Gröbner Bases and the Neural Ideal

Jessica Liu

July 21, 2016

Abstract

The spatial location of an organism is encoded by place cells, neurons that correspond to convex receptive fields in a stimulus space. The brain must infer properties of a stimulus space using only information extracted from a neural code, leading to the question of how a map of receptive fields can be extracted from neural codes. In a recent paper, Curto, Itskov, Veliz-Cuba, and Youngs develop methods to tackle this question. They define a **neural ring** and a **neural ideal**, algebraic objects that encode the combinatorial data of a neural code. A specific generating set of the neural ideal, called the **canonical form**, translates to a minimal description of the relationships between receptive fields in the stimulus space. Knowing these minimal relationships makes it much easier to understand the stimulus space structure associated with a given code. There are two known algorithms to find the canonical form, the more recent version of which has been implemented in MatLab and SageMath. Both algorithms are extremely computationally inefficient when the size of the code is large. In contrast, Petersen et al. have found that computing another generating set of the neural ideal, the universal Gröbner basis, is vastly more efficient, suggesting the possibility of using Gröbner bases to improve the canonical form algorithms. Here we find that the canonical form is the smallest generating set of the neural ideal that can be a Gröbner basis, which in some cases allows us to compute the Gröbner basis as a shortcut to computing the canonical form. Along the way, we develop a representation of pseudo-monomials as hypercubes.

1 Introduction

The brain encodes and processes information with neurons, cells that fire in response to some stimuli. In order to understand how the brain works, scientists must not only study the physical processes the underlie brain function; they must also understand how the brain represents information abstractly. In 2014, the Nobel Prize in Medicine was awarded for the discovery of place cells, neurons that encode the spatial location of an organism. Every place cell is associated with a convex region in a stimulus space called a place field, and will fire if and only if the organism is in the corresponding place field. However, although we know the basics of how individual place cells encode information, the brain's understanding of spatial awareness depends on the inputs from many different place cells that may have overlapping place fields.

In *The Neural Ring* [2], Curto et al. use the tools of algebraic geometry to construct methods of inferring the relationships between place fields from only the combinatorial information of a neural code. Given a neural code, one can generate a *Neural Ideal* which has a generating set called the canonical form that gives a minimal description of the relationships between receptive fields.

One algorithm to compute the canonical form is described in *The Neural Ring*, and another is described in subsequent paper *Neural Ring Homomorphisms and Maps between Neural Codes* [1]. Both algorithms have been implemented in Sage Math by Peterson et al [4]; however, they are both too slow to make it feasible to compute the canonical forms of larger codes. In contrast, Peterson et al. found that computing the Gröbner basis, another generating set of the Neural Ideal, has a much faster runtime. Furthermore, in some cases the Gröbner basis and the canonical form are equal, suggesting that computing the Gröbner basis can be a shortcut to computing the canonical form.

In this paper, we further explore the relationship between the canonical form and the universal Gröbner basis of a neural idea. We find that if the canonical form is a Gröbner basis, then it is a reduced Gröbner basis, and, moreover, it is the universal Gröbner basis (this was first asserted by Peterson et al.). Conversely, if the universal Gröbner basis only consists of so-called pseudo-monomials, then it is equal to the canonical form. Additionally, if a nontrivial code is complement-complete, then the canonical form of its neural ideal is *not* a Gröbner basis.

2 Background

The following background is adapted from *The Neural Ring* and *Ideals, Varieties and Algorithms*.

Definition 2.1. Given a set of neurons labeled {1,..., n}, a **neural code** on n neurons is a set of binary firing patterns $C \subset \{0,1\}^n$ i.e. a set of binary strings of neural activity. An element $c \in C$ of a neural code is a **codeword**, where each codeword corresponds to a subset of neurons

$$\text{supp}(c) = \{i \in [n] | c_i = 1\} \subset [n].$$

The entire code can be identified with a set of subsets of neurons,

$$\text{supp}(C) = \{\text{supp}(c) | c \in C\} \subset 2^{[n]}.$$

Notice that we discard the details of timing or rate of neural activity. The neural code as defined here is what is known as a *combinatorial code*.

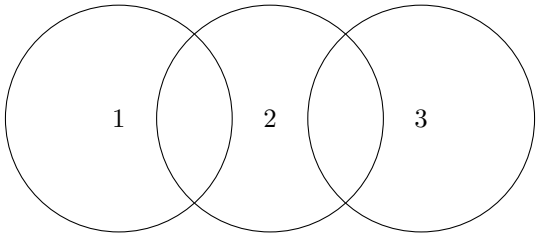
2.1 Receptive Fields

In many areas of the brain, neurons can be associated with **receptive fields** in a stimulus space. We are particular interested in the receptive fields of *place cells*, which are neurons that fire in response to a animal's location. More specifically, each individual place cell is associated with a **place field**, a convex region of the animal's physical environment where the place cell has a high firing rate.

Example 2.1. Let us consider the following code on 3 neurons: $C = \{100, 110, 010, 011, 001\}$, where each neuron corresponds to a place field in $\mathbb{R}^2$.

The codewords 100, 010, and 001 tell us that the place fields associated with all three neurons

each have some section that does not overlap with another place fields. Additionally, the codewords 110 and 011 tell us that the place fields of the first and second neuron intersect, and the place fields of the second and third place field intersect. The codeword 111 is *not* in the code, so we know that there is no region of the stimulus space where all three place fields intersect. Now we have enough information to draw a picture of the place fields:



For a set of receptive fields $\mathcal{U} = \{U_i\}_{i=1}^n$ in a stimulus space X, the **receptive field code** $C(\mathcal{U})$ is the neural code that encodes for the stimulus space structure of X. We are interested in knowing what information about the stimulus space can be inferred from just $C(\mathcal{U})$, without knowing anything prior about the receptive fields. While in the above example the stimulus space structure can be directly deduced from the code, as the number of neurons grows the problem of determining the proper relationships between receptive fields quickly becomes intractable. In *The Neural Ring*, Curto et al. use objects from algebraic geometry to develop a method to extract a minimal description of the receptive field structure.

2.2 The Neural Ring

Before defining the neural ring, we briefly review some necessary algebraic geometry concepts.

Ideals: Let R be a commutative ring. A subset $I \subset R$ is an **ideal** of R if it has the following properties:

1. I is a subgroup of R under addition.

2. If $a \in I$, then $ra \in I$ for all $r \in R$.

An ideal I is said to be **generated** by a set A, and we write $I = \langle A \rangle$, if I is the set of all finite combinations of elements of A with coefficients in R.

Varieties: Let k be a field, n the number of neurons, and $k[x_1, \ldots, x_n]$ a polynomial ring with one indeterminate x_i for each neuron. In the neural activity space k^n, each point $v = (v_1, \ldots, v_n) \in k^n$ is a vector tracking the state v_i of each neuron. We will regard neurons as having only two states, firing and silent, and so we choose $k = \mathbb{F}_2 = \{0, 1\}$. For a polynomial $f \in \mathbb{F}_2[x_1, \ldots, x_n]$ and a point

$v \in \{0,1\}^n$, we will denote $f(v)$ as the evaluation of f by setting $x_i = v_i$ whenever x_i appears in f. Let $J \subset \mathbb{F}_2[x_1, \ldots, x_n]$ be an ideal, and define the variety

$$V(J) = \{v \in \{0,1\}^n | f(v) = 0 \text{ for all } f \in J\}.$$

Similarly, for a subset of the neural activity space $S \subset \{0,1\}^n$, we can define the ideal of this subset as

$$I(S) = \{f \in \mathbb{F}_2[x_1, \ldots, x_n] | f(v) = 0 \text{ for all } f \in J\}.$$

Now we can define a more specific kind of ideal that contains the combinatorial information of a neural code.

Definition 2.2. Let $C \subset \{0,1\}^n$ be a neural code. I_C is defined as the ideal corresponding to the set of polynomials that vanish on all codewords of C.

$$I_C = I(C) = \{f \in \mathbb{F}_2[x_1, \ldots, x_n] | f(c) = 0 \text{ for all } c \in C\}.$$

Notice that for all C, I_C contains the ideal generated by the **Boolean relations**, defined as

$$B = \langle x_1^2 - x_1, \ldots, x_n^2 - x_n \rangle.$$

While I_C contains all of the combinatorial information of the code, we would like to be able to represent the code without the Boolean relations, which are redundant. The **neural ideal**, J_C, can be defined via an explicit set of generating relations.

The Neural Ideal:
For any $v \in \{0,1\}^n$, consider ρ_v, defined as

$$\rho_v = \prod_{i=1}^{n}(1 - v_i - x_i) = \prod_{\{i|v_i=1\}} x_i \prod_{\{j|v_j=0\}} (1 + x_j) = \prod_{\{i \in \text{supp}(v)\}} x_i \prod_{\{j \notin \text{supp}(v)\}} (1 - x_j).$$

Notice that for all v, the polynomial ρ_v satisfies $\rho_v(v) = 1$ and $\rho_v(x) = 0$ for all $x \neq v$. Therefore we think of ρ_v as the characteristic function for v. For a code C, the **neural ideal** J_C is generated by all ρ_v for $v \notin C$.

$$J_C = \langle \{\rho_v | v \notin C\} \rangle.$$

Note that if $C = \{0,1\}^n$, then $J_C = \emptyset$.

The ideal I_C and the neural ideal J_C have the following relation, as described in *The Neural Ring*.

Lemma 2.1 (Curto et al.)**.** Let $C \subset \{0,1\}^n$ be a neural code. Then $I_C = J_C + B$, where $\mathcal{B}$ is the set of boolean relations in $\mathbb{F}_2[x_1, \ldots, x_n]$.

This tells us that for a code C, the structure of I_C is captured by J_C, since the Boolean relations $\mathcal{B}$ are in the ideal of any neural code.

It turns out that we can interpret elements of I_C (and thus J_C) in terms of relationships between receptive fields. First we need the following notation: for any $\sigma \subset [n]$, define

$$U_\sigma = \bigcap_{i \in \sigma} U_i, \text{ and } x_\sigma = \prod_{i \in \sigma} x_i$$

Lemma 2.2 (Curto et al.)**.** Let X be a stimulus space, let $\mathcal{U} = \{U_i\}_{i=1}^n$ be a collection of open sets in X, and consider the receptive field code $C = C(\mathcal{U})$. Then for any pair of subsets $\sigma, \tau \in [n]$,

$$x_\sigma \prod_{i \in \tau} (1 - x_i) \in I_C \iff U_\sigma \subseteq \bigcup_{i \in \tau} U_i.$$

Of course, not all of the relations in the ideal are necessary to infer the receptive field structure. For example, $x_1(1 - x1) \to U_1 \subseteq U_1$ is clearly redundant. Another example is the set of relations $x_1(1-x_3)(1-x_4) \implies U_1 \subseteq U_3 \bigcup U_4$ and $x_1x_2(1-x_3)(1-x_4) \implies U_1 \bigcap U_2 \subseteq U_3 \bigcup U_4$. The first relation implies the second, so we obviously don't need both of them to describe the receptive field relationships. In fact, we can define "minimal" generating set of the neural ideal that captures the essential receptive field structure without redundancy.

2.3 The Canonical Form

First we introduce some definitions and notation.

Definition 2.3. For some $f \in \mathbb{F}_2[x_1, \ldots, x_n]$, f is a **pseudo-monomial** if f has the form

$$f = \prod_{i \in \sigma} x_i \prod_{j \in \tau} (1 + x_j).$$

for some $\sigma, \tau \subset [n]$ with $\sigma \cap \tau = \emptyset$.

An ideal $J \subset \mathbb{F}_2[x_1, \ldots, x_n]$ is a **pseudo-monomial ideal** if J can be generated by a set of pseudo-monomials.

Let $J \subset \mathbb{F}_2[x_1, \ldots, x_n]$ be an ideal, and let f be a pseudo-monomial in J. We say that f is a **minimal** pseudo-monomial of J if there does not exist another pseudo-monomial $g \in J$, where $\deg(g) < \deg(f)$, such that $f = gh$ for some $h \in \mathbb{F}_2[x_1, \ldots, x_n]$.

The set of *all* minimal pseudo-monomials in a pseudo-monomial J gives us a unique and compact description of J, called the canonical form.

Definition 2.4. Let J be a pseudo-monomial ideal. The **canonical form** of J is the set $CF(J) = \{f_1, \ldots, f_l\}$ that is the set of *all* minimal pseudo-monomials of J.

Note that each generator of the neural ideal, ρ_v, is a multiple of some element of the canonical form. Thus the canonical form is a generating set of the neural ideal. The canonical form tells us the minimal relationships that must be satisfied by any receptive field representation of a code as $C = C(\mathcal{U})$, as proven in the following theorem from *The Neural Ring*:

Theorem 2.1 (Curto et al.)**.** Let $C \subset \{0,1\}^n$ be a neural code, and let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be any collection of opens sets in a nonempty stimulus space X such that $C = C(\mathcal{U})$. The canonical form of J_C is:

$$J_C = \langle \{x_\sigma | \sigma \text{ is minimal w.r.t } U_\sigma = \emptyset\},$$

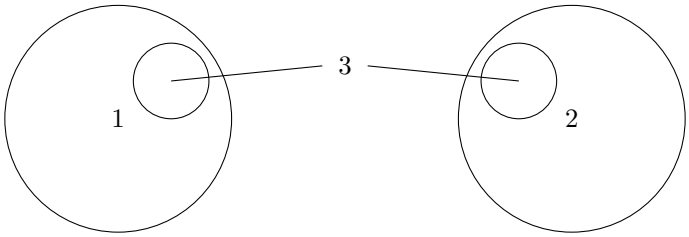
$$\{x_\sigma \prod_{i \in \tau}(1+x_i) | \sigma, \tau \neq \emptyset, U_\sigma \neq \emptyset, \bigcup_{i\in\tau} U_i \neq X, \text{ and } \sigma, \tau \text{ are each minimal w.r.t. } U_\sigma \subset \bigcup_{i\in\tau} U_i\},$$

$$\{\prod_{i\in\tau}(1+x_i) | \tau \text{ is minimal w.r.t } X \subset \bigcup_{i\in\tau} U_i\}\rangle.$$

Example 2.2. Let $C = \{000, 100, 010, 101, 011\}$. The canonical form of J_C is $\langle x_1x_2, x_3(1+x_1)(1+x_2)$, from which we can read off the following receptive-field relationships:

$$U_1 \bigcap U_2 = \emptyset, \qquad U_3 \subset U_1 \bigcup U_2$$

From these relationships we can draw a realization of the receptive fields:



The Neural Ring contains an algorithm to compute the canonical form, using the primary decomposition of the neural ideal. In *Neural Ring Homomorphisms and Maps Between Neural Ideals*, Curto and Youngs present an alternative algorithm that computes the canonical form iteratively, based on the operation of adding a codeword to a code. Youngs has implemented the iterative algorithm in Matlab, and both algorithms have been implemented in SageMath by Petersen et al.

Petersen et al. also provide a table of runtime comparisons between the three implementations. Both implementations of the iterative algorithm vastly outperform the primary decomposition algorithm, but they are still slow enough that computation of the canonical form quickly becomes

infeasible as the number of neurons grows. Petersen et al. found that computing the universal Gröbner basis of the neural ideal is significantly faster, especially as the size of the code grows. Knowing this, we turn our attention to Gröbner bases, with the goal of using them to improve the runtime of computing the canonical form.

2.4 Gröbner Bases

Before we can work with Gröbner bases, we must first define some necessary terminology. The following definitions are adapted from *Ideals, Varieties and Algorithms*.

Definition 2.5. A **monomial ordering** $>$ on $k[x_1, \ldots, x_n]$ is a relation $>$ on the set of monomials $\{x^\alpha, \alpha \in \mathbb{Z}^n_{\geq 0}\}$, or, equivalently, on $\mathbb{Z}^n_{\geq 0}$, satisfying:

(i) $>$ is a total (or linear) ordering on $\mathbb{Z}^n_{\geq 0}$.

(ii) If $\alpha > \beta$ and $\gamma \in \mathbb{Z}^n_{\geq 0}$, then $\alpha + \gamma > \beta + \gamma$.

(iii) $>$ is a well-ordering on $\mathbb{Z}^n_{\geq 0}$.

Once we define a monomial ordering on a polynomial ring, it makes sense to talk about the **leading monomial** of a polynomial in the ring. In this paper the field we are working with is $\mathbb{F}_2$, so the leading monomial and the leading term of a polynomial are the same since all terms have coefficient 1. Let f be a nonzero polynomial in $k[x_1, \ldots, x_n]$, and fix a monomial ordering on $k[x_1, \ldots, x_n]$. We denote the leading term of f as $\mathrm{LT}(f)$.

For an ideal $I \subset \mathbb{F}_2[x_1, \ldots, x_n]$, the **ideal of leading terms** is defined as follows. Let I be an ideal other than $\langle 0 \rangle$, and fix a monomial ordering on $k[x_1, \ldots, x_n]$. Then we denote by $\mathrm{LT}(I)$ the set of leading terms of nonzero elements of I. Thus,

$$\mathrm{LT}(I) = \{x^\alpha | \text{ there exists } f \in I \backslash \{0\} \text{ with } \mathrm{LT}(f) = x^\alpha\}.$$

We denote by $\langle \mathrm{LT}(I) \rangle$ the ideal generated by the elements of $\mathrm{LT}(I)$.

Now we are ready to define a Gröbner basis.

Definition 2.6. Fix a monomial order on on the polynomial ring $k[x_1, \ldots, x_n]$. A finite subset $G = \{g_1, \ldots, g_t\}$ of an ideal $I \subset k[x_1, \ldots, x_n]$ different from $\{0\}$ is said to be a **Gröbner basis** if

$$\langle \mathrm{LT}(g_1), \ldots, \mathrm{LT}(g_t) \rangle = \langle \mathrm{LT}(I) \rangle.$$

Equivalently, a set $\{g_1, \ldots, g_t\} \subseteq I$ is a Gröbner basis of I if and only if the leading term of any element of I is divisble by one of the $\mathrm{LT}(g_i)$.

One of the more useful properties of a Gröbner basis is that given a polynomial f and a Gröbner basis G, the remainder of f when divided by the set of elements in G is uniquely determined. It follows that for an ideal $I \subset k[x_1, \ldots, x_n]$, a polynomial $f \in k[x_1, \ldots, x_n]$ is in I if and only if the remainder of division on f by a Gröbner basis of I is 0.

We are interested in Gröbner bases because for some codes, the Gröbner basis of the neural ideal is equal the canonical form of the neural ideal.

When most computer algebra systems compute the Gröbner basis of an ideal, they compute the *reduced* Gröbner basis with respect to a given monomial ordering. A Gröbner basis G is a **reduced Gröbner basis** for all $g \in G$, no trailing term of any $g \in G$ is divisible by the leading term of any element of G. For any given monomial ordering, the reduced Gröbner basis of an ideal is unique, but changing the monomial ordering can change the reduced Gröbner basis of an ideal. A Gröbner basis of an ideal that is a Gröbner basis for all monomial orderings is a **universal Gröbner basis**.

Lemma 2.3. [3] If the canonical form of a neural ideal is a Gröbner basis, then the canonical form is a universal Gröbner basis.

Proof outline. Since the leading term of a pseudo-monomial is consistent across all monomial orderings, we know that if the canonical form is a Gröbner basis for one monomial ordering, then it is a Gröbner basis for all monomial orderings, and thus is a *universal Gröbner basis.* □

An ideal can admit multiple Gröbner bases that are universal, but in this paper we will use the term "the universal Gröbner basis " to refer to a specific basis that *is* unique for each ideal, and is defined as the following:

Definition 2.7. Let I be an ideal. The **universal Gröbner basis** is the union of all the reduced Gröbner bases of I with respect to any monomial order.

Note that *the* universal Gröbner basis is an instance of *a* universal Gröbner basis. Also note that the polynomials in the universal Gröbner basis are always square-free, because they are the result of applying Buchberger's algorithm to a set of pseudo-monomials.

3 Results

Recall that our ultimate goal is to use Gröbner bases to find a faster way to compute the canonical form. Our results, together with those of Peterson et al., form the first steps in this direction. Our main result, proven in Section 3.2, is that if a reduced Gröbner basis contains only pseudo-monomials, then it must be equal to the canonical form (Corollary 3.2). In order to prove this, we first build further understanding of pseudo-monomials in Section 3.1 by representing them as hypercubes. Finally, in Section 3.2, we prove that for codes where all codewords have a corresponding complement in the code, the canonical form is not a Gröbner basis(Theorem 3.4), thereby resolving a conjecture posed by Peterson et al.

3.1 Pseudo-monomials and hypercubes

The neural ideal and its canonical form are defined in terms of pseudo-monomials, which nicely represent the information contained in the neural code and the receptive fields (recall Lemma 2.2). Pseudo-monomials turn out to have many nice properties that make them much easier to work with than general polynomials. We prove some properties below.

The following result states that the monomials in the expansion of a pseudo-monomial $f = x_\sigma \prod_{i\in\tau}(1+x_i)$ are in bijection with the *powerset of* τ.

Example 3.1. Let $f = x_1(1+x_2)(1+x_3) = x_\sigma \prod_{i\in\tau}(1+x_i)$, where $\tau = \{2,3\}$. Note that

$$P(\tau) = \{\{2,3\},\{2\},\{3\},\emptyset\}$$

and the expansion of f is

$$x_1x_2x_3 + x_1x_2 + x_1x_3 + x_1.$$

Clearly, we can define a bijection between the elements in $P(\tau)$ and the terms of f.

Proposition 3.1. Let $f = x_\sigma \prod_{i\in\tau}(1+x_i)$ be a pseudo-monomial. Then we can write f as

$$f = \sum_{\gamma\in P(\tau)} x_\sigma x_\gamma,$$

where $P(\tau)$ is the powerset of τ.

Proof. Follows from Binomial theorem. □

This way of representing pseudo-monomials as the sum of its terms makes it clear that each term of f corresponds to an element in the power set of τ, and the following properties of pseudo-monomials obvious:

Lemma 3.1. Let f be a pseudo-monomial such that $f = x_\sigma \prod_{i\in\tau}(1+x_i)$. Then:

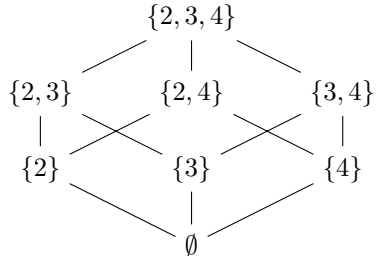
1. The leading term of f is $x_\sigma x_\tau$.
2. All terms of f divide $x_\sigma x_\tau$.
3. x_σ divides all terms of f.

Notice that like the power set of τ, the terms of f form a partially ordered set, with the partial order defined by the degree of the terms. Thus we can represent f using a Hasse diagram, with the terms of f as the vertices. Since there is an obvious bijection between the set of terms of f and the powerset of τ, the Hasse diagram turns out to be a *hypercube* of dimension $|\tau|$.

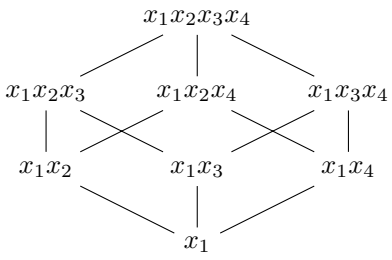
Definition 3.1. Let f be a pseudo-monomial, and let $f = x_\sigma \prod_{i\in\tau}(1+x_i)$. We define the **hypercube** of f as the Hasse diagram of the set of terms in f, where the partial ordering of the terms is defined by the degree of the terms.
Note: Since the monomials of the pseudo-monomial $g = \prod_{i\in\beta}(1+x_i)$ for some set β correspond directly to the elements of the power-set of β, we will refer to the hypercube of g as the hypercube of $P(\beta)$.

By Lemma 3.1, the maximal element of the partially ordered set is $x_\sigma x_\tau$, and the minimal element is x_σ.

Example 3.2. Let $f = x_1(1+x_2)(1+x_3)(1+x_4)$. In this case, $\sigma = \{1\}$ and $\tau = \{2,3,4\}$.



Hasse diagram of the powerset of τ



hypercube of f

The representation of pseudo-monomials as hypercubes is related, but not equivalent to the interpretation of pseudo-monomials on the Boolean lattice discussed in *The Neural Ring.* In *the Neural Ring*, the authors define the following variety for any $a \in \{0,1,\star\}^n$, where the $\star$ can be thought of as a "wildcard" element:

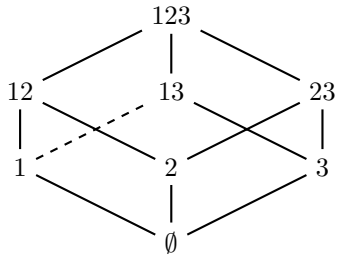
$$V_a = \{v \in \{0,1\}^n | v_i = a_i \text{ for all } i \text{ s.t. } a_i \neq \star\} \subset \{0,1,\}^n$$

.
For a pseudo-monomial f, they associate an element $b \in \{0,1,\star\}^n$ to f if

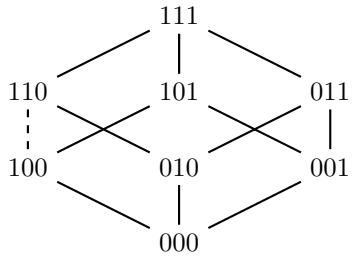
$$f = \prod_{\{i|b_i=1\}} x_i \prod_{\{j|b_j=0\}} (1+x_j).$$

Every element b corresponds to some interval of the Boolean lattice in the complement of C. Like how the vertices of a hypercube of a pseudo-monomial are in bijection with the powerset of τ, the elements of the interval associated with b are in bijection with the powerset of $\{i|b_i = \star\}$.

Example 3.3. Let $f = x_1(1+x_3)$. The hypercube of f shown below as the dashed line. The element associated with f is $1 \star 0$. On the Boolean lattice, f describes the Boolean lattice interval that corresponds to the variety compatible with $1 \star 0$, which is $[110, 100]$



hypercube of $P(\{1,2,3\})$



Boolean lattice

Note that the dimension of the hypercube of f is $|\tau|$, while the dimension of the interval of the Boolean lattice corresponding to f is the number of "wildcards" in the element associated to f (i.e. the difference between the number of neurons of the code and the number of variables in f).

This representation of a pseudo-monomial as a hypercube is a useful way to conceptualize the relationship between pseudo-monomials, since we can represent pseudo-monomials as sub-cubes of a larger hypercube by the following lemma:

Lemma 3.2. For any pseudo-monomial $f = x_\sigma \prod_{i\in\tau}(1+x_i)$, the hypercube of f is a sub-cube of the hypercube of $g = \prod_{j\in\sigma\cup\tau}(1+x_j)$.

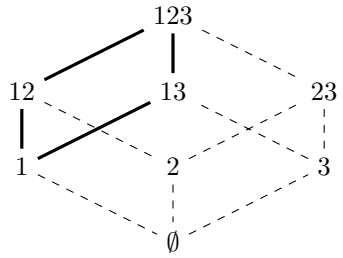
Proof. Writing g as the sum of its terms gives us

$$g = \sum_{\{\gamma\in P(\sigma\cup\tau)\}} x_\gamma.$$

Every term of f has the form $x_\sigma x_{\hat{\tau}}$ where $\hat{\tau}\subseteq\tau$. For all $\hat{\tau}$, $\sigma\cup\hat{\tau}$ clearly is a subset of $\sigma\cup\tau$, so all terms of f are terms of g. Thus the hypercube of f must be a subcube of the hypercube of g.

□

Example 3.4. Let $f = x_1(1+x_2)(1+x_3)$. Then the hypercube f form a subcube of the hypercube of $P(\{1,2,3\})$, consistent with Lemma 3.2.
Note: Since the number of variables in this example is small and all the terms are square-free, we will abuse notation and simply write the labels of the variables in each term in the hypercube



The Hasse diagram of f is the square in bold above containing vertices 1, 12, 13, and 123.

These hypercubes of pseudo-monomials portray a very nice geometric relationship between a pseudo-monomial f and the set of pseudo-monomials that divide f. To justify this, we need the following lemma:

Lemma 3.3. Let f, g be pseudo-monomials such that $g = x_\alpha \prod_{i\in\beta}(1+x_i)$ and $f = x_\sigma \prod_{j\in\tau}(1+x_j)$. Then $g|f$ if and only if $\alpha \subset \sigma$ and $\beta \subset \tau$.

Proof. Suppose $g|f$. Then $\mathrm{LT}(g)|\mathrm{LT}(f)$. Note that $\mathrm{LT}(g) = x_\alpha x_\beta$ and $\mathrm{LT}(f) = x_\sigma x_\tau$, so $\alpha \cup \beta \subset \sigma \cup \tau$.
Suppose $k \in \alpha$. Then $x_k|g$, so $x_k|f$. Since x_k is a monomial, it must divide all of the monomials of f. The monomial of f with the smallest degree is x_σ, so x_k must divide x_σ. Thus $k \in \sigma$. Since $k \in \alpha$ implies $k \in \sigma$, it follows that $\alpha \subset \sigma$.
Now suppose $k \in \beta$. Then $(1 + x_k)|g$, so $(1 + x_k)|f$ and for some quotient q, $q + qx_k = f$. It follows that $x_k|f - q$. Note that $\mathrm{LT}(q) = \frac{\mathrm{LT}(f)}{x_k}$, so $x_k \not|\mathrm{LT}(q)$. Suppose $k \in \sigma$. Then $x_k|x_\sigma$, so $x_\sigma \not|\mathrm{LT}(q)$. For every monomial x_f in f, x_σ divides x_f, so since $x_\sigma \not|\mathrm{LT}(q)$, it follows that $x_f \neq \mathrm{LT}(q)$. Therefore the coefficient of $\mathrm{LT}(q)$ in $f - q$ is non-zero. Then since $x_k|f - q$, $x_k|\mathrm{LT}(q)$. We have reached a contradiction. Thus $k \notin \sigma$, which implies $k \in \tau$. We've shown $k \in \beta$ implies $k \in \tau$, therefore $\beta \subset \tau$.
Suppose that $\alpha \subset \sigma$ and $\beta \subset \tau$. Then $x_\alpha|x_\sigma$ and $\prod_{i\in\beta}(1 + x_i)|\prod_{j\in\tau}(1 + x_j)$. Thus $g|f$.

□

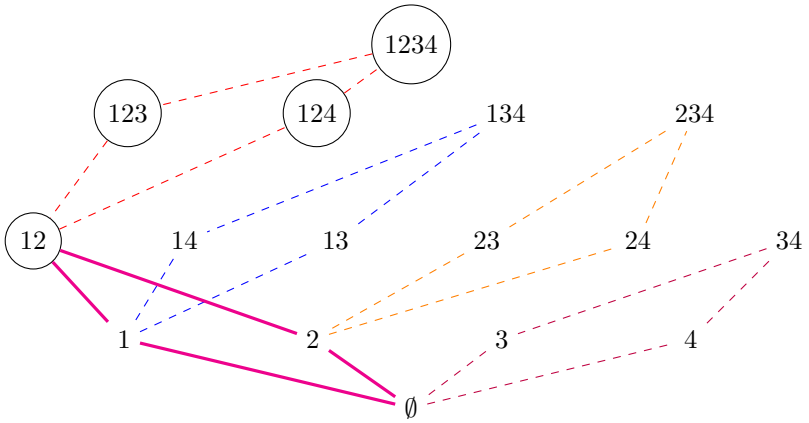
Remark: Note that a faster proof of Lemma 3.3 is that $\mathbb{F}_2[x_1, \ldots, x_n]$ is a unique factorization domain.

Lemma 3.4. Let $f = x_\sigma \prod_{i\in\tau}(1+x_i)$ and let H be the hypercube of $P(\sigma\cup\tau)$. A pseudo-monomial h divides f if and only if the hypercube of h is a sub-cube of H and the hypercube of h intersects the Hasse diagram of $P(\sigma)$ at a unique vertex.

Proof. Let $h = x_\alpha \prod_{i\in\beta}(1 + x_i)$ and suppose that $h|f$. By Lemma 3.2, h is a subcube of $P(\alpha \cup \beta)$, so since $\alpha \cup \beta \subset \sigma \cup \tau$ by Lemma 3.3, h is a subcube of H. Note that $\alpha \subset \sigma$ by Lemma 3.3, so the hypercubes of h and $P(\sigma)$ have at least one intersection. If the hypercube of h contains more than 1 vertex of $P(\sigma)$, then there exists some k such that $k \in \sigma$ and $k \in \beta$, so $h \not|f$ by Lemma 3.3. We've reached a contradiction, so the common vertex must be unique.
Conversely, suppose the hypercube of h is a subcube of H that intersects the hypercube of $P(\sigma)$ at a unique vertex. Because h is a subcube of H, it follows that $\alpha \cup \beta \subset \sigma \cup \tau$. Then since x_α divides all monomials of h, $x_\alpha|x_\sigma$ so $\alpha \subset \sigma$. Additionally, since the intersection of the hypercubes of h and $P(\sigma)$ is unique, for all $k \in \beta$, $k \notin \sigma$, so $k \in \tau$. Thus $\beta \subset \tau$, so $h|f$ by Lemma 3.3.

□

Example 3.5. Let $f = x_1x_2(1+x_3)(1+x_4)$. In the figure below, all the monomials of f are circled, and the hypercube of $P(\sigma)$ is shown in bold. A pseudo-monomial h divides f if and only if h is contained within one of the squares "parallel" to the hypercube of f and h includes a vertex from $P(\sigma)$.



Theorem 3.1. Let $f \in \mathbb{F}_2[x_1, \ldots, x_n]$ such that $f = x_\sigma \prod_{i \in \tau}(1 + x_i)$, and let $G = \{g_1, \ldots, g_s\} \subset \mathbb{F}_2[x_1, \ldots, x_n]$ be a set of pseudo-monomials. If the remainder on division of f by G is 0 for some monomial ordering, then g divides f for some $g \in G$.

Proof. Suppose that the remainder on division of f by G is 0. Let r_k denote the remainder after k steps of the division algorithm. Then there exists $i_1, \ldots, i_m \in \{1, \ldots, s\}$ such that $f = \frac{\mathrm{LT}(f)}{\mathrm{LT}(g_{i_1})} g_{i_1} + \frac{\mathrm{LT}(r_1)}{\mathrm{LT}(g_{i_2})} g_{i_2} \cdots = h_1 + \cdots + h_n$, where $h_k = \frac{\mathrm{LT}(r_k)}{\mathrm{LT}(g_{i_k})} g_{i_k}$. Since f and $h_1, \ldots, h_k$ are pseudo-monomials, we know that all terms of f divide $\mathrm{LT}(f)$, and all terms of h_k divide $LT(h_k)$ for all k. Since $r_{k+1} = r_k - h_{k+1}$, where $\mathrm{LT}(h_{k+1}) | \mathrm{LT}(r_k)$, all terms in r_{k+1} divide $\mathrm{LT}(f)$ by induction. Thus for all h_k, the leading term of h_k divides the leading term of f.

At each step of the division algorithm, the remainder is $r_{k+1} = r_k - h_{k+1}$, where $r_0 = f$. From above, we know that $\mathrm{LT}(h_k) | \mathrm{LT}(f)$, so the hypercube of h_k must be a sub-cube of the hypercube of $P(\sigma \bigcup \tau)$. The intersection of two hypercubes is always a hypercube, which contains 2^q vertices for some $n \in \mathbb{N}$, so the intersection of the hypercubes of h_k and $P(\sigma)$ must have 2^q vertices in common.

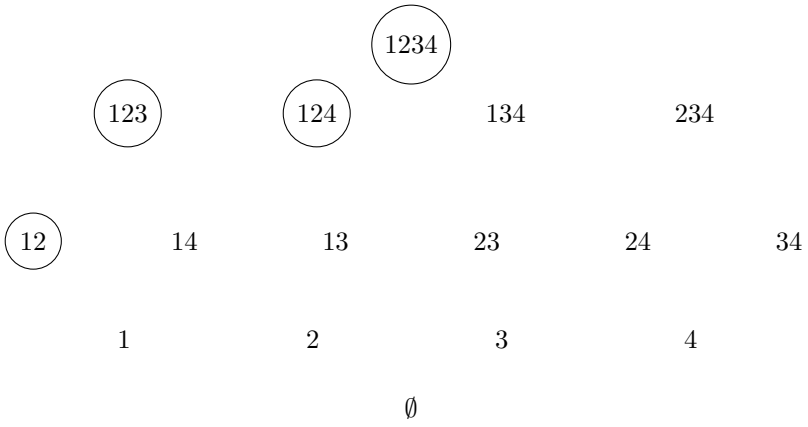
Recall that our polynomial addition is over the field $\mathbb{F}_2$. Thus since we know that $f - h_1 - \cdots - h_n = 0$, it follows that $f, h_1, \ldots, h_n$ together must contain an even number of each term. Consider the set of vertices in the hypercube of $P(\sigma)$. At the initial state of the algorithm, we start with the pseudo-monomial f. Thus in the Hasse diagram of r_0, the only vertex from $P(\sigma)$ is x_σ, the intersection of the Hasse diagram of r_0 and the hypercube of $P(\sigma)$ contains an odd number of vertices. Since $f - \sum h_i = 0$, the hypercubes of $h_1, \ldots, h_n$ together must contain an odd number of vertices from the hypercube of $P(\sigma)$. It follows that for some h_k, the hypercube of h_k and the hypercube of $P(\sigma)$ have an odd number of vertices in common. Since the intersection of two hypercubes is always a hypercube, the hypercubes of h_k and $P(\sigma)$ must have 2^q vertices in common for some q, so the only odd number of vertices that they can have in common is 1.

Since $\mathrm{LT}(h_k) | \mathrm{LT}(f)$, by Lemma 3.2 the hypercube ofh_k is a subcube of $P(\sigma \cup \tau)$. We've shown that the hypercube of h_k and the hypercube of of $P(\sigma)$ intersect at a unique vertex, so by Lemma 3.4, $h_k | f$. Recall that h_k is the multiple of some $g_i \in G$, i.e., $g_i | h_k$, and thus there exists $g_i \in G$ such that $g_i | f$.

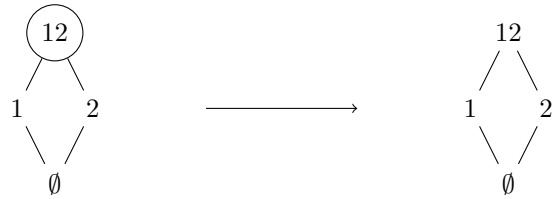
□

The following example illustrates the ideas behind the proof.

Example 3.6. We can think of each term in r_k as having a binary state space of 1 or 0, which we can represent graphically on the Hasse diagram of $P(\sigma \cup \tau)$ by circling a vertex of the corresponding term that has a non-zero coefficient in r_k. At the first step of the division algorithm, all the vertices corresponding to terms of f are circled. Let $f = x_1x_2(1 + x_3)(1 + x_4)$. At the initial state of the division algorithm, $r_0 = f$, so only the terms of f have non-zero coefficients.



Our focus is on the terms in the hypercube of $P(\sigma)$. At the beginning of the division algorithm, the vertex corresponding to x_1x_2 is the only one circled. We know that at the end of the division algorithm, none of the vertices should be circled. Whenever we go from r_k to r_{k+1}, where $r_{k+1} = r_k + h_k$, we can think of h_k as "flipping" the state of the set of monomials in h_k. In other words, f a monomial in h_k is circled at r_k, then it becomes uncircled at r_{k+1} and vice versa. The Hasse diagram of $\prod_{i\in\sigma}(1 + x_i)$ goes from having one vertex circled at the beginning of the algorithm to having no vertices circled at the end of the algorithm.



Initial State End State

"Flipping" a vertex an even number of times has no effect on its state, so we know that at some point some h_k must flip an odd number of vertices (by the previous proof). Then the intersection of the Hasse diagram of h_k with the Hasse diagram of $\prod_{i\in\sigma}(1 + x_i)$ is a single vertex, so h_k divides f by Lemma 3.4.

3.2 The Universal Gröbner Basis

By Lemma 2.3, we know that if the canonical form is a Gröbner basis, then it is universal. Therefore if the canonical form is a *reduced* Gröbner basis for some monomial ordering, then it is the reduced Gröbner basis for *all* monomial orderings, making it equal to the universal Gröbner basis.

By definition, all elements of the canonical form are minimal pseudo-monomials, so the canonical form is in some way a 'reduced' generating set of the neural ideal. Note that the canonical form is not necessarily a minimal generating set of the neural ideal, i.e. in some cases a proper subset of the canonical form can generate the neural ideal. However, we find that the canonical form is the minimal generating set of pseudo-monomials that can form a Gröbner basis of the neural ideal.

Theorem 3.2. If the canonical form, CF, of a neural ideal J_C is a Gröbner basis of J_C, then the canonical form is a reduced Gröbner basis of J_C.

Proof. Suppose that CF is a Gröbner basis, but not a reduced Gröbner basis. Then for some pseudo-monomial f in CF, a monomial of f lies in $\langle \text{LT}(CF\backslash\{f\})\rangle$. Since every monomial of f divides $\text{LT}(f)$, it follows that $\text{LT}(f) \in \text{LT}(CF\backslash\{f\})\rangle$. Thus for some $g \in CF$, with $g \neq f$, $\text{LT}(g)|\text{LT}(f)$. Let $r = f - \frac{\text{LT}(f)}{\text{LT}(g)}g$. Since the canonical form is a Gröbner basis, we know that the remainder of division on r by CF is 0. In this division of r by CF, the pseudo-monomial f is never used, since $\deg(f)>\deg(r)$. It follows that the remainder of division of f by $CF\backslash\{f\}$ is 0, so by Theorem 3.1, there exists some pseudo-monomial $g_i \in CF\backslash\{f\}$ such that $g_i|f$. Thus f is not minimal, but we assumed that f was in the canonical form, so we have reached a contradiction. Thus if the canonical form is a Gröbner basis, the canonical form must be a reduced Gröbner basis. □

Corollary 3.1. If the canonical form of a neural ideal J_C is a Gröbner basis, then it is the universal Gröbner basis of J_C.

Proof. Follows from Lemma 2.3 and Theorem 3.2.

□

Theorem 3.3. Let J_C be a neural ideal and let G be the universal Gröbner basis of J_C. For every $g \in G$, if g is a pseudo-monomial, then g is in the canonical form of J_C.

Proof. By definition, G is the union of the reduced Gröbner fan of J_C. Thus if $g \in G$, g is in the reduced Gröbner basis of some monomial ordering. Let R be a reduced Gröbner basis of J_C such that $g \in R$. Suppose that g is not minimal in J_C. Then for some pseudo-monomial $h \in J_C$ such that $\deg(h)<\deg(g)$, $h|g$. Then for some $r \in R$, $\text{LT}(r)|\text{LT}(h)$. Note that $\deg(r)<\deg(g)$, so $r \neq g$. But $\text{LT}(r)|\text{LT}(g)$, because $\text{LT}(h)|\text{LT}(g)$, so g and r cannot both be in a reduced Gröbner basis. We have reached a contradiction. Therefore the only pseudo-monomials in the universal Gröbner basis are minimal pseudo-monomials. □

Corollary 3.2. Let J_C be a neural ideal, and let G be the universal Gröbner basis of J_C. If for all $g \in G$, g is a pseudo-monomial, then the canonical form of J_C is equal to G.

Proof. If all elements of G are pseudo-monomials, then all elements of G are in the canonical form by Theorem 3.3, so G is a subset of the canonical form.
Note that since g only contains pseudo-monomials, g is the reduced Gröbner basis for any monomial ordering by Lemma 2.3. Let f be a pseudo-monomial in the canonical form. We know that f reduces to 0 when divided by G, so for some $g \in G$, it must be the case that g divides f. Since f is in the canonical form, f is minimal, and the only pseudo-monomial that divides f is f itself. Thus $g = f$, so f is in G. Therefore the canonical form is a subset of G and thus is equal to G.

□

The implication of this result is an improved average runtime for calculating the canonical form when the canonical form is a Gröbner basis. The amount of time it takes to compute the Gröbner basis is trivial compared to the runtime of the iterative algorithm, so when the Gröbner basisThen for all reduced Gröbner bases R of J_C, $R \subset J_C$. If for some R, R is not equal to the canonical form, the time wasted computing the Gröbner basis is not significant. On the other hand, if it does turn out that all elements of a reduced Gröbner basis of J_C are pseudo-monomials, then the runtime is improved by multiple orders of magnitude.

3.3 Complement-Complete Codes

Definition 3.2. Let $c \in \{0,1\}^n$ be a code. The **complement** of c is the code $c' \in \{0,1\}^n$ such that $c'_i = 1$ if and only if $c_i = 0$.
Let f be a pseudo-monomial such that $f = x_\sigma \prod_{i\in\tau}(1 + x_i)$. The **complement** of f, denoted f', is $f' = x_\tau \prod_{j\in\sigma}(1 + x_j)$.
A code $C \subset \{0,1\}^n$ is called **complement-complete** if for all $c \in C$, $c' \in C$ as well.

Lemma 3.5. Let f, g be pseudo-monomials such that $f = x_\sigma \prod_{i\in\tau}(1+x_i)$ and $g = x_\alpha \prod_{j\in\beta}(1+x_j)$ If g divides f, then g' divides f'.

Proof. If $g|f$, then $\alpha \subset \sigma$ and $\beta \subset \tau$. From Lemma 3.3 it follows that $g'|f'$. □

Proposition 3.2. Let I be an ideal. Then for all pseudo-monomials $f \in I$, there exists a minimal pseudo-monomial $g \in I$ such that $g|f$.

Proof. Only finitely many pseudo-monomials can divide f, so for some $g \in J_C$ such that $g|f$, g must be minimal. □

Theorem 3.4. Let C be a code on n neurons such that $C \subsetneq \{0,1\}^n$. If C is complement-complete, then the canonical form is not a Gröbner basis.

Proof. Note that since $C \neq \{0,1\}^n$, J_C is not the trivial ideal. Let f be a pseudo-monomial in J_C, and let S be the set of all pseudo-monomials of degree n that are multiples of f. The only pseudo-monomials in J_C of degree n are the characteristic functions ρ_v, where $v \notin C$. If v is not in C, then its complement v' is also not in C. Then $\rho_{v'} = (\rho_v)'$ is also one of the characteristic functions that generate J_C. Note that for all pseudo-monomials $s \in S$, $s = fq$ for some pseudo-monomial q. Then fq' is also in S. Since the gcd of q and q' is 1, it follows that f is the gcd of s and fq', so f

is the gcd of S. Then f' is the gcd of S' (the set of all complements of elements of S), so we can express f' as a finite combination of elements in S'.

It follows that for all pseudo-monomials $f \in J_C$, if there exists a pseudo-monomial $d \in J_C$ such that $d|J_C$, then $d' \in J_C$ and $d'|f'$. Thus a pseudo-monomial f in J_C is minimal if and only if f' is minimal, so if f is in the canonical form, then f' is in the canonical form. However, $\mathrm{LT}(f) = \mathrm{LT}(f')$, so they cannot both be in a reduced Gröbner basis. Then by Theorem 3.2, the canonical form is not a Gröbner basis.

□

4 Future work

Using the previous results, we can drastically improve the speed of computing the canonical form in cases where the canonical form is a reduced Gröbner basis. The hope is that in cases where the canonical form is *not* a reduced Gröbner basis, we can still use the polynomials present in the Gröbner basis to infer what the minimal pseudo-monomials of the neural ideal are.

Example 4.1. Let C be a code on 3 neurons such that the canonical form of J_C is

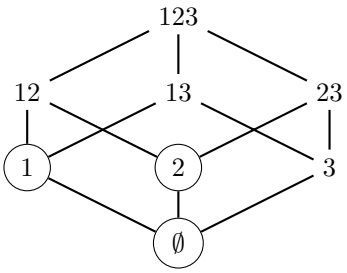
$$J_C = \langle x_1x_2, x_1x_3, (1+x_1)(1+x_2), x_2(1+x_3), x_3(1+x_2), (1+x_2)(1+x_3)\rangle.$$

Note that for all pseudo-monomials f in the canonical form, f' is in the canonical form, telling us that C is complement-complete.

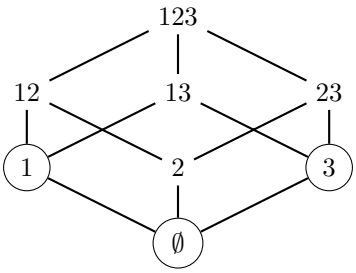
Using SageMath, we find the universal Gröbner basis of J_C is

$$\langle x_1 + x_3 + 1, x_1 + x_2 + 1, x_2 + x_3\rangle.$$

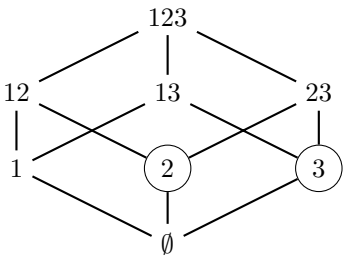
We represent each element of the Gröbner basis by circling the vertices corresponding to their terms on the hypercube of $P(\{1,2,3\})$.



$x_1 + x_2 + 1$



$x_1 + x_3 + 1$



$x_2 + x_3$

Note that every element in the universal Gröbner basis is a combination of pseudo-monomials that share a leading term. Looking at the picture above, it is easy to see what pseudo-monomials combine to produce the elements of the Gröbner basis:

$$x_1 + x_2 + 1 = ((1+x_1)(1+x_2)) + (x_1x_2),$$

$$x_1 + x_3 + 1 = ((1 + x_1)(1 + x_3)) + (x_1x_3),$$
$$x_2 + x_3 = (x_2(1 + x_3)) + (x_3(1 + x_2)).$$

In this example, the smallest pseudo-monomials that sum to elements of the universal Gröbner basis are exactly the elements of the canonical form. Additionally, notice that the two pseudo-monomials that combine to give each element of the Gröbner basis are complements. In general, polynomials of the form $x_a + x_b$ and $x_a + x_b + 1$ can be expressed as the sum of complementary pseudo-monomials, where

$$x_a + x_b = x_a(1 + x_b) + x_b(1 + x_a)$$
$$x_a + x_b + 1 = x_ax_b + (1 + x_a))1 + x_b.$$

In this example, the universal Gröbner basis is not a reduced Gröbner basis in any monomial ordering. Rather, it is the union of the Gröbner bases

$$\langle x_1 + x_3 + 1, x_2 + x_3\rangle, \qquad \langle x_1 + x_2 + 1, x_2 + x_3\rangle$$

.

In the above example, the canonical form is not a minimal generating set of the neural ideal, since the pseudo-monomials corresponding to one of the reduced Gröbner bases can be generated by the other reduced Gröbner basis. This observation gives rise to the following conjecture:

Conjecture 4.1. Let J_C be a neural ideal. If the universal Gröbner basis of J_C is not a reduced Gröbner basis, then the canonical form is not a minimal generating set of the neural ideal.

Example 4.2. Now let us consider a code on 4 neurons such that the universal Gröbner basis of the neural ideal is

$$\langle x_1 + x_4, x_2 + 1, x_1 + x_3 + 1, x_3 + x_4 + 1\rangle$$

By Theorem 3.3, we know that the pseudo-monomial $x_2 + 1$ must be in the canonical form. We deduce from the previous example that the other elemnts in the Gröbner basis correspond to the following complementary pseudo-monomials

$$x_1 + x_3 + 1 \implies x_1x_3, (1 + x_1)(1 + x_3)$$
$$x_1 + x_4 \implies x_1(1 + x_4), x_4(1 + x_1)$$
$$x_3 + x_4 + 1 \implies x_3x_4, (1 + x_3)(1 + x_4)$$

SageMath confirms that the complementary pseudo-monomials above along with x_2+1 are precisely the pseudo-monomials in the canonical form.

5 Appendix

5.1 Code

The following code builds upon the canonical form package written by Peterson et al. in SageMath. If there is a polynomial in the Gröbner basis that is not a pseudo-monomial, then the method returns the canonical form using the default canonical form algorithm. If all polynomials in the Gröbner basis are pseudo-monomials, the method returns the Gröbner basis, which in that case is equal to the canonical form.

```
def improved_canonical(self):
        gb = self.get_groebner_basis()
        product = 1
        for i in range(self.d):
            product = product * self.x[i]*(1+self.x[i])

        for g in gb.gens():
            if g != g.gcd(product):
                return self.get_canonical()
        return gb
```

6 Acknowledgements

I would like to thank my mentor, Anne Shiu, for her help and guidance. I would also like to thank Kaitlyn Phillipson and Ola Sobieska, and Luis Garcia Puente for helpful discussions.

This project was conducted as part of the NSF-funded REU in Mathematics at Texas A&M University (DMS-1460766), Summer 2016.

References

[1] C. Curto and N. Youngs. Neural ring homomorphisms and maps between neural codes. *ArXiv e-prints*, November 2015.

[2] Carina Curto, Vladimir Itskov, Alan Veliz-Cuba, and Nora Youngs. The neural ring: an algebraic tool for analyzing the intrinsic structure of neural codes. *Bull. Math. Biol.*, 75(9):1571–1611, 2013.

[3] Ethan Peterson, Dane Miyata, Ryan Kruse, and Ihmar Aldana. Neural mapping using gröbner bases. 2016.

[4] Ethan Peterson, Nora Youngs, Ryan Kruse, Dane Miyata, Rebecca Garcia, and Luis D. Garcia Puente. Neural codes in sage. 2016.