Inductively Pierced Codes and Toric Ideals

Samuel Muthiah

Westmont College

July 17, 2017

Place Cells

- In 2014 John O'Keefe received the Nobel Prize for his discovery of place cells
- Place cells are part of the way certain mammals' brains identify where there are spatially
- Place cells fire in approximately convex regions

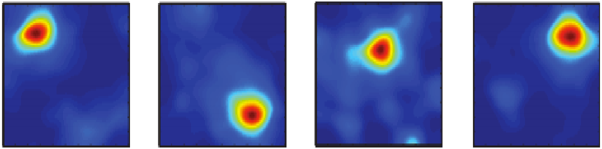


Figure: Place Cells

Neural Codes

Definition

A **neural code** on n neurons is a set of binary strings $\mathcal{C} \subseteq \{0,1\}^n$. The elements of $\mathcal{C}$ are called *codewords*.

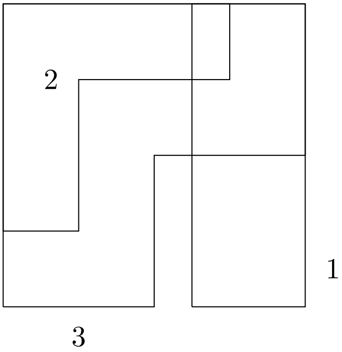
$$\mathcal{C} = \{000, 100, 001, 101, 011, 111\}$$

Neural Codes

Definition

A **realization** of a code $\mathcal{C}$ on n neurons is a collection of sets $\mathcal{U} = \{U_1, \ldots, U_n\}$ such that $\mathcal{C}(\mathcal{U}) = \mathcal{C}$.

$$\mathcal{C} = \{000, 100, 001, 101, 011, 111\}$$

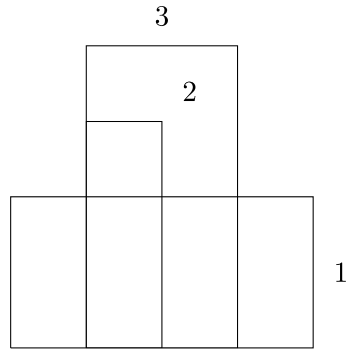


Neural Codes

Definition

A code $\mathcal{C}$ is **convex** if there exists a realization of $\mathcal{C}$ by convex sets.

$$\mathcal{C} = \{000, 100, 001, 101, 011, 111\}$$



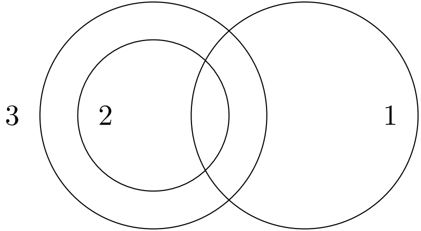
Neural Codes

Definition

The realization of a code $\mathcal{C}$ is **well-formed** if

- Curves intersect at a finite number of points
- At any given point, at most two curves intersect
- Each zone is connected

$$\mathcal{C} = \{000, 100, 001, 101, 011, 111\}$$

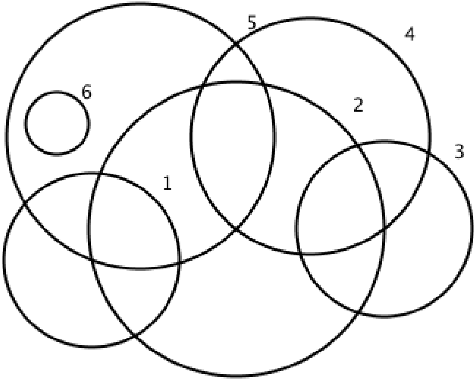


Need for Algorithms

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

Need for Algorithms

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



k-Piercings

Definition

A **k-piercing** is a curve that pierces (intersects) k other curves and that adds 2^k zones when added to an existing diagram.

k-Piercings

Definition

A **k-piercing** is a curve that pierces (intersects) k other curves and that adds 2^k zones when added to an existing diagram.

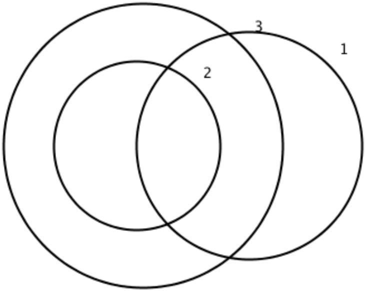


Figure: Example of a 1-Piercing

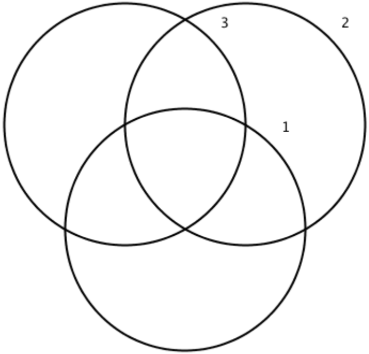


Figure: Example of a 2-Piercing

k-Inductively Pierced

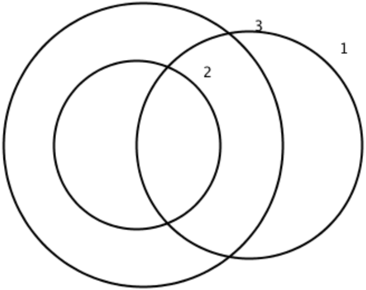
Definition

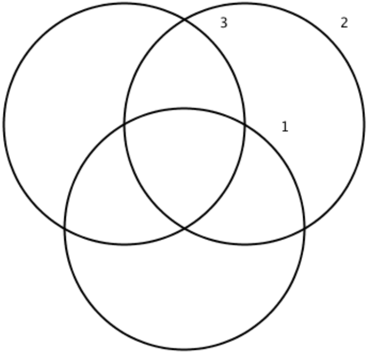
A neural code $\mathcal{C}$ is **k-inductively pierced** if $\mathcal{C}$ has a $0-, 1-, ...,$ or $k-$ piercing λ and $\mathcal{C} - \lambda$ is k-inductively pierced.

k-Inductively Pierced

Definition

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





Toric Ideals

- Let $\mathcal{C} = \{c_1, ..., c_m\}$ be a neural code on n neurons
- Let $\phi_c : \mathbb{K}[p_c | c \in \mathcal{C} \backslash (0, .., 0)] \to \mathbb{K}[x_i | i \in [n]]$

$$p_c \mapsto \prod_{i \in supp(c)} x_i$$

Toric Ideals

- Let $\mathcal{C} = \{c_1, ..., c_m\}$ be a neural code on n neurons
- Let $\phi_c : \mathbb{K}[p_c | c \in \mathcal{C} \backslash (0, .., 0)] \to \mathbb{K}[x_i | i \in [n]]$

$$p_c \mapsto \prod_{i \in supp(c)} x_i$$

Definition

The **toric ideal of the neural code** $\mathcal{C}$ is $I_{\mathcal{C}} := \ker \phi_c$

$$p_{101}p_{110} - p_{111}p_{100} \mapsto x_1x_3 \cdot x_1x_2 - x_1x_2x_3 \cdot x_1 = 0$$

Identifying k-Piercings

Theorem (Gross-Obatake-Youngs)

Let $\mathcal{C}$ be well formed.

- *The neural code $\mathcal{C}$ is 0-inductively pierced if and only if $I_{\mathcal{C}} = \langle 0 \rangle$.*
- *If the neural code C is 0- or 1- inductively pierced then $I_{\mathcal{C}} = \langle 0 \rangle$ or generated by quadratics.*
- *If the neural code C has a 2-piercing then $I_{\mathcal{C}}$ contains a binomial of degree 3 of particular form, in particular $p_{111w}p_{000v}^2 - p_{100v}p_{010v}p_{001w}$ or $p_{111w} - p_{100\ldots0}p_{010\ldots0}p_{001w}$ where v, w are zones in $\mathcal{C}(\mathcal{U})$.*

Identifying k-Piercings

Take the code $\mathcal{C} = \{0001, 1001, 0101, 0011, 1101, 1011, 0111, 1111\}$.

One set of generators of its toric ideal is:

$$\langle -p_{1011}p_{0111} + p_{0011}p_{1111}, -p_{1101}p_{0111} + p_{0101}p_{1111},$$
$$-p_{0101}p_{1011} + p_{1001}p_{0111}, -p_{0011}p_{1101} + p_{1001}p_{0111},$$
$$-p_{1011}p_{1011} + p_{1001}p_{1111}, -p_{1001}p_{0101} + p_{0001}p_{1101},$$
$$-p_{1001}p_{0011} + p_{0001}p_{1011}, -p_{1001}p_{0111} + p_{0001}p_{1111},$$
$$-p_{0101}p_{0011} + p_{0001}p_{0111}\rangle.$$

Identifying k-Piercings

Take the code $\mathcal{C} = \{0001, 1001, 0101, 0011, 1101, 1011, 0111, 1111\}$.

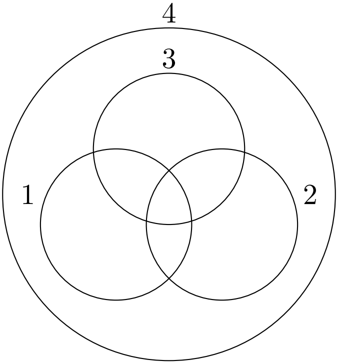


Figure: Realization of $\mathcal{C}$

Identifying k-Piercings

$$\langle -p_{1011}p_{0111} + p_{0011}p_{1111}, -p_{1101}p_{0111} + p_{0101}p_{1111}, -p_{0101}p_{1011} + p_{1001}p_{0111}, -p_{0011}p_{1101} + p_{1001}p_{0111}, -p_{1011}p_{1011} + p_{1001}p_{1111}, -p_{1001}p_{0101} + p_{0001}p_{1101}, -p_{1001}p_{0011} + p_{0001}p_{1011}, -p_{1001}p_{0111} + p_{0001}p_{1111}, -p_{0101}p_{0011} + p_{0001}p_{0111}\rangle$$

Constructing Cubics

$$p_{0001}p_{1111} - p_{1001}p_{0111}, p_{0001}p_{0111} - p_{0101}p_{0011} \in I_C$$
$$\Downarrow$$
$$p_{0001}(p_{1111}p_{0001} - p_{1001}p_{0111}) + p_{1001}(p_{0111}p_{0001} - p_{0101}p_{0011}) \in I_C$$

Constructing Cubics

$$p_{0001}p_{1111} - p_{1001}p_{0111}, p_{0001}p_{0111} - p_{0101}p_{0011} \in I_C$$
$$\Downarrow$$
$$p_{0001}(p_{1111}p_{0001} - p_{1001}p_{0111}) + p_{1001}(p_{0111}p_{0001} - p_{0101}p_{0011}) \in I_C$$

and

$$p_{0001}(p_{1111}p_{0001} - p_{1001}p_{0111}) + p_{1001}(p_{0111}p_{0001} - p_{0101}p_{0011})$$
$$=$$
$$p_{1111}p_{0001}^2 - p_{1001}p_{0101}p_{0011}$$

Special Quadratics

$$p_{000v}(p_{111w}p_{000v} - p_{110v}p_{001w}) + p_{001w}(p_{110w}p_{000v} - p_{100v}p_{010v})$$
$$p_{000v}(p_{111w}p_{000v} - p_{101v}p_{010w}) + p_{010w}(p_{101w}p_{000v} - p_{100v}p_{001v})$$
$$p_{000v}(p_{111w}p_{000v} - p_{011v}p_{100w}) + p_{001w}(p_{011w}p_{000v} - p_{010v}p_{001v})$$

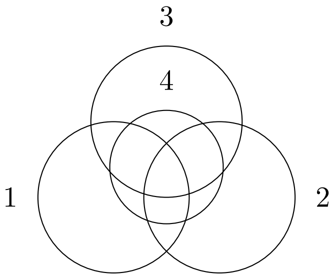
Special Quadratics

$$p_{000v}(p_{111w}p_{000v} - p_{110v}p_{001w}) + p_{001w}(p_{110w}p_{000v} - p_{100v}p_{010v})$$
$$p_{000v}(p_{111w}p_{000v} - p_{101v}p_{010w}) + p_{010w}(p_{101w}p_{000v} - p_{100v}p_{001v})$$
$$p_{000v}(p_{111w}p_{000v} - p_{011v}p_{100w}) + p_{001w}(p_{011w}p_{000v} - p_{010v}p_{001v})$$

Sufficient Condition?

Take the code

$$\mathcal{C} = \{1000, 0100, 0010, 1100, 1010, 1001, 0110, 0101, 0011, 1101, 1011, 0111, 1111\}$$



$$p_{1111} - p_{1000}p_{0100}p_{0011} \in I_C$$

Theorem (Hoch-M.-Obatake)

Let $\mathcal{C}$ be a well-formed code, and let $I_{\mathcal{C}}$ be its toric ideal. If there exists a cubic generator of $I_{\mathcal{C}}$ of the form $p_{111w}p_{000v}^2 - p_{100w}p_{010v}p_{001v}$, then $\mathcal{C}(\mathcal{U}) \setminus \bigcup_{i=4}^{m} U_i$ is 2-inductively pierced.

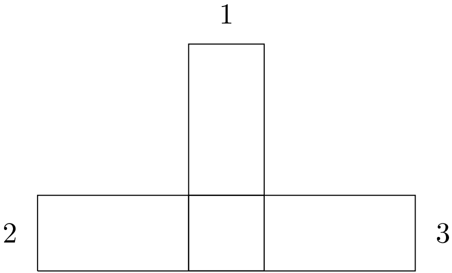
Corollary

If $p_{111w}p_{000v}^2 - p_{100w}p_{010v}p_{001v} \in I_{\mathcal{C}}$, then $\mathcal{C}$ is not 1-inductively pierced.

Example

$$p_{111w}p_{000v}^2 - p_{100v}p_{010v}p_{001w}$$

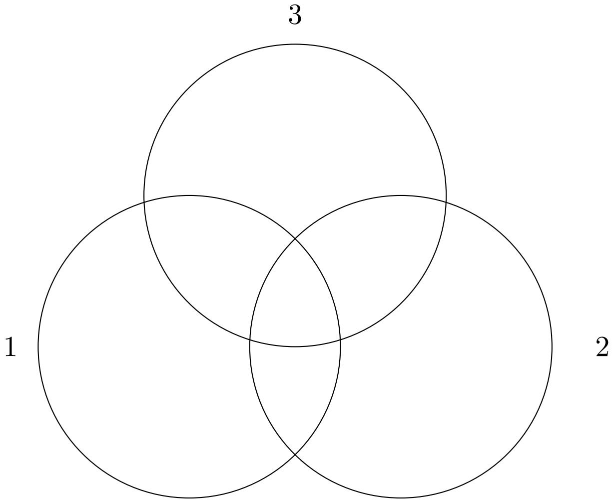
$$p_{111w} - p_{100\ldots0}p_{010\ldots0}p_{001w}$$



Example

$$p_{111w}p_{000v}^2 - p_{100v}p_{010v}p_{001w}$$

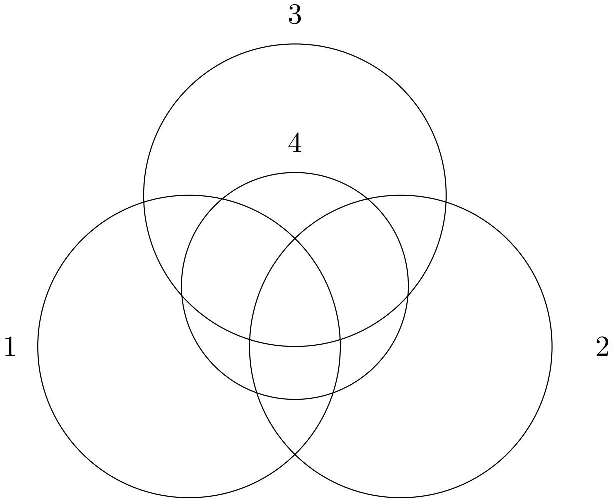
$$p_{111w} - p_{100\ldots0}p_{010\ldots0}p_{001w}$$



Example

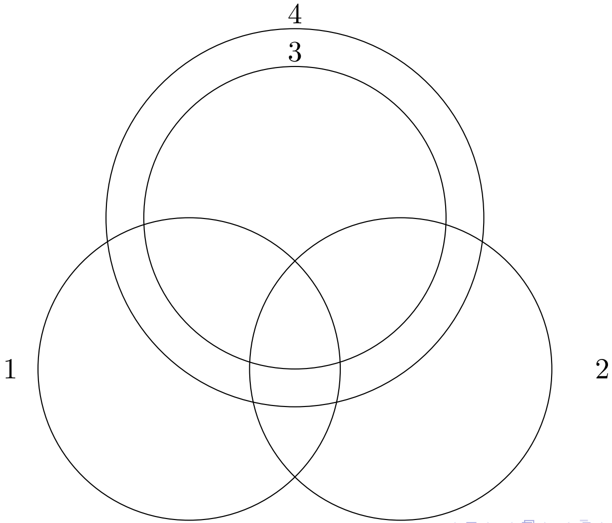
$$p_{111w}p_{000v}^2 - p_{100v}p_{010v}p_{001w}$$

$$p_{111w} - p_{100\ldots0}p_{010\ldots0}p_{001w}$$



Example

$$p_{111w}p_{000v}^2 - p_{100v}p_{010v}p_{001w}$$
$$p_{111w} - p_{100\ldots0}p_{010\ldots0}p_{001w}$$



Discussion

- Can we classify all possible ways of generating the cubics of a particular form so we can identify a 1-piercing from any generating set of the toric ideal?

Discussion

- Can we classify all possible ways of generating the cubics of a particular form so we can identify a 1-piercing from any generating set of the toric ideal?
- Which codes realizable in 2 dimensions are well-formed?

Discussion

- Can we classify all possible ways of generating the cubics of a particular form so we can identify a 1-piercing from any generating set of the toric ideal?
- Which codes realizable in 2 dimensions are well-formed?
- Can we identify which neurons potentially form a 2-piercing?

Acknowledgments

Special thanks to

- Molly Hoch
- Dr. Anne Shiu
- Nida Obatake
- Ola Sobieska
- Jonathan Tyler
- The National Science Foundation
- Texas A&M University

Thank You!