Neural Ring Homomorphisms and Maps Between Neural Codes

Carina Pamela Curto and Nora Youngs

Abstract Neural codes are binary codes that are used for information processing and representation in the brain. In previous work, we have shown how an algebraic structure, called the *neural ring*, can be used to efficiently encode geometric and combinatorial properties of a neural code (Curto et al., Bull Math Biol 75(9), 2013). In this work, we consider maps between neural codes and the associated homomorphisms of their neural rings. In order to ensure that these maps are meaningful and preserve relevant structure, we find that we need additional constraints on the ring homomorphisms. This motivates us to define *neural ring homomorphisms*. Our main results characterize all code maps corresponding to neural ring homomorphisms as compositions of five elementary code maps. As an application, we find that neural ring homomorphisms behave nicely with respect to convexity. In particular, if $\mathcal{C}$ and $\mathcal{D}$ are convex codes, the existence of a surjective code map $\mathcal{C} \to \mathcal{D}$ with a corresponding neural ring homomorphism implies that the minimal embedding dimensions satisfy $d(\mathcal{D}) \leq d(\mathcal{C})$.

1 Introduction

A major challenge of mathematical neuroscience is to determine how the brain processes and stores information. By recording the spiking from a population of neurons, we obtain insights into their coding properties. A *neural code* on n neurons is a subset $\mathcal{C} \subset \{0, 1\}^n$, with each binary vector in $\mathcal{C}$ representing an on-off pattern of neural activity. This type of neural code is referred to in the neuroscience literature as a combinatorial neural code [15, 16] as it contains only the combinatorial information of which neurons fire together, ignoring precise spike times and firing

C. P. Curto (✉)
The Pennsylvania State University, State College, PA, USA
e-mail: ccurto@psu.edu; cpc16@psu.edu

N. Youngs
Colby College, Waterville, ME, USA
e-mail: nora.youngs@colby.edu

N. A. Baas et al. (eds.), *Topological Data Analysis*, Abel Symposia 15,
https://doi.org/10.1007/978-3-030-43408-3_7

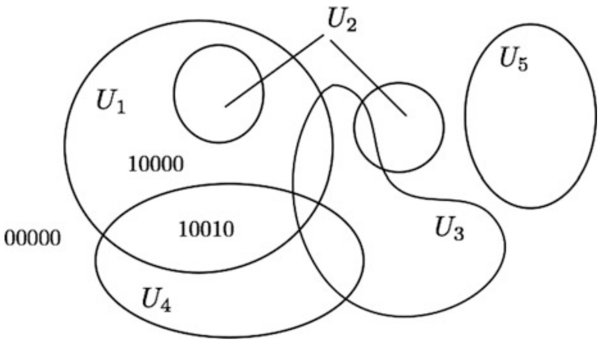


Fig. 1 An arrangement of five receptive fields $U_1, \ldots, U_5$ in a stimulus space. Here, U_i represents the receptive field of neuron i. The full code for the arrangement is: $C = \{$00000, 10000, 01000, 00100, 00010, 00001, 11000, 10100, 10010, 01100, 00110, 10110$\}$

rates. These codes can be analyzed to determine important features of the neural data, using tools from coding theory [7] and topology [3, 6].

A particularly interesting kind of neural code arises when neurons have *receptive fields*. These neurons are selective to a particular type of stimulus; for example, place cells respond to the animal's spatial location [14], and orientation-tuned neurons in visual cortex respond to the orientation of an object in the visual field [11]. The neuron's *receptive field* is the specific subset of the stimulus space to which that neuron is particularly sensitive, and within which the neuron exhibits a high firing rate. If all receptive fields for a set of neurons are known, one can infer the expected neural code by considering the overlap regions formed by the receptive fields. Figure 1 shows an arrangement of receptive fields, and gives the corresponding neural code.

An arrangement of receptive fields whose regions correspond precisely to the neural code $\mathcal{C}$ is called a *realization* of $\mathcal{C}$. If the receptive fields can be chosen to be convex, then $\mathcal{C}$ is a *convex* neural code. Many neural codes are observed to be convex [3, 17]. In this case, we can leverage results from the extensive literature on arrangements of convex sets, such as Helly's theorem [9], to give bounds on the dimension of the space of stimuli (see [8] for some examples). Note that the code in Fig. 1 is convex, even though the realization depicted there is not; we leave it as an exercise for the reader to draw a convex realization of this code.

In previous work [8], we introduced the neural ideal and the corresponding neural ring, algebraic objects associated to a neural code that capture its combinatorial properties. Thus far, work involving the neural ring has been primarily concerned with using the algebraic framework to extract structural information about the code [5, 8] and to determine which codes have convex realizations [4]. However, a neural code $\mathcal{C}$ is not an isolated object, but rather a member of a family of codes. We define a *code map* from a code $\mathcal{C}$ to another code $\mathcal{D}$ to be any well-defined function $q : \mathcal{C} \rightarrow \mathcal{D}$. A code map may preserve important structural properties of a code, or it may completely ignore them and just send codewords to codewords in an arbitrary manner. We are interested in a set of 'nice' code maps that reflect meaningful relationships between the corresponding neural codes. Our primary motivating examples of 'nice' maps are those which leave the structure of a code essentially intact:

1. **Permutation:** If $\mathcal{C}$ and $\mathcal{D}$ are identical codes up to a re-ordering of the neurons, then the permutation map $q : \mathcal{C} \to \mathcal{D}$ is 'nice,' as it perfectly preserves the combinatorial structure.
2. **Adding or removing trivial neurons:** A code $\mathcal{C}$ can be trivially changed by appending an extra neuron that has uniform behavior in all codewords – i.e., always silent or always firing. Similarly, a code that has a neuron which is always "on" or always "off" is structurally equivalent to the code obtained by removing this trivial neuron, and the corresponding maps are 'nice.'

One way to obtain a code with trivial neurons is via localization. For example, consider the code in Fig. 1, restricted to the codewords whose regions are all contained inside U_1. This code has five codewords: $\mathcal{C}' = \{10000, 11000, 10100, 10010, 10110\}$. There is a natural map $q : \mathcal{C}' \to \mathcal{D}$ that drops neurons 1 and 5, which are both trivial, to obtain $\mathcal{D} = \{000, 100, 010, 001, 011\}$, which is structurally equivalent to $\mathcal{C}'$. Not all code maps respect the structure of the corresponding codes, however. For example, there is no guarantee that an arbitrary code map $\mathcal{C}' \to \mathcal{D}$ will reflect the fact that these codes are structurally equivalent.

In this article, we consider how maps between neural codes relate to neural rings, as first defined in [8]. Our main questions are, simply:

Questions What types of maps between neural rings should be considered 'nice'? How should we define neural ring homomorphisms? What other code maps correspond to nice maps between the associated neural rings?

These questions are analogous to studying the relationship between maps on algebraic varieties and their associated rings [1]. However, as we will see in the next section, the standard notions of ring homomorphism and isomorphism are much too weak to capture any meaningful structure in the related codes. Recent work [12] considered which ring homomorphisms preserve neural ideals as a set, and described corresponding transformations to codes through that lens. In this article, we will define a special class of maps, called *neural ring homomorphisms*, that capture the structure of the nice code maps described above, and also guide us to discover additional code maps which should be considered 'nice.' Our main result, Theorem 3.4, characterizes all code maps that correspond to neural ring homomorphisms and isomorphisms as compositions of five elementary code maps (including the two 'nice' types above). As an application, Theorem 4.3 shows that any surjective code map with a corresponding neural ring homomorphism preserves convexity and can only lower the minimal embedding dimension.

The organization of this paper is as follows. In Sect. 2, we review the neural ring of a code and describe the relevant pullback map, which gives a correspondence between code maps and ring homomorphisms. This allows us to see why the usual ring homomorphisms between neural rings are insufficiently restrictive. In Sect. 3 we define *neural ring homomorphisms*, a special class of maps that preserve code structure, and state Theorem 3.4. In Sect. 3.1 we take a closer look at the new elementary code maps that emerged in Theorem 3.4, and we prove the theorem. Finally, in Sect. 4, we state and prove Theorem 4.3, showing that surjective code maps corresponding to neural ring homomorphisms are particularly well-behaved with respect to convexity.

2 Neural Rings and the Pullback Map

First, we briefly review the definition of a neural code and its associated neural ring, as previously defined in [8]. We then present the pullback map, which naturally relates maps between codes to homomorphisms of neural rings.

Definition 2.1 A *neural code* on n neurons is a set of binary firing patterns of length n. Given neural codes $\mathcal{C} \subset \{0,1\}^n$ and $\mathcal{D} \subset \{0,1\}^m$, on n and m neurons, a *code map* is any function $q : \mathcal{C} \to \mathcal{D}$.

For any neural code $\mathcal{C} \subset \{0,1\}^n$, we define the associated ideal $I_\mathcal{C} \subset \mathbb{F}_2[x_1, \dots, x_n]$ as follows:

$$I_\mathcal{C} \overset{\text{def}}{=} \{f \in \mathbb{F}_2[x_1, \dots, x_n] \mid f(c) = 0 \text{ for all } c \in \mathcal{C}\}.$$

The *neural ring* $R_\mathcal{C}$ is then defined to be $R_\mathcal{C} = \mathbb{F}_2[x_1, \dots, x_n]/I_\mathcal{C}$.

Note that the neural ring $R_\mathcal{C}$ is precisely the ring of functions $\mathcal{C} \to \{0,1\}$, denoted $\mathbb{F}_2^\mathcal{C}$. Since the ideal $I_\mathcal{C}$ consists of polynomials that vanish on $\mathcal{C}$, we can make use of the ideal-variety correspondence to obtain an immediate relationship between code maps and ring homomorphisms by using the pullback map. Given a code map $q : \mathcal{C} \to \mathcal{D}$, each $f \in R_\mathcal{D}$ is a function $f : \mathcal{D} \to \{0,1\}$, and therefore we may "pull back" f by q to a function $f \circ q : \mathcal{C} \to \{0,1\}$, which is an element of $R_\mathcal{C}$. Hence, for any $q : \mathcal{C} \to \mathcal{D}$, we may define the pullback map $q^* : R_\mathcal{D} \to R_\mathcal{C}$, where $q^*(f) = f \circ q$, as illustrated below:

$$\begin{tikzcd} \mathcal{C} \arrow[r, "q"] \arrow[dr, "q^* f = f \circ q"'] & \mathcal{D} \arrow[d, "f"] \\ & \{0,1\} \end{tikzcd}$$

It is easy to check that for any code map $q : \mathcal{C} \to \mathcal{D}$, the pullback $q^* : R_\mathcal{D} \to R_\mathcal{C}$ is a ring homomorphism. In fact, the pullback provides a bijection between code maps and ring homomorphisms, as the following proposition states.

Proposition 2.2 *There is a 1–1 correspondence between code maps $q : \mathcal{C} \to \mathcal{D}$ and ring homomorphisms $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$, given by the pullback map. That is, given a code map $q : \mathcal{C} \to \mathcal{D}$, its pullback $q^* : R_{\mathcal{D}} \to R_{\mathcal{C}}$ is a ring homomorphism; conversely, given a ring homomorphism $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$, there is a unique code map $q_\phi : \mathcal{C} \to \mathcal{D}$ such that $q_\phi^* = \phi$.*

Proposition 2.2 is a special case of [1, Proposition 8, p. 234]. Note that both this proposition and the next can be easily understood from a category-theoretic perspective. Specifically, the pullback map construction provides a natural contravariant functor from the category of binary codes (with code maps as morphisms) to the category of rings and ring homomorphisms. This functor sends each code $\mathcal{C}$ to the corresponding neural ring $R_{\mathcal{C}}$, and each code map q to the corresponding pullback q^*. However, to illustrate precisely how all the objects interact, and for the benefit of readers unfamiliar with category theory language, we include our own elementary proof of this proposition in Sect. 2.1. In particular, we show how to go backwards from ring homomorphisms to code maps, so that the reader can see explicitly how to construct q_ϕ from ϕ.

Unfortunately, Proposition 2.2 makes it clear that ring homomorphisms $R_{\mathcal{D}} \to R_{\mathcal{C}}$ need not preserve the structure of the associated codes, as *any* code map has a corresponding ring homomorphism. The next proposition tells us that even ring *isomorphisms* are quite weak: any pair of codes with the same number of codewords admits an isomorphism between the corresponding neural rings.

Proposition 2.3 *A ring homomorphism $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ is an isomorphism if and only if the corresponding code map $q_\phi : \mathcal{C} \to \mathcal{D}$ is a bijection.*

Propositions 2.2 and 2.3 highlight the main difficulty with using ring homomorphism and isomorphism alone: the neural rings are rings of functions from $\mathcal{C}$ to $\{0, 1\}$, and the abstract structure of such a ring depends solely on the number of codewords, $|\mathcal{C}|$. Considering such rings abstractly, independent of their presentation, reflects no additional structure—not even the code length (the number of neurons, n) matters. In particular, we cannot track the behavior of the variables x_i that represent individual neurons. This raises the question: what algebraic constraints can be put on homomorphisms between neural rings in order to capture a meaningfully restricted class of code maps?

2.1 The Pullback Correspondence: A Closer Look

Before moving on to defining a more restricted class of homomorphisms, we introduce some notation to take a closer look at neural rings, and how the correspondence between code maps and homomorphisms occurs. Using this, we provide concrete and elementary proofs of Propositions 2.2 and 2.3.

Elements of neural rings may be denoted in different ways. First, they can be written as polynomials, where it is understood that the polynomial is a representative of its equivalence class mod I_C. Alternatively, using the vector space structure, an element of R_C can be written as a function $C \to \{0, 1\}$ defined completely by the codewords that support it. We will make use of the latter idea frequently, so it is helpful to identify a canonical basis of characteristic functions $\{\rho_c \mid c \in C\}$, where

$$\rho_c(v) = \begin{cases} 1 \text{ if } & v = c, \\ 0 \text{ otherwise.} \end{cases}$$

In polynomial notation,

$$\rho_c = \prod_{c_i=1} x_i \prod_{c_j=0} (1 - x_j),$$

where c_i represents the ith component of codeword c. The characteristic functions ρ_c form a basis for R_C as an $\mathbb{F}_2$-vector space, and they have several useful properties:

- Each element f of R_C can be represented as the formal sum of basis elements for the codewords in its support: $f = \sum\limits_{\{c \in C | f(c)=1\}} \rho_c$.
- In particular, we can write $x_i = \sum\limits_{\{c \in C \,|\, c_i=1\}} \rho_c$. So, if $c_i = c_j$ for all $c \in C$, then $x_i = x_j$. Likewise, if $c_i = 1$ for all $c \in C$, we have $x_i = 1$.
- The product of two basis elements is 0 unless they are identical:

$$\rho_c \rho_d = \begin{cases} \rho_c \text{ if } & c = d, \\ 0 \quad \text{otherwise} \end{cases}.$$

- If 1_C is the identity of R_C, then $1_C = \sum\limits_{c \in C} \rho_c$.

Once we have a homomorphism $\phi : R_{\mathcal{D}} \to R_C$, we necessarily have a map which sends basis elements of $R_{\mathcal{D}}$ to sums of basis elements in R_C. We will now show how this illustrates the corresponding code map. First, a technical lemma.

Lemma 2.4 *For any ring homomorphism $\phi : R_{\mathcal{D}} \to R_C$, and any element $c \in C$, there is a unique $d \in \mathcal{D}$ such that $\phi(\rho_d)(c) = 1$.*

Proof To prove existence, note that $\sum_{c \in C} \rho_c = 1_C = \phi(1_{\mathcal{D}}) = \phi(\sum_{d \in \mathcal{D}} \rho_d) = \sum_{d \in \mathcal{D}} \phi(\rho_d)$. For each $c \in C$, $1 = \rho_c(c) = \left(\sum_{c' \in C} \rho_{c'}\right)(c) = \left(\sum_{d \in \mathcal{D}} \phi(\rho_d)\right)(c)$, and thus $\phi(\rho_d)(c) = 1$ for at least one $d \in \mathcal{D}$. To prove uniqueness, suppose there exist distinct $d, d' \in \mathcal{D}$ such that $\phi(\rho_d)(c) = \phi(\rho_{d'})(c) = 1$. Then as ϕ is a ring homomorphism, we would have $1 = (\phi(\rho_d)\phi(\rho_{d'}))(c) = \phi(\rho_d \rho_{d'})(c) = \phi(0)(c) = 0$, but this is a contradiction. Thus such a d must be unique. □

This result allows us to describe the unique code map corresponding to any ring homomorphism.

Definition 2.5 Given a ring homomorphism $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$, we define the associated code map $q_\phi : \mathcal{C} \to \mathcal{D}$ as follows:

$$q_\phi(c) = d_c$$

where d_c is the unique element of $\mathcal{D}$ such that $\phi(\rho_{d_c})(c) = 1$, guaranteed by Lemma 2.4.

Using this definition, we are able to prove Proposition 2.2.

Proof (Proposition 2.2) It is easy to check that the pullback q^* is a ring homomorphism; we now prove that any homomorphism can be obtained as the pullback of a code map. Given a ring homomorphism $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$, define q_ϕ as above. We must show that the $q_\phi^* = \phi$, and moreover that q_ϕ is the only code map with this property.

The fact that $q_\phi^* = \phi$ holds essentially by construction: let $f \in R_{\mathcal{D}}$, so $f = \sum_{f(d)=1} \rho_d$. Then, for any $c \in \mathcal{C}$,

$$q_\phi^*(f)(c) = f(q_\phi(c)) = \sum_{f(d)=1} \rho_d(q_\phi(c)) = \sum_{f(d)=1} \rho_d(d_c) = \begin{cases} 1 & \text{if } f(d_c) = 1 \\ 0 & \text{if } f(d_c) = 0 \end{cases}$$

whereas, remembering from above that there is exactly one $d \in \mathcal{D}$ such that $\phi(\rho_d)(c) = 1$ and that this d may or may not be in the support of f, we have

$$\phi(f)(c) = \sum_{f(d)=1} \phi(\rho_d)(c) = \begin{cases} 1 & \text{if } d_c \in f^{-1}(1) \\ 0 & \text{if } d_c \notin f^{-1}(1) \end{cases} = \begin{cases} 1 & \text{if } f(d_c) = 1 \\ 0 & \text{if } f(d_c) = 0 \end{cases}.$$

Thus, $\phi = q_\phi^*$.

Finally, to see that q_ϕ is the only code map with this property, suppose we have a different map $q \neq q_\phi$. Then there is some $c \in \mathcal{C}$ with $q(c) \neq q_\phi(c)$; let $d_c = q_\phi(c)$, so $q(c) \neq d_c$. Then $\phi(\rho_{d_c})(c) = 1$ by definition, but $q^*(\rho_{d_c})(c) = \rho_{d_c}(q(c)) = 0$ as $q(c) \neq d_c$. So q^* does not agree with ϕ and hence ϕ is not the pullback of q, so q_ϕ is the unique code map with pullback ϕ. □

The following example illustrates the connection between a homomorphism ϕ and the corresponding code map q_ϕ.

Example 2.6 Let $\mathcal{C} = \{110, 111, 010, 001\}$ and $\mathcal{D} = \{00, 10, 11\}$. Let $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ be defined by $\phi(\rho_{11}) = \rho_{110} + \rho_{111} + \rho_{010}$, $\phi(\rho_{00}) = \rho_{001}$, and $\phi(\rho_{10}) = 0$. Then the corresponding code map q_ϕ will have $q_\phi(110) = q_\phi(111) = q_\phi(010) = 11$, and $q_\phi(001) = 00$. Note that there is no element $c \in \mathcal{C}$ with $q_\phi(c) = 10$ so q_ϕ is not surjective.

Finally, we provide a proof of Proposition 2.3.

Proof (Proposition 2.3) Note that $R_\mathcal{C} \cong \mathbb{F}_2^{|\mathcal{C}|}$ and $R_\mathcal{D} \cong \mathbb{F}_2^{|\mathcal{D}|}$, and $\mathbb{F}_2^{|\mathcal{C}|} \cong \mathbb{F}_2^{|\mathcal{D}|}$ if and only if $|\mathcal{C}| = |\mathcal{D}|$. Suppose ϕ is an isomorphism; then we must have $|\mathcal{C}| = |\mathcal{D}|$. If q_ϕ is not injective, then there is some $d \in \mathcal{D}$ such that $\phi(\rho_d)(c) = 0$ for all $c \in \mathcal{C}$. But then $\phi(\rho_d) = 0$, which is a contradiction since ϕ is an isomorphism so $\phi^{-1}(0) = \{0\}$. Thus q_ϕ is injective, and since $|\mathcal{C}| = |\mathcal{D}|$, this means q_ϕ is a bijection.

On the other hand, suppose $q_\phi : \mathcal{C} \to \mathcal{D}$ is a bijection. Then $|\mathcal{C}| = |\mathcal{D}|$, so $R_\mathcal{C} \cong R_\mathcal{D}$, and as both are finite, $|R_\mathcal{C}| = |R_\mathcal{D}|$. Consider an arbitrary element $f \in R_\mathcal{C}$. For each $c \in f^{-1}(1)$, there is a unique $d \in \mathcal{D}$ so $\phi(\rho_d) = c$; furthermore as q_ϕ is a bijection, all these d are distinct. Then

$$\phi\Big(\sum_{\substack{d=q_\phi(c),\\ c\in f^{-1}(1)}} \rho_d\Big) = \sum_{\substack{d=q_\phi(c)\\ c\in f^{-1}(1)}} \phi(\rho_d) = \sum_{c\in f^{-1}(1)} \rho_c = f.$$

Hence ϕ is surjective, and since $|R_\mathcal{C}| = |R_\mathcal{D}|$, ϕ is also bijective and hence an isomorphism. □

3 Neural Ring Homomorphisms

In order to define a restricted class of ring homomorphisms that preserve certain structural similarities of codes, we consider how our motivating maps (permutation and adding or removing trivial neurons) preserve structure. In each case, note that the code maps act by preserving the activity of each neuron: we do not combine the activity of neurons to make new ones, or create new neurons that differ in a nontrivial way from those we already have. Following this idea, we restrict to a class of maps that respect the elements of the neural ring corresponding to individual neurons: the variables x_i. Here we use the standard notation $[n]$ to denote the set $\{1, \dots, n\}$.

Definition 3.1 Let $\mathcal{C} \subset \{0, 1\}^n$ and $\mathcal{D} \subset \{0, 1\}^m$ be neural codes, and let $R_\mathcal{C} = \mathbb{F}_2[y_1, \dots, y_n]/I_\mathcal{C}$ and $R_\mathcal{D} = \mathbb{F}_2[x_1, \dots, x_m]/I_\mathcal{D}$ be the corresponding neural rings. A ring homomorphism $\phi : R_\mathcal{D} \to R_\mathcal{C}$ is a *neural ring homomorphism* if $\phi(x_j) \in \{y_i \mid i \in [n]\} \cup \{0, 1\}$ for all $j \in [m]$. We say that a neural ring homomorphism ϕ is a *neural ring isomorphism* if it is a ring isomorphism and its inverse is also a neural ring homomorphism.

It is important to remember that when we refer to the 'variables' of $R_\mathcal{D}$, we actually mean the *equivalence class* of the variables under the quotient ring structure. Thus, it is possible in some cases to have $x_i = x_j$, or $x_i = 0$, depending on whether these variables give the same function on all codewords. We now provide some examples to illustrate neural ring homomorphisms.

Example 3.2 Here we consider three different code maps: one that corresponds to a neural ring isomorphism, one that corresponds to a neural ring homomorphism but not to a neural ring isomorphism, and one that does not correspond to a neural ring homomorphism at all.

1. Let $\mathcal{D} = \{0000, 1000, 0001, 1001, 0010, 1010, 0011\}$, and let $C = \{0000, 0001, 0010, 0011, 0100, 0101, 0110\}$.
   Define $\phi : R_{\mathcal{D}} \to R_C$ as follows:
   $$\begin{array}{ll} \phi(\rho_{0000}) = \rho_{0000}, & \phi(\rho_{1000}) = \rho_{0001} \\ \phi(\rho_{0001}) = \rho_{0010}, & \phi(\rho_{1001}) = \rho_{0011} \\ \phi(\rho_{0010}) = \rho_{0100}, & \phi(\rho_{1010}) = \rho_{0101} \\ \phi(\rho_{0011}) = \rho_{0110}, & \end{array}$$
   Note that $\phi(x_1) = \phi(\rho_{1000} + \rho_{1010}) = \rho_{0001} + \rho_{0101} = y_4$, and $\phi(x_2) = \phi(0) = 0 = y_1$. By similar calculations, we have $\phi(x_3) = y_2$, and $\phi(x_4) = y_3$. Thus, ϕ is a neural ring homomorphism; in fact, since ϕ is a ring isomorphism and its inverse is a neural ring homomorphism sending $\phi^{-1}(y_1) = 0 = x_2$, $\phi^{-1}(y_2) = x_3, \phi^{-1}(y_3) = x_4$, and $\phi^{-1}(y_4) = x_1$, ϕ is a neural ring isomorphism.
2. Let $\mathcal{D} = \{000, 110\}$ and $C = \{00, 01, 10\}$. Define $\phi : R_{\mathcal{D}} \to R_C$ by $\phi(\rho_{000}) = \rho_{00} + \rho_{10}$ ad $\phi(\rho_{110}) = \rho_{01}$. In $R_{\mathcal{D}}$, $x_1 = x_2 = \rho_{110}$, and $x_3 = 0$. In R_C, we have $y_1 = \rho_{10}$ and $y_2 = \rho_{01}$. Under this map, we find $\phi(x_1) = \phi(x_2) = y_1$ and $\phi(x_3) = 0$, so ϕ is a neural ring homomorphism. However, it is not a neural ring isomorphism, as it is not a ring isomorphism.
3. Let $\mathcal{D} = \{00, 10\}$ and $C = \{00, 10, 01\}$. Define the ring homomorphism $\phi : R_{\mathcal{D}} \to R_C$ as follows: $\phi(\rho_{00}) = \rho_{00}$, $\phi(\rho_{10}) = \rho_{10} + \rho_{01}$. In $R_{\mathcal{D}}$, $x_1 = \rho_{10}$. However, $\phi(x_1) = \rho_{10} + \rho_{01}$, which is not equal to either $y_1 = \rho_{10}$, $y_2 = \rho_{01}$, $1 = \rho_{00} + \rho_{10} + \rho_{01}$, or 0. Thus, ϕ is not a neural ring homomorphism.

It is straightforward to see that the composition of neural ring homomorphisms is again a neural ring homomorphism.

Lemma 3.3 *If $\phi : R_{\mathcal{D}} \to R_C$ and $\psi : R_{\mathcal{E}} \to R_{\mathcal{D}}$ are neural ring homomorphisms, then their composition $\phi \circ \psi$ is also a neural ring homomorphism. If ϕ and ψ are both neural ring isomorphisms, then their composition $\phi \circ \psi$ is also a neural ring isomorphism.*

As we have seen in Example 3.2, both permutations and appending a trivial neuron correspond to neural ring isomorphisms. The following theorem introduces three other types of elementary code maps, which also yield neural ring homomorphisms. All of these code maps are meaningful in a neural context, and preserve the behavior of individual neurons. And, as seen in Theorem 3.4, it turns out that *all* neural ring homomorphisms correspond to code maps that are compositions of these five elementary types of maps. The proof is given in Sect. 3.1.

Theorem 3.4 *A map $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ is a neural ring homomorphism if and only if q_ϕ is a composition of the following elementary code maps:*

1. *Permutation*
2. *Adding a trivial neuron (or deleting a trivial neuron)*
3. *Duplication of a neuron (or deleting a neuron that is a duplicate of another)*
4. *Neuron projection (deleting a not necessarily trivial neuron)*
5. *Inclusion (of one code into another)*

Moreover, ϕ is a neural ring isomorphism if and only if q_ϕ is a composition of maps (1)–(3).

The ability to decompose any 'nice' code map into a composition of these five elementary maps has immediate consequences for answering questions about neural codes. For example, one of the questions that motivated the definition of the neural ring and neural ideal was that of determining which neural codes are *convex*. In Sect. 4, we look at how each of these maps affect convexity.

The following example provides a sense of what these different operations mean.

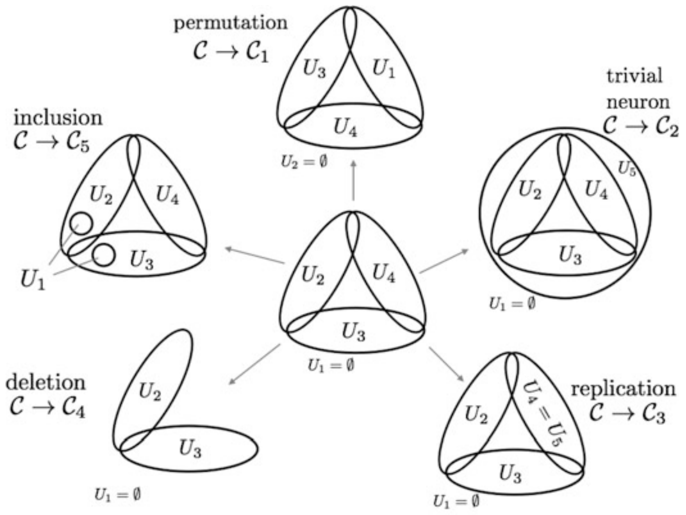
Example 3.5 In Fig. 2 we show a code $\mathcal{C}$, and the resulting codes $\mathcal{C}_1, \ldots, \mathcal{C}_5$ after applying the following elementary code maps:

1. the cyclic permutation (1234) ($\mathcal{C}_1$),
2. adding a trivial always-on neuron ($\mathcal{C}_2$),
3. duplication of neuron 4 ($\mathcal{C}_3$),
4. deleting neuron 4 (projecting onto neurons 1–3) ($\mathcal{C}_4$)
5. an inclusion map into a larger code ($\mathcal{C}_5$).

The effects of these code maps on a realization of $\mathcal{C}$ are shown on in Fig. 2. The succeeding columns in the table below give the image of $\mathcal{C}$ under each of the five code maps.

3.1 Proof of Theorem 3.4

To prove Theorem 3.4, we will first focus on the structure of neural ring homomorphisms. As neural ring homomorphisms strictly control the possible images of variables, they can be described succinctly by an index 'key' vector that captures the information necessary to determine the map. Since the index for the first variable will use the symbol '1', we will where necessary denote the multiplicative identity 1 of the ring with the symbol u to distinguish the two. Throughout, we will use the notation c_i to indicate the ith component of a codeword c.



$\mathcal{C}$	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_4$	$\mathcal{C}_5$
0000	0000	00001	00000	000	0000
0001	1000	00011	00011	(000)	0001
0010	0001	00101	00100	001	0010
0011	1001	00111	00111	(001)	0011
0100	0010	01001	01000	010	0100
0101	1010	01011	01011	(010)	0101
0110	0011	01101	01100	011	0110
					1100
					1010

Fig. 2 A code $\mathcal{C}$ and its image under five elementary code maps. (Top) The effect of each codeword on a realization of $\mathcal{C}$. (Bottom) A table showing how each codeword of $\mathcal{C}$ is transformed by each map. In each case, the code map sends a codeword $c \in \mathcal{C}$ to the codeword in its row

Definition 3.6 Let $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ be a neural ring homomorphism, where $\mathcal{C}$ and $\mathcal{D}$ are codes on n and m neurons, respectively. The *key vector* of ϕ is the vector $V \in \{1, \dots, n, 0, u\}^m$ such that

$$V_j = \begin{cases} i & \text{if } \phi(x_j) = y_i \\ 0 & \text{if } \phi(x_j) = 0 \\ u & \text{if } \phi(x_j) = 1 \end{cases}.$$

This key vector completely describes a neural ring homomorphism, since once the image of each variable is determined the rest of the homomorphism is given by the usual properties of homomorphism. In cases where we have $y_i = y_k$ for some i, k, then only one representative of the equivalence class need appear in V.

Because of the close correspondence of code maps and ring homomorphisms, the key vector also completely determines the associated code map. The following lemma gives the explicit relationship.

Lemma 3.7 *Let $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ be a neural ring homomorphism with key vector V. Then the corresponding code map $q_\phi : \mathcal{C} \to \mathcal{D}$ is given by $q_\phi(c) = d$, where*

$$d_j = \begin{cases} c_i & \text{if } V_j = i \\ 0 & \text{if } V_j = 0 \\ 1 & \text{if } V_j = u \end{cases}.$$

Furthermore, any code map that aligns with a key vector must be associated to a neural ring homomorphism.

Lemma 3.8 *Let $\mathcal{C}$ and $\mathcal{D}$ be codes on n and m neurons, respectively. Suppose $q : \mathcal{C} \to \mathcal{D}$ is a code map and $V \in \{1, \dots, n, 0, u\}^m$ such that q is described by V; that is, for all $c \in \mathcal{C}$, $q(c) = d$ where $d_j = \begin{cases} c_i & \text{if } V_j = i \\ 0 & \text{if } V_j = 0 \\ 1 & \text{if } V_j = u \end{cases}$. Then the associated ring homomorphism ϕ_q is a neural ring homomorphism with key vector V.*

Proof Let q be as described above, and ϕ_q the associated ring homomorphism. We will show that for $j \in [m]$, we have $\phi_q(x_j) = \begin{cases} x_i & \text{if } V_j = i \\ 0 & \text{if } V_j = 0 \\ 1 & \text{if } V_j = 1 \end{cases}$ and thus that ϕ_q is a neural ring homomorphism with key vector V. We will examine the three options for V_j separately.

First, suppose $V_j = i \in [n]$. Then for all $c \in \mathcal{C}$, we have $q(c)_j = c_i$, and thus that $x_j(q(c)) = c_i$. Hence, $x_j \circ q = y_i$, since both functions act the same on all codewords $c \in \mathcal{C}$. But by definition of the pullback map, $\phi(x_j) = x_j \circ q$, so $\phi(x_j) = y_i$. Next, suppose $V_j = 0$. Then for all $c \in \mathcal{C}$ we have $q(c)_j = 0$ and thus that $x_j(q(c)) = 0$. Hence, $x_j \circ q = 0$, since both functions act the same on all codewords $c \in \mathcal{C}$. But by definition of the pullback map, $\phi(x_j) = x_j \circ q$, so $\phi(x_j) = 0$ in this case.

Finally, suppose $V_j = u$. Then for all $c \in \mathcal{C}$ we have $q(c)_j = 1$ and thus that $x_j(q(c)) = 1$. Hence, $x_j \circ q = 1$, since both functions act the same on all codewords $c \in \mathcal{C}$. But by definition of the pullback map, $\phi(x_j) = x_j \circ q$, so $\phi(x_j) = 1$ in this case. □

Remark 3.9 It is important to note here that the key vector for a particular code map may not be unique. In Example 3.2 (1), we saw an example of a permutation code map that could be described by key vector $(4, 1, 2, 3)$. However, as $\phi(x_2) = y_1 = 0$, we could replace this key vector with $(4, 0, 2, 3)$ and describe the same homomorphism. In cases like these, either choice is valid. However, this does not mean that the corresponding homomorphism is not unique.

Now that we have shown that neural ring homomorphisms (and their corresponding code maps) are precisely those determined by key vectors, we need only show the following:

- All five code maps listed have key vectors.
- Any code map with a key vector can be written as a composition of these five maps.
- The first three code maps correspond precisely to neural ring isomorphisms.

To see that all five elementary code maps in Theorem 3.4 have key vectors, we simply exhibit the key vector for each. In the process, we will show that the first three maps correspond to neural ring isomorphisms. To describe these code maps, we will consider an arbitrary word $c \in \mathcal{C}$, written as $c = c_1 c_2 \cdots c_n$, and describe the image $q(c) \in \mathcal{D}$. Throughout, $\mathcal{C}$ is a code on n neurons and $\mathcal{D}$ is a code on m neurons.

1. Permutation maps: If the code map $q : \mathcal{C} \to \mathcal{D}$ is a permutation map, then $n = m$, $q(\mathcal{C}) = \mathcal{D}$, and each codeword is permuted by the same permutation σ. That is, for each $c \in \mathcal{C}$, we know $q(c) = c_{\sigma(1)} c_{\sigma(2)} \cdots c_{\sigma(n)}$. In this case, the key vector is given by $V_j = \sigma(j)$. As permutation yields a bijection on codewords, and the inverse permutation also has a key vector, permutation maps correspond to neural ring isomorphisms.
2. Adding a trivial neuron to the end of each codeword: in this case, $m = n + 1$ and $q(\mathcal{C}) = \mathcal{D}$. Consider first the case of adding a trivial neuron that is never firing to the end of each codeword, so that $q : \mathcal{C} \to \mathcal{D}$ is described by $q(c) = c_1 c_2 \cdots c_n 0$, and $q(\mathcal{C}) = \mathcal{D}$. The key vector is given by $V_j = j$ for $j \in [n]$ and $V_{n+1} = 0$. Similarly, if we add a neuron that is always firing, so $q(c) = c_1 \cdots c_n 1$, then $V_j = j$ for $j \in [n]$ and $V_{n+1} = u$. Such a map will be a bijection; moreover, the reverse map (where we delete the trivial neuron at the end of each word) also has a key vector: $W_i = i$ for all $i \in [n]$. Thus, this map (and its inverse) correspond to neural ring isomorphisms.
3. Adding a duplicate neuron to the end of each codeword: in this case, $m = n + 1$ and $q(\mathcal{C}) = \mathcal{D}$. If the new neuron $n + 1$ duplicates neuron i, then the code map is given by $q(c) = c_1 \cdots c_n c_i$, and the key vector is given by $V_j = j$ for $j \in [n]$ and $V_{n+1} = i$. Such a map will be a bijection on codewords, and moreover, the inverse code map corresponds to the key vector where $W_i = i$ for all $i \in [n]$, and so its inverse corresponds to a neural ring homomorphism. Thus, this map and its inverse correspond to neural ring isomorphisms.
4. Projection (deleting the last neuron): in this case, $m = n - 1$ and $q(\mathcal{C}) = \mathcal{D}$. The code map is given by $q(c) = c_1 \cdots c_{n-1}$ and we have the key vector $V_j = j$ for $j \in [n - 1]$.

   This map corresponds to a neural ring isomorphism precisely when the deleted neuron is either trivial, or a duplicate of another neuron. If neither of these hold, then there are two possibilities: either the code map is not a bijection, in which case the corresponding ring homomorphism is not an isomorphism, or the code map is a bijection, but the inverse will not be a neural ring homomorphism, as $\phi^{-1}(y_{n+1}) \notin \{x_1, \dots, x_m, 0, 1\}$.

5. Inclusion: in this case, $m = n$, and we have $q(c) = c$ for all $c \in \mathcal{C}$. However, we do not demand $q(\mathcal{C}) = \mathcal{D}$. Since in this case each codeword maps to itself, we can use the key vector $V_j = j$ for $j \in [n]$.

Finally, we prove the main substance of Theorem 3.4, which is that any code map corresponding to a neural ring homomorphism can be written as a composition of the five listed maps, and furthermore that any isomorphism requires only the first three.

Proof (Theorem 3.4) Let $\mathcal{C}$ and $\mathcal{D}$ be codes on n and m neurons, respectively, and let $\phi : R_{\mathcal{D}} \to R_{\mathcal{C}}$ be a neural ring homomorphism with corresponding code map q. Our overall steps will be as follows:

1. Append the image $q(c)$ to the end of each codeword c using a series of maps that duplicate neurons or add trivial neurons, as necessary.
2. Use a permutation map to move the image codeword $q(c)$ to the beginning, and the original codeword c to the end.
3. Use a series of projection maps to delete the codeword c from the end, resulting in only $q(c)$.
4. Use an inclusion map to include $q(\mathcal{C})$ into $\mathcal{D}$ if $q(\mathcal{C}) \subsetneq \mathcal{D}$.

First we define some intermediate codes: let $\mathcal{C}_0 = \mathcal{C}$. For $j = 1, \dots, m$, let

$$\mathcal{C}_j = \{(c_1, \dots, c_n, d_1, \dots, d_j) \mid c \in \mathcal{C}, d = q(c)\} \subset \{0, 1\}^{n+j}.$$

For $i = 1, \dots, n$, let

$$\mathcal{C}_{m+i} = \{(d_1, \dots, d_m, c_1, \dots, c_{n-i+1}) \mid c \in \mathcal{C}, d = q(c)\} \subset \{0, 1\}^{m+n-i+1}.$$

Finally, define $\mathcal{C}_{m+n+1} = q(\mathcal{C}) \subset \mathcal{D}$.

Now, for $j = 1, \dots, m$, let the code map $q_j : \mathcal{C}_{j-1} \to \mathcal{C}_j$ be defined for $v = (c_1, \dots, c_n, d_1, \dots, d_{j-1}) \in \mathcal{C}_{j-1}$ by $q_j(v) = (c_1, \dots, c_n, d_1, \dots, d_j) \in \mathcal{C}_j$. Since ϕ is a neural ring homomorphism, the associated code map q has a corresponding key vector V; note that q_j is described by the key vector $W^j = (1, \dots, n + j - 1, V_j)$, so q_j is either repeating a neuron, or adding a trivial neuron, depending on whether $V_j = i$, or one of $u, 0$.

Next, take the permutation map given by $\sigma = (n + 1, \dots, n + m, 1, \dots, n)$, so all the newly added neurons are at the beginning and all the original neurons are at the end. That is, define $q_\sigma : \mathcal{C}_m \to \mathcal{C}_{m+1}$ so if $v = (v_1, \dots, v_{n+m})$, then $q_\sigma(v) = (v_{n+1}, \dots, v_{n+m}, v_1, \dots, v_n)$.

We then delete the neurons $m + 1$ through $n + m$ one by one in n code maps. That is, for $i = 1, \dots, n$ define $q_{m+i} : \mathcal{C}_{m+i} \to \mathcal{C}_{m+i+1}$ by $q_{m+i}(v) = (v_1, \dots, v_{m+n-i})$.

Lastly, if $q(\mathcal{C}) \subsetneq \mathcal{D}$, then add one last inclusion code map $q_a : q(\mathcal{C}) \hookrightarrow \mathcal{D}$ to add the remaining codewords of $\mathcal{D}$.

Thus, given $c = (c_1, \dots, c_n)$ with $q(c) = d = (d_1, \dots, d_m)$, the first m steps give us $q_m \circ \cdots \circ q_1(c) = (c_1, \dots, c_n, d_1, \dots, d_m) = x$. The permutation then gives us $q_\sigma(x) = (d_1, \dots, d_m, c_1, \dots, c_n) = y$, and then we compose $q_{m+n} \circ \cdots \circ q_{m+1}(y) = (d_1, \dots, d_n) = d = q(c)$. Finally, if $q(\mathcal{C}) \subsetneq \mathcal{D}$, we do our inclusion map, but as $q_a(d) = d$, the overall composition is a map $\mathcal{C} \to \mathcal{D}$ taking c to $q_\phi(c) = d$ as desired. At each step, the map we use is from our approved list.

Finally, to show that code maps corresponding to neural ring isomorphisms only use maps (1)–(3), note that in the case that ϕ is a neural ring isomorphism, it is in particular an isomorphism, so the corresponding code map q_ϕ is a bijection and thus $q_\phi(\mathcal{C}) = \mathcal{D}$; no inclusion map is necessary in the last step of the process described above. We have also noted above that projection maps correspond to neural ring isomorphisms only when the deleted neuron is either trivial or a duplicate of another. Thus, only maps (1)–(3) are necessary to describe all neural ring isomorphisms. $\square$

4 Neural Ring Homomorphisms and Convexity

One of the questions which has motivated a deeper understanding of the neural ring is that of determining which neural codes are convex.

Definition 4.1 A neural code $\mathcal{C}$ on n neurons is *convex in dimension d* if there is a collection $\mathcal{U} = \{U_1, \dots, U_n\}$ of convex open sets in $\mathbb{R}^d$ such that $\mathcal{C} = \{c \in \{0,1\}^n \mid \left(\bigcap_{c_i=1} U_i\right) \setminus \left(\bigcup_{c_j=0} U_j\right) \neq \emptyset\}$. If additionally no such collection exists in $\mathbb{R}^{d-1}$, then d is known as the minimal embedding dimension of the code, denoted $d(\mathcal{C})$. If there is no dimension d where $\mathcal{C}$ is convex, then $\mathcal{C}$ is a *non-convex* code; in this case we use the convention $d(\mathcal{C}) = \infty$.

Example 4.2 In Example 3.5 (illustrated in Fig. 2), we showed the results of applying five elementary code maps to the code $\mathcal{C}$. In that case, code $\mathcal{C}$ and its images $\mathcal{C}_1 - \mathcal{C}_3$ are convex codes of dimension 2 and code $\mathcal{C}_4$ is convex of dimension 1. On the other hand, $\mathcal{C}_5$ cannot be realized with convex sets in any dimension, as $U_1 \cap U_2$ and $U_1 \cap U_3$ necessarily form a disconnection of U_1.

In general, determining whether or not a code has a convex realization is a difficult question. Some partial results exist that give guarantees of convexity or of non-convexity, or that bound the embedding dimension (see for example [2, 4, 5, 8, 10, 13]). One way to extend such results is to show that once a code is known to have certain properties related to convexity, we can generate other codes from it via code maps that would preserve these properties. The following theorem shows that if a *surjective* code map is 'nice' (i.e., has a corresponding neural ring homomorphism), then it preserves convexity and the embedding dimension can only decrease.

Theorem 4.3 *Let $\mathcal{C}$ be a code containing the all-zeros codeword and $q : \mathcal{C} \to \mathcal{D}$ a surjective code map corresponding to a neural ring homomorphism. Then if $\mathcal{C}$ is* *convex, $\mathcal{D}$ is also convex with $d(\mathcal{D}) \leq d(C)$; it follows that if $\mathcal{D}$ is not convex, then C is not convex.*

Corollary 4.4 *Let C be a code containing the all zeros codeword, and $q : C \to \mathcal{D}$ a code map corresponding to a neural ring isomorphism. Then C and $\mathcal{D}$ are either both convex, with $d(C) = d(\mathcal{D})$, or both not convex.*

The proof of this theorem and its corollary relies on Theorem 3.4, and in particular uses the decomposition of these code maps to reduce the convexity question to code maps of just the five elementary types. As Theorem 4.3 addresses all neural ring homomorphisms that correspond to surjective code maps, it covers any such maps that are composed of permutation, duplication, deletion, or adding on trivial neurons.

Note that the theorem would not necessarily hold for arbitrary surjective code maps that do *not* correspond to a neural ring homomorphism. It would be a simple matter to create a bijection between a non-convex and a convex code with the same number of codewords, which would correspond to a ring isomorphism, but would not preserve convexity.

The only non-surjective elementary code map corresponding to a neural ring homomorphism is inclusion, and this theorem cannot generally be extended to inclusion maps. Because the inclusion map can be used to include codes into arbitrary larger ones of the same length, it is possible to change convexity and dimension in arbitrary ways. The following examples show how to include convex codes in non-convex codes and vice versa, as well as ways to change the realization dimension by an arbitrary amount.

Example 4.5 Note that in Examples (1) and (3) below, we rely on results and constructions detailed in other work, especially [4].

1. Non-convex codes can be included into convex codes. If C is any non-convex code, then we can include C into the larger code $\Delta(C)$, the simplicial complex of C, which is necessarily convex. For more details, see for example [4, 10].
2. Convex codes (of arbitrary dimension) can also be included into non-convex codes. Let C_1 be a convex code on n neurons, and C_2 a non-convex code on m neurons. Define the code C to be the code C_1 with m always-zero neurons appended to the end of each codeword; note that C is still convex, by the arguments above. Similarly, define the code C' to be the code C_2 with n always-zero neurons appended to the beginning of each codeword. The code C' is still not convex, again by the previous theorem. Define the code $\mathcal{D}$ to be the code $C \cup C'$, and note that as the first n neurons never interact with the last m, this code is not convex, but we can include C into $\mathcal{D}$.
3. Even when we include one convex code into another convex code, examples exist that change the dimension arbitrarily far in either direction. Let $n > 2$ be arbitrarily large. Then, $C = \{0, 1\}^n \backslash \{11 \ldots 1\}$ (the code on n neurons with the all-ones codeword removed) is convex of dimension $n-1$. We can include C into the code $\mathcal{D} = \{0, 1\}^n$, which is convex of dimension 2, reducing the dimension by $n - 3$. We can also increase the dimension as far as we wish, for example

by including the simple one-dimensional code $\{00\dots0, 10\dots0\}$ into the code $C = \{0,1\}^n \backslash \{11\dots1\}$, which was convex of dimension $n-1$, increasing the dimension by $n-2$. For a further discussion of the dimension and convexity of these codes, see [4].

We now give the proof of Theorem 4.3 and Corollary 4.4.

Proof (Theorem 4.3) If C is a surjective code map corresponding to a neural ring homomorphism, then it can be written as a composition of just the first four maps described by Theorem 3.4, following the process outlined in the proof. Thus, to prove both theorem and corollary, it suffices to show that if a code C' is obtained from C via a projection map, then $d(C') \leq d(C)$, and that if C' is obtained from C via one of the first three maps, then $d(C') = d(C)$. In general, if a convex realization of C can be transformed, in the same dimension, into a convex realization for C', then we have shown both that C' is convex whenever C is, and also that $d(C') \leq d(C)$.

Permutation Maps If C' is obtained from C via a permutation map, then any convex realization $\mathcal{U}$ of C is also a realization of C' by permuting the labels on the sets accordingly. Likewise, any realization $\mathcal{U}'$ of C' is a realization of C, by permuting the labels inversely. Thus, C is convex if and only if C' is also convex, and in addition $d(C') = d(C)$.

Adding/Deleting a Trivial Neuron If C' is obtained from C by adding a trivial always-zero neuron $n+1$, then a realization $\mathcal{U}$ of C can be transformed into a realization of C' by adding a set $U_{n+1} = \emptyset$. Likewise, a convex realization $\mathcal{U}'$ of C' can be transformed into a convex realization of C by removing the set U_{n+1}, which is necessarily empty as neuron $n+1$ never fires. For the second case, if C' is obtained from C by adding a trivial always-one neuron $n+1$, then we can transform a realization $\mathcal{U}$ of C into a realization of C' by adding the set U_{n+1} that is made up of the entire ambient space X in which the realization is set. This ambient space may be assumed to be convex, as C contains the all-zeros codeword. Likewise, a realization of C' can be transformed to that for C by removing the set U_{n+1}. Thus, for such maps, C is convex if and only if C' is convex and, in addition, $d(C) = d(C')$.

Adding/Deleting a Duplicate Neuron If C' is obtained from C by duplicating neuron i to a new neuron $n+1$, then any convex realization $\mathcal{U}$ of C can be transformed into a convex realization of C' by adding a set U_{n+1} that is identical to the set U_i. Likewise, any convex realization $\mathcal{U}'$ of C' can also realize C, by removing the set U_{n+1} that must be identical to U_i. Since C is obtained from C' by deleting a duplicate neuron, this argument also works for deleting a duplicate neuron. Hence, under such maps, C is convex if and only if C' is convex, and in addition, $d(C) = d(C')$.

Projection (Deletion) Maps If C' is obtained from C by deleting neuron n, then a convex realization $\mathcal{U}$ of C can be transformed into a realization of C' by removing the set U_n from the realization. Thus, if C is convex, then C' must also be convex, and in particular, $d(C') \leq d(C)$. □

Acknowledgements CC was supported by NIH R01 EB022862 and NSF DMS-1516881, NSF DMS-1225666/1537228, and an Alfred P. Sloan Research Fellowship; NY was supported by the Clare Boothe Luce Foundation. The authors thank Katie Morrison, Mohamed Omar, and R. Amzi Jeffs for many helpful discussions.

References

1. Cox, D., Little, J., O'Shea, D.: Ideals, varieties, and algorithms, second edn. Undergraduate Texts in Mathematics. Springer-Verlag, New York (1997). An introduction to computational algebraic geometry and commutative algebra
2. Cruz, J., Giusti, C., Itskov, V., Kronholm, W.: On open and closed convex codes. Discrete and Computational Geometry **61**(2), 247–270 (2019)
3. Curto, C.: What can topology tells us about the neural code? Bulletin of the AMS **54**(1), 63–78 (2017)
4. Curto, C., Gross, E., Jeffries, J., Morrison, K., Omar, M., Rosen, Z., Shiu, A., Youngs, N.: What makes a neural code convex? SIAM Journal on Applied Algebra and Geometry **1**(1), 222–238 (2017)
5. Curto, C., Gross, E., Jeffries, J., Morrison, K., Rosen, Z., Shiu, A., Youngs, N.: Algebraic signatures of convex and non-convex codes. Journal of Pure and Applied Algebra (2018)
6. Curto, C., Itskov, V.: Cell groups reveal structure of stimulus space. PLoS Computational Biology **4**(10) (2008)
7. Curto, C., Itskov, V., Morrison, K., Roth, Z., Walker, J.: Combinatorial neural codes from a mathematical coding theory perspective. Neural computation **25**(7), 1891–1925 (2013)
8. Curto, C., Itskov, V., Veliz-Cuba, A., Youngs, N.: The neural ring : an algebraic tool for analyzing the intrinsic structure of neural codes. Bulletin of Mathematical Biology **75**(9) (2013)
9. Danzer, L., Grünbaum, B., Klee, V.: Helly's theorem and its relatives. In: Proc. Sympos. Pure Math., Vol. VII, pp. 101–180. Amer. Math. Soc., Providence, R.I. (1963)
10. Giusti, C., Itskov, V.: A no-go theorem for one-layer feedforward networks. Neural Computation **26**(11), 2527–2540 (2014)
11. Hubel, D.H., Wiesel, T.: Receptive fields of single neurons in the cat's striate cortex. Journal of Physiology **148**(3), 574–591 (1959)
12. Jeffs, R.A., Omar, M., Youngs, N.: Neural ideal preserving homomorphisms. J. Pure and Applied Algebra **222**(11), 3470–3482 (2018)
13. Lienkaemper, C., Shiu, A., Woodstock, Z.: Obstructions to convexity in neural codes. Adv. Appl. Math. **85**, 31–59 (2017)
14. O'Keefe, J., Dostrovsky, J.: The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat. Brain Research **34**(1), 171–175 (1971)
15. Osborne, L., Palmer, S., Lisberger, S., Bialek, W.: The neural basis for combinatorial coding in a cortical population response. Journal of Neuroscience **28**(50), 13522–13531 (2008)
16. Schneidman, E., Puchalla, J., Segev, R., Harris, R., Bialek, W., Berry II, M.: Synergy from silence in a combinatorial neural code. J. Neuroscience **31**(44), 15732–15741 (2011)
17. Yartsev, M., Ulanovsky, N.: Representation of three-dimensional space in the hippocampus of flying bats. Science **340**(6130), 367–372 (2013)