NEW TOOLS FOR CLASSIFYING CONVEX NEURAL CODES: THE FACTOR COMPLEX AND THE WHEEL

A Dissertation

by

ALEXANDER MARK RUYS DE PEREZ

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2021

Major Subject: Mathematics

ABSTRACT

The neural code has prompted many questions in pure mathematics concerning how much topological data can be stored combinatorially. The question of whether one can determine the convexity of a neural code is particularly prominent. In this dissertation, we provide new tools toward answering this question. First, we introduce a related object called the factor complex, and show how it encodes a property of the neural code called max-intersection-completeness. Second, we introduce a new type of nonconvex phenomenon called a wheel, and show how to read it combinatorially.

DEDICATION

To Mom, Dad, Stephanie, and Andrew, for all your love and support throughout.

ACKNOWLEDGMENTS

I’d like to thank my advisors, Anne Shiu and Laura Matusevich, for helping me through this project. Their advice on everything, be it research, writing, or presenting, was invaluable. I could not have done this without them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLE Page

1. INTRODUCTION

In 1971, John O'Keefe discovered a group of neurons in the hippocampus of a rat that had a distinctive function. Each of these neurons, which would later be called *place cells*, had a relationship with a certain region of physical space, such that the neuron would fire if and only if the rat was currently in that specific region [1]. To the exclusion of any other activity in the brain, even the connections of that place cell with other neurons, one could determine the rat's presence in the region (called a *receptive field*) solely on the neuron's firing.

This discovery prompted a flurry of new activity in pure mathematics, creating the area of neural codes [2][3][4][5][6][7][8][9][10]. A rat (or more generally a mammal, as later work found place cells in mammals besides rats) does not just have one place cell for a specific location. Often an area will have several different receptive fields with their own corresponding place cells. These receptive fields are not necessarily disjoint from another; indeed, one may have different place cells firing simultaneously as the mammal moves into an intersection of their fields. Thus from a collection of place cells for a particular location, one can create a set of codewords that record the different ways the neurons' fields overlap. Each codeword is a binary string of 1's and 0's, with a 1 as the ith digit meaning that the ith neuron is firing, and 0 as the ith digit indicating otherwise. The set of all these codewords, called the *neural code*, is thus a record of all the possible combinations of firings of the place cells.

What motivates the interest in neural codes from mathematics is the question of how much information about the receptive fields these codes contain. Clearly, the code details the various intersections of the receptive fields, but can something be said about the topological properties of the receptive fields? In particular, can we determine if a neural code encodes whether the receptive fields are convex? The question of convexity comes from experimental data, which describes the receptive fields using points on a map of a location, with each point representing an occurrence of a place cell firing. The contours of the point map for a specific place cell are convex, suggesting that receptive fields are also convex. Furthermore, there is no strict cutoff between the area in which a place cell fires and the area in which it does not. Rather, the majority of firings is concentrated in the interior of the observed receptive field, with firings decreasing in frequency as one approaches the boundary. Thus, all the neural codes recorded in the lab appear to be for collections of convex, open receptive fields. Are these codes a special class of elements from the set of collections of binary strings? And if so, can we determine whether a collection of binary strings belongs to this class? That is, by examining the strings of 1's and 0's of a code, can we determine whether it is possible to put down a collection of convex open sets, in $\mathbb{R}^2$, $\mathbb{R}^3$, or even arbitrary $\mathbb{R}^d$, whose realization would be that code?

The question as to which codes are convex is nontrivial. Just as the experiments of O'Keefe and others show the existence of such neural codes with convex open realizations, so too are there collections of binary strings for which such a realization is impossible. Previous work has been successful in classifying many codes as convex [10], as well as ruling out convexity for many others [9][6]. However, there is still a large number of codes yet to be classified as either convex or nonconvex. In this dissertation, we make a contribution to solving the problem of convexity in two ways. First, we show how a certain property that guarantees convexity, called max-intersection-completeness, can be found in certain algebraic and combinatorial objects related to the neural code. Second, we introduce a new phenomenon, called a *wheel*, the presence of which in a code forbids convexity.

This dissertation is organized as follows: In Chapter 2, we introduce the concepts and objects related to neural codes, as well as review prior results. Chapter 3 pertains to irreducible, decomposable, and pure neural codes. The next two chapters discuss our main results. Chapter 4 details an object we call the *factor complex*, and explains how it helps determine max-intersection-completeness. Chapter 5 introduces the *wheel*. Finally, we review our work and explore possible future directions in Chapter 6.

2. BACKGROUND

Neural codes have a biological inspiration. Mammals have certain neurons, known as *place cells*, that correspond with certain regions of physical space [1]. Whether or not a place cell fires depends on whether the mammal is currently in that place cell's corresponding region, called a *receptive field*. Thus, the layout of the receptive fields gives rise to a collection of certain combinations of neurons firing, which we call the *(biological) neural code*. In mathematics, the neural code has prompted questions into how topological and geometric information about a space can be stored in combinatorial data. Mathematicians generalize the receptive fields into subsets U_i of $\mathbb{R}^d$ and replace the firing neurons with Boolean variables i, where i denotes inclusion or exclusion in U_i. Thus the *(mathematical) neural code* $\mathcal{C}$ is the set of all possible firing combinations of place cells for a mammal with given receptive field placement $\mathcal{U} = \{U_i\}$. That is, $\mathcal{C}$ encodes the intersections of the U_i's.

It is natural to ask whether we can get $\mathcal{C}$ from a $\mathcal{U}$ when the elements of $\mathcal{U}$ have certain properties. For instance, one can find a realization for any $\mathcal{C}$ if there are no restrictions on the properties of the U_i [2, Lemma 2.1]. However, there might be no such realization if we require each U_i to be open and connected. As an example, consider the code $\mathcal{C}_0 = \{\varnothing, 12, 13\}$, where if we require each U_i to be open, then U_2 and U_3 necessarily form a disconnection of U_1. In another case, there might be a realization where each U_i is open and connected, but not when each U_i is closed and convex.

Convexity, in particular, has been the main focus in the research on neural codes. $\mathcal{C}$ is *open convex* if it has a realization $\mathcal{U}$, where each U_i is open and convex. Similarly, $\mathcal{C}$ is *closed convex* if it has a realization where each U_i is closed and convex. (For this dissertation, I will focus only on open convexity. Thus, for the sake of brevity, assume "convex" means "open convex" unless specifically stated otherwise.) Perhaps the biggest breakthrough has been the introduction of methods showing how this purely geometric property, convexity, can be read from the purely combinatorial information of the code $\mathcal{C}$.

2.1 The Neural Code as a Mathematical Object

Definition 2.1.1. A *code on n neurons* is a subset $\mathcal{C} \subseteq 2^{[n]}$, where $[n] = \{1, 2, \ldots, n\}$. Elements of $\mathcal{C}$ are *codewords*, and a *maximal codeword* is a codeword that is maximal with respect to inclusion among codewords of $\mathcal{C}$.

Remark 2.1.2. Alternatively, we may describe a code on n neurons as a collection of binary strings of length n:

$$\mathcal{C} \subseteq \{0,1\}^n.$$

The codewords are the individual binary strings. The equivalence between a codeword $c \in 2^{[n]}$ and a codeword $c \in \{0,1\}^{[n]}$ is

$$i \in c \in 2^{[n]} \iff x_i = 1 \text{ for } c \in \{0,1\}^n,$$

where x_i is the ith digit of c as a binary string. For example, if $c = \{2,3\}$ is a codeword of a code on 3 neurons, then its representation as a binary string would be 011.

We will default to considering $\mathcal{C}$ as a subset of $2^{[n]}$ for this dissertation, although there will be occasions where it makes more sense to consider the codewords as binary strings.

One final note concerning notation: when describing a codeword $c \in 2^{[n]}$, we will eschew the set notation. For example, instead of describing a code as $\mathcal{C} = \{\varnothing, \{1\}, \{2,3\}, \{1,2,3\}\}$ we will instead write it as $\mathcal{C} = \{\varnothing, 1, 23, 123\}$

Definition 2.1.3. A *realization* of a code $\mathcal{C}$ in $\mathbb{R}^d$ is a collection $\mathcal{U} = \{U_i\}_{i=1}^n$ of subsets of some $X \subseteq \mathbb{R}^d$ such that $c \in \mathcal{C}$ if and only if

$$(\cap_{i\in c} U_i) \setminus (\cup_{j \in [n]\setminus c} U_j) \neq \varnothing.$$

A realization is a *realization of open sets* if each U_i is open, and a *realization of closed sets* if each U_i is closed. (Recall that unless stated otherwise, a realization is assumed to be open.)

The set $(\cap_{i\in c}U_i) \setminus (\cup_{j\in[n]\setminus c}U_j)$ is the *atom of* c *in the realization* $\mathcal{U}$. We denote $U_\sigma := \cap_{i\in\sigma}U_i$, and, by convention, $U_\varnothing := X$.

Intuitively, we have that $\mathcal{U}$ is a realization of $\mathcal{C}$ on n neurons if and only if for each $c \in 2^{[n]}$, we have that c is a codeword if and only if there is some point that is contained in precisely those U_i for which $i \in c$, and none other.

Definition 2.1.4. If the sets $U_1, U_2, \ldots, U_n$ are convex, then $\mathcal{U}$ is a *convex realization* of $\mathcal{C}$. Codes that possess convex realizations are known as *convex codes*.

Remark 2.1.5. All codes in this article are assumed to contain the empty codeword $\varnothing$, and thus we may always assume that $X = \mathbb{R}^d$. Whether or not a code contains $\varnothing$ does not affect convexity, [8, Remark 2.19].

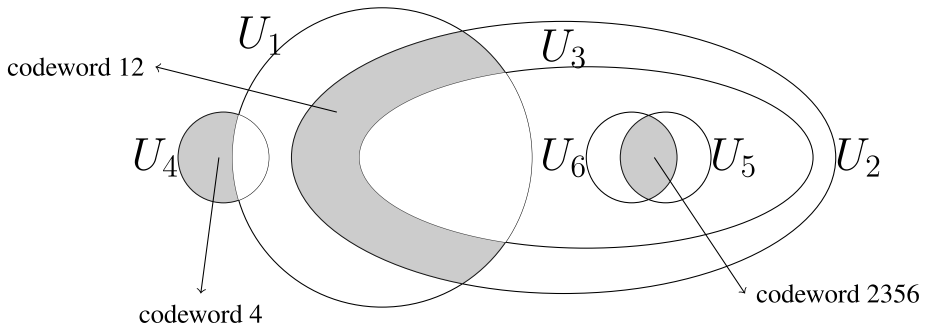


Figure 2.1: Realization of the code in Example 2.1.6. Each receptive field U_i is the region enclosed by the labeled ellipse. For illustration, the atoms of three codewords have been shaded and labeled.

Example 2.1.6. Consider the code

$$\mathcal{C} = \{2356, 123, 235, 236, 12, 14, 23, 1, 2, 4, \varnothing\},$$

which per Remark 2.1.2 can equivalently be described as

$$\mathcal{C} = \{011011, 111000, 011010, 011001, 110000, 100100, 011000, 100000, 010000, 000100, 000000\}.$$

Figure 2.1 shows one possible realization of $\mathcal{C}$ in $\mathbb{R}^2$. Here, the receptive fields are the interiors of the ellipses. As examples, the areas corresponding to codewords 4, 12, and 2356 have been filled in gray. As one can observe from the realization, $\mathcal{C}$ is a convex code.

Example 2.1.7. The code $\mathcal{D} = \{\varnothing, 12, 13\}$ is not convex. As an informal proof, consider a realization $\mathcal{U} = \{U_1, U_2, U_3\}$ of $\mathcal{D}$. We have that U_2 and U_3 form a disconnection of U_1, thus U_1 cannot be convex.

2.2 The Simplicial Complex of a Neural Code

Recall that an *abstract simplicial complex* Δ on n vertices is a subset of $2^{[n]}$ that is closed under inclusion. That is, if $\tau \subseteq \sigma$ and $\sigma \in \Delta$, then $\tau \in \Delta$ as well. An element σ of Δ is called a *face*, and those faces that are maximal in Δ with respect to inclusion are called *facets*. The *dimension* $\dim(\sigma)$ of a face σ is $|\sigma| - 1$, and the *dimension* of Δ is $\max\{\dim(\sigma) \mid \sigma \in \Delta\}$. If for every facet F of Δ we have $\dim(F) = \dim(\Delta)$, then Δ is *pure*. Lastly, if σ is a face of Δ, then the *link* of σ in Δ is the simplicial complex

$$\mathrm{Lk}_\sigma(\Delta) := \{\tau \in \Delta \mid \sigma \cap \tau = \varnothing \text{ and } \sigma \cup \tau \in \Delta\}.$$

Links are usually written as $\mathrm{Lk}_\Delta(\sigma)$, rather than $\mathrm{Lk}_\sigma(\Delta)$; however, we follow the notation of [9].

Previous work in classifying convex and nonconvex codes makes extensive use of the simplicial complex $\Delta(\mathcal{C})$:

Definition 2.2.1. Let $\mathcal{C} \subseteq 2^{[n]}$ be a code on n neurons. The *simplicial complex* $\Delta(\mathcal{C})$ *of* $\mathcal{C}$ is the smallest simplicial complex containing $\mathcal{C}$. That is,

$$\Delta(\mathcal{C}) = \{\sigma \subseteq [n] \mid \sigma \subseteq c \text{ for some } c \in \mathcal{C}\}.$$

Example 2.2.2. The simplicial complex of $\mathcal{C}$ from Example 2.1.6 is (nonempty codewords of $\mathcal{C}$ in **bold**)

$$\Delta(\mathcal{C}) = \{\mathbf{2356}, \mathbf{123}, \mathbf{235}, \mathbf{236}, 256, 356, \mathbf{12}, 13, \mathbf{14}, \mathbf{23}, 25, 26, 35, 36, 56, \mathbf{1}, \mathbf{2}, 3, \mathbf{4}, 5, 6, \varnothing\}.$$

Its facets are 2356 and 123, the maximal codewords of $\mathcal{C}$. In fact for any $\mathcal{C}$ the facets of $\Delta(\mathcal{C})$ are precisely the maximal codewords of $\mathcal{C}$.

The significance of $\Delta(\mathcal{C})$ is that it is the *nerve* of any realization $\mathcal{U}$ of $\mathcal{C}$:

Definition 2.2.3. Let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be a collection of subsets of $\mathbb{R}^d$. The *nerve* $\mathcal{N}(\mathcal{U})$ of $\mathcal{U}$ is the simplicial complex on n vertices such that for $\sigma \in 2^{[n]}$, we have

$$\sigma \in \mathcal{N}(\mathcal{U}) \quad \text{if and only if} \quad \bigcap_{i\in\sigma} U_i \neq \varnothing.$$

Thus, given a code $\mathcal{C}$ on n neurons and its simplicial complex $\Delta(\mathcal{C})$, for each $\sigma \in 2^{[n]}$ we can describe the intersection $\bigcap_{i\in\sigma} U_i$ from any realization of $\mathcal{C}$ in one of three ways:

(1) $\sigma \notin \Delta(\mathcal{C})$ means $\bigcap_{i\in\sigma} U_i$ is empty,

(2) $\sigma \in \Delta(\mathcal{C}) \setminus \mathcal{C}$ means $\bigcap_{i\in\sigma} U_i$ is nonempty but covered by the U_j's for $j \notin \sigma$, and

(3) $\sigma \in \mathcal{C}$ means $\bigcap_{i\in\sigma} U_i$ is nonempty and not covered by the other U_j's.

Every simplicial complex Δ provides two sets of faces used to classify those codes $\mathcal{C}$ for which $\Delta(\mathcal{C}) = \Delta$.

Definition 2.2.4. Let Δ be a simplicial complex, and let $\sigma \in \Delta$.

- σ is a *max-intersection face* if there exist facets $F_1, F_2, \ldots, F_s$ such that $\sigma = \bigcap_{t=1}^{s} F_t$.
- σ is a *mandatory face* if the link $\mathrm{Lk}_\sigma(\Delta)$ of σ in Δ is not contractible.
- The code that consists only of the facets and mandatory faces of Δ is the *minimal code* $\mathcal{C}_{\min}(\Delta)$ of Δ.

Remark 2.2.5.

- Every mandatory face is also a max-intersection face [9, Corollary 4.6]. However, not every max-intersection face is a mandatory face [9, Example 2.2].
- Previous work does not use the term "mandatory faces"; when introduced in [9] they were called "mandatory codewords". We prefer using "faces" instead of "codewords" since we are considering the objects as belonging to a simplicial complex Δ, but not necessarily a neural code $\mathcal{C}$.
- We say that a code containing all of its max-intersection faces is *max-intersection-complete*. We say that a code is *intersection-complete* if the intersection of any codewords of $\mathcal{C}$, not just those of the maximal ones, is contained in $\mathcal{C}$ [9].

From the max-intersection and mandatory faces of Δ, we can classify as either convex or nonconvex a significant number of those codes $\mathcal{C}$ for which $\Delta(\mathcal{C}) = \Delta$.

Proposition 2.2.6. *Let $\mathcal{C}$ be a code on n neurons, and $\Delta(\mathcal{C})$ its simplicial complex.*

(i) [10, Theorem 1.2] If $\mathcal{C}$ contains every max-intersection face of $\Delta(\mathcal{C})$, then $\mathcal{C}$ is convex.

(ii) [9, Theorem 1.3] If $\mathcal{C}$ does not contain every mandatory face of $\Delta(\mathcal{C})$, then $\mathcal{C}$ is not convex.

Note the significance that Proposition 2.2.6 holds for the minimal code $\mathcal{C}_{\min}(\Delta)$ of Δ: for any $\mathcal{C}$ such that $\Delta(\mathcal{C}) = \Delta$, if $\mathcal{C}_{\min}(\Delta) \nsubseteq \mathcal{C}$, then $\mathcal{C}$ is not convex.

Remark 2.2.7. The importance of $\mathcal{C}_{\min}(\Delta)$ is not just limited to determining nonconvexity. There is a monotonicity result for convexity of codes: if $\mathcal{C}$ and $\mathcal{D}$ are codes on n neurons with $\mathcal{C} \subseteq \mathcal{D}$ and $\Delta(\mathcal{C}) = \Delta(\mathcal{D})$, then $\mathcal{C}$ convex implies $\mathcal{D}$ convex [10, Theorem 1.3]. As a result, proving that $\mathcal{C}_{\min}(\Delta)$, the smallest code of Δ with all the mandatory faces, is indeed convex shows that any code of Δ with all the mandatory faces is convex.

The reason why a missing mandatory face causes nonconvexity comes from a phenomenon known as a *local obstruction*, introduced in [9] and [11]. Codes with local obstructions are known to be nonconvex [9].

Definition 2.2.8. Let $\mathcal{C}$ be a code on n neurons, and assume that $\mathcal{C}$ is realized by a collection $\mathcal{U} = \{U_i\}_{i=1}^n$ of open subsets of some $\mathbb{R}^d$. A *local obstruction* of $\mathcal{C}$ is a pair (σ, τ) of nonempty, disjoint subsets of $[n]$ such that

$$U_\sigma \subseteq \bigcup_{j \in \tau} U_j$$

and the link $\mathrm{Lk}_\sigma(\Delta(\mathcal{C})|_{\sigma\cup\tau})$ is not contractible.

Proposition 2.2.9. *[9, Lemma 3.6] If $\mathcal{C}$ is convex, then $\mathcal{C}$ has no local obstructions.*

A missing mandatory face is a type of local obstruction. Indeed, if σ is a face of the simplicial complex Δ such that $\mathrm{Lk}_\sigma(\Delta)$ is not contractible, then $(\sigma, [n]\setminus\sigma)$ is a local obstruction for any code $\mathcal{C}$ for which $\sigma \notin \mathcal{C}$ and $\Delta(\mathcal{C}) = \Delta$. Furthermore, any face σ of $\Delta(\mathcal{C})$ with $\mathrm{Lk}_\sigma(\Delta)$ not contractible must be a max-intersection face due to the following result:

Proposition 2.2.10. *[9, Corollary 4.6] Let $\sigma \in \Delta$ be nonempty. If σ is not an intersection of facets of Δ, then $Lk_\sigma(\Delta)$ is a cone and hence contractible.*

However, what about those local obstructions (σ, τ) where $\mathrm{Lk}_\sigma(\Delta(\mathcal{C})|_{\sigma\cup\tau})$ is not contractible, with $\Delta(\mathcal{C})|_{\sigma\cup\tau} \subsetneq \Delta(\mathcal{C})$? Is it possible that a code $\mathcal{C}$ can contain all the faces of $\Delta(\mathcal{C})$ whose link (in $\Delta(\mathcal{C})$) is not contractible, yet still contain such an obstruction? The answer is no:

Proposition 2.2.11. *[9, Proposition 4.8] A code $\mathcal{C}$ has no local obstructions if and only if $\sigma \in \mathcal{C}$ for every $\sigma \in \Delta(\mathcal{C})$ such that $Lk_\sigma(\Delta)$ is noncontractible.*

This is because if a code $\mathcal{C}$ has a local obstruction, then $\mathcal{C}$ must have a local obstruction (σ, τ) where σ is a missing mandatory face. The essential idea is that if $\mathrm{Lk}_\sigma(\Delta(\mathcal{C})|_{\sigma\cup\tau})$ is non-contractible, then for $v \notin \sigma\cup\tau$ either $\mathrm{Lk}_\sigma(\Delta(\mathcal{C})|_{\sigma\cup\tau\cup\{v\}})$ or $\mathrm{Lk}_{\sigma\cup\{v\}}(\Delta(\mathcal{C})|_{\sigma\cup\tau\{v\}})$ is non-contractible. In this way, the local obstruction "bubbles up" through increasingly larger simplicial complexes until finally $\Delta(\mathcal{C})|_{\sigma\cup\tau} = \Delta(\mathcal{C})$.

We stress this "bubble up" result and its proof idea for two reasons. The first is computational: with this result the number of links one must check to verify the existence of a local obstruction decreases dramatically. Secondly, the result explains how local obstructions interact with the theory of max-intersection faces. If local obstructions prevent convexity while containment of all the max-intersection faces guarantee convexity, then the presence of a local obstruction must also prevent containment of all the max-intersection faces. We will make use of this reasoning when we introduce our own nonconvex phenomenon in Chapter 5.

With the results on the containment of max-intersection faces or the non-containment of mandatory faces respectively confirming or forbidding convexity, we may now classify a broad swathe of codes. The frontier of convex codes, therefore, lies in analyzing those codes $\mathcal{C}$ that contain all of the mandatory faces of $\Delta(\mathcal{C})$ but not all of the max-intersection faces. For the sake of brevity I will use the term *undecided* codes to refer to these codes. There are both convex and nonconvex examples of undecided codes. The following is an undecided code that turns out to be convex:

Example 2.2.12 (A code that is convex but not max-intersection-complete)**.** The following code is not max-intersection-complete, as $1 = 123 \cap 134 \cap 145$ is missing:

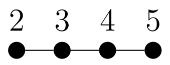
$$\mathcal{C} = \{\mathbf{123}, \mathbf{134}, \mathbf{145}, 13, 14, \varnothing\} .$$

However, $\mathcal{C}$ is convex, as shown in [9, Figure 3c].

Example 2.2.13 (A non-convex code with no local obstructions)**.** The following code was the first example found of a nonconvex code with no local obstructions [12, Theorem 3.1]:

$$\mathcal{C}^\star = \{\mathbf{2345}, \mathbf{123}, \mathbf{134}, \mathbf{145}, 13, 14, 23, 34, 45, 3, 4, \varnothing\} .$$

Note that $\mathcal{C}^\star$ contains all of the mandatory faces of $\Delta(\mathcal{C}^\star)$: The intersections of the maximal codewords are 23, 34, 45, 13, 14, 1, 3, 4, and $\varnothing$, each of which – except 1 – is a codeword of $\mathcal{C}^\star$. However, 1 is not mandatory, as $\Delta(\mathcal{C}^\star)$ has facets $2345,\ 123,\ 134$, and 145, and so the link of 1 in

$\Delta(\mathcal{C}^\star)$ is the following path graph, which is contractible: 

Despite containing all its mandatory codewords, $\mathcal{C}^\star$ is nonconvex [12].

2.3 Neural Ideals

It is also possible to represent a neural code algebraically, as a specific ideal called the *neural ideal*, which was first used in [2]. To introduce this ideal, we first give a definition for the complement of a neural code:

Definition 2.3.1. Let $\mathcal{C}$ be a code on n neurons. The *complement* of a code $\mathcal{C}$ on n neurons is the code

$$\mathcal{C}' := 2^{[n]} \setminus \mathcal{C}. \tag{2.1}$$

Remark 2.3.2. To make $\mathcal{C}'$ well-defined, we require that each neuron $1, \ldots, n$ appears in at least one codeword of $\mathcal{C}$. For example, if $\mathcal{C} = \{\varnothing, 1, 2, 23\}$, we must consider it to be a code on 3 neurons, and not a code on $n \geq 4$ neurons, with neurons $4, \ldots, n$ trivial.

The neural ideal is a type of ideal known as a *pseudomonomial ideal*. We denote by $\mathbb{F}_2$ the field with two elements, and let $R = \mathbb{F}_2[x_1, \ldots, x_n] = \mathbb{F}_2[x]$.

Definition 2.3.3.

- A *pseudomonomial* is a polynomial $\prod_{i\in\sigma} x_i \prod_{j\in\tau}(1 - x_j) \in R$, where $\sigma, \tau \subseteq [n]$ are disjoint. A *pseudomonomial ideal* is an ideal generated by pseudomonomials.

- If $c \in 2^{[n]}$, the pseudomonomial

$$\phi_c := \prod_{i\in c} x_i \prod_{j\in[n]\setminus c} (1 - x_j) \tag{2.2}$$

is called the *indicator polynomial* of c.

Recall from Remark 2.1.2 that we may consider the codeword c as a binary string of length n. If we think of the binary strings as the elements of $\mathbb{F}_2^n$, then the indicator polynomial ϕ_c is the unique polynomial $\phi \in R$ satisfying

$$\phi(d) = \begin{cases} 1 & \text{if } c = d \\ 0 & \text{if } c \neq d. \end{cases}$$

Definition 2.3.4. The *neural ideal* $J_{\mathcal{C}}$ of a code $\mathcal{C}$ is the (pseudomonomial) ideal generated by the indicator polynomials of its non-codewords; in symbols,

$$J_{\mathcal{C}} := \langle \phi_c \mid c \in \mathcal{C}' \rangle.$$

Note that, since the generators of $J_{\mathcal{C}}$ are precisely those indicator polynomials for all $c \notin \mathcal{C}$ the zero-set of $J_{\mathcal{C}}$ is $\mathcal{C}$. While $J_{\mathcal{C}}$ is not the vanishing ideal $I_{\mathcal{C}}$ of $\mathcal{C}$, the two ideals have a close relationship. If $\mathcal{B} \subseteq R$ is the ideal generated by the Boolean relations $x_i^2 - x_i$, $i = 1, \dots, n$, then $I_{\mathcal{C}} = J_{\mathcal{C}} + \mathcal{B}$ [2, Lemma 3.2]. However, since $x_i^2 - x_i$ will vanish on any element of $\mathbb{F}_2^n$, $\mathcal{B} \subseteq I_{\mathcal{C}}$ for any code $\mathcal{C}$. Thus we may think of $J_{\mathcal{C}}$ as the "nontrivial" part of the vanishing ideal of $\mathcal{C}$. Furthermore, since there is a one-to-one correspondence between $\mathcal{C}$ and $J_{\mathcal{C}}$, the code and its neural ideal contain the same information.

Remark 2.3.5. The set of pseudomonomial ideals and the set of neural ideals are the same. It is clear from the definition that every neural ideal is a pseudomonomial ideal, and for any pseudomonomial ideal J one can find a code $\mathcal{C}$ for which $J = J_{\mathcal{C}}$ [13, Theorem 2.1].

When describing a neural ideal, we frequently describe it using a specific generating set called the *canonical form*:

Definition 2.3.6. Let $J \subseteq R$ be a pseudomonomial ideal.

- A pseudomonomial in J is *minimal* if it is minimal with respect to divisibility among all pseudomonomials in J.
- The *canonical form* of J is the set $\mathrm{CF}(J)$ of all minimal pseudomonomials of J.

The canonical form of a pseudomonomial ideal is a generating set for the ideal [2]. However, it is important to note that in general, the canonical form and the generating set of indicator polynomials from Definition 2.3.4 are not the same.

Example 2.3.7. Consider the code $\mathcal{C} = \{\varnothing, 2, 3, 12, 13\}$, which can be alternately represented as $\{000, 010, 001, 110, 101\}$ with the codewords as binary strings. The complement code is $\mathcal{C}' = \{1, 23, 123\}$ ($\{100, 011, 111\}$ as binary strings). Thus, the neural ideal of $\mathcal{C}$ is

$$J_{\mathcal{C}} = \langle x_1(1-x_2)(1-x_3),\ x_2x_3(1-x_1),\ x_1x_2x_3\rangle,$$

and the canonical form is $\mathrm{CF}(J_{\mathcal{C}}) = \{x_1(1-x_2)(1-x_3),\ x_2x_3\}$.

The neural ideal $J_{\mathcal{C}}$ has a unique irredundant decomposition

$$J_{\mathcal{C}} = \bigcap_{h=1}^{g} P_h, \tag{2.3}$$

where each P_h is a pseudomonomial ideal that is prime [2, Proposition 6.8]. In particular, $J_{\mathcal{C}}$ is a radical ideal. We remark that a pseudomonomial ideal P is prime if and only if it is of the form

$$P = \langle \{x_i \mid i \in \sigma\} \cup \{(1-x_j) \mid j \in \tau\}\rangle \quad \text{for } \sigma, \tau \text{ disjoint subsets of } [n]. \tag{2.4}$$

2.4 Boolean Intervals

Pseudomonomials are significant in that they represent Boolean intervals of $2^{[n]}$. Given $c \subseteq d \subseteq [n]$, the *Boolean interval* between c and d is

$$[c, d] := \{w \in 2^{[n]} \mid c \subseteq w \subseteq d\}.$$

Definition 2.4.1. Let $\mathcal{C}$ be a code. The *intervals* of $\mathcal{C}$ are the Boolean intervals contained in $\mathcal{C}$. The *maximal intervals* of $\mathcal{C}$ are the intervals of $\mathcal{C}$ that are maximal with respect to inclusion.

Example 2.4.2 (Example 2.3.7, continued)**.** For the code $\mathcal{C} = \{\varnothing, 2, 3, 12, 13\}$, the maximal intervals are $[\varnothing, 2]$, $[\varnothing, 3]$, $[2, 12]$, and $[3, 13]$.

Given $\mathcal{C}$, the pseudomonomials of $J_{\mathcal{C}}$ tell us what the intervals of the complement code $\mathcal{C}'$ are. We have that $[c, d] \subseteq \mathcal{C}'$ if and only if $\prod_{i \in c} x_i \prod_{j \notin d}(1 - x_j) \in J_{\mathcal{C}}$ [2, Lemma 5.7].

2.5 Polarization and Squarefree Monomial Ideals

Let $S = \mathbb{F}_2[x_1, \ldots, x_n, y_1, \ldots, y_n] = \mathbb{F}_2[x, y]$.

The idea of using y_i to encode $1 - x_i$ is well known (see, for instance, [14, 15]). In the context of neural ideals, the following construction was introduced in [16].

Definition 2.5.1.

- The *polarization* of a pseudomonomial $\phi = \prod_{i \in \sigma} x_i \prod_{j \in \tau}(1 - x_j) \in R$ is

$$\mathcal{P}(\phi) := \prod_{i \in \sigma} x_i \prod_{j \in \tau} y_j \in S.$$

- If $J \subseteq R$ is a pseudomonomial ideal, the *polarization* of J is the ideal in S obtained by polarizing the pseudomonomials in the canonical form of J, that is,

$$\mathcal{P}(J) := \langle \mathcal{P}(\phi) \mid \phi \in \mathrm{CF}(J) \rangle \subseteq S.$$

Note that the polarization of a pseudomonomial ideal is a *squarefree* monomial ideal in S, that is, an ideal generated by monomials that are not divisible by the squares of the variables (so $\mathcal{P}(J)$ is radical). We recall the relationship between squarefree monomial ideals and simplicial complexes:

Definition 2.5.2. Let Δ be a simplicial complex on $[n]$, and let $\mathbb{K}$ be a field. The *Stanley–Reisner ideal* of Δ is

$$I(\Delta) := \langle \prod_{i \in \sigma} x_i \mid \sigma \notin \Delta \rangle \subseteq \mathbb{K}[x_1, \ldots, x_n].$$

The ideal $I(\Delta)$ is radical, with prime decomposition

$$I(\Delta) = \bigcap_{\sigma \in \text{Facets}(\Delta)} \langle x_i \mid i \notin \sigma \rangle. \tag{2.5}$$

It follows that Δ can be recovered from $I(\Delta)$. In fact, (2.5) can be used to conclude that any squarefree monomial ideal is the Stanley–Reisner ideal of some simplicial complex. It is a fact that $I(\Delta(\mathcal{C}))$ is generated by the monomials in $\mathrm{CF}(J_{\mathcal{C}})$ [2, Lemma 4.4].

Example 2.5.3 (Example 2.4.2, continued)**.** For $\mathcal{C} = \{\varnothing, 2, 3, 12, 13\}$, the simplicial complex $\Delta(\mathcal{C})$ has two facets, 12 and 13. The corresponding Stanley–Reisner ideal is $I(\Delta(\mathcal{C})) = \langle x_2x_3 \rangle$, which is generated by the unique monomial in the canonical form $\mathrm{CF}(J_{\mathcal{C}}) = \{x_1(1-x_2)(1-x_3),\ x_2x_3\}$.

Remark 2.5.4. As noted above, the ideals that are associated to codes (the neural ideal $J_{\mathcal{C}}$, the ideal $I(\Delta(\mathcal{C}))$, and the factor ideal $\mathrm{FI}(\mathcal{C})$, to be introduced in Chapter 4) are *radical ideals*, that is, they can be expressed as intersections of prime ideals. We emphasize that the sets of associated primes, minimal primes, and primary components of a radical ideal all coincide.

However, it is important to note that there is in general not a one-to-one correspondence between the prime ideals of a pseudomonomial ideal J and its polarization $\mathcal{P}(J)$. This is because while a polarized ideal could have a prime containing both x_i and y_i, the "depolarized" counterpart to this prime would be $\langle 1 \rangle$, as it would contain both x_i and $1 - x_i$. As an example, consider $J = \langle x_1x_2, (1 - x_1)x_3 \rangle$, whose polarization is $\mathcal{P}(J) = \langle x_1x_2, x_3y_1 \rangle$. We have that the minimal primes of $\mathcal{P}(J)$ are $\langle x_1, x_3 \rangle$, $\langle x_1, y_1 \rangle$, $\langle x_2, x_3 \rangle$, and $\langle x_2, y_1 \rangle$, but J has only three minimal primes: $\langle x_1, x_3 \rangle$, $\langle x_2, x_3 \rangle$, and $\langle x_2, (1 - x_1) \rangle$,

We conclude the section on polarization with a remark on notation. When constructing the Stanley-Reisner complexes of squarefree monomial ideals in $S = \mathbb{F}_2[x, y]$, we will use $\{1, \ldots, n, \overline{1}, \ldots, \overline{n}\}$ as a vertex set, with the understanding that x_i corresponds to i and y_i corresponds to $\overline{i}$. If $B \subseteq [n]$, we denote $\overline{B} := \{\overline{i} \mid i \in B\}$. In particular,

$$\overline{[n]} = \{\overline{1}, \ldots, \overline{n}\} \quad \text{and} \quad [n] \cup \overline{[n]} = \{1, \ldots, n, \overline{1}, \ldots, \overline{n}\}.$$

We always use overline notation to denote subsets of $\overline{[n]}$; this is justified, as any subset of $\overline{[n]}$ is of the form $\overline{B}$ for some $B \subseteq [n]$.

3. REDUCIBLE, DECOMPOSABLE, AND PURE CODES

In this chapter, we outline three families of neural codes. In Section 3.1, we review the family of reducible codes introduced in [7]. In Section 3.2 we present a new family, called the decomposable codes, and use results from [10] to show how one can analyze them for convexity. Both the reducible and the decomposable families can be thought of as codes with extraneous structure. That is, a realization of such a code $\mathcal{C}$ can be obtained by building up from a realization of a simpler code $\mathcal{D}$. Thus, the question of convexity for $\mathcal{C}$ on n neurons can be answered by solving the question for a code $\mathcal{D}$ on m neurons, with $m \leq n$. Lastly, Section 3.3 covers the family of pure neural codes, those codes for which the maximal codewords have k many neurons.

In what follows, we use the following definition: let $\mathcal{C}$ be a code on n neurons, and σ a subset of $[n]$. The *restricted code* $\mathcal{C}|_\chi$ is $\{c \cap \chi \mid c \in \mathcal{C}\}$. If $\mathcal{U} = \{U_1, \ldots, U_n\}$ is a realization of $\mathcal{C}$, then we let $\mathcal{U}|_\chi$ denote the realization $\{U_i\}_{i\in\chi}$ of $\mathcal{C}|_\chi$.

Lemma 3.0.1. *Let $\mathcal{C}$ be a code on n neurons, and let $\chi \subseteq [n]$. If $\mathcal{C}$ is convex, then the restricted code $\mathcal{C}|_\chi$ is also convex.*

Proof. If $\{U_i\}_{i\in[n]}$ is a convex realization of $\mathcal{C}$, then $\{U_i\}_{i\in\chi}$ is a convex realization of $\mathcal{C}|_\chi$. □

3.1 Reducible Codes

The following terminology, introduced by Jeffs [7, §3], captures the case of a superfluous neuron. From [7] we have the following definition:

Definition 3.1.1. Let $\mathcal{C}$ be a code on n neurons, and let $\sigma \subseteq [n]$. The *trunk* $\mathrm{Tk}_\mathcal{C}(\sigma)$ of σ in $\mathcal{C}$ is the set of all codewords containing σ: $\mathrm{Tk}_\mathcal{C}(\sigma) := \{c \in \mathcal{C} \mid \sigma \subseteq c\}$.

From the trunk we can then define what it means for a neuron to be redundant, and a code to be reducible.

Definition 3.1.2. Let $\mathcal{C}$ be a code on n neurons.

(1) A neuron i is *redundant* in $\mathcal{C}$ if there exists $\sigma \subseteq [n]$ with $i \notin \sigma$ such that $\text{Tk}(\{i\}) = \text{Tk}(\sigma)$.

(2) A neuron i is *trivial* in $\mathcal{C}$ if i is not in any codeword of $\mathcal{C}$, that is, $\text{Tk}_{\mathcal{C}}(\{i\}) = \varnothing$.

(3) The code $\mathcal{C}$ is *reduced* if it has no redundant neurons and no trivial neurons.

We say that a code is *reducible* if it is not reduced.

The following result follows directly from [7, Theorem 1.3 and Lemma 3.11]:

Proposition 3.1.3. *Let $\mathcal{C}$ be a code on n neurons. Assume j is a redundant neuron of $\mathcal{C}$. Then $\mathcal{C}$ is convex if and only if the restricted code $\mathcal{C}|_{[n]\setminus\{j\}}$ is convex.*

Concretely, Proposition 3.1.3 is proven as follows. The forward implication is Lemma 3.0.1. Conversely, one obtains a convex realization of $\mathcal{C}$ from a convex realization of $\mathcal{C}|_{[n]\setminus\{j\}}$ by setting $U_j := U_\sigma$ (where $\text{Tk}(\{i\}) = \text{Tk}(\sigma)$). Essentially, the intuition for a redundant neuron is that its receptive field will always coincide with an intersection of receptive fields of some other neurons.

3.2 Decomposable Codes

The reducible codes introduce an extraneous neuron i in that the boundary of its receptive field U_i is contained in the boundaries of the other receptive fields. However, what about the case where the boundary of the receptive field does not intersect any of the other receptive fields' boundaries?

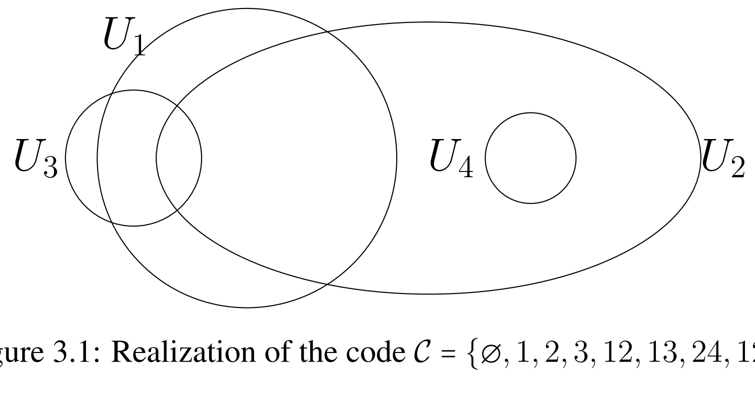


Figure 3.1: Realization of the code $\mathcal{C} = \{\varnothing, 1, 2, 3, 12, 13, 24, 123\}$

As an example, consider the code $\mathcal{C} = \{\varnothing, 1, 2, 3, 12, 13, 24, 123\}$, a realization of which is given in Figure 3.1. Note that the boundary of U_4 does not intersect the boundary of any other U_i. Indeed, 4 is not a redundant neuron, since $\text{Tk}_{\mathcal{C}}(4) = \{24\}$, and the only $\sigma \in \Delta(\mathcal{C})$ for which $\text{Tk}_{\mathcal{C}}(\sigma) = \{24\}$ is $\sigma = 24$, but $4 \in \sigma$. While $\mathcal{C}$ is not a reduced code, one can construct a realization of $\mathcal{C}$ by taking a realization of $\{\varnothing, 1, 2, 3, 12, 13, 123\}$ and placing U_4 in the interior of U_2. It is this type of simplification that we are generalizing in introducing the family of decomposable codes. We call such a code $\mathcal{C}$ decomposable since the idea is to decompose $\mathcal{C}$ into two smaller codes $\mathcal{C}_1$ and $\mathcal{C}_2$. Thus, a convex realization of $\mathcal{C}$, if it exists, can be obtained by "embedding" a convex realization of $\mathcal{C}_1$ into that of $\mathcal{C}_2$ (see Theorem 3.2.4). Accordingly, we introduce the following definition.

Definition 3.2.1. Let $\mathcal{C}$ be a code on n neurons. Then $\mathcal{C}$ is *decomposable* if there exist disjoint subsets $\varphi, \psi \subsetneq [n]$ with $\varphi \neq \varnothing$ such that:

(i) $\psi \in \mathcal{C}$, and

(ii) every $c \in \mathcal{C}$ that contains at least one neuron of φ has the form $c = \widetilde{\varphi} \cup \psi$ for some $\widetilde{\varphi} \subseteq \varphi$.

We call $\mathcal{C}|_{\varphi}$ the *embedded code* and $\mathcal{C}|_{[n]\smallsetminus\varphi}$ the *ambient code*. Also, ψ is the *ambient codeword.*

Example 3.2.2. The code $\mathcal{C}$ with a realization given in Figure 3.1 is decomposable, with $\phi = \{4\}$ and $\psi = \{2\}$.

Example 3.2.3. The code $\mathcal{C} = \{\mathbf{2356}, \mathbf{123}, 235, 236, 12, 14, 23, 1, 2, 4, \varnothing\}$ from Example 2.1.6 is decomposable, where $\varphi = \{5, 6\}$, $\psi = \{2, 3\}$. Indeed, the codewords $c \in \mathcal{C}$ intersecting φ are $235 = \{5\} \cup \psi$, $236 = \{6\} \cup \psi$, and $2356 = \{5, 6\} \cup \psi$. Recall the realization $\mathcal{U}$ of $\mathcal{C}$ depicted in Figure 2.1.

Note that $\mathcal{U}$ can be obtained by first drawing a realization of $\mathcal{C}|_{[n]\smallsetminus\varphi} = \{\varnothing, 1, 2, 4, 12, 14, 23, 123\}$ and then placing a realization of $\mathcal{C}|_{\varphi} = \{\varnothing, 5, 6, 56\}$ inside the atom of the ambient codeword 23.

We saw in Example 3.2.3 that a realization of $\mathcal{C}$ was obtained by placing a realization of the embedded code inside the atom of the ambient codeword. We also saw in Figure 3.1 the visual intuition into how a decomposable code can be broken down into or constructed from its embedded code and ambient code. However, there is an important concern over whether we can embed the embedded code in the ambient code. Given the ambient codeword ψ, is it true that there will always be some realization of the ambient code for which the atom of the ambient codeword $\mathcal{A}_\psi$ has nonempty interior? While in general this is not known, we do know the answer for codes with 6 or fewer neurons. Theorem 3.2.4, the main result of this section, states that for a decomposable code on up to 6 neurons, there exists a realization where the embedded code can be embedded in the atom of the ambient codeword. As this work on decomposable codes was undertaken to better understand the results of Chapter 5 on 6-neuron codes, we halted our efforts after obtaining Theorem 3.2.4.

Theorem 3.2.4. *Suppose that $\mathcal{C}$ is a decomposable code on up to 6 neurons with embedded code $\mathcal{C}|_\varphi$ and ambient code $\mathcal{C}|_{[n]\smallsetminus\varphi}$. Then $\mathcal{C}$ is convex if and only if $\mathcal{C}|_\varphi$ and $\mathcal{C}|_{[n]\smallsetminus\varphi}$ are convex.*

The proof of Theorem 3.2.4, which appears at the end of this section, requires us to know the idea of nondegeneracy introduced by Cruz *et al.* [10, §2]. Indeed, condition (i) of the following definition will guarantee that, for a decomposable code, the atom of the ambient codeword ψ has full dimension, and hence we can place within it a realization of the embedded code.

Definition 3.2.5. [10, Definition 2.10] For a collection $\mathcal{U} = \{U_i\}_{i=1}^n$ of subsets of $\mathbb{R}^d$, consider the following properties:

(i) For all $\sigma \subseteq [n]$, the set $(\cap_{i\in\sigma} U_i) \smallsetminus (\cup_{j\in[n]\smallsetminus c} U_j)$ is either empty or *top-dimensional*, i.e., every nonempty intersection with an open subset of $\mathbb{R}^d$ has nonempty interior.

(ii) For all nonempty $\sigma \subseteq [n]$, we have $\bigcap_{i\in\sigma} \partial U_i \subseteq \partial(\bigcap_{i\in\sigma} U_i)$.

Then $\mathcal{U}$ is *top-dimensional* if condition (i) holds, and is *nondegenerate* if both (i) and (ii) hold.

We say that a code $\mathcal{C}$ is *top-dimensionally convex* (respectively, *nondegenerately convex*) if it has a convex realization $\mathcal{U} = \{U_i\}_{i=1}^n$ that is top-dimensional (respectively, nondegenerate).

Proposition 3.2.6. *Suppose that $\mathcal{C}$ is a decomposable code with embedded code $\mathcal{C}|_\varphi$ and ambient code $\mathcal{C}|_{[n]\smallsetminus\varphi}$. If $\mathcal{C}|_\varphi$ is top-dimensionally convex and $\mathcal{C}|_{[n]\smallsetminus\varphi}$ is convex, then $\mathcal{C}$ is convex.*

Proof. Assume that $\mathcal{C}|_\varphi$ is top-dimensionally convex and $\mathcal{C}|_{[n]\smallsetminus\varphi}$ is convex. Relabel the neurons so that $\varphi = \{1,2,\ldots,k\}$ and $[n]\smallsetminus\varphi = \{k+1,k+2,\ldots,n\}$. Let $\mathcal{V} = \{V_i\}_{i=1}^k$ be a convex realization for $\mathcal{C}|_\varphi$ and $\mathcal{W} = \{W_i\}_{i=k+1}^n$ a top-dimensional convex realization for $\mathcal{C}|_{[n]\smallsetminus\varphi}$. We may assume that both $\mathcal{V}$ and $\mathcal{W}$ are realizations in $\mathbb{R}^d$, for some d. (Indeed, every realization $\mathcal{U}$ in some $\mathbb{R}^{d_1}$ can be elevated to a realization in $\mathbb{R}^{d_1+d_2}$ by taking the product of each U_i in $\mathcal{U}$ with $(0,1)^{d_2}$.)

As $\mathcal{W}$ is top-dimensional and ψ is a codeword of $\mathcal{C}$, there exists an open ball B in $\mathbb{R}^d$ that is strictly contained in the atom for the codeword ψ. Also, recall that we may assume that every V_i in the realization $\mathcal{V}$ is contained in an open ball $\widetilde{B}$ in $\mathbb{R}^d$ (cf. [8, Remark 2.19]). Let $F : \widetilde{B} \to B$ be the scaling bijection between the two balls (which preserves convexity and is a homeomorphism).

By construction, $\{F(V_i)\}_{i=1}^k \cup \{W_i\}_{i=k+1}^n$ is a convex realization of $\mathcal{C}$. □

In light of Proposition 3.2.6, we need only show that convexity and top-dimensional convexity are equivalent for codes on up to 5 neurons (see Proposition 3.2.12 below), and then Theorem 3.2.4 will follow. To prove Proposition 3.2.12, we need the following two results on nondegenerate convexity, the first of which follows directly from the results of Cruz *et al.* [10, Proposition 4.3 and Lemma 3.1]:

Lemma 3.2.7. *Every max-intersection-complete code is nondegenerately convex.*

Proof. [10, Proposition 4.3 and Lemma 3.1] give an explicit construction of an open convex realization of a max-intersection-complete code. One can verify that this realization is also nondegenerate. □

The following result is also, in essence, due to Cruz *et al.* [10]:

Lemma 3.2.8 (Monotonicity of top-dimensional and nondegenerate convexity)**.** *Let $\mathcal{C}$ be a code on n neurons, and let $\mathcal{D}$ be a code such that $\mathcal{C} \subseteq \mathcal{D} \subseteq \Delta(\mathcal{C})$.*

(i) If $\mathcal{C}$ is top-dimensionally convex, then $\mathcal{D}$ is also top-dimensionally convex.

(ii) If $\mathcal{C}$ is nondegenerately convex, then $\mathcal{D}$ is also nondegenerately convex.

Proof. Both (i) and (ii) follow directly from the proof of Theorem [10, Theorem 1.3], if we can add the following assertion to the statement of [10, Lemma 3.1]: "Also, if $\mathcal{U}$ is top-dimensional, then $\mathcal{V}$ can also be chosen to be top-dimensional". Indeed, the proof of [10, Lemma 3.1] directly accommodates this additional assertion once we add the assertion (which is worded identically to the prior one) to [10, Lemma A.7]: "Also, if $\mathcal{U}$ is top-dimensional, then $\mathcal{V}$ can also be chosen to be top-dimensional." The proof of this assertion is achieved by starting from the proof of [10, Lemma A.7], replacing the two occurrences of the word "non-degenerate" with "top-dimensional" and then deleting the final sentence of that proof.

□

In light of Lemma 3.2.8, to show top-dimensional convexity for convex codes on up to 5 neurons, we can proceed one simplicial complex Δ at a time: it suffices to check that $\mathcal{C}_{\min}(\Delta)$, the minimal code of Δ, is top-dimensionally convex. (Recall Definition 2.2.4 for $\mathcal{C}_{\min}(\Delta)$.)

However, there is one simplicial complex for which a different approach must be taken, namely, the simplicial complex of the non-convex code $\mathcal{C}^\star$ (with no local obstructions) from Example 2.2.13:

$$\Delta(\mathcal{C}^\star) = \text{the simplicial complex with facets } 2345,\ 123,\ 134,\ 145\ . \tag{3.1}$$

Note that $\mathcal{C}_{\min}(\Delta(\mathcal{C}^\star)) = \mathcal{C}^\star$. Part (ii) of the next result clarifies which codes with simplicial complex (3.1) are convex, while part (i), which is due to Goldrup and Phillipson [17], pertains to all other simplicial complexes on 5 vertices.

Remark 3.2.9. The code $\mathcal{C}^\star$ has importance beyond classification of codes on 5 neurons. It is the code from which we will generalize the wheel object in Chapter 5.

Proposition 3.2.10 (Convexity of 5-neuron codes)**.** *Let Δ be a connected simplicial complex on 5 vertices.*

(i) *If Δ is non-isomorphic to the simplicial complex* (3.1)*, then the code $\mathcal{C}_{\min}(\Delta)$ is top-dimensionally convex.*

(ii) *If Δ is the simplicial complex* (3.1)*, then a code $\mathcal{C}$ with simplicial complex $\Delta(\mathcal{C}) = \Delta$ is top-dimensionally convex if and only if $\mathcal{C}$ contains at least one of the following codewords:* 1*,* 234*, and* 345*.*

Proof. (i) If $\mathcal{C}_{\min}(\Delta)$ is max-intersection-complete, the result follows from Lemma 3.2.7. By [17, Theorem 3.1] (in that article, an isomorphic copy of $\mathcal{C}_{\min}(\Delta(\mathcal{C}^{\star}))$ is given the name "C4"), all remaining minimal codes other than $\mathcal{C}_{\min}(\Delta(\mathcal{C}^{\star}))$ have convex realizations depicted in [17, Appendix B]. These realizations are easily seen to be top-dimensional.

(ii) We first prove $\Leftarrow$. The code $\mathcal{C}^{\star} \cup \{1\}$ is max-intersection-complete and thus, by Lemma 3.2.8, is nondegenerately convex. The code $\mathcal{C}^{\star} \cup \{234\}$ is top-dimensionally convex: such a realization will appear in the forthcoming work of Magaña and Phillipson [18]. The code $\mathcal{C}^{\star} \cup \{345\}$ is also top-dimensionally convex: after relabeling the neurons via the permutation $(25)(34)$, the resulting code is the earlier code $\mathcal{C}^{\star} \cup \{234\}$. Thus, by Lemma 3.2.8(i), every code $\mathcal{C}$ with simplicial complex $\Delta(\mathcal{C}) = \Delta$ that contains 1, 234, or 345 is convex.

Finally, the contrapositive of $\Rightarrow$ follows directly from Proposition 5.2.13. $\square$

Remark 3.2.11. Proposition 3.2.10(ii) implies that $\mathcal{C}^{\star} \cup \{1\}$, $\mathcal{C}^{\star} \cup \{234\}$, and $\mathcal{C}^{\star} \cup \{345\}$ are the minimal (with respect to inclusion) convex codes with neural complex equal to $\Delta(\mathcal{C}^{\star})$. This corrects an error in [12, Remark 3.5], where it was asserted that $\mathcal{C}^{\star} \cup \{234, 345\}$ is such a minimal convex code.

Next, we show that convexity and top-dimensional convexity are equivalent for codes on up to 5 neurons.

Proposition 3.2.12. *Let $\mathcal{C}$ be a code on 5 or fewer neurons. Then $\mathcal{C}$ is convex if and only if $\mathcal{C}$ is top-dimensionally convex.*

Proof. The direction $\Leftarrow$ holds by definition. For $\Rightarrow$, let $\mathcal{C}$ be a convex code on n neurons, where $n \leq 5$. If $n \leq 4$, then [9, Proposition 1.7], which states that a code with $n \leq 4$ neurons is max-intersection-complete if and only if it is convex, and Lemma 3.2.7 imply that $\mathcal{C}$ is nondegenerately convex.

Now assume $n = 5$. If the neural complex $\Delta(\mathcal{C})$ is non-isomorphic to $\Delta(\mathcal{C}^{\star})$, the simplicial complex shown in (3.1), then the result follows from Proposition 3.2.10(i) and Proposition 3.2.7(i). The only remaining case, therefore, is when $\Delta(\mathcal{C})$ is isomorphic to $\Delta(\mathcal{C}^{\star})$. By relabeling neurons, if necessary, we may assume that $\Delta(\mathcal{C}) = \Delta(\mathcal{C}^{\star})$. As $\mathcal{C}$ is convex, then by Proposition 3.2.10((ii)), the code $\mathcal{C}$ contains at least one of the codewords 1, 234, and 245. We saw in the proof of Proposition 3.2.10((ii)) that $\mathcal{C}$ is top-dimensionally convex. This completes the proof. □

We can now prove our main result.

Proof of Theorem 3.2.4. The implication ⇒ follows from Lemma 3.0.1, and ⇐ is immediate from Propositions 3.2.6 and 3.2.12.

□

3.3 Pure Neural Codes

We now turn to the family of pure neural codes. Unlike the reducible or decomposable codes, the pure codes are not instances of a simple code disguised with extra complexity, but rather an interesting variety of codes with properties not applicable to all codes.

Definition 3.3.1. A neural code $\mathcal{C}$ is *pure* if its neural complex $\Delta(\mathcal{C})$ is pure, that is, every facet of $\Delta(\mathcal{C})$ has the same dimension.

Next, we show that for pure codes of low or high dimension, being convex is equivalent to being max-intersection-complete, and thus is easy to check. For a discussion of the algorithmic aspects of checking whether a code is max-intersection-complete, see [19, §6].

Theorem 3.3.2. *Let $\mathcal{C}$ be a code on n neurons. If $\mathcal{C}$ is pure of dimension 0, 1, $n-2$, or $n-1$, then the following are equivalent:*

(i) *$\mathcal{C}$ is convex,*

(ii) *$\mathcal{C}$ has no local obstructions, and*

(iii) *$\mathcal{C}$ is max-intersection-complete.*

Proof. The implications (iii) ⇒ (i) ⇒ (ii) follow from Propositions 2.2.6 and 2.2.9.

The implication (ii) ⇒ (iii) holds for dimension 0 (trivially, as by assumption $\varnothing \in \mathcal{C}$), dimension 1 (this follows easily from [20, Theorem 1.3]), and dimension $n-1$ (by [9, Lemma 2.5]). Now assume that $\mathcal{C}$ is pure of dimension $n-2$ and has no local obstructions.

Let $F_1, F_2, \ldots, F_m$ be distinct facets of $\Delta(\mathcal{C})$ (with $m \geq 2$); we must show that $\sigma := \bigcap_{j=1}^{m} F_j$ is a codeword of $\mathcal{C}$. As $\mathcal{C}$ has dimension $n-2$, there exist distinct $i_1, i_2, \ldots, i_m$ such that $F_j = [n] \smallsetminus \{i_j\}$ (for every j). Thus, $\sigma = [n] \smallsetminus \{i_1, i_2, \ldots, i_m\}$.

We now claim that every facet of $\Delta(\mathcal{C})$ that contains σ is one of the F_j. Suppose F is such a facet.

As $\Delta(\mathcal{C})$ is pure, $F = [n] \smallsetminus \{i\}$ for some i. Then $i \notin \sigma$, and so $i \in \{i_1, i_2, \ldots, i_m\}$. Therefore, $i = i_j$ (for some $j = 1, \ldots, m$) and thus $F = F_j$.

The claim implies that the facets of $\mathrm{Lk}_\sigma(\Delta(\mathcal{C}))$ are precisely the sets

$$F_j \smallsetminus \sigma = \{i_1, i_2, \ldots, i_{j-1}, \hat{i}_j, i_{j+1}, \ldots, i_m\}, \text{ for } j = 1, \ldots, m.$$

Thus, $\mathrm{Lk}_\sigma(\Delta(\mathcal{C}))$ is the hollow simplex on m vertices (that is, it contains all possible faces on the m vertices except the top face) and so is not contractible (recall that $m \geq 2$). Thus, as $\mathcal{C}$ has no local obstructions by assumption, we have $\sigma \in \mathcal{C}$, and so (ii) ⇒ (iii) holds. □

The analogous result for pure codes of dimension 2 or $n-3$ (or any dimension in between) does not hold. In fact, we have already seen examples of such codes. One is the code from Example 2.2.12, which is a convex code on $n = 5$ neurons that fails to be max-intersection-complete; this code is pure and of dimension $n - 3 = 2$. Another example is the code in Example 5.2.10, which is pure of dimension 2, has no local obstructions, and is non-convex.

4. THE FACTOR COMPLEX AND MAX-INTERSECTION-COMPLETE CODES*

The fact that the neural ideal $J_{\mathcal{C}}$ of a code $\mathcal{C}$ retains all the information about $\mathcal{C}$ (due to the one-to-one correspondence between $J_{\mathcal{C}}$ and $\mathcal{C}$) has prompted investigations into how one can discover properties of $\mathcal{C}$ just from exmaining $J_{\mathcal{C}}$. In Curto et al., the authors discovered a signature for a code to be intersection-complete in $J_{\mathcal{C}}$ [21, Theorem 1.9]. They then asked whether there is such an algebraic signature for a code to be max-intersection-complete.

In this chapter we answer the question posed by the authors of [21]. Our main result, Theorem 4.0.1 below, gives a characterization for when a code is max-intersection-complete in terms of the canonical form of its neural ideal and the Stanley–Reisner ideal $I(\Delta(\mathcal{C}))$ of its simplicial complex $\Delta(\mathcal{C})$. Recall the definitions of max-intersection-completeness and intersection-completeness (Remark 2.2.5), the canonical form of a neural ideal (Definition 2.3.6), and the Stanley-Reisner ideal (Definition 2.5.2).

Theorem 4.0.1. *A code $\mathcal{C}$ on n neurons is max-intersection-complete if and only if for every non-monomial ϕ in the canonical form of the neural ideal of $\mathcal{C}$, there exists $i \in [n]$ such that*

(i) every associated prime of $I(\Delta(\mathcal{C}))$ that contains x_i also contains ϕ, and

(ii) $(1 - x_i) \mid \phi$.

The proof of Theorem 4.0.1 makes use of a new combinatorial object, the *factor complex* of a code. This factor complex, which is closely related to the *polar complex* introduced in [16], is a simplicial complex that, like the neural ideal but unlike $\Delta(\mathcal{C})$, captures all the combinatorial information in a code $\mathcal{C}$. The proof uses the factor complex as a stepping stone to the neural ideal. First, we show how to determine max-intersection-completeness of $\mathcal{C}$ from its corresponding factor complex. From there, we translate these conditions for max-intersection-completeness found in the factor complex into conditions found in the neural ideal. However, it is not required that

*Reprinted with permission from "Neural codes and the factor complex" by Alexander Ruys de Perez, Laura Matusevich, and Anne Shiu, 2020. *Advances in Applied Mathematics*, 114, 101977, Copyright 2019 Elsevier Inc.

one first understand the signature for max-intersection-completeness in the factor complex to be able to state the corresponding signature in the neural ideal. One may think of a neural code, its neural ideal, and its factor complex as three equivalent worlds for storing information about the neural code. We show how to translate a feature about the neural code, such as what the maximal codewords are, into the language of any of these three worlds. Therefore, as a bonus, we will also show how to determine whether a code is intersection-complete directly from its factor complex (Theorem 4.1.6). Indeed, we expect in the future that the factor complex may help us understand more properties of neural codes.

We finish this chapter by describing how Theorem 4.0.1 improves the computational time for determining whether a code is max-intersection-complete. There are two naive methods of determining max-intersection-completeness: (1) checking whether all intersections of maximal codewords are contained in $\mathcal{C}$ and (2) checking whether for each $\sigma \in \Delta(\mathcal{C}) \smallsetminus \mathcal{C}$, the intersection c_σ of all maximal codewords of $\mathcal{C}$ which contain σ, is not equal to σ. As the runtime of either method is exponential on the number of neurons, neither of these two methods is practical for analyzing codes on n neurons as n gets large. From Theorem 4.0.1 we now have a method to check the max-intersection-completeness of $\mathcal{C}$ from the canonical form of $J_{\mathcal{C}}$. This method is an improvement on the previously mentioned naive methods in that instead of checking each individual missing face of $\Delta(\mathcal{C})$, the canonical form method checks entire Boolean intervals of missing faces at once. However, there is a caveat: we assume that one knows the canonical form of $J_{\mathcal{C}}$, a piece of information that is not needed by the naive methods.

4.1 Main Results Concerning the Factor Complex

In this section we introduce a new combinatorial tool to study neural codes: the factor complex (Definition 4.1.1), and state our four main results. Theorems 4.1.3 and 4.1.4 summarize the relationships between codes, their factor complexes, and their related ideals (neural ideals and Stanley–Reisner ideals). These results are used to prove Theorems 4.1.6 and 4.1.7, which characterize intersection-complete codes and max-intersection-complete codes in two ways: combinatorially and algebraically.

Definition 4.1.1. Let $\mathcal{C}$ be a code on n neurons, and recall the primary decomposition of the neural ideal $J_{\mathcal{C}}$ given in Chapter 2. The *factor ideal* of $\mathcal{C}$ is obtained by polarizing the components of $J_{\mathcal{C}}$, namely,

$$\mathrm{FI}(\mathcal{C}) := \bigcap_{h=1}^{g} \mathcal{P}(P_h).$$

The *factor complex* $\Delta_{\cap}(\mathcal{C})$ of $\mathcal{C}$ is the simplicial complex on $[n] \cup \overline{[n]}$ whose Stanley–Reisner ideal is $\mathrm{FI}(\mathcal{C})$. (Recall Remark 2.5.4: $\mathrm{FI}(\mathcal{C}) \subseteq S = \mathbb{F}_2[x_1, \ldots, x_n, y_1, \ldots, y_n]$, while $\Delta_{\cap}(\mathcal{C})$ is on vertices $\{1, \ldots, n, \overline{1}, \ldots, \overline{n}\}$, with the correspondence being $x_i \leftrightarrow i$ and $y_i \leftrightarrow \overline{i}$.) A face of $\Delta_{\cap}(\mathcal{C})$ is *defective* if it contains neither i nor $\overline{i}$ for some $i \in [n]$ (we think of the lack of i and $\overline{i}$ as a defect, or flaw); faces that are not defective are called *effective*. We say that $\overline{B} \subseteq \overline{[n]}$ is a *prime-set* of $\Delta_{\cap}(\mathcal{C})$ if $[n] \cup \overline{B} \notin \Delta_{\cap}(\mathcal{C})$, and $\overline{B}$ is furthermore *minimal* if $\overline{B}$ is minimal with respect to inclusion among prime-sets. Lemma 4.2.5 gives the reason why we chose this terminology.

Example 4.1.2 (Example 2.5.3, continued)**.** For $\mathcal{C}' = \{1, 23, 123\}$, the neural ideal decomposes as follows:

$$J_{\mathcal{C}'} = \langle (1-x_1)(1-x_3),\ (1-x_1)(1-x_2),\ x_2(1-x_3),\ x_3(1-x_2) \rangle = \langle x_2, x_3,\ 1-x_1 \rangle \cap \langle 1-x_2,\ 1-x_3 \rangle.$$

The factor ideal is therefore

$$FI(\mathcal{C}') = \langle x_2, x_3, y_1 \rangle \cap \langle y_2, y_3 \rangle,$$

and so the two facets of the factor complex $\Delta_{\cap}(\mathcal{C}')$ are $1\overline{2}\overline{3}$ and $123\overline{1}$ (both are effective). The minimal prime-sets of $\Delta_{\cap}(\mathcal{C}')$ are $\{\overline{2}\}$ and $\{\overline{3}\}$.

Theorem 4.1.3 (Codes, factor complexes, and neural ideals)**.** *Let $\mathcal{C}$ be a code on n neurons, and $\mathcal{C}'$ its complement code defined in* (2.1). *The following two maps are bijections:*

$$\begin{array}{ccccc} \{\text{pseudomonomials in } J_{\mathcal{C}'}\} & \leftarrow & \{\text{intervals in } \mathcal{C}\} & \rightarrow & \{\text{effective faces of } \Delta_{\cap}(\mathcal{C})\} \\ \prod_{i \in c} x_i \prod_{j \in [n] \smallsetminus d} (1 - x_j) & \leftarrow\!\shortmid & [c, d] & \mapsto & d \cup \overline{[n] \smallsetminus c} \end{array}$$

Moreover, every facet of $\Delta_{\cap}(\mathcal{C})$ is effective, and the following are equivalent:

(1) $[c,d]$ *is a maximal interval in* $\mathcal{C}$*,*

(2) $\prod_{i\in c} x_i \prod_{j\in[n]\setminus d}(1-x_j) \in \mathrm{CF}(J_{\mathcal{C}'})$*, and*

(3) $d \cup \overline{[n]\setminus c}$ *is a facet of* $\Delta_\cap(\mathcal{C})$*.*

Theorem 4.1.4 (Codes, factor complexes, and Stanley–Reisner ideals)**.** *Let* $\mathcal{C}$ *be a code on* n *neurons, with complement code* $\mathcal{C}'$ *and factor complex* $\Delta_\cap(\mathcal{C})$*. The following two maps are bijections:*

$$\begin{array}{ccccc} \{\text{minimal primes of } I(\Delta(\mathcal{C}))\} & \leftarrow & \{\text{maximal codewords of } \mathcal{C}\} & \rightarrow & \left\{\begin{array}{c}\text{minimal prime-sets} \\ \text{of } \Delta_\cap(\mathcal{C}')\end{array}\right\} \\ \langle x_i \mid i\in[n]\setminus M\rangle & \leftarrow\!\shortmid & M & \mapsto & \overline{[n]\setminus M} \end{array}$$

The proofs of Theorems 4.1.3 and 4.1.4 are postponed until Sections 4.2.1 and 4.2.2, respectively.

Example 4.1.5 (Example 4.1.2, continued)**.** According to Theorem 4.1.3, the facets $1\overline{23}$ and $123\overline{1}$ of $\Delta_\cap(\mathcal{C}')$ correspond to the two maximal intervals of $\mathcal{C}'$, $[1,1]$ and $[23,123]$, respectively, and also to the two pseudomonomials in $\mathrm{CF}(J_{\mathcal{C}})$, namely, $x_1(1-x_2)(1-x_3)$ and x_2x_3, respectively.

Similarly, Theorem 4.1.4 implies that the minimal prime-sets $\{\overline{2}\}$ and $\{\overline{3}\}$ of $\Delta_\cap(\mathcal{C}')$ correspond to the minimal primes $\langle x_2\rangle$ and $\langle x_3\rangle$ of $I(\Delta(\mathcal{C})) = \langle x_2x_3\rangle$ and also to the maximal codewords 13 and 12 of $\mathcal{C}$.

The following result translates the algebraic characterization of intersection-complete codes from [21] into a new combinatorial criterion.

Theorem 4.1.6 (Intersection-complete codes)**.** *Let* $\mathcal{C}$ *be a code on* n *neurons with neural ideal* $J_{\mathcal{C}}$*, and let* $\mathcal{C}'$ *be the complement code of* $\mathcal{C}$ *with factor complex* $\Delta_\cap(\mathcal{C}')$*. The following are equivalent:*

(1) $\mathcal{C}$ *is intersection-complete,*

(2) every pseudomonomial $\prod\limits_{i\in\sigma} x_i \prod\limits_{j\in\tau}(1-x_j)$ *in* $\mathrm{CF}(J_{\mathcal{C}})$ *satisfies* $|\tau|\le 1$*, and*

(3) every facet F *of* $\Delta_\cap(\mathcal{C}')$ *satisfies* $|F\cap[n]|\ge n-1$*.*

Proof. The equivalence between *4.1.6* and *4.1.6* is [21, Theorem 1.9]. By Theorem 4.1.3, $\prod_{i\in\sigma} x_i \prod_{j\in\tau}(1-x_j)$ belongs to the canonical form of $J_{\mathcal{C}}$ if and only if $F = [n] \smallsetminus \tau \cup \overline{[n] \smallsetminus \sigma}$ is a facet of $\Delta_{\cap}(\mathcal{C}')$. Thus, the condition $|\tau| \leq 1$ is equivalent to $|F \cap [n]| \geq n-1$, and so *4.1.6* is equivalent to *4.1.6*. □

The following result is an expanded version of Theorem 4.0.1.

Theorem 4.1.7 (Max-intersection-complete codes)**.** *Let $\mathcal{C}$ be a code on n neurons with neural ideal $J_{\mathcal{C}}$, and let $\mathcal{C}'$ be the complement code of $\mathcal{C}$ with factor complex $\Delta_{\cap}(\mathcal{C}')$. The following are equivalent:*

(1) $\mathcal{C}$ is max-intersection-complete,

(2) for every facet F of $\Delta_{\cap}(\mathcal{C}')$ that does not contain $[n]$, there exists $i \in [n]$ such that

(i) every minimal prime-set of $\Delta_{\cap}(\mathcal{C}')$ that contains $\bar{i}$ also contains some $\bar{j}$ such that $\bar{j} \notin F$, and

(ii) $i \notin F$,

(3) for every $\phi \in \mathrm{CF}(J_{\mathcal{C}})$ that is not a monomial, there exists $i \in [n]$ such that

(i) every minimal prime of $I(\Delta(\mathcal{C}))$ that contains x_i also contains ϕ, and

(ii) $(1-x_i) \mid \phi$.

Proof. We begin by proving *(2)⇔(3).* By Theorem 4.1.3, $\phi = \prod_{i\in c} x_i \prod_{j\in[n]\smallsetminus d}(1-x_j) \in \mathrm{CF}(J_{\mathcal{C}})$ if and only if $F = d \cup \overline{[n] \smallsetminus c}$ is a facet of $\Delta_{\cap}(\mathcal{C}')$. Furthermore, ϕ is a non-monomial exactly when $d \nsupseteq [n]$, if and only if F does not contain $[n]$. Thus, by inspection of ϕ and F, *(2)(i)* is equivalent to *(3)(i)*, and so we need only show *(2)(ii)⇔(3)(ii).*

By Theorem 4.1.4, the prime ideal $P = \langle x_j \mid j \in B \rangle$ is associated to $I(\Delta(\mathcal{C}))$ if and only if $\overline{B}$ is a minimal prime-set of $\Delta_{\cap}(\mathcal{C}')$. Thus, $x_i \in P$ exactly when $\bar{i} \in \overline{B}$. Next, it is straightforward to check that P contains $\phi = \prod_{i\in c} x_i \prod_{j\in[n]\smallsetminus d}(1-x_j)$ if and only if $B \cap c \neq \varnothing$. As ϕ corresponds to the facet $F = d \cup \overline{[n] \smallsetminus c}$ of $\Delta_{\cap}(\mathcal{C}')$, it follows that P contains ϕ if and only if $\bar{j} \notin F$ for some $\bar{j} \in \overline{B}$. This concludes the proof of *(2)⇔(3).*

We set up notation needed to prove *(1)⇔(2)*. Let $\overline{B}_1, \overline{B}_2, \dots, \overline{B}_u$ be the minimal prime-sets of $\Delta_\cap(\mathcal{C}')$. By Theorem 4.1.4, the maximal codewords of $\mathcal{C}$ are $m_1 = [n] \smallsetminus B_1, \dots, m_u = [n] \smallsetminus B_u$.

We claim that *(2)* is equivalent to the following:

(2') for every facet F of $\Delta_\cap(\mathcal{C}')$ that does not contain $[n]$,

$$([n] \smallsetminus \bigcup_{v \in H_F} B_v) \notin F, \tag{$\star$}$$

where

$$H_F := \{v \in [u] \mid \overline{B}_v \subseteq F\}.$$

Indeed, $(\star)$ states that there exists $i \in [n]$ such that $i \notin F$ and $\bar{i}$ is <u>not</u> in any minimal prime-set $\overline{B}_v \subseteq \{\bar{1}, \bar{2}, \dots, \bar{n}\}$ for which $\overline{B}_v \subseteq F$. This latter condition exactly matches *(2)(ii)*. Hence, our claim holds, and we may complete this proof by showing *(1)⇔(2')*.

($\Leftarrow$) We prove the contrapositive. Suppose that the intersection of maximal codewords $c = \bigcap_{v \in V} m_v$ (for some $\varnothing \neq V \subseteq [u]$) is not in $\mathcal{C}$, that is, $c \in \mathcal{C}'$. By Theorem 4.1.3, $c \cup \overline{[n] \smallsetminus c}$ is a face of $\Delta_\cap(\mathcal{C}')$. Note that

$$\overline{[n] \smallsetminus c} = \overline{[n] \smallsetminus \bigcap_{v \in V} m_v} = \bigcup_{v \in V} \overline{[n] \smallsetminus m_v} = \bigcup_{v \in V} \overline{B}_v. \tag{4.1}$$

Let F be a facet of $\Delta_\cap(\mathcal{C}')$ containing $c \cup \overline{[n] \smallsetminus c}$. It follows from (4.1) that F contains the union of minimal prime-sets $\bigcup_{v \in V} \overline{B}_v$, which implies that F does not contain $[n]$ (as, otherwise, each $\overline{B}_v \cup [n]$ is contained in F and hence is a face of $\Delta_\cap(\mathcal{C}')$, contradicting the fact that B_v is a prime-set). Since $F \supseteq \overline{[n] \smallsetminus c} = \bigcup_{v \in V} \overline{B}_v$, we have that $V \subseteq H_F$. Therefore, $[n] \smallsetminus \bigcup_{v \in H_F} B_v \subseteq [n] \smallsetminus \bigcup_{v \in V} B_v = c$, where the equality comes from (4.1). We conclude that F is a facet of $\Delta_\cap(\mathcal{C}')$ not containing $[n]$ such that $([n] \smallsetminus \bigcup_{v \in H_F} B_v) \subseteq c \subseteq (c \cup \overline{[n] \smallsetminus c}) \subseteq F$.

($\Rightarrow$) Suppose $\mathcal{C}$ is max-intersection-complete. Let F be a facet of $\Delta_\cap(\mathcal{C}')$ that does not contain $[n]$. Set $c := [n] \smallsetminus \bigcup_{v \in H_F} B_v$. Our goal is to show that $c \notin F$.

We accomplish this by proving two facts. First, that $c \cup \overline{[n] \smallsetminus c}$ is not a face of $\Delta_\cap(\mathcal{C}')$, and second, that $\overline{[n] \smallsetminus c} = \bigcup_{v \in H_F} \overline{B}_v$. The first fact implies that $c \cup \overline{[n] \smallsetminus c} \notin F$ and the second yields

$\overline{[n] \setminus c} \subseteq F$. Our desired relation $c \notin F$ will then follow.

For the first fact, recall that $[n] \setminus B_v = m_v$. Therefore,

$$c = [n] \setminus \bigcup_{v \in H_F} B_v = \bigcap_{v \in H_F} [n] \setminus B_v = \bigcap_{v \in H_F} m_v,$$

so c is the intersection of maximal codewords. As $\mathcal{C}$ is max-intersection-complete, $c \in \mathcal{C}$, and thus $c \notin \mathcal{C}'$. Now Theorem 4.1.3 implies that $c \cup \overline{[n] \setminus c} \notin \Delta_\cap(\mathcal{C}')$.

For the second fact, $\overline{[n] \setminus c} = \overline{[n] \setminus ([n] \setminus \bigcup_{v \in H_F} B_v)} = \overline{\bigcup_{v \in H_F} B_v} = \bigcup_{v \in H_F} \overline{B}_v$. □

Example 4.1.8 (Example 4.1.5, continued)**.** The code $\mathcal{C} = \{\varnothing, 2, 3, 12, 13\}$ is neither intersection-complete nor max-intersection-complete (as $1 = 12 \cap 13 \notin \mathcal{C}$). We can read this information from Theorems 4.1.6 and 4.1.7, as follows. For non-intersection-completeness, this can be seen in two ways: first, the pseudomonomial $x_1(1 - x_2)(1 - x_3)$ is in the canonical form of $J_\mathcal{C}$, and, second, the intersection of the facet $1\overline{23}$ with 123 has size 1, rather than 2 or 3.

For non-max-intersection-completeness, recall that the minimal prime-sets of $\Delta_\cap(\mathcal{C}')$ are $\{\overline{2}\}$ and $\{\overline{3}\}$ (equivalently, the minimal primes of $I(\Delta(\mathcal{C}))$ are $\langle x_2 \rangle$ and $\langle x_3 \rangle$). Now, $1\overline{23}$ is a facet of $\Delta_\cap(\mathcal{C}')$ that does not contain 123, but for $i \in \{1, 2, 3\}$, either part *(2)(ii)* of Theorem 4.1.7 is violated (when $i = 2, 3$) or part *(2)(i)* is violated (when $i = 1$). Alternatively, $\mathrm{CF}(J_\mathcal{C})$ contains the non-monomial $x_1(1-x_2)(1-x_3)$, but for $i \in \{1, 2, 3\}$, either part *(3)(ii)* of Theorem 4.1.7 is violated (when $i = 2, 3$) or part *(3)(i)* is violated (when $i = 1$). Thus, $\mathcal{C}$ is not max-intersection-complete.

4.2 Factor Complexes, Neural Ideals, and Codes

In this section, we prove Theorems 4.1.3 and 4.1.4.

4.2.1 Proof of Theorem 4.1.3

We wish to prove that the following maps are bijections:

$$\begin{array}{ccccc} \{\text{pseudomonomials in } J_{\mathcal{C}'}\} & \overset{\alpha}{\leftarrow} & \{\text{intervals in } \mathcal{C}\} & \overset{\beta}{\rightarrow} & \{\text{effective faces of } \Delta_\cap(\mathcal{C})\} \\ \prod_{i \in c} x_i \prod_{j \in [n] \setminus d} (1 - x_j) & \leftarrow\!\shortmid & [c, d] & \mapsto & d \cup \overline{[n] \setminus c} \end{array}$$

The fact that α is a bijection is straightforward from [2, Lemma 5.7]. To show that β is a bijection, we need to better understand the factor ideal and factor complex of $\mathcal{C}$.

Lemma 4.2.1. *Let $\mathcal{C}$ be a code with neural ideal $J_{\mathcal{C}}$, and let ϕ be a pseudomonomial. Then $\phi \in J_{\mathcal{C}}$ if and only if $\mathcal{P}(\phi) \in \mathrm{FI}(\mathcal{C})$.*

Proof. Recall the decomposition $J_{\mathcal{C}} = \bigcap_{h=1}^{g} P_h$ from (2.3). Hence, $\phi \in J_{\mathcal{C}}$ if and only if $\phi \in P_h$ for all h. Given the form (2.4) of each component P_h, it is straightforward to check that $\phi \in P_h$ is equivalent to $\mathcal{P}(\phi) \in \mathcal{P}(P_h)$. Thus, as $\mathrm{FI}(\mathcal{C}) = \bigcap \mathcal{P}(P_h)$, the desired result follows. $\square$

Our next results shows how to use the factor complex of a code to read off its codewords.

Lemma 4.2.2. *Let $\mathcal{C}$ be a code on n neurons. Then $c \in 2^{[n]}$ is a codeword of $\mathcal{C}$ if and only if $c \cup \overline{[n] \setminus c}$ is a face of $\Delta_{\cap}(\mathcal{C})$.*

Proof. By [2, Lemma 3.2], $c \in \mathcal{C}$ if and only if $\phi_c = \prod_{i \in c} x_i \prod_{j \notin c}(1 - x_j) \notin J_{\mathcal{C}}$. This is equivalent to $\mathcal{P}(\phi_c) \notin \mathrm{FI}(\mathcal{C})$ by Lemma 4.2.1. Since $\mathrm{FI}(\mathcal{C})$ is the Stanley–Reisner ideal of $\Delta_{\cap}(\mathcal{C})$, we have that $\mathcal{P}(\phi_c) \notin \mathrm{FI}(\mathcal{C})$ exactly when $c \cup \overline{[n] \setminus c}$ is a face of $\Delta_{\cap}(\mathcal{C})$, which concludes the proof. $\square$

We now extend Lemma 4.2.2 to show how to extract the intervals of $\mathcal{C}$ from its factor complex.

Lemma 4.2.3. *(Interval-Face Correspondence) Let $\mathcal{C}$ be a code on n neurons, and let $c, d \in 2^{[n]}$. Then $[c, d] \subseteq \mathcal{C}$ if and only if $d \cup \overline{[n] \setminus c}$ is a face of $\Delta_{\cap}(\mathcal{C})$.*

Proof. ($\Leftarrow$) Suppose $d \cup \overline{[n] \setminus c}$ is a face of $\Delta_{\cap}(\mathcal{C})$, and let $w \in [c, d]$. Then $w \cup \overline{[n] \setminus w} \subseteq d \cup \overline{[n] \setminus c}$ is a face of $\Delta_{\cap}(\mathcal{C})$ and thus $w \in \mathcal{C}$ by Lemma 4.2.2.

($\Rightarrow$) We now assume that $d \cup \overline{[n] \setminus c}$ is not a face of $\Delta_{\cap}(\mathcal{C})$ and show that $[c, d]$ is not an interval of $\mathcal{C}$. As $\mathrm{FI}(\mathcal{C})$ is the Stanley–Reisner ideal of $\Delta_{\cap}(\mathcal{C})$, the decomposition (2.5) implies that the ideal

$$\left\langle \{x_i \mid i \notin d \cup \overline{[n] \setminus c}\} \cup \{y_j \mid \overline{j} \notin d \cup \overline{[n] \setminus c}\} \right\rangle = \left\langle \{x_i \mid i \in [n] \setminus d\} \cup \{y_j \mid j \in c\} \right\rangle$$

is not associated to $\mathrm{FI}(\mathcal{C})$, and therefore the following ideal is not associated to $J_{\mathcal{C}}$:

$$\big\langle \{x_i \mid i \in [n] \smallsetminus d\} \cup \{(1-x_j) \mid j \in c\} \big\rangle. \tag{4.2}$$

Thus, as $\mathrm{CF}(J_{\mathcal{C}})$ is a generating set for $J_{\mathcal{C}}$, there exists a pseudomonomial $\phi = \prod_{i\in\sigma} x_i \prod_{j\in\tau}(1-x_j)$ in $\mathrm{CF}(J_{\mathcal{C}})$ that is not in the ideal (4.2), and so $\sigma \subseteq d$ and $\tau \subseteq [n] \smallsetminus c$. Note that the indicator pseudomonomial $\phi_{c\cup\sigma}$ is in $J_{\mathcal{C}}$, as it is divisible by ϕ. We conclude that $\sigma \cup c \in [c,d] \smallsetminus \mathcal{C}$, and so $[c,d] \notin \mathcal{C}$. □

We can now better understand the facets of $\Delta_{\cap}(\mathcal{C})$.

Lemma 4.2.4. *Let $\mathcal{C}$ be a code on n neurons. Every facet of $\Delta_{\cap}(\mathcal{C})$ is effective.*

Proof. By (2.5), the facets of $\Delta_{\cap}(\mathcal{C})$ correspond to associated primes of $\mathrm{FI}(\mathcal{C})$, which are polarizations of associated primes of $J_{\mathcal{C}}$. Since the latter primes cannot contain both x_ℓ and $1-x_\ell$, it follows that the former primes cannot contain both x_ℓ and y_ℓ, which concludes the proof. □

Proof of Theorem 4.1.3. By [2, Lemma 5.7], the map α is a bijection, and the correspondence between minimal pseudomonomials and maximal intervals follows from the fact for any two intervals M_1 and M_2 of $\mathcal{C}$, we have $M_1 \subseteq M_2$ if and only if $\alpha(M_2) \mid \alpha(M_1)$. By Lemma 4.2.3, plus the fact that effective faces have the form $d \cup \overline{[n] \smallsetminus c}$ for some $c \subseteq d$, the map β is also a bijection. Lemma 4.2.4 states that all facets of $\Delta_{\cap}(\mathcal{C})$ are effective, and thus for each facet F we have $F = \beta(M)$ for some interval M of $\mathcal{C}$. The correspondence between facets and maximal intervals then follows from the fact that for intervals M_1 and M_2 of $\mathcal{C}$, we have $M_1 \subseteq M_2$ if and only if $\beta(M_1) \subseteq \beta(M_2)$. □

4.2.2 Proof of Theorem 4.1.4

We wish to show that the maps

$$\begin{array}{ccccc}
\{\text{minimal primes of } I(\Delta(\mathcal{C}))\} & \overset{\gamma}{\leftarrow} & \{\text{maximal codewords in } \mathcal{C}\} & \overset{\delta}{\rightarrow} & \left\{ \begin{array}{c} \text{minimal prime-sets} \\ \text{of } \Delta_{\cap}(\mathcal{C}') \end{array} \right\} \\
\langle x_i \mid i \in [n] \smallsetminus M \rangle & \leftarrow\!\shortmid & M & \mapsto & \overline{[n] \smallsetminus M}
\end{array}$$

are bijections. The main step is to understand the relationship between the prime-sets of $\Delta_{\cap}(\mathcal{C}')$ and the associated primes of $I(\Delta(\mathcal{C}))$.

Lemma 4.2.5. *Let $\mathcal{C}$ be a code on n neurons with complement code $\mathcal{C}'$. A subset $\overline{B} \subseteq \overline{[n]}$ is a prime-set of $\Delta_{\cap}(\mathcal{C}')$ if and only if $\langle x_i \mid i \in B \rangle$ contains $I(\Delta(\mathcal{C}))$. Consequently, $\overline{B}$ is a minimal prime-set of $\Delta_{\cap}(\mathcal{C}')$ if and only if $\langle x_i \mid i \in B \rangle$ is a minimal prime of $I(\Delta(\mathcal{C}))$.*

Proof. By definition, $\overline{B}$ is a prime-set of $\Delta_{\cap}(\mathcal{C}')$ if and only if $[n] \cup \overline{B}$ is not a face of $\Delta_{\cap}(\mathcal{C}')$. Equivalently, every facet of $\Delta_{\cap}(\mathcal{C}')$ of the form $F = [n] \cup \overline{[n] \smallsetminus c}$ satisfies $B \cap c \neq \varnothing$. By Theorem 4.1.3, $F = [n] \cup \overline{[n] \smallsetminus c}$ is a facet of $\Delta_{\cap}(\mathcal{C}')$ if and only if the monomial $\prod_{i \in c} x_i$ belongs to $\mathrm{CF}(J_{\mathcal{C}})$. Also, $B \cap c \neq \varnothing$ if and only if $\prod_{j \in c} x_j \in \langle x_i \mid i \in B \rangle$. Now the result follows, because the monomials in $\mathrm{CF}(J_{\mathcal{C}})$ generate $I(\Delta(\mathcal{C}))$. □

Proof of Theorem 4.1.4. The map γ is a bijection, by (2.5) and the fact that maximal codewords of $\mathcal{C}$ are facets of $\Delta(\mathcal{C})$, and $I(\Delta(\mathcal{C}))$ is its Stanley–Reisner ideal. Given that γ is a bijection, Lemma 4.2.5 shows that $\delta \circ \gamma^{-1}$ is a bijection, and so, δ is a bijection, completing the proof. □

4.3 The Factor Complex and the Polar Complex

In this section, we explore the relationship between the factor complex and the polar complex introduced in [16]. For a code $\mathcal{C}$, the *polar complex*, denoted by $\Delta_{\mathcal{P}}(\mathcal{C})$, is the simplicial complex whose Stanley–Reisner ideal is $\mathcal{P}(J_{\mathcal{C}})$, the polarization of the neural ideal of $\mathcal{C}$. The ideal $\mathcal{P}(J_{\mathcal{C}})$ is the *polar ideal* of $\mathcal{C}$.

We first show in an example that the polar and factor complexes associated to a code are, in general, not the same.

Example 4.3.1. For the complement code $\mathcal{C}' = \{1, 23, 123\}$ from Example 4.1.8, we polarize the neural ideal $J_{\mathcal{C}'} = \langle (1-x_1)(1-x_3),\ (1-x_1)(1-x_2),\ x_2(1-x_3), x_3(1-x_2)\rangle$ to obtain the polar ideal

$$\mathcal{P}(J_{\mathcal{C}'}) = \langle y_1y_3,\ y_1y_2,\ x_2y_3,\ x_3y_2\rangle = \langle x_2, x_3, y_1\rangle \cap \langle y_2, y_3\rangle \cap \langle x_3, y_1, y_3\rangle \cap \langle x_2, y_2, y_3\rangle.$$

It follows that the set of facets of the polar complex $\Delta_{\mathcal{P}}(\mathcal{C}')$ is $\{1\overline{23}, 123\overline{1}, 12\overline{2}, 13\overline{3}\}$. Thus, the polar complex has 2 more facets than the corresponding factor complex (recall Example 4.1.2).

On the other hand, the polar ideal and the factor ideal (and their corresponding complexes) share many features. A first observation is that $\mathcal{P}(J_{\mathcal{C}}) \subseteq \mathrm{FI}(\mathcal{C})$ by construction and Lemma 4.2.1. Furthermore, Lemma 4.2.1 is valid when we replace $\mathrm{FI}(\mathcal{C})$ by $\mathcal{P}(J_{\mathcal{C}})$ [16, Theorem 3.2], and consequently Lemma 4.2.2 holds for $\Delta_{\mathcal{P}}(\mathcal{C})$. Lemma 4.2.3 also is valid for $\Delta_{\mathcal{P}}(\mathcal{C})$ [16, Corollary 5.2].

As Example 4.3.1 illustrates, $\mathrm{FI}(\mathcal{C})$ strictly contains $\mathcal{P}(J_{\mathcal{C}})$ in general. A larger ideal makes for a smaller simplicial complex. The following result explains the relationship between $\Delta_{\cap}(\mathcal{C})$ and $\Delta_{\mathcal{P}}(\mathcal{C})$.

Proposition 4.3.2. *For every code $\mathcal{C}$, the factor complex $\Delta_{\cap}(\mathcal{C})$ is the subcomplex of the polar complex $\Delta_{\mathcal{P}}(\mathcal{C})$ whose facets are the effective facets of $\Delta_{\mathcal{P}}(\mathcal{C})$.*

Proof. Lemma 4.2.4 states that all facets of $\Delta_{\cap}(\mathcal{C})$ are effective, and $\mathcal{P}(J_{\mathcal{C}}) \subseteq \mathrm{FI}(\mathcal{C})$ implies that $\Delta_{\cap}(\mathcal{C}) \subseteq \Delta_{\mathcal{P}}(\mathcal{C})$. So, it suffices to show that every effective facet of $\Delta_{\mathcal{P}}(\mathcal{C})$ is a face of $\Delta_{\cap}(\mathcal{C})$. By [16, Corollaries 5.2 and 5.3], the effective facets of $\Delta_{\mathcal{P}}(\mathcal{C})$ are of the form $d \cup \overline{[n] \setminus c}$ where $[c, d]$ is a maximal interval of $\mathcal{C}$. Now apply Lemma 4.2.3. □

The key difference between the factor complex and the polar complex of a code is that the latter can have defective facets. While these facets hold useful information about quotient codes, as shown in [16], the structure of the smaller factor complex is more convenient for our purposes here.

4.4 Computational Considerations

The main result of this chapter, Theorem 4.0.1, gives a new method for checking whether a code is max-intersection-complete (Algorithm 1 below). In this section we provide an infinite family $\mathfrak{F}$ of codes for which this method is more efficient at checking max-intersection-completeness than the natural brute-force approaches.

In order to analyze the runtime of our proposed algorithm, we write it explicitly below. Correctness follows directly from Theorem 4.0.1 and the correspondence between maximal codewords of $\mathcal{C}$ and minimal primes of $I(\Delta(\mathcal{C}))$ in Theorem 4.1.4.

Algorithm 1: Checking Max-Intersection-Completeness

```
input:
    1. $\mathcal{C}$, a neural code on $n$ neurons
    2. $\mathcal{C}_{\max}$, the list of the maximal codewords of $\mathcal{C}$
    3. $\mathrm{CF}(J_{\mathcal{C}})$, the canonical form of the neural ideal of $\mathcal{C}$
output: True if $\mathcal{C}$ is max-intersection-complete and False otherwise
initialize $\mathrm{Min}(I(\Delta(\mathcal{C}))) = \varnothing$;
for (FIRST LOOP) $c \in \mathcal{C}_{\max}$ do
    Add $\langle \{x_i \mid i \in [n] \setminus c\} \rangle$ to $\mathrm{Min}(I(\Delta(\mathcal{C})))$;
end
for (OUTER LOOP) non-monomial $\phi \in CF(J_{\mathcal{C}})$ do
    for (MIDDLE LOOP) s such that $(1 - x_s) | \phi$ do
        for (INNER LOOP) $P \in \mathrm{Min}(I(\Delta(\mathcal{C})))$ do
            if $x_s \in P$ and no $x_r \in P$ divides $\phi$ then
                Go back to MIDDLE LOOP (next iteration of loop, or – if none – end loop);
            end
        end
        Go back to OUTER LOOP (next iteration of loop, or – if none – end loop);
    end
    return False;
    end algorithm
end
return True;
end algorithm
```

Remark 4.4.1. We point out that Algorithm 1 requires $\mathrm{CF}(J_{\mathcal{C}})$ as part of its input, but the brute-force methods below do not. For this reason, a complete runtime analysis of Algorithm 1 requires knowing the complexity of computing canonical forms, which is not currently well understood. The canonical form algorithm given in [22] is easily seen to be exponential in the number of neurons. A faster procedure for finding $\mathrm{CF}(J_{\mathcal{C}})$ would be very desirable, and would have implications beyond this chapter.

We now define $\mathfrak{F}$ to be the family of all neural codes $\mathcal{C}$ satisfying the following properties:

(i) The number of maximal intervals of $\mathcal{C}'$ is at most n, the number of neurons of $\mathcal{C}$.

(ii) There exists a maximal interval $[c,d]$ of $\mathcal{C}'$ with $d \neq [n]$ and $|d \setminus c| = n/2$.

(iii) There exists a maximal interval $[a,[n]]$ of $\mathcal{C}'$, where a contains $n/2$ neurons.

(iv) For every maximal interval of $\mathcal{C}'$ that has the form $[b,[n]]$, if $a \neq b$ then $a \cap b = \varnothing$.

(v) $\mathcal{C}'$ contains at most $\log_2(n)$ maximal intervals of the form $[b,[n]]$.

Note that $\mathfrak{F}$ is infinite, since the number of neurons has not been fixed. We emphasize that a code $\mathcal{C} \in \mathfrak{F}$ is given as the maximal intervals of $\mathcal{C}'$. This information is equivalent to knowing $\mathrm{CF}(J_{\mathcal{C}})$. Thus, for codes in $\mathfrak{F}$, the issue raised in Remark 4.4.1 is avoided. Finally, $\mathfrak{F}$ contains infinitely many max-intersection-complete codes, and infinitely many codes that are not max-intersection-complete, as shown in the following two examples.

Example 4.4.2 (Family of max-intersection-complete codes)**.** Let $n \geq 4$ be even. Define $\mathcal{D}_n'$ to be the union of the Boolean intervals $M_l = [1, 12\cdots(\frac{n}{2}+1)]$ and $M_L = [12\cdots(\frac{n}{2}), 123\cdots n]$, and define $\mathcal{D}_n$ to be the code on n neurons whose complement code is $\mathcal{D}_n'$. We will show that $\mathcal{D}_n$ is max-intersection-complete and in $\mathfrak{F}$. The maximal codewords of $\mathcal{D}_n$ are $234\cdots n$, $134\cdots n$, $124\cdots n$, $\ldots$, $1234\cdots(\frac{n}{2}-2)(\frac{n}{2})(\frac{n}{2}+1)\cdots n$, and $1234\cdots(\frac{n}{2}-2)(\frac{n}{2}-1)(\frac{n}{2}+1)\cdots n$. Thus any intersection of maximal codewords of $\mathcal{D}_n$ must contain n (and thus is not contained in M_l) and does not contain at least one of $1,2,3,\ldots,(\frac{n}{2})$ (and thus is not contained in M_L). Thus no intersection of maximal codewords is contained in $\mathcal{D}_n'$, and so $\mathcal{D}_n$ is max-intersection-complete.

Furthermore, $\mathcal{D}_n \in \mathfrak{F}$ since $\mathcal{D}_n'$

- has only two maximal intervals, so $\mathcal{D}_n$ satisfies (i),
- contains M_l, so $\mathcal{D}_n$ satisfies (ii),
- contains M_L, so $\mathcal{D}_n$ satisfies (iii),
- and has no maximal intervals of the form $[b, [n]]$ besides M_L, so $\mathcal{D}_n$ satisfies both (iv) and (v)).

Hence $\mathfrak{F}$ contains an infinite family of max-intersection-complete codes.

Example 4.4.3 (Family of non-max-intersection-complete codes)**.** Let $n \geq 4$ be even. Define M_l and M_L as in Example 4.4.2, and define $\mathcal{E}_n'$ to be the union of M_l, M_L, and $\{34\cdots n\}$ (the last being a Boolean interval consisting of only a single codeword). Let $\mathcal{E}_n$ be the code on n neurons whose complement code is $\mathcal{E}_n'$. One can verify that $\mathcal{E}_n \in \mathfrak{F}$ in the same manner as for $\mathcal{D}_n$ in Example 4.4.2. Like in the case with $\mathcal{D}_n$, the maximal codewords of $\mathcal{E}_n$ are $234\cdots n$, $134\cdots n$, $124\cdots n$, $\dots$, $1234\cdots(\frac{n}{2}-2)(\frac{n}{2})(\frac{n}{2}+1)\cdots n$, and $1234\cdots(\frac{n}{2}-2)(\frac{n}{2}-1)(\frac{n}{2}+1)\cdots n$. However, the intersection of the maximal codewords $234\cdots n$ and $134\cdots n$ is $34\cdots n$, which belongs to $\mathcal{E}_n'$ and hence not in $\mathcal{E}_n$. Thus $\mathcal{E}_n$ is not max-intersection-complete.

We compare Algorithm 1 to two brute-force methods for checking max-intersection-completeness:

Brute Force 1: Take all possible intersections of maximal codewords of $\mathcal{C}$, and check whether all are contained in $\mathcal{C}$.

Brute Force 2: For every $\sigma \in \mathcal{C}'$, compute c_σ, the intersection of all maximal codewords of $\mathcal{C}$ that contain σ. Then check whether $c_\sigma = \sigma$.

Proposition 4.4.4. *For every code $\mathcal{C}$ in $\mathfrak{F}$, Brute Force 1 and Brute Force 2 are exponential in the number of neurons, while Algorithm 1 is sub-exponential in the number of neurons.*

Proof. We begin by showing that the number of maximal codewords of any $\mathcal{C} \in \mathfrak{F}$ is at least $n/2$ and at most $n^{\log_2(n)}$. Recall that these maximal codewords are in bijection with the minimal primes of $I(\Delta(\mathcal{C}))$ (Theorem 4.1.4), and also that

$$I(\Delta(\mathcal{C})) = \langle \{x_\sigma \mid [\sigma, [n]] \text{ a maximal interval of } \mathcal{C}'\}\rangle. \tag{4.3}$$

(Recall that for $\sigma \subseteq [n]$, we use the notation x_σ to denote the monomial $\prod_{i\in\sigma} x_i$.) Note that 4.3 is true since $I(\Delta(\mathcal{C})) = \langle\{x_\sigma \mid \sigma \notin \Delta(\mathcal{C})\}\rangle$. We have that $[\sigma, [n]] \subseteq \mathcal{C}'$ if and only if no $c \in \mathcal{C}$ contains σ, which in turn holds if and only if $\sigma \notin \Delta(\mathcal{C})$. The fact that we only need to consider σ when $[\sigma, [n]]$ is maximal follows from the fact that if $x_\sigma | x_\tau$, then $\sigma \subseteq \tau$, and thus $[\tau, [n]] \subseteq [\sigma, [n]]$.

The monomial generators of $I(\Delta(\mathcal{C}))$ in (4.3) satisfy the following:

$(*)$ there is a generator x_a of degree $n/2$ (from condition (iii)),

$(**)$ if $x_b \neq x_a$ is a generator of $I(\Delta(\mathcal{C}))$, then $\gcd(x_a, x_b) = 1$ (from condition (iv)), and

$(***)$ there are at most $\log_2(n)$ generators (from condition (v)).

We calculate the upper bound by observing that every minimal prime P of $I(\Delta(\mathcal{C}))$ has a generating set $G_P \subseteq \{x_1, x_2, \ldots, x_n\}$, with every monomial in (4.3) divisible by at least one $x_i \in G_P$. It follows that the number of ways to choose some divisor x_i from each generator of $I(\Delta(\mathcal{C}))$ is an upper bound on the number of minimal primes. This upper bound is the product of the degrees of the monomial generators of $I(\Delta(\mathcal{C}))$, which in turn is bounded above by $n^{N_{\text{mon}}}$, where N_{mon} is the number of monomials in $CF(J_{\mathcal{C}})$. By $(***)$ there are at most $\log_2(n)$ such monomials, so the number of minimal primes – and thus the number of maximal codewords of $\mathcal{C}$ – is at most $n^{\log_2(n)}$.

For the lower bound, we first note that by $(*)$ there is a monomial generator x_a of $I(\Delta(\mathcal{C}))$ that has degree $n/2$. If $I(\Delta(\mathcal{C})) = \langle x^a \rangle$, then $I(\Delta(\mathcal{C}))$ has $n/2$ minimal primes. If $I(\Delta(\mathcal{C}))$ strictly contains $\langle x^a \rangle$, then let $\widetilde{P}$ be a minimal prime of the following nonzero ideal:

$$\widetilde{I} := \langle x_b \mid [b, [n]] \text{ is a maximal interval of } \mathcal{C}' \text{ and } b \neq a\rangle \subseteq I(\Delta(\mathcal{C})).$$

For every x_i that divides x_a, we claim that $P_i = \langle x_i \rangle + \widetilde{P}$ is a minimal prime of $I(\Delta(\mathcal{C}))$. By construction, P_i contains $I(\Delta(\mathcal{C}))$. If $Q \subsetneq P_i$ is another prime monomial ideal, either $x_i \notin Q$ or there exists $x_j \in \widetilde{P} \smallsetminus Q$. In the first case, condition $(**)$ implies that $x_a \notin Q$. In the second case, by $(**)$ and the fact that $\widetilde{P}$ is a minimal prime of $\widetilde{I}$, it follows that $\widetilde{I} \nsubseteq Q$. In both cases Q does not contain $I(\Delta(\mathcal{C}))$, and consequently P_i is a minimal prime of $I(\Delta(\mathcal{C}))$. As a distinct minimal prime $P_i = \langle x_i \rangle + \widetilde{P}$ arises from each of the $n/2$ divisors x_i of x_a, the number of minimal primes – and also the number of maximal codewords of $\mathcal{C}$ – is at least $n/2$.

Having found the upper and lower bounds on the number of maximal codewords of a code $\mathcal{C} \in \mathfrak{F}$, we now use these bounds to analyze the brute-force methods and Algorithm 1.

As there are at least $n/2$ maximal codewords, Brute Force 1 checks at least $2^{n/2}$ intersections of maximal codewords, and so is exponential in the number of neurons.

Next, Brute Force 2 checks whether each codeword of $\mathcal{C}'$ is contained in each maximal codeword of $\mathcal{C}$. So, the runtime will be at least the number of codewords of $\mathcal{C}'$ times the number of maximal codewords of $\mathcal{C}$. There are at least $n/2$ maximal codewords and, by condition (ii), at least $2^{n/2}$ elements of $\mathcal{C}'$. Thus, the runtime is at least $(n/2) * 2^{n/2}$, and so is exponential in n.

For Algorithm 1, the First Loop iterates over the maximal codewords of $\mathcal{C}$ (of which there are at most $n^{\log_2(n)}$), and the runtime of each iteration is at most n. So, the runtime of the First Loop is $O(n^{1+\log_2(n)})$. The runtime for the subsequent part of the algorithm is the product of the number of iterations of the Outer Loop, the number of iterations of the Middle Loop, and the runtime of the Inner Loop. Since the Outer Loop iterates over a subset of $\mathrm{CF}(J_{\mathcal{C}})$, by Theorem 4.1.3 and condition (i) there are at most n such iterations. Since the Middle Loop iterates over the neurons, there are at most n iterations of this loop. Finally, the Inner Loop iterates over the number of minimal primes of $I(\Delta(\mathcal{C}))$, of which there are at most $n^{\log_2(n)}$. Checking to see if x_s is in some minimal prime P takes at most n steps (one must check each generator of P) and checking to see if any $x_r \in P$ divides ϕ takes at most n^2 steps (one must compare each generator of P with each divisor of ϕ). Thus the runtime of Inner Loop is at most $n^{3+\log_2(n)}$. We conclude that the combined runtime of the Outer, Middle, and Inner Loops is $O(n^{5+\log_2(n)})$, which, it is straightforward to check, is sub-exponential in n. □

5. WHEELS

Remark 5.0.1. Throughout this chapter, unless explicitly stated otherwise, "convex" means "open convex". In particular, when we say that a code is nonconvex, we mean that it does not have a realization of open convex sets. It is possible that the code has a realization of closed convex sets.

Recall that from a simplicial complex Δ, Proposition 2.2.6 classifies a significant number of codes $\mathcal{C}$ for which $\Delta(\mathcal{C}) = \Delta$. However, this theory leaves unclassified those codes which contain all the mandatory faces but not all the max-intersection faces. Classifying such codes is nontrivial, as there are both convex (Example 2.2.12) and nonconvex (Example 2.2.13) examples.

In this chapter we introduce a new phenomenon that forbids convexity, even in codes that contain all their mandatory codewords. We show that this phenomenon – which we call a wheel – captures all nonconvex codes (with no local obstructions) on up to 5 neurons (Theorem 5.4.1). We also give combinatorial criteria for possessing a wheel, which we use to classify many codes on six neurons. As an application, we show how these combinatorial criteria reveal a code that answers a question posed by Chen, Frick, and Shiu [8]: a code that is nonconvex, has no local obstructions, and has simplicial complex of dimension two.

The outline for this chapter is as follows: in Section 5.1 we define the wheel. Sections 5.2 and 5.3 give combinatorial criteria for wheels and other results that allow us to search computationally for wheels. In Section 5.4, we use the combinatorial criteria, along with our knowledge of reducible and decomposable codes from Chapter 3, to search for wheels in two families of 6-neuron codes: the pure codes and the codes with 7 or fewer maximal codewords.

5.1 Wheels

Definition 5.1.1. Let $\mathcal{C}$ be a code on n neurons, and let $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ be a realization of $\mathcal{C}$. A tuple $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) \in (2^{[n]})^4$ is a *wheel of the realization* $\mathcal{U}$ if it satisfies:

W(i) $U_{\sigma_j} \cap U_{\sigma_k} = U_{\sigma_1} \cap U_{\sigma_2} \cap U_{\sigma_3} \neq \varnothing$ for all $1 \leq j < k \leq 3$,

W(ii) $U_{\sigma_1} \cap U_{\sigma_2} \cap U_{\sigma_3} \cap U_\tau = \varnothing$,

W(iii) if U_τ and $U_\tau \cap U_{\sigma_j}$ are convex for $j = 1, 2, 3$, then there exists a line segment whose endpoints lie one in $U_{\sigma_1} \cap U_\tau$ and the other in $U_{\sigma_3} \cap U_\tau$, that also meets $U_{\sigma_2} \cap U_\tau$.

The sets σ_1, σ_2, and σ_3 are the *spokes*, their union $\sigma_1 \cup \sigma_2 \cup \sigma_3$ is the *hub*, and τ is the *rim*. The tuple $\mathcal{W}$ is a *wheel of the code* $\mathcal{C}$ if it is a wheel of *every* realization of $\mathcal{C}$.

We remark that W(iii) immediately implies the following useful condition:

W(iii)$_\circ$ $U_{\sigma_j} \cap U_\tau \neq \varnothing$ for $j = 1, 2, 3$.

Remark 5.1.2. In the above definition, the spokes U_{σ_1} and U_{σ_3} seem to play a different role than U_{σ_2}, but this is not really the case. A more symmetric way of stating W(iii) would be to ask for a line segment that intersects all three sets $U_{\sigma_j} \cap U_\tau$, for $j = 1, 2, 3$. We have adopted the current numbering convention to simplify the writing in some of the proofs below.

The wheel is in fact a generalization of the nonconvexity from Example 2.2.13. In the next section, we show that for the nonconvex code $\mathcal{C}^\star$ from this example, every realization has a wheel. See Proposition 5.2.13.

Wheels are relevant because they forbid convexity:

Theorem 5.1.3. *Let $\mathcal{U}$ be a realization of a code $\mathcal{C}$. If $\mathcal{U}$ has a wheel, then $\mathcal{U}$ is not a convex realization. Consequently, if $\mathcal{C}$ has a wheel, then $\mathcal{C}$ is nonconvex.*

Proof. Let $(\sigma_1, \sigma_2, \sigma_3, \tau)$ be a wheel of the realization $\mathcal{U} = \{U_i\}_{i=1}^n$ of $\mathcal{C}$ in $\mathbb{R}^d$, and assume by contradiction that $\mathcal{U}$ is convex. Let L be the line segment from W(iii). Since U_τ is convex, $L \subseteq U_\tau$.

The sets U_{σ_i}, for $i = 1, 2, 3$, are convex and open. Condition W(i) implies that L intersects $U_{\sigma_1} \cap U_{\sigma_2} \cap U_{\sigma_3}$ by [12, Lemma 3.2]. As $L \subseteq U_\tau$, it follows that $U_{\sigma_1} \cap U_{\sigma_2} \cap U_{\sigma_3} \cap U_\tau \neq \varnothing$, which contradicts W(ii).

□

The geometric intuition behind the proof of Theorem 5.1.3 (and the reason for our chosen terminology) is that in a realization that has a wheel $(\sigma_1, \sigma_2, \sigma_3, \tau)$, the U_{σ_i}'s force U_τ to be nonconvex by bending U_τ around their intersection $U_{\sigma_1 \cup \sigma_2 \cup \sigma_3}$. See Figure 5.1.

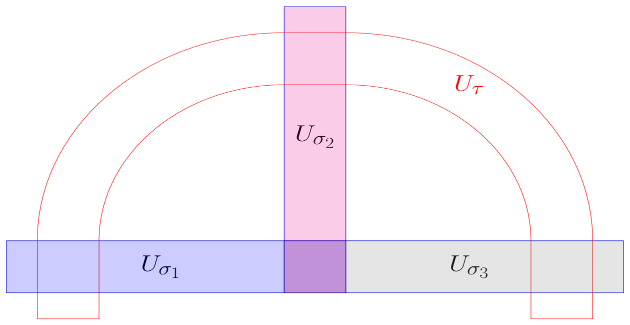


Figure 5.1: Conceptual picture of a wheel: If U_{σ_1}, U_{σ_2}, and U_{σ_3} are convex, then U_τ "bends around" $U_{\sigma_1} \cap U_{\sigma_2} \cap U_{\sigma_3}$ and so is nonconvex.

Remark 5.1.4 (Open vs. closed sets)**.** Recall from Remark 5.0.1 that Theorem 5.1.3 only applies to open convexity. Indeed, while wheels prevent codes from being open convex, it is the case that some codes with wheels can still be closed convex. For instance, the code Example 2.2.13 has a closed convex realization given in [10].

Remark 5.1.5 (Relation to sunflowers)**.** Wheels are closely related to ideas in recent work of Jeffs [6]. As a start, in a realization that has a wheel, the sets U_{σ_1}, U_{σ_2}, and U_{σ_3} form what Jeffs calls a 3-petal sunflower.

Jeffs uses sunflowers to construct an infinite family of nonconvex codes $\mathcal{C}_2, \mathcal{C}_3, \ldots$ [6, Definition 4.1]. We show in Example 5.1.6 that $\mathcal{C}_2$ contains a wheel. The codes $\mathcal{C}_m$, for $m \geq 3$, have sunflowers with $m+1$ petals. Our criteria for detecting wheels (see Section 5.2.2) have not yet been generalized to such higher-dimensional cases, where, instead of a line intersecting the 3 petals of a sunflower, a $(d-1)$-dimensional hyperplane intersects $d+1$ petals (cf. [6, Theorem 1.1]). Indeed, we checked that the codes $\mathcal{C}_m$, for small $m \geq 3$, do not have the combinatorial wheels introduced in the next section. We expect that all $\mathcal{C}_m$, for $m \geq 3$, lack wheels; however, checking condition W(iii) is difficult.

Example 5.1.6. The following code was introduced by Jeffs [6, Definition 4.1]:

$$\mathcal{C}_2 = \{\mathbf{1236}, \mathbf{234}, \mathbf{135}, \mathbf{456}, 13, 23, 4, 5, 6, \varnothing\} .$$

This code is nonconvex [6, Theorem 4.2] and is in fact minimal with respect to a certain partial order among all nonconvex neural codes (relabeling the neurons via the permutation $(1,4,2,6,3,5)$ yields the code $\mathcal{C}_0$ in [7, Theorem 5.10]). We will see that $\mathcal{C}_2$ contains the wheel $(5,6,4,3)$ (see Example 5.2.6).

5.2 Combinatorics of Wheels

Definition 5.1.1 introduces wheels using realizations because it is convenient for proving nonconvexity. That being said, the true goal of this chapter is to obtain nonconvexity criteria in terms of $\mathcal{C}$ and $\Delta(\mathcal{C})$, in a similar way to how $\mathrm{Man}(\Delta(\mathcal{C}))$ is used to determine whether $\mathcal{C}$ has local obstructions [9, Corollary 4.3].

It turns out that conditions W(i), W(ii), and W(iii)$_\circ$ from Definition 5.1.1 can be restated combinatorially, and depend only on the code $\mathcal{C}$ and not on the specific realization $\mathcal{U}$ (Proposition 5.2.3 below). The geometric condition W(iii), however, is more subtle. Nevertheless, we are able to provide combinatorial criteria which imply the existence of wheels in every realization of $\mathcal{C}$ (see Section 5.2.2). It is an open problem to recast W(iii) completely in combinatorial terms, or show that no such characterization exists.

It will be convenient to use the notion of trunks introduced in Definition 3.1.1.

5.2.1 Combinatorial Versions of W(i), W(ii), and W(iii)$_\circ$

We start with a useful technical result.

Lemma 5.2.1. *Consider a code $\mathcal{C}$ on n neurons, a realization $\mathcal{U} = \{U_i\}_{i=1}^n$ of $\mathcal{C}$, and subsets $\varphi, \psi_1, \psi_2, \dots, \psi_r \subseteq [n]$. Then $U_\varphi \subseteq \bigcup_{t=1}^r U_{\psi_t}$ if and only if $\mathrm{Tk}_\mathcal{C}(\varphi) \subseteq \bigcup_{t=1}^r \mathrm{Tk}_\mathcal{C}(\psi_t)$.*

Proof. $(\Rightarrow)$ Suppose $U_\varphi \subseteq \bigcup_{t=1}^r U_{\psi_t}$. Let $c \in \mathrm{Tk}_\mathcal{C}(\varphi)$, that is, $\varphi \subseteq c \in \mathcal{C}$. (We must show that $\psi_s \subseteq c$ for some $1 \le s \le r$.) We have that

$$U_c \subseteq U_\varphi \subseteq \bigcup_{i=1}^r U_{\psi_t}, \tag{5.1}$$

where the first containment is given by $\varphi \subseteq c$ and the second containment is by hypothesis. Next,

$c \in \mathcal{C}$ implies $U_c \setminus (\bigcup_{i \notin c} U_i) \neq \varnothing$. Combining this inequality with (5.1) yields $U_{\psi_s} \setminus (\bigcup_{i \notin c} U_i) \neq \varnothing$ for some s, which implies that $\psi_s \subseteq c$.

($\Leftarrow$) Assume $\mathrm{Tk}_{\mathcal{C}}(\varphi) \subseteq \bigcup_{t=1}^{r} \mathrm{Tk}_{\mathcal{C}}(\psi_t)$, that is, $\varphi \subseteq c \in \mathcal{C}$ implies $\psi_s \subseteq c$ for some s. Let $p \in U_\varphi$; we must show that $p \in U_{\psi_s}$ for some s. Define $c = \{i \in [n] \mid p \in U_i\}$. By construction, $p \in U_c \setminus \bigcup_{i \notin c} U_i$; so, $c \in \mathcal{C}$. Also by construction (of p and c), $\varphi \subseteq c$. Hence, by hypothesis, there exists s such that $\psi_s \subseteq c$. So, $p \in U_i$ for all $i \in \psi_s$. In other words, $p \in U_{\psi_t}$. □

We are now ready to combinatorially recast part of Definition 5.1.1.

Definition 5.2.2. A tuple $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) \in (\Delta(\mathcal{C}))^4$ is a *partial wheel* of a code $\mathcal{C}$ if it satisfies the following conditions:

P(i) $\sigma_1 \cup \sigma_2 \cup \sigma_3 \in \Delta(\mathcal{C})$, and $\mathrm{Tk}_{\mathcal{C}}(\sigma_i \cup \sigma_j) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$ for every $1 \leq j < k \leq 3$,

P(ii) $\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau \notin \Delta(\mathcal{C})$, and

P(iii)$_\circ$ $\sigma_j \cup \tau \in \Delta(\mathcal{C})$ for $j = 1, 2, 3$.

Proposition 5.2.3. *Let $\mathcal{C}$ be a code on n neurons, and let $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) \in (\Delta(\mathcal{C}))^4$. Using the notation from Definition 5.2.2, the following equivalences hold:*

(1) $\mathcal{W}$ satisfies W(i) (respectively W(ii), W(iii)$_\circ$) for some realization $\mathcal{U}$ of $\mathcal{C}$.

(2) $\mathcal{W}$ satisfies W(i) (respectively W(ii), W(iii)$_\circ$) for all realizations $\mathcal{U}$ of $\mathcal{C}$.

(3) $\mathcal{W}$ satisfies P(i) (respectively P(ii), P(iii)$_\circ$).

Proof. The fact that $\Delta(\mathcal{C})$ is the nerve of every realization of $\mathcal{C}$ directly implies the equivalences involving W(ii) and W(iii)$_\circ$. For the equivalences involving W(i), use Lemma 5.2.1 with $r = 1$, $\varphi = \sigma_i \cup \sigma_j$, and $\psi = (\sigma_1 \cup \sigma_2 \cup \sigma_3)$. □

5.2.2 Combinatorial Wheels

Our next goal is to give (combinatorial) criteria in terms of $\mathcal{C}$ and $\Delta(\mathcal{C})$ that imply the existence of wheels (and therefore nonconvexity of $\mathcal{C}$). We introduce three such criteria for a wheel, and name the wheel based on which criterion it satisfies: sprocket, wire wheel, or wheel frame (see Propositions 5.2.5, 5.2.9, and 5.2.12).

Definition 5.2.4. A *sprocket* of a code $\mathcal{C}$ is a partial wheel (Definition 5.2.2) $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ of $\mathcal{C}$ that in addition satisfies:

S(iii) There exist ρ_1, $\rho_3 \subseteq [n]$ such that:

> **S(iii)(1)** $\mathrm{Tk}_{\mathcal{C}}(\sigma_j \cup \tau) \subseteq \mathrm{Tk}_{\mathcal{C}}(\rho_j)$ for $j = 1, 3$,
>
> **S(iii)(2)** $\mathrm{Tk}_{\mathcal{C}}(\tau) \subseteq \mathrm{Tk}_{\mathcal{C}}(\rho_1) \cup \mathrm{Tk}_{\mathcal{C}}(\rho_3)$, and
>
> **S(iii)(3)** $\mathrm{Tk}_{\mathcal{C}}(\rho_1 \cup \rho_3 \cup \tau) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_2)$.

The following result shows that sprockets are wheels.

Proposition 5.2.5 (Every sprocket is a wheel)**.** *Let $\mathcal{C}$ be a code on n neurons and $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) \in (2^{[n]})^4$ a quadruple that satisfies W(iii)$_\circ$ (or equivalently P(iii)$_\circ$).*

If $\mathcal{W}$ satisfies condition S(iii) from Definition 5.2.4, then $\mathcal{W}$ satisfies W(iii). In particular, if $\mathcal{W}$ is a sprocket of $\mathcal{C}$, then $\mathcal{W}$ is a wheel of $\mathcal{C}$. Consequently, if $\mathcal{C}$ has a sprocket, then $\mathcal{C}$ is nonconvex.

Proof. We must show that S(iii) and P(iii)$_\circ$ together imply W(iii). Using Lemma 5.2.1, we see that S(iii) is equivalent to the following condition in terms of covers:

G(iii) There exist $\rho_1, \rho_3 \subseteq [n]$ such that in every realization $\mathcal{U} = \{U_i\}_{i=1}^n$ of $\mathcal{C}$,

> **G(iii)(1)** $U_{\sigma_j} \cap U_\tau \subseteq U_{\rho_j}$ for $j = 1, 3$,
>
> **G(iii)(2)** $U_\tau \subseteq U_{\rho_1} \cup U_{\rho_3}$, and
>
> **G(iii)(3)** $U_{\rho_1} \cap U_{\rho_3} \cap U_\tau \subseteq U_{\sigma_2}$.

To complete the proof, we now show that G(iii) and P(iii)$_\circ$ imply W(iii).

Let $\mathcal{U}$ be a realization of $\mathcal{C}$ such that U_τ and $U_\tau \cap U_{\sigma_j}$ are convex for $j = 1, 2, 3$. Then, for $j = 1, 3$, condition P(iii)$_\circ$ and the definition of nerve imply the inequality here:

$$\varnothing \neq (U_{\sigma_j} \cap U_\tau) \subseteq U_{\rho_j} ,$$

and the containment follows from G(iii)(1). So, for $j = 1, 3$, there exist $p_j \in U_{\sigma_j} \cap U_\tau \subseteq U_{\rho_j}$. As p_1 and p_3 are in the convex set U_τ, so is the line segment, denoted by L, between p_1 and p_3:

$$L \subseteq U_\tau \subseteq (U_{\rho_1} \cup U_{\rho_3}) , \tag{5.2}$$

and the second containment is by G(iii)(2). Thus, the (connected) set L is covered by the nonempty sets $U_{\rho_j} \cap L$, which are open in the subspace topology of L. So,

$$\varnothing \neq (U_{\rho_1} \cap U_{\rho_3} \cap L) \subseteq (U_{\rho_1} \cap U_{\rho_3} \cap U_\tau) \subseteq U_{\sigma_2} ,$$

where the containments are by (5.2) and G(iii)(3), respectively. It follows that L meets $U_{\sigma_2} \cap U_\tau$, and so W(iii) holds. □

Note that in the previous proof, we showed that *every* line segment whose endpoints are one in $U_{\sigma_1} \cap U_\tau$ and the other in $U_{\sigma_3} \cap U_\tau$ meets $U_{\sigma_2} \cap U_\tau$. This is, on its face, stronger than W(iii), which requires the existence of only one such line segment.

To explain the terminology, a sprocket is a toothed wheel, such as the gear wheel on a bicycle; we imagine the sets U_{ρ_1} and U_{ρ_3} as overlapping links in a roller chain set on a sprocket.

Example 5.2.6 (A code with a sprocket)**.** Recall the code $\mathcal{C}_2 = \{\mathbf{1236}, \mathbf{234}, \mathbf{135}, \mathbf{456}, 13, 23, 4, 5, 6, \varnothing\}$ from Example 5.1.6. We show that $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) = (5, 6, 4, 3)$ is a sprocket, where $\rho_1 = 13$ and $\rho_3 = 23$. First, $\sigma_1 \cup \sigma_2 \cup \sigma_3 = 456 \in \Delta(\mathcal{C})$, and $\text{Tk}_\mathcal{C}(\sigma_i \cup \sigma_j) = \{456\} = \text{Tk}_\mathcal{C}(456)$ for all $1 \leq i < j \leq 3$. So P(i) holds. Next, $\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau = 3456 \notin \Delta(\mathcal{C})$, so P(ii) is satisfied. Also, $\sigma_1 \cup \tau = 35$, $\sigma_2 \cup \tau = 36$, and $\sigma_3 \cup \tau = 34$ are all faces of $\Delta(\mathcal{C})$; this verifies P(iii)$_\circ$.

To satisfy S(iii), set $\rho_1 = 13$ and $\rho_3 = 23$.

Then for S(iii)(1), we have

$$\text{Tk}_\mathcal{C}(\sigma_1 \cup \tau) = \text{Tk}_\mathcal{C}(35) = \{135\} \subseteq \{13, 135\} = \text{Tk}_\mathcal{C}(13) = \text{Tk}_\mathcal{C}(\rho_1) \text{ and}$$

$$\text{Tk}_\mathcal{C}(\sigma_3 \cup \tau) = \text{Tk}_\mathcal{C}(34) = \{234\} \subseteq \{23, 234\} = \text{Tk}_\mathcal{C}(23) = \text{Tk}_\mathcal{C}(\rho_3).$$

For S(iii)(2), we have

$$\mathrm{Tk}_{\mathcal{C}}(\tau) = \mathrm{Tk}_{\mathcal{C}}(3) = \{1236, 234, 135, 13, 23\} = \{1236, 135, 13\} \cup \{1236, 234, 23\}$$

$$= \mathrm{Tk}_{\mathcal{C}}(13) \cup \mathrm{Tk}_{\mathcal{C}}(23) = \mathrm{Tk}_{\mathcal{C}}(\rho_1) \cup \mathrm{Tk}_{\mathcal{C}}(\rho_3).$$

Lastly, for S(iii)(3), we have

$$\mathrm{Tk}_{\mathcal{C}}(\rho_1 \cup \rho_3 \cup \tau) = \mathrm{Tk}_{\mathcal{C}}(123) = \{1236\} \subseteq \{1236, 456, 6\} = \mathrm{Tk}_{\mathcal{C}}(6) = \mathrm{Tk}_{\mathcal{C}}(\sigma_2).$$

Thus, S(iii) is satisfied, and so $(5, 6, 4, 3)$ is a sprocket of $\mathcal{C}_2$.

However, not every wheel is a sprocket.

Example 5.2.7 (A wheel that is not a sprocket)**.** Consider the code

$$\mathcal{C}_{\mathrm{TL}} = \{\mathbf{123}, \mathbf{145}, \mathbf{245}, \mathbf{246}, \mathbf{346}, 24, 45, 46, 1, 2, 3, \varnothing\} ,$$

and consider $\mathcal{W}_{\mathrm{TL}} = (1, 2, 3, 4)$, which we will later show is a wheel (Proposition 5.2.10). We claim that $\mathcal{W}_{\mathrm{TL}}$ is not a sprocket. We show this by proving there is no eligible pair ρ_1 and ρ_3.

Suppose for contradiction that ρ_1 and ρ_3 satisfy S(iii). The codewords 145 and 346 contain, respectively, $\sigma_1 \cup \tau = 14$ and $\sigma_3 \cup \tau = 34$, so S(iii)(1) implies that $\rho_1 \subseteq 145$ and $\rho_3 \subseteq 346$. Next, 24 is a codeword that contains $\tau = 4$, so, by S(iii)(2), we have $\rho_1 \subseteq 24$ or $\rho_3 \subseteq 24$. The above constraints imply that $\rho_1 \subseteq \{4\}$ or $\rho_3 \subseteq \{4\}$. Hence, $\rho_1 \cup \rho_3 \cup \tau = \rho_1 \cup \rho_3 \cup \{4\}$ is a subset of 145 or 346, and so $\mathrm{Tk}_{\mathcal{C}}(\rho_1 \cup \rho_3 \cup \tau)$ contains 145 or 346. However, neither 145 nor 346 is in $\mathrm{Tk}_{\mathcal{C}}(\{2\}) = \mathrm{Tk}_{\mathcal{C}}(\sigma_2)$, contradicting S(iii)(3).

Thus $\mathcal{W}_{\mathrm{TL}}$ is not a sprocket of $\mathcal{C}_{\mathrm{TL}}$.

Definition 5.2.8. Let $\mathcal{C}$ be a code. A *wire wheel* of $\mathcal{C}$ is a tuple $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ satisfying P(i) and P(ii), such that

- $\tau \notin \mathcal{C}$,

- $\mathrm{Lk}_\tau(\Delta(\mathcal{C}))$ is a tree,
- $|\sigma_i \smallsetminus \tau| = 1$ for $i = 1, 2, 3$, and
- the unique path in the tree $\mathrm{Lk}_\tau(\Delta(\mathcal{C}))$ between $\sigma_1 \smallsetminus \tau$ and $\sigma_3 \smallsetminus \tau$ contains $\sigma_2 \smallsetminus \tau$.

The intuition behind this name is that wire wheels have thin spokes.

Proposition 5.2.9. *Wire wheels are wheels. In particular, if $\mathcal{C}$ has a wire wheel, then it is nonconvex.*

Note that if $\mathcal{C}$ is a code and $\tau \in \Delta(\mathcal{C}) \smallsetminus \mathcal{C}$ is such that $\mathrm{Lk}_\tau(\Delta(\mathcal{C}))$ is one-dimensional, then $\mathrm{Lk}_\tau(\Delta(\mathcal{C}))$ is contractible if and only if it is a tree. Thus, if $\mathrm{Lk}_\tau(\Delta(\mathcal{C}))$ is not a tree then $\mathcal{C}$ has a local obstruction and so is nonconvex. Our interest in Proposition 5.2.9 is that it provides a situation when there is no local obstruction, but the code is still nonconvex.

Proof of Proposition 5.2.9. Let $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ be a wire wheel of $\mathcal{C}$. By the assumption that $\mathcal{W}$ satisfies P(i) and P(ii), and Proposition 5.2.3, we have that $\mathcal{W}$ satisfies W(i) and W(ii). Thus it suffices to show that $\mathcal{W}$ satisfies W(iii).

Relabel the neurons, if necessary, so that $\sigma_\ell \smallsetminus \tau = \{\ell\}$ for $\ell = 1, 2, 3$. Now let $\mathcal{U} = \{U_i\}_{i=1}^n$ be a realization of $\mathcal{C}$ such that U_τ and $U_\tau \cap U_{\sigma_j}$ are convex for $j = 1, 2, 3$. Note that since $\{\ell\} \cup \tau = \sigma_\ell \cup \tau$ for $\ell = 1, 2, 3$, we have that $U_{\sigma_\ell} \cap U_\tau = U_\ell \cap U_\tau$. Thus it suffices to show that there is a line segment from a point in $U_1 \cap U_\tau$ to a point in $U_3 \cap U_\tau$ that meets $U_2 \cap U_\tau$.

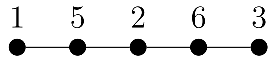
First, as $1, 3 \in \mathrm{Lk}_\tau(\Delta(\mathcal{C}))$, the sets $U_1 \cap U_\tau$ and $U_3 \cap U_\tau$ are nonempty, so let L be a line segment from a point in one set to a point in the other. The endpoints of L are in the convex set U_τ and so L is contained in U_τ. As $\tau \notin \mathcal{C}$, the line segment L is covered by the relatively open intervals $U_j \cap L$, where $j \in \mathrm{Lk}_\tau(\Delta(\mathcal{C}))$. Additionally, as the link is one-dimensional, these intervals have only pairwise intersections. Hence, for each such interval $U_j \cap L$ that is nonempty, there exists a point p_j that is in that interval and no other intervals. We conclude that the intersection patterns of the intervals correspond to a path in the link, which by assumption contains the vertex 2. Hence, the desired point p_2 in $U_j \cap L$ exists. □

The method used in the proof of Proposition 5.2.9 is called *order-forcing*, and is described further in [23].

Here we present an application of wheels, in particular wire wheels, in answering a question prompted by previous work on neural codes. A *3-sparse* code is a code for which the neural complex has dimension at most 2. Recently, the authors of [8] asked whether every 3-sparse code with no local obstructions, is convex. (The answer is "yes" for codes on up to 5 neurons [9, 17].) The following result answers this question in the negative, by providing an example of a code on 6 neurons that has a wire wheel.

Proposition 5.2.10. *There is a nonconvex 3-sparse code with no local obstructions.*

Proof. Recall the code $\mathcal{C}_{\text{TL}} := \{\mathbf{123}, \mathbf{145}, \mathbf{245}, \mathbf{246}, \mathbf{346}, 24, 45, 46, 1, 2, 3, \varnothing\}$ from Example 5.2.7. Here we use the notation TL to stand for "tree link".

The maximal codewords have length 3, so $\dim(\Delta(\mathcal{C}_{\text{TL}})) = 2$. The max-intersection faces are $\varnothing, 1, 2, 3, 4, 24, 45$, and 46. With the exception of 4, all of these intersections are codewords of $\mathcal{C}_{TL}$. While $4 \notin \mathcal{C}$, the link $\text{Lk}_{\{4\}}(\Delta(\mathcal{C}_{\text{TL}}))$ is the following path, which is contractible: 

Thus, by Proposition 2.2.11, the code $\mathcal{C}_{\text{TL}}$ has no local obstructions.

Consider $\mathcal{W}_{\text{TL}} := (\sigma_1, \sigma_2, \sigma_3, \tau) = (1, 2, 3, 4)$. First, $\mathcal{W}_{\text{TL}}$ satisfies P(i), as $123 \in \Delta(\mathcal{C}_{\text{TL}})$ and $\text{Tk}_{\mathcal{C}_{\text{TL}}}(\sigma_i \cup \sigma_j) = \{123\} = \text{Tk}_{\mathcal{C}_{\text{TL}}}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$ for $1 \le i < j \le 3$. Next, $\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau = 1234 \notin \mathcal{C}_{\text{TL}}$, so P(ii) also holds. Finally, we already saw that the link $\text{Lk}_{\{4\}}(\Delta(\mathcal{C}_{\text{TL}}))$ is a path (and thus a tree) in which the unique path from vertex $(\sigma_1 \setminus \tau) = 1$ to $(\sigma_3 \setminus \tau) = 3$ passes through the vertex $(\sigma_2 \setminus \tau) = 2$. Hence $\mathcal{W}_{\text{TL}}$ is a wire wheel, and so Proposition 5.2.9 implies that $\mathcal{C}_{\text{TL}}$ is nonconvex. □

Recall that in Definition 5.1.1 we distinguished between the wheel of a code and the wheel of a realization. The definition of wheel of a code requires the tuple under consideration to be a common wheel of every realization of the code. However, for proving that certain codes are not convex, this is too strong a requirement. Indeed, it suffices to show that every realization has a wheel, which may vary from one realization to another. Accordingly, we now introduce a different kind of combinatorial wheel, which by Proposition 5.2.12 implies nonconvexity using this more flexible approach.

Definition 5.2.11. Let $\mathcal{C}$ be a code on n neurons and $\Delta(\mathcal{C})$ its neural complex. A triple $(\sigma_1, \sigma_3, \tau) \in (2^{[n]})^3$ is a *wheel frame* of $\mathcal{C}$ if it satisfies the following conditions:

F(i) $\sigma_1 \cup \sigma_3 \in \Delta(\mathcal{C})$, and for all $\omega \subseteq \sigma_1 \cup \sigma_3$ such that

(1) neither $\omega \subseteq \sigma_1$ nor $\omega \subseteq \sigma_3$, and

(2) $\omega \cup \tau \in \Delta(\mathcal{C})$,

it follows that $\mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \omega) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_3)$ and $\mathrm{Tk}_{\mathcal{C}}(\sigma_3 \cup \omega) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_3)$,

F(ii) $\tau \cup \sigma_1 \cup \sigma_3 \notin \Delta(\mathcal{C})$.

F(iii) $\sigma_1 \cup \tau \in \Delta(\mathcal{C})$ and $\sigma_3 \cup \tau \in \Delta(\mathcal{C})$,

F(iv) $\sigma_1 \cap \sigma_3 = \varnothing$ and $\mathrm{Tk}_{\mathcal{C}}(\tau) \subseteq \cup_{i \in \sigma_1 \cup \sigma_3} \mathrm{Tk}_{\mathcal{C}}(\{i\})$

Proposition 5.2.12 (Wheel frames generate wheels)**.** *Let $(\sigma_1, \sigma_3, \tau)$ be a wheel frame of a code $\mathcal{C}$ on n neurons. Then, for every realization $\mathcal{U}$ of $\mathcal{C}$, there exists $\sigma_2 \subseteq [n]$ such that $(\sigma_1, \sigma_2, \sigma_3, \tau)$ is a wheel of $\mathcal{U}$. In particular, codes with wheel frames are nonconvex.*

Proof. Let $\mathcal{U} = \{U_i\}_{i=1}^n$ be a realization of $\mathcal{C}$. Our first task is to construct σ_2. First consider the case when U_τ and $U_\tau \cap U_i$ are convex for all $i \in [n]$. As $\Delta(\mathcal{C})$ is the nerve of $\mathcal{U}$, F(ii) and F(iii) imply that $U_{\sigma_1 \cup \tau}$ and $U_{\sigma_3 \cup \tau}$ are disjoint, nonempty sets. Let $p_1 \in U_{\sigma_1 \cup \tau}$ and $p_3 \in U_{\sigma_3 \cup \tau}$, and let L denote the line segment with endpoints p_1 and p_3. Then $L \subseteq U_\tau$, as $p_1, p_2 \in U_\tau$ and U_τ is convex.

We claim that there exists some $p_2 \in L$ between p_1 and p_3 such that $p_2 \notin U_{\sigma_j}$ for $j = 1, 3$. Set $L_j = L \cap (\bigcup_{i \in \sigma_j} U_i)$ for $j = 1, 3$. By F(iv), we have that U_τ (and thus L) is covered by $\{U_i\}_{i \in \sigma_1 \cup \sigma_3}$, so that $L = L_1 \cup L_3$. Both L_1 and L_3 are relatively open subsets of the connected set L, and $L_1, L_3 \neq \varnothing$ (because $p_1 \in L_1$ and $p_3 \in L_3$), so it follows that $L_1 \cap L_3 \neq \varnothing$. We conclude that there exist $i_1 \in \sigma_1$, $i_3 \in \sigma_3$, and $p_2 \in L$ such that $p_2 \in U_{i_1} \cap U_{i_3}$. We remark that $i_1 \in (\sigma_1 \smallsetminus \sigma_3)$ and $i_3 \in (\sigma_3 \smallsetminus \sigma_1)$ because $\sigma_1 \cap \sigma_3 = \varnothing$, as required by F(iv). Let $\sigma_2 := \{i_1\} \cup \{i_3\}$.

In the remaining case, define σ_2 as follows. Note that $\sigma_1 \nsubseteq \sigma_3$ and $\sigma_3 \nsubseteq \sigma_1$, as otherwise F(ii) and F(iii) do not both hold. So, pick $i_1 \in (\sigma_1 \smallsetminus \sigma_3)$ and $i_2 \in (\sigma_3 \smallsetminus \sigma_1)$, and then set $\sigma_2 := \{i_1\} \cup \{i_3\}$.

We now claim that $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ is a wheel of $\mathcal{U}$.

First, by construction, $\mathcal{W}$ satisfies W(iii). To complete the proof, we will show that $\mathcal{W}$ also satisfies P(i) and P(ii) (recall Proposition 5.2.3).

For P(i): By construction $\sigma_2 \subseteq \sigma_1 \cup \sigma_3$. Therefore, $\sigma_1 \cup \sigma_2 \cup \sigma_3 = \sigma_1 \cup \sigma_3$, and so $\sigma_1 \cup \sigma_2 \cup \sigma_3$ is in $\Delta(\mathcal{C})$ (due to P(i)) and $\mathrm{Tk}(\sigma_1 \cup \sigma_3) \subseteq \mathrm{Tk}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$. It remains only to show that $\mathrm{Tk}_{\mathcal{C}}(\sigma_i \cup \sigma_2) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$ for $i = 1, 3$. We verify this by applying F(i) with $\omega = \sigma_2$, as follows. We already saw that $\sigma_2 \subseteq (\sigma_1 \cup \sigma_3)$. Also, $\sigma_2 \nsubseteq \sigma_1$ and $\sigma_2 \nsubseteq \sigma_3$ hold, because $i_3 \in (\sigma_2 \smallsetminus \sigma_3)$. $i_1 \in (\sigma_2 \smallsetminus \sigma_1)$. Lastly, since $p_2 \in U_{\sigma_2} \cap U_\tau$ we see that $U_{\sigma_2} \cap U_\tau \neq \varnothing$, and so (as $\Delta(\mathcal{C})$ is the nerve of $\mathcal{U}$) $\sigma_2 \cup \tau \in \Delta(\mathcal{C})$. Thus, by F(i), the desired trunk containments hold. Hence, $\mathcal{W}$ satisfies P(i).

Finally, $\mathcal{W}$ satisfies P(ii) because of F(ii), since $\sigma_2 \subseteq \sigma_1 \cup \sigma_3$ implies $\tau \cup \sigma_1 \cup \sigma_3 = \tau \cup \sigma_1 \cup \sigma_2 \cup \sigma_3$. □

We end this section by showing that the code $\mathcal{C}^\star$ from Example 2.2.13 has a wheel frame.

Proposition 5.2.13 ($\mathcal{C}^\star$ has a wheel frame)**.** *Let $\mathcal{C}^\star$ be the code from Example 2.2.13. If $\mathcal{D}$ is a code such that*

(1) $\mathcal{C}^\star \subseteq \mathcal{D} \subseteq \Delta(\mathcal{C}^\star)$ and

(2) $\mathcal{D}$ does not *contain any of the following codewords:* 1, 234*, and* 245*,*

then $\mathcal{D}$ contains a wheel frame and thus is nonconvex. In particular, $\mathcal{C}^\star$ has a wheel frame.

Proof. Assume $\mathcal{D}$ satisfies (5.2.13) and (5.2.13). It follows that $\Delta(\mathcal{D}) = \Delta(\mathcal{C}^\star)$.

We will show that $(\sigma_1, \sigma_3, \tau) = (23, 45, 1)$ is a wheel frame. It is straightforward to check that conditions F(iii) and F(ii) hold. Condition F(iv) is also easy to check (here, the assumption $1 \notin \mathcal{D}$ is used). Finally we consider F(i). The only set $\omega \subseteq (\sigma_1 \cup \sigma_3) = 2345$ for which $\omega \nsubseteq \sigma_1 = 23$, $\omega \nsubseteq \sigma_3 = 45$, and $\omega \cup \tau = \omega \cup \{1\} \in \Delta(\mathcal{C}^\star)$ is the set $\omega = 34$. The trunk conditions in F(i), namely, $\mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \omega) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_3)$ and $\mathrm{Tk}_{\mathcal{C}}(\sigma_3 \cup \omega) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_3)$, are now readily seen to hold, as (respectively) $234 \notin \mathcal{D}$ and $345 \notin \mathcal{D}$. □

5.3 Narrowing Down the Search for Wheels

In the process of analyzing $\mathcal{C}$ for a wheel, one would naturally ask whether it is necessary to look at every quadruple of faces in $\Delta(\mathcal{C})$. Fortunately, at least for the case of wheels satisfying the above criteria, the answer is no. In this section, we show how one can narrow down the search. First, we show that none of the σ_i's can contain one another, and, at least in the case of sprockets, τ must not be a codeword of $\mathcal{C}$. Second, taking inspiration from the work of [9] showing that every local obstruction "bubbles up" to a max-intersection face, we show that in a code with a sprocket, the sprocket will "bubble up" to a wheel where the rim is a missing max-intersection face of $\Delta(\mathcal{C})$.

5.3.1 Non-Wheels

Here we present several cases where a quadruple $(\sigma_1, \sigma_2, \sigma_3, \tau)$ cannot satisfy one of the combinatorial criteria introduced in the previous section. Before proceeding, we must warn that these conditions do not necessarily mean that the quadruple cannot be a wheel; they just mean that the quadruple will not satisfy any of the above criteria.

First, when checking for sprockets, one need only check those quadruples $(\sigma_1, \sigma_2, \sigma_3, \tau)$ where $\tau \notin \mathcal{C}$.

Proposition 5.3.1. *Suppose $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ is a sprocket. Then $\tau \notin \mathcal{C}$.*

Proof. Suppose for a contradiction that $\mathcal{W}$ is a sprocket of $\mathcal{C}$, but $\tau \in \mathcal{C}$. By assumption there exist ρ_1 and ρ_3 by which $\mathcal{W}$ satisfies S(iii).

Since τ is a codeword, $\mathrm{Tk}_{\mathcal{C}}(\tau)$ contains τ. Hence, by S(iii)(2), at least one of $\mathrm{Tk}_{\mathcal{C}}(\rho_1)$ or $\mathrm{Tk}_{\mathcal{C}}(\rho_3)$ contains τ. By symmetry, we may assume that $\tau \in \mathrm{Tk}_{\mathcal{C}}(\rho_1)$.

By P(iii)$_\circ$ and Proposition 5.2.3 there exists a codeword c such that $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_3 \cap \tau)$. Then by S(iii)(1) we have $c \in \mathrm{Tk}_{\mathcal{C}}(\rho_3)$, and thus $c \in \mathrm{Tk}_{\mathcal{C}}(\rho_3 \cup \tau)$.

Since $\tau \in \mathrm{Tk}_{\mathcal{C}}(\rho_1)$, we have that $\rho_1 \subseteq \tau$, and thus $\rho_1 \cup \rho_3 \cup \tau = \rho_3 \cup \tau$. Therefore, $c \in \mathrm{Tk}_{\mathcal{C}}(\rho_1 \cup \rho_3 \cup \tau)$, and so by S(iii)(3) it follows that $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_2)$.

Thus $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_2) \cap \mathrm{Tk}_{\mathcal{C}}(\sigma_3) = \mathrm{Tk}_{\mathcal{C}}(\sigma_2 \cup \sigma_3)$. Then P(i) implies that $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$. But then $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3) \cap \mathrm{Tk}_{\mathcal{C}}(\tau) = \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau)$, contradicting P(ii). □

The next proposition states that in order for a quadruple $(\sigma_1, \sigma_2, \sigma_3, \tau)$ to be a wheel, no spoke may be contained in another spoke. Note that this proposition is more broadly applicable than any other result in this section. It does not apply specifically to any criterion. In fact, any wheel, even one not satisfying any of the sufficient criteria in this chapter, must satisfy this condition.

Proposition 5.3.2. *Suppose $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ is a wheel of a code $\mathcal{C}$. Then $\sigma_j \nsubseteq \sigma_k$ for all j, k with $j \neq k$.*

Proof. Suppose that $\mathcal{W}$ is a wheel of $\mathcal{C}$, and suppose for a contradiction that $\sigma_j \subseteq \sigma_k$ for some $1 \leq j, k \leq 3$ with $j \neq k$. By W(iii)$_\circ$ we have that $\sigma_k \cup \tau \in \Delta(\mathcal{C})$. Therefore, there exists some $c \in C$ such that $\sigma_k \cup \tau \subseteq c$, and so it follows from $\sigma_j \subseteq \sigma_k$ that $\sigma_j \cup \tau \subseteq c$ as well. Thus, c contains both σ_j and σ_k, so $c \in \mathrm{Tk}_\mathcal{C}(\sigma_j \cup \sigma_k) \subseteq \mathrm{Tk}_\mathcal{C}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$, with the containment following from P(i) and Proposition 5.2.3. But then $c \in \mathrm{Tk}_\mathcal{C}(\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau)$, so $\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau \in \Delta(\mathcal{C})$, which contradicts W(ii). □

The significance of Proposition 5.3.2 is its usefulness for algorithms that detect wheels. Specifically, it decreases the number of triples $(\sigma_1, \sigma_2, \sigma_3)$ one must check for being the spokes of a wheel. The next result, Proposition 5.3.3, serves a similar, albeit more restricted, purpose in that it gives a case where the combination of $(\sigma_1, \sigma_2, \sigma_3, \tau)$ cannot be a sprocket.

Proposition 5.3.3. *Suppose $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau) \in (2^{[n]})^4$, $\mathcal{C}$ is a code on n neurons, and $\rho \subseteq [n]$ is such that:*

(1) *for $j = 1, 3$, we have $Tk_\mathcal{C}(\sigma_j \cup \tau) \subseteq Tk_\mathcal{C}(\rho)$,*

(2) *$Tk_\mathcal{C}(\tau) \subseteq Tk_\mathcal{C}(\rho)$, and*

(3) *$Tk_\mathcal{C}(\rho \cup \tau) \subseteq Tk_\mathcal{C}(\sigma_2)$.*

(That is, $\mathcal{W}$ satisfies S(iii) via $\rho := \rho_1$ and $\rho := \rho_3$.) Then $\mathcal{W}$ is not a sprocket of $\mathcal{C}$.

Proof. Assume for a contradiction that $\mathcal{W}$ satisfies S(iii) when $\rho := \rho_1$ and $\rho := \rho_3$. If $\mathcal{W}$ is a sprocket, then by P(iii)$_\circ$ there exists a codeword $c \in \mathcal{C}$ such that $c \in \mathrm{Tk}_\mathcal{C}(\sigma_1 \cup \tau)$. Thus, by S(iii)(1), we have that $c \in \mathrm{Tk}_{\mathcal{C}}(\rho)$. So, $c \in \mathrm{Tk}_{\mathcal{C}}(\rho \cup \tau)$, and thus by S(iii)(3) we have that $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_2)$. Furthermore, $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_1) \cap \mathrm{Tk}_{\mathcal{C}}(\sigma_2) = \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3)$, where the last containment is from P(i). Hence, $c \in \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3) \cap \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \tau) = \mathrm{Tk}_{\mathcal{C}}(\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau)$, and thus $(\sigma_1 \cup \sigma_2 \cup \sigma_3 \cup \tau) \subseteq c$, which contradicts P(ii). □

5.3.2 Bubbling Up Property for Sprockets

The next result, like Propositions 5.3.2 and 5.3.3, reduces the number of quadruples that must be checked for a wheel. This approach, however, is inspired by the theory of local obstructions from [9]. Recall that a local obstruction guarantees nonconvexity, and to check for local obstructions of $\mathcal{C}$ it suffices to see whether all mandatory faces of $\Delta(\mathcal{C})$ are contained in $\mathcal{C}$. While not every local obstruction is a mandatory face, nonetheless it suffices to check only the mandatory faces, due to the bubbling up condition stated in Proposition 2.2.11.

In a similar manner, our goal is to restrict the task of searching for wheels from checking all the quadruples of $\Delta(\mathcal{C})$ to just those quadruples that involve at least one max-intersection face of $\Delta(\mathcal{C})$. Specifically, we would like to show that if $\mathcal{C}$ has a wheel, then $\mathcal{C}$ also has a wheel where the rim is a max-intersection face. Here, we show this is true for sprockets:

Proposition 5.3.4 (Bubble Up Property for Sprockets)**.** *Let $\mathcal{C}$ be a code. Suppose $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ is a sprocket. Let $\tilde{\tau}$ be the intersection of all maximal codewords of $\mathcal{C}$ that contain τ. Then $\widetilde{\mathcal{W}} = (\sigma_1, \sigma_2, \sigma_3, \tilde{\tau})$ is also a sprocket.*

Proof. $\widetilde{\mathcal{W}}$ satisfies P(i) since $\mathcal{W}$ also satisfies P(i), and P(i) depends only on σ_1, σ_2, and σ_3.

Next, we show that $\widetilde{\mathcal{W}}$ satisfies P(ii). The containment $\tau \subseteq \tilde{\tau}$ implies that $(\sigma_1 \cup \sigma_2 \cup \sigma_3) \cup \tau \subseteq (\sigma_1 \cup \sigma_2 \cup \sigma_3) \cup \tilde{\tau}$. Then, from $(\sigma_1 \cup \sigma_2 \cup \sigma_3) \cup \tau \notin \Delta(\mathcal{C})$ (because $\mathcal{W}$ satisfies P(ii)), we have that $(\sigma_1 \cup \sigma_2 \cup \sigma_3) \cup \tilde{\tau} \notin \Delta(\mathcal{C})$.

To show that $\widetilde{\mathcal{W}}$ satisfies P(iii)$_\circ$, we must show that $\sigma_j \cup \tilde{\tau} \in \Delta(\mathcal{C})$ for $j = 1, 2, 3$. Since $\mathcal{W}$ satisfies P(iii)$_\circ$, it must be that $\sigma_j \cup \tau$ is contained in some codeword c_j of $\mathcal{C}$. We may assume that c_j is maximal. Then by construction of $\tilde{\tau}$, it must be that c_j contains $\tilde{\tau}$ as well. Thus c_j contains $\sigma_j \cup \tilde{\tau}$, and so $\sigma_j \cup \tilde{\tau} \in \Delta(\mathcal{C})$.

To show that $\widetilde{\mathcal{W}}$ satisfies S(iii), let ρ_1, ρ_3 be some ρ_1, ρ_3 for which $\mathcal{W}$ satisfies S(iii):

- $\widetilde{\mathcal{W}}$ satisfies S(iii)(1): For $j = 1, 3$, since $\tau \subseteq \tilde{\tau}$ implies $\sigma_j \cup \tau \subseteq \sigma_j \cup \tilde{\tau}$, we have that $\mathrm{Tk}_{\mathcal{C}}(\sigma_j \cup \tilde{\tau}) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_j \cup \tau) \subseteq \mathrm{Tk}_{\mathcal{C}}(\rho_j)$, which, combined with the fact that $\mathcal{W}$ satisfies S(iii)(1), implies the containment $\mathrm{Tk}_{\mathcal{C}}(\sigma_j \cup \tilde{\tau}) \subseteq \mathrm{Tk}_{\mathcal{C}}(\rho_j)$.

- $\widetilde{\mathcal{W}}$ satisfies S(iii)(2): Since $\tau \subseteq \tilde{\tau}$ implies $\mathrm{Tk}(\tilde{\tau}) \subseteq \mathrm{Tk}(\tau)$, we have that $\mathrm{Tk}(\tilde{\tau}) \subseteq (\mathrm{Tk}(\rho_1) \cup \mathrm{Tk}(\rho_3))$ follows from the fact that $\mathcal{W}$ satisfies S(iii)(2).

- $\widetilde{\mathcal{W}}$ satisfies S(iii)(3): Since $\tau \subseteq \tilde{\tau}$ implies that $\rho_1 \cup \rho_3 \cup \tau \subseteq \rho_1 \cup \rho_3 \cup \tilde{\tau}$, we have that $\mathrm{Tk}(\rho_1 \cup \rho_3 \cup \tilde{\tau}) \subseteq \mathrm{Tk}(\rho_1 \cup \rho_3 \cup \tau)$, which combined with the fact that $\mathcal{W}$ satisfies S(iii)(3) implies that $\mathrm{Tk}_{\mathcal{C}}(\rho_1 \cup \rho_3 \cup \tilde{\tau}) \subseteq \mathrm{Tk}_{\mathcal{C}}(\sigma_2)$.

Thus $\widetilde{\mathcal{W}}$ satisfies S(iii) with ρ_1 and ρ_3, and thus $\widetilde{\mathcal{W}}$ is a sprocket. □

5.3.3 Conjectures

If one is only interested in checking for sprockets, Propositions 5.3.1 and 5.3.4 together reduce the task to considering only max-intersection faces of $\Delta(\mathcal{C})$:

Corollary 5.3.5. *Let $\mathcal{C}$ be a code. If $\mathcal{C}$ has a sprocket, then $\mathcal{C}$ has a sprocket $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ where $\tau \notin \mathcal{C}$ but τ is a max-intersection face of $\Delta(\mathcal{C})$.*

One may observe that searching for sprockets parallels searching for local obstructions, as both tasks entail examining certain max-intersection faces.

Arising naturally from this observation is the question of whether searching for other types of wheels also reduces to checking missing max-intersection faces. The results of [10] lead us to believe that this is indeed the case. Namely, since a wheel of any kind guarantees nonconvexity, while max-intersection-completeness guarantees convexity, it seems as though a wheel's existence in a code $\mathcal{C}$ must force the exclusion of some max-intersection face from $\mathcal{C}$. We summarize this thinking in the following conjecture:

Conjecture 5.3.6. *Let $\mathcal{C}$ be a code. If $\mathcal{C}$ has a wheel, then $\mathcal{C}$ has a wheel $\mathcal{W} = (\sigma_1, \sigma_2, \sigma_3, \tau)$ where $\tau \notin \mathcal{C}$ but τ is a max-intersection face of $\Delta(\mathcal{C})$. Furthermore, the rim of any wheel cannot be a codeword of $\mathcal{C}$.*

5.4 Codes on 5 and 6 Neurons

The combination of wheels and decomposable codes allows us to completely classify all codes on 5 neurons.(cf. [17, Theorem 3.1]):

Theorem 5.4.1. *A code $\mathcal{C}$ on 5 neurons is convex if and only if it has no local obstructions and no wheels.*

Proof. $(\Rightarrow)$ This implication follows from Proposition 2.2.9 and Theorem 5.1.3.

$(\Leftarrow)$ Let $\mathcal{C}$ be a code on 5 neurons, and let $\mathcal{C}^*$ be the code from Example 2.2.13. We condition based on whether $\Delta(\mathcal{C})$ is isomorphic to $\Delta(\mathcal{C}^*)$.

If $\Delta(\mathcal{C})$ is not isomorphic to $\Delta(\mathcal{C}^*)$, then by Proposition 3.2.10 we have that the minimal code (recall Definition 2.2.4) of $\Delta(\mathcal{C})$ is convex. Since $\mathcal{C}$ has no local obstructions, $\mathcal{C}$ must contain all its mandatory codewords and thus the minimal code. Therefore, by Lemma 3.2.8, $\mathcal{C}$ is convex.

If $\Delta(\mathcal{C})$ is indeed isomorphic to $\Delta(\mathcal{C}^*)$, then $\mathcal{C}^* \subseteq \mathcal{C}$ (recall that $\mathcal{C}^*$ is the minimal code of $\Delta(\mathcal{C}^*)$). By Proposition 5.2.13 any code containing $\mathcal{C}^*$ but not any of 1, 234, or 345 must have a partial wheel. Therefore, since $\mathcal{C}$ does not have a wheel, $\mathcal{C}$ contains at least one of 1, 234, or 345, and so by Proposition 3.2.10 is convex. $\square$

Having shown that wheels – together with local obstructions – completely characterize non-convexity for codes on up to 5 neurons, we now turn our attention to codes on 6 neurons. Due to the large number of such codes, we restrict our analysis to two subsets of codes on 6 neurons: (1) codes with seven or fewer maximal codewords, and (2) codes whose simplicial complex is pure (recall that a simplicial complex is pure if all of its facets have the same dimension). Also, like the classification process of [17], for each Δ we consider only its minimal code $\mathcal{C}_{\text{Min}}(\Delta)$. Indeed, when such a code is convex, then all codes with neural complex equal to Δ (and no local obstructions) are convex (recall Remark 2.2.7).

Surveying the codes with ≤ 7 maximal codewords had promising results, as we found roughly 300 minimal codes to be nonconvex due to a wheel (see Table 5.1). Moreover, wheels completely characterize nonconvexity for 6-neuron minimal codes with up to four maximal codewords. The search for minimal codes with pure simplicial complex was less fruitful, as only six such codes with wheels were found (see Table 5.2).

Table 5.1: Minimal Codes on 6 Neurons with k Maximal Codewords, $4 \le k \le 7$.

	Four	Five	Six	Seven
Total	210	691	1578	2578
Red. or Dec.	204	482	528	341
Max-∩	4	79	399	909
Wheel	2	36	118	159
Unknown	0	94	533	1169

This table gives an analysis of those minimal codes on 6 neurons with at least 3 and no more than 7 maximal codewords. The columns are the codes sorted by the dimension of $\Delta(\mathcal{C})$. **Total** refers to the total number of minimal codes for which $\Delta(\mathcal{C})$ is pure of that dimension. **Red. or Dec.** is the number of codes that can be either reduced or decomposed into a code on fewer than 6 neurons. Of the codes that could not be reduced or decomposed, **Max-∩** is the number of codes for which every max-intersection face is mandatory (implying convexity), while **Wheel** refers to those codes that contain a wheel (implying nonconvexity). Lastly, **Unknown** refers to those codes that do not belong in any of the first three categories, and so have unknown convex classification.

Our workflow to obtain the tables was as follows. We first used `nauty` [24] to enumerate all connected simplicial complexes on 6 vertices (up to isomorphism). Next, we used `SageMath` [25] to compute the minimal code of each simplicial complex (see [12, Algorithm 4.1]). Finally,

Table 5.2: Pure Minimal Codes on 6 Neurons.

	Pure Dim 1	Pure Dim 2	Pure Dim 3	Pure Dim 4	Pure Dim 5
Total	112	2101	150	5	1
Red. or Dec.	51	153	40	4	1
Max-$\cap$	61	944	32	1	0
Wheel	0	0	6	0	0
Unknown	0	1004	72	0	0

The table gives an analysis of those minimal codes $\mathcal{C}$ on 6 neurons whose simplicial complex $\Delta(\mathcal{C})$ is pure. The columns are the codes sorted by the dimension of $\Delta(\mathcal{C})$. The rows have the same meaning as they do in Table 5.1.

we classified a given code $\mathcal{C}$ by performing the steps in Algorithm 2.

Algorithm 2: Classifying codes on 6 neurons

INPUT: A code $\mathcal{C}$ on 6 neurons.

OUTPUT: "Reducible or decomposable", "Max-intersection-complete", "Wheel", or "Unknown".

STEPS:

1. Determine whether $\mathcal{C}$ is reducible or decomposable. (We saw in Chapter 3 – specifically, Proposition 3.1.3 and Theorem 3.2.4 – that, in terms of convexity, such codes are equivalent to codes on fewer neurons.) If not, proceed to the next step.
2. Determine whether $\mathcal{C}$ is max-intersection-complete. (Such codes are convex, by Proposition 2.2.6.) If not, proceed to the next step.
3. Determine whether $\mathcal{C}$ has a combinatorial wheel – that is, if it satisfies the criterion from at least one of Propositions 5.2.5, 5.2.9, and 5.2.12. If not, the convexity status of $\mathcal{C}$ is "Unknown".

A package containing the scripts used in the Algorithm, as well as data on which specific codes have wheels and what those wheels are, is available on `GitHub` (DOI: 10.5281/zenodo.4662199).

Remark 5.4.2 (Codes with up to three maximal codewords)**.** For codes with at most three maximal codewords – and any number of neurons – convexity has been fully characterized: such codes are convex if and only if they have no local obstructions [26]. We therefore do not include codes with three or fewer maximal codewords in Table 5.1.

We conclude from Table 5.1 and Remark 5.4.2 that for a simplicial complex Δ on 6 vertices with up to 4 facets, the minimal code $\mathcal{C}_{\min}(\Delta)$ is convex if and only if it has no local obstructions and no wheel. We also see from the table that the number of codes with wheels appears to increase with the number of maximal codewords – and we already have found wheels in over 300 codes. Wheels are therefore surprisingly efficient in detecting nonconvexity.

Going forward, there are still many codes whose convexity status is unknown (as seen in Tables 5.1 and 5.2). Moreover, we would like to continue the classification for codes with more maximal codewords, but this task is currently computationally challenging. Nevertheless, we hope that results like those in Section 5.3 will make this approach more tractable.

6. CONCLUSIONS

Our work on the wheel, while shedding more light on the question of which codes are convex, has raised several questions of its own. To begin, the theory for detecting wheels is not complete. Unlike with the local obstruction, we do not have any way to guarantee that a code does not contain a wheel, short of proving the code is convex. Therefore, learning a necessary and sufficient condition for a wheel is the foremost remaining problem.

However, we should be mindful of other issues that are tangled up in the question of finding a necessary and sufficient condition for wheels. In the case of the local obstruction, the signature for the necessary and sufficient condition may be found in the set of max-intersection faces not in $\mathcal{C}$, which makes sense in light of [10, Theorem 1.2]. As wheels are also a type of nonconvexity, it should follow that the presence of a wheel must also prevent max-intersection-completeness. Therefore, we expect that the signature for a wheel in a code will always present itself in the max-intersection faces. Proving Conjecture 5.3.6 seems the correct starting point for this work.

Another twist concerns adding more spokes in the wheel. Throughout this dissertation, we have only considered wheels with three spokes, with the condition being that a line in the rim τ must pass through each spoke. However, there are more complex versions of the wheel that act in higher dimensions. From [6, Theorem 1.1], the phenomenon holds not just for when there are three spokes and a line, but $d+1$ convex sets and a $d-1$-dimensional hyperplane. From this result, [6] constructs an infinite family of nonconvex codes with neither local obstructions nor wheels. Generalizing wheels to include these larger phenomena seems like the natural next step. In fact, it might be that making this generalization is necessary to finding the necessary and sufficient condition for wheels. It is possible that when a wheel bubbles up to the set of max-intersection faces, that what bubbles up is no longer an object with three spokes, but more.

Lastly, we recall that wheels only prevent a code from being open convex. It is possible for a code to have a wheel and yet still be closed convex. Therefore, we ask if there is a counterpart to the wheel that, independent of the local obstruction, can guarantee closed convexity.

REFERENCES

[1] J. O'Keefe, "Place units in the hippocampus of the freely moving rat," *Experimental neurology*, vol. 51, no. 1, pp. 78–109, 1976.

[2] C. Curto, V. Itskov, A. Veliz-Cuba, and N. Youngs, "The neural ring: an algebraic tool for analyzing the intrinsic structure of neural codes," *Bulletin of mathematical biology*, vol. 75, no. 9, pp. 1571–1611, 2013.

[3] C. Curto, V. Itskov, K. Morrison, Z. Roth, and J. L. Walker, "Combinatorial neural codes from a mathematical coding theory perspective," *Neural computation*, vol. 25, no. 7, pp. 1891–1925, 2013.

[4] C. Curto, "What can topology tell us about the neural code?," *Bulletin of the American Mathematical Society*, vol. 54, no. 1, pp. 63–78, 2017.

[5] R. A. Jeffs and I. Novik, "Convex union representability and convex codes," *International Mathematics Research Notices*, 2018.

[6] R. A. Jeffs, "Sunflowers of convex open sets," *Advances in Applied Mathematics*, vol. 111, p. 101935, 2019.

[7] R. A. Jeffs, "Morphisms of neural codes," *SIAM Journal on Applied Algebra and Geometry*, vol. 4, no. 1, pp. 99–122, 2020.

[8] A. Chen, F. Frick, and A. Shiu, "Neural codes, decidability, and a new local obstruction to convexity," *SIAM Journal on Applied Algebra and Geometry*, vol. 3, no. 1, pp. 44–66, 2019.

[9] C. Curto, E. Gross, J. Jeffries, K. Morrison, M. Omar, Z. Rosen, A. Shiu, and N. Youngs, "What makes a neural code convex?," *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 222–238, 2017.

[10] J. Cruz, C. Giusti, V. Itskov, and B. Kronholm, "On open and closed convex codes," *Discrete & Computational Geometry*, vol. 61, no. 2, pp. 247–270, 2019.

[11] C. Giusti and V. Itskov, "A no-go theorem for one-layer feedforward networks," *Neural Computation*, vol. 26, no. 11, pp. 2527–2540, 2014.

[12] C. Lienkaemper, A. Shiu, and Z. Woodstock, "Obstructions to convexity in neural codes," *Advances in Applied Mathematics*, vol. 85, pp. 31–59, 2017.

[13] R. A. Jeffs, "Convexity of neural codes." Undergraduate Thesis, Harvey Mudd College, 2016.

[14] A. S. Jarrah, R. Laubenbacher, B. Stigler, and M. Stillman, "Reverse-engineering of polynomial dynamical systems," *Advances in Applied Mathematics*, vol. 39, no. 4, pp. 477–489, 2007.

[15] A. Veliz-Cuba, "An algebraic approach to reverse engineering finite dynamical systems arising from biology," *SIAM Journal on Applied Dynamical Systems*, vol. 11, no. 1, pp. 31–48, 2012.

[16] S. Güntürkün, J. Jeffries, and J. Sun, "Polarization of neural rings," *Journal of Algebra and Its Applications*, p. 2050146, 2019.

[17] S. A. Goldrup and K. Phillipson, "Classification of open and closed convex codes on five neurons," *Advances in Applied Mathematics*, vol. 112, p. 101948, 2020.

[18] E. Magaña and K. Phillipson, "Classifying neural codes as closed or open convex," *In preparation*, 2020.

[19] A. R. de Perez, L. F. Matusevich, and A. Shiu, "Neural codes and the factor complex," *Advances in Applied Mathematics*, vol. 114, p. 101977, 2020.

[20] R. A. Jeffs, M. Omar, N. Suaysom, A. Wachtel, and N. Youngs, "Sparse neural codes and convexity," *Involve, a Journal of Mathematics*, vol. 12, no. 5, pp. 737–754, 2019.

[21] C. Curto, E. Gross, J. Jeffries, K. Morrison, Z. Rosen, A. Shiu, and N. Youngs, "Algebraic signatures of convex and non-convex codes," *Journal of Pure and Applied Algebra*, vol. 223, no. 9, pp. 3919 – 3940, 2019.

[22] E. Petersen, N. Youngs, R. Kruse, D. Miyata, R. Garcia, and L. D. García Puente, "Neural ideals in SageMath," in *Mathematical Software – ICMS 2018* (J. H. Davenport, M. Kauers, G. Labahn, and J. Urban, eds.), (Cham), pp. 182–190, Springer International Publishing, 2018.

[23] R. A. Jeffs, C. Lienkaemper, and N. Youngs, "Order-forcing in neural codes," *arXiv preprint arXiv:2011.03572*, 2020.

[24] B. D. McKay and A. Piperno, "Practical graph isomorphism, II," *Journal of Symbolic Computation*, vol. 60, no. 0, pp. 94–112, 2014.

[25] The Sage Developers, *SageMath, the Sage Mathematics Software System (Version 6.7)*, 2015. `https://www.sagemath.org`.

[26] K. Johnston, A. Shiu, and C. Spinner, "Neural codes with three maximal codewords: Convexity and minimal embedding dimension," *arXiv preprint arXiv:2008.13192*, 2020.