Generalized Adinkra Homology

Jacqueline A. Stone
Bard College

Generalized Adinkra Homology

A Senior Project submitted to
The Division of Science, Mathematics, and Computing
of
Bard College

by
Jackie Stone

Annandale-on-Hudson, New York
May, 2011

Abstract

Adinkras are graphs that encode the supersymmetric pairings between particles in physics. However they are also cubical complexes,which are structures built not only from points and lines, but also squares, cubes, and hypercubes, and therefore have higher dimensional topological structure. Additionally, Adinkras are equivalent to N-cubes quotiented by doubly-even codes of length N, which are a specific type of subgroup of ${\mathbb{Z}_2}^N$. Within this paper, we consider generalized Adinkras, which are N-cubes quotiented by any codes of length N.

Homology is an algebraic invariant of topological spaces. In order to learn more about their topology, we compute the homology of a variety of generalized Adinkras. Within this paper we examine the relationship between the codes by which we are quotienting and the homology of the cubes quotiented by these codes. Our main result is that the first homology group of any N-cube quotiented by a code of length N is isomorphic to the code itself.

Contents

List of Figures

Dedication

To my family and friends, for their love and support.

Acknowledgments

First I'd like to thank my adviser, Greg Landweber, for his time, patience, enthusiasm, and excellent guidance. Working with him made my senior project both intellectually fulfilling and a lot of fun. I'd also like to thank Matt Deady for being on my senior project board, for advising me for the past four years, and for teaching me Plato, Quantum Mechanics and everything in between. I'd like to thank Jim Belk for all of his suggestions for my project, for his contributions to my mathematical knowledge in general, and for being on my senior project board.

Additionally, I'd like to thank the entire Bard Mathematics Department for helping me discover my passion for math and for all the help and support they have given me over the past four years. Finally, I'd like to thank Julie Bergner for giving me my first taste of Algebraic Topology, Marisa Hughes for her helpful correspondence about homology and orbifolds, and Chris Stone for helping me write the 7 pages of code that can be found in the appendix.

1 Introduction

In algebraic topology, a common way to learn more about topological spaces is to associate algebraic invariants with them. These algebraic invariants help us classify spaces. Two of the major invariants that are used are the fundamental group (and other homotopy groups) and homology and cohomology groups. Homology groups are often easier to compute than homotopy groups, since homology groups can be computed combinatorially, whereas homotopy groups need to be computed topologically. Therefore, even if we do not know what these spaces look like topologically, if we know the type of cells the space is built from, we can compute its homology. Additionally, homotopy groups are not necessarily abelian, whereas homology groups always are. In this paper we will be computing cubical homology of Adinkras and generalizations of Adinkras.

Adinkras, as defined by Faux and Gates, are graphs that encode information about off-shell supersymmetry [3] [6]. However, in addition to being graphs, Adinkras are actually cubical complexes. That means that Adinkras have additional, higher dimensional topological structure. Figure 1.0.1 is an example of an $N = 4$ Adinkra. As proved in [4] [5], Adinkras are equivalent to N-cubes quotiented by a class of codes called doubly-even

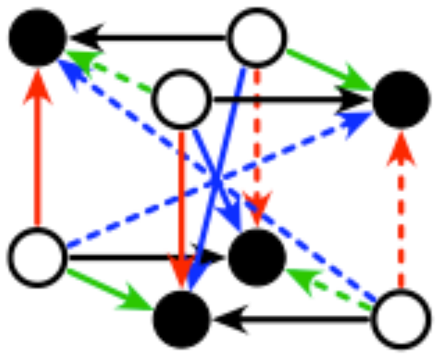


Figure 1.0.1. An $N = 4$ Adinkra [2]

codes. Codes are subgroups of ${\mathbb{Z}_2}^N$, and they are used as error correcting mechanisms in the transmission of binary or digital data. There will be a more detailed explanation of Adinkras, codes, and quotienting cubes by codes in Sections 2.2, 2.1, and 3.2. Our original hope was to be able to use homology to classify the topological spaces that these Adinkras are associated with. However, we soon realized that the topology of these quotients were more complicated than we originally thought, and that rather than manifolds, these spaces were actually orbifolds. Manifolds are spaces that look locally like $\mathbb{R}^N$, whereas orbifolds look locally like a quotient of $\mathbb{R}^N$ by a finite group, and are obtained when we quotient by a group where the action fixes some points [12]. Since in many of our cases the action of the code on the cube is not free of fixed points, the way we were computing the homology of these quotients turned out to be combinatorial, rather than topological.

To learn more about how the homology of these spaces behaves, we decided to compute the homology of generalized Adinkras, which are cubes quotiented by any type of code, not just doubly even codes. This allowed us to notice certain relationships between the code and the cubical $\mathbb{Z}_2$-homology of these spaces.

In Chapter 2, we will give a more formal introduction to codes, Adinkras, and homology. In Chapter 3 we will explain how to compute the homology of cubical complexes, introduce the notation we are using to talk about these quotients, and describe exactly how we are creating these quotient spaces. In Chapter 4, we compute the homology of the $N = 4$, $N = 5$ and $N = 6$ Adinkras, and then list the homology of various other generalized Adinkras. After computing the homology of a variety of generalized Adinkras, we formulated several conjectures, some false and some true, about the relationship between the homology of these quotients and the codes by which we were quotienting. In Chapter 5 we present these conjectures with either proofs or counterexamples. Our first theorem is that that the homology of the N-cube quotiented by the maximal code of length N (in other words, ${\mathbb{Z}_2}^N$) is the same as the homology of the N-torus, which is the product of N 1-spheres. The second is that the first homology group of an N-cube quotiented by a code of length N is isomorphic to the code itself. In Chapter 6 we will introduce some other ways we explored to compute homology, such as Equivariant Homology, Integer Homology, and the homology of the cross polytopes, and discuss some topics of future study.

2
Background

2.1 Codes

Since a generalized Adinkra is a cube quotiented by a code, we will have to specify what we mean by a code. We are specifically referring to binary linear codes.

Definition 2.1.1. A *binary linear code* of length N is a linear subspace of ${\mathbb{Z}_2}^N$ as a vector space over $\mathbb{Z}_2$ or equivalently a subgroup of ${\mathbb{Z}_2}^N$ [4]. △

The elements of a code C are called codewords, and the weight of a codeword is the number of 1s in the codeword. For example, if $C = \{0000, 1111, 1100, 0011\}$ then the codeword 1111 has weight 4, the codewords 1100 and 0011 have weight 2, and the codeword 0000 has weight 0. An even code is a code which only has codewords with even weight. The code C above is an example of an even code. A doubly even code is a code which only has codewords with weights divisible by four, such as $\{1111, 0000\}$. We are specifically interested in doubly even codes because Adinkras are quotients of N-cubes by doubly even codes. However, we will be looking at N-cubes quotiented by codes with arbitrary codeword weights as well.

2.2 Generalized Adinkras

An Adinkra is a specific type of graph with additional markings on its vertices and edges that encode information about off-shell supersymmetry. It turns out these graphs are cubical complexes, which means they are actually topological objects. By learning about the topological spaces that these Adinkras are associated with, we hoped to be able to learn more about the algebra or the physics that they represent.

From Reference [2], we have that every connected Adinkra is the quotient of an N-cube by a binary linear, doubly even code C of length N. We will actually be looking at generalized Adinkras, which are N-cubes quotiented by binary linear codes of length N, where the code words can have any weight. In the next chapter we will explain what it means to quotient a cube by a code.

2.3 Homology

Homology is a method of associating a series of abelian groups to a topological space. The homology groups of a space are algebraic invariants. In other words, if two spaces have different homology, then they are not homeomorphic. Since Adinkras are cubical complexes, we will be using cubical homology, which is very similar to simplicial homology as presented in [8]. We will be computing homology using $\mathbb{Z}_2$-coefficients, which allows us to ignore signs. We were hoping that the homology groups of some of the Adinkras would correspond to the homology groups of topological spaces we were already familiar with, since that would suggest that these spaces may be the same, and could in turn tell us more about the Adinkra. As mentioned earlier, since these spaces turned out to be orbifolds, the homology did not end up telling us what we expected it to. However, when we started looking at the homology of generalized Adinkras, we noticed some relationships between the homology of the quotient space and the code by which we were quotienting.

Before we can define homology, we must first define the boundary homomorphism and the chain groups. The boundary homomorphism just maps an n-cube to its boundary. For example, consider a cube.

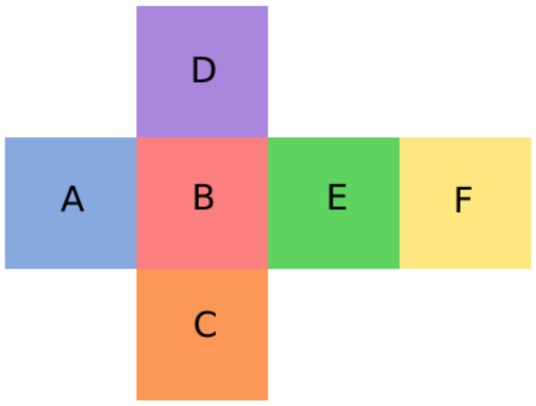


Figure 2.3.1. An unfolded cube

As you can see in Figure 2.3.1 it has 6 faces, so the boundary of the cube, which is denoted $\partial(cube)$ is just the sum of the 6 faces (again, we are working mod 2 so signs do not matter). Thus, $\partial(cube) = A + B + C + D + E + F$.

The boundary homomorphisms are maps between chain groups. An element of the nth chain group is a linear combination of the n-cells. For instance in the example of the cube above, a general element of the second chain group C_2 would be $\alpha A + \beta B + \gamma C + \delta D + \lambda E + \mu F$ where $\alpha, \beta, \gamma, \delta, \lambda, \mu$ are elements of the coefficient group.

Definition 2.3.1. The *nth chain group*, denoted C_n, is isomorphic to the direct sum of one copy of the coefficient group, G for each n-cube. In $\mathbb{Z}_2$-homology, if W_n is the set of n-cubes in a space, then $C_n = \mathbb{Z}_2[W_n]$. △

For example a cube has eight vertices, so $W_0 = \{1, 2, 3, 4, 5, 6, 7, 8\}$. Then C_0 is isomorphic to

$$\mathbb{Z}_2[1] \oplus \mathbb{Z}_2[2] \oplus \mathbb{Z}_2[3] \oplus \mathbb{Z}_2[4] \oplus \mathbb{Z}_2[5] \oplus \mathbb{Z}_2[6] \oplus \mathbb{Z}_2[7] \oplus \mathbb{Z}_2[8] \cong (\mathbb{Z}_2)^8.$$

Definition 2.3.2. Let $\omega \in C_n$. Then $\omega = \lambda^1 c_n^1 + \lambda^2 c_n^2 + ... + \lambda^k c_n^k$, where $\lambda^i \in \mathbb{Z}_2$ for $i \in \{1, ..., k\}$ and the $c_n^i s$ are n-cubes. Then the nth boundary map $\partial_n : C_n \longrightarrow C_{n-1}$ can be defined as

$$\partial_n(\omega) = \partial_n(\lambda^1 c_n^1 + \lambda^2 c_n^2 + ... + \lambda^k c_n^k) = \lambda^1 \partial_n(c_n^1) + \lambda^2 \partial_n(c_n^2) + ... + \lambda^k \partial_n(c_n^k),$$

where the $\partial_n(c_n^i)$ is just the sum of the $(n-1)$-cubes on the boundary of c_n^i, as shown in the example above. Note that this definition is specific to $\mathbb{Z}_2$-homology, and if we were not working over $\mathbb{Z}_2$ we would have to consider signs. △

Definition 2.3.3. The nth homology group is

$$H_n = \frac{\ker \partial_n}{\operatorname{im} \partial_{n+1}}.$$

△

This definition involves taking $\ker \partial_n$ modulo $\operatorname{im} \partial_{n+1}$. Therefore, we must show that $\operatorname{im} \partial_{n+1} \subset ker\partial_n$, or in other words, $\partial_n \circ \partial_{n+1} = 0$. First consider the square f in Figure 2.3.2, with edges a, b, c, d and vertices $1, 2, 3, 4$.

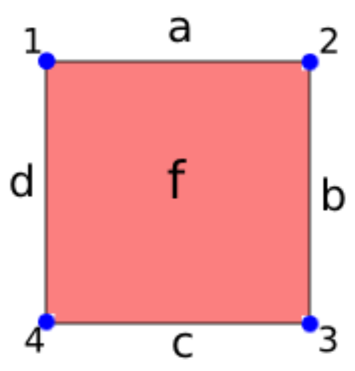


Figure 2.3.2. A square as a cubical complex

Then

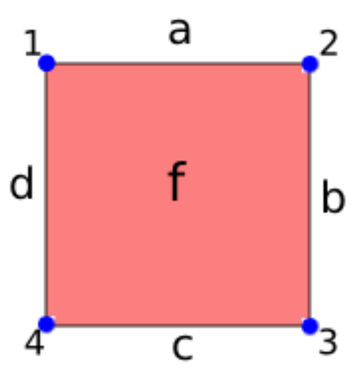
$$\begin{aligned}
\partial_1(\partial_2(f)) &= \partial_1(a+b+c+d) \\
&= \partial_1(a)+\partial_1(b)+\partial_1(c)+\partial_1(d) \\
&= 1+4+1+2+2+3+3+4 \\
&= 0.
\end{aligned}$$

The reason $\partial_1(\partial_2(f))$ is zero is because each vertex is attached to two edges. In general, on an $(n+1)$-cube, each $(n-1)$-cube will be attached to two n-cubes. Therefore, for any $(n+1)$-cube c_{n+1}, we have $(\partial_n \circ \partial_{n+1})(c_{n+1}) = 0$. By Definition 2.3.2 this implies that for any $(n+1)$-chain ω we have $(\partial_n \circ \partial_{n+1})(\omega) = 0$.

Since the nth chain group is generated by the n-cubes, we can find the image of ∂_n by finding the boundaries of all the n-cubes and then finding all possible linear combinations. Then, if we know the dimension of the chain group, we can use the rank-nullity theorem to find the dimension of the kernel of the ∂_n. In the next section we will do a simple example in which we compute the homology of a square.

3
Computing Homology

3.1 Simple Example: Homology of a Square



The chain groups are generated by the n-cells of the square. For example there are four 0-cells, which are the four vertices, a, b, c, d, so $C_0 \cong \mathbb{Z}_2[a] \oplus \mathbb{Z}_2[b] \oplus \mathbb{Z}_2[c] \oplus \mathbb{Z}_2[d] \cong (\mathbb{Z}_2)^4$. Similarly, there are four edges, so $C_1 \cong (\mathbb{Z}_2)^4$, and one face, so $C_2 \cong \mathbb{Z}_2$.

Now we consider the boundary of each of the cells. Each vertex is a 0-cube, so it has no boundary and thus $\partial_0(1) = \partial_0(2) = \partial_0(3) = \partial_0(4) = 0$. Therefore, $\ker \partial_0 \cong (\mathbb{Z}_2)^4$.

The boundary of each edge is the two vertices it connects. Thus we have:

$$\partial_1(a) = 1 + 4,$$

$$\partial_1(b) = 1 + 2,$$

$$\partial_1(c) = 2 + 3,$$

$$\partial_1(d) = 3 + 4.$$

We do not care about the signs, since we are working mod 2. Since the edges generate C_1, to find the image of the ∂_1 map, we need to find all the linear combinations of the boundaries of the edges. We end up with a total of 8 elements, so $\operatorname{im}\partial_1 \cong (\mathbb{Z}_2)^3$.

Finally, the boundary of the face f is the four edges. Thus, $\partial_2(f) = a + b + c + d$, so $\operatorname{im}\partial_2 \cong \mathbb{Z}_2$ and $\ker\partial_2 \cong 0$.

Now, we have enough information to compute the homology groups of the square.

$$H_0 \cong \frac{\ker\partial_0}{\operatorname{im}\partial_1} \cong \frac{(\mathbb{Z}_2)^4}{(\mathbb{Z}_2)^3} \cong \mathbb{Z}_2,$$

$$H_1 \cong \frac{\ker\partial_1}{\operatorname{im}\partial_2} \cong \frac{\mathbb{Z}_2}{\mathbb{Z}_2} \cong 0,$$

$$H_2 \cong \frac{\ker\partial_2}{\operatorname{im}\partial_3} \cong \frac{0}{0} \cong 0.$$

3.2 Notation and Quotienting by Codes

It becomes harder to keep track of all the vertices, edges, and faces with just letters and numbers once we get to higher dimensional cases. From now on we will use the notation presented in [2]. In other words, the N-cube is $[0,1]^N$, and we use a star $*$ to represent the interval $[0,1]$, so for example, the 3-cube can be represented by $(* * *)$. Each of the faces that is on the boundary of the 3-cube can be found by substituting a 1 or 0 in place of one of the stars. Therefore, the boundary of the 3-cube is as follows:

$$\partial_3(***) = (1**) + (0**) + (*1*) + (*0*) + (**1) + (**0).$$

Similarly, each of the edges that is on the boundary of a given face can be found by substituting a 1 or 0 in place of one of the stars. For example,

$$\partial_2(1**) = (11*) + (10*) + (1*1) + (1*0).$$

The same process can be repeated to find the boundaries of the edges. Once again, if we were not working over $\mathbb{Z}_2$, we would have to consider signs when computing the boundary.

We will often specifically refer to n-cubes with stars in the same entries as each other. For instance, the 2-cubes $(*1*)$ and $(*0*)$ or the 1-cubes $(10*)$ and $(00*)$ have all of their stars in the same entries. We will say that these n-cubes are of the same *type*. Later on, it will be especially useful to refer to specific edge types. We can specify an edge type by the location of the $*$.

Definition 3.2.1. An edge of *type m* is an edge with a star in the mth position. $\triangle$

Therefore, $(*10)$ is an edge of type 1, $(1*10)$ and $(1*0)$ are both examples of edges of type 2, although $(1*10)$ is on a 4-cube and $(1*0)$ is on a 3-cube, and so on.

When we talk about quotienting by a code, we mean that we are identifying n-cubes that can be mapped to each other by addition of one of the codewords. For instance, when we quotient the 3-cube by the code $\{000, 111\}$, the 2-cube $(1**)$ gets identified with the 2-cube $(0**)$, since $(1**) + (111) = (0**)$. Similarly, $(*1*)$ is identified with $(*0*)$, and $(**1)$ with $(**0)$. The same can be done with the 1-cubes, or edges. For example, $(11*)$ is identified with $(00*)$ and $(0*1)$ is identified with $(1*0)$. Note that two n-cubes can be identified only if they are of the same type. For example, there is no possible codeword that could be added to $(*11)$ to get $(1*1)$ since adding a codeword can change only entries that contain a 1 or 0.

It is important to note that our generalized Adinkras are really only quotients of cubes by codes at the level of the vertices. We are not taking the quotient of the cube as a topological space. Instead we are identifying certain vertices by addition of codewords. Therefore, the quotient Adinkra is not the same as the quotient topological space. For example, consider the 1-cube, which is just two vertices and one edge, as shown in Figure 3.2.1. Using our method, quotienting by the code $\{0, 1\}$ identifies the two vertices, giving us a circle, as shown in Figure 3.2.2, as opposed to identifying point along the line segment, giving us the original segment folded in half.

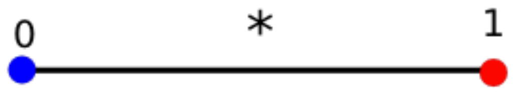


Figure 3.2.1. 1-cube before identification

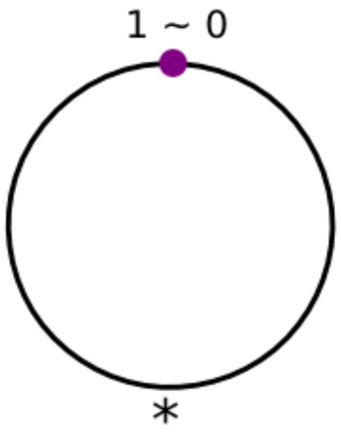


Figure 3.2.2. 1-cube quotiented by $\{0, 1\}$

4
Adinkra and Generalized Adinkra Homology

4.1 Homology of $N = 4$ Adinkras

The two connected four dimensional Adinkras, by which we mean 4-cubes quotiented by doubly-even codes, are the hypercube and the hypercube quotiented by the code $\{0000, 1111\}$. This is because $\{0000, 1111\}$ is the only non-trivial doubly-even code of length 4. Since we know that the hypercube is contractible, then we have $H_0 \cong \mathbb{Z}_2$ and $H_k = 0$ for all $k > 0$. The homology of the quotiented hypercube is more interesting.

To determine the chain groups, we need to consider how the code acts on the hypercube. Each vertex gets identified with its antipode by addition of the codeword 1111. Since a hypercube has 16 vertices, the quotiented hypercube has 8. Each edge gets identified with another edge, so the number of edges drops from 32 to 16. Similarly, the number of faces goes from 24 down to 12 and the number of cubes goes from 8 to 4. Therefore, the chain groups of the quotiented hypercube are as follows:

$$C_0 \cong (\mathbb{Z}_2)^8,$$
$$C_1 \cong (\mathbb{Z}_2)^{16},$$
$$C_2 \cong (\mathbb{Z}_2)^{12},$$
$$C_3 \cong (\mathbb{Z}_2)^4,$$
$$C_4 \cong \mathbb{Z}_2\,.$$

To compute the homology, we must find the image and kernel of each of the boundary maps. First, consider $\partial_4 : C_4 \to C_3$. This is the map from the 4-cube to its boundary, which consists of 3-cubes. We have

$$\begin{aligned}\partial_4(***\,*) = (1**\,*) + (0**\,*) + (*1**) + (*0**)\\ +(**1*) + (**0*) + (***1) + (***0).\end{aligned}$$

The code identifies $(1***) \sim (0***)$, and $(*1**) \sim (*0**)$ and so on. Therefore, since we are working mod 2, we have that $\partial_4(****) = 0$. That means that $\ker \partial_4 \cong \mathbb{Z}_2$ and $\operatorname{im} \partial_4 \cong 0$.

Now, we consider $\partial_3 : C_3 \to C_2$. To find the image and kernel of the boundary map, we need to consider where it maps each of the 3-cubes. We have

$$\partial_3(1***) = (11**) + (10**) + (1*1*) + (1*0*) + (1**1) + (1**0),$$
$$\partial_3(*1**) = (11**) + (01**) + (*11*) + (*10*) + (*1*1) + (*1*0),$$
$$\partial_3(**1*) = (1*1*) + (0*1*) + (*11*) + (*01*) + (**11) + (**10),$$
$$\partial_3(***1) = (1**1) + (0**1) + (*1*1) + (*0*1) + (**11) + (**01).$$

To find the image ∂_3, we need to find all the linear combinations of these four maps. This is equivalent to finding the rank of the following matrix:

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Using SAGE, we were able to determine that the rank of this matrix is 3, so $\operatorname{im} \partial_3 \cong (\mathbb{Z}_2)^3$. Since $C_3 \cong (\mathbb{Z}_2)^4$, then by the rank-nullity theorem, $\ker \partial_3 \cong \mathbb{Z}_2$.

By a similar process, we can find the image and kernel of the remaining boundary maps. Using SAGE to find the rank of the matrices corresponding to each of the boundary maps, we found

$$\operatorname{im} \partial_2 \cong (\mathbb{Z}_2)^8,$$
$$\ker \partial_2 \cong (\mathbb{Z}_2)^4,$$
$$\operatorname{im} \partial_1 \cong (\mathbb{Z}_2)^7,$$
$$\ker \partial_1 \cong (\mathbb{Z}_2)^9.$$

Since $\partial_0 : C_0 \to 0$, we have that $\operatorname{im} \partial_0 \cong 0$ and $\ker \partial_0 \cong C_0 \cong (\mathbb{Z}_2)^8$. Also, since $\partial_5 : 0 \to C_4$, we have $\operatorname{im} \partial_5 \cong 0$. Now that we have the image and kernel of all the boundary maps, we have enough information to compute the homology groups.

The homology of the quotiented hypercube is

$$H_0 \cong \frac{\ker \partial_0}{\operatorname{im} \partial_1} \cong \frac{(\mathbb{Z}_2)^8}{(\mathbb{Z}_2)^7} \cong \mathbb{Z}_2,$$

$$H_1 \cong \frac{\ker \partial_1}{\operatorname{im} \partial_2} \cong \frac{(\mathbb{Z}_2)^9}{(\mathbb{Z}_2)^8} \cong \mathbb{Z}_2,$$

$$H_2 \cong \frac{\ker \partial_2}{\operatorname{im} \partial_3} \cong \frac{(\mathbb{Z}_2)^4}{(\mathbb{Z}_2)^3} \cong \mathbb{Z}_2,$$

$$H_3 \cong \frac{\ker \partial_3}{\operatorname{im} \partial_4} \cong \frac{\mathbb{Z}_2}{0} \cong \mathbb{Z}_2,$$

$$H_4 \cong \frac{\ker \partial_4}{\operatorname{im} \partial_5} \cong \frac{\mathbb{Z}_2}{0} \cong \mathbb{Z}_2\,.$$

This is also the homology of $\mathbb{R}P^4$, which is not surprising, since this Adinkra can also be thought of as a solid three dimensional cube, with edges connecting its antipodes.

4.2 Homology of an $N = 5$ Adinkra

Computing the homology of the 5-cube quotiented by the code $\{00000, 11110\}$ is similar to the previous example. However, in this case, the top cube has a non-zero boundary. We have that $\partial_5 : C_5 \to C_4$ is

$$\begin{aligned}\partial_5(*****) = {} & (1****) + (0****) \\ & +(*1***) + (*0***) \\ & +(**1**) + (**0**) \\ & +(***1*) + (***0*) \\ & +(****1) + (****0).\end{aligned}$$

Since $(1****) + 11110 = (0****)$ we have that $(1****) \sim (0****)$. Similarly, $(*1***) \sim (*0***)$ and $(**1**) \sim (**0**)$ and $(***1*) \sim (***0*)$.

However, $(****1) + 11110 = (****1)$ and $(****0) + 11110 = (****0)$. Therefore, $(****1)$ is not equivalent to $(****0)$, and thus $\partial_5(*****) = (****1) + (****0)$. Therefore, the rank of ∂_5 is 1, which means $\operatorname{im}\partial_5 \cong \mathbb{Z}_2$.

We can find the image of the rest of the boundary maps by the same method we used for the 4-cube: by finding the corresponding matrices and computing their ranks. We also find the chain groups in the same way as before. After identification, there are 16 0-cubes, 40 1-cubes, 40 2-cubes, 20 3-cubes, 6 4-cubes, and 1 5-cube. Therefore, the chain groups are

$$C_0 \cong (\mathbb{Z}_2)^{16},$$
$$C_1 \cong (\mathbb{Z}_2)^{40},$$
$$C_2 \cong (\mathbb{Z}_2)^{40},$$
$$C_3 \cong (\mathbb{Z}_2)^{20},$$
$$C_4 \cong (\mathbb{Z}_2)^{6},$$
$$C_5 \cong \mathbb{Z}_2\,.$$

The homology of the 5-cube quotiented by the code $\{00000, 11110\}$ is as follows:

$$H_0 \cong \frac{\ker\partial_0}{\operatorname{im}\partial_1} \cong \frac{(\mathbb{Z}_2)^{16}}{(\mathbb{Z}_2)^{15}} \cong \mathbb{Z}_2,$$

$$H_1 \cong \frac{\ker\partial_1}{\operatorname{im}\partial_2} \cong \frac{(\mathbb{Z}_2)^{25}}{(\mathbb{Z}_2)^{24}} \cong \mathbb{Z}_2,$$

$$H_2 \cong \frac{\ker\partial_2}{\operatorname{im}\partial_3} \cong \frac{(\mathbb{Z}_2)^{16}}{(\mathbb{Z}_2)^{15}} \cong \mathbb{Z}_2,$$

$$H_3 \cong \frac{\ker \partial_3}{\operatorname{im} \partial_4} \cong \frac{{\mathbb{Z}_2}^5}{{\mathbb{Z}_2}^4} \cong \mathbb{Z}_2,$$

$$H_4 \cong \frac{\ker \partial_4}{\operatorname{im} \partial_5} \cong \frac{{\mathbb{Z}_2}^2}{\mathbb{Z}_2} \cong \mathbb{Z}_2,$$

$$H_5 \cong \frac{\ker \partial_5}{\operatorname{im} \partial_6} \cong \frac{0}{0} \cong 0.$$

4.3 Homology of an $N = 6$ Adinkra

The homology of the $N = 6$ Adinkra, which is the 6-cube quotiented by $\{00000, 111100, 001111, 110011\}$ can be computed in the same way as the $N = 5$ and $N = 4$ cases. The chain groups are

$$C_0 \cong (\mathbb{Z}_2)^{16},$$

$$C_1 \cong (\mathbb{Z}_2)^{48},$$

$$C_2 \cong (\mathbb{Z}_2)^{60},$$

$$C_3 \cong (\mathbb{Z}_2)^{40},$$

$$C_4 \cong (\mathbb{Z}_2)^{18},$$

$$C_5 \cong (\mathbb{Z}_2)^{6},$$

$$C_6 \cong \mathbb{Z}_2\,.$$

The homology of the 6-cube quotiented by the code $\{00000, 111100, 001111, 110011\}$ is as follows:

$$H_0 \cong \frac{\ker \partial_0}{\operatorname{im} \partial_1} \cong \frac{(\mathbb{Z}_2)^{16}}{(\mathbb{Z}_2)^{15}} \cong \mathbb{Z}_2,$$

$$H_1 \cong \frac{\ker \partial_1}{\operatorname{im} \partial_2} \cong \frac{(\mathbb{Z}_2)^{33}}{(\mathbb{Z}_2)^{31}} \cong (\mathbb{Z}_2)^2,$$

$$H_2 \cong \frac{\ker \partial_2}{\operatorname{im} \partial_3} \cong \frac{(\mathbb{Z}_2)^{29}}{(\mathbb{Z}_2)^{26}} \cong (\mathbb{Z}_2)^3,$$

$$H_3 \cong \frac{\ker \partial_3}{\operatorname{im} \partial_4} \cong \frac{(\mathbb{Z}_2)^{14}}{(\mathbb{Z}_2)^{10}} \cong (\mathbb{Z}_2)^4,$$

$$H_4 \cong \frac{\ker \partial_4}{\operatorname{im} \partial_5} \cong \frac{(\mathbb{Z}_2)^{8}}{(\mathbb{Z}_2)^{3}} \cong (\mathbb{Z}_2)^5,$$

$$H_5 \cong \frac{\ker \partial_5}{\operatorname{im} \partial_6} \cong \frac{(\mathbb{Z}_2)^{3}}{0} \cong (\mathbb{Z}_2)^3,$$

$$H_6 \cong \frac{\ker \partial_5}{\operatorname{im} \partial_6} \cong \frac{\mathbb{Z}_2}{0} \cong \mathbb{Z}_2\,.$$

According to Poincaré duality, for a closed, oriented manifold, the kth homology group should be isomorphic to the $(N-k)$th cohomology group, that is $H_k \cong H^{N-k}$, where N is the top dimension [8]. Since we are computing $\mathbb{Z}_2$-homology, and $\mathbb{Z}_2$ is a field, then by the Universal Coefficient Theorem, the kth homology group is isomorphic to the kth cohomology group, or in other words $H_k \cong H^k$ [8]. Thus, Poincaré duality implies that for the $\mathbb{Z}_2$-homology of a closed, oriented manifold, $H_k \cong H_{N-k}$. The 4-cube, quotiented by $\{0000, 1100, 0011, 1111\}$ is an example of a manifold that obeys Poincaré duality, and the homology can be found in the next section. As we can see above, for the $N = 6$

Adinkra, Poincaré duality does not hold. This means that the Adinkra must not give rise to a manifold.

4.4 Other Homology Computations

When we realized Poincaré Duality does not always hold for the homology of Adinkras, to see where it started to break down, we tried to find the simplest code for which the homology of the quotient did not obey Poincaré duality.

The homology of the 1-cube, quotient $\{0, 1\}$ is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong \mathbb{Z}_2\,.$$

The homology of the 2-cube, quotient $\{00, 11\}$ is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong \mathbb{Z}_2,$$

$$H_2 \cong \mathbb{Z}_2\,.$$

The homology of the 3-cube, quotient $\{000, 111, 100, 011\}$ is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong (\mathbb{Z}_2)^2,$$

$$H_2 \cong (\mathbb{Z}_2)^2,$$

$$H_3 \cong \mathbb{Z}_2\,.$$

The homology of the 3-cube, quotient $\{000, 110, 011, 101\}$ is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong (\mathbb{Z}_2)^2,$$

$$H_2 \cong (\mathbb{Z}_2)^3,$$

$$H_3 \cong \mathbb{Z}_2\,.$$

This quotient is not Poincaré Dual.

The homology of the 4-cube, quotient $\{0000, 1100, 0011, 1111\}$ is

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^2,$$
$$H_2 \cong (\mathbb{Z}_2)^3,$$
$$H_3 \cong (\mathbb{Z}_2)^2,$$
$$H_4 \cong \mathbb{Z}_2\,.$$

The homology of the 4-cube, quotient $\{0000, 1000, 0111, 1111\}$ is

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^2,$$
$$H_2 \cong (\mathbb{Z}_2)^2,$$
$$H_3 \cong (\mathbb{Z}_2)^2,$$
$$H_4 \cong \mathbb{Z}_2\,.$$

The homology of the 4-cube, quotient $\{0000, 1100, 0010, 0001, 0011, 1101, 1110, 1111\}$ is

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^3,$$
$$H_2 \cong (\mathbb{Z}_2)^4,$$
$$H_3 \cong (\mathbb{Z}_2)^3,$$
$$H_4 \cong \mathbb{Z}_2\,.$$

The homology of the 4-cube, quotient $\{0000, 1110, 0011, 1101\}$ is

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^2,$$
$$H_2 \cong (\mathbb{Z}_2)^3,$$
$$H_3 \cong (\mathbb{Z}_2)^3,$$
$$H_4 \cong \mathbb{Z}_2\,.$$

This quotient is not Poincaré Dual.

The homology of the 4-cube, quotient $\{0000, 1100, 0110, 1010, 0001, 1101, 0111, 1011\}$ is

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^3,$$
$$H_2 \cong (\mathbb{Z}_2)^5,$$
$$H_3 \cong (\mathbb{Z}_2)^4,$$
$$H_4 \cong \mathbb{Z}_2\,.$$

This quotient is not Poincaré Dual.

The homology of the 5-cube, quotient $\{00000, 11100, 00111, 11011\}$

$$H_0 \cong \mathbb{Z}_2,$$
$$H_1 \cong (\mathbb{Z}_2)^2,$$
$$H_2 \cong (\mathbb{Z}_2)^3,$$
$$H_3 \cong (\mathbb{Z}_2)^4,$$
$$H_4 \cong (\mathbb{Z}_2)^3,$$
$$H_5 \cong \mathbb{Z}_2\,.$$

This quotient is not Poincaré Dual.

In summary, from the computations above, the codes that give quotients that do satisfy Poincaré Duality are $\{0, 1\}$, $\{00, 11\}$, $\{000, 111, 100, 011\}$, $\{0000, 1100, 0011, 1111\}$, and $\{0000, 1100, 0010, 0001, 0011, 1101, 1110, 1111\}$.

5
More About Generalized Adinkra Homology

5.1 Codes That Obey Poincaré Duality

The computations above seem to suggest that the generalized Adinkras that obey Poincaré duality are precisely those that have the element containing N 1's in the code. Thus, we made the following conjecture.

Conjecture 5.1.1. *Let X be an N-cube and C be a code. Then the $\mathbb{Z}_2$-homology of X/C will obey Poincaré duality if and only if C contains the codeword of N 1s.*

However, after writing SAGE code (see Appendix A) to compute the homology of additional generalized Adinkras, we found a counterexample.

Example 5.1.2. The set $\{01010101, 00110011, 00001111, 11111111\}$ is a basis for a doubly even code of length 8. The 8-cube, quotiented by the code generated by $\{01010101, 00110011, 00001111, 11111111\}$ is an $N = 8$ Adinkra. The homology of this

Adinkra is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong (\mathbb{Z}_2)^4,$$

$$H_2 \cong (\mathbb{Z}_2)^{10},$$

$$H_3 \cong (\mathbb{Z}_2)^{20},$$

$$H_4 \cong (\mathbb{Z}_2)^{35},$$

$$H_5 \cong (\mathbb{Z}_2)^{42},$$

$$H_6 \cong (\mathbb{Z}_2)^{28},$$

$$H_7 \cong (\mathbb{Z}_2)^{8},$$

$$H_8 \cong (\mathbb{Z}_2).$$

Thus, since the code generated by $\{01010101, 00110011, 00001111, 11111111\}$ contains the codeword 11111111, this Adinkra is a counterexample to Conjecture 5.1.1. $\diamondsuit$

5.2 Code Weights and Homology

We know that the homology of these quotients of cubes by codes is an invariant under permutation of the code. That is, if two codes C_1 and C_2 have length N and are permutation equivalent, then the homology groups of the N-cube quotiented by C_1 are the same as the homology groups of the N-cube quotiented by C_2. For example, the homology of the 4-cube quotiented by $\{0000, 1000, 0111, 1111\}$ is the same as the homology of the 4-cube quotiented by $\{0000, 1110, 0001, 1111\}$.

Definition 5.2.1. Let C be a binary linear code of length N. The *weight distribution* of a code specifies the number of codewords in C of each weight from 1 to N. We can denote the weight distribution of C by $A_0(C), A_1(C), ..., A_N(C)$ where $A_i(C)$ is the number of codewords of weight i[9]. $\triangle$

Example 5.2.2. Let $C = \{0000, 1000, 0111, 1111\}$. Then the weight distribution is $A_0 = 1, A_1 = 1, A_2 = 0, A_3 = 1, A_4 = 1$. ◊

Two codes have the same weight distribution if they have the same number of codewords of each weight. We noticed that many cubes quotiented by codes that have the same weight distribution also have the same homology, which lead us to the following conjecture.

Conjecture 5.2.3. *Suppose X is an N-cube and C_1 and C_2 are two codes with the name weight distribution. Then $H_k(X/C_1) \cong H_k(X/C_2)$ for all k.*

It turns out this conjecture is false. Since many codes that have the same weight distribution also are permutation equivalent, to test this conjecture, we computed the homologies of the 6-cube quotiented by the code generated by $\{110000, 001100, 000011\}$ and the 6-cube quotiented by $\{110000, 101000, 111111\}$. Both of these codes have weight distribution $A_0 = 1, A_1 = 0, A_2 = 3, A_3 = 0, A_4 = 3, A_5 = 0, A_6 = 1$ but they are not permutation equivalent [9].

Example 5.2.4. The homology of the 6-cube quotiented by the code generated by $\{110000, 001100, 000011\}$ is

$$\begin{aligned}
H_0 &\cong \mathbb{Z}_2, \\
H_1 &\cong (\mathbb{Z}_2)^3, \\
H_2 &\cong (\mathbb{Z}_2)^6, \\
H_3 &\cong (\mathbb{Z}_2)^7, \\
H_4 &\cong (\mathbb{Z}_2)^6, \\
H_5 &\cong (\mathbb{Z}_2)^3, \\
H_6 &\cong \mathbb{Z}_2\,.
\end{aligned}$$

However, the homology of the 6-cube quotiented by $\{110000, 101000, 111111\}$ is

$$
\begin{aligned}
H_0 &\cong \mathbb{Z}_2,\\
H_1 &\cong (\mathbb{Z}_2)^3,\\
H_2 &\cong (\mathbb{Z}_2)^6,\\
H_3 &\cong (\mathbb{Z}_2)^7,\\
H_4 &\cong (\mathbb{Z}_2)^7,\\
H_5 &\cong (\mathbb{Z}_2)^4,\\
H_6 &\cong \mathbb{Z}_2\,.
\end{aligned}
$$

Therefore these two codes are a counterexample to Conjecture 5.2.3. Note also that the homology of the 6-cube quotiented by $\{110000, 101000, 111111\}$ is another counterexample to Conjecture 5.1.1. $\Diamond$

5.3 Homology of N-cube Quotiented by 2^N-code

The homology of the cube quotiented by the maximal code, which is the code containing all possible codewords of length N is actually the same as the homology of a recognizable space: the N-torus.

Theorem 5.3.1. *For the N-cube, quotiented by the code containing all 2^N codewords, $H_k \cong (\mathbb{Z}_2)^{\binom{N}{k}}$. In other words, the kth homology group is isomorphic to the kth homology group of the N-torus.*

Proof. Let C be the code with all 2^N possible code words of length N. Suppose A and B are two k-cubes of the same type. Then, there exists $c_n \in C$ such that $A + c_n = B$. Therefore, these two k-cubes are equivalent, after quotienting the N-cube by C. We have that the kth chain group is generated by the k-cubes. Since each k-cube has k stars, then there will be $\binom{N}{k}$ different possible types of k-cubes, and thus there are $\binom{N}{k}$ non-equivalent k-cubes. Thus, $C_k \cong (\mathbb{Z}_2)^{\binom{N}{k}}$.

Since the boundary of any $(k+1)$-cube contains exactly two k-cubes of a given type, these terms will cancel (mod 2), so $\partial_{k+1} = 0$. Since $\operatorname{im} \partial_{k+1} \cong 0$, we have $H_k \cong C_k \cong (\mathbb{Z}_2)^{\binom{N}{k}}$, which is the same as the kth homology group of the N-torus [10]. □

5.4 The First Homology Group

After computing the homology of many N-cubes quotiented by codes, we noticed that the first homology group was always isomorphic to the code. In Reference [2], Theorem 4.1 states that for the quotient of an N-cube by a binary linear $[N, k]$ block code C, if the minimal weight of a non-zero codeword in C is greater than or equal to 4, then $H_1 = (\mathbb{Z}_2)^k$. We will generalize this theorem to include codes with codewords of any weight but first we will need the following definitions and lemmas.

Definition 5.4.1. A graph is *edge-N-partite* if for each vertex v, there is a unique edge of each type incident with v. △

Lemma 5.4.2. *Let X be a N-cube and C be a binary linear code of length N and dimension k. Then X/C is edge-N-partite.*

Proof. Any N-cube is edge-N-partite. Since quotienting by C does not change the type of any edge, then X/C is also edge-N-partite. □

Lemma 5.4.3. *Let X be a N-cube and C be a binary linear code of length N and dimension k. Fix a vertex, $v_0 \in X/C$. Any path of edges starting at v_0 in X/C can be uniquely specified by a sequence of edge types.*

Proof. Since X/C is edge-N-partite then a path in X/C beginning at v_0 can be specified by the edge types that are followed, in order. □

Lemma 5.4.4. *Let X be a N-cube and C be a binary linear code of length N and dimension k. Fix a vertex, $v_0 \in X/C$. For any path of edges f, beginning at v_0 in X/C, there exists a unique lift of f to a path of edges $\tilde{f}$ beginning at a corresponding vertex $\tilde{v_0}$ in X.*

Proof. By the previous lemma, we can lift a path f beginning at v_0 in X/C to a path $\tilde{f}$ in X by starting at a corresponding vertex $\tilde{v_0}$ (where $\tilde{v_0}$ is a vertex in X that gets mapped to v_0 by C) and following the same sequence of edge types. This path is unique because X is edge-N-partite. □

Lemma 5.4.5. *Suppose we have a cycle beginning at v_0 with a sequence of edge types $\{t_1, t_2, ..., t_n\}$. This cycle is homologous to a cycle beginning at v_0 with the sequence of edge types $\{t_1, t_2, ..., t_{i+1}, t_i, ..., t_n\}$ for any $i \in \{1, ..., n\}$. In other words, switching any two adjacent edge types in a cycle gives us a homologous cycle.*

Proof. In the cube, starting at any vertex v and following $t_i, t_{i+1}, t_i, t_{i+1}$ will give us the boundary of a face. Since under the quotient, face get mapped to faces, $t_i, t_{i+1}, t_i, t_{i+1}$ is the boundary of a face in the quotient. Then $\{t_1, t_2, ..., t_n\}$ differs from $\{t_1, t_2, ..., t_{i+1}, t_i, ..., t_n\}$ only by a face. Since H_1 is the cycles of edges modulo boundaries of 2-cells, then these two cycles are homologous. □

Corollary 5.4.6. *Let l be a cycle beginning at v_0 with a sequence of edge types $\{t_1, t_2, ..., t_n\}$. Any path beginning at v_0 that is a permutation of these edge types is a cycle that is homologous to l.*

Proof. Any permutation of $\{t_1, t_2, ..., t_n\}$ can be obtained by switching two edges at a time. □

Lemma 5.4.7. *Let l be a cycle beginning at v_0 with a sequence of edge types $\{t_1, t_1, t_2, ..., t_n\}$ and l' be a cycle beginning at v_0 with a sequence of edge types $\{t_2, ..., t_n\}$. Then $[l] = [l']$.*

Proof. We are working mod 2, which means in C_1 we have that $\{t_1, t_1, t_2, ..., t_n\}$ and $\{t_2, ..., t_n\}$ are the equivalent, since $2t_1 = 0t_1 \pmod 2$. □

Lemma 5.4.8. *Any cycle of edges in X/C is homologous to a cycle based at v_0.*

Proof. Suppose l is a cycle in X/C. If l passes through v_0 then we can choose v_0 as our basepoint, since homology does not depend on choice of basepoint. Suppose l is cycle based at v and does not go through v_0. There exists a path of edges from v to v_0 that we can specify by a sequence of edge types $\{t_1, t_2, ..., t_n\}$. Then let l' be a cycle that starts at v and follows $\{t_1, t_2, ..., t_n, t_1, t_2, ..., t_n\}$. Then $l' + l$ is a cycle that starts at v and passes through v_0. Since by Lemma 5.4.7 l' is homologous to the trivial cycle, then $l' + l$ is homologous to l. □

Now, we must specify a map from $w \in C$ to elements of $H_1(X/C)$. Fix a vertex $v_0 \in X/C$. Recall Definition 3.2.1 of edge type. Then let P_w be the path obtained by starting at v_0 and following the edge types specified by the locations of the 1s in the codeword, w. For instance the if $w = (10110)$, we would start at v_0 and follow an edge of type 1, then an edge of type 3, and then an edge of type 4.

Theorem 5.4.9. *Fix a basepoint, v_0 in X/C. Define a function $\phi : C \longrightarrow H_1(X/C)$ where $\phi(w) = [P_w]$. Then ϕ is an isomorphism.*

Proof. First, we will show that ϕ is a homomorphism. Let l be the cycle that begins at v_0 and follows the sequence of edge types for P_{w_1} and then for P_{w_2}. Then $l \in [P_{w_1}] + [P_{w_2}]$. We can delete any edge types that occur twice, which will be precisely those edges types that correspond to the location of a 1 in both codewords. Thus, $[P_{w_1}] + [P_{w_2}] = [P_{w_1+w_2}]$.

To show ϕ is injective, we will show ϕ has a trivial kernel. Suppose $\phi(w)$ is homologous to the constant loop. Then any cycle must only differ from the constant loop by a boundary, or a sum of boundaries, of a 2-cell. Since boundaries of 2-cells always have 2 edges of a given type, we can cancel these edges. Therefore, w is the codeword of all $0s$.

Finally, we must show ϕ is surjective. Suppose $[l] \in H_1(X/C)$. Since any cycle is homologous to a cycle based at v_0 then for any homology class in $H_1(X/C)$, choose a representative of $[l]$ that is based at v_0. Deleting any duplicate edges types gives us a cycle $l' \in [l]$ that has at most one of any edge type. Let K be the set of edge types that are in l'. Then let w be the code word that has a 1 in entry k for each $k \in K$, and a 0 in every other entry. Therefore, $\phi(w) = [l'] = [l]$, so ϕ is surjective. $\square$

6

Other Methods and Topics for Future Study

6.1 Equivariant Homology

The fact that the homology of some of the Adinkras did not obey Poincaré duality suggests that these Adinkras are not manifolds. Instead, they might be orbifolds, which are manifolds quotiented by a group where the group action is not free.

Definition 6.1.1. Let G be a group and X be a set. Suppose $g, h \in G$. If $gx = hx$ if and only if $g = h$ for all $x \in X$ then we say that the action of G on X is *free*. $\triangle$

In our case, the group is the code, and it acts of the sets of vertices, edges, faces, cube, etc by identification via addition of codewords.

Example 6.1.2. Let $C = \{000, 110, 011, 101\}$. Consider the set of faces of the 3-cube $\{(1{*}{*}), (0{*}{*}), ({*}1{*}), ({*}0{*}), ({*}{*}1), ({*}{*}0)\}$. We have that $(1{*}{*}) + 101 = (0{*}{*})$ and $(1{*}{*}) + 110 = (0{*}{*})$. Therefore C does not act freely on the faces of the 3-cube. $\diamondsuit$

We hoped to fix this problem by computing the equivariant homology of these Adinkras.

Definition 6.1.3. For any group G we can define a space EG such that EG is contractible and has a free G action. $\triangle$

By crossing a space that does not have a free G-action with EG, we can "free up" the action without changing the topology. Then we can compute the equivariant homology, which is defined as follows.

Definition 6.1.4. The nth equivariant homology group is defined as follows:

$$H_n^G(X) = H_n((X \times EG)/G).$$

$\triangle$

First, we tried to compute the $\mathbb{Z}_2$-equivariant homology of the 1-cube. To do this, we cross by a space $E\,\mathbb{Z}_2$ and mod out by a $\mathbb{Z}_2$-action. The space $E\,\mathbb{Z}_2$, which has a free $\mathbb{Z}_2$-action and is contractible, has the chain complex of S^∞. The space S^∞ is the limit of S^n as $n \longrightarrow \infty$, via inclusion. We have that S^∞ is contractible because it is the union of all S^n and each S^n is contractible in S^{n+1}. The cells of $E\,\mathbb{Z}_2$ are $\{c_n^+, c_n^- | n \in \mathbb{Z}_{\geq 0}\}$, where $c_n^\pm$ correspond to upper and lower n-hemispheres. The boundary of the cells are $\partial c_n^\pm = c_{n-1}^+ + c_{n-1}^-$. The 1-cube that we are crossing with $E\,\mathbb{Z}_2$ has two 0-cells, a and b, and one 1-cell, e. The boundary of e is $a + b$.

If we have two spaces X and Y, then an n-cell of $X \times Y$ is an ordered pair (x_p, y_q), where x_p is a p-cell of X and y_q is a q-cell of Y and $p + q = n$. The boundary of this n-cell is

$$\partial(x_p, y_q) = (\partial x_p, y_q) + (x_p, \partial y_q).$$

Thus, when we cross the 1-cube with $E\,\mathbb{Z}_2$, the 0-cells are $\{(a, c_0^+), (a, c_0^-), (b, c_0^+), (b, c_0^-)\}$, the 1-cells are $\{(e, c_0^+), (e, c_0^-), (a, c_1^+), (a, c_1^-), (b, c_1^+), (b, c_1^-)\}$, the 2-cells are $\{(e, c_1^+), (e, c_1^-), (a, c_2^+),$ $(a, c_2^-), (b, c_2^+), (b, c_2^-)\}$ and so on. Some of these cells get identified by the $\mathbb{Z}_2$-action. The chain groups and the boundary maps are computed below.

The 0th chain group is

$$C_0 = \text{span}\{(a, c_0^+), (a, c_0^-), (b, c_0^+), (b, c_0^-)\}.$$

Under the $\mathbb{Z}_2$ action, (a, c_0^+) gets identified with (b, c_0^-) and (a, c_0^-) gets identified with (b, c_0^+). Therefore

$$\frac{C_0}{\mathbb{Z}_2} = \text{span}\{[(a, c_0^+)], [(a, c_0^-)]\} \cong (\mathbb{Z}_2)^2.$$

Similarly, since

$$C_1 = \text{span}\{(e, c_0^+), (e, c_0^-), (a, c_1^+), (a, c_1^-), (b, c_1^+), (b, c_1^-)\},$$

then

$$\frac{C_1}{\mathbb{Z}_2} = \text{span}\{[(e, c_0^+)], [(a, c_1^+)], [(a, c_1^-)]\} \cong (\mathbb{Z}_2)^3.$$

Next we have

$$C_2 = \text{span}\{(e, c_1^+), (e, c_1^-), (a, c_2^+), (a, c_2^-), (b, c_2^+), (b, c_2^-)\}.$$

Which, after quotienting by $\mathbb{Z}_2$ gives us

$$\frac{C_2}{\mathbb{Z}_2} = \text{span}\{[(e, c_1^+)], [(a, c_2^+)], [(a, c_2^-)]\} \cong (\mathbb{Z}_2)^3.$$

In general, we have

$$\frac{C_n}{\mathbb{Z}_2} = \text{span}\{[(e, c_{n-1}^+)], [(a, c_n^+)], [(a, c_n^-)]\} \cong (\mathbb{Z}_2)^3$$

for all $n > 0$. We now consider $\partial_1 : \frac{C_1}{\mathbb{Z}_2} \to \frac{C_0}{\mathbb{Z}_2}$. We have

$$\partial_1(e, c_0^+) = (a, c_0^+) + (b, c_0^+) = (a, c_0^+) + (a, c_0^-),$$

$$\partial_1(a, c_1^+) = (a, c_0^+) + (a, c_0^-),$$

$$\partial_1(a, c_1^-) = (a, c_0^+) + (a, c_0^-).$$

Since $\partial_1(e, c_0^+) = \partial_1(a, c_1^+) = \partial_1(a, c_1^-)$ then $\operatorname{im} \partial_1 = \mathbb{Z}_2$. Then, consider $\partial_2 : \frac{C_2}{\mathbb{Z}_2} \to \frac{C_1}{\mathbb{Z}_2}$. We have

$$\partial_2(e, c_1^+) = (a, c_1^+) + (b, c_1^+) + (e, c_0^+) + (e, c_0^-) = (a, c_1^+) + (a, c_1^-),$$

$$\partial_2(a, c_2^+) = (a, c_1^+) + (a, c_1^-),$$

$$\partial_2(a, c_2^-) = (a, c_1^+) + (a, c_1^-).$$

Since $\partial_1(e, c_1^+) = \partial_1(a, c_2^+) = \partial_1(a, c_2^-)$ then $\operatorname{im} \partial_1 = \mathbb{Z}_2$. In general, we have $\partial_n : \frac{C_n}{\mathbb{Z}_2} \to \frac{C_{n-1}}{\mathbb{Z}_2}$ such that

$$\partial_n(e, c_{n-1}^+) = (a, c_{n-1}^+) + (a, c_{n-1}^-),$$

$$\partial_n(a, c_n^+) = (a, c_{n-1}^+) + (a, c_{n-1}^-),$$

$$\partial_n(a, c_n^-) = (a, c_{n-1}^+) + (a, c_{n-1}^-).$$

Since $\partial_n(e, c_{n-1}^+) = \partial_n(a, c_n^+) = \partial_n(a, c_n^-)$ then $\operatorname{im} \partial_n = \mathbb{Z}_2$, for all $n > 0$. Therefore, we have

$$H_0 = \frac{\ker \partial_0}{\operatorname{im} \partial_1} = \frac{(\mathbb{Z}_2)^2}{\mathbb{Z}_2} = \mathbb{Z}_2,$$
$$H_n = \frac{\ker \partial_n}{\operatorname{im} \partial_{n-1}} = \frac{(\mathbb{Z}_2)^2}{\mathbb{Z}_2} = \mathbb{Z}_2$$

for all $n > 0$.

6.2 Cross Polytopes

The reason we get orbifolds for some of these quotients is because there are certain points that are fixed under the group action. Often these points occur somewhere in the middle of an edge or a face, so we decided instead to try to compute the homology of the cross polytope, which is the dual of the N-cube.

Definition 6.2.1. Suppose we have an N-cube, $[0,1]^N$. Let P_0 be the set of points consisting of all possible permutations of $(1/2, 1/2, ..., 0)$ and P_1 be the set of points consisting of all possible permutations of $(1/2, 1/2, ..., 1)$. Then the corresponding *cross polytope* that is dual to this N-cube is the N-dimensional regular polytope that is the convex hull of $P_0 \cup P_1$ [1]. △

The cross polytope is dual to the N-cube in that it has a k-cell for every $(N-k-1)$-cell of the N-cube. When we are comparing the N-cube to its cross polytope, we will throw out the top dimensional cube. The cross polytope will have its fixed points on vertices, rather than in the middle of an edge or face, so we thought it might be helpful to consider the homology of the quotient of this space instead. Since the cross polytope is dual to the original polytope, the homology of the cross polytope should be the cohomology of the original polytope. Cohomology is a way of associating a series of abelian groups to a topological space that is similar to homology. In fact, the coboundary operators in the cohomology are just the transpose of the boundary operators from the homology, and the cochain groups are isomorphic to the chain groups. It makes sense that the homology of cross polytope should be the same as the cohomology of the N-cube, due to the duality of the cells described above. Since we are working with $\mathbb{Z}_2$ as our coefficient group, the Universal Coefficient Theorem implies that their homologies should be the same. As an example, we computed the homology of the cross polytope of the cube, i.e. the octahedron, quotiented by the code $\{000, 110, 011, 101\}$. The homology is

$$H_0 \cong \mathbb{Z}_2,$$

$$H_1 \cong 0,$$

$$H_2 \cong \mathbb{Z}_2\,.$$

There is no H_3, since there is no 3 dimensional cell in the octahedron. This homology is not the same as the homology of the cube, quotiented by $\{000, 110, 011, 101\}$, which as we listed earlier is

$$H_0 \cong \mathbb{Z}_2,$$

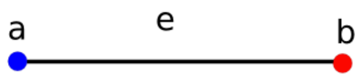
$$H_1 \cong (\mathbb{Z}_2)^2,$$

$$H_2 \cong (\mathbb{Z}_2)^3.$$

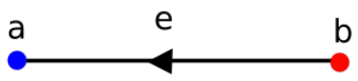
However, the homology of the octahedron quotiented by $\{000, 110, 011, 101\}$ is the same as the homology of the 2-sphere. We tried to picture the quotient, in order to see why. It has 3 vertices, 3 edges, and 2 faces. Therefore, the quotient looks like a triangular pillow, or two triangles sewn together, which is like a sphere but has 3 orbifold points. This suggests that we might need to use some type of orbifold homology to study these spaces further. Regardless, the discrepancy between the homology of the quotient of the N-sphere and the homology of the quotient of the cross polytope is interesting and merits further study.

6.3 Integer Homology

So far, we have computed only the $\mathbb{Z}_2$-homology of generalized Adinkras. However, we can also compute the homology using another coefficient group, such as the integers. Computing homology over the integers is more difficult, because we now have to take signs into account. That means N-cubes must have an orientation, and we must keep track of the orientation when computing the boundary maps. For instance, consider an edge, connected by two vertices.



Before, when we were computing $\mathbb{Z}_2$-homology, the boundary of e would have been $a+b$. However, when computing integer homology, we must assign an orientation to this edge.



Now the boundary of e is $a - b$. We can do the same with the faces, cubes, etc. We computed the integer homology of the $N = 4$ Adinkra, which is the 4-cube quotiented by $\{0000, 1111\}$. As an example, the boundary of the 4-cube is

$$\begin{aligned}\partial_4(****) =& (1***) - (0***)\\ &-(*1**) + (*0**)\\ &+(**1*) - (**0*)\\ &-(***1) + (***0).\end{aligned}$$

This time, when we are identifying n-cube by addition of a codeword, adding a 1 to a $*$ flips the orientation. Therefore, after identification, the boundary becomes

$$\begin{aligned}\partial_4(****) =&(1***)+(1***)\\ &-(*1**)-(*1**)\\ &+(**1*)+(**1*)\\ &-(***1)-(***1)\\ &=2\,[(1***)-(*1**)+(**1*)-(***1)].\end{aligned}$$

Therefore $\operatorname{im}\partial_4 \cong 2\,\mathbb{Z}$. Since we are working over the integers, the image of the boundary map can now be a subgroup of $\mathbb{Z}^k$, where k is the rank of the boundary map. Thus, when we form the boundary matrix, we need to find the Smith Normal Form of the matrix, which is a diagonal form that can be obtained by multiplying on the left and right by square integer matrices with determinant ± 1, in addition to the rank and nullity, to be able to find $\operatorname{im}\partial_n$ [7]. After doing these computations in SAGE, we find that the integer homology of the $N = 4$ Adinkra is

$$H_0 \cong \frac{\ker\partial_0}{\operatorname{im}\partial_1} \cong \frac{(\mathbb{Z})^8}{(\mathbb{Z})^7} \cong \mathbb{Z},$$

$$H_1 \cong \frac{\ker\partial_2}{\operatorname{im}\partial_3} \cong \frac{(\mathbb{Z})^9}{(\mathbb{Z})^8 \oplus 2\,\mathbb{Z}} \cong \mathbb{Z}_2,$$

$$H_2 \cong \frac{\ker\partial_1}{\operatorname{im}\partial_2} \cong \frac{(\mathbb{Z})^3}{(\mathbb{Z})^3} \cong 0,$$

$$H_3 \cong \frac{\ker\partial_3}{\operatorname{im}\partial_4} \cong \frac{\mathbb{Z}}{2\,\mathbb{Z}} \cong \mathbb{Z}_2,$$

$$H_4 \cong \frac{\ker\partial_4}{\operatorname{im}\partial_5} \cong \frac{0}{0} \cong 0.$$

Again, this is the same as the integer homology of $\mathbb{R}P^4$, as we expected. It would be interesting to compute the integer homology of some of the other spaces we considered when we were finding $\mathbb{Z}_2$-homology, to see we can find any additional information about these spaces.

6.4 Additional Future Research

As mentioned earlier, when we were quotienting cubes by codes, we were really taking only the quotient at the level of the vertices. It is also possible to consider the topological quotient space, where the code acts linearly on the cube as a topological space, and to compute the homology of those spaces. In Reference [11], Ed Swarz classifies quotients of spheres by groups of the form $(\mathbb{Z}_p)^N$, where p is a prime. It would be interesting to consider these spaces, their homology, and their relation to Adinkras.

Bibliography

[1] H. S. M. Coxeter, *Regular Polytops*, Dover Publications, New York, 1973.

[2] C.F. Doran, M.G. Faux, S.J. Gates Jr., T. Hübsch, K.M. Iga, and G.D. Landweber, *An Application of Cubical Cohomology to Adinkras and Supersymmetry Representations.*

[3] ———, *On graph-theoretic identifications of Adinkras, supersymmetry representations and superfields*, International Journal of Modern Physics A **22** (2007).

[4] C.F. Doran, M.G. Faux, S.J. Gates Jr., T. Hübsch, K.M. Iga, G.D. Landweber, and R.L. Miller, *Topology Types of Adinkras and the Corresponding Representations of N-Extended Supersymmetry.*

[5] ———, *Adinkras for Clifford Algebras, and Worldline Supermultiplets.*

[6] M.G. Faux and S.J. Gates Jr., *Adinkras: A graphical technology for supersymmetric representation theory*, Physical Review D **71** (2005).

[7] John Guillaume Dumas, Frank Heckenbach, B. David Saunders, and Volkmar Welker, *Algebra, Geometry, and Software Systems*, Springer, 2003.

[8] Allen Hatcher, *Algebraic Topology*, Cambridge University Press, 2002.

[9] William Cary Huffman and Vera Pless, *Fundamentals for Error Correcting Codes*, Cambridge University Press, 2003.

[10] V. V. Prasolov, *Elements of Homology Theory*, American Mathematical Society, 2007.

[11] Ed Swarz, *Matroids and Quotients of Spheres*, Mathematische Zeitschrift **241** (2002).

[12] William P. Thurston, *The Geometry and Topology of Three-Manifolds* (2002).

Appendix A

```
#Takes an n-cube (a list of *s, 0s, and 1s) and outputs a list
of n-1 cubes, by taking each * in turn and replacing it with a 1.
def boundary1(cube):
    a = []
    i = 0
    while i < len(cube):
        if cube[i] == '*':
             y = []
             y = y + cube
             y[i] = '1'
             a = a + [y]
        i = i + 1
    return a

#Takes an n-cube (a list of *s, 0s, and 1s) and outputs a list
of n-1 cubes, by taking each * in turn and replacing it with a
0.
def boundary0(cube):
    a = []
    i = 0
    while i < len(cube):
        if cube[i] == '*':
             y = []
             y = y + cube
             y[i] = '0'
             a = a + [y]
        i = i + 1
    return a


#Function for adding '1', '0', and '*', mod 2.  Takes two
strings, containing either *, 0, or 1 and adds them, so that 1+1
= 0, 1+0 = 0+1 = 1 and *+1 = 1+* = *+0 = 0+* = *.  Returns a
string containing the sum.
def mod2(a,b):
    if a == '*' or b == '*':
        return '*'
    elif a == '0':
        return b
    elif b == '0':
        return a
    else:
        return '0'

#Takes a code (which is a list of codewords, i.e. a list of
strings of 0s and 1s) and a list of n-cubes (which are lists
'0', '1', and '*'), and creates a list of equivalence classes.
An equivalence class is a list of n-cubes that can be mapped to
each other by mod 2 addition of a codeword.  This may output
duplicates of equivalence classes, and n-cubes within the
equivalence classes.
def codeAdd(code, cubes):
    output = []
    i = 0
    while i < len(cubes):
        outputi = []
        j = 0
        while j < len(code):
            outputj = []
            k = 0
            while k < len(code[j]):
                outputj = outputj + [mod2(code[j][k], cubes[i]
[k])]
                k = k + 1
            outputi = outputi + [outputj]
            j = j+1
        output = output + [outputi]
        i = i+1
    return output

#Takes a list of equivalence classes that contains duplicate
equivalence classes, and returns a list of equivalence classes
with no duplicates.
def equivclass(classes):
    newlist = []
    i = 0
    while i < len(classes):
        j = i + 1
        number = 0
        while j < len(classes):
            k = 0
            while k < len(classes[j]):
                if classes[i][0] == classes[j][k]:
                    number = number + 1
                k = k + 1
            j = j + 1
        if number == 0:
            newlist = newlist + [classes[i]]
        i = i + 1
    return newlist

#Takes a list of equivalence classes that contains duplicate
elements within the equivalence classes, and returns a list of
equivalence classes with no duplicates within the equivalence
classes.
def equivclass2(classes):
    classes = equivclass(classes)
    newlist = []
    i = 0
    while i < len(classes):
        nlist = []
        j = 0
        while j < len(classes[i])-1:
            k = j + 1
            number = 0
            while k< len(classes[i]):
                if classes[i][j] == classes[i][k]:
                    number = number + 1
                k = k + 1
            if number == 0:
                nlist = nlist + [classes[i][j]]
            j = j + 1
        nlist = nlist + [classes[i][len(classes[i])-1]]
        newlist = newlist + [nlist]
        i = i + 1
    return newlist

#Takes a list of equivalence class and returns a list of one
element of each equivalence class.
def chaingp(eqclasses):
    cg = []
    i = 0
    while i < len(eqclasses):
        cg = cg + [eqclasses[i][0]]
        i = i + 1
    return cg

#Takes an an equivalence class and returns a list of 0s, one for
each equivalence class.  This is later used to make a vector for
the boundary matrix.
def nlist(eqclasses):
    newvec = []
    i = 0
    while i < len(eqclasses):
        newvec = newvec + [0]
        i = i + 1
    return newvec

#Takes a list of equivalence classes, the boundary of n-cube,
and a list of 0s and makes a vector that represents boundary map
of the n-cube.
def vectorop(eqclasses, boundary, newvec):
    j = 0
    while j < len(boundary):
        k = 0
        while k < len(eqclasses):
            l = 0
            while l < len(eqclasses[k]):
                if boundary[j] == eqclasses[k][l]:
                    newvec[k] = (newvec[k] + 1) % 2
                l = l + 1
            k = k + 1
        j = j + 1
    return newvec

#Takes a list of equivalence classes and the list of elements on
the boundaries of all n-cubes.  Creates vectors for each
boundary and then combines all the vectors into a matrix and
returns this boundary operator matrix.
def matrixop(eqclasses, boundarylist):
    i = 0
    newmat = []
    while i < len(boundarylist):
        newmat = newmat + [vectorop(eqclasses, boundarylist[i],
nlist(eqclasses))]
        i = i + 1
    return newmat

#Takes a list of n-cubes and makes a list of the boundaries of
the n-cubes.  (The boundary of an n-cube is a list of the n-1
cubes that are on its boundary).  Returns the list of the
boundaries.
def bdoflist(cubes):
    i = 0
    bd = []
    while i < len(cubes):
        bd = bd + [boundary1(cubes[i]) + boundary0(cubes[i])]
        i = i + 1
    return bd

#Takes a list of n-cubes and returns a list of all the possible
n-1 cubes.
def bigbdoflist(cubes):
    chaingrouplist
    i = 0
    bd = []
    while i < len(cubes):
        bd = bd + boundary1(cubes[i]) + boundary0(cubes[i])
        i = i + 1
    return bd

#Takes a code, the highest dimensional cube, and the dimension
of the cube and returns a list of all of the boundary operator
matrices.
def bdryops(code, topcube, n):
    i = n
    bdoplist = [0]
    while i > 0:
        ec = equivclass2(codeAdd(code, bigbdoflist(topcube)))
        matrixbd = matrixop(ec, bdoflist(topcube))
        bdoplist = bdoplist + [matrixbd]
        topcube = chaingp(ec)
        i = i - 1
    return bdoplist

#Takes a list of boundary operators, computes the output of
each, and returns a list of the ranks, from the highest
dimensional boundary operator to the lowest.
def ranks(bdoplist):
    i = 0
    ranklist = []
    while i < len(bdoplist):
        ranklist = ranklist + [Matrix(Integers(2),
bdoplist[i]).rank()]
        i = i + 1
    ranklist = ranklist + [0]
    return ranklist

#Takes a code, the highest dimensional cube, and the dimension
of the cube and returns a list of the dimension of the chain
groups, from the highest dimensional chains to the lowest.
```

```
def chaingrouplist(code, topcube, n):
    i = n
    chaingplist = [1]
    while i > 0:
        ec = equivclass2(codeAdd(code, bigbdoflist(topcube)))
        #print ec
        matrixbd = matrixop(ec, bdoflist(topcube))
        chaingplist = chaingplist + [len(chaingp(ec))]
        topcube = chaingp(ec)
        i = i - 1
    return chaingplist

#Takes the list of ranks, and the list of chain groups and uses
the Rank-Nullity theorem to compute the dimension of the
homology groups.  Returns a list, from the the nth homology
group, down to the 0th homology group.
def homology(ranklist, chaingplist):
    homlist = []
    i = 0
    while i < len(chaingplist):
        homlist = homlist +
[chaingplist[i]-ranklist[i]-ranklist[i+1]]
        i = i + 1
    return homlist
```

```
#Example of the homology of 6-cube, quotiented by the code
['000000', '101000', '111111', '100111', '110000', '011000',
'001111',
'010111'].
topcube = [['*', '*', '*', '*', '*', '*']]
code = ['000000', '101000', '111111', '100111', '110000',
'011000', '001111',
'010111']
n = 6

print chaingrouplist(code, topcube, n)
print ranks(bdryops(code, topcube, n))
homology(ranks(bdryops(code, topcube, n)), chaingrouplist(code,
topcube, n))
```

```
    [1, 6, 18, 32, 36, 24, 8]
    [0, 0, 2, 9, 16, 14, 7, 0]
    [1, 4, 7, 7, 6, 3, 1]
```

```
#Takes in a generating set and returns the code that it
generates.  This will have some duplicate codewords.
```

```
def genCode1(genset):
    m = len(genset)/2
    l = 0
    while l < m+1:
        n = len(genset)
        i = 0
        while i < (n-1):
            j = i + 1
            while j < n:
                k = 0
                cdwd = ''
                while k < len(genset[i]):
                    cdwd = cdwd + mod2(genset[i][k],genset[j][k])
                    k = k+1
                genset = genset + [cdwd]
                j = j+1
            i = i + 1
        l = l + 1
    return genset

#Takes in a generating set, and uses genCode1 to generate a code
with duplicates.  Then deletes the duplicate codewords and
returns the code that is generated from genset.
def genCode2(genset):
    i = 0
    newcode = []
    while i < len(genset)-1:
        j = i + 1
        number = 0
        while j < len(genset):
            if genset[i] == genset[j]:
                number = number + 1
            j = j + 1
        if number == 0:
            newcode = newcode + [genset[i]]
        i = i + 1
    newcode = newcode + [genset[len(genset)-1]]
    return newcode
```