Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 32(5): 3162–3170.
DOI: 10.3934/era.2024145
Received: 20 December 2023
Revised: 28 March 2024
Accepted: 28 April 2024
Published: 06 May 2024

Theory article

The algorithm for canonical forms of neural ideals

Licui Zheng*, **Yiyao Zhang and Jinwang Liu**

School of Mathematics and Computational Science, Hunan University of Science and Technology, Xiangtan 411201, China

* **Correspondence:** Email: zhenglicui@126.com.

Abstract: To elucidate the combinatorial architecture of neural codes, the neural ideal J_C, an algebraic object, was introduced. Represented in its canonical form, J_C provides a succinct characterization of the inherent receptive field architecture within the code. The polynomials in J_C are also instrumental in determining the relationships among the neurons' receptive fields. Consequently, the computation of the collection of canonical forms is pivotal. In this paper, based on the study of relations between pseudo-monomials, the authors present a computationally efficient iterative algorithm for the canonical forms of the neural ideal. Additionally, we introduce a new relationship among the neurons' receptive fields, which can be characterized by if-and-only-if statements, relating both to J_C and to a larger ideal of a code $I(C)$.

Keywords: neural code; neural ideals; canonical forms

1. Introduction

One of the pivotal realms of neuroscience research revolves around unraveling the intricate mechanisms by which the brain perceives and interprets spatial information. Within this framework, neurons in the brain employ sophisticated neural codes to encapsulate and manifest external stimuli. With the continuous advancement of neuroscience technology, researchers' ability to collect neural data is also constantly increasing, hence the requirement for alternative methodologies to analyze and process these data. A significant challenge in current research is understanding how brain processes encode environmental spatial characteristics through neural activity patterns. In 2013, C. Curto and others studied the encoding of the stimulus space region corresponding to each neuron [1, 2], introducing algebraic objects to present neural activity data in the form of neural codes, and mapping the corresponding codes to pseudo-monomials in $\mathbb{F}_2$, thereby defining the related neural ring and neural ideal. The neural ring is a quotient ring in $\mathbb{F}_2$, and the neural ideal encompasses the entire combinatorial dataset of neural coding. This concept is represented by a unique set of minimal pseudo-monomials within the neural

ideal, known as the canonical form, which is the minimal representation of the receptive field structure in the stimulus space. Therefore, the study of stimulus space features can be transformed into the study of the neural ideal or its canonical form. Since then, this field has been very active, especially in recent years when some relatively important works have appeared. For example, the works [3–9] have done a good job of processing the receptive field structure formed in the brain. The study of the specific algorithm for the canonical form of neural ideals is a question of concern to many scholars at present. In 2013, C. Curto and others gave the original method for computing the canonical form [1]. This specific algorithm mainly uses ideas similar to monomial ideal theory and is implemented by computing the primary decomposition of the pseudo-monomial ideal. E. Petersen and others gave an iterative method [10]. In the same year, A. R. Perez and others also involved the calculation of the canonical form when studying the homomorphism that preserves the neural ideal on the neural ring [11]. We have more interest in [10]. In [10], the authors present an alternative specific algorithm that diverges from utilizing primary decomposition. Instead, it commences with the canonical form of a code made up of a solitary codeword, proceeding to incorporate the residual codewords singly while readjusting the canonical form correspondingly through iteration. But the specific algorithm should examine the multiplication of f with all possible linear terms $(x_i - c_i)$ and it involves individually verifying whether f is indivisible by each $x_i - c_i - 1$. We offer some criteria that, by employing these principles, allow for the efficient identification of those pseudo-monomials whose product with $(x_i - c_i)$ need not be computed.

Besides, Curto et al. showed that the presence of certain types of polynomials in the neural ideal gives information about the relationships among receptive fields. They found three receptive field relationships, known as the Type 1–3 relationships, that can be read off from the neural ideal. Later, Garcia et al. discovered three more such relations, known as the Type 4–6 relations [2]. A. Morvant modified statements of Type 4–6 as a if-and-only-if statements to both J_C and $I(C)$ [9]. Within this study, we identify additional types of receptive field relationships that stem from non-pseudo-monomials.

The paper is organized as follows. Some basic concepts for neural ideals are introduced in Section 2. In Section 3, the main results are presented. And in Section 4, we have presented an improved, specific algorithm and clearly illustrated its modifications through a concrete example.

2. Preliminaries

Moving on to the next section, we give a short overview of the neural ideal of neural code. For a more detailed background, including relevant theorems and proofs, please refer to [2].

Definition 1. A neural code on n neurons is a set of binary firing patterns $C \subseteq \{0, 1\}^n$, and any vector c within this set is referred to as a codeword.

Each individual codeword c within the context of a neural code represents a firing pattern of n neurons: 1 signifies an active neuron, whereas 0 denotes an inactive or quiescent neuron. For example, the presence of the word 0101 in a 4-neuron code would indicate an instance when neurons 2 and 4 were firing but neurons 1 and 3 were not.

Alternatively, a codeword is characterized by the collection of neurons that exhibit activity:

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

Consequently, the entire neural code is associated with a set of subsets comprising neurons that fire simultaneously: $supp(C) = \{supp(c)|c \in C\} \subseteq 2^{[n]}$.

Neurons like place cells, are activated in specific areas of a stimulus space, known as receptive fields.

Definition 2. Assume X be a stimulus space (such as $X \subseteq \mathbb{R}^d$) and $\mathcal{U} = \{U_1, \ldots, U_n\}$ as a set of open sets, where each $U_i \subseteq X$ corresponds to the receptive field of the i-th neuron in a group of n neurons. The receptive field code (RF code) $C(\mathcal{U}) \subseteq \{0,1\}^n$ is the set of all binary codewords corresponding to stimuli in X:

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

In the previous discussion, we frequently associated this code with the set of corresponding subsets within $[n]$. Moreover, when it comes to an empty intersection, we adhere to the following convention: $\bigcap_{i \in \emptyset} U_i = X$.

Definition 3. A pseudo-monomial in $F_2[x_1, ..., x_n]$ is a polynomial exemplified by the following structural expression:

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

where $\sigma, \tau \in [n]$ with $\sigma \cap \tau = \emptyset$.

For any $v \in \{0,1\}^n$, consider the polynomial:

$$\rho_v = \prod_{i=1}^{n} (1 - v_i - x_i) = \prod_{\{i|v_i=1\}} x_i \prod_{\{j|v_j=0\}} (1 - x_j).$$

Notice that $\rho_v(x)$ acts as a characteristic function for v, since it satisfies $\rho_v(v) = 1$ and $\rho_v(x) = 0$ for any $x \neq v$.

Definition 4. Let $C \subseteq \{0,1\}^n$ be a neural code. The neural ideal $J_C \subset F_2[x_1, ..., x_n]$ associated to the neural code C is the ideal generated by the polynomials ρ_v with $v \notin C$, that is:

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

Notice that this implies that if $f \in J_C$, then $f(c) = 0$ for any $c \in C$.

Definition 5. Let $C \subseteq \{0,1\}^n$ be a neural code, then the ideal of C is:

$$I(C) := \{f \in F_2[x_1, \ldots, x_n] | \forall c \in C : f(c) = 0\}.$$

By this definition, $J_C \subseteq I(C)$. A lemma is provided below, which gives a version in terms of algebra for the previous assertion.

Lemma 6. *([1]) Let $C \subseteq \{0,1\}^n$ be a neural code. Then:*

$$I(C) = J_C + \langle x_i(1 + x_i) | i \in [n] \rangle.$$

The collection of polynomials that make up a neural ideal offers a way to understand the connections between the receptive fields within a stimulus space. These interconnections were initially explored by Curto, Itskov, Veliz-Cuba, and Youngs, who identified the first three types of relationships [1]. To abbreviate the notation, we introduce:

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

Note that if $\sigma = \emptyset$, then $x_\sigma = \prod_{i\in\sigma} x_i = 1$ and $U_\sigma = X$.

Lemma 7. *([1]) Given a stimulus space X and a collection of sets $U = \{U_i\}_{i=1}^n$ in X, we have the receptive field code $C = C(U)$. For any two subsets $\sigma, \tau \subseteq [n]$,*

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

In essence, three distinct types of receptive-field relationships (RF relationships) can be extracted from the pseudo-monomials within a neural ideal. For further details, please refer to reference [5].

Type 1: $x_\sigma \in J_C \iff U_\sigma = \emptyset$ *(where* $\sigma \neq \emptyset$*).*

Type 2: $x_\sigma \prod_{i\in\tau}(1 + x_i) \in J_C \iff U_\sigma \subseteq \bigcup_{i\in\tau} U_i$ *(where* $\sigma, \tau \neq \emptyset$*).*

Type 3: $\prod_{i\in\tau}(1 + x_i) \in J_C \iff X \subseteq \bigcup_{i\in\tau} U_i$ *(where* $\tau \neq \emptyset$*), so* $X = \bigcup_{i\in\tau} U_i$*.*

In the present work, we identify additional types of receptive field relationships that stem from non-pseudo-monomials.

Definition 8. Let J_C be a neural ideal. Then its canonical form, indicated as $CF(J_C)$, is the set of all minimal pseudo-monomials of J_C.

Next, we will give the algorithm of the $CF(J_C)$.

3. Main results

We will present the main results of our paper in this section.

Theorem 9. *If* $x_{i_1}x_{i_2}\ldots x_{i_n} + x_{i_{n+1}} + \cdots + x_{i_{n+m}} + 1 \in J_C \iff (\bigcap_{k=1}^{n} U_{i_k}) \cup \left(\bigcup_{l=1}^{m} U_{i_l}\right) = X$ *and* $U_{i_1}, \ldots, U_{i_{n+m}}$ *are pairwise disjoint.*

Proof. " $\Leftarrow$ " Suppose that $(\bigcap_{k=1}^{n} U_{i_k}) \bigcup \left(\bigcup_{l=1}^{m} U_{i_l}\right) = X$ and $M = \{i_{n+1}, ..., i_{m+n}\}, N = \{i_1, ..., i_n\}$. Then $\bigcap_{k=1}^{n} U_{i_k} = \emptyset$ as $U_{i_1}, \ldots, U_{i_n}$ are pairwise disjoint, and it follows that $X = \bigcup_{l=1}^{m} U_{i_l}$. By the Type 3 relation, we know that $\prod_{i\in M}(1+x_i) \in J_C$, whereas $\prod_{i\in M}(1+x_i) = \sum_{\tau\subseteq M} x_\tau$. Furthermore, considering that $U_{i_{n+1}}, \ldots, U_{i_{n+m}}$ are pairwise disjoint, by the Type 1 relation, $x_\tau \subseteq J_C$ for all $\tau \subseteq M$ where$|\tau| \geq 2$. Therefore,

$$\prod_{\sigma\subseteq N} x_\sigma \sum_{\tau\subseteq M} x_\tau + \sum_{\tau\subseteq M\ |\tau|\geq 2} x_\tau = \prod_{\sigma\subseteq N} x_\sigma \sum_{\tau\subseteq M\ |\tau|<2} x_\tau = 1 + x_{i_{n+1}} + \cdots + x_{i_{n+m}} + x_{i_1}x_{i_2}\ldots x_{i_n} \in J_C.$$

" $\Rightarrow$ " Let $h = x_{i_1}x_{i_2}...x_{i_n} + x_{i_{n+1}} + x_{i_{n+2}} + ... + x_{i_{m+n}} + 1$, $r \in X$, then $c(r) \in C$. Next, we just need to prove $r \in \left(\bigcap_{k=1}^{n} U_{i_k}\right) \bigcup \left(\bigcup_{l=1}^{m} U_{i_l}\right)$.

Then $h(c(r)) = 0$ as $h \in J_C$, it follows:

$$c(r)_{i_1}c(r)_{i_2}...c(r)_{i_n} + c(r)_{i_{n+1}} + c(r)_{i_{n+2}} + ... + c(r)_{i_{m+n}} + 1 = 0,$$

so:

(1) There exist $k \in [n]$, $p \in [m]$, such that $c(r)_{i_p} = 0$ and $c(r)_{i_k} = 1$, where k is odd, or
(2) $c(r)_{i_s} = 1$ for all $s \in \{1, ..., n\}$ and $c(r)_{i_{n+1}} + c(r)_{i_{n+2}} + ... + c(r)_{i_{m+n}} = 0$.

It follows that $r \in U_{i_l}$ when (1) is correct, and $r \in (\bigcap_{l=1}^{n} U_{i_l}) \bigcup (\bigcup_{k=1}^{m} U_{i_k})$.

If (2) is true, then $r \in \cap_{k=1}^{m} U_{i_k}$, similarly obtained $r \in (\bigcap_{l=1}^{n} U_{i_l}) \bigcup (\bigcup_{k=1}^{m} U_{i_k})$.

We can extend the above theorem from ideal J_C to a larger ideal $I(C)$.

Theorem 10. *If* $x_{i_1} x_{i_2} \ldots x_{i_n} + x_{i_{n+1}} + \cdots + x_{i_{n+m}} + 1 \in I(C) \Longleftrightarrow (\bigcap_{k=1}^{n} U_{i_k}) \cup \left(\bigcup_{l=1}^{m} U_{i_l} \right) = X$ *and* $U_{i_1}, \ldots, U_{i_{n+m}}$ *are pairwise disjoint.*

Proof. Having established the validity of the reverse directions via the theorem above and the inclusion relation $J_C \subseteq I(C)$, our remaining task is to establish the forward implications.

However, the demonstration procedure for the forward implications parallels that of the above theorem, with the sole modification being the substitution of "$h \in J_C$" for "$h \in I(C)$".

It can be gleaned from the above- mentioned theorem that J_C and $I(C)$ yield equivalent information with regards to the stimulus space.

Definition 11. Let $m_1, m_2 \in F_2[x_1, \ldots, x_n]$ be two pesudo-monomials. We say m_1 and m_2 share an index i if x_i divides one of them and $1 + x_i$ divides the other. If a single pesudo-monomial is divisible by $x_i(1 + x_i)$, we do not count this as sharing an index with itself.

For example, $x_5(1 + x_6)$ and $x_7(1 + x_5)$ share the index 5. However, $x_5(1 + x_6)$ and $x_5(1 + x_7)$ share no index, even though x_5 divides both.

Before introducing the following theorem, let us first briefly review the algorithm mentioned in [10]. In [10], they must consider the product of f (which is a pseudo-monomial) with every possible linear term $(x_j - c_j)$ as a potential candidate for $CF(J_C)$; however, they also have to eliminate any such products deemed redundant. The process of scrutinizing each product individually results in significant redundancy, which not only demands substantial memory usage but also reduces computational efficiency. In the following, we introduce some criteria that can be used not only to swiftly identify which pseudo-monomial products with $(x_j - c_j)$ need not be computed but also to directly discern which of these products are superfluous.

Theorem 12. *Let* $f \in F_2[x_1, \ldots, x_n]$. *If* f *and* $(x_i - c_i)$ *share indices* i, *then there is no need to calculate their product.*

Proof. Set $f = \prod_{i \in \sigma} x_i \prod_{l \in \tau} (1 + x_l)$, then there exists a specific index $i \in \tau$ or $i \in \sigma$ such that under one condition where $(1 + x_i) | f$, x_i also divides $(x_i - c_i)$; Conversely, when f can be divided by x_i, in this case $(1 + x_i)$ also divides $(x_i - c_i)$. Regardless of the given conditions, $f \times (x_i - c_i)$ will consistently contain both x_i and $(1 + x_i)$. As such, it can safely be removed.

According to the aforementioned theorem, there is no need to individually compute the product of f and $(x_i - c_i)$ like in the original specific algorithm; instead, we only need to compute the products of f and $(x_i - c_i)$ that do not share any indices. Furthermore, in the original specific algorithm after calculating the product of f and $(x_i - c_i)$, we still need to verify whether $(x_i - c_i - 1)$ divides f or not.

Next, we present another determining condition under which verification of whether $(x_i - c_i - 1)$ divides f becomes unnecessary.

Theorem 13. *Let $f \in F_2[x_1, \ldots, x_n]$, $lt(f), (x_i - c_i)$ share no indices and $gcd(lt(f), x_i - c_i) = 1$, then $(x_i - c_i - 1) \nmid f$, where $lt(f)$ refers to the leading term of the polynomial f under a given monomial order.*

Proof. Using proof by contradiction, suppose $(x_i - c_i - 1)|f$, in that case, $(x_i - c_i - 1)|\mathrm{lt}(f)$; hence, $(x_i - c_i)$ shares indices with $\mathrm{lt}(f)$, which contradicts the premise.

4. Algorithm and example

Based on the above theorem, in this section we have presented an improved specific algorithm and clearly illustrated its modifications using a concrete example.

To further improve the efficiency of the specific algorithm given in [10], we propose some criteria to detect the useless production of f and $(x_i - c_i)$. First, we propose the conception of sharing an index and prove that the production is useless if $\mathrm{lt}(f)$ and $(x_i - c_i)$ share an index. Second, we propose the leading term coprime to detect not only useless production but also eliminate the process of examining whether f is divisible by $(x_i - c_i - 1)$. Hence, the proposed algorithm has the potential to significantly enhance the computational efficiency of the initial, specific algorithm. We refer to the improved, specific algorithm in Figure 1.

	An improvement specific algorithm for canonical form
Input:	$CF(J_C) = \{f_1, ..., f_k\}$, a codeword $c \in \{0, 1\}^n$.
Output:	$CF(J_{C\cup\{c\}})$.
Variables:	$L \leftarrow \{\quad\}, M \leftarrow \{\quad\}, N \leftarrow \{\quad\}$
step 1:	Compute $f_j(c)$ where j from 1 to k step 1a:if $f_j(c) = 0$ then $L \leftarrow L \cup \{f_j\}$; step 1b: if $f_j(c) \neq 0$ then $M \leftarrow M \cup \{f_j\}$;
step 2:	Take any $f_j \in M$ where j ranges from 1 to k. step 2a: if $\mathrm{lt}(f_j)$ and $(x_i - c_i)$ share indices i then go to step 2, where i from 1 to n. (The same applies for i below.) step 2b: if $\mathrm{lt}(f_j)$ and $(x_i - c_i)$ share no indices and $\gcd(\mathrm{lt}(f_j), (x_i - c_i)) = 1$ then go to step 2; step 2c: if $\mathrm{lt}(f_j)$ and $(x_i - c_i)$ share no indices and $\gcd(\mathrm{lt}(f_j), (x_i - c_i)) \neq 1$ then compute $f_j(x_i - c_i)$. step 2c1: if $f_j(x_i - c_i)$ is not divisible by any element of L then $N \leftarrow N \cup \{f_j(x_i - c_i)\}$, go to step 2; step 2c2: if $f_j(x_i - c_i)$ can be divisible by some element of L then delete it, go to step 2.
Until:	$M = \emptyset$.
return:	$L \cup N = CF(J_{C\cup\{c\}})$

Figure 1. An improvement- specific algorithm for canonical forms.

Subsequently, we shall furnish a demonstration of the specific algorithm's validity.

Lemma 14. *The set $L \cup N$ in the specific algorithm does not contain elements that are multiples of each other.*

Proof. We will complete the proof from four aspects.

(1) The set L does not contain elements that are multiples of each other. According to the specific algorithm, set L belongs to $CF(J_C)$, so it is evident that there are no elements in L that are multiples of each other.

(2) According to the specific algorithm, we discard them if they are a multiples of anything in L, so the elements in N cannot be multiples of the elements in L.

(3) There are no elements in N that are multiples of any other elements in N.

We shall employ proof by contradiction to demonstrate that this scenario is untenable.

Suppose $f(x_i-c_i), h(x_k-c_k) \in N$, and there is a pseudo-monomial m, such that $f(x_i-c_i) = mh(x_k-c_k)$, then it follows that $i \neq k$ as $f \nmid h$ and $h \nmid f$, so $(x_k - c_k) \mid f$, and furthermore $f(x_k - c_k) = f$. Similarly, if $h(x_i - c_i) = h$, then:

$$ff(x_i - c_i) = mh(x_k - c_k)f \Rightarrow f(x_i - c_i) = mh$$

and

$$f(x_i - c_i)h = mhh \Rightarrow f = mh,$$

which is a contradiction. So (3) is correct.

(4) There are no elements in L that are multiples of any other elements in N.

Assuming that $f(x_i - c_i)m = h$, where $f(x_i - c_i) \in N, h \in L$, and m is a pseudo-monomial, then $f \mid h$; however, this is impracticable due to the fact that $f, h \in CF(J_C)$.

In summary, the lemma holds.

Theorem 15. *If $C, c \in \{0,1\}^n$, and L, N are the sets mentioned pertain to those described in the specific algorithm, then $L \cup N = CF(J_{C\cup\{c\}})$.*

Proof. First, we show $L \cup N \subset CF(J_{C\cup\{c\}})$. For any $h \in L \cup N$, there exist $f_j \in CF(J_C)$ and $(x_i - c_i) \in CF(J_{\{c\}})$ such that $h = f_j(x_i - c_i)$. It follows that $h \in J_C$ as $f_j \mid h$ and $h \in J_{\{c\}}$ as $(x_i - c_i) \mid h$. Then $h(c^{'}) = 0$ for any $c^{'} \in C \cup \{c\}$, which means $h \in J_{C\cup\{c\}}$. Thus, there exists $f_i^{'} \in CF(J_{C\cup\{c\}})$, such that $h = h_1 f_i^{'}$. But as there are no elements in $L \cup N$ that are multiples of any other elements in $L \cup N$, $h = f_i^{'}$, furthermore, $h \in CF(J_{C\cup\{c\}})$.

For the converse inclusion, let us postulate $h \in CF(J_{C\cup\{c\}})$, then there is some $f_j \in CF(J_C)$, such that $h = h_1 f_j$ as $J_{C\cup\{c\}} \subset J_C$, where h_1 is a pseudo-monomial. Similarly, there are some elements of $CF(J_{\{c\}})$, and let us just assume it is $(x_i - c_i)$ such that $h = (x_i - c_i)h_2$, where h_2 is a pseudo-monomial. So h is a multiple of $f_j(x_i - c_i)$. Therefore, $f_j(x_i - c_i)$ either belongs to $L \cup N$ or is a multiple of elements in $L \cup N$. But as $h \in CF(J_{C\cup\{c\}})$, h can-not be a multiple of elements in $L \cup N$. Then $h \in L \cup N$, and $CF(J_{C\cup\{c\}}) \in L \cup N$.

Next, we will illustrate the operational steps and improvements of the new specific algorithm through a specific example.

Example 1: Let $C = (001, 010, 110)$, then the canonical form of it is $CF(J_C) = (x_1x_2, (1 + x_1)(1 + x_2), x_0(1 + x_1), x_0x_2)$, and we denote them as f_1, f_2, f_3, f_4, respectively.

Now add a new code $c = (111)$.

Next, we calculate $CF(J_{C\cup\{c\}})$ according to the specific algorithm.

$$f_1(c) = 1, f_2(c) = 0, f_3(c) = 0, f_4(c) = 1.$$

Add f_2, f_3 to L and f_1, f_4 to M. Chose f_1 from M, then $\mathrm{lt}(f_1) = x_1x_2$. Besides, $(x_i - c_i)$ is $(x_0 - 1)$, $(x_1 - 1)$, and $(x_2 - 1)$, respectively. After verification, it was found that $\mathrm{lt}(f_1)$ and $x_1 - 1, x_2 - 1$, share indicators, respectively. Therefore, according to Theorem 3.3, there is no need to calculate the product of them.

$\gcd(\mathrm{lt}(f_1), x_0 - 1) = 1$ and $f_1(x_0 - 1) = x_1x_2(x_0 - 1)$, which cannot be expressed as the product of an elemental constituent of the set L, so add it to N and label it as f_5.

We chose f_4, which shares indicators respectively with $(x_0 - 1)$ and $(x_2 - 1)$, as a result, it would be unnecessary to proceed with calculating the product between them. Hence, we will simply skip this step.

But $\gcd(\mathrm{lt}(f_4), (x_1 - 1)) = 1$, therefore, we need to calculate the product of them,

$$f_4(x_1 - 1) = x_0x_2(x_1 - 1),$$

which is not divisible by any element of L, so add it to N, label it as f_6.

Finally, we obtained the $CF(J_C) = (f_2, f_3, f_5, f_6)$.

Remark To further clarify, the aforementioned calculation examples demonstrate the benefits of reducing the number of multiplication operations required for calculating the products of f_1 and $(x_1 - 1), (x_2 - 1)$, as well as f_2 and $(x_0 - 1), (x_2 - 1)$. and we also do not need to verify whether $(x_j - c_j - 1)$ divides f_i. By eliminating these additional multiplication calculations and simplifying the process, we achieve improved efficiency and significantly reduce the amount of time required for the overall computation.

5. Conclusions

In this study, we introduce an improved, specific algorithm for computing canonical forms of neural ideals. This new specific algorithm provides several criteria, resulting in a reduction of computational requirements and increased efficiency. Moreover, we introduce a new RF-relation and extend it from J_C to $I(C)$. Identifying such receptive field relationships is an interesting direction for future work.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant Nos. 12201204, 11971161, and 12271154, and the Natural Science Foundation of Hunan Provincial under Grant Nos. 2022JJ30234 and 2023JJ40275, Scientific Research Fund of Hunan Province Education Department under Grant Nos. 21A0299 and 22A0334.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. C. Curto, V. Itskov, A. Veliz-Cuba, N. Youngs, The neural ring: an algebraic tool for analyzing the intrinsic structure of neural codes, *Bull. Math. Biol.*, **75** (2013), 1571–1611. https://doi.org/10.1007/s11538-013-9860-3
2. C. Curto, N. Youngs, Neural ring homomorphisms and maps between neural codes, in *Topological Data Analysis*, (2020), 163–180. https://doi.org/10.1007/978-3-030-43408-3_7
3. C. Giusti, E. Pastalkova, C. Curto, V. Itskov, Clique topology reveals intrinsic geometric structure in neural correlations, *PNAS*, **112** (2015), 13455–13460. https://doi.org/10.1073/pnas.1506407112
4. C. S. Gunturkun, J. Jeffries, J. Sun, Polarization of neural rings, *J. Algebra Appl.*, **19** (2020), 2050146. https://doi.org/10.1142/S0219498820501467
5. H. Geller, R. G. Rebecca, Canonical forms of neural ideals, preprint, arXiv:2209.09948.
6. D. Li, J. Liu, L. Zheng, A zero-dimensional valuation ring is 1-Gröbner , *J. Algebra*, **484** (2017), 334–343. https://doi.org/10.1016/j.jalgebra.2017.04.015
7. L. Zheng, D. Li, J. Liu, An improvement for GVW, *J. Syst. Sci. Complexity*, **35** (2022), 427–436. https://doi.org/10.1007/s11424-021-9051-5
8. L. Zheng, D. Li, J. Liu, Some improvements for the specific algorithm of Gröbner bases over dual valuation domain, *Electron. Res. Arch.*, **31** (2023), 3999–4010. https://doi.org/10.3934/era.2023203
9. A. Morvant, Strengthening relationships between neural ideals and receptive fields, preprint, arXiv:1803.03204.
10. E. Petersen, N. Youngs, R. Kruse, D. Miyata, R. Garcia, L. D. G. Puente, Neural ideals in SageMath, in *Mathematical Software – ICMS 2018*, (2018), 182–190. https://doi.org/10.1007/978-3-319-96418-8_22
11. A. R. Perez, L. F. Matusevich, A. Shiu, Neural codes and the factor complex, *Adv. Appl. Math.*, **114** (2020), 101977. https://doi.org/10.1016/j.aam.2019.101977