen/software/gfan/gfan.html.

[MS05] E. Miller and B. Sturmfels, *Combinatorial commutative algebra*, vol. 227, Springer-Verlag, 2005, Graduate Texts in Mathematics.

[MT05] D. Maclagan and R. R. Thomas, *Computational algebra and combinatorics of toric ideals*, Commutative Algebra and Combinatorics, Harish Chandra Research Institute, 2005, Available from www.math.rutgers.edu/ maclagan/. With S. Faridi, L. Gold, A. V. Jayanthan, A. Khetan, and T. Puthenpurakal.

[OD71] J. O'Keefe and J. Dostrovsky, *The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat.*, Brain Res. **34** (1971), no. 1, 171–175.

[PS14] S. Petrović and D. Stasi, *Toric algebra of hypergraphs*, Journal of Algebraic Combinatorics **39** (2014), no. 1, 187–208.

[RZF08] P. Rodgers, L. Zhang, and A. Fish, *General euler diagram generation*, International Conference on the Theory and Application of Diagrams, Springer, 2008.

[SAA09] P. Simonetto, D. Auber, and D. Archambault, *Fully automatics visualization of overlapping sets*, Computer Graphics Forum **28** (2009), no. 3.

[Sot03] Frank Sottile, *Toric ideals, real toric varieties, and the algebraic moment map*, Topics in Algebraic Geometry and Geometric Modeling, Contemp. Math. **334** (2003).

[Stu96] Bernd Sturmfels, *Equations defining toric varieties*, arXiv:alg-geom/9610018v1 (1996).

[SZHR11] G. Stapleton, L. Zhang, J. Howse, and P. Rodgers, *Drawing euler diagrams with circles: The theory of piercings*, Visualization and Computer Graphics, IEEE Transactions **17** (2011), no. 7, 1020–1032.

[tt] 4ti2 team, *4ti2—a software package for algebraic, geometric and combinatorial problems on linear spaces*, Available at www.4ti2.de.

[Ver14] J. Vershelde, *Analytic symbolic computation notes*, Jan 2014.

[Vil01] R. H. Villareal, *Monomial algebras*, Marcel Dekker, 2001.

APPENDIX A

COMPUTATIONS

A.1 Catalogue of neural codes on three neurons

Label	Code
A1	000,100,010,001,110,101,011,111
A2	000,100,010,110,101,111
A3	000,100,010,001,110,101,111
A4	000,100,010,110,101,011,111
A5	000,100,010,110,111
A6	000,100,110,101,111
A7	000,100,010,101,111
A8	000,100,010,001,110,111
A9	000,100,001,110,011,111
A10	000,100,010,101,011,111
A11	000,100,110,101,011,111
A12	000,100,110,111
A13	000,100,010,111
A14	000,100,010,001,111
A15	000,110,101,011,111
A16	000,100,011,111
A17	000,110,101,111
A18	000,100,111

A19	000,110,111
A20	000,111
B1	000,100,010,001,110,101
B2	000,100,010,110,101
B3	000,100,010,101,011
B4	000,100,110,101
B5	000,100,110,011
B6	000,110,101
C1	000,100,010,001,110
C2	000,100,010,101
C3	000,100,011
D1	000,100,010,001
E1	000,100,010,001,110,101,011
E2	000,100,010,110,101,011
E3	000,100,110,101,011
E4	000,110,011,101
F1	000,100,010,110
F2	000,100,110
F3	000,110
G1	000,100
H1	000
I1	000,100,010

A.2 Generators of toric ideals for codes on three neurons

Below is a table of the generating sets of I_A, the toric ideal, for the all the different codes on $n = 3$ neurons (up to symmetry) listed in Figure 6 of the Neural Ring paper [CIVCY13].

Generators of I_A	**Codes**
$p_{111} - p_{100}p_{010}p_{001}$ $p_{110} - p_{100}p_{010}$ $p_{101} - p_{100}p_{001}$ $p_{011} - p_{010}p_{001}$	$A1$
$p_{111} - p_{010}p_{101}$ $p_{110} - p_{100}p_{010}$	$A2$
$p_{111} - p_{100}p_{010}p_{001}$ $p_{110} - p_{100}p_{010}$ $p_{101} - p_{100}p_{001}$	$A3$
$p_{111} - p_{100}p_{011}$ $p_{110} - p_{100}p_{010}$ $p_{100}p_{011} - p_{010}p_{101}$	$A4$
$p_{110} - p_{100}p_{010}$	$A5, B2, C1, F1$
$p_{100}p_{111} - p_{110}p_{101}$	$A6$
$p_{111} - p_{010}p_{101}$	$A7$
$p_{111} - p_{100}p_{010}p_{001}$ $p_{110} - p_{100}p_{010}$	$A8$
$p_{111} - p_{100}p_{011}$	$A9, A16$
$p_{111} - p_{010}p_{101}$	$A10$

$p_{100}p_{011} - p_{010}p_{101}$	
$p_{100}^2 p_{011} - p_{110}p_{101}$ $p_{111} - p_{100}p_{011}$	$A11$
$p_{111} - p_{100}p_{010}p_{001}$	$A14$
$p_{111}^2 - p_{110}p_{101}p_{011}$	$A15$
$p_{101} - p_{100}p_{001}$ $p_{110} - p_{100}p_{010}$	$B1$
$p_{100}p_{011} - p_{010}p_{101}$	$B3$
$p_{011} - p_{010}p_{001}$ $p_{101} - p_{100}p_{001}$ $p_{110} - p_{100}p_{001}$	$E1$
$p_{110} - p_{100}p_{010}$ $p_{100}p_{011} - p_{010}p_{101}$	$E2$
$p_{100}^2 p_{011} - p_{110}p_{101}$	$E3$
0	$A12, A13, A17, A18, A19, A20, B4, B5, B6,$ $C2, C3, D1, E4, F2, F3, G1, H1, I1$

A.3 M2 code for calculating Gröbner bases of three neuron codes

```
+ M2 --no-readline --print-width 79
Macaulay2, version 1.7
with packages: ConwayPolynomials, Elimination, IntegralClosure,
LLLBases, PrimaryDecomposition, ReesAlgebra, TangentCone
i1 : installPackage"FourTiTwo"
i2 : p0={0,0,0}, p1={1,0,0}, p2={0,1,0}, p3={0,0,1}, p4={1,1,0},
p5={1,0,1}, p6={0,1,1}, p7={1,1,1}
o2 = ({0, 0, 0}, {1, 0, 0}, {0, 1, 0}, {0, 0, 1}, {1, 1, 0}, {1, 0,
1}, {0, 1, 1}, {1, 1, 1})
o2 : Sequence
i3 : Rweights=QQ[y_1..y_8, MonomialOrder => Weights => (0,0,0,1,1,1,0,0)]
o3 = Rweights
o3 : PolynomialRing
i4 : A2=transpose(matrix{p1,p2,p4,p5,p7})
o4 = | 1 0 1 1 1 |
     | 0 1 1 0 1 |
     | 0 0 0 1 1 |
o4 : Matrix ZZ <--- ZZ
              3       5
i5 :  gens(gb(toricMarkov(A2,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/1
stdio:6:23:(3):[2]: error: no method for assignment to adjacent
objects:
```

```
-- -- -- --
SPACE
I (of class Symbol)
(| 1 0 1 1 1 |, Rweights) (of class Sequence)
|01101| |00011|
i6 : gens(gb(toricMarkov(A2,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/2
o6 = | y_1y_2-y_3 y_3y_4-y_1y_5 y_2y_4-y_5 |
13 o6 : Matrix Rweights <--- Rweights
i7 : toricMarkov(A2,Rweights)
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/3
o7=ideal(yy -y,yy -y)
245123 o7 : Ideal of Rweights
i8 : A3=transpose(matrix{p1,p2,p3,p4,p5,p7})
o8 = | 1 0 0 1 1 1 | |010101| |001011|
36 o8 : Matrix ZZ <--- ZZ
i9 : toricMarkov(A3,Rweights)
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/4
o9=ideal(-y +yy,-y +yy,-y +yyy) 513 412 6123
o9 : Ideal of Rweights
i10 : gens(gb(toricMarkov(A3,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/5
```

```
o10 = | y_6-y_1y_2y_3 y_5-y_1y_3 y_4-y_1y_2 |

13 o10 : Matrix Rweights <--- Rweights

i11 : A4=transpose(matrix{p1,p2,p4,p5,p6,p7})

o11 = | 1 0 1 1 0 1 | |011011| |000111|

36 o11 : Matrix ZZ <--- ZZ

i12 : toricMarkov(A4,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/6

o12=ideal(-yy +yy,yy -y,yy -y) 24 1515 612 3

o12 : Ideal of Rweights

i13 : gens(gb(toricMarkov(A4,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/7

o13 = | y_1y_2-y_3 y_3y_5-y_2y_6 y_1y_5-y_6 y_3y_4-y_1y_6 y_2y_4-y_6 |

15 o13 : Matrix Rweights <--- Rweights

i14 : A5=transpose(matrix{p1,p2,p4,p7})

o14 = | 1 0 1 1 | |0111| |0001|

34 o14 : Matrix ZZ <--- ZZ

i15 : toricMarkov(A5,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/8

o15=ideal(yy -y) 123

o15 : Ideal of Rweights

i16 : gens(gb(toricMarkov(A5,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/9
```

```
o16 = | y_1y_2-y_3 |

11 o16 : Matrix Rweights <--- Rweights

i17 : A6=transpose(matrix{p1,p4,p5,p7})

o17 = | 1 1 1 1 | |0101| |0011|

34 o17 : Matrix ZZ <--- ZZ

i18 : toricMarkov(A6,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/10

o18=ideal(yy -yy) 14 23

o18 : Ideal of Rweights

i19 : gens(gb(toricMarkov(A6,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/11

o19 = | y_1y_4-y_2y_3 |

11 o19 : Matrix Rweights <--- Rweights

i20 : A7=transpose(matrix{p1,p2,p5,p7})

o20 = | 1 0 1 1 | |0101| |0011|

34 o20 : Matrix ZZ <--- ZZ

i21 : toricMarkov(A7,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/12

o21=ideal(-y +yy) 423

o21 : Ideal of Rweights

i22 : gens(gb(toricMarkov(A7,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/13
```

```
o22 = | y_4-y_2y_3 |

11 o22 : Matrix Rweights <--- Rweights

i23 : A8=transpose(matrix{p1,p2,p3,p4,p7})

o23 = | 1 0 0 1 1 | |01011| |00101|

35 o23 : Matrix ZZ <--- ZZ

i24 : toricMarkov(A8,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/14

o24=ideal(-y +yy,-y +yyy) 412 5123

o24 : Ideal of Rweights

i25 : gens(gb(toricMarkov(A8,Rweights)))

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/15

o25 = | y_5-y_1y_2y_3 y_4-y_1y_2 |

12 o25 : Matrix Rweights <--- Rweights

i26 : A9=transpose(matrix{p1,p3,p4,p6,p7})

o26 = | 1 0 1 0 1 | |00111| |01011|

35 o26 : Matrix ZZ <--- ZZ

i27 : toricMarkov(A9,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/16

o27=ideal(-y +yy,yy -yy) 5 2314 23

o27 : Ideal of Rweights

i28 : gens(gb(toricMarkov(A9,Rweights)))

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/17
```

```
o28 = | y_5-y_2y_3 y_1y_4-y_2y_3 |

12 o28 : Matrix Rweights <--- Rweights

i29 : A10=transpose(matrix{p1,p2,p5,p6,p7})

o29 = | 1 0 1 0 1 | |01011| |00111|

35

o29 : Matrix ZZ  <--- ZZ

i30 : toricMarkov(A10,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/18

o30=ideal(-y +yy,yy -yy) 5 2314 23

o30 : Ideal of Rweights

i31 : gens(gb(toricMarkov(A10,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/19

o31 = | y_5-y_2y_3 y_1y_4-y_2y_3 |

12 o31 : Matrix Rweights <--- Rweights

i32 :  A11=transpose(matrix{p1,p4,p5,p6,p7})

o32 = | 1 1 1 0 1 | |01011| |00111|

35 o32 : Matrix ZZ <--- ZZ

i33 : toricMarkov(A11,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/20

o33=ideal(-y +yy,yy -yy) 5 2314 23

o33 : Ideal of Rweights

i34 : toricMarkov(A11,Rweights)

using temporary file name /var/folders/jh/
```

```
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/21

2 o34=ideal(yy -y,yy -yy)

14 514 23 o34 : Ideal of Rweights

i35 : gens(gb(toricMarkov(A11,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/22

o35 = | y_1y_5-y_2y_3 y_1y_4-y_5 y_5^2-y_2y_3y_4 |

13 o35 : Matrix Rweights <--- Rweights

i36 : A12=transpose(matrix{p1,p4,p7})

o36 = | 1 1 1 | |011| |001|

33 o36 : Matrix ZZ <--- ZZ

i37 : toricMarkov(A12,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/23

o37 = ideal 0

o37 : Ideal of Rweights

i38 : gens(gb(toricMarkov(A12,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/24

o38 = 0

                    1

o38 : Matrix Rweights  <--- 0

i39 : A13=transpose(matrix{p1,p2,p7})

o39 = | 1 0 1 | |011| |001|

33 o39 : Matrix ZZ <--- ZZ

i40 : toricMarkov(A13,Rweights)
```

```
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/25
o40 = ideal 0
o40 : Ideal of Rweights
i41 : gens(gb(toricMarkov(A13,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/26
o41 = 0
                     1
o41 : Matrix Rweights  <--- 0
i42 : A14=transpose(matrix{p1,p2,p3,p7})
o42 = | 1 0 0 1 | |0101| |0011|
34 o42 : Matrix ZZ <--- ZZ
i43 : toricMarkov(A14,Rweights)
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/27
o43=ideal(-y +yyy) 4 123
o43 : Ideal of Rweights
i44 : gens(gb(toricMarkov(A14,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/28
o44 = | y_4-y_1y_2y_3 |
11 o44 : Matrix Rweights <--- Rweights
i45 : A15=transpose(matrix{p4,p5,p6,p7})
o45 = | 1 1 0 1 | |1011| |0111|
34 o45 : Matrix ZZ <--- ZZ
```

```
i46 : toricMarkov(A15,Rweights)
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/29
2
o46=ideal(-y +yyy) 4 123
o46 : Ideal of Rweights
i47 : gens(gb(toricMarkov(A15,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/30
o47 = | y_4^2-y_1y_2y_3 |
11 o47 : Matrix Rweights <--- Rweights
i48 :  A16=transpose(matrix{p1,p6,p7})
o48 = | 1 0 1 | |011| |011|
33 o48 : Matrix ZZ <--- ZZ
i49 : toricMarkov(A16,Rweights)
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/31
o49=ideal(yy -y) 123
o49 : Ideal of Rweights
i50 : gens(gb(toricMarkov(A16,Rweights)))
      gens(gb(toricMarkov(A16,Rweights)))
stdio:52:1:(3): error: missing semicolon or comma on previous line?
i50 : gens(gb(toricMarkov(A16,Rweights)))
using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/32
o50 = | y_1y_2-y_3 |
```

```
11 o50 : Matrix Rweights <--- Rweights

i51 : A17=transpose(matrix{p4,p5,p7})

o51 = | 1 1 1 | |101| |011|

33 o51 : Matrix ZZ <--- ZZ

i52 : toricMarkov(A17,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/33

o52 = ideal 0

o52 : Ideal of Rweights

i53 : gens(gb(toricMarkov(A17,Rweights)))

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/34

o53 = 0

                     1

o53 : Matrix Rweights  <--- 0

i54 : A18=transpose(matrix{p1,p7})

o54 = | 1 1 | |01| |01|

32 o54 : Matrix ZZ <--- ZZ

i55 : toricMarkov(A18,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/35

o55 = ideal 0

o55 : Ideal of Rweights

i56 : gens(gb(toricMarkov(A18,Rweights)))

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/36
```

```
o56 = 0

o56 : Matrix Rweights  <--- 0

1

i57 :  A19=transpose(matrix{p4,p7})

o57 = | 1 1 |
      | 1 1 |
      | 0 1 |

o57 : Matrix ZZ <--- ZZ
               3       2

i58 : toricMarkov(A19,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/37

o58 = ideal 0

o58 : Ideal of Rweights

i59 : gens(gb(toricMarkov(A19,Rweights)))

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/38

o59 = 0
                     1

o59 : Matrix Rweights  <--- 0

i60 : A20=transpose(matrix{p7})

o60 = | 1 | |1| |1|

31 o60 : Matrix ZZ <--- ZZ

i61 : toricMarkov(A20,Rweights)

using temporary file name /var/folders/jh/
1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/39
```

```
o61 = ideal 0

o61 : Ideal of Rweights

i62 : gens(gb(toricMarkov(A20,Rweights)))

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/40

o62 = 0

                      1

o62 : Matrix Rweights  <--- 0

i63 : B1=transpose(matrix{p1,p2,p3,p4,p5})

o63 = | 1 0 0 1 1 | |01010| |00101|

35 o63 : Matrix ZZ <--- ZZ

i64 : toricMarkov(B1,Rweights)

using temporary file name /var/folders/jh/

1kvb85j94090q8g5yvl_z95w0000gn/T/M2-516-0/41

o64=ideal(-y +yy,-y +yy) 513412

o64 : Ideal of Rweights
```

A.4 M2 code for final six neuron code example

$\mathcal{C} = \{000000, 100000, 010000, 001000, 000100, 000010, 110000, 011000, 000011, 001100, 000110, 100010, 110010, 010010, 010100, 010110, 011100\}$

```
+ M2 --no-readline --print-width 79
Macaulay2, version 1.7
with packages: ConwayPolynomials, Elimination, IntegralClosure, LLLBases,
               PrimaryDecomposition, ReesAlgebra, TangentCone
i1 : installPackage"FourTiTwo"
i2 : A=transpose(matrix{{1,0,0,0,0,0},{0,1,0,0,0,0},{0,0,1,0,0,0},
{0,0,0,1,0,0},{0,0,0,0,1,0},{1,1,0,0,0,0},{0,1,1,0,0,0},{0,0,0,0,1,1},
{0,0,1,1,0,0},{0,0,0,1,1,0},{1,0,0,0,1,0},{1,1,0,0,1,0},{0,1,0,0,1,0},
{0,1,0,1,0,0},{0,1,0,1,1,0},{0,1,1,1,0,0}})
o2 = | 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 |
     | 0 1 0 0 0 1 1 0 0 0 0 1 1 1 1 1 |
     | 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 1 |
     | 0 0 0 1 0 0 0 0 1 1 0 0 0 1 1 1 |
     | 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 0 |
     | 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 |
               6        16
o2 : Matrix ZZ  <--- ZZ
i3 : R=QQ[x_1..x_16]
o3 = R
o3 : PolynomialRing
i4 : toricMarkov(A,R)
using temporary file name /var/folders/jh/1kvb85j94090q8g5yvl_z95w0000gn/
```

```
T/M2-710-0/1
o4 = ideal (x x  - x  , x x  - x , x x  - x  , x x  - x  , x x x  - x  , x x  -
             4 5    10   3 4    9   2 5    13   2 4    14   2 4 5    15   2 3

     ------------------------------------------------------------------------

     x , x x x  - x  , x x  - x  , x x  - x , x x x  - x  )
      7   2 3 4    16   1 5    11   1 2    6   1 2 5    12
o4 : Ideal of R
i5 : R2=QQ[y_1..y_16, MonomialOrder => Weights=> {0,0,0,0,0,1,1,1,1,1,1,
0,1,1,0,0}]
o5 = R2
o5 : PolynomialRing
i6 : I=toricMarkov(A,R2)
using temporary file name /var/folders/jh/1kvb85j94090q8g5yvl_z95w0000gn/
T/M2-710-0/2
o6 = ideal (- y   + y y , - y  + y y , - y   + y y , - y   + y y , y y y  -
               10    4 5     9    3 4     13    2 5     14    2 4   2 4 5

     ------------------------------------------------------------------------

     y  , - y  + y y , y y y  - y  , - y   + y y , - y  + y y , y y y  - y  )
      15     7    2 3   2 3 4    16     11    1 5     6    1 2   1 2 5    12
o6 : Ideal of R2
i7 : gens(gb(I))
o7 = | y_3y_15-y_5y_16 y_4y_12-y_1y_15 y_2y_4y_5-y_15 y_1y_2y_5-y_12

     ------------------------------------------------------------------------

     y_2y_3y_4-y_16 y_14-y_2y_4 y_13-y_2y_5 y_11-y_1y_5 y_10-y_4y_5 y_9-y_3y_4

     ------------------------------------------------------------------------

     y_7-y_2y_3 y_6-y_1y_2 |
```

```
                1        12
o7 : Matrix R2  <--- R2
```