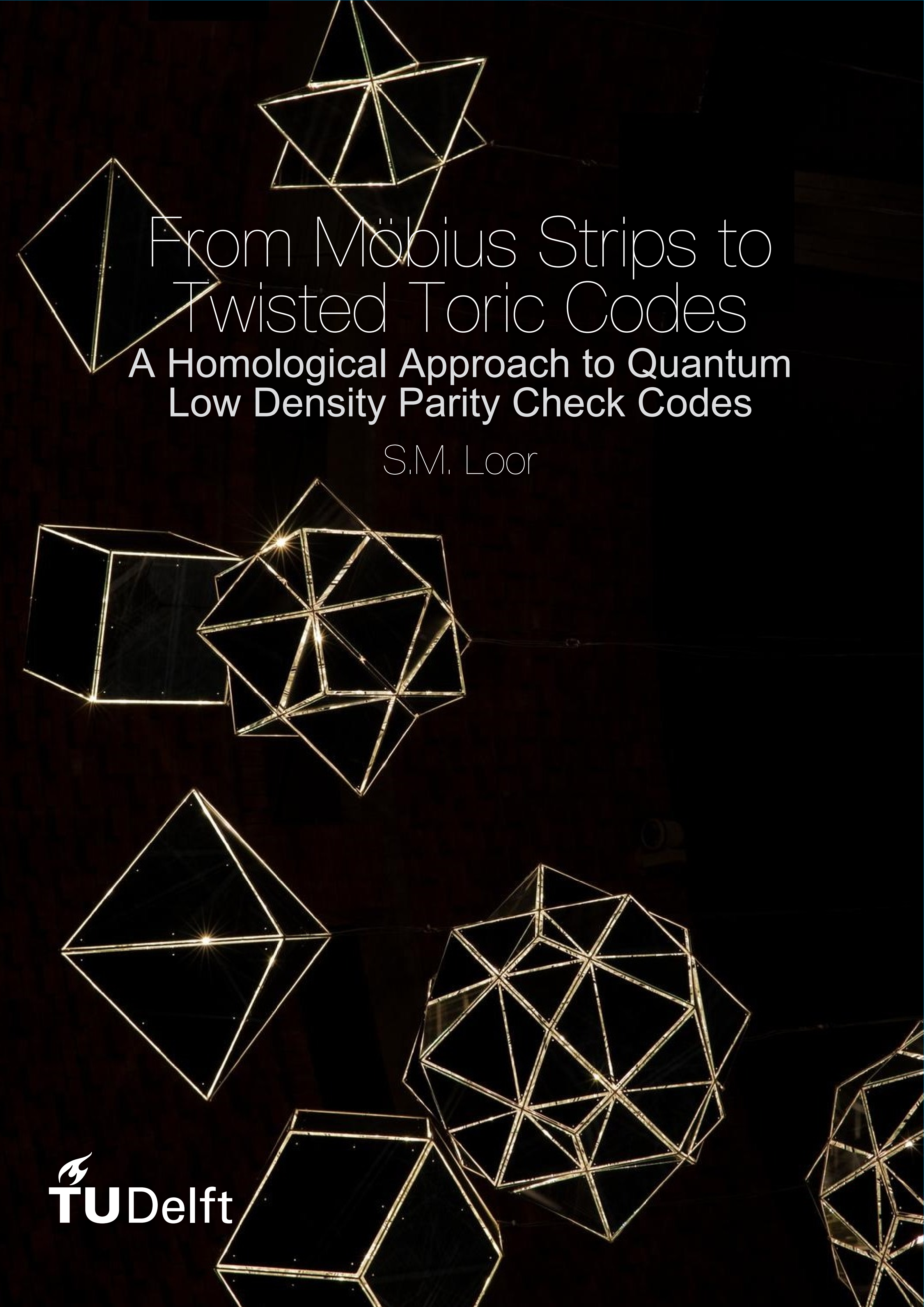


From Möbius Strips to Twisted Toric Codes

A Homological Approach to Quantum Low Density Parity Check Codes

S.M. Loor

From Möbius Strips to Twisted Toric Codes

A Homological Approach to Quantum Low Density Parity Check Codes

by

S.M. Loor

to obtain the degree of Master of Science in
Applied Physics and Applied Mathematics
at Delft University of Technology,
to be defended publicly on Monday, September 5th, 2022 at 15:00.

Student number:	4574117	
Project duration:	November 1, 2021 – August 31, 2022	
Thesis committee:	Prof. dr. B. M. Terhal	QuTech, supervisor
	Dr. B. Janssens	TU Delft, supervisor
	Prof. dr. D. C. Gijswijt	TU Delft
	Dr. J. Borregaard	QuTech

Cover image:
Eliasson, Olafur. *Your Sound Galaxy*. 2012. Photograph by Xing Yu. Olafureliasson.net. Retrieved on June 5th, 2022.

TU Delft

The stars we are given. The constellations we make.
Rebecca Solnit

Preface

Today, exactly six years ago, I left my home country to pursue my studies in Applied Physics and Applied Mathematics, two fields that I found to be utterly fascinating. Today, six years later, so much has changed: I have met many beautiful souls, got to enjoy many great things, but also got to endure many hardships. Much so during the pandemic, when studying, at least for me, lost all of its charm. Reduced to a never-ending sequence of online lectures, its social and formative aspects died along with my love for it. Consequently, bound by lockdowns and without any distractions, my relationship with physics and mathematics became quite rocky. After many, many months that I remember as just endlessly staring at a screen that went on and on about universal enveloping algebras, Haar measures and cryogenic CMOS, I had become convinced that my future did not lie within the walls of a research institute, and had started to look for it elsewhere.

Last year, in April, I approached Barbara about writing a thesis at her group, and she suggested looking into LDPC codes. Bas, coincidentally, had attended a talk on these codes once, where he had noticed their rich homological structure. Although I knew nothing about algebraic topology or homology (and, admittedly, also very little about algebra and topology), I decided to give it a shot — and that has made all the difference.

Much to my surprise, writing my thesis has been an incredibly enjoyable adventure. I got the opportunity to explore amazingly beautiful mathematics that, given my background in applied mathematics, lay far — very far — outside of my comfort zone, and to take a deep dive into the world of quantum error correction. But, much more importantly, I was taken by the hand by two of the most inspiring people I have ever met. I am greatly indebted to Barbara and Bas. To Barbara, a truly encyclopedic quantum powerhouse, for the many very, very long meetings that did justice to both the words *brain* and *storm* in the word *brainstorming*, for the incredibly fun conversations before, during, and after lunch, and for fiercely trying to show me how to think like a researcher. To Bas, a mathematical poet of whom I am certain that he dreams in the language of mathematics, for the very, very long meetings, after each of which I left the room convinced that deep secrets of Nature had just been revealed to me, for the little words of wisdom with a very big significance, and for trying to teach me how to speak in (mathematical) tongues. Finally, to you both, for helping me see the beauty of our fields again. *Really* seeing, the seeing that is done with the heart.

I am also indebted to others for enriching those myriad aspects of life that exist beyond the reach of lemmata and formulæ. Thank you, first and foremost, to my mother and to Elmar, for always being there for me — and for everything else. To Sieske and Roel, for their support throughout: *gran tangi*. To Sam, my unbelievably brilliant study buddy since the very beginning and through thick and thin, with whom every conversation, whether about *stare decisis*, category theory, or Sesame Street memes, is always a great joy. To Charlie, my partner in philosophy, for refusing to just calculate lest we falter to the challenging of the *Gestell*. To Saqar, in whose fascination for the beauty of mathematics I am reminded of my own, for the napkins full of mathematics. To my sister Shannon and to my friends, of whom I must surely mention Armanda, Tahisa, Fabian, and Bowy, for always keeping me caffeinated, and for always believing in me.

Stephan Loor
Delft, July 23rd, 2022

Abstract

In the past few years, the search for good quantum low density parity check (qLDPC) codes suddenly took flight, and many different constructions of these codes have since been presented, including many product constructions. As these code constructions have a natural interpretation in the language of homology, this thesis studies the interplay between homological algebra and various recent product constructions of qLDPC codes.

First, we provide an overview of the theory of singular homology, cellular homology, and homological algebra over vector spaces, and use this theory to analyse two product constructions: the hypergraph product construction and the distance balancing procedure. These constructions can be interpreted as tensor products of chain complexes over vector spaces. We present new proofs for results from the literature regarding the distance and the number of encoded qubits of these product codes.

Secondly, we survey the theory of homology over ring modules, and use this theory to interpret and analyse another product code construction (the lifted product code construction), which is a generalisation of the hypergraph product construction. We prove a Künneth theorem for these codes, and use this theorem to prove a formula for the number of qubits such codes encode.

Thirdly, we investigate the homology of fibre bundles. To this end, we provide an overview of the theory of covering spaces and of fibre bundles, as well as an overview of the theory of homology with local coefficients and of spectral sequences. We then calculate the homology of a specific class of fibre bundles using three different methods. After this discussion, we consider two product code constructions: the fibre bundle product construction and the balanced product construction. We explicate their mathematical foundations, and use these insights to prove two new results for the number of qubits that these product codes encode. Finally, we explain under which conditions these two code constructions and the lifted product construction coincide.

Lastly, we consider a specific example of fibre bundle product codes: the twisted toric code. We determine an analytic expression for the distance of such a code and verify this expression using numerical simulations. Furthermore, we perform an extensive numerical study on these codes to determine how performing a twist alters the scaling of the logical error rate of such codes. We present a new analytic method to explain the scaling of these codes on a small domain, and verify the validity of these calculations by comparing them with the results of our numerical simulations.

Contents

Introduction

One of the grand promises of quantum computing is the ability to solve some hard computational problems exponentially faster than any classical computer can or ever could, like factorising large numbers or simulating complex molecular structures [Sho94, NC10, CRO$^+$19]. Recently, some first experimental demonstrations of this so-called quantum supremacy have been presented [ZCC$^+$22, MLFA$^+$22, AAB$^+$19], albeit in very limited settings. Regardless of the large potential of quantum computing, the road towards building a fully functional quantum processor is plagued by many difficulties. Firstly, quantum systems are subject to decoherence: due to interaction with the environment, the information encoded in the system is quickly lost. Secondly, errors occur while manipulating or reading out the information stored.

The way forward is to try to mitigate the effects of physical errors on the system by using quantum error correction. In short, this amounts to encoding information into a large, highly entangled multi-qubit system in such a way that one can perform measurements to detect and correct errors without destroying the encoded information. The threshold theorem by Aharanov and Ben-Or [ABO08] guarantees that this is not a completely hopeless endeavour, as we can be confident that any quantum circuit can be simulated in a fault-tolerant manner with polylogarithmic overhead, provided that the rate of occurrence of physical errors in the system is below some threshold value, which, for the surface code [BK98], was proved to be around 1% [FSG09].

Looking at the performance of the best-known classical error correcting codes, one would hope to be able to find families of quantum error correcting codes that perform just as well as so-called good codes, in the sense that they possess a non-vanishing encoding rate and a distance that is linear in the code size. While such well-performing quantum correcting schemes do exist [BE21b], the weight of the parity checks involved in these codes grows rapidly with the system size (i.e. the number of qubits), therefore rendering these codes experimentally infeasible. This issue can be resolved by restricting one's attention to the class of quantum Low Density Parity Check (qLDPC) codes.

Interestingly enough, in 2013, Gottesman proved that by considering this class of codes, one could improve Aharanov and Ben-Or's threshold theorem to require only an asymptotically constant overhead to fault-tolerantly simulate an arbitrary quantum circuit [Got13]. Yet, although many examples of qLDPC codes, like Kitaev's toric code [Kit03] and the surface code, were known at the time, none of these known examples could come close to their classical counterparts in terms of their performance. Therefore, researchers set out to search for families of good qLDPC codes.

The first major result in the search for good qLDPC codes was obtained in 2009, when Zémor and Tillich constructed a family of qLDPC codes with a non-vanishing rate and a distance $\Omega\left(\sqrt{n}\right)$, where n is the number of physical qubits [TZ09]. This construction entailed taking products of graphs that can be associated to classical codes, so-called *Tanner graphs*, and was called the hypergraph product construction. Researchers soon realised that this construction could be understood in terms of homology. This homological interpretation of the hypergraph product construction is no mere coincidence — in fact, it was already known that the class of so-called Calderbank-Shor-Steane (CSS) codes, which contains all the well-known examples of qLDPC codes, admits a natural description in homological terms [AC19]. Although homology was initially developed as a technique for answering questions in algebraic topology regarding the structure of topological spaces, it has since grown into its own sub-field of mathematics: homological algebra. Within the framework of homological algebra, CSS codes can be interpreted as chain complexes, the principal objects of homological algebra, whereas the parity check matrices and their stabiliser checks can be seen as the differentials of the chain complexes.

Taking a homological approach to the hypergraph product codes offered a natural interpretation of this construction in terms of the tensor product of chain complexes over vector spaces, and, moreover, this approach allowed one to determine the number of encoded qubits k in a very elegant and easy manner.

Ever since, researchers have set out to improve on this tensor product construction by generalising it: in 2020, Evra, Kaufman, and Zémor introduced the so-called distance balancing procedure, which they used to construct codes with distance $\Omega\left(n^{1/2}\log(n^{1/2})\right)$ [EKZ20]. Afterwards, researchers tried to introduce *twists* and *lifts* to the hypergraph product construction to improve upon them: Hastings and Freedman used their fibre bundle product to obtain a family of codes with distances $\Omega(n^{3/5}\text{polylog}(n))$ [HHO21] (but vanishing rates), Breuckmann and Eberhardt used their balanced product construction to obtain distances of $\Omega(n^{3/5})$ (but rates of $\Theta(n^{-1/5})$), and finally, Panteleev and Kalachev used their lifted product code construction to construct asymptotically good LDPC codes for the first time ever [PK22b, PK22a].

Although motivated by the equivalence between the hypergraph product of codes and the tensor product of chain complexes over vector spaces, the homological aspects of the code constructions developed afterwards are substantially less clear: firstly, no general way of determining the number of encoded qubits k for the lifted product codes is known in the current literature. Secondly, the exact nature of the relation of the fibre bundle product codes and the balanced product codes to fibre bundles is unclear. Thirdly, although it is clear that some relationship exists between the fibre bundle product codes and the balanced product [BE21b], the literature is unclear about what this relationship is and when it holds precisely. Similarly, it is unclear how to relate the fibre bundle product to the lifted product construction. In this thesis, we will therefore try to shed a light on these issues from the perspective of homological algebra, and we will, moreover, try to understand how applying a twist affects a code by studying the example of the toric code. To this end, we will answer the following questions:

1. How can one understand the hypergraph product construction without the use of Tanner graphs?
2. Can we determine the number of encoded qubits k of the lifted product code construction using ring module homology?
3. How can we use fibre bundles to understand the underlying ideas of the fibre bundle and balanced product code construction?
4. How can one calculate the homology of a fibre bundle, and how can one use these techniques to analyse the different code constructions?
5. How do the fibre bundle product, balanced product, and lifted product constructions relate, and when do they coincide?
6. How does the introduction of a twist alter the performance of Kitaev's toric code?

We provide an extensive survey of all the mathematical theory needed to understand these code constructions, to rephrase them in terms of homology, and to answer the questions pertaining to them. This thesis is therefore divided into three parts: in Part I, we survey the theory of vector space homology and consider the product constructions that can be understood in those terms: the hypergraph product, the distance balancing procedure, and their generalisation [ZP19]. Afterwards, in Part II, we discuss homological algebra over ring modules and present and analyse the lifted product construction. Subsequently, in Part III, we consider the theory of fibre bundles and the related theories of homology with local coefficients and spectral sequences, after which we turn to our study of the homology of fibre bundles and to the twisted product constructions, i.e., the fibre bundle product and the balanced product. Finally, we study the effects of applying a twist to the toric code.

For the convenience of the more mathematically inclined reader, we note that some sections of this thesis provide an overview of standard mathematical theory and can therefore be skipped by the reader who is well acquainted with these topics. In particular, Chapter 1 provides an account of homotopy theory, of singular and cellular homology, and of homological algebra over vector spaces, Chapter 4 provides an account of ring module homology, Section 6.1 provides an overview of standard covering space theory, and Section 6.2 provides an overview of the theory of fibre bundles.

I

Vector Space Homology and Tensor Product Codes

1 Homology and Homotopy

In this chapter, we will introduce two fundamental tools in algebraic topology: the homotopy and homology groups of a topological space. Both homology and homotopy theory revolve around associating algebraic objects — so-called *algebraic invariants* — to topological spaces, in order to answer questions about these topological objects that are often otherwise difficult to answer using standard topological arguments. Such questions are often (non-)existence questions — the classical motivational example of such a question is the question whether there exists a homeomorphism between a torus and a sphere. We will first show how homotopy answers such a question, and we will provide an account of homotopy theory. Next, we will show how homology aims to answer such questions, after which we will formally introduce two forms of homology: singular and cellular homology. Afterwards, we will take the first few steps towards formalising these constructions into a more abstract framework: homological algebra. Before all of that, however, we present the classical motivational example of the torus and the sphere. The interested reader can find a more elaborate treatment of the contents of this chapter in [Hat01, Sag21, Wei94, Rie16].

1.1. Finding Homeomorphisms between S^2 and $\mathbb{T}$

Are there any homeomorphisms between the torus $\mathbb{T}$ and the 2-sphere S^2? To answer such a question affirmatively, is — in principle — rather simple: one has but to give an example of a map between these two spaces that is an open, continuous bijective map. To prove that the answer to this question is "no", however, is substantially harder, as one basically has to prove for each and every map $S^2 \to \mathbb{T}$ that it is not a homeomorphism. To do so by direct computation is a hopeless endeavour, thus, one will have to resort to alternative methods that take into account the structure of the two spaces at hand.

An interesting way forward is given by the observation that homeomorphisms necessarily preserve loops. Therefore, in order to answer questions concerning the existence of homeomorphisms between e.g. the torus and the sphere, it could be instructive to consider the structure of the loops on these spaces — this is the fundamental idea behind homotopy. Let us think this through, by first considering two loops on the sphere S^2 that go through the same basepoint b:

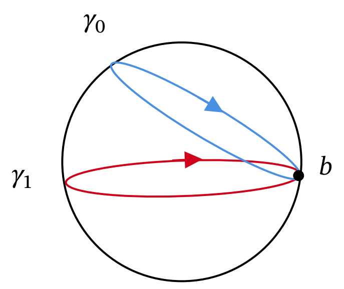


Figure 1.1: The 2-sphere with base point b and two loops γ_0 and γ_1 that are based in b.

With a little bit of imagination, one can see that the loop γ_1 can be continuously deformed into loop γ_0

and vice versa by "interpolating" between these two loops:

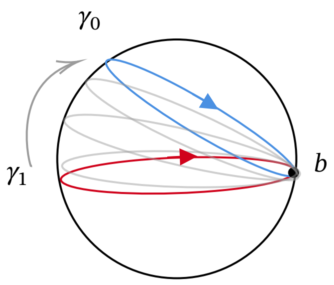


Figure 1.2: The 2-sphere with base point b and two loops γ_0 and γ_1 that are based in b. By continuously deforming γ_1 in the direction of the arrow, one obtains γ_0.

Although there are awfully many possible loops on the sphere with basepoint b, we see that — up to continuous deformation — many of them are in fact similar. By restricting our attention to loops that cannot be deformed into each other, we see that not much remains, as every loop over the 2-sphere can be continuously deformed into the constant loop, even more exotic ones:

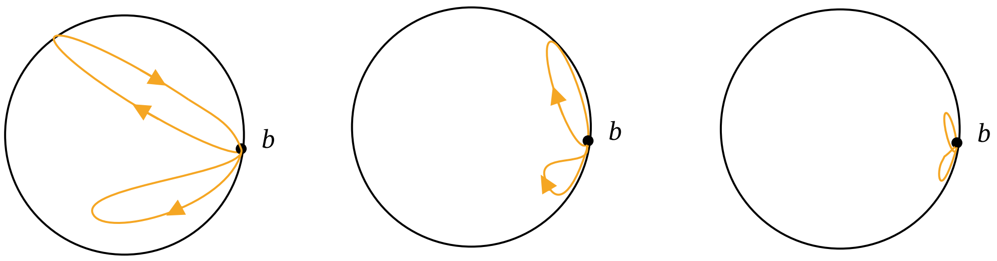


Figure 1.3: The 2-sphere with base point b and two loops γ_0 and γ_1 that are based in b.

This information is encoded in the so-called *fundamental group* of S^2, $\pi_1(S^2)$:

$$\pi_1(S^2) = 0 \tag{1.1}$$

Next, let us consider the torus. Following the same procedure of modding out continuous deformations, one sees that there are at least two fundamental classes of loops that cannot be deformed into each other:

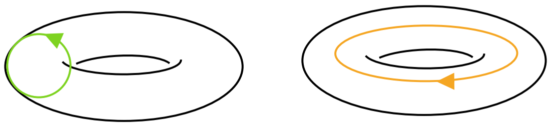


Figure 1.4: The torus $\mathbb{T}$, along with two loops that cannot be continuously deformed into each other.

Let us consider just one of these loops with basepoint b, say, γ_1. By flipping the direction of the loop, we actually obtain a new loop that cannot be deformed into our original loop. Let us denote this loop by γ_1^{-1}. Similarly, we see that by composing this loop with itself, we obtain a new loop that goes around twice. This newly found loop, however, cannot be deformed into our original loop either. Let us denote the composition of two loops by $\gamma_1^2 := \gamma_1 * \gamma_1$. We can proceed inductively to define γ_1^n. Similarly, we see that $\gamma_1 * \gamma_1^{-1} = \gamma_1^{-1} * \gamma_1$ is contractible to the point b, and is therefore, up to continuous deformation, the constant loop. As such, we see that the composition operation, when acting on the classes of loops, allows one to identify this loop space with a group, where each of the loops generates a subgroup isomorphic to $\mathbb{Z}$. Therefore, we conclude that the fundamental group of the torus $\mathbb{T}$ is:

$$\pi_1(\mathbb{T}) = \mathbb{Z} \oplus \mathbb{Z} \tag{1.2}$$

The point, now, is that homeomorphisms do not just send loops to loops, rather, they also preserve the loop structure of spaces and hence descend to isomorphisms between the loop spaces modulo continuous deformations, that is, to the fundamental groups in the form of group isomorphisms. As the fundamental groups of the torus and the sphere are not isomorphic, we can thus safely conclude that no homeomorphism between these two spaces can exist.

A different, although somewhat related approach to solving this conundrum begins with the observation that the holes of the torus and the hole within the 2-sphere are also quite different. This is the information that homology tries to capture. Yet, a hole is the absence of space, so how can it be captured purely in terms of spatial objects? Let us demonstrate this by considering a closed, 1-dimensional loop on the sphere. One can convince oneself quite easily that such a closed, 1-dimensional loop is always the boundary of some 2-dimensional subspace on the sphere:

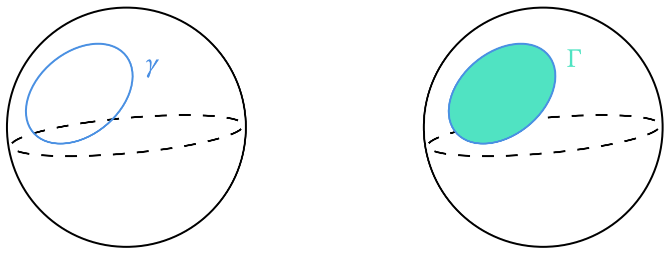


Figure 1.5: On the left, we see the 2-sphere, along with a closed 1-dimensional subspace on the 2-sphere, γ. On the right, we see that this closed 1-dimensional subspace can be interpreted as being the boundary of Γ, a 2-dimensional subspace of the 2-sphere.

In some sense, this means that there are no "1-dimensional holes", as every possible 1-dimensional hole — i.e. every 1-dimensional boundary without any space inside of it — is actually filled by a 2-dimensional part of the space. In the language of homology, this is equivalent to saying that the 1st homology group is trivial, i.e.:

$$\mathcal{H}_1(S^2) = 0 \tag{1.3}$$

For the torus, however, this is not the case. Consider these two loops:

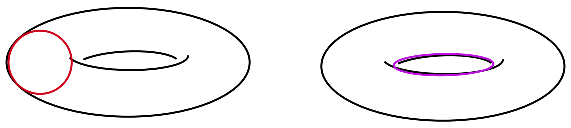


Figure 1.6: Two examples of 1-dimensional subspaces of the torus that cannot be interpreted as the boundary of a 2-dimensional subspace on the torus.

One sees that these two loops can both not be considered to be the boundary of some two-dimensional object in the space. The first homology group of the torus captures precisely this information:

$$\mathcal{H}_1(\mathbb{T}) = \mathbb{Z} \oplus \mathbb{Z} \tag{1.4}$$

If one can prove that homeomorphisms preserve the holey structure of spaces (as captured by its homology), one immediately finds that no homeomorphisms between the torus and the 2-sphere can exist. This is indeed the case, and we will do so rigorously in the upcoming sections.

1.2. Homotopy Theory

As we showed in the example above, homotopy tries to capture the structure of the loops of topological spaces. The time has come to formalise this idea.

Before we proceed, let us first establish the following conventions: in this section, X, Y will denote topological spaces, and $f, g : X \to Y$ will denote continuous maps between them.

We first formalise the notion of a continuous deformation between functions.

Definition 1.1. A *homotopy* $H : X \times [0,1] \to Y$ between $f : X \to Y$ and $g : X \to Y$ is a continuous map such that:

$$H(x,0) = f(x), \quad H(x,1) = g(x) \tag{1.5}$$

If a homotopy between f and g exists, we call f and g homotopic, and denote this by $f \simeq g$.

In our introductory example, the paths on the 2-sphere were maps $\gamma_i : [0,1] \to S^2$, and the continuous deformation between them was actually a continuous map $H : [0,1] \times [0,1] \to S^2$ such that $H(t,i) = \gamma_i(t)$.

The Homotopy (and homology) of two spaces is not just preserved by homeomorphisms, but also by a weaker form of equivalence of topological spaces.

Definition 1.2. $f : X \to Y$ is said to be a *homotopy equivalence* between X and Y if $\exists g : Y \to X$:

$$f \circ g \simeq \mathrm{id}_Y \quad \wedge \quad g \circ f \simeq \mathrm{id}_X \tag{1.6}$$

If a homotopy equivalence between X and Y exists, X and Y are said to be *homotopy equivalent*. We denote this by $X \simeq Y$.

We are now ready to consider homotopy groups.

Definition 1.3. Let X be a topological space, and let $x_0 \in X$ be its base point. For every $n \geq 0$, we define the n^{th} *homotopy group* to be the homotopy class of basepoint preserving maps between the n-sphere and X, i.e.:

$$\pi_n(X, x_0) = [(S^n, s_0), (X, x_0)]_* \tag{1.7}$$

We note that the homotopy groups indeed possess group structure, but given that we will only be concerned with the first homotopy group (i.e., the fundamental group) in this thesis, we do not discuss this matter any further and refer the interested reader to e.g. [Sag21, Hat01]. Furthermore, one can take on a categorical approach to homotopy. For example, one can define the fundamental group functor from the category of based topological spaces to the category of groups, i.e. π_1 : **Top*** $\to$ **Grp**.

On a final note, however, we remark that homotopy groups tend to become complicated to calculate quite quickly, for example, the homotopy groups $\pi_n(S^m, s_0)$ are quite complicated for $n \geq m$, whereas the homology groups $\mathcal{H}_n(S^m)$ are all trivial. Furthermore, there are relatively simple spaces, like the wedge product $S^1 \bigvee S^3$ and its double cover, that have isomorphic homotopy groups but are not homotopy equivalent, as follows e.g. from the fact that their homology groups are different. Therefore, we see that there are more than enough reasons not to restrict one's attention to just homotopy groups.

1.3. Homology

Homology aims to resolve questions pertaining to (the equivalence between) topological spaces by looking at the structures of the holes of these spaces, as was shown in the example of the torus and the 2-sphere. We first formalise the theory behind that example, the so-called singular homology theory. Afterwards, we present a different homology theory, which can often be used to calculate the singular homology of a topological space more easily: cellular homology.

1.3.1. Singular Homology

In the motivational example, we distinguished 1-dimensional and 2-dimensional subspaces of the 2-sphere and the torus. We can formalise this idea of n-dimensional subspaces with the notion of an n-simplex. Before we can introduce those, however, we first need the notion of a standard n-simplex:

Definition 1.4. The *standard n-simplex* $\Delta_n \subseteq \mathbb{R}^{n_1}$ is the subspace given by:

$$\Delta^n := \left\{ (t_0, \dots, t_n) : t_i \geq 0 \, \forall i \wedge \sum_i t_i = 1 \right\} \tag{1.8}$$

The standard n-simplices can be seen as n-dimensional subspaces of $\mathbb{R}^{n+1}$:

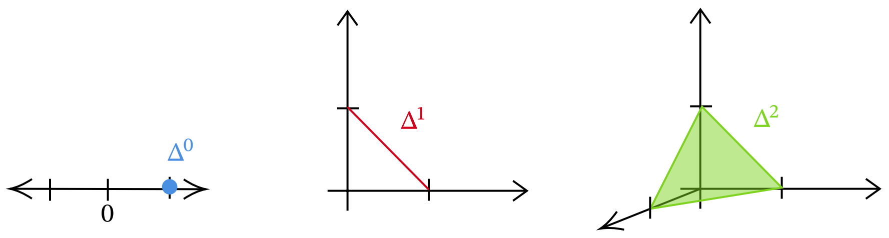


Figure 1.7: From left to right: the standard 0-simplex Δ^0, the standard 1-simplex Δ^1, and the standard 2-simplex Δ^2.

Embedding these into X gives us a way of talking about n-dimensional subspaces of X:

Definition 1.5. A continuous function $\sigma_n : \Delta^n \to X$ is called an *n-simplex* of X. We denote the set of all n-simplices of X by S_n(X).

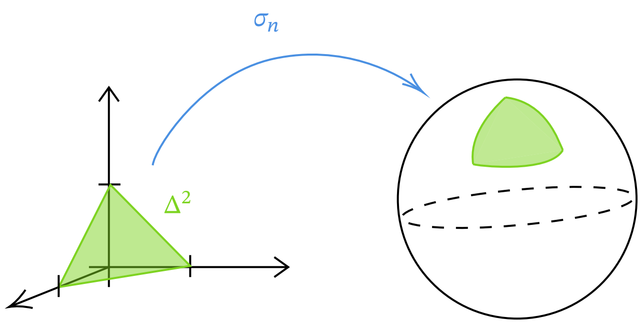


Figure 1.8: An example of a 2-simplex σ_2 of the 2-sphere S^2, which continuously maps the standard 2-simplex Δ^2 to the 2-sphere.

Note that we can identify $S_0(X)$ with X, while we can identify $S_1(X)$ with all the paths in X.

In order to speak of holes, we need to be able to talk about boundaries of spaces. This implies that we need a way to e.g. say that x and y are the boundary points of some path γ. To do so, we consider not just the simplices, but take them as the basis of a free abelian group:

Definition 1.6. Let A be an abelian group and let X be a topological space. The set of *singular n-chains of X with coefficients in A*, $C_n(X;A)$, is defined as:

$$C_n(X;A) := A[S_n(X)] \tag{1.9}$$

That is, $C_n(X;A)$ can be interpreted as the abelian group consisting of all finite, formal sums of elements of the form $a \cdot \sigma$, where $a \in A$ and $\sigma \in S_n(X)$, such that:

1. $a_1 \cdot \sigma + a_2 \cdot \sigma = (a_1 + a_2) \cdot \sigma$ for all $a_1, a_2 \in A$ and $\sigma \in S_n(X)$
2. $0 \cdot \sigma_1 = 0 \cdot \sigma_2$ for all $\sigma_1, \sigma_2 \in S_n(X)$.

We now have all the preliminary notions we need to work towards defining boundary maps. To this end, we first note introduce the concept of a face map, which are simply the maps that return a boundary of a standard n-simplex:

Definition 1.7. The map $\delta_i^n : \Delta^{n-1} \to \Delta^n$, given by:

$$\delta_i^n(t_0, \dots, t_{n-1}) = (t_0, \dots, t_{i-1}, 0, t_{i+1}, \dots, t_n) \tag{1.10}$$

is called the ith *face map*.

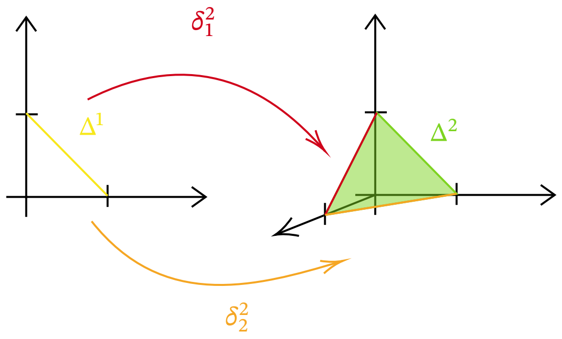


Figure 1.9: Two examples of face maps. One can see that each of these face map returns a different side of the standard 2-simplex.

By precomposing an n-simplex σ_n with a face map, we see that it is possible to define an $n-1$-simplex, which is given by the restriction of σ_n to one of its boundaries, as is illustrated in Figure 1.10.

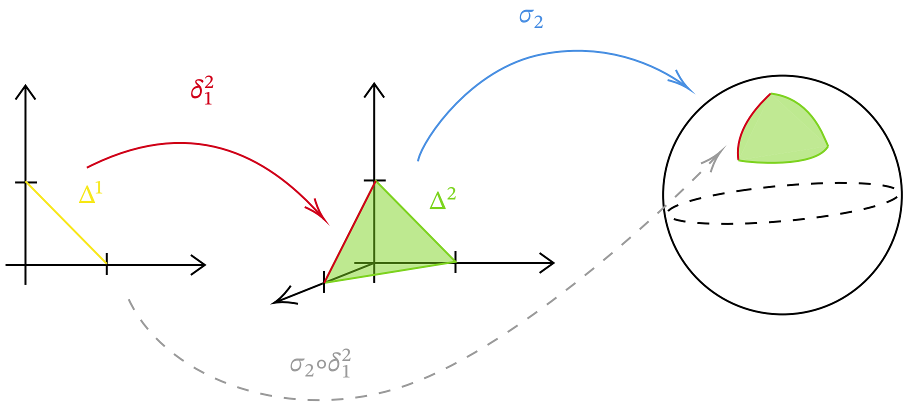


Figure 1.10: An example of how a 1-simplex σ_1 can be constructed from a 2-simplex σ_2 by precomposing it with a face map δ_1^2, which effectively embeds the standard 1-simplex Δ^1 into the boundary of the standard 2-simplex Δ^2.

By precomposing with the various face maps, therefore, we can find the various sides of an n-simplex. To determine the complete boundary of such an n-simplex, we take alternating sums of its sides:

Definition 1.8. We denote by $d_i^n : C_n(X;A) \to C_{n-1}(X;A)$ the map $a \cdot \sigma_n \mapsto a \cdot (\sigma_n \circ \delta_i^n)$. Using these maps, we can define the n^{th} boundary map $\partial_n : C_n(X;A) \to C_{n-1}(X;A)$ as:

$$\partial_n = \sum_{i=0}^{n} (-1)^n d_i^n \tag{1.11}$$

We can nicely display the constructions presented up until now in the form of a so-called *chain complex*:

$$\cdots \xrightarrow{\partial_4} C_3(X;A) \xrightarrow{\partial_3} C_2(X;A) \xrightarrow{\partial_2} C_1(X;A) \xrightarrow{\partial_1} C_0(X;A)$$

Later, we will explain in detail why this is much more than just a nice way of displaying these constructions. For now, however, let us work towards defining the central objects of this section: homology groups. Before doing so, however, we first point to the following lemma, which can be proved in a straightforward way by writing out the definitions (see e.g. [Sag21]):

Lemma 1.1. *Given a topological space X and an abelian group A, the boundary maps $\partial_n : C_n(X;A) \to C_{n-1}(X;A)$ satisfy the following property:*

$$\partial_n \circ \partial_{n+1} = 0 \tag{1.12}$$

Or, equivalently: Im$(\partial_{n+1}) \subseteq$ ker(∂_n).

Because of this lemma, we can now finally define the homology groups of a topological space.

Definition 1.9. Let A be an abelian group, let X be a topological space, and let $\partial_n : C_n(X;A) \to C_{n-1}(X;A)$ denote the boundary maps of X. The n^{th} *homology group* of X with coefficients in A, $\mathcal{H}_n(X;A)$, is defined as:

$$\mathcal{H}_n(X;A) := \ker(\partial_n)/\operatorname{Im}(\partial_{n+1}) \tag{1.13}$$

We take ∂_0 to be the zero map, such that $\mathcal{H}_0(X;A) = C_0(X;A)/\operatorname{Im}(\partial_1)$.

One can take a categorical approach to singular homology, and interpret this construction as a *homology functor* from the category of topological spaces to the category of abelian groups, $\mathcal{H}_n : \mathbf{Top} \to \mathbf{Ab}$. This perspective is quite fruitful and will be exploited later on. For now, however, we simply note that this perspective can be used to show that the homology functor is invariant under homotopy equivalences (see e.g. [Sag21] for a detailed proof).

We now introduce one concept:

Definition 1.10. Given a topological space X and a subspace $X' \subseteq X$. The *relative homology groups* of the pair (X, X'), which are denoted by $\mathcal{H}_n(X, X'; A)$ are defined as the homology groups of the following quotient complex:

$$C_n(X, X'; A) := C_n(X;A)/C_n(X';A) \tag{1.14}$$

Before proceeding, we quickly mention that the homology and homotopy groups of spaces can, in some instances, be related to each other, for example using Hurewicz' theorem (see e.g. [Sag21]). Moreover, one can easily relate the first homology group and the fundamental group of a path-connected space [Sag21]:

Theorem 1.2. *Let X be a path-connected topological space. Then:*

$$\mathcal{H}_1(X;\mathbb{Z}) \simeq \pi_1(X, x_0)/[\pi_1(X, x_0), \pi_1(X, x_0)] \tag{1.15}$$

On a final note, we note that it suffices to restrict one's attention to the case where $A = \mathbb{Z}$, as the so-called *universal coefficient theorem* (see e.g. [Hat01]) allows one to obtain the homology groups of X with coefficients in A from its homology groups with coefficients in $\mathbb{Z}$. In particular, if A is a vector space, one finds:

$$\mathcal{H}_n(X;A) \simeq \mathcal{H}_n(X;\mathbb{Z}) \otimes_{\mathbb{Z}} A \tag{1.16}$$

In what is to come, we will therefore mostly discuss the case for $A = \mathbb{Z}$, and, when we do so, we will omit A from our notation, such that e.g. $C_n(X)$ will denote $C_n(X;\mathbb{Z})$, and $\mathcal{H}_n(X)$ will denote $\mathcal{H}_n(X;\mathbb{Z})$.

1.3.2. Cellular Homology

Calculating the singular homology of spaces using elementary definitions is often quite cumbersome. One way of greatly simplifying calculations for well-behaved spaces (which we call *CW-complexes*) is using cellular homology.

The fundamental idea behind cellular homology is to break larger spaces up into smaller pieces (called *cells*) in a systematic manner. As the definitions and the constructions involved are quite formal, let us first provide a small example of how one can break up a CW-complex into smaller pieces. We show this process for a filled triangle in Figure 1.11.

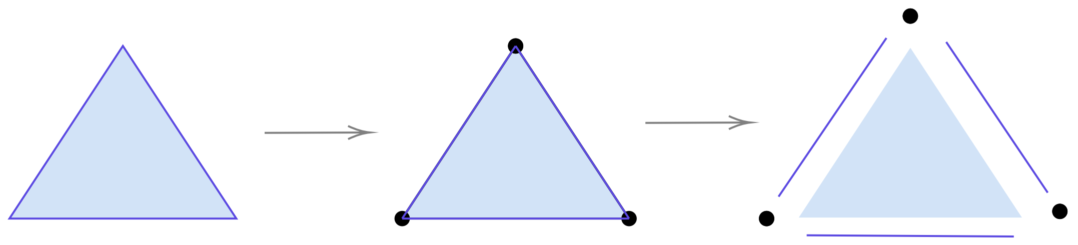


Figure 1.11: From left to right: we consider a filled triangle, which is a two-dimensional space. We can distinguish the interior of the triangle, and its boundary, which is a 1 dimensional space. Each of the edges forming its boundary, however, has a boundary as well, consisting of two points, which are 0-dimensional spaces. We can thus compose the triangle out of three points (0-cells), three edges (1-cells), and its interior (a 2-cell).

As Figure 1.11 illustrates, we can build up a triangle using three 0-dimensional cells (the vertices of the triangle), three 1-cells (the edges forming its boundary), and one 2-dimensional cell (its interior). This can be done as follows: one starts with the three 0-cells. One introduces the three 1-cells and identifies their boundary points with the three 0-cells. Afterwards, one introduces the 2-cell and identifies its boundary with the space thus far constructed. Such a procedure of constructing a space is formalised using the notion of a *cellular attachment*.

We now proceed with the formal definitions. For the purpose of defining cellular attachments, we first define the notion of a pushout.

Definition 1.11. Let X, X', Y be topological spaces, and consider the continuous maps $i : X' \to X$ and $f : X' \to Y$. We can represent this with the following diagram:

$$\begin{tikzcd} X & X' \arrow[l, "i"'] \arrow[r, "f"] & Y \end{tikzcd}$$

The *pushout* of this diagram, $X \cup_{X'} Y$ is defined as $X \cup_{X'} Y := X \sqcup Y / \sim$, with $i(x') \sim f(x')$.

Alternatively, one can define the pushout in categorical terms. To this end, we note that the pushout $X \cup_{X'} Y$ is the space satisfying the following universal property: For every topological space Z for which the following diagram commutes [1]

$$\begin{tikzcd} X' \arrow[r, "f"] \arrow[d, "i"'] & Y \arrow[d] \\ X \arrow[r] & Z \end{tikzcd}$$

we have that the following diagram commutes as well:

$$\begin{tikzcd} X' \arrow[r, "f"] \arrow[d, "i"'] & Y \arrow[d] \arrow[ddr] & \\ X \arrow[r] \arrow[drr] & X \cup_{X'} Y \arrow[dr, dashed, "\exists!"] & \\ & & Z \end{tikzcd}$$

Within this categorical framework, we call the square a *pushout square* if the map $X \cup_{X'} Y \to Z$ is a homeomorphism.

One can use the concept of a pushout to define cell attachments. To this end, we denote the n-disk by D^n for every $n \geq 0$. Note that we take the 0-disk D^0 to be the 1-point set $*$, and that, moreover, its boundary is taken to be empty.

Definition 1.12. Let X, Y be topological spaces such that $X \to Y$ is a continuous map. We say that *Y arises from X by attaching n-cells* if there is some index set $\mathcal{J}$ along with a so-called *attaching map* $f : \mathcal{J} \times \partial D^n \to X$ such that the following holds:

$$Y \cong \mathcal{J} \times D^n \cup_{\mathcal{J} \times \partial D^n} X \tag{1.17}$$

Phrased in categorical terms, this is equivalent to requiring that the following square is a pushout square:

$$\begin{tikzcd} \mathcal{J} \times \partial D^n \arrow[r, "f"] \arrow[d, "i"'] & X \arrow[d] \\ \mathcal{J} \times D^n \arrow[r] & Y \end{tikzcd}$$

[1]We note that a diagram is said to commute if all compositions starting from one set and going to another yield the same result. Explicitly, in this diagram, it means that the composition of the map $X \to Z$ with i is equal to the composition of the map $Y \to Z$ with f.

Let us try to give some intuition for this definition using an example: noting that $D^1 = [-1, 1]$, we show how S^1 arises from the one-point space $X = *$ by attaching a 1-cell with attaching map $f : [-1, 1] \to *$ in Figure 1.12.

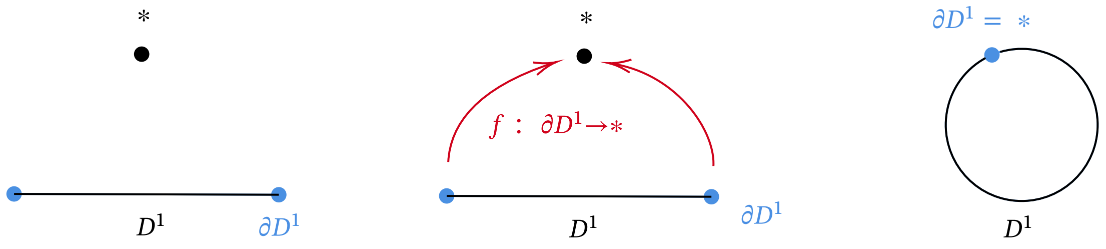


Figure 1.12: From left to right: The 1-disk D^1, which is homeomorphic to the unit sphere, along with its boundary ∂D^1, which consists of two points, and the one-point set $*$. One can use the attaching map $f : \partial D^1 \to *$ to identify both points in ∂D^1 with $*$. In doing so, one obtains the circle S^1.

The spaces we are interested in, CW-complexes, are spaces that can be built up by repeatedly attaching n-cells.

Definition 1.13. Given a topological space X_{-1}. A topological space X is called a *CW-complex relative to X_{-1}* if:

1. there is a sequence of subspaces $X_0, X_1, \ldots$ such that $X_{-1} \subseteq X_0 \subseteq X_1 \subseteq \ldots \subseteq X$.
2. for each $n \geq 0$, we have that X_n arises from attaching n-cells to X_{n-1}
3. a set $O \subseteq X$ is open if and only if $O \cap X_n$ is open in X_n for each $n \geq -1$.

If $X_{-1} = \emptyset$, we call X an *absolute CW-complex*.

Let us establish the convention that we will denote the index set of the n-cells by $\mathcal{J}_n$.

A useful example of a CW-complex is the 1-sphere S^1. This space can be interpreted as an absolute CW-complex with k 0-cells and k 1-cells for every integer $k \geq 1$. We have already seen this for $k = 1$ in Figure 1.12. Another example of this can be found in Figure 1.13.

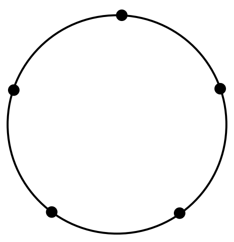


Figure 1.13: An example of a cellular complex of S^1 consisting of five 0-cells and five 1-cells.

We note that under certain conditions, products of CW-complexes are again CW-complexes (see e.g. [Hat01]). For example, if X and Y are finite CW-complexes, their Cartesian product $X \times Y$ is again a CW complex, and, moreover, we can actually identify the n-cells of $X \times Y$ with all pairs of p-cells of X and $n - p$ cells of Y for each $p = 0, \ldots, n$.

One can also visualise this quite nicely for small complexes, as is done for the product of two straight lines in Figure 1.14.

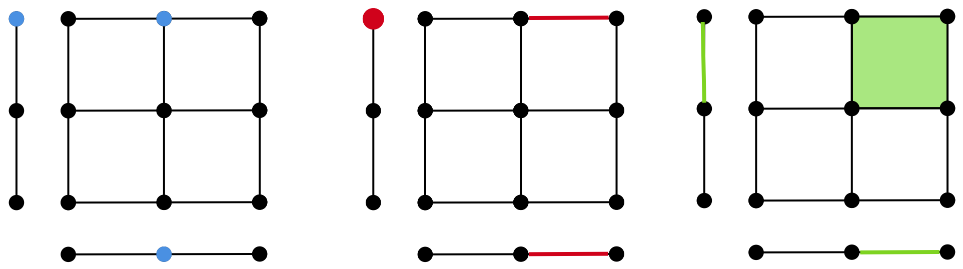


Figure 1.14: The product of two straight lines is a square. Taking two straight lines as CW-complexes with three 0-cells and two 1-cells each, we find a CW-complex for the square consisting of nine 0-cells, twelve 1-cells and four 2-cells. From left to right, one can see how the 0-cells can be identified with pairs of 0 -cells (blue), while 1-cells can be identified with pairs of one 0-cell and one 1-cell (red), and 2-cells can be identified with pairs of 1-cells (green).

Given that we have now formally defined CW-complexes, we can now proceed with defining cellular homology. To this end, we introduce the definition of a cellular complex.

Definition 1.14. Given a CW complex X, we define the *cellular chain complex* of X with coefficients in A as the chain complex:

$$\tilde{C}_n(X;A) := \mathcal{H}_n(X_n, X_{n-1}; A) \tag{1.18}$$

The differential $\tilde{\partial}_n : \mathcal{H}_n(X_n, X_{n-1}; A) \to \mathcal{H}_{n-1}(X_{n-1}, X_{n-2}; A)$ is the *connecting homomorphism* of the triple (X_n, X_{n-1}, X_{n-2}).

The definition of the differential $\tilde{\partial}$ will become clear when we discuss the more general theory of homological algebra. For now, however, it suffices to know that we can define a cellular complex for every CW-complex, and that this cellular complex is isomorphic to a very simple complex [Sag21]:

Theorem 1.3. *Given a CW complex X and its cellular complex $\tilde{C}_n(X;A)$. Then:*

$$\tilde{C}_n(X;A) \simeq A[J_n] \simeq A^{\oplus |J_n|} \tag{1.19}$$

So, returning to the example of the CW-complex of S^1 in Figure 1.13, this theorem immediately tells us that its cellular complex is of the following form:

$$\begin{tikzcd} \cdots \arrow[r] & 0 \arrow[r] & 0 \arrow[r] & A^{\oplus 5} \arrow[r, "\tilde{\partial}_1"] & A^{\oplus 5} \end{tikzcd}$$

For such cellular complexes, one can compute the cellular homology of the underlying topological space.

Definition 1.15. Given a CW complex X. The n^{th} *cellular homology group* of X with coefficients in A is defined as:

$$\tilde{\mathcal{H}}_n(X;A) := \ker(\tilde{\partial}_n) / \operatorname{Im}(\tilde{\partial}_{n+1}) \tag{1.20}$$

Returning to the example of S^1 in Figure 1.13, we see that, as we know the form of its cellular complex, most of its homology groups are trivial:

$$\tilde{\mathcal{H}}_n(S^1;A) = 0 \quad \forall n \geq 2 \tag{1.21}$$

The question, now, is whether using these cellular computations to calculate the cellular homology of a space is in any way relatable to its singular homology. This is, fortunately, indeed the case[Sag21]:

Theorem 1.4. *Given an absolute CW-complex X. Then:*

$$\mathcal{H}_n(X;A) \simeq \mathcal{H}_n(\tilde{C}_n(X);A) \tag{1.22}$$

The hardest part of performing cellular computations is usually determining the differential. The examples we consider here, however, will not be very complicated, and therefore, we omit a discussion of how to do so here, and refer the interested reader to e.g. [Sag21].

1.4. Homological Algebra over Vector Spaces

After having given a brief overview of the theory of singular and cellular homology, let us now present a generalised perspective on the objects that we constructed.

1.4.1. Generalising Chain Complexes and Homology Groups

Firstly, we can introduce the general notion of a chain complex.

Definition 1.16. A *chain complex* (of abelian groups) $(\mathcal{C}_*, \partial_*)$ is a family of abelian groups C_n (for $n \in \mathbb{Z}$), together with group homomorphisms $\partial_n : C_n \to C_{n-1}$ called differentials, such that:

$$\partial_n \circ \partial_{n+1} = 0 \quad \forall n \in \mathbb{Z} \tag{1.23}$$

We note that these chain complexes are the objects of a category, which we denote by **Ch**(**Ab**). A category, however, has morphisms too — these can be defined as follows:

Definition 1.17. Given two chain complexes $(\mathcal{C}_*, \partial_*^C)$ and $(\mathcal{D}_*, \partial_*^D)$. Then a collection of group homomorphisms $f_n : C_n \to D_n$ is called a *morphism of chain complexes*, or a *chain map*, if for each $n \in \mathbb{Z}$, the following diagram commutes:

$$\begin{tikzcd}
C_n \arrow[r, "\partial_n^C"] \arrow[d, "f_n"'] & C_{n-1} \arrow[d, "f_{n-1}"] \\
D_n \arrow[r, "\partial_n^D"'] & D_{n-1}
\end{tikzcd}$$

That is, if the following equation holds:

$$\partial_n^D \circ f_n = f_{n-1} \circ \partial_n^C \tag{1.24}$$

We can now also generalise the notion of homology to arbitrary chain complexes.

Definition 1.18. The n^{th} *homology group* of the chain complex $(\mathcal{C}_*, \partial_*)$ is given by:

$$\mathcal{H}_n(\mathcal{C}) = \ker(\partial_n) / \operatorname{Im}(\partial_{n+1}) \tag{1.25}$$

As the differentials of chain complexes and their morphisms commute, we see that every morphism of chain complexes $f : \mathcal{C}_* \to \mathcal{D}_*$ descends to a sequence of maps between their homology groups. We will denote each of these induced maps by $(f_n)_* : \mathcal{H}_n(\mathcal{C}) \to \mathcal{H}_n(\mathcal{D})$.

For the category-minded reader, we note that these considerations (by definition) imply that there actually exists a homology functor from the category of chain complexes to the category of abelian groups, $\mathcal{H}_n$: **Ch**(**Ab**) → **Ab**. This perspective allows us to prove that the homology is invariant under homotopy equivalences, as is done in e.g. [Sag21].

1.4.2. Exact Sequences of Chain Complexes

In this section, we introduce exact sequences, which are essential tools for relating (the homology of) different chain complexes to each other. Before we can define exact sequences for chain complexes, however, we must first define them for abelian groups.

Definition 1.19. Let $\{A_n\}_{n\in\mathbb{Z}}$ be a family of abelian groups, and consider homomorphisms $f_n : A_n \to A_{n-1}$. This can be represented as the following sequence:

$$\dots \xrightarrow{f_{n+2}} A_{n+1} \xrightarrow{f_{n+1}} A_n \xrightarrow{f_n} A_{n-1} \xrightarrow{f_{n-1}} A_{n-2} \xrightarrow{f_{n-2}} \dots$$

Such a sequence is called *exact at A_i* if $\ker(f_i) = \operatorname{Im}(f_{i+1})$. The sequence is called *exact* if it is exact at A_i for each $i \in \mathbb{Z}$.

Note that a chain complex $(\mathcal{C}_*, \partial_*)$ is exact at C_i if and only if its ith homology group $\mathcal{H}_i(\mathcal{C})$ is trivial.

We can distinguish two special types of exact sequences that will play an important role in our later work.

Definition 1.20. A *short exact sequence* (of abelian groups) is an exact sequence of the following form:

$$0 \longrightarrow A_3 \xrightarrow{f_3} A_2 \xrightarrow{f_2} A_1 \longrightarrow 0$$

A *long exact sequence* is an exact sequence of the following form:

$$\cdots \xrightarrow{f_5} A_4 \xrightarrow{f_4} A_3 \xrightarrow{f_3} A_2 \xrightarrow{f_2} A_1 \longrightarrow 0$$

While exact sequences are sequences of abelian groups, we can lift these notions to define exact sequences of chain complexes. As we will only be interested in short exact sequences of chain complexes, let us define this notion explicitly.

Definition 1.21. Let $\mathcal{C}'_*, \mathcal{C}_*$ and $\overline{\mathcal{C}}_*$ be chain complexes, and let $i : \mathcal{C}'_* \to \mathcal{C}_*$ and $p : \mathcal{C}_* \to \overline{\mathcal{C}}_*$ denote morphisms of chain complexes. We then say that the following sequence of chain complexes:

$$0 \longrightarrow \mathcal{C}'_* \xrightarrow{i} \mathcal{C}_* \xrightarrow{p} \overline{\mathcal{C}}_* \longrightarrow 0$$

is a *short exact sequence* if the chain complexes are level-wise exact, that is, if for each $n \in \mathbb{Z}$, the following sequence of abelian groups is exact:

$$0 \longrightarrow C'_n \xrightarrow{i_n} C_n \xrightarrow{p_n} \overline{C}_n \longrightarrow 0$$

Such short exact sequences of chain complexes are quite useful, as applying the homology functor to them yields a long exact sequence of homology groups, i.e.:

Theorem 1.5. *Given the following short exact sequence of chain complexes:*

$$0 \longrightarrow \mathcal{C}'_* \xrightarrow{i} \mathcal{C}_* \xrightarrow{p} \overline{\mathcal{C}}_* \longrightarrow 0$$

This sequence induces a long exact sequence of the following form:

$$\mathcal{H}_{n+1}(\overline{\mathcal{C}}) \xrightarrow{\delta_{n+1}} \mathcal{H}_n(\mathcal{C}') \xrightarrow{(i_n)_*} \mathcal{H}_n(\mathcal{C}) \xrightarrow{(p_n)_*} \mathcal{H}_n(\overline{\mathcal{C}}) \xrightarrow{\delta_n} \mathcal{H}_{n-1}(\mathcal{C}') \xrightarrow{(i_{n-1})_*} \mathcal{H}_{n-1}(\mathcal{C}) \longrightarrow \ldots \xrightarrow{(p_0)_*} \mathcal{H}_0(\overline{\mathcal{C}})$$

where the maps δ_n are called the connecting homomorphisms.

Proof. The proof of this theorem follows from diagram chasing and can be found in e.g. [Sag21]. □

In what is to come, we will abbreviate “short exact sequence” to SES, and similarly, “long exact sequence” to LES, and we will refer to this theorem as the “SES implies LES property”.

Before proceeding, we remark that this theorem is quite useful, as it, for example, tells us that there is always a long exact sequence relating the relative homology groups $\mathcal{H}_*(X, X'; A)$ to the homology groups of X and X'. Similarly, the differentials in the cellular complex of a CW-complex are defined as the connecting homomorphism that is induced by the short exact sequence of a triple of spaces.

Lastly, we want to point out one important corollary of this theorem, which allows us to calculate the homology of a space in terms of the homology of two of its (often simpler) subspaces, and which we will use (much) later in this work.

Theorem 1.6. *Given a topological space X and given two open sets $U_1, U_2 \subseteq X$ such that $U_1 \cup U_2 = X$. We have the following SES:*

$$0 \longrightarrow C_n(U_1 \cap U_2) \xrightarrow{(i_1)_* \oplus (i_2)_*} C_n(U_1) \oplus C_n(U_2) \xrightarrow{(p_1)_* - (p_2)_*} C_n(X) \longrightarrow 0$$

where $i_k : U_1 \cap U_2 \hookrightarrow U_k$ and $p_k : U_k \to X$ is the projection map (for $k = 1, 2$). This SES induces the following LES, which we call the Mayer-Vietoris Sequence:

$$\ldots \longrightarrow \mathcal{H}_{n+1}(X) \longrightarrow \mathcal{H}_n(U_1 \cap U_2) \longrightarrow \mathcal{H}_n(U_1) \oplus \mathcal{H}_n(U_2) \longrightarrow \mathcal{H}_n(X) \longrightarrow \mathcal{H}_{n-1}(X) \longrightarrow \ldots$$

1.4.3. Tensor Products of Chain Complexes and The Künneth Theorem

Given two topological spaces X and Y, the so-called *Eilenberg-Zilber* theorem relates the homology of their Cartesian product to the tensor product of their chain complexes (which we will soon define) [Hat01]:

Theorem 1.7. *Let X, Y be topological spaces. Then:*

$$\mathcal{H}_n(X \times Y; A) \simeq \mathcal{H}_n(\mathcal{C}_*(X;A) \otimes \mathcal{C}_*(Y;A)) \tag{1.26}$$

Taken together with the so-called *Künneth theorem* and requiring that A is a field, we obtain the following neat result for the homology of $X \times Y$:

$$\mathcal{H}_n(X \times Y; A) \simeq \bigoplus_{p=0}^{n} \mathcal{H}_n(X;A) \otimes \mathcal{H}_{n-p}(Y;A) \tag{1.27}$$

The goal of this section will be to prove (a slightly more general) version of the Künneth theorem. To do so, we will have to define the tensor product of chain complexes. Before we can do that, however, we will need the notion of a direct sum of chain complexes.

Definition 1.22. Let $\{(\mathcal{C}^i_*, \partial^i_*)\}_{i\in\mathcal{I}}$ be a family of chain complexes with index set $\mathcal{I}$. The *direct sum* of these chain complexes, $\left((\bigoplus_{i\in\mathcal{I}} \mathcal{C}^i)_*, \partial^{\oplus i}_*\right)$ is again a chain complex, and is defined as follows:

$$\left(\bigoplus_{i\in\mathcal{I}} C^i\right)_n = \bigoplus_{i\in\mathcal{I}} \left(C^i\right)_n \text{ with } \partial^{\oplus i}((a_i)_{i\in\mathcal{I}}) = (\partial^i(a_i))_{i\in\mathcal{I}} \tag{1.28}$$

We note that the direct sum is already useful on its own in the context of singular homology, as we have the following two results (see [Wei94, Sag21]). Firstly, for any indexed family of topological spaces X_i, we have that:

$$C_n(\sqcup_i X_i; A) = \bigoplus_i C_n(X_i; A) \tag{1.29}$$

and, moreover, given an indexed family of chain complexes $\mathcal{C}^i_*$, we have that:

$$\mathcal{H}_n(\bigoplus_i \mathcal{C}^i; A) = \bigoplus_i \mathcal{H}_n(\mathcal{C}^i; A) \tag{1.30}$$

Before we can define tensor products of chain complexes, we first have to introduce two more definitions.

Definition 1.23. A *double complex* $(\mathcal{B}_{**}, d^v_{**}, d^h_{**})$ is a family of abelian groups $B_{p,q}$ for $p, q \in \mathbb{Z}$ with so-called *horizontal differentials* $d^h_{p,q} : B_{p,q} \to B_{p,q-1}$ and *vertical differentials* $d^v_{p,q} : B_{p,q} \to B_{p-1,q}$ that anticommute, i.e.:

$$d^v_{p,q-1} d^h_{p,q} + d^h_{p-1,q} d^v_{p,q} = 0 \tag{1.31}$$

To every double complex, one can associate a chain complex called the *total complex*:

$$\text{Tot}(\mathcal{B}_{**})_n := \bigoplus_{p=0}^{n} B_{p,n-p} \tag{1.32}$$

One can represent a double complex as follows:

$$\begin{tikzcd}
 & \vdots \arrow[d] & \vdots \arrow[d] & \vdots \arrow[d] & \vdots \arrow[d] & \\
\cdots & B_{2,-1} \arrow[l] \arrow[d] & B_{2,0} \arrow[l] \arrow[d] & B_{2,1} \arrow[l] \arrow[d] & B_{2,2} \arrow[l, "d^h_{2,2}"'] \arrow[d, "d^v_{2,2}"] & \cdots \arrow[l] \\
\cdots & B_{1,-1} \arrow[l] \arrow[d] & B_{1,0} \arrow[l] \arrow[d] & B_{1,1} \arrow[l] \arrow[d] & B_{1,2} \arrow[l] \arrow[d] & \cdots \arrow[l] \\
\cdots & B_{0,-1} \arrow[l] \arrow[d] & B_{0,0} \arrow[l] \arrow[d] & B_{0,1} \arrow[l] \arrow[d] & B_{0,2} \arrow[l] \arrow[d] & \cdots \arrow[l] \\
\cdots & B_{-1,-1} \arrow[l] \arrow[d] & B_{-1,0} \arrow[l] \arrow[d] & B_{-1,1} \arrow[l] \arrow[d] & B_{-1,2} \arrow[l] \arrow[d] & \cdots \arrow[l] \\
 & \vdots & \vdots & \vdots & \vdots &
\end{tikzcd}$$

In order to define the tensor product of chain complexes, we note that one can construct a double complex by taking the term-wise tensor product of chain complexes. That is, given two chain complexes $(\mathcal{C}_*, \partial^C_*)$ and $(\mathcal{D}_*, \partial^D_*)$, we can define a double complex as follows:

$$B_{p,q} = C_p \otimes D_q \tag{1.33}$$

moreover:

$$d^h_{p,q} = \text{id} \otimes \partial^D_p, \quad d^v_{p,q} = \partial^C_p \otimes \text{id} \tag{1.34}$$

This newly obtained double complex can be visualised as follows:

$$\begin{tikzcd}
 & \vdots \arrow[d] & \vdots \arrow[d] & \vdots \arrow[d] & \vdots \arrow[d] & \\
\cdots & C_2 \otimes D_{-1} \arrow[l] \arrow[d] & C_2 \otimes D_0 \arrow[l] \arrow[d] & C_2 \otimes D_1 \arrow[l] \arrow[d] & C_2 \otimes D_2 \arrow[l, "\text{id}\otimes\partial^D_2"'] \arrow[d, "\partial^C_2\otimes\text{id}"] & \cdots \arrow[l] \\
\cdots & C_1 \otimes D_{-1} \arrow[l] \arrow[d] & C_1 \otimes D_0 \arrow[l] \arrow[d] & C_1 \otimes D_1 \arrow[l] \arrow[d] & C_1 \otimes D_2 \arrow[l] \arrow[d] & \cdots \arrow[l] \\
\cdots & C_0 \otimes D_{-1} \arrow[l] \arrow[d] & C_0 \otimes D_0 \arrow[l] \arrow[d] & C_0 \otimes D_1 \arrow[l] \arrow[d] & C_0 \otimes D_2 \arrow[l] \arrow[d] & \cdots \arrow[l] \\
\cdots & C_{-1} \otimes D_{-1} \arrow[l] \arrow[d] & C_{-1} \otimes D_0 \arrow[l] \arrow[d] & C_{-1} \otimes D_1 \arrow[l] \arrow[d] & C_{-1} \otimes D_2 \arrow[l] \arrow[d] & \cdots \arrow[l] \\
 & \vdots & \vdots & \vdots & \vdots &
\end{tikzcd}$$

The tensor product of these two chain complexes, now, is simply defined as the total complex of this double complex:

Definition 1.24. Let $(\mathcal{C}_*, \partial^C_*)$ and $(\mathcal{D}_*, \partial^D_*)$ be chain complexes of abelian groups. We define their *tensor product* $((\mathcal{C} \otimes \mathcal{D})_*, \partial^{C\otimes D}_*)$ as the chain complex satisfying:

$$(\mathcal{C} \otimes \mathcal{D})_n = \bigoplus_{p=0}^{n} C_p \otimes D_{n-p} \tag{1.35}$$

and whose differential $\partial^{C\otimes D}_*$ is the group homomorphism induced by the bilinear maps:

$$C_p \times D_{n-p} \to (C \otimes D)_{n-1} : (x, y) \mapsto \partial^C_p(x) \otimes y + (-1)^p x \otimes \partial^D_{n-p}(y) \tag{1.36}$$

We recall that the Cartesian product of two finite CW-complexes is again a CW-complex, and that its n cells are given by the products of p-cells and $n-p$ cells for $p = 0, \dots, n$. This gives us a natural interpretation for the notion of the tensor product of a chain complex: if X and Y are finite CW-complexes, the following holds:

$$\tilde{C}_*(X \times Y) \simeq (\tilde{C}_*(X) \otimes \tilde{C}_*(Y))_* \tag{1.37}$$

In summary, in the context of finite CW-complexes, we see that the notions of tensor products of cellular complexes and Cartesian products of spaces coincide.

We now turn to the goal of this section, which was to prove the Künneth theorem. This theorem allows us to relate the homology groups of the tensor product of chain complexes to the tensor product of their respective homology groups. Phrased in the general setting of chain complexes of abelian groups, however, this relationship is quite delicate, and even more advanced mathematical machinery than we have seen thus far is needed to state it. We will introduce this more advanced machinery in Chapter 4 — for now, we restrict ourselves to the case in which our chain complexes are all chain complexes of vector spaces, as this will be the setting of Chapter 2 and Chapter 3. Now, fortunately, the Künneth theorem takes on quite an elegant form [Hat01]:

Theorem 1.8 (Künneth Formula)**.** *Let $(\mathcal{C}_*, \partial_*^C)$ and $(\mathcal{D}_*, \partial_*^D)$ be chain complexes of vector spaces. Then:*

$$\mathcal{H}_n(\mathcal{C} \otimes \mathcal{D}) \simeq \bigoplus_{p=0}^{n} \mathcal{H}_p(\mathcal{C}) \otimes \mathcal{H}_{n-p}(\mathcal{D}) \tag{1.38}$$

Proof. We consider the subspaces $Z_n = \ker(\partial_n^C)$ and $B_n = \operatorname{Im}(\partial_{n+1}^C)$. Note that $\mathcal{Z} := (Z_*, \partial_*^C)$ is a subcomplex of $\mathcal{C}$, with $\partial_n^C|_{Z_n} = 0$. As $B_n \subseteq Z_n$, $\mathcal{B} = (B_n, \partial_n^C)$ is also a subcomplex of $\mathcal{C}$. As such, these subspaces yield the following SES:

$$\begin{tikzcd} 0 \arrow[r] & (\mathcal{Z} \otimes \mathcal{D})_n \arrow[r, hook] & (\mathcal{C} \otimes \mathcal{D})_n \arrow[r, "\partial_n^C \otimes \mathrm{id}"] & (\mathcal{B} \otimes \mathcal{D})_{n-1} \arrow[r] & 0 \end{tikzcd}$$

where the map $(\mathcal{Z} \otimes \mathcal{D})_n \hookrightarrow (\mathcal{C} \otimes \mathcal{D})_n$ is the inclusion map.

Given that $B_n \subseteq Z_n$ are in the kernel of ∂_n^C, the following diagram commutes:

$$\begin{tikzcd} 0 \arrow[r] & (\mathcal{Z} \otimes \mathcal{D})_n \arrow[r, hook] \arrow[d, "(-1)^p \mathrm{id} \otimes \partial^D"] & (\mathcal{C} \otimes \mathcal{D})_n \arrow[r, "\partial_n^C \otimes \mathrm{id}"] \arrow[d, "\partial^C \otimes \mathrm{id} + (-1)^p \mathrm{id} \otimes \partial^D"] & (\mathcal{B} \otimes \mathcal{D})_{n-1} \arrow[r] \arrow[d, "(-1)^p \mathrm{id} \otimes \partial^D"] & 0 \\ 0 \arrow[r] & (\mathcal{Z} \otimes \mathcal{D})_{n-1} \arrow[r, hook] & (\mathcal{C} \otimes \mathcal{D})_{n-1} \arrow[r, "\partial^C \otimes \mathrm{id}"] & (\mathcal{B} \otimes \mathcal{D})_{n-2} \arrow[r] & 0 \end{tikzcd}$$

Hence, we see that the inclusion maps, along with the maps $\partial_*^C \otimes \mathrm{id}$, are actually chain maps, and as such, we have the following SES of chain complexes:

$$\begin{tikzcd} 0 \arrow[r] & (\mathcal{Z} \otimes \mathcal{D})_* \arrow[r, hook] & (\mathcal{C} \otimes \mathcal{D})_* \arrow[r, "\partial_*^C \otimes \mathrm{id}"] & (\mathcal{B} \otimes \mathcal{D})_{(*-1)} \arrow[r] & 0 \end{tikzcd}$$

This SES now induces the following LES:

$$\begin{tikzcd} \dots \arrow[r] & \mathcal{H}_n(\mathcal{B} \otimes \mathcal{D}) \arrow[r, "\delta_{n+1}"] & \mathcal{H}_n(\mathcal{Z} \otimes \mathcal{D}) \arrow[r] & \mathcal{H}_n(\mathcal{C} \otimes \mathcal{D}) \arrow[r] & \mathcal{H}_{n-1}(\mathcal{B} \otimes \mathcal{D}) \arrow[r, "\delta_n"] & \mathcal{H}_{n-1}(\mathcal{Z} \otimes \mathcal{D}) \arrow[r] & \dots \end{tikzcd}$$

Our goal, now, is to deduce a SES from this LES. We proceed in three steps: we first determine the connecting homomorphism δ_n. Afterwards, we will use this to deduce a right and left-exact sequence from the LES, respectively.

In order to determine δ_{n+1}, we first note that as $B_n \subseteq Z_n \subseteq C_n$, there is actually a sequence of inclusion maps:

$$(\mathcal{B} \otimes \mathcal{D})_n \hookrightarrow (\mathcal{Z} \otimes \mathcal{D})_n \hookrightarrow (\mathcal{C} \otimes \mathcal{D})_n \tag{1.39}$$

We thus claim that any class $[x] \in \mathcal{H}_n(\mathcal{B} \otimes \mathcal{D})$ is mapped to the class $[x] \in \mathcal{H}_n(\mathcal{Z} \otimes \mathcal{D})$ by the connecting homomorphism, that is, we claim for any class in $\mathcal{H}_n(\mathcal{B} \otimes \mathcal{D})$, one can determine the image of this class under the connecting homomorphism by simple taking a representative of this class and returning the class that it represents in $\mathcal{H}_n(\mathcal{Z} \otimes \mathcal{D})$.

For our claim to be true, x, when seen as an element of $(\mathcal{Z}\otimes\mathcal{D})_n$, must satisfy the following requirement: given any $y \in (\mathcal{C} \otimes \mathcal{D})_{n+1}$ such that $x = (\partial^C_n \otimes \mathrm{id})(y)$, then $x + \mathrm{Im}\left((-1)^p \mathrm{id} \otimes \partial^D_{n+1}\right)$ must be in the $(\partial^C_{n+1} \otimes \mathrm{id} + (-1)^p \mathrm{id} \otimes \partial^D_{n+1})$-image of y. This is indeed true, as:

$$(\partial^C_{n+1} \otimes \mathrm{id} + (-1)^p \mathrm{id} \otimes \partial^D_{n+1})y = x + (-1)^p \mathrm{id} \otimes \partial^D_{n+1} y \tag{1.40}$$

Hence, we see that δ_n is induced by the inclusion $B_n \hookrightarrow Z_n$, and is, furthermore, injective.

We can now use the injectivity of the connecting homomorphism to deduce a right-exact sequence from the LES: as the homomorphism $H_{n-1}(\mathcal{B} \otimes \mathcal{D}) \to H_{n-1}(\mathcal{Z} \otimes \mathcal{D})$ is injective, it has a trivial kernel, and hence, by exactness, the map before it must have a trivial image. Restricting to the image of that map, we find the following right-exact LES:

$$\begin{tikzcd}
\dots \arrow[r] & H_n(\mathcal{B} \otimes \mathcal{D}) \arrow[r, "\delta_*"] & H_n(\mathcal{Z} \otimes \mathcal{D}) \arrow[r] & H_n(\mathcal{C} \otimes \mathcal{D}) \arrow[r] & 0
\end{tikzcd}$$

In order to extract a left-exact sequence from this right-exact sequence, we first show that there is a natural isomorphism $\mathcal{H}_n(\mathcal{Z}_* \otimes \mathcal{D}_*) \simeq (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n$. Indeed:

$$\begin{aligned}
\mathcal{H}_n(\mathcal{Z}_* \otimes \mathcal{D}_*) = \ker\left(\partial^{Z\otimes D}_n\right)/\mathrm{Im}\left(\partial^{Z\otimes D}_{n+1}\right) &= \left(\bigoplus_{p=0}^{n} Z_p \otimes \ker\left(\partial^D_{n-p}\right)\right) / \left(\bigoplus_{p=0}^{n} Z_p \otimes \mathrm{Im}\left(\partial^D_{n+1-p}\right)\right) && (1.41)\\
&\simeq \bigoplus_{p=0}^{n} \left(Z_p \otimes \ker\left(\partial^D_{n-p}\right)\right) / \left(Z_p \otimes \mathrm{Im}\left(\partial^D_{n+1-p}\right)\right) && (1.42)\\
&\simeq \bigoplus_{p=0}^{n} Z_p \otimes \left(\ker\left(\partial^D_{n-p}\right)/\mathrm{Im}\left(\partial^D_{n+1-p}\right)\right) && (1.43)\\
&= \bigoplus_{p=0}^{n} Z_p \otimes \mathcal{H}_{n-p}(\mathcal{D}) && (1.44)\\
&= (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n && (1.45)
\end{aligned}$$

where the first isomorphism follows from the fact that $Z_p \otimes \ker\left(\partial^D_{n-p}\right) \subseteq Z_p \otimes \mathrm{Im}\left(\partial^D_{n+1-p}\right)$, and the second isomorphism follows from the exactness of the tensor product between vector spaces.
Similarly, it follows that $\mathcal{H}_n(\mathcal{B}_* \otimes \mathcal{D}_*) \simeq (\mathcal{B}_* \otimes \mathcal{H}_*(\mathcal{D}))_n$. But then, by the naturality of these isomorphisms, it can easily be seen that the following diagram commutes:

$$\begin{tikzcd}
\mathcal{H}_n(\mathcal{B}_* \otimes \mathcal{D}_*) \arrow[r, hook] \arrow[d, "\simeq"] & \mathcal{H}_n(\mathcal{Z}_* \otimes \mathcal{D}_*) \arrow[d, "\simeq"'] \\
(\mathcal{B}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[r, hook] & (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n
\end{tikzcd}$$

where the upper arrow is the injection induced by the inclusion. Hence, we can deduce the following commutative diagram:

$$\begin{tikzcd}
\dots \arrow[r] & \mathcal{H}_n(\mathcal{B} \otimes \mathcal{D}) \arrow[r, "\delta_*"] \arrow[d, "\simeq"] & \mathcal{H}_n(\mathcal{Z} \otimes \mathcal{D}) \arrow[r] \arrow[d, "\simeq"'] & \mathcal{H}_n(\mathcal{C} \otimes \mathcal{D}) \arrow[r] & 0 \\
0 \arrow[r] & (\mathcal{B}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[r, hook] & (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[ur, dashed] & &
\end{tikzcd}$$

We have thus obtained the desired SES (in black). Now, as the kernel of the map $(\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \to \mathcal{H}_n(\mathcal{C} \otimes \mathcal{D})$ is equal to the image of the inclusion map, we see that taking quotients induces an injective map:

$$\begin{tikzcd}
\cdots \arrow[r] & (\mathcal{B} \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[r, hook] & (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[r] \arrow[d] & \mathcal{H}_n(\mathcal{C} \otimes \mathcal{D}) \arrow[r] & 0 \\
& 0 \arrow[r] & (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n / (\mathcal{B} \otimes \mathcal{H}_*(\mathcal{D}))_n \arrow[ur, dashed] \arrow[d] & & \\
& & 0 & &
\end{tikzcd}$$

We have therefore found the desired isomorphism:

$$\mathcal{H}_n(\mathcal{C} \otimes \mathcal{D}) \simeq (\mathcal{Z}_* \otimes \mathcal{H}_*(\mathcal{D}))_n / (\mathcal{B}_* \otimes \mathcal{H}_*(\mathcal{D}))_n \simeq \bigoplus_{p=0}^{n} Z_p/B_p \otimes \mathcal{H}_{n-p}(\mathcal{D}) = \bigoplus_{p=0}^{n} \mathcal{H}_p(\mathcal{C}) \otimes \mathcal{H}_{n-p}(\mathcal{D}) \tag{1.46}$$

□

1.4.4. The Landscape of Homological Algebra

We dedicate a few words to the observation that we have thus far already seen two different homology theories: singular homology and cellular homology. There are, in fact, many more homology theories, like group homology (see e.g. [Wei94]), but also homology with local coefficients (see e.g. [DK01]) or homology over ring modules (see e.g. [Wei94]), the latter two of which we will discuss in Chapter 7 and Chapter 4, respectively. One can unite these different theories in one unifying framework by taking on an axiomatic approach to homology, as was first set out in [ES45]. We mention this fact, because some of the results we derived here, like the Mayer-Vietoris sequence, can actually be derived from these axioms alone [Hat01], and therefore can be used regardless of which specific homology theory is considered.

Furthermore, the categorically oriented reader might not be surprised to read that there is also a theory of *cohomology*. Here, the primary objects of study are cochain complexes, which are similar to chain complexes, except for the fact that the direction of their arrows is flipped. When considering e.g. chain complexes consisting of finite dimensional inner product spaces, this may seem like a mere notational difference (as one can always switch between a chain complex and a cochain complex by taking the dual), yet, in more general settings, the theory of cohomology offers additional richness than the theory of homology. For the purpose of our study, however, we will only be interested in the structure that can be captured using just homology, and therefore, we refer the interested reader to [SH22].

2 Quantum LDPC Codes

Quantum Low-Density Parity Check (qLDPC) codes are a subclass of the so-called Calderbank-Shor-Steane (CSS) codes, which are again a subclass of the stabiliser codes. We therefore first review the stabiliser formalism. Afterwards, we present the definitions of CSS and qLDPC codes, and we discuss how CSS codes can be understood in terms of homology. Lastly, we take a look ahead by considering how (good) qLDPC codes could be constructed.

2.1. The Stabiliser Formalism

The stabiliser formalism [Ter15, Bro14] allows us to understand a large class of quantum error correcting codes, the so-called *stabiliser codes*, in terms of Pauli operators instead of state vectors. Stabiliser codes are defined by two ingredients: a so-called *stabiliser group*, and a set of *logical operators*. In order to define these, we first introduce the Pauli group for an n-qubit system: the *Pauli group* $\mathcal{P}_n$ is the group generated by the Pauli operators X_i and Z_i for each $i \in \{1, \dots, n\}$, as well as the operator iI — that is:

$$\mathcal{P}_n := \langle iI, X_1, Z_1, X_2, \dots, Z_n \rangle \tag{2.1}$$

The *stabiliser group* $\mathcal{S}$, now, is any subgroup of the Pauli group that does not contain $-I$. Choosing the stabiliser group fixes the code space of the stabiliser code at hand, as we define this code space to be the simultaneous $+1$ eigenspace of all the elements of the stabiliser group, i.e.:

$$\mathcal{C} := \{|\psi\rangle : S|\psi\rangle = |\psi\rangle \quad \forall S \in \mathcal{S}\} \tag{2.2}$$

Given that $\mathcal{S}$ has $n-k$ independent generators, i.e. $\mathcal{S} = \langle S_1, \dots, S_{n-k} \rangle$, one can deduce that the dimension of the code space is given by $n-(n-k) = k$.

We must now still determine the final ingredient: the logical operators. Given that $\mathcal{S}$ has $n-k$ independent generators, one can still choose k pairs of elements of the Pauli group, which we denote by $(\overline{X}_i, \overline{Z}_i)$, that commute with all of $\mathcal{S}$ and with all other pairs, but whose elements anticommute. We call these operators the *logical operators*, and note for completeness that the centraliser of the stabiliser group can be generated by the stabilisers and the logical operators, i.e.:

$$C(\mathcal{S}) = \langle S_1, \dots, S_{n-k}, \overline{X}_1, \overline{Z}_1, \dots, \overline{Z}_k \rangle \tag{2.3}$$

We defining the *weight* of a Pauli operator L, $|L|$, to be the number of qubits on which the operator acts non-trivially. The *(minimum) distance* of a code, d, is then defined as the minimum weight of all possible logical operators of the code, i.e.:

$$d := \min_{L \in C(\mathcal{S}) \setminus \mathcal{S}} |L| \tag{2.4}$$

We usually denote a stabiliser code that encodes k logical qubits into an n-qubit system with distance d by $[[n, k, d]]$.

An exemplar of stabiliser codes is the toric code. Given an $L \times L$ lattice with periodic boundary conditions, we can associate qubits to each of its edges, as is shown in Figure 2.1.

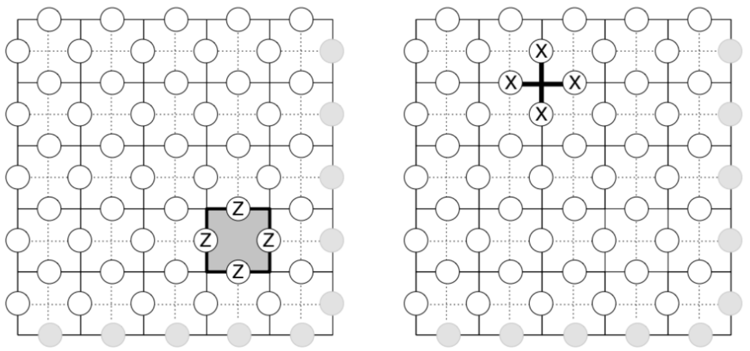


Figure 2.1: Source: [Bro14]. A square (5 by 5) lattice with periodic boundary conditions encoding the toric code. The grey edges are identified with the edges on the opposite side. The white circles represent qubits, while the Z-checks correspond to plaquettes (left), and the X-checks correspond to the vertices (right).

As Figure 2.1 illustrates, we can define Z-stabilisers for every plaquette on the lattice, i.e. by taking the operator that acts as a Z on each of the qubits bordering the plaquette, and by acting as the identity operator on the other qubits. Similarly, one can define X-stabilisers for every vertex. Note that these commute, as they always overlap in an even number of qubits.

One can now identify the following minimum weight logical operators as the operators corresponding to closed horizontal and vertical lops on the lattice, as is illustrated in Figure 2.2.

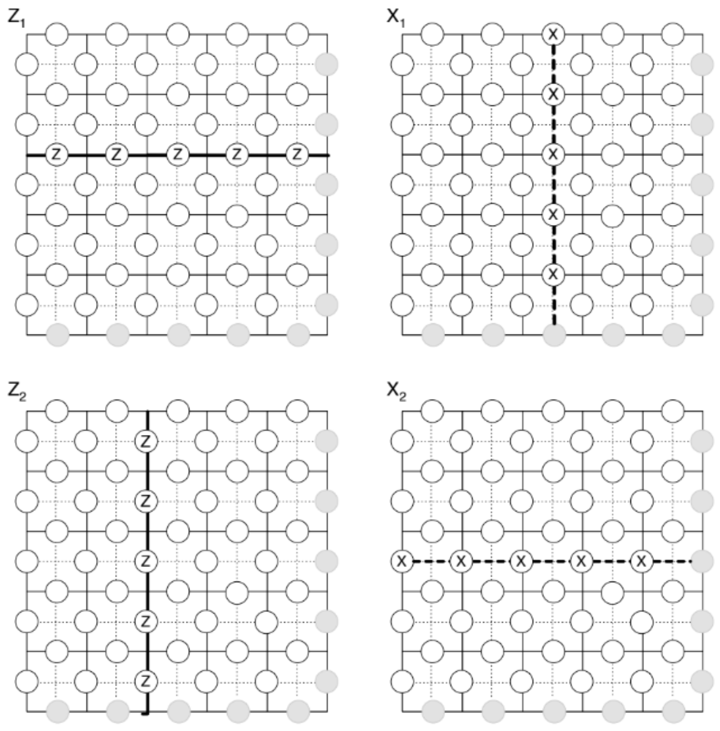


Figure 2.2: Source: [Bro14]. Four examples of independent logical operators for the toric code encoded on a square (5 by 5) lattice with periodic boundary conditions. One can see that the $\overline{Z}_1$-logical operator ($\overline{Z}_2$) and the $\overline{X}_1$-logical operator ($\overline{X}_2$) anticommute, as they overlap in one qubit. Each of these logical operators has a weight of 5.

As there are 2 pairs of independent logical operators, we see that the number of encoded qubits is

$k = 2$. Furthermore, as the weight of the minimum weight logical operators is L, we deduce that the toric code has minimum distance $d = L$ (which is indeed the case, see e.g. [Kit03]).

2.2. CSS and qLDPC Codes

CSS codes are particular stabiliser codes that are generated by two linear classical codes. It is therefore instructive to first recall some basic facts and definitions from classical coding theory.

2.2.1. Basic Facts from Classical Coding Theory

Recall that each binary, linear classical code $\mathcal{C}$ that encodes k (logical) bits into n (physical) bits can be specified by a so-called *parity check matrix* $H \in \mathcal{M}_{r\times n}(\mathbb{F}_2)$ with $n-k$ independent rows, such that $r \geq n-k$. The defining property of a parity check matrix is that a word $c \in \mathbb{F}_2^n$ is a *code word* of the code $\mathcal{C}$ if and only if $Hc = 0$. Therefore, we can identify $\mathcal{C}$ with the kernel of H, $\ker(H)$.

Analogously, every binary, linear classical code that encodes k (logical) bits into n (physical) bits can be specified by a *generator matrix* $G \in \mathcal{M}_{s\times n}(\mathbb{F}_2)$ with k independent rows, such that $s \geq k$. The defining property of a generator matrix of a code $\mathcal{C}$ is that its row space equals the space of codewords of the code, that is: $\text{Row}(G) = \mathcal{C}$.

Furthermore, every classical code has a *minimum distance*, d, which is defined as the minimum weight of its non-zero codewords, i.e.:

$$d := \min_{c\in\ker(H)\setminus\{0\}} |c| \tag{2.5}$$

where $|c| := \sum_{i=1}^{n} c_i$ is the *weight* of the code word c.

We note that every classical code is fully characterised by the size of its code space, n, the number of bits it encodes, k, and its minimum distance, d. We therefore often refer to a classical code using the notation $[n, k, d]$.

Finally, for every such code $\mathcal{C}$, one can define its *dual code*, $\mathcal{C}^\perp$, as the code with code words $c^\perp \in \mathbb{F}_2^{n-k}$ satisfying the following relation[1]:

$$\langle c^\perp, c\rangle := \sum_{i=1}^{n-k} c_i^\perp c_i = 0 \pmod 2 \qquad \forall\, c \in \mathcal{C} \tag{2.6}$$

Equivalently, every parity check matrix of $\mathcal{C}$ is a generator matrix of $\mathcal{C}^\perp$ (and vice versa).

2.2.2. CSS codes

Having recalled these definitions from classical coding theory, are finally ready to introduce the definition of a CSS code.

Definition 2.1 (CSS code). Let $\mathcal{C}_X$ and $\mathcal{C}_Z$ be two binary, linear codes with parity check matrices $H_X \in \mathcal{M}_{r_X\times n}(\mathbb{F}_2)$ and $H_Z \in \mathcal{M}_{r_Z\times n}(\mathbb{F}_2)$, and parameters $[n, k_X, d_1]$ and $[n, k_Z, d_2]$, respectively, such that $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$. The pair $(\mathcal{C}_X, \mathcal{C}_Z)$ generates an $[[n, k, d]]$-stabiliser code, whose parameters are given by:

$$k = k_x + k_Z - n, \quad d = \min\{d_X, d_Z\} \tag{2.7}$$

Here, we have that d_X and d_Z are given by:

$$d_Z = \min_{x\in\ker(H_X)\setminus\text{Im}(H_Z^T)} |x|, \quad d_X = \min_{x\in\ker(H_Z)\setminus\text{Im}(H_X^T)} |x| \tag{2.8}$$

The X-stabilisers (Z-stabilisers) of the code are generated by the rows of H_X (H_Z) and are thus of the following form:

$$\Pi_{j=1}^{n} X_j^{(H_X)_{ij}} \qquad \text{for each } i \in \{1, \dots, n\} \tag{2.9}$$

Such a stabiliser code is called the *CSS code generated by the pair* $(\mathcal{C}_X, \mathcal{C}_Z)$.

[1]For the mathematically oriented reader, we emphasise that this is a *bilinear form* on the vector space $\mathbb{F}_2$.

Before proceeding to an example, let us unwrap this definition: firstly, we remark that the requirement $\mathcal{C}_X^{\perp} \subseteq \mathcal{C}_Z$ is equivalent to the requirement that $H_Z H_X^T = \mathbf{0}$. This requirement on the two classical codes is necessary, because it ensures that the stabiliser checks commute. Indeed, we see that:

$$\left(H_Z H_X^T\right)_{ij} = 0 \iff \sum_{m=1}^{n} (H_Z)_{im} \left(H_X^T\right)_{mj} = (H_Z)_{im} (H_X)_{jm} = 0 \pmod 2 \tag{2.10}$$

In words, this boils down to requiring that every row of H_X be dual to every row of H_Z, or, equivalently, that the number of qubits that any Z-stabiliser and any X-stabiliser simultaneously act on is *even*. Therefore, we see that the product of two such stabilisers contains an even number of terms XZ-terms. Under commutation, each such term contributes a factor -1. As there is an even number of these terms, we see that the product actually commutes.

Furthermore, we see that the orthogonality condition gives us a convenient way of determining the distance of a CSS code. Indeed, the distance of the code is the minimum weight of non-stabiliser elements in the centraliser of the stabiliser group. Furthermore, an element of the centraliser is — per definition — an element that commutes with both the X and the Z stabilisers. To this end, we note that we can decompose every string of Paulis, L, into an X-string, L_X, and a Z-string, L_Z. Therefore, we see that the requirement of being in the centraliser corresponds to the requirement that L_X commutes with all Z-stabilisers, and L_Z commutes with all X-stabilisers. But this is equivalent to requiring that L_X lies in ker(H_Z), and L_Z lies in ker(H_X). The weight of L is always greater than the weight of L_X and L_Z, hence, the lowest possible weight for L is the minimum of the lowest possible weights for L_X and L_Z. However, as the stabilisers correspond to Row(H_X) = Im$\left(H_X^T\right)$ and Row(H_Z) = Im$\left(H_Z^T\right)$, we have to exclude the possibility of L_X, L_Z lying in these. We then see that the minimum weight of L_X corresponds to d_X in our definition, while the minimum weight of L_Z corresponds to d_Z.

Lastly, we can easily see that the expression for the number of encoded qubits holds true by counting: there are n qubits, $n - k_X$ independent X-stabilisers, and $n - k_Z$ independent Z-stabilisers, so there number of encoded qubits is:

$$k = n - (n - k_X) - (n - k_Z) = k_X + k_Z - n \tag{2.11}$$

Example 2.1.1. The [[7,1,3]] *Steane code* can be seen as the CSS code generated by taking C_X, C_Z to both be the [7,4,3] Hamming code. Indeed, the [7,4,3] code has parity check matrix:

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{2.12}$$

And hence, we see that the Z-check matrices are $Z_1Z_3Z_5Z_7$, $Z_2Z_3Z_6Z_7$ and $Z_4Z_5Z_6Z_7$, as expected.

One may wonder whether considering the class of CSS codes might be too restrictive. This is, fortunately, not the case: it has been proved that every $[[n,k,d]]$ stabiliser code can be mapped onto a $[[4n,2k,2d]]$ CSS code [BTL10].

2.2.3. qLDPC Codes and Good Codes

In classical coding theory, a low-density parity check code is a family of codes for which both the number of bits involved in each check, and the number of checks a bit is involved in, is bounded by a constant for the whole code family. As each parity check corresponds to a row in the parity check matrix, the number of bits involved in a check is the number of 1's in the corresponding row. Similarly, each column in the parity check matrix represents a bit, and hence, the number of 1's in each column represents the number of parity checks performed on each bit. Therefore, one can introduce the following quantity in order to understand classical LDPC codes:

Definition 2.2. Let H be a parity check matrix of the code $\mathcal{C}$. The *Hamming weight* of the code, $w(H)$ is defined as the maximum of the number of non-zero elements in the rows and the columns of H.

We say that a family of classical codes is *LDPC* if the weight of the code family is asymptotically bounded by a constant, i.e. if $w(H) = \mathcal{O}(1)$. A simple family of LDPC codes are the repetition codes: for every n, we have that $w(R_n) = 2$.

As a CSS code is defined by two classical codes, we can easily generalise this definition to the quantum case: given a CSS code $\mathcal{C}$ generated by the parity check matrices H_X and H_Z, we define the weight of the CSS code to be:

$$w(\mathcal{C}) = \max\{w(H_X), w(H_Z)\} \tag{2.13}$$

In turn, a family of codes $\mathcal{C}$ is said to be a qLDPC code if $w(\mathcal{C}) = \mathcal{O}(1)$.

Being LDPC does not necessarily imply that a family of codes performs well, as it could be the case that this family encodes few qubits, or that the distance scales poorly with the number of qubits (like the toric code, which is an LDPC code, but has parameters $[[n, 2, \sqrt{n}]]$). Therefore, we introduce the notion of a *good* code.

Definition 2.3. A $[[n, k, d]]$-code is called *good* if it has a constant encoding rate $\frac{k}{n} = \Omega(1)$ and linear distance $d = \Omega(n)$.

2.3. Error Correcting Codes and Homology

We now delve into the question of how one can use the language of homology to understand error correcting codes.

2.3.1. A Homological Interpretation of Classical Codes

We start off this journey by making the (somewhat trivial) observation that every classical code (with parity check matrix H) can be represented by a chain complex $\mathcal{C}_*$ of $\mathbb{F}_2$-vector spaces of the following form:

$$\begin{tikzcd} C_1 \arrow[r, "H"] & C_0 \end{tikzcd}$$

and, conversely, every chain complex of this form corresponds to a classical code.

Let us consider the first homology group of this chain complex. We find:

$$\mathcal{H}_1(\mathcal{C}) = \ker(H) \tag{2.14}$$

Hence, we see that the number of encoded bits k can be defined purely in homological terms, namely as $k := \dim(\mathcal{H}_1(\mathcal{C}))$.

Moreover, such a chain complex can always be related to a hypergraph, simply by taking the parity check matrix to represent a hyperedge-vertex incidence matrix. For example, the repetition-5 code R_5 can be related to the following graph:

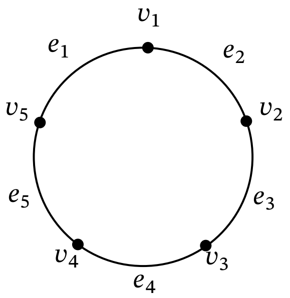


Figure 2.3: The graph associated to the 5-repetition code by interpreting its parity check matrix as a edge-vertex incidence matrix. One can see that this graph corresponds to a cellular complex of the circle S^1.

This graph, now, is actually a cellulation of the circle! Moreover, this conclusion holds for the whole family of repetition codes, not just for R_5: each R_n arises from a cellulation of a circle with n 1-cells and

n 0-cells. This yields some important insights: firstly, as the homology of a space is invariant under the choice of cellulation, we can immediately conclude that the repetition code encodes the same number of bits, regardless of the code size. Secondly, we see that the code distance is *not* a homological property: the distance of the repetition code R_n is n, and hence, is not invariant under changes in the cellulation of the circle.

2.3.2. A Homological Interpretation of CSS Codes

Naturally, one may wonder whether we can also represent quantum codes as a chain complex. As all length 1 complexes correspond to classical code, one would need a complex of (at least) length two. For such complexes, however, we will need to proceed more cautiously, as the condition on the boundary maps ($\partial^2 = 0$) is no longer automatically satisfied. The attentive reader will have noticed that this condition is reminiscent of the orthogonality condition for CSS codes ($H_Z H_X^T = 0$). Indeed, given a CSS code with parity check matrices H_X and H_Z, the orthogonality condition allows us to represent this code as a chain complex $\mathcal{C}$ of $\mathbb{F}_2$-vector spaces:

$$\begin{tikzcd} C_2 \arrow[r, "\partial_2 = H_X^T"] & C_1 \arrow[r, "\partial_1 = H_Z"] & C_0 \end{tikzcd}$$

Given that we understand how to use homology for the purpose of analysing classical codes, let us try to apply these same techniques to CSS codes. To this end, let us first calculate the first homology group:

$$\mathcal{H}_1(\mathcal{C}) = \ker(H_X)/\operatorname{Im}\left(H_Z^T\right). \tag{2.15}$$

The dimension of the first homology group is:

$$\dim(\mathcal{H}_1(\mathcal{C})) = \dim(\ker(H_X)) - \dim\left(\operatorname{Im}\left(H_Z^T\right)\right) = k_X - \dim\left(\ker(H_Z)^\perp\right) = k_X + k_Z - n \tag{2.16}$$

which is the number of encoded qubits of the CSS code! Note that the converse also holds true: given a 2-chain complex with first homology group of dimension k, then we can associate to this chain complex a quantum CSS code that encodes k qubits. Even stronger: given *any* n-complex with $n \geq 2$, say:

$$\begin{tikzcd} 0 \arrow[r] & C_n \arrow[r] & C_{n-1} \arrow[r] & \cdots \arrow[r] & C_0 \end{tikzcd}$$

We can always take a subcomplex of non-zero chains, say:

$$\begin{tikzcd} C_k \arrow[r] & C_{k-1} \arrow[r] & C_{k-2} \end{tikzcd}$$

And interpret this complex as a CSS-code. In this case, the $(k-1)^{\text{th}}$ homology group of our initial complex gives the number of encoded qubits.

As in the classical case, the distance of a quantum CSS code is not fixed by the homology of the underlying space. We can, however, phrase it in terms of these homological notions. We saw earlier that:

$$d_X = \min_{x \in \ker(H_Z) \setminus \operatorname{Im}\left(H_X^T\right)} |x| \tag{2.17}$$

Recalling that the first homology group is the quotient $\ker(H_Z)/\operatorname{Im}\left(H_X^T\right)$, we see that the minimum weight of the X-logicals can be interpreted as the minimum weight of the representatives of all the trivial classes in the first homology group. Similarly, we can flip the arrows in the complex to obtain a new chain complex[2]:

$$\begin{tikzcd} C_0 \arrow[r, "H_Z^T"] & C_1 \arrow[r, "H_X"] & C_2 \end{tikzcd}$$

We now see that the minimum weight of non-trivial elements of the first homology group of this newly obtained complex[3] corresponds to the minimum weight of the Z-logicals, d_Z.

[2]For completeness, we note that this is actually the cocomplex of our complex, however, when considering complexes of vector spaces, these notions can be identified in a natural manner.

[3]For completeness, we note that this is actually the first *cohomology group* of the complex corresponding to the CSS code. As we are working over vector spaces, however, these distinctions are merely formal.

Finally, as mentioned in the previous chapter, every finite dimensional chain complex of this form can actually be interpreted as a CW-complex, and hence as a tesselation of some topological space. Since we can relate quantum CSS codes to chain complexes, we can therefore also relate them to topological spaces. In the case of the toric code, for example, one can interpret the 2-chains as the plaquettes of the underlying lattice, while the 1-chains correspond to the edges, and the 0-chains correspond to the vertices. We will return to this example later in this thesis in order to explore this connection more thoroughly.

2.4. Constructing qLDPC codes

Let us briefly consider how one could construct a family of qLDPC codes. A simple idea would be to concatenate codes. This is not a very fruitful exercise, however: consider a $[[n, 1, d]]$ with weight w, and suppose that we concatenate this code with itself repeatedly to generate a family of larger codes. The parameters of the k-times concatenated code become $[[n^k, 1, d^k]]$ but the weight is also w^k. Therefore, given that the weight is larger than 1 (which is a necessary requirement for a non-trivial code), the generated code family cannot have asymptotically bounded weight, and hence cannot be qLDPC.

We are therefore led to consider more complex ways of constructing qLDPC codes. Given the fact that classical LDPC codes are understood very well and that good classical LDPC codes can be generated quite easily, it is instructive to consider whether we can somehow construct CSS codes out of classical codes. Surely, one can simply take the two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ to be LDPC, however, these codes cannot be simultaneously good: the orthogonality condition $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$, together with the fact that the rows of H_X are in $C_X^\perp$, implies that $\mathcal{C}_Z$ contains rows of H_X. But as these codes are LDPC, the rows of H_X are of bounded weight, and hence, the distance of $\mathcal{C}_Z$ is $\mathcal{O}(1)$.

Therefore, one will need to find more complex ways of combining good classical LDPC codes in order to construct good qLDPC codes. The homological interpretation of these codes suggests that one can seek an answer to this question by looking into ways in which one can combine two chain complexes (or two topological spaces) to form a third, larger chain complex.

3 Products of Codes and Tensor Products of Chain Complexes

In this section, we review three product constructions of CSS codes: the hypergraph product, the distance balancing procedure, which we shall dub the EKZ-product, and their generalisation.

3.1. Hypergraph Product Codes

The hypergraph product code construction, which was devised by Tillich and Zémor [TZ09], was actually first constructed in a non-homological way. We therefore begin by presenting the initial construction, and afterwards show how it can be understood much more naturally in the language of homology.

3.1.1. The Tanner Graph Approach: The Construction

There is a natural way to associate graphs to parity check matrices: one simply creates vertices for each row (that is, for each check) and for each column (that is, for each physical qubit), and adds an edge between a row vertex and a column vertex if the element of the parity check matrix corresponding to that row and that column is 1 (so, if the qubit is involved in the check). Proceeding in this way, one finds a bipartite graph, which is called the Tanner graph. We can formalise this construction as follows:

Definition 3.1. Given a classical code with an $r \times n$ parity check matrix H. Let $V = \{1, 2, \dots, n\}$ and $C = \{1, 2, \dots, r\}$. The *Tanner graph* $\mathrm{T}(V, C, E)$ associated to this code is a bipartite graph with node set $V \cup C$ and edge set consisting of all pairs (v, c) such that $H_{vc} = 1$.

Note that not only can we construct a Tanner graph for every parity check matrix, but every bipartite graph is also the Tanner graph of some classical code. As an example, let us consider the following parity check matrix of a repetition code with open boundaries:

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \tag{3.1}$$

The Tanner graph associated to this code can be found in Figure 3.1.

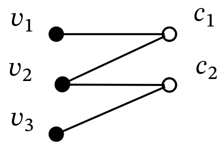


Figure 3.1: The Tanner graph corresponding to the repetition code with open boundaries. There are two vertices c_1 and c_2 that correspond to the rows of the parity check matrix, and three vertices v_1, v_2, v_3 corresponding to the columns.

Given two classical codes, we can construct their Tanner graphs. For graphs, one has the notion of a graph product. It turns out that graph products can be used to define CSS codes from two classical codes. The definition of the graph product is quite formal, however, and this construction is better understood visually. We therefore present this construction in Figure 3.2 after giving the formal definition.

Definition 3.2 (Graph Product)**.** Let $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$ be Tanner graphs. We define their *product graph* $\mathcal{G}_1 \times \mathcal{G}_2$ as the graph $\mathrm{T}(V, C, E)$, where $C = C_X \cup C_Z$, and furthermore:

$$V = V_1 \times V_2 \cup C_1 \times C_2 \tag{3.2}$$

$$C_X = C_1 \times V_2 \tag{3.3}$$

$$C_Z = V_1 \times C_2 \tag{3.4}$$

The edge set is the union of the edge set of the subgraphs $\mathcal{G}_1 \times_X \mathcal{G}_2$ and $\mathcal{G}_1 \times_Z \mathcal{G}_2$. We define $\mathcal{G}_1 \times_X \mathcal{G}_2 = \mathrm{T}(V, C_X, E_X)$, and take the edge set E_X as follows:

$$E_X = \{((x, y), (x', y')) \in V \times C_X \,:\, (x = x' \in C_1 \wedge (y, y') \in E_2) \vee (y = y' \in V_2 \wedge (x, x') \in E_1)\} \tag{3.5}$$

and take an analogous definition for $\mathcal{G}_1 \times_Z \mathcal{G}_2 = \mathrm{T}(V, C_Z, E_Z)$.

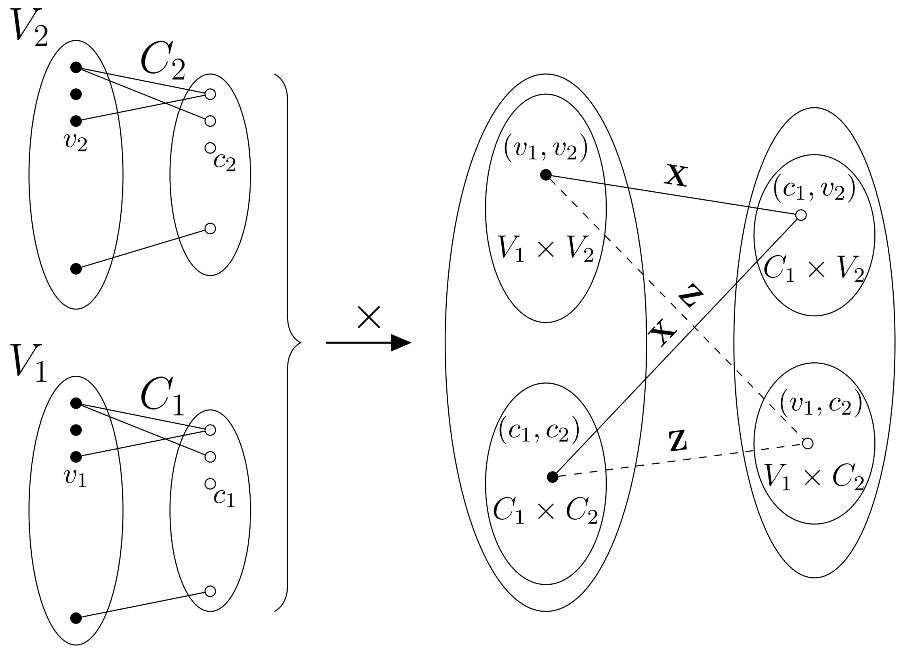


Figure 3.2: Source: [TZ09]. An illustration of the graph product of two codes. On the left, the Tanner graphs $\mathrm{T}(V_1.C_1, E_1)$ and $\mathrm{T}(V_2, C_2, E_2)$ of two classical codes are visualised. The Tanner graph of their graph product is a bipartite graph consisting of a vertex set $V_1 \times V_2 \cup C_1 \times C_2$ and another vertex set $C_1 \times V_2 \cup V_1 \times C_2$. To construct the edge set E_X, one has to draw edges between all pairs of vertices (v_1, v_2) and (c_1, v_2), and between all pairs of vertices (c_1, c_2) and (c_1, v_2). This is represented by the straight lines labelled with the letter X. Similarly, the edges belonging to edge set E_Z are represented by dashed lines, and are labelled with the letter Z.

Definition 3.3 (Graph Product Code)**.** Given two classical codes $\mathcal{C}_1, \mathcal{C}_2$ and their corresponding Tanner graphs $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$. We define their graph product code $\mathcal{Q}(\mathcal{G}_1 \times \mathcal{G}_2)$ by setting $\mathcal{C}_X = \mathcal{C}(\mathcal{G}_1 \times_X \mathcal{G}_2)$ and $\mathcal{C}_Z = \mathcal{C}(\mathcal{G}_1 \times_Z \mathcal{G}_2)$.

Note that this is a CSS code: the two parity check matrices generated by the graph product satisfy the condition $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$. To prove this, we first note that $\mathcal{C}_Z = \ker(H_Z) = \mathrm{Row}(H_Z)^\perp$. Hence, the statement is equivalent to proving that $\mathrm{Row}(H_X) \subseteq \mathrm{Row}(H_Z)^\perp$, that is, for every row h_x of H_X, that $h_z \perp h_x \forall h_z \in \mathrm{Row}(H_Z)$. It suffices to prove this for all rows of H_Z. To this end, consider a row h_Z corresponding to $(v_1, c_2) \in C_Z$ and a row h_X corresponding to an element $(c_1, v_2) \in C_X$:

$$\langle h_X, h_Z \rangle = \sum_{(v,v') \in V} h_{X_{(v,v')}} \cdot h_{Z_{(v,v')}} = h_{X_{(v_1,v_2)}} \cdot h_{Z_{(v_1,v_2)}} + h_{X_{(c_1,c_2)}} \cdot h_{Z_{(c_1,c_2)}} = 0 \text{ (mod 2)} \tag{3.6}$$

This sum simplifies to just two terms, because we have that:

$$h_{X_{(v,v')}} \cdot h_{Z_{(v,v')}} = 1 \iff h_{X_{(v,v')}} = h_{Z_{(v,v')}} = 1 \iff \begin{cases} v_1 = v = c_1 \wedge (c_2, v'), (v_2, v') \in E_2 \\ v_1 = v \wedge v_2 = v' \wedge (c_1, v) \in E_1 \wedge (v', c_2) \in E_2 \\ v' = c_2 \wedge v = c_1 \wedge (v_1, v) \in E_1 \wedge (v_2, v') \in E_2 \\ c_2 = v' = v_2 \wedge (v_1, v), (c_1, v) \in E_1 \end{cases} \tag{3.7}$$

$$= \begin{cases} v_1 = v \wedge v_2 = v' \wedge (c_1, v_1) \in E_1 \wedge (v_2, c_2) \in E_2 \\ v' = c_2 \wedge v = c_1 \wedge (v_1, c_1) \in E_1 \wedge (v_2, c_2) \in E_2 \end{cases} \tag{3.8}$$

From the final expression, we see that $h_{X_{(v_1,v_2)}} \cdot h_{Z_{(v_1,v_2)}} = h_{X_{(c_1,c_2)}} \cdot h_{Z_{(c_1,c_2)}}$, and thus, we conclude that the orthogonality condition indeed holds.

Let us now work out an example of this graph product. To this end, we again consider the repetition code with open boundaries (whose Tanner graph $\mathcal{G}_1$ can be found in Figure 3.1). We denote its parity check matrix by H_1. Moreover, let us consider the code[1] whose parity check matrix H_2 is taken to be $H_2 = H_1^T$, and whose Tanner graph we denote by $\mathcal{G}_2$. We can then determine the graph product of these two codes. For completeness, we first display their Tanner graphs in Figure 3.3.

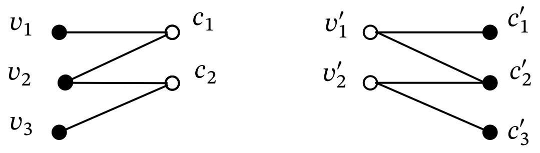


Figure 3.3: On the left: the Tanner graph of the repetition code with open boundaries, $\mathcal{G}_1$. On the right, the Tanner graph of the code whose parity check matrix is the transpose of the parity check matrix of the repetition code with open boundaries, $\mathcal{G}_2$.

We can now determine their product graph. We first display the subgraph $\mathcal{G}_1 \times_X \mathcal{G}_2$ in Figure 3.4.

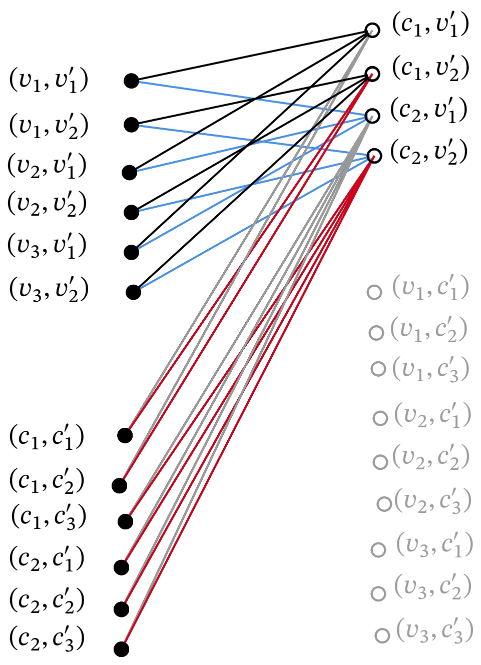


Figure 3.4: The subgraph $\mathcal{G}_1 \times_X \mathcal{G}_2$ of the product graph of the repetition code with open boundaries and its transpose code. Different groups of edges are coloured for clarity.

[1] In our discussion of the homological approach to these product codes, it will become clear why this is not just an arbitrary choice for the second code.

From this subgraph, we can derive an expression for the parity check H_X, namely:

$$H_X = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} = \left(H_1 \otimes I_2 | I_2 \otimes H_2^T\right) \tag{3.9}$$

Furthermore, we present the subgraph $\mathcal{G}_1 \times_Z \mathcal{G}_2$ in Figure 3.5.

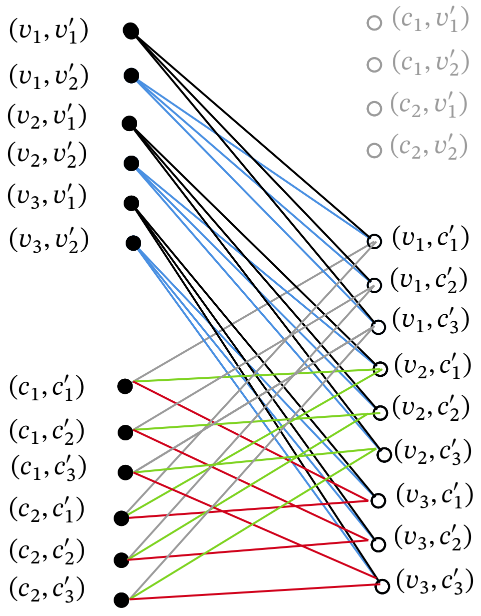


Figure 3.5: The subgraph $\mathcal{G}_1 \times_Z \mathcal{G}_2$ of the product graph of the repetition code with open boundaries and its transpose code. Different groups of edges are coloured for clarity.

The corresponding parity check matrix H_Z is now given by:

$$H_Z = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} = \left(I_3 \otimes H_1^T | H_2 \otimes I_3\right) \tag{3.10}$$

To understand this code, we introduce a 3 by 3 lattice and label its edges as is done in Figure 3.6.

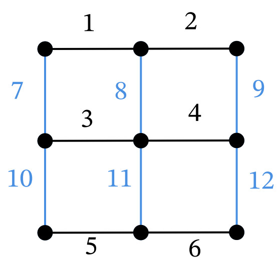


Figure 3.6: A 3 by 3 lattice along with a labelling of its edges.

As Figure 3.7 shows, the Z-checks of the CSS code (i.e. the rows of H_Z) can be identified with the vertices of this lattice.

$$H_Z = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ & & & & & \vdots & & & & & & \end{pmatrix}$$

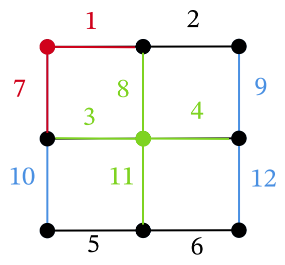


Figure 3.7: On the left, the first row of the Z-check matrix of the product code considered above is given in red. On the right, one can see that the check corresponding to this row can be identified with the top left vertex of the lattice (in red). Similarly, one can identify the fifth row of H_Z (in green) with the middle vertex of the lattice (also in green).

Furthermore, as is illustrated in Figure 3.8, the X-checks can be identified with the plaquettes of this lattice.

$$H_X = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & \end{pmatrix}$$

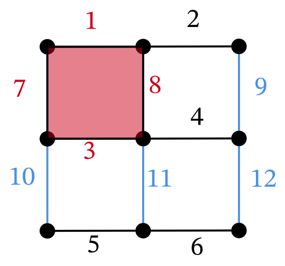


Figure 3.8: On the left, the first row of the X-check matrix of the product code considered above is given in red. On the right, one can see that the check corresponding to this row can be identified with the top left plaquette of the lattice (in red).

We have thus found the product code $\mathcal{Q}(\mathcal{G}_1 \times \mathcal{G}_2)$. While the construct code is not very useful (as it does not encode any qubits[2]), it will help to get an intuition for the product construction once we discuss this product within the context of homology.

3.1.2. The Tanner Graph Approach: Determining n, k and w

The parameters of the hypergraph product of the repetition code found in the previous section can be derived from more general results, which were first presented in [TZ09]. In this section, we will present a different proof than the one presented in [TZ09] for their most generic results, which establish a relationship for the number of encoded qubits, the weight and the number of physical qubits of the graph product of two classical codes in terms of the parameters of these classical codes.

Theorem 3.1. *Given two classical codes $\mathcal{C}_1, \mathcal{C}_2$ with parameters $[n_i, k_i, d_i]$ and weight w_i (for $i = 1, 2$) and their corresponding Tanner graphs $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$ such that their corre-*

[2]This can be seen by calculating the number of independent stabilisers. As these correspond to the rows of the parity check matrices, the number of independent stabilisers is the sum of their row ranks. As row and column ranks are equal, the fact that the first two and last two columns of H_X are linearly independent ensures that the rank of H_X is 4. The rank of H_Z is 8, although this can best be seen by manual computation.

sponding parity check matrices H_1 and H_2 are full-rank matrices, i.e. H_2 has full row-rank and H_1 has full column-rank. The parity check matrices of the code corresponding to their graph product are then given by:

$$H_X = \left(H_1 \otimes I_{n_2} | I_{n_1-k_1} \otimes H_2^T\right), \quad H_Z = \left(I_{n_1} \otimes H_2 | H_1^T \otimes I_{n_2-k_2}\right) \tag{3.11}$$

and, moreover, the product code is an $[[n_1n_2 + (n_1 - k_1)(n_2 - k_2), k_1k_2, \min\{d_1, d_2\}]]$-code with weight $w_1 + w_2$.

As stated above, we will not prove this theorem, but rather, we will prove a more general result that omits the requirement that these matrices be full-rank. To do so, however, we will need the concept of a transpose graph.

Definition 3.4 (Transpose Graph)**.** Let $G = \mathrm{T}(V, C, E)$ be the Tanner graph of a code $\mathcal{C}$ with parity check matrix H^T. Then its transpose graph G^T is the Tanner graph $\mathrm{T}(C, V, E)$. Furthermore, the classical code associated to the transpose graph has parity check matrix H^T, and we denote the number of bits in encodes by k^T bits. This code associated to the transpose graph is denoted by $\mathcal{C}^T$.

We can easily determine the number of encoded qubits of this transpose code:

Lemma 3.2. *Consider a code $\mathcal{C}$ with an $r \times n$ parity check matrix H encoding k bits and consider its transpose code $\mathcal{C}^T$. Then:*

$$k^T = r - n - k \tag{3.12}$$

Proof.

$$k = \dim(\ker(H)) = n - \dim(\mathrm{Row}(H)) = n - \dim(\mathrm{Col}(H)) = n - \dim\left(\mathrm{Row}\left(H^T\right)\right) \tag{3.13}$$

$$= n - \left(r - \dim\left(\ker\left(H^T\right)\right)\right) \tag{3.14}$$

$$= n - r + k^T \tag{3.15}$$

□

We are now ready to state the general result.

Theorem 3.3. *Given two classical codes $\mathcal{C}_1, \mathcal{C}_2$ with parameters $[[n_i, k_i, d_i]]$ and weight w_i (for i = 1, 2) and their corresponding Tanner graphs $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$. The product code $\mathcal{Q}(C_1, C_2)$ has parameters $n = n_1n_2 + r_1r_2$, $k = k_1k_2 + k_1^T k_2^T$ and has weight $w = w_1 + w_2$.*

Proof. First note that $V = V_1 \times V_2 \cup C_1 \times C_2$, and hence the code space is a subspace of a $n = |V| = |V_1||V_2| + |C_1||C|_2 = n_1n_2 + r_1r_2$ dimensional code.
We now prove that the quantum code encodes $k := k_1k_2 + k_1^T k_2^T$ qubits. To this end, note that:

$$k = \dim\left(\mathcal{C}_X / \mathcal{C}_Z^\perp\right) = \dim(\mathcal{C}_X) + \dim(\mathcal{C}_Z) - (n_1n_2 + r_1r_2) \tag{3.16}$$

Hence, it suffices to determine $\ker(H_X)$ and $\ker(H_Z)$. We claim that $\dim(\ker(H_X)) = n_1n_2 + r_1r_2 - r_1n_2 + k_1^T k_2$.

Consider $H_X = \left(H_1 \otimes I_{n_2} | I_{r_1} \otimes H_2^T\right)$. First note that for $x \in \mathbb{F}_2^{n_1n_2}, y \in \mathbb{F}_2^{r_1r_2}$, we have that $H_X(x\ y)^T = (H_1 \otimes I_{n_2})x + (I_{r_1} \otimes H_2)^T y$. Hence:

$$\begin{pmatrix} x \\ y \end{pmatrix} \in \ker(H_X) \iff (H_1 \otimes I_{n_2})x = (I_{r_1} \otimes H_2^T)y \tag{3.17}$$

We thus write $x = \sum_{\substack{1 \le i \le n_1 \\ 1 \le j \le n_2}} \alpha_{ij} e_i \otimes f_j$ and $y = \sum_{\substack{1 \le k \le r_1 \\ 1 \le l \le r_2}} \beta_{kl} g_k \otimes h_l$, where we take e_i and g_k such that:

1. $\{H_1 e_i | 1 \le i \le n_1 - k_1\}$ is a basis of $\mathrm{Col}(H_1) \subseteq \mathbb{F}_2^{r_1}$
2. $g_i = H_1 e_i$ for all $1 \le i \le n_1 - k_1$
3. $\{e_i \,|\, n_1 - k_1 < i \le n_1\}$ is a basis of $\ker(H_1) \subseteq \mathbb{F}_2^{n_1}$

4. $\{g_k = (\delta_i)_k \,|\, n_1 - k_1 < k \leq r_1\}$ with $(\delta_i)_j = \delta_{ij}$ extends $\{g_k \,|\, 1 < k \leq n_1 - k_1\}$ to a basis of $\subseteq \mathbb{F}_2^{r_1}$.

Similarly, we take f_j and h_l such that:

1. $\left\{H_2^T h_l | 1 \leq l \leq n_2 - k_2\right\}$ is a basis of Col$\left(H_2^T\right) \subseteq \mathbb{F}_2^{n_2}$
2. $f_j = H_2^T h_j$ for all $1 \leq j \leq n_2 - k_2$
3. $\{h_l \,|\, n_2 - k_2 < l \leq r_2\}$ is a basis of ker$\left(H_2^T\right) \subseteq \mathbb{F}_2^{r_2}$
4. $\left\{f_j = (\delta_j)_k \,|\, n_2 - k_2 < j \leq n_2\right\}$ extends $\left\{f_j \,|\, 1 \leq j \leq n_2 - k_2\right\}$ to a basis of $\mathbb{F}_2^{n_2}$.

We first determine $(H_1 \otimes I_{n_2})x$.

$$\begin{aligned}
(H_1 \otimes I_{n_2})x = (H_1 \otimes I_{n_2})\left(\sum_{\substack{1\leq i\leq n_1\\1\leq j\leq n_2}} \alpha_{ij} e_i \otimes f_j\right) &= \sum_{\substack{1\leq i\leq n_1\\1\leq j\leq n_2}} \alpha_{ij} H_1 e_i \otimes f_j && (3.18)\\
&= \sum_{\substack{1\leq i\leq n_1-k_1\\1\leq j\leq n_2}} \alpha_{ij} H_1 e_i \otimes f_j = \sum_{\substack{1\leq i\leq n_1-k_1\\1\leq j\leq n_2}} \alpha_{ij} g_i \otimes f_j && (3.19)\\
&= \sum_{\substack{1\leq i\leq n_1-k_1\\1\leq j\leq n_2-k_2}} \alpha_{ij} g_i \otimes f_j + \sum_{\substack{1\leq i\leq n_1-k_1\\n_2-k_2< j\leq n_2}} \alpha_{ij} g_i \otimes f_j && (3.20)\\
&= \sum_{\substack{1\leq i\leq n_1-k_1\\1\leq j\leq n_2-k_2}} \alpha_{ij} g_i \otimes H_2^T h_j + \sum_{\substack{1\leq i\leq n_1-k_1\\n_2-k_2< j\leq n_2}} \alpha_{ij} g_i \otimes f_j && (3.21)\\
& && (3.22)
\end{aligned}$$

We now determine $(I_{r_1} \otimes H_2^T)y$:

$$\begin{aligned}
(I_{r_1} \otimes H_2^T)y = (I_{r_1} \otimes H_2^T)\left(\sum_{\substack{1\leq k\leq r_1\\1\leq l\leq r_2}} \beta_{kl} g_k \otimes h_l\right) &= \sum_{\substack{1\leq k\leq r_1\\1\leq l\leq r_2}} \beta_{kl} g_k \otimes H_2^T h_l = \sum_{\substack{1\leq k\leq r_1\\1\leq l\leq n_2-k_2}} \beta_{kl} g_k \otimes H_2^T h_l && (3.23)\\
&= \sum_{\substack{1\leq k\leq n_1-k_1\\1\leq l\leq n_2-k_2}} \beta_{kl} g_k \otimes H_2^T h_l + \sum_{\substack{n_1-k_1< k\leq r_1\\1\leq l\leq n_2-k_2}} \beta_{kl} g_k \otimes H_2^T h_l\\
& && (3.24)
\end{aligned}$$

For $(x\ y)^T$ to be in ker(H_X), we thus have that:

$$\begin{cases}
\alpha_{ij} = \beta ij & i \in \{1, \dots, n_1 - k_1\}, j \in \{1, \dots, n_2 - k_2\}\\
\alpha_{ij} = 0 & i \in \{1, \dots, n_1 - k_1\}, j \in \{n_2 - k_2, \dots, n_2\}\\
\beta_{kl} = 0 & k \in \{n_1 - k_1, \dots, r_1\}, j \in \{1, \dots, n_2 - k_2\}
\end{cases} \tag{3.25}$$

We thus conclude that all elements of the kernel of H_X are of the form:

$$\sum_{\substack{1\leq i\leq n_1-k_1\\1\leq j\leq n_2-k_2}} \alpha_{ij} \begin{pmatrix} e_i \otimes H_2^T h_j \\ H_1 e_i \otimes h_j \end{pmatrix} + \sum_{\substack{n_1-k_1< i\leq n_1\\1\leq j\leq n_2}} \alpha_{ij} \begin{pmatrix} e_i \otimes f_j \\ 0 \end{pmatrix} + \sum_{\substack{1\leq k\leq r_1\\n_2-k_2< l\leq r_2}} \beta_{kl} \begin{pmatrix} 0 \\ g_k \otimes h_l \end{pmatrix} \tag{3.26}$$

We thus see that the dimension of the kernel is:

$$\begin{aligned}
(n_1 - k_1)(n_2 - k_2) + n_2 k_1 + r_1(r_2 - n_2 + k_2) &= n_1 n_2 + k_1 k_2 - k_2 n_1 + r_1 r_2 - r_1 n_2 + r_1 k_2 && (3.27)\\
&= n_1 n_2 + r_1 r_2 - r_1 n_2 + (k_1 - n_1 + r_1) k_2 && (3.28)
\end{aligned}$$

By the lemma above, the proof of the claim is finished. The proof that $\dim(\ker(H_Z)) = n_1n_2 + r_1r_2 - r_2n_1 + k_1k_2^T$ can be derived in an analogous manner. We thus see that:

$$\begin{aligned} k &= 2n_1n_2 + 2r_1r_2 - r_1n_2 - r_2n_1 + k_1k_2^T + k_1^Tk_2 - n_1n_2 - r_1r_2 && (3.29)\\ &= n_1n_2 + r_1r_2 - r_1n_2 - r_2n_1 + k_1k_2^T + k_1^Tk_2 && (3.30)\\ &= n_1n_2 + r_1r_2 - r_1n_2 - r_2n_1 + k_1(k_2 + r_2 - n_2) + k_1^Tk_2 && (3.31)\\ &= k_1k_2 + r_2(k_1 + r_1 - n_1) + n_2(n_1 - r_1 - k_1) + k_1^Tk_2 && (3.32)\\ &= k_1k_2 + r_2k_1^T - n_2k_1^T + k_1^Tk_2 && (3.33)\\ &= k_1k_2 + k_1^T(k_2 - n_2 + r_2) = k_1k_2 + k_1^Tk_2^T && (3.34)\end{aligned}$$

Lastly, from the expressions for H_X and H_Y, one can easily see that each row has weight at most $w_1 + w_2$. □

We see that the rate of the hypergraph product of two codes can, in principle, scale nicely. Let us take two full-rank parity check matrices H_1 and H_2 of two codes with rates $R_1 = k_1/n_1$ and $R_2 = k_2/n_2$, respectively. The rate of their hypergraph product, R, becomes:

$$R = \frac{k}{n} = \frac{k_1k_2}{n_1n_2 + (n_1-k_1)(n_2-k_2)} = \frac{R_1R_2}{1+(1-R_1)(1-R_2)} = \frac{1}{1+\frac{1}{R_1}\left(\frac{1}{R_2}-1\right)+\frac{1}{R_2}\left(\frac{1}{R_1}-1\right)} \tag{3.35}$$

One sees that by taking the hypergraph product of two codes, the rate of the new code scales like

$$R \sim \frac{1}{1+\Omega\left(\frac{1}{R_1}\right)} \tag{3.36}$$

Hence, for good input codes, the rate of their hypergraph product can be $\Omega(1)$. The authors of the original paper were indeed able to construct a code family with positive rate [TZ09].

3.1.3. The Tanner Graph Approach: Determining the Distance

We now turn our attention to determining the distance of the graph product of two classical codes in terms of their respective distances. In order to find an explicit expression for the distance, we need to make some assumptions.

Theorem 3.4. *Given two classical codes $\mathcal{C}_1, \mathcal{C}_2$ with parameters $[[n_i, k_i, d_i]]$ ($i = 1, 2$) and their corresponding Tanner graphs $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$. The distance of their product code $\mathcal{Q}(C_1, C_2)$, d, satisfies the following lower bound*[3]*:*

$$d \geq \min\{d_1, d_2, d_1^T, d_2^T\} \tag{3.37}$$

Furthermore, if $d_1 = \min\{d_1, d_2, d_1^T, d_2^T\}$ and $d_2 \neq \infty$, then $d = \min\{d_1, d_2, d_1^T, d_2^T\}$. This final statement still holds if we swap d_1 and d_2, or if we swap the distances for the transposed distances.

Proof. We first prove that $d \geq \min\{d_1, d_2, d_1^T, d_2^T\}$. Note that it suffices to prove that $d_X \geq \min\{d_1, d_2^T\}$ and that $d_Z \geq \min\{d_2, d_1^T\}$.
As $d_X = \min_{x\in\mathcal{C}_X\setminus\mathcal{C}_Z^\perp}\{|x|\}$, proving that $d_X \geq \min\{d_1, d_2^T\}$ is equivalent to proving that $\forall x \in \mathcal{C}_X : |x| < \min\{d_1, d_2^T\}$, we must have that $x \in \mathcal{C}_Z^\perp$. Consider one such x. We have already seen that x can be written as:

$$x = \sum_{\substack{1\leq i\leq n_1-k_1\\ 1\leq j\leq n_2-k_2}} \alpha_{ij}\begin{pmatrix} e_i\otimes H_2^Th_j\\ H_1e_i\otimes h_j\end{pmatrix} + \sum_{\substack{n_1-k_1<i\leq n_1\\ 1\leq j\leq n_2}} \alpha_{ij}\begin{pmatrix} e_i\otimes f_j\\ 0\end{pmatrix} + \sum_{\substack{1\leq k\leq r_1\\ n_2-k_2<l\leq r_2}} \beta_{kl}\begin{pmatrix} 0\\ g_k\otimes h_l\end{pmatrix} \tag{3.38}$$

We now prove that $x \in \mathcal{C}_Z^\perp$ by showing that it is in $\mathrm{Row}(H_Z)$ (note that the parity check matrix H_Z is the generator matrix of $\mathcal{C}_Z^\perp$). We thus first determine a basis of $\mathrm{Row}(H_Z)$. Note that $\mathrm{Row}(H_Z) = \mathrm{Col}(H_Z^T)$,

[3]Here we take the distance of a code encoding zero bits to be ∞.

and furthermore note that given some basis $\{v_i\} \subseteq \mathbb{F}_2^{n_1 r_2}$, Col$(H_Z) = \text{Span}_{\mathbb{F}_2}\{v_i\}$. We already have such a basis, the basis consisting of vectors $e_i \otimes h_j$ for $i = 1, \dots, n_1$ and $j = 1, \dots, r_2$. Now:

$$H_Z^T(e_i \otimes h_j) = \begin{pmatrix} I_{n_1} \otimes H_2^T \\ H_1 \otimes I_{r_2} \end{pmatrix}(e_i \otimes h_j) = \begin{pmatrix} (I_{n_1} \otimes H_2^T)(e_i \otimes h_j) \\ H_1 \otimes I_{r_2}(e_i \otimes h_j \end{pmatrix} = \begin{pmatrix} e_i \otimes H_2^T h_j) \\ H_1 e_i \otimes h_j \end{pmatrix} \tag{3.39}$$

Therefore:

$$\text{Row}(H_Z) = \sum_{\substack{1\le i\le n_1\\1\le j\le r_2}} \begin{pmatrix} e_i \otimes H_2^T h_j) \\ H_1 e_i \otimes h_j \end{pmatrix} \tag{3.40}$$

$$= \sum_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \gamma_{ij} \begin{pmatrix} e_i \otimes H_2^T h_j \\ H_1 e_i \otimes h_j \end{pmatrix} + \sum_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \gamma_{ij} \begin{pmatrix} e_i \otimes f_j \\ 0 \end{pmatrix} + \sum_{\substack{1\le k\le n_1-k_1\\n_2-k_2< l\le r_2}} \gamma_{kl} \begin{pmatrix} 0 \\ g_k \otimes h_l \end{pmatrix} \tag{3.41}$$

For x to be in Row(H_Z), we thus have to prove that

1. $\alpha_{ij} = 0\ \forall\, i, j$ such that $n_1 - k_1 < i \le n_1, n_1 - k_1 < j \le n_2$
2. $\beta_{k,l} = 0\ \forall\, k, l$ such that $n_1 - k_1 < k \le r_1, n_2 - k_2 < l \le r_2$.

We prove the former statement, and note that the proof of the latter statement runs parallel to the proof of the former. To this end, we note that as $x = (x_1, x_2)^T$, the requirement $|x| \le \min\{d_1, d_2^T\}$ in particular implies that $|x_1| \le \min\{d_1, d_2^T\} \le d_1$. Now suppose that the statement does not hold. Then we can write:

$$x_1 = \sum_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_i \otimes H_2^T h_j) + \sum_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_i \otimes f_j) + \sum_{\substack{n_1-k_1< i\le n_1\\n_2-k_2< j\le n_2}} \alpha_{ij}(e_i \otimes f_j) \tag{3.42}$$

where the last term is non-zero. Note that as e_i with $i > n_1 - k_1$ is an element of ker(H_1), we have that $|e_i| \ge d_1$. Furthermore, as $\alpha_{ij} \in \{0, 1\}$, we can write $\hat{e}_j = \sum_{n_1-k_1<i\le n_1} \alpha_{ij} e_i$, such that:

$$x_1 = \sum_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_i \otimes H_2^T h_j) + \sum_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_i \otimes f_j) + \sum_{n_2-k_2< i\le n_2} (\hat{e}_j \otimes f_j) \tag{3.43}$$

Now, we can determine $|x_1|$:

$$|x_1| = \left\| \begin{pmatrix} \sum\limits_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_1} \cdot H_2^T h_j) + \sum\limits_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_1} \cdot f_j) + \sum\limits_{n_2-k_2< j\le n_2} \hat{e}_{j1} \cdot f_j \\ \vdots \\ \sum\limits_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_{n_1}} \cdot H_2^T h_j) + \sum\limits_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_{n_1}} \cdot f_j) + \sum\limits_{n_2-k_2< j\le n_2} \hat{e}_{jn_1} \cdot f_j \end{pmatrix} \right\| \tag{3.44}$$

Which is equal to:

$$|x_1| = \sum_{k=1}^{n_1} \left| \sum_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_k} \cdot H_2^T h_j) + \sum_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_k} \cdot f_j) + \sum_{n_2-k_2< j\le n_2} \hat{e}_{jk} \cdot f_j \right| \tag{3.45}$$

let us consider some $j_0 \in \{n_2 - k_2 + 1, \dots, n_2\}$ such that $\hat{e}_{j_0} \ne 0$. Then we can write:

$$|x_1| = \sum_{k=1}^{n_1} \left| \sum_{\substack{1\le i\le n_1-k_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_k} \cdot H_2^T h_j) + \sum_{\substack{n_1-k_1< i\le n_1\\1\le j\le n_2-k_2}} \alpha_{ij}(e_{i_k} \cdot f_j) + (\hat{e}_{i_{0_k}} \cdot f_{i_0}) + \sum_{\substack{n_2-k_2< j\le n_2\\ j\ne j_0}} \hat{e}_{j_k} \cdot f_j \right| \tag{3.46}$$

Let us define the set $K := \left\{k \in \{1, \dots, n_1\} : \hat{e}_{i_{0}k} = 1\right\}$. Note that as these $\hat{e}_i$'s are linear combinations of elements in $\mathcal{C}_1$, we have that $|\hat{e}_{i_0}| \geq d_1$, and therefore $\#K \geq d_1$. Hence, we see that:

$$|x_1| \geq \sum_{k \in K} \left| \sum_{\substack{1 \leq i \leq n_1 - k_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot H_2^T h_j) + \sum_{\substack{n_1 - k_1 < i \leq n_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot f_j) + \hat{e}_{i_{0}k} \cdot f_{i_0} + \sum_{\substack{n_2 - k_2 < j \leq n_2 \\ j \neq j_0}} \hat{e}_{i_k} \cdot f_j \right| \tag{3.47}$$

We now claim that $\forall k \in K$:

$$\left| \sum_{\substack{1 \leq i \leq n_1 - k_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot H_2^T h_j) + \sum_{\substack{n_1 - k_1 < i \leq n_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot f_j) + \hat{e}_{i_{0}k} \cdot f_{i_0} + \sum_{\substack{n_2 - k_2 < j \leq n_2 \\ j \neq j_0}} \hat{e}_{i_k} \cdot f_j \right| \geq 1 \tag{3.48}$$

We only have to rule out the case that the Hamming weight equals 0 (as Hamming weights are always positive). To this end, we note that $e_{i_{0}k} = 1$ for all $k \in K$. Now suppose the weight were zero for some $k \in K$. Then we this would imply that:

$$\sum_{\substack{1 \leq i \leq n_1 - k_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot H_2^T h_j) + \sum_{\substack{n_1 - k_1 < i \leq n_1 \\ 1 \leq j \leq n_2 - k_2}} \alpha_{ij}(e_{i_k} \cdot f_j) = f_{i_0} + \sum_{\substack{n_2 - k_2 < j \leq n_2 \\ j \neq j_0}} \hat{e}_{i_k} \cdot f_j \tag{3.49}$$

This can only be the case, however, if both sides equal 0, i.e. if:

$$f_{i_0} = \sum_{\substack{n_2 - k_2 < j \leq n_2 \\ j \neq j_0}} \hat{e}_{i_k} \cdot f_j \tag{3.50}$$

But this cannot be the case, as all f_i's considered are different basis vectors. $\lightning$. Hence, the Hamming weight is at least 1. But then, we see that:

$$|x_1| \geq \#K \cdot 1 = d_1 \tag{3.51}$$

That is, $|x| \geq |x_1| \geq d_1$. $\lightning$.
Note that the proof of $d_Z \geq \min\{d_2, d_1^T\}$ is completely analogous to this proof. This thus proves the first part of the theorem.

Now suppose that $d_1 = \min\{d_1, d_2, d_1^T, d_2^T\}$ and that $d_2 \neq \infty$. We can now explicitly construct a codeword $x \in \mathcal{C}_X \setminus \mathcal{C}_Z^\perp$ such that $|x| = d_1$. First consider a vector $x_0 \in \mathcal{C}_1$ with Hamming weight $d_1 = |x_0|$. As $d_2 \neq \infty$, $k_2 \neq 0$, and we see that there are f_j's for $j > n_2 - k_2$. But these f_j's all have weight 1 (by construction). Pick one such f_j, say f_{j_0}. Then:

$$|x_0 \otimes f_{j_0}| = |x_0| \cdot |f_{j_0}| = |x_0| = d_1 \tag{3.52}$$

Furthermore, $\left(x_0 \otimes f_{j_0}, 0\right)^T \in \mathcal{C}_X \setminus \mathcal{C}_Z^\perp$, as $f_{j_0} \notin \text{Col}\left(H_2^T\right)$.
The proof for the case where $d_2^T = \min\{d_1, d_2, d_1^T, d_2^T\}$ and $d_1^T \neq \infty$ runs similarly. The other two cases can be proved similarly by constructing a codeword $x \in \mathcal{C}_Z \setminus \mathcal{C}_X^\perp$. $\square$

This theorem points towards a fundamental limitation of the hypergraph product construction: considering two families of classical codes with full rank parity check matrices and with linear distance, we see that the minimum distance of the hypergraph product of these two codes scales like $\Theta(\sqrt{n})$, and hence, is no longer linear. Thus, even with two good classical codes, the hypergraph product is still not able to yield a good CSS code.

One may wonder why this code construction is called the hypergraph product construction. The reason for this is there exists a different product of two Tanner graphs than the graph product we have seen thus far, which is the so-called *hypergraph product*:

Definition 3.5. Given two Tanner graphs $\mathcal{G}_1 = \mathrm{T}(V_1, C_1, E_1)$ and $\mathcal{G}_2 = \mathrm{T}(V_2, C_2, E_2)$. The *hypergraph product* of these two graphs, $G_1 \otimes G_2$, is the Tanner subgraph of $G_1 \times G_2$ with variable node set $V_1 \times V_2$ and check node set $C_1 \times V_2 \cup V_1 \times C_2$.

One can use this hypergraph product, which is much easier to construct than the graph product, to analyse the graph product of two codes, since the following relations hold:

$$\mathcal{G}_1 \times_X \mathcal{G}_2 = (\mathcal{G}_1^T \otimes \mathcal{G}_2)^T \tag{3.53}$$

$$\mathcal{G}_1 \times_Z \mathcal{G}_2 = (\mathcal{G}_1 \otimes \mathcal{G}_2^T)^T \tag{3.54}$$

In [TZ09], this hypergraph product was used for the analysis of the graph product construction, and therefore, their code construction was dubbed the hypergraph product construction. This point of view will turn out to be valuable when we consider the homological perspective on these codes.

3.1.4. The Homological Approach: The Construction

Now that we have seen how the hypergraph product of two codes can be constructed using Tanner graphs, we change perspectives and try to construct the hypergraph product from the point of view of homology. Let us first consider the chain complexes which one can associate to two classical codes, $\mathcal{C}_1^T$ and $\mathcal{C}_2$:

$$\mathbb{F}_2^{r_1} \xrightarrow{H_1^T} \mathbb{F}_2^{n_1}$$

$$\mathbb{F}_2^{n_2} \xrightarrow{H_2} \mathbb{F}_2^{r_2}$$

By taking the tensor product of these complexes, we obtain the following product complex:

$$\mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{n_2} \xrightarrow{\partial_2} \mathbb{F}_2^{n_1} \otimes \mathbb{F}_2^{n_2} \oplus \mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{r_2} \xrightarrow{\partial_1} \mathbb{F}_2^{n_1} \otimes \mathbb{F}_2^{r_2}$$

Where:

$$\partial_2 = \begin{pmatrix} H_1^T \otimes I_{n_2} \\ I_{r_1} \otimes H_2 \end{pmatrix} = H_X^T \tag{3.55}$$

$$\partial_1 = \left(I_{n_1} \otimes H_2 \;\middle|\; H_1^T \otimes I_{r_2} \right) = H_Z \tag{3.56}$$

Taking the transposed complex of the one obtained here gives us H_X as a differential. This procedure shows the symmetry between taking tensor products of chain complexes and taking hypergraph products: we obtain H_X as the differential of the complex $(\mathcal{C}_1^T \otimes \mathcal{C}_2)^T$, while we can obtain the Tanner graph associated to the X-stabilisers, $\mathcal{G}_1 \times_X \mathcal{G}_2$, by taking the transpose graph of the hypergraph product of the tranpose of one Tanner graph and the other, i.e. as $(\mathcal{G}_1 \otimes \mathcal{G}_2^T)^T$. Hence, we see that the operation that the hypergraph product encodes in the world of Tanner graphs is equivalent to the operation that the tensor product encodes in the world of chain complexes!

Let us return to the example of the graph product of a repetition codes with open boundaries and its transpose code, as was presented at the end of Section 3.1.1. We denote this code by $\mathcal{C}$, and we recall that its parity check matrix of this code is given by:

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \tag{3.57}$$

The chain complex corresponding to this code can actually be interpreted as the cellulation of a line, as is illustrated in Figure 3.9.

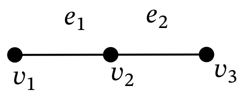


Figure 3.9: The cellular complex corresponding to a repetition code with open boundaries.

By shifting to the perspective of chain complexes and denoting the chain complex corresponding to the code $\mathcal{C}$ by $\mathcal{C}_*$, we see that this graph product code is given by the chain complex $\mathcal{C}_* \otimes \mathcal{C}_*$. By Figure 3.9, we see that the chain complex $\mathcal{C}_*$ actually corresponds to the cellular complex belonging to a straight line, which is homeomorphic to the closed unit interval $[-1, 1]$. As the tensor product of two cellular complexes is a cellular complex of the product of the corresponding spaces, we see that $\mathcal{C}_* \otimes \mathcal{C}_*$ is actually a cellular complex of $[-1, 1] \times [-1, 1]$. This allows us to visualise this product construction quite nicely, as is done in Figure 3.10.

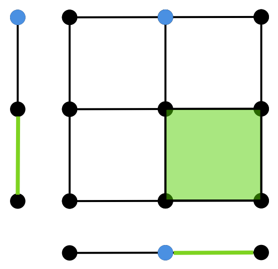


Figure 3.10: A visualisation of the hypergraph product of the repetition code with open boundaries and its transpose code. As this construction is equivalent to taking the tensor product of the chain complex corresponding to this code with itself, we see that the product code is described by the tensor product complex. As the repetition code arises from a cellular complex of a straight line, the product code therefore arises from the cellular complex of a square. This is illustrated in this figure. One can see that the Z-checks of the product code, which correspond to the 0-cells of the square (in blue), arise from taking the product of 0-cells of the straight lines, whereas the X-checks, which correspond to the plaquettes (in green), arise from taking the product of two 1-cells.

3.1.5. The Homological Approach: Determining k

Given that we can therefore understand hypergraph product codes in terms of tensor products of chain complexes, homology becomes the natural language for analysing this product. To this end, let us use the Künneth formula to calculate the number of encoded qubits k in a substantially easier manner. To do so, however, we first need to determine the homology groups of the chain complexes related to $\mathcal{C}_1$ and $\mathcal{C}_2$:

$$H_0(\mathcal{C}_1^T) = \mathbb{F}_2^{n_1}/\mathrm{Im}\left(H_1^T\right) = \mathbb{F}_2^{n_1}/\mathrm{Row}\left(H_1\right) \tag{3.58}$$

$$H_1(\mathcal{C}_1^T) = \ker\left(H_1^T\right)/0 \tag{3.59}$$

$$H_0(\mathcal{C}_2) = \mathbb{F}_2^{r_2}/\mathrm{Im}\left(H_2\right) = \mathbb{F}_2^{r_2}/\mathrm{Row}\left(H_2^T\right) \tag{3.60}$$

$$H_1(\mathcal{C}_2) = \ker\left(H_2\right) \tag{3.61}$$

Using the following Kunneth formula:

$$H_1(\mathcal{C}_1^T \otimes \mathcal{C}_2) = H_1(\mathcal{C}_1^T) \otimes H_0(\mathcal{C}_2) \oplus H_0(\mathcal{C}_1^T) \otimes H_1(\mathcal{C}_2) \tag{3.62}$$

And hence, we almost immediately find the expected result:

$$k := \dim\left(H_1(\mathcal{C}_1^T \otimes \mathcal{C}_2)\right) = \dim\left(H_1(\mathcal{C}_1^T)\right) \cdot \dim\left(H_0(\mathcal{C}_2)\right) + \dim\left(H_0(\mathcal{C}_1^T)\right) \cdot \dim\left(H_1(\mathcal{C}_2)\right) = k_1 k_2 + k_1^T k_2^T \tag{3.63}$$

We note that, although this perspective proves useful when determining the number of encoded qubits, the distance is not captured by the homology of the space. Therefore, in order to analyse the distance, one would still have to resort to alternative (linear algebraic) methods, like the one demonstrated in the previous section.

3.2. The EKZ product

3.2.1. Motivation and Construction

In the previous section, we saw that the hypergraph product of two codes, which is a CSS code, can be interpreted using homology. From that point of view, the hypergraph product becomes the tensor product of two length-1 chain complexes, which correspond to two classical codes. Naturally, the question arises whether one can extend this construction by taking the tensor product between a classical and a quantum code. This idea was worked out in [EKZ20]. We will present this construction here, and determine the properties of such codes.

To this end, let us consider the following chain complexes, which correspond to a quantum CSS code $\mathcal{Q}$ with parity matrices H_1 and H_2, and a classical code $\mathcal{C}$ with a full rank parity check matrix H.

$$\begin{tikzcd} \mathbb{F}_2^{r_1} \arrow[r, "H_1^T"] & \mathbb{F}_2^{n_Q} \arrow[r, "H_2"] & \mathbb{F}_2^{r_2} \end{tikzcd}$$

$$\begin{tikzcd} \mathbb{F}_2^{n_c} \arrow[r, "H"] & \mathbb{F}_2^{r_c} \end{tikzcd}$$

We can take the tensor product of these chain complexes to find:

$$\begin{tikzcd} \mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{n_c} \arrow[r] & \mathbb{F}_2^{n_Q} \otimes \mathbb{F}_2^{n_c} \oplus \mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{r_c} \arrow[r] & \mathbb{F}_2^{r_2} \otimes \mathbb{F}_2^{n_c} \oplus \mathbb{F}_2^{n_Q} \otimes \mathbb{F}_2^{r_c} \arrow[r] & \mathbb{F}_2^{r_2} \otimes \mathbb{F}_2^{r_c} \end{tikzcd}$$

This is a chain complex of length 3, whereas CSS codes correspond to chain complexes of length 2. Hence, we cannot interpret this complex as a CSS code. This complication can be resolved quite simply, however: we can just ignore the 0th or the 3rd chain in the complex, in order to obtain a complex of length 2. Let us ignore the 0th chain, and consider the properties of the CSS code associated to the three higher chains of the tensor product complex:

$$\begin{tikzcd} \mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{n_c} \arrow[r, "\partial_2"] & \mathbb{F}_2^{n_Q} \otimes \mathbb{F}_2^{n_c} \oplus \mathbb{F}_2^{r_1} \otimes \mathbb{F}_2^{r_c} \arrow[r, "\partial_1"] & \mathbb{F}_2^{r_2} \otimes \mathbb{F}_2^{n_c} \oplus \mathbb{F}_2^{n_Q} \otimes \mathbb{F}_2^{r_c} \end{tikzcd}$$

Where:

$$\partial_2 = \begin{pmatrix} H_1^T \otimes I_{n_c} \\ I_{r_1} \otimes H \end{pmatrix} \tag{3.64}$$

$$\partial_1 = \begin{pmatrix} H_2 \otimes I_{n_c} & \emptyset \\ I_{n_Q} \otimes H & H_1^T \otimes I_{r_c} \end{pmatrix} \tag{3.65}$$

As a sanity check, we note that this is indeed a chain complex:

$$\partial_1 \circ \partial_2 = \begin{pmatrix} H_2 \otimes I_{n_c} & \emptyset \\ I_{n_Q} \otimes H & H_1^T \otimes I_{r_c} \end{pmatrix} \cdot \begin{pmatrix} H_1^T \otimes I_{n_c} \\ I_{r_1} \otimes H \end{pmatrix} = \begin{pmatrix} H_2 H_1^T \otimes I_{n_c} \\ 2H_Z^T \otimes H \end{pmatrix} = 0 \tag{3.66}$$

We dub this newly constructed code the *EKZ product* of $\mathcal{Q}$ and $\mathcal{C}$, and denote it by EKZ($\mathcal{Q}, \mathcal{C}$).

3.2.2. Properties

We claim that this code has the following parameters:

Theorem 3.5. *Let $\mathcal{Q}$ be a CSS code with parameters $[[n_\mathcal{Q}, k_\mathcal{Q}, d_1, d_2]]$, and let $\mathcal{C}$ be a classical code with parameters $[n_c, k_c, d_c]$. Their EKZ product, EKZ($\mathcal{Q}, \mathcal{C}$), has the following parameters:*

$$n = n_\mathcal{Q} \cdot n_c + r_1 \cdot r_c \tag{3.67}$$

$$k = k_\mathcal{Q} \cdot k_c \tag{3.68}$$

$$d_X = d_1 \tag{3.69}$$

$$d_Z = d_2 \cdot d_c \tag{3.70}$$

Proof. The number of encoded qubits, k, easily follows from a Künneth formula:

$$\begin{aligned} k &= \dim(H_1(\mathrm{EKZ}(\mathcal{Q},\mathcal{C}))) = \dim(H_2(\mathcal{Q}\otimes\mathcal{C})) && (3.71)\\ &= \dim(H_2(\mathcal{Q})\otimes H_0(\mathcal{C})\oplus H_1(\mathcal{Q})\otimes H_1(\mathcal{C})\oplus H_0(\mathcal{Q})\otimes H_2(\mathcal{C})) && (3.72)\\ &= \dim(H_2(\mathcal{Q})\otimes H_0(\mathcal{C})\oplus H_1(\mathcal{Q})\otimes H_1(\mathcal{C})) && (3.73)\\ &= \dim(H_2(\mathcal{Q}))\cdot\dim(H_0(\mathcal{C}))+\dim(H_1(\mathcal{Q}))\cdot\dim(H_1(\mathcal{C})) && (3.74)\\ &= \dim(H_1(\mathcal{Q}))\cdot\dim(H_1(\mathcal{C})) && (3.75)\\ &= k_{\mathcal{Q}}k_c && (3.76)\end{aligned}$$

where the fact that $\dim(H_0(\mathcal{C}))=0$ follows from the fact that H has full rank:

$$\dim(H_0(\mathcal{C})) = \dim\left(\mathbb{F}_2^{r_c}/\mathrm{Im}(H)\right) = r_c - \mathrm{Rank}(H) = 0 \tag{3.77}$$

Furthermore, the distance of this code can be determined in an analogous manner as the distance of the hypergraph product. Let us determine the parameter d_Z. We proceed in two steps: first, we prove that $d_2\cdot d_{\mathcal{C}}$ is a lower bound on the distance, and afterwards, we prove that there is a codeword in the EKZ product code that attains this bound.
We thus first show that $d_Z \geq d_2\cdot d_{\mathcal{C}}$, where $d_2 = \min\{|x|\}_{x\in\ker(H_2)\setminus\mathrm{Row}(H_1)}$. To this end, we take $x\in\ker(H_2)$ with $|x| < d_2\cdot d_{\mathcal{C}}$, and show that $x\in\mathrm{Row}(H_1)$. Analogous to the proof of the distance of the hypergraph product code, we can give an explicit expression for $\ker(H_2)$ (and hence also for x):

$$\ker(H_Z) = \left\{\begin{pmatrix} x\\ y\end{pmatrix} : (H_2\otimes I_{n_c})x = 0 \text{ and } (I_{n_{\mathcal{Q}}}\otimes H)x = (H_1^T\otimes I_{r_c})y\right\} \tag{3.78}$$

which yields a general expression for x:

$$x = \sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\begin{pmatrix} H_1^T a_i\otimes b_j\\ a_i\otimes Hb_j\end{pmatrix} + \sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_2\\ n_c-k_c<j\leq n_c}} \beta_{ij}\begin{pmatrix} c_i\otimes b_j\\ 0\end{pmatrix} + \sum_{\substack{n_{\mathcal{Q}}-k_1<i\leq r_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\begin{pmatrix} 0\\ a_i\otimes Hb_j\end{pmatrix} \tag{3.79}$$

where we note that:

1. $a_i\in\mathbb{F}_2^{r_1}$ are taken such that $\{H_1^Ta_i | 1\leq i\leq n_{\mathcal{Q}-k_1}\}$ is a basis of $\mathrm{Col}\left(H_1^T\right)$ and $\{a_i | n_{\mathcal{Q}}-k_1 < i\leq r_1\}$ is a basis of $\ker\left(H_1^T\right)$.

2. $c_i = H_1^Ta_i$ for $1\leq i\leq n_{\mathcal{Q}}-k_1$, and for $i\leq i\leq n_{\mathcal{Q}-k_2}$, $\{c_i\}$ is an extension to a basis of $\ker(H_2)$. (Note that $H_2H_1^T=0$ implies in particular that $\mathrm{Col}\left(H_1^T\right)\subseteq\ker(H_2)$).

3. $b_j\in\mathbb{F}_2^{n_c}$ are taken such that $\{Hb_j | 1\leq j\leq n_c-k_c\}$ is a basis of $\mathrm{Col}(()H) = \mathbb{F}_2^{n_c-k_c}$ (as H has full rank), and $\{b_j | n_c-k_c<j\leq n_c\}$ extends to a basis of $\ker(H)$.

Furthermore, we can determine a basis of $\mathrm{Row}(H_X) = \mathrm{Col}(H_X)$ by considering the span of all $H_X\left(a_i,b_j\right)^T$:

$$y\in\mathcal{C}_X^\perp \iff y = \sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_1\\ 1\leq j\leq n_c-k_c}} \gamma_{ij}\begin{pmatrix} H_1^T a_i\otimes b_j\\ a_i\otimes Hb_j\end{pmatrix} + \sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\begin{pmatrix} c_i\otimes b_j\\ 0\end{pmatrix} + \sum_{\substack{n_{\mathcal{Q}}-k_1<i\leq r_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\begin{pmatrix} 0\\ a_i\otimes Hb_j\end{pmatrix} \tag{3.80}$$

We thus have to prove that for our choice of x: $\beta_{ij}=0$ for all $n_{\mathcal{Q}}-k_1 < i\leq n_{\mathcal{Q}}-k_2$ and $n_c-k_c<j\leq n_c$. We do this by deriving a contradiction: if there is such an x, then this x necessarily has Hamming weight $|x|\geq d_2d_c$. We restrict ourselves to the upper part of x, as its weight is always smaller than the weight of all of x. Furthermore, we split up this upper part of x into a $\mathcal{C}_X^\perp$-term and a term outside of this subcode:

$$|x| \geq \left|\sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\left(H_1^Ta_i\otimes b_j\right) + \sum_{\substack{1\leq i\leq n_{\mathcal{Q}}-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(c_i\otimes b_j\right) + \sum_{\substack{n_{\mathcal{Q}}-k_1<i\leq n_{\mathcal{Q}}-k_2\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(c_i\otimes b_j\right)\right| \tag{3.81}$$

To avoid having to deal with any dependencies in the basis vectors c_i in the last sum later on, we group all terms corresponding to the same c_i together — that is, we take $\hat{b}_i = \sum_{ij} b_j$. Now, let us consider a non-zero term, and derive a contradiction. To this end, let i_0 be an index such that $\hat{b}_{i_0} \neq 0$. Now:

$$|x| \geq \left\| \begin{pmatrix} \sum\limits_{\substack{1\leq i\leq n_Q-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\left(b_{j1}H_1^T a_i\right) + \sum\limits_{\substack{1\leq i\leq n_Q-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(b_{j1}\cdot c_i\right) + \sum\limits_{n_Q-k_1<i\leq n_Q-k_2} \beta_{ij}\left(\hat{b}_{i1}\cdot c_i\right) \\ \vdots \\ \sum\limits_{\substack{1\leq i\leq n_Q-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\left(b_{jn_c}H_1^T a_i\right) + \sum\limits_{\substack{1\leq i\leq n_Q-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(b_{jn_c}\cdot c_i\right) + \sum\limits_{n_Q-k_1<i\leq n_Q-k_2} \beta_{ij}\left(\hat{b}_{in_c}\cdot c_i\right) \end{pmatrix} \right\| \tag{3.82}$$

The latter term is equal to:

$$\sum_{l=1}^{n_c} \left| \sum_{\substack{1\leq i\leq n_Q-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\left(b_{jl}H_1^T a_i\right) + \sum_{\substack{1\leq i\leq n_Q-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(b_{jl}\cdot c_i\right) + b_{i_0 l}c_{i_0} + \sum_{\substack{n_Q-k_1<i\leq n_Q-k_2\\ i\neq i_0}} \beta_{ij}\left(\hat{b}_{il}\cdot c_i\right) \right| \tag{3.83}$$

Note that $b_{i_0} \in \ker(H)$, and hence it is non-zero at at least d_c indices. We this sum is then larger than the sum over those indices:

$$\geq d_c \cdot \min_{1\leq l\leq n_c} \left\{ \left| \sum_{\substack{1\leq i\leq n_Q-k_1\\ 1\leq j\leq n_c-k_c}} \beta_{ij}\left(b_{jl}H_1^T a_i\right) + \sum_{\substack{1\leq i\leq n_Q-k_1\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(b_{jl}\cdot c_i\right) + c_{i_0} + \sum_{\substack{n_Q-k_1<i\leq n_Q-k_2\\ i\neq i_0}} \beta_{ij}\left(\hat{b}_{il}\cdot c_i\right) \right| \right\} \tag{3.84}$$

But the term that is being minimised is simply a sum over $c_i's$ in both Row(H_1) and non-zero terms in ker$(H_2) \setminus$ Row(H_1), and is therefore in ker$(H_2) \setminus H_1$. The latter terms are non-zero, as we chose our $\hat{b}_i$'s such that all different c_i's were linearly independent. But now, the minimum weight of elements in this ker$(H_2) \setminus H_1$ is d_2 (by definition!), hence we find that:

$$|x| \geq d_c \cdot d_2 \tag{3.85}$$

which is indeed a contradiction, thus $d_X \geq d_c \cdot d_2$.
We now construct a codeword with Hamming weight $d_c d_2$. To this end, we can first construct a codeword element $x_1 \in \mathcal{C}$ with $|x_1| = d_c$. Similarly, we can find a codeword $x_2 \in$ ker$(H_2) \setminus H_1$ such that $|x_2| = d_2$. But then, the codeword $x := x_1 \otimes x_2$ has Hamming weight $d_2 d_c$, and is in $\mathcal{C}_Z/\mathcal{C}_X^\perp$, as it is of the form

$$x = \sum_{\substack{n_Q-k_1<i\leq n_Q-k_2\\ n_c-k_c<j\leq n_c}} \beta_{ij}\left(c_i \otimes b_j\right) \tag{3.86}$$

for some non-zero β_{ij}. The proof for $d_X \geq d_1$ is more simple than this proof (and is in fact almost completely analogous to the proof of the distance of the hypergraph product code). □

Although this construction is interesting already for the mere fact that it is a generalisation of the hypergraph product, it does offer another, somewhat surprising, advantage: given a quantum CSS code with distances d_X and d_Z. it can of course happen that one distance is much larger than the other (or scales more favourably than the other), for example that $d_X \geq d_Z$. In this case, d_X imposes a bottleneck on the performance of this code, and hence, it can be useful to consider ways of increasing d_X to (approximately) d_Z without creating too many drawbacks on the code. One way of achieving this is actually by taking the EKZ product with another classical code: take a classical code with a distance of $d_c \simeq d_Z/d_X$. The EKZ product of these codes, then, yields a CSS code with distances d_Z and $d_c \cdot d_X \simeq d_Z$, and as such, greatly improves the distance of our initial code, albeit at the expense of needing more qubits. The EKZ product thus (partially) mitigates the distance limitation of the hypergraph product. This is why this product was called a *distance balancing* method by the authors of the original paper [EKZ20]. They were indeed able to improve upon the distance record of the hypergraph product, by devising a family of codes with a distance of $\Omega(\sqrt{n\log n})$ — although they were unable to prove that this code family had a non-vanishing rate.

3.3. Products of Classical Codes and Chain Complexes of Arbitrary Length

The EKZ product generalised the hypergraph product by replacing one of the classical input codes with a CSS code. In the spirit of regarding such codes as chain complexes, one can wonder whether this approach can be generalised even further by considering products of classical codes and chain complexes of arbitrary length. It is precisely this that the authors in [ZP19, ZP20] set out to do.

To this end, let us consider a $[n_c, k_c, d_c]$ classical code $\mathcal{C}$ and a chain complex $\mathcal{K}$ of length $K \geq 2$:

$$\begin{tikzcd} \mathbb{F}_2^n \arrow[r, "H"] & \mathbb{F}_2^r \end{tikzcd}$$

$$\begin{tikzcd} \mathbb{F}_2^{\alpha_K} \arrow[r] & \cdots \arrow[r] & \mathbb{F}_2^{\alpha_1} \arrow[r] & \mathbb{F}_2^{\alpha_0} \end{tikzcd}$$

The tensor product of these two complexes is simply:

$$\begin{tikzcd} C_{K+1} \arrow[r] & C_K \arrow[r] & \cdots \arrow[r] & C_1 \arrow[r] & C_0 \end{tikzcd}$$

where:

$$C_i = \mathbb{F}_2^n \otimes \mathbb{F}_2^{\alpha_{i-1}} \oplus \mathbb{F}_2^r \otimes \mathbb{F}_2^{\alpha_i} \tag{3.87}$$

We can pick out a length 2 subcomplex of this tensor product complex, which is of the following form:

$$\begin{tikzcd} C_{i+1} \arrow[r, "\partial_{i+1}"] & C_i \arrow[r, "\partial_i"] & C_{i-1} \end{tikzcd}$$

We can now easily calculate the number of encoded qubits using the Künneth formula:

$$k = \dim_{\mathbb{F}_2}(\mathcal{H}_i(\mathcal{C} \otimes \mathcal{K})) = \dim_{\mathbb{F}_2}(\mathcal{H}_0(\mathcal{C})) \cdot \dim_{\mathbb{F}_2}(\mathcal{H}_i(\mathcal{K})) + \dim_{\mathbb{F}_2}(\mathcal{H}_1(\mathcal{C})) \cdot \dim_{\mathbb{F}_2}(\mathcal{H}_{i-1}(\mathcal{K})) \tag{3.88}$$

Using linear algebraic arguments, one can again derive an upper bound on the minimum distance of the obtained code, and can furthermore prove that this bound is tight. To this end, let $d_i(\mathcal{C})$ denote the minimum weight of a non-zero representative of $\mathcal{H}_i(C)$, then:

$$d = \min\{d_i(K) \cdot d_0(\mathcal{C}), d_{i-1}(K) \cdot d_1(\mathcal{C})\} \tag{3.89}$$

We do not present a proof for this statement, and refer the interested reader to [ZP19]. An analogous statement can be derived for the minimum weight of a non-zero cohomology representative. In this way, one can bound the minimum distance of the product code.

3.4. Beyond Tensor Products Codes

Even with the last, most generic, tensor product construction, good qLDPC codes had still not been found. One is therefore led to conclude that tensor products of chain complexes over vector spaces, unfortunately, are not tools powerful enough to devise good qLDPC codes. We are therefore led to consider ways in which we can improve upon these tensor product constructions. The next two parts of this thesis present two different approaches in that regard: one that can be said to generalise the structure of the chains in the chain complexes, and another one which adds additional structure to the differentials of the chain complexes.

II

Ring Module Homology and Lifted Product Codes

4

Homology of Ring Modules

The homological algebra presented up to this point mostly revolved around chain complexes over vector spaces: for example, the Künneth formula we used to analyse the various product constructions of LDPC codes establishes a relationship between the homology groups of the tensor product of chain complexes over vector fields and the tensor products of their homology groups. A natural direction in which this theory can be generalised lies in imposing less structure on the chains of our chain complexes.

In this chapter, we will consider chain complexes of modules over rings. This abstraction, however, comes at a price, as tensor products over modules behave less nicely than tensor products over vector spaces. In particular, the Künneth formula that we have seen up until now does not hold in the more general context of chain complexes of modules over rings — at least, not in the same form. The final goal of this chapter, therefore, will be to present a more general version of the Künneth theorem we saw up until now. All of this work serves in preparation of the next chapter, where we will seek to generalise the hypergraph product construction of qLDPC codes. The results in this chapter are mostly based on [Wei94, Rie16, Len18].

4.1. Rings and Ring Modules

Before we can start discussing the homological aspects of ring modules, we first briefly remind the reader of the definitions of rings and their modules.

Definition 4.1. A set $\mathcal{R}$ equipped with two binary operations called addition $+ : \mathcal{R} \times \mathcal{R} \to \mathcal{R}$ and *multiplication* $\cdot : \mathcal{R} \times \mathcal{R} \to \mathcal{R}$ and two neutral elements denoted by $0, 1 \in \mathcal{R}$ is called a *ring* if:

1. $(\mathcal{R}, +, 0)$ is an abelian group
2. $\cdot$ is associative, i.e. $(r \cdot s) \cdot t = r \cdot (s \cdot t)$ for all $r, s, t \in \mathcal{R}$
3. $1 \cdot r = r \cdot 1 = r$ for all $r \in \mathcal{R}$
4. $r \cdot (s + t) = r \cdot s + r \cdot t$ and $(r + s) \cdot t = r \cdot t + s \cdot t$ for all $r, s, t \in \mathcal{R}$.

Finally, if the multiplication operation is commutative, we call $\mathcal{R}$ a *commutative ring*.

Some important examples of rings include the integers $\mathbb{Z}$ and the cyclic groups $C_n := \mathbb{Z}/n\mathbb{Z}$. Given any ring $\mathcal{R}$, the n by n matrices with coefficients in $\mathcal{R}$ equipped with the standard addition and matrix multiplication form a ring, which we denote by $\mathcal{M}_n(\mathcal{R})$. Lastly, we note that every field is a ring.

Definition 4.2. Let $\mathcal{R}$ be a ring and let $(M, +)$ be an abelian group that is equipped with an operation $\cdot : M \times \mathcal{R} \to M$. M is called a *(left) $\mathcal{R}$-module* if the following conditions are satisfied for all $x, y \in M$ and $r, s \in \mathcal{R}$:

1. $r \cdot (x + y) = r \cdot x + r \cdot y$
2. $(r + s) \cdot x = r \cdot x + s \cdot x$

3. $(r \cdot s) \cdot x = r \cdot (s \cdot x)$
4. $1 \cdot x = x$

We see that if $\mathcal{R}$ is a field, then a $\mathcal{R}$-module is nothing more than a vector field. Next to that, we note that every abelian group A is a $\mathbb{Z}$ module. Moreover, we note that every ideal of a ring is a module of that ring.

Given two left-modules N, M such that $N \subseteq M$. N is a submodule of M if it is a subgroup of M that is closed under multiplication. Supposing that N is a submodule of M, we can now define the quotient module M/N as the quotient group equipped with the $\mathcal{R}$-action $r\cdot(m+N) = r\cdot m+N$ for all $r \in \mathcal{R}, m \in M$.

For the categorically oriented reader, we note that given a ring $\mathcal{R}$, its left-modules form a category, which we denote by $\mathcal{R}$**-Mod**. Let us therefore recall the definition of morphisms of ring modules.

Definition 4.3. Let M, N be $\mathcal{R}$-modules. A map $f : M \to N$ is called a *morphism of $\mathcal{R}$-modules* if for all $m, m' \in M$ and for all $r \in \mathcal{R}$, the following two statements hold:

1. $f(m + m') = f(m) + f(m')$
2. $f(r \cdot m) = r \cdot f(m)$

We note that f is an *isomorphism of $\mathcal{R}$-modules* if it is also bijective.

Before we proceed, let us note that one can still speak of kernels and images of a morphism of left-modules, and that, moreover, these are also proper submodules of the domain and the codomain, respectively.

There is a well-behaved class of ring modules, called *free modules*, that will be of interest to us in the next chapter.

Definition 4.4. Let M be a $\mathcal{R}$-module, and let $(x_i)_{i\in\mathcal{I}}$ be an indexed family of elements in M. These elements are called a *basis* of M if for every $x \in M$, there is a unique family of elements $(r_j)_{j\in\mathcal{J}}$ in $\mathcal{R}$ with finite index set $\mathcal{J} \subseteq \mathcal{I}$ such that:

$$x = \sum_{j\in\mathcal{J}} r_j \cdot x_j \tag{4.1}$$

Lastly, we call M a *free left $\mathcal{R}$-module* if M admits a basis.

Having recalled all the basic definitions, we note that one can also define the concept of a *right $\mathcal{R}$-module* by letting $\mathcal{R}$ act on M on the right. Furthermore, one can also define *bimodules*, which are abelian groups that are left- and right-modules such that the left and right $\mathcal{R}$ actions commute. As we will only be interested in commutative rings, however, we do not worry about this distinction, as for such rings, every left-module is a right-module, and therefore also a bimodule.

By restricting ourselves to discussing just commutative rings, we can avoid a very technical treatment of the notion of a tensor product of modules. We thus present the definition of tensor products within this context, but warn the reader that this definition does not make sense as soon as we consider non-abelian rings (for a general treatment of this theory, see e.g. [Len18]).

Definition 4.5. Let $\mathcal{R}$ be a **commutative** ring. Let M, N be two (left) $\mathcal{R}$-modules. We can then define the tensor product of these two modules, $M \otimes N$, as the following quotient of a free abelian group:

$$M \otimes N := \mathcal{R}[M \times N]/\sim \tag{4.2}$$

where we mod out the following relations for all $m, m' \in M$, $n, n' \in N$ and $r \in \mathcal{R}$:

1. $(m + m', n) \sim (m, n) + (m', n)$
2. $(m, n + n') \sim (m, n) + (m, n')$
3. $(rm, n) \sim (m, rn)$

4.2. Chain Complexes of Ring Modules and Their Homology

In this section, we generalise the notions from homological algebra over vector fields to ring modules. We let $\mathcal{R}$ denote a ring.

We first note that one can define a chain complex of ring-modules in a similar manner as was done in the case of chain complexes of abelian groups.

Definition 4.6. A *chain complex* (of left $\mathcal{R}$-modules) $(\mathcal{C}_*, \partial_*)$ is a family of left-$\mathcal{R}$-modules C_n (for $n \in \mathbb{Z}$), together with ring module morphisms $\partial_n : C_n \to C_{n-1}$ called *differentials*, such that:

$$\partial_n \circ \partial_{n+1} = 0 \quad \forall n \in \mathbb{Z} \tag{4.3}$$

As before, we also have the notion of a morphism of chain complexes over ring modules.

Definition 4.7. Given two chain complexes of left-$\mathcal{R}$-modules, $(\mathcal{C}_*, \partial_*^C)$ and $(\mathcal{D}_*, \partial_*^D)$. A collection of left-$\mathcal{R}$-module morphisms $f_n : C_n \to D_n$ is a *morphism of chain complexes of left-$\mathcal{R}$-modules* if, for every $n \in \mathbb{Z}$, the following diagram commutes:

$$\begin{tikzcd} C_n \arrow[r, "\partial_n^C"] \arrow[d, "f_n"'] & C_{n-1} \arrow[d, "f_{n-1}"] \\ D_n \arrow[r, "\partial_n^D"'] & D_{n-1} \end{tikzcd}$$

Moreover, we can also lift the theory of homology groups to ring modules in a straight-forward manner.

Definition 4.8. The n^{th} *homology group* of the chain complex of left $\mathcal{R}$-models $(\mathcal{C}_*, \partial_*)$ is given by:

$$\mathcal{H}_n(\mathcal{C}) = \ker(\partial_n)/\operatorname{Im}(\partial_{n+1}) \tag{4.4}$$

We note that within the context of left-ring modules, homology groups are not just groups, rather, they are left-ring modules as well. As such, we actually have a homology functor going from the category of chain complexes over ring modules, **Ch**($\mathcal{R}$-**Mod**), to the category of ring modules $\mathcal{H}_n$: **Ch**($\mathcal{R}$-**Mod**) $\to$ $\mathcal{R}$-**Mod**.

As has probably become abundantly clear to the reader, the basic objects of the theory of homological algebra over ring modules are defined in precisely the same manner as those of the homology over abelian groups. We thus simply note that the concepts of an exact sequence of ring modules, of a short and long exact sequence of ring modules (SES and LES) and of tensor products of chain complexes of ring modules can also be lifted effortlessly to the context of ring module homology, and, as a consequence, theorems like the SES implies LES property and the Mayer-Vietoris sequence are all still valid within the context of ring module homology.

Given how effortlessly one can generalise these notions, one may wonder whether this can be drawn any further. We note that this is indeed the case — one can devise a theory of homological algebra over very abstract categories — so-called *abelian categories* — by simply replacing the word “abelian group” by the word “abelian category” in all the definitions we have seen in the theory of homological algebra over abelian groups. The reader eager to know the details of this construction is referred to Appendix A, where we present the basic theory of abelian categories. For completeness, we simply remark, however, that by the *Freyd-Mitchell embedding theorem*, we do not lose much generality by restricting ourselves to the category of ring modules (see Appendix A for the theorem and further details).

4.3. Torsion and the Künneth Theorem

The tensor product of vector spaces is an exact functor. That is, given a SES of vector spaces, e.g.:

$$\begin{tikzcd} 0 \arrow[r] & V_1 \arrow[r] & V_2 \arrow[r] & V_3 \arrow[r] & 0 \end{tikzcd}$$

The sequence we obtain by taking the tensor product with some other vector space W

$$0 \longrightarrow V_1 \otimes W \longrightarrow V_2 \otimes W \longrightarrow V_3 \otimes W \longrightarrow 0$$

is still a proper SES.

This is, however, no longer the case when one works over general ring modules — in this case, the tensor product is only *right-exact*. Let us illustrate this with an example. Consider the following SES:

$$0 \longrightarrow \mathbb{Z} \xrightarrow{2} \mathbb{Z} \longrightarrow \mathbb{Z}/2\mathbb{Z} \longrightarrow 0$$

After taking the tensor product of this sequence with $\mathbb{Z}/2\mathbb{Z}$, the map induced by $\mathbb{Z} \to \mathbb{Z} : n \mapsto 2n$ no longer has a non-trivial kernel — instead, the kernel is now all of $\mathbb{Z}/2\mathbb{Z}$. Therefore, if we want to recover an exact sequence, we would need to extend the newly found sequence:

$$0 \longrightarrow \mathbb{Z}/2\mathbb{Z} \longrightarrow \mathbb{Z}/2\mathbb{Z} \xrightarrow{2} \mathbb{Z}/2\mathbb{Z} \longrightarrow \mathbb{Z}/2\mathbb{Z} \longrightarrow 0$$

As the Künneth formula deals with tensor products of chain complexes, the more general Künneth formula over ring modules will have to take this effect into account. This effect, which is called torsion, is encoded in the so-called Tor-functor. We therefore work towards the construction of this functor. To do so, we must first formalise the idea of extending sequences to make them exact. In every definition, we take $\mathcal{R}$ to be a ring, and take M to be a (left) $\mathcal{R}$-module.

Definition 4.9. A left $\mathcal{R}$-module P is called *projective* if for every surjective map $f : A \to B$ and every map $\pi : P \to B$, it satisfies the following universal property: there is a map $\overline{\pi} : P \to A$ such that the following diagram commutes:

$$\begin{tikzcd} & P \arrow[dl, dashed, "\overline{\pi}"'] \arrow[d, "\pi"] \\ A \arrow[r, two heads, "f"'] & B \end{tikzcd}$$

When considering the category of $\mathcal{R}-$modules, one can see that an object is projective if it is a direct summand of a free $\mathcal{R}$-module. Therefore, we see that this category has enough projectives, that is, we see that for every left-module M, there exists a projective module P together with a surjective map $P \to M$.

Given a left-module A, we would like to extend A by a chain complex of projectives.

Definition 4.10. Let A be a left-module, then a *left-resolution* of A is a chain complex $\mathcal{P}_*$ along with an *augmentation* $\epsilon : P_0 \to A$ such that the following is exact:

$$\dots \xrightarrow{\partial_3} P_2 \xrightarrow{\partial_2} P_1 \xrightarrow{\partial_1} P_0 \xrightarrow{\epsilon} A \longrightarrow 0$$

If, moreover, $\mathcal{P}_*$ is a chain complex of projectives, such a left-resolution is called a *projective resolution*.

It should be noted that as the category of ring modules has enough projectives, one can always construct projective resolution inductively for any given element of the category. One is left to wonder, however, whether projective resolutions behave well in relation to homology. For this, we turn to the following theorem:

Theorem 4.1. *Let A, B be two $\mathcal{R}$-modules. Given two projective resolutions $P \overset{\epsilon}{\to} A$ and $Q \overset{\eta}{\to} B$, and let $f : A \to B$ be a morphism in $\mathcal{A}$. Then there exists a chain map $\overline{f} : P \to Q$ such that the following diagram commutes:*

$$\begin{tikzcd} \dots \arrow[r] & P_2 \arrow[r] \arrow[d, "\bar{f}_2"'] & P_1 \arrow[r] \arrow[d, "\bar{f}_1"'] & P_0 \arrow[r, "\epsilon"] \arrow[d, "\bar{f}_0"'] & A \arrow[r] \arrow[d, "f"'] & 0 \\ \dots \arrow[r] & Q_2 \arrow[r] & Q_1 \arrow[r] & Q_0 \arrow[r, "\eta"] & B \arrow[r] & 0 \end{tikzcd}$$

Moreover, this map is unique up to chain homotopy.

Taking $B = A$ and Q to be another projective resolution for A, one sees that there is a chain isomorphism between the two projective resolutions, and hence, we conclude that the two resolutions are — at least, homologically speaking — identical. This allows us to unambiguously define the notion of a left-derived functor, of which the Tor-functor is a specific example.

Definition 4.11. Let $F : \mathcal{R}$**-Mod** $\to \mathcal{R}$**-Mod** be a right-exact functor. For an arbitrary left-module A, take a projective resolution $P \overset{\epsilon}{\to} A$. The *left-derived functors* of F, L_nF $(n \geq 0)$ are defined as follows:

$$L_nF(A) = \mathcal{H}_n(F(P)) \tag{4.5}$$

For reasons of clarity and brevity, we will refrain from explaining all the relevant mathematical details here[1], and we will simply note that, given that the left-derived functors of a functor F exist, we are ensured of the fact that every SES

$$\begin{tikzcd} 0 \arrow[r] & A \arrow[r] & B \arrow[r] & C \arrow[r] & 0 \end{tikzcd}$$

Induces a LES of the following form:

$$\begin{tikzcd} \cdots \arrow[r] & L_2F(C) \arrow[r] & L_1F(A) \arrow[r] & L_1F(B) \arrow[r] & L_1F(C) \arrow[r] & F(A) \arrow[r] & F(B) \arrow[r] & F(C) \arrow[r] & 0 \end{tikzcd}$$

We can now define the Tor-functor.

Definition 4.12. Let $B \in \mathcal{R} -$ **Mod**, and consider the right-exact functor F : **Mod** $- \mathcal{R} \to$ **Ab** : $A \mapsto A \otimes_{\mathcal{R}} B$. We define the n^{th} Tor-functor as follows:

$$\text{Tor}_n^{\mathcal{R}}(A,B) = L_n(F(A)) \tag{4.6}$$

The property of left-derived functors we stated implies that, for every $A_1, A_2, A_3, B \in \mathcal{R} -$ **Mod** such that the following is a SES:

$$\begin{tikzcd} 0 \arrow[r] & A_3 \arrow[r] & A_2 \arrow[r] & A_1 \arrow[r] & 0 \end{tikzcd}$$

There is a LES of the following form (see e.g. [Wei94]):

$$\begin{tikzcd} \cdots \arrow[r] & \text{Tor}_2^{\mathcal{R}}(A_1,B) \arrow[r] & \text{Tor}_1^{\mathcal{R}}(A_3,B) \arrow[r] & \text{Tor}_1^{\mathcal{R}}(A_2,B) \arrow[d] & & \\ & & & \text{Tor}_1^{\mathcal{R}}(A_1,B) \arrow[d] & & \\ & & & A_3 \otimes B \arrow[d] & & \\ & & & A_2 \otimes B \arrow[r] & A_1 \otimes B \arrow[r] & 0 \end{tikzcd}$$

Hence, we see that the Tor-functor allows us to determine the extension of a SES after applying the tensor product with another module. Moreover, we see that in the example outlined in the introduction, $\mathbb{Z}/2\mathbb{Z}$ was precisely $\text{Tor}_1^{\mathbb{Z}/2\mathbb{Z}}(\mathbb{Z}/2\mathbb{Z}, \mathbb{Z}/2\mathbb{Z})$, and that $\text{Tor}_1^{\mathbb{Z}/2\mathbb{Z}}(\mathbb{Z}, \mathbb{Z}/2\mathbb{Z}) = 0$. One might (rightly) expect the modules satisfying the latter property to form an interesting class of objects, because they allow us to use the Künneth formula we derived in the vector space case. Such modules are called *flat* left $\mathcal{R}$-modules, and are defined as the modules for which $- \otimes_{\mathcal{R}} B$ is exact. While all projective modules are flat modules (simply take the projective resolution $P_n = \delta_{n0}P$), not all flat modules are projective modules. Using the Tor-functor, we can thus characterise flat $\mathcal{R}$-modules.

Theorem 4.2. *Let B be a left $\mathcal{R}$-module. Then:*

$$B \textit{ is flat} \iff \forall n \geq 1, \forall A : \textit{Tor}_n^{\mathcal{R}}(A,B) = 0 \iff \forall A : \textit{Tor}_1^{\mathcal{R}}(A,B) = 0. \tag{4.7}$$

[1]Formally, this comes down to the fact that left-derived functors are δ-functors. The formal treatment of these details can be found in [Wei94].

We are now ready to proceed a more general form of the Künneth formula. Before doing so, however, we remark that one can define various notions that are dual to the ones introduced here: *injectives* and *injective resolutions*, *right-derived functors* and the Ext-functor. These concepts form the building blocks of the cohomology theory over ring modules. Hence, we omit them from our discussion, and refer the interested reader to [Wei94].

Theorem 4.3. *Let $\mathcal{C}_*, \mathcal{D}_*$ be chain complexes of right and left $\mathcal{R}$-modules, respectively, such that C_n and $\partial_n(C_n)$ are flat $\forall n \geq 0$. Then the following short sequence is exact:*

$$0 \longrightarrow \bigoplus_{i=1}^{n} \mathcal{H}_i(\mathcal{C}) \otimes \mathcal{H}_{n-i}(\mathcal{D}) \longrightarrow \mathcal{H}_n(\mathcal{C} \otimes_{\mathcal{R}} \mathcal{D}) \longrightarrow \bigoplus_{i=1}^{n-1} \mathit{Tor}_1^{\mathcal{R}}(\mathcal{H}_i(\mathcal{C}), \mathcal{H}_{n-1-i}(\mathcal{D})) \longrightarrow 0$$

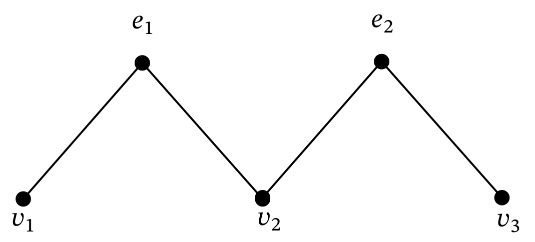
Note that for vector spaces, the tensor product is indeed flat, and hence the first Tor-functors vanish, such that the SES induces an isomorphism $\mathcal{H}_n(C \otimes_{\mathcal{R}} \mathcal{D}) \simeq \bigoplus_{i=1}^{n} \mathcal{H}_i(\mathcal{C}) \otimes \mathcal{H}_{n-i}(\mathcal{D})$, as expected.

5 Lifted Product Codes

We now turn to the *lifted product* of Panteleev and Kalachev [PK22b]. This product can be interpreted as a generalisation of the hypergraph product. We first explain why this is the case by taking on the Tanner graph perspective on these codes. Afterwards, we will take on a homological point of view, and show how the lifted product can be interpreted in terms of homology over ring modules. We prove a Künneth formula for these codes, and present an approach that allows one to calculate the number of encoded qubits of these codes more easily.

5.1. Motivation using Tanner Graphs

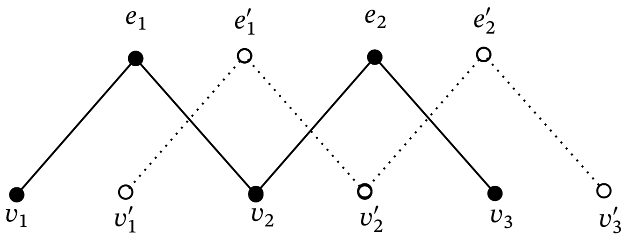
The lifted product of two codes is constructed by first "lifting" their Tanner graphs to much larger Tanner graphs. We can try to visualise this idea using a small example. To this end, consider the Tanner graph of the repetition code with open boundaries, as displayed in Figure 5.1.



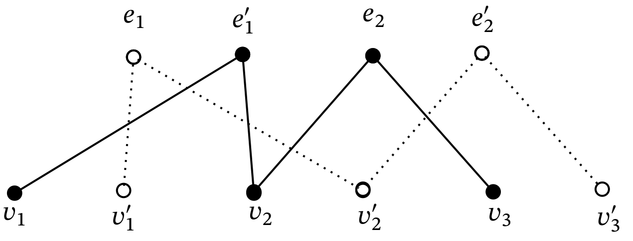
$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Figure 5.1: The Tanner graph of the repetition code with open boundaries, along with the corresponding parity check matrix.

We now show how to perform a 2-lift: first, we make a copy of the graph, such that for every edge (v_i, e_j) there is now a copy (v_i', e_j'). Now, we have the freedom to apply permutations between the edges (v_i, e_j) and (v_i', e_j') — we could, for instance, choose to switch (cq. permute) e_1 and e_1'. We consider the effects of two examples in Figure 5.2: a trivial 2-lift, and a 2-lift where we apply the aforementioned permutation.



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



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

Figure 5.2: Two examples of a lift of the Tanner graph of the repetition code with open boundaries. The first of these corresponds to a trivial lift, where one just copies the vertices and edges of the Tanner graph. The second example corresponds to a twisted lift, as the edge (e_1, v_1) and the edge (e_1', v_1') are twisted.

Note that these adjacency matrices can be recovered from the adjacency matrix of the initial Tanner graph: if we replace the 1's in the initial adjacency matrix by the 2 by 2 identity matrix, and all the 0's by the 2 by 2 zero matrix, we find the adjacency matrix of the trivial 2-lift!
Furthermore, we can retrieve the adjacency matrix of the twisted 2-lift by replacing the 1's in the initial adjacency matrix by the identity matrix for the unpermuted elements, and by the permutation matrix for the permuted elements.

This is an example of a more general concept: we can construct an *l-lift* of a graph by making $l-1$ copies of the graph and by additionally applying permutations between copies of the same edges. If these permutations are contained within a subgroup Γ of the permutation group S_l such that $|\Gamma| = l$, we speak of a *Γ-lift* of the graph. Moreover, if Γ is generated by the 1-shift $(123\dots l)$, we speak of an *l-shift*. We will be considering the group algebra $\mathbb{F}_2\Gamma$, which, given that it possesses a lot of algebraic structure, allows us to take tensor products of the chain complexes associated to the Tanner graphs of lifted codes in order to construct new codes — this is the central idea behind the lifted product construction.

Before proceeding, we note that one can use the l-shifts to generate an important class of examples of classical codes, which are often referred to as *quasicyclical codes*. Such l-shifts are formalised using the group algebra of the cyclic groups $\mathcal{R} := \mathbb{F}_2[C_l]$. We note for future reference that $\mathbb{F}_2[C_l] \simeq \mathbb{F}_2[x]/(x^l - 1)$. We will consider this important class of examples in more detail throughout our discussion.

5.2. The Homological Construction

We first present the formal construction of the lifted product using a homological approach. In what is to come, we let $\mathcal{R}$ denote a commutative, unital, l-dimensional, associative $\mathbb{F}_2$-algebra. Of course, one immediately wonders why we are even considering such an $\mathbb{F}_2$-algebra in the first place. The reason is that $\mathcal{R}$ admits a faithful representation $\mathbb{B} : \mathcal{R} \to \mathrm{End}(\mathbb{F}_2^l)$, that, in turn, allows us to think of the elements of $\mathcal{R}$ as l by l matrices over $\mathbb{F}_2$, and hence, such $\mathbb{F}_2$-algebras indeed allow us to capture the structure needed in the constructions in the motivating examples. We will provide a rigorous account of these results later on, but first, let us take the existence of such a representation as given and present the construction of the lifted product from a homological perspective.

To start off, we first note that $\mathcal{R}$, as it is a commutative, unital, associative $\mathbb{F}_2$-algebra, is actually a commutative ring. Therefore, we can consider maps between free modules of $\mathcal{R}$:

$$\begin{tikzcd} \mathcal{R}^{r_1} \arrow[r, "H_1^T"] & \mathcal{R}^{n_1} \end{tikzcd}$$

$$\begin{tikzcd} \mathcal{R}^{n_2} \arrow[r, "H_2"] & \mathcal{R}^{r_2} \end{tikzcd}$$

We see that this closely resembles the setting of the hypergraph product. Therefore, it is instructive to consider the tensor product of these complexes:

$$\begin{tikzcd} \mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{n_2} \arrow[r, "\partial_2^{\mathcal{R}}"] & \mathcal{R}^{n_1} \otimes_{\mathcal{R}} \mathcal{R}^{n_2} \oplus \mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{r_2} \arrow[r, "\partial_1^{\mathcal{R}}"] & \mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{r_2} \end{tikzcd}$$

We end up with a length 2 chain complex over $\mathcal{R}$. To define a CSS code, however, we would need a chain complex over $\mathbb{F}_2$. To this end, we can evoke our representation $\mathbb{B}$:

$$\begin{tikzcd} \mathbb{B}\left(\mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{n_2}\right) \arrow[r, "\mathbb{B}(\partial_2^{\mathcal{R}})"] & \mathbb{B}\left(\mathcal{R}^{n_1} \otimes_{\mathcal{R}} \mathcal{R}^{n_2} \oplus \mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{r_2}\right) \arrow[r, "\mathbb{B}(\partial_1^{\mathcal{R}})"] & \mathbb{B}\left(\mathcal{R}^{r_1} \otimes_{\mathcal{R}} \mathcal{R}^{r_2}\right) \end{tikzcd}$$

But, in order to do so, we need to be able to lift $\partial_i^{\mathcal{R}}$ while preserving the (homological) structure of the complex. We will explain how to do so rigorously in the next section, by delving more deeply into the nature of the representation $\mathbb{B}$.

5.3. Understanding the Representation $\mathbb{B}$

Let us now construct this faithful representation $\mathbb{B} : \mathcal{R} \to \text{End}(\mathbb{F}_2^l)$ explicitly. As $\mathcal{R}$ is an l-dimensional $\mathbb{F}_2$-algebra, we can find a basis $\{r_1, \dots, r_l\}$ of $\mathcal{R}$. Let us define a map $\mathbb{b} : \mathcal{R} \to \mathbb{F}_2^l$, which is the linear extension of the identification $r_i \mapsto e_i$. This map allows us to define $\mathbb{B}$ as the linear extension of the map for which the following expression holds:

$$\mathbb{b}(r_i r_j) = \mathbb{B}(r_i)\mathbb{b}(r_j) \quad \forall i, j \tag{5.1}$$

Note, moreoever, that $\mathbb{B}$ is a homomorphism, as:

$$\mathbb{B}(r_i r_j)\mathbb{b}(r_k) = \mathbb{b}(r_i r_j r_k) = \mathbb{B}(r_i)\mathbb{b}(r_j r_k) = \mathbb{B}(r_i)\mathbb{B}(r_j)\mathbb{b}(r_k) \tag{5.2}$$

This construction may seem to be basis dependent. Fortunately, it is not, due to the linear nature of all maps involved.

Let us go back to our motivating example of the non-trivial 2-lift we worked out above. In this case, one finds that the homomorphism $\mathbb{B} : \mathbb{F}_2[x]/(x^2-1) \to \text{End}(\mathbb{F}_2^2)$ is given by the map:

$$[1] \mapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad [x] \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{5.3}$$

which is exactly in line with our example! Similarly, it can be shown that for the Galois field $\text{GF}(4) := \{[0], [1], [x]\,[x^2]\}$, which can be interpreted as an $\mathbb{F}_2$-algebra of dimension 2, one has the following representation:

$$[0] \mapsto \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad [1] \mapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{5.4}$$

$$[x] \mapsto \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, \quad [x^2] \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \tag{5.5}$$

Having constructed this representation $\mathbb{B}$, one can easily generalise this to free modules $\mathcal{R}^n$ by making the simple observation that $\mathcal{R}^n \simeq \mathcal{R} \otimes_{\mathbb{F}_2} \mathbb{F}_2^{\alpha}$. This observation also allows us to determine what $\mathbb{B}(\partial_i^{\mathcal{R}})$

must be. Better yet, we can tackle the more general problem of determining what $\mathbb{B}(H)$ should be for any map $H : \mathcal{R}^n \to \mathcal{R}^m$. To this end, let us try to understand $\mathbb{B}(H)$ with the following commutative diagram:

$$\begin{tikzcd}
\mathcal{R}^n \arrow[r, "H"] \arrow[d, "(r_1,\dots,r_n)\mapsto\sum_{i=1}^n r_i\otimes e_i"'] & \mathcal{R}^m \arrow[d, "(r_1,\dots,r_m)\mapsto\sum_{j=1}^m r_j\otimes f_j"] \\
\mathcal{R}\otimes\mathbb{F}_2^n \arrow[r, "\tilde{H}"] \arrow[d, "\mathbb{B}\otimes\mathrm{id}"'] & \mathcal{R}\otimes\mathbb{F}_2^m \arrow[d, "\mathbb{B}\otimes\mathrm{id}"] \\
\mathbb{B}(\mathcal{R})\otimes\mathbb{F}_2^n \arrow[r, "\mathbb{B}(H)"] & \mathbb{B}(\mathcal{R})\otimes\mathbb{F}_2^m
\end{tikzcd}$$

As $\mathbb{B}$ is faithful, it is an isomorphism between its domain and its image. Hence, we can easily define $\mathbb{B}(H)$ as:

$$\mathbb{B}(H) = (\mathbb{B}\otimes\mathrm{id})\tilde{H}(\mathbb{B}^{-1}\otimes\mathrm{id}) \tag{5.6}$$

All of this means that if we can understand $\tilde{H}$, we are done. This is quite simple, however: given a vector

$$\mathbf{r}_i := (0,\dots,0,r,0,\dots,0)^T \in \mathcal{R}^n \tag{5.7}$$

Then:

$$H\mathbf{r}_i = (H_{1i}r,\dots,H_{mi}r)^T \in \mathcal{R}^m \tag{5.8}$$

Chasing through the diagram, we see that:

$$\tilde{H}(r\otimes e_i) = \sum_{j=1}^m H_{1i}\cdot r\otimes f_j \tag{5.9}$$

But then, we see that:

$$\mathbb{B}(H)(\mathbb{B}(r)\otimes e_i) = \sum_{j=1}^m \mathbb{B}(H_{1i}r)\otimes f_j = \sum_{j=1}^m \mathbb{B}(H_{1i})\mathbb{B}(r)\otimes f_j \tag{5.10}$$

Hence, we see that $\mathbb{B}(H)$ takes on quite a simple form, namely:

$$\mathbb{B}(H) = \begin{pmatrix} \mathbb{B}(H_{11}) & \cdots & \mathbb{B}(H_{1m}) \\ & \vdots & \\ \mathbb{B}(H_{n1}) & \cdots & \mathbb{B}(H_{nm}) \end{pmatrix} \tag{5.11}$$

Defining $\mathbb{B}(H)$ in such a manner is quite useful, as it immediately shows us that the lift $\mathbb{B}$ preserves the homological structure of the differentials. We can work out a small example to see this: let us take $\mathcal{R} = \mathbb{F}_2[C_4] = \mathbb{F}_2[\sigma_0,\dots,\sigma_3]$. We want to find a relation between the homology of $\mathbb{B}(H)$ and H itself, where $H : \mathcal{R}\to\mathcal{R}$ is taken to be the left-multiplication with $\sigma_1+\sigma_3$.
First note that the kernel of H is equal to $\ker(H) = \mathbb{F}_2[\sigma_1+\sigma_3, \sigma_0+\sigma_2]$. Furthermore, we see that the image of H is also $\mathrm{Im}(H) = \mathbb{F}_2[\sigma_1+\sigma_3,\sigma_0+\sigma_4]$. Furthermore, we find that:

$$\mathbb{B}(H) = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \tag{5.12}$$

One can see that:

$$\ker(\mathbb{B}(H)) = \mathrm{Im}(\mathbb{B}(H)) = \mathrm{Span}_{\mathbb{F}_2}\left\{\begin{pmatrix}1\\0\\1\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\\1\end{pmatrix}\right\} \tag{5.13}$$

We thus see the symmetry between the kernels and images of H and $\mathbb{B}(H)$ appear: by identifying $\mathcal{R}$ with $\mathbb{F}_2^4$ with the map $\sigma_i \mapsto e_i$, we actually find vector space isomorphisms $\ker(H) = \mathrm{Im}(H) \simeq \mathrm{Im}(\mathbb{B}(H)) = \ker(\mathbb{B}(H))$, and therefore, we see that also $H_1(\mathbb{B}(H)) \simeq H_1(H)$.

5.4. An Explicit Expression for H_X and H_Z

Given two parity check matrices H_1 and H_2, we could find an explicit expression for the parity check matrices H_X and H_Z in terms of these matrices. Ideally, this would still be the case for our lifted product codes, as we could then try to e.g. calculate the distance or weight in a similar manner. We show that this fundamentally requires that $\mathcal{R}$ be commutative.
Let us now consider two matrices over $\mathcal{R}$:

$$H_1 \in \mathcal{M}_{r_1 \times n_1}(\mathcal{R}), H_2 \in \mathcal{M}_{n_2 \times r_2}(\mathcal{R}) \tag{5.14}$$

Given these two matrices, we can now construct the parity check matrices H_X and H_Z using the tensor product of the underlying chain complexes of these matrices:

$$H_X = \left(H_1 \otimes_{\mathcal{R}} I_{n_2} | I_{r_1} \otimes_{\mathcal{R}} H_2^T\right), \quad H_Z = \left(I_{n_1} \otimes_{\mathcal{R}} H_2 | H_1^T \otimes_{\mathcal{R}} I_{r_2}\right) \tag{5.15}$$

Using the representation $\mathbb{B}$, we can lift these matrices to matrices over $\mathbb{F}_2$:

$$\mathbb{B}(H_X) \in \mathcal{M}_{lr_1n_2 \times l(n_1n_2 + r_1r_2)}(\mathbb{F}_2), \quad \mathbb{B}(H_Z) \in \mathcal{M}_{ln_1r_2 \times l(n_1n_2 + r_1r_2)}(\mathbb{F}_2) \tag{5.16}$$

We have now obtained the lifted product of the two codes, which has parity check matrices $\mathbb{B}(H_X)$ and $\mathbb{B}(H_Z)$, provided that $\mathbb{B}(H_X)\mathbb{B}(H_Z)^T = 0$. As $\mathbb{B}$ is a faithful representation, this is equivalent to requiring that $H_X H_Z^T = 0$. This, however, requires that H_X and H_Z be entry-wise commuting matrices (and hence, that $\mathcal{R}$ is itself commutative). Let us first write out the product:

$$H_X H_Z^T = \left(H_1 \otimes I_{n_2} | I_{r_1} \otimes H_2^T\right) \begin{pmatrix} I_{n_1} \otimes H_2^T \\ H_1 \otimes I_{r_2} \end{pmatrix} = \left(H_1 \otimes I_{n_2}\right)\left(I_{n_1} \otimes H_2^T\right) + \left(I_{r_1} \otimes H_2^T\right)\left(H_1 \otimes I_{r_2}\right) \tag{5.17}$$

Now, we can consider each of the terms separately using the Kronecker product:

$$\left(H_1 \otimes I_{n_2}\right)\left(I_{n_1} \otimes H_2^T\right) = \begin{pmatrix} (H_1)_{11} I_{n_2} & \dots & (H_1)_{1n_1} I_{n_2} \\ & \ddots & \\ (H_1)_{r_11} I_{n_2} & \dots & (H_1)_{r_1n_1} I_{n_2} \end{pmatrix} \begin{pmatrix} H_2^T & & \varnothing \\ & \ddots & \\ \varnothing & & H_2^T \end{pmatrix} = \begin{pmatrix} (H_1)_{11} H_2^T & \dots & (H_1)_{1n_1} H_2^T \\ & \ddots & \\ (H_1)_{r_11} H_2^T & \dots & (H_1)_{r_1n_1} H_2^T \end{pmatrix} \tag{5.18}$$

Similarly, we find:

$$\left(I_{r_1} \otimes H_2^T\right)\left(H_1 \otimes I_{r_2}\right) = \begin{pmatrix} H_2^T (H_1)_{11} & \dots & H_2^T (H_1)_{1n_1} \\ & \ddots & \\ H_2^T (H_1)_{r_11} & \dots & H_2^T (H_1)_{r_1n_1} \end{pmatrix} \tag{5.19}$$

For this to be zero, we indeed see that H_1 and H_2 must commute elementwise.

Now that we have an explicit expression for the parity check matrices, we note that this lifted product construction is indeed a generalisation of the hypergraph product construction: we can recover the hypergraph product by simply taking $\mathcal{R} = \mathbb{F}_2$ — the restriction on H_1 and H_2 is then automatically satisfied.

5.5. The Weight of Lifted Product Codes

Given that we understand the construction of the lifted product codes, we now determine their parameters. We start off by determining the weight of the lifted product in the most general case: let

$G = \{g_1, \dots, g_l\}$ be a finite, abelian group and consider the F_2-algebra $\mathcal{R} := \mathbb{F}_2 G$. Given an element in $\mathcal{R}$, which we denote by $a := \sum_{i=1}^{l} \alpha_i g_i$, we can use the map $\mathbb{b}$ to write:

$$\mathbb{b}(a) = \begin{pmatrix} \alpha_1 & \dots & \alpha_l \end{pmatrix} \tag{5.20}$$

Furthermore, by definition of $\mathbb{B}$, the following expression holds:

$$\mathbb{B}(a) = \sum_{i=1}^{l} \alpha_i \mathbb{B}(g_i) \tag{5.21}$$

where $\mathbb{B}(g_k)$ can be explicitly determined as the matrix satisfying $[\mathbb{B}(g_i)]_{jk} = 1$ if and only if $g_j = g_i g_k$. From this, one can now easily determine the weight of $\mathbb{B}(a)$: Firstly, note that $\sum_j \mathbb{B}(g_i)_{jk} = 1$, as there is precisely one g_j such that $g_j = g_i g_k$. Thus, we find:

$$\sum_{j=1}^{l} \mathbb{B}(a)_{jk} = \sum_{i=1}^{l} \alpha_i \left(\sum_{j=1}^{l} \mathbb{B}(g_i)_{jk} \right) = \sum_{i=1}^{l} \alpha_i = \text{wgt}(\mathbb{b}(a)) \tag{5.22}$$

The same results hold when we sum over k, and hence, we conclude that:

$$\text{wgt}(\mathbb{B}(a)) = \text{wgt}(\mathbb{b}(a)) \tag{5.23}$$

We thus conclude that the lifted product construction yields qLDPC codes, given that the input consists of two classical LDPC codes.

5.6. An Algebraic Approach to Determining k: The Case of Quasicyclic Codes

Let us start our discussion of determining the number of encoded qubits by returning to the example of quasicyclic codes. Using an algebraic approach to these codes, we determine the number of encoded qubits of the lifted product.

First, let us take $\mathcal{R} := \mathbb{F}_2[C_l]$, with l odd. By Maschke's theorem[PK22b], we know that:

$$\mathcal{R} \simeq \prod_{i=1}^{k} \mathbb{F}_{2^{s_i}} \tag{5.24}$$

let us write $\varphi_i : \mathcal{R} \to \mathbb{F}_{2^{s_i}}$, where $\mathbb{F}_{2^{s_i}} := \text{GL}(2^{s_i})$. for the composition of the projection and the isomorphism. Therefore, given any two $\mathcal{R}$-matrices A and B, we can represent them using submatrices $\varphi_i(A)$ and $\varphi_i(B)$. The point is, now, that the degeneracy of codewords is respected by the isomorphism, and that therefore, we easily find that:

$$k = \dim(LP(A, B)) = \sum_{i=1}^{k} \dim(LP(\varphi(A_i), \varphi(B_i))) \tag{5.25}$$

This result still holds when l is even, but now, Maschke's theorem cannot be used, but instead, we find a direct product decomposition from the Chinese remainder theorem[LO14]:

$$\mathcal{R} \simeq \prod_{i=1}^{k} \mathcal{R}_i \tag{5.26}$$

We can still define a projection onto each of the rings $\mathcal{R}_i$, and as such, arrive at the same formula for k.

For illustrative purposes, let us work out a small example: let us take $l = 3$ and apply Maschke's theorem to find that:

$$\mathcal{R} := \mathbb{F}_2[x]/(x^3 - 1) \simeq \mathbb{F}_2 \times \mathbb{F}_4 \tag{5.27}$$

Note that this is indeed the case, as we have the following irreducible decomposition: $x^3 + 1 = (x + 1)(x^2 + x + 1)$, and $\mathbb{F}_2[x]/(x+1) \simeq \mathbb{F}_2$, whereas $\mathbb{F}_2[x]/\left(x^2 + x + 1\right) \simeq \mathbb{F}_4$. Now suppose that we want to calculate the lifted product of the following matrix with itself:

$$\begin{pmatrix} [0] & [x^2] \\ [x] & [x^2+1] \end{pmatrix} \tag{5.28}$$

Using the isomorphism, we can decompose this matrix as follows:

$$\begin{pmatrix} [0] & [x^2] \\ [x] & [x^2+1] \end{pmatrix}_{\mathcal{R}} \simeq \begin{pmatrix} [0] & [1] \\ [1] & [0] \end{pmatrix}_{\mathbb{F}_2} \times \begin{pmatrix} [0] & [x^2] \\ [x] & [x] \end{pmatrix}_{\mathbb{F}_4} \tag{5.29}$$

The lifted product of this matrix with itself can now be calculated by determining the lifted product of these other matrices with themselves. The first of these reduces to a calculation of the hypergraph product, while the second case can be tackled by using the Kronecker product, and then using the representation $\mathbb{B} : \mathsf{GL}(4) \to \mathsf{End}(\mathbb{F}_2^2)$ that we determined in one of the previous sections.

5.7. A Homological Approach to Determining k

In the case of the hypergraph product, we could determine the number of encoded qubits using purely linear algebraic arguments. One sees, however, that when studying the lifted product code, this is no longer the case: even if we consider a specific subclass of such codes, like the quasicyclic codes, one needs to make use of various results from the field of algebra to find an explicit expression for the number of encoded qubits. One could hope that a different approach to determining k could offer us more solace. As one could also make use of homological arguments for the hypergraph product codes, let us thus try to see whether extending these to the lifted product codes offers us a better tool for this purpose.

5.7.1. The Case of Fields and Principal Ideal Domains

A first attempt at using homology for the purpose of determining the number of encoded qubits would start from considering the structure of $\mathcal{R}$. As there is no ready-made theory of homology for $\mathbb{F}_2$-algebras, one could first try to add more structure to $\mathcal{R}$ and see whether this simplifies our calculations.
To this end, we turn to the case in which $\mathcal{R}$ is a field, as this ensures that the Tor-functor vanishes. We then find the regular Künneth formula:

$$\mathcal{H}_1(A \otimes_{\mathcal{R}} B) \simeq \mathcal{H}_1(A) \otimes_{\mathcal{R}} \mathcal{H}_0(B) \oplus \mathcal{H}_0(A) \otimes_{\mathcal{R}} \mathcal{H}_1(B) \tag{5.30}$$

This theorem, however, is of limited use, as $\mathcal{R}$ should also be an $\mathbb{F}_2$-algebra. Hence, we can fully classify all possible choices of $\mathcal{R}$ satisfying these properties: these are the Galois fields $\mathbb{F}_{2^l} := \mathsf{GF}(2^l)$. Although this theorem offers us a way to simplify calculations, e.g. in the context of the quasicyclic codes discussed in the previous section, we would ideally like to have a more generic form of the Künneth formula.

A second attempt could spring from considering imposing less stringent conditions on $\mathcal{R}$. One idea would be to assume that $\mathcal{R}$ is a principal ideal domain (PID), as over such rings, modules behave nicely — in particular, we know that all submodules of free modules in PIDs are again free, which thus means that the kernels and images of the maps we are considering are free modules. We can therefore invoke the following theorem [Hat01]:

Theorem 5.1. *Let $\mathcal{R}$ be a principal ideal domain, and consider two 2-complexes over $\mathcal{R}$, say $\mathcal{A}$ and $\mathcal{B}$. These complexes satisfy the following Künneth formula:*

$$\mathcal{H}_1(\mathcal{A} \otimes_{\mathcal{R}} \mathcal{B}) = \mathcal{H}_1(A) \otimes_{\mathcal{R}} H_0(B) \oplus \mathcal{H}_0(A) \otimes_{\mathcal{R}} H_1(B) \tag{5.31}$$

And, moreover, this sequence splits non-canonically, i.e., we have that:

$$\mathcal{H}_1(\mathcal{A} \otimes_{\mathcal{R}} \mathcal{B}) \simeq (\mathcal{H}_1(A) \otimes \mathcal{H}_0(B) \oplus \mathcal{H}_0(A) \otimes \mathcal{H}_1(B)) \oplus \left(\mathit{Tor}_1^{\mathcal{R}}(\mathcal{H}_0(A), \mathcal{H}_0(B))\right) \tag{5.32}$$

Note, however, that even this theorem is also of limited use when trying to devise a general formula for the number of encoded qubits. There are two reasons for this: firstly, the requirement that $\mathcal{R}$ be a PID is also too stringent: the previously discussed quasi-cyclical codes, which can be formalised using the group algebra of the cyclic groups $\mathcal{R} := \mathbb{F}_2[C_l]$, are not PIDs: indeed, let $\mathcal{R} \simeq \mathbb{F}_2[x]/(x^l - 1)$. Then $x^l - 1 = (x-1)(\sum_{i=0}^{l-1} x^i)$, hence $x - 1$ is a zero divisor, and thus $\mathcal{R}$ is not a domain, and therefore certainly also not a PID.

Secondly, and more importantly, one cannot determine the $\mathrm{Tor}_1^{\mathcal{R}}$-functor in full generality. Even though the zeroth homology groups $\mathcal{H}_0(A)$ and $\mathcal{H}_0(B)$ are quotient groups of free modules, this does not guarantee that these are well-behaved.

Indeed, let us consider a simple example in which this theorem already fails to deliver useful, general results: Let $H_1 : \mathcal{R} \to \mathcal{R}$ be the left-multiplication with some element $a \in \mathcal{R}$ that is not a zero divisor (as PIDs are in particular domains), i.e. $H_1 : r \mapsto ar$, and let H_2 be some arbitrary map given by the 1-complex $\mathcal{B}$. Then $\mathrm{Im}\,(H_1) = aR$ is an ideal of $\mathcal{R}$, and hence we find the following SES:

$$\begin{tikzcd} 0 \arrow[r] & \mathcal{R} \arrow[r, "H_1"] & \mathcal{R} \arrow[r] & \mathcal{R}/a\mathcal{R} \arrow[r] & 0 \end{tikzcd}$$

which, as $\mathrm{Tor}_1^{\mathcal{R}}(\mathcal{R}, \mathcal{H}_0(B)) = 0$, induces the following LES:

$$\begin{tikzcd} 0 \arrow[r] & \mathrm{Tor}_1^{\mathcal{R}}(\mathcal{R}/a\mathcal{R}, \mathcal{H}_0(B)) \arrow[r] & \mathcal{R} \otimes_{\mathcal{R}} \mathcal{H}_0(B) \arrow[r] & \mathcal{R} \otimes_{\mathcal{R}} \mathcal{H}_0(B) \arrow[r] & \mathcal{R}/a\mathcal{R} \otimes_{\mathcal{R}} \mathcal{H}_0(B) \arrow[r] & 0 \end{tikzcd}$$

We can simplify the central part of the sequence and use the first isomorphism theorem to derive that:

$$\mathrm{Tor}_1^{\mathcal{R}}(\mathcal{R}/a\mathcal{R}, \mathcal{H}_0(B)) \simeq {}_a\mathcal{H}_0(B) := \{x \in \mathcal{H}_0(B) : ax = 0\} \tag{5.33}$$

And, given that $\mathcal{H}_1(A) = \ker\,(x \mapsto ax) = 0$ (as a is not a zero-divisor), we thus find:

$$\mathcal{H}_1(\mathcal{A} \otimes_{\mathcal{R}} \mathcal{B}) \simeq (\mathcal{R}/a\mathcal{R} \otimes_{\mathcal{R}} \mathcal{H}_1(B)) \oplus {}_a\mathcal{H}_0(B) \tag{5.34}$$

Assuming that $H_0(B)$ is a finitely generated module [LO14], the structure theorem for finitely generated modules tells us that there is an $n \in \mathbb{N}$ and there are ideals $I_1, \dots, I_k$, such that:

$$\mathcal{H}_0(B) = \mathcal{R}^n \oplus \left(\bigoplus_{i=1}^{k} \mathcal{R}/I_k \right) \tag{5.35}$$

We thus see that ${}_a\mathcal{H}_0(B)$ is non-trivial when this decomposition contains a summand $\mathcal{R}/a\mathcal{R}$. Hence, we cannot make any more general inferences about the first homology group of $\mathcal{A} \otimes_{\mathcal{R}} \mathcal{B}$.

Both of these concerns can be resolved by going back to the basic structure of $\mathcal{R}$. Let us do just this.

5.7.2. Towards a General Expression for k: A Künneth Theorem

Given that we take $\mathcal{R}$ to be an $\mathbb{F}_2$-module, one knows that $\mathcal{R}$-modules are themselves $\mathbb{F}_2$-modules. But, as $\mathbb{F}_2$ is a field, every $\mathcal{R}$-module is automatically flat with respect to $\mathbb{F}_2$. One may wonder, therefore, whether we cannot apply a change of rings, and show that the flatness with respect to $\mathbb{F}_2$ implies flatness with respect to $\mathcal{R}$. The following theorem allows us to do just that [Wei94]:

Theorem 5.2. *Let $\mathcal{S}$ be a commutative ring, and let $\mathcal{R}$ be a flat $\mathcal{S}$-algebra. Then for every $\mathcal{S}$-module A and B, and for every $n \in \mathbb{N}$, the following holds:*

$$\mathcal{R} \otimes_{\mathcal{S}} Tor_n^{\mathcal{S}}(A, B) \simeq Tor_n^{\mathcal{R}}(A \otimes_{\mathcal{S}} T, B \otimes_{\mathcal{S}} \mathcal{R}) \tag{5.36}$$

Taking $\mathcal{S} := \mathbb{F}_2$, and letting $\mathcal{R}$ denote our $\mathbb{F}_2$-algebra, for any two $\mathcal{R}$-modules, this theorem tells us that:

$$\mathrm{Tor}_1^{\mathcal{R}}(A \otimes_{\mathbb{F}_2} \mathcal{R}, B \otimes_{\mathbb{F}_2} \mathcal{R}) \simeq T \otimes_{\mathbb{F}_2} \mathrm{Tor}_1^{\mathbb{F}_2}(A, B) = 0 \tag{5.37}$$

Furthermore, as $\mathcal{R}$ is a vector space, it is actually a free $\mathbb{F}_2$-module, such that:

$$\text{Tor}_1^{\mathcal{R}}(A \otimes_{\mathbb{F}_2} \mathcal{R}, B \otimes_{\mathbb{F}_2} \mathcal{R}) \simeq \text{Tor}_1^{\mathcal{R}}(A \otimes_{\mathbb{F}_2} \mathbb{F}_2^l, B \otimes_{\mathbb{F}_2} \mathbb{F}_2^l) = \bigoplus_{l^2} \text{Tor}_1^{\mathcal{R}}(A \otimes_{\mathbb{F}_2} \mathbb{F}_2, B \otimes_{\mathbb{F}_2} \mathbb{F}_2) = \bigoplus_{l^2} \text{Tor}_1^{\mathcal{R}}(A, B) \tag{5.38}$$

Hence, we conclude that $\text{Tor}_1^{\mathcal{R}}(A, B) = 0$. As such, we see that, for $\mathbb{F}_2$-algebras, the general Künneth formula for ring modules specialises to the formula we know and love from vector spaces:

$$\mathcal{H}_n(A \otimes_{\mathcal{R}} B) \simeq \bigoplus_{i=0}^{n} \mathcal{H}_i(A) \otimes_{\mathcal{R}} \mathcal{H}_{n-i}(B) \tag{5.39}$$

Based on our analysis of the representation $\mathbb{B}$, and on the Künneth theorem derived above, we find the following expression for k:

$$\begin{aligned} k &= \dim_{\mathbb{F}_2}(\mathcal{H}_1(\mathbb{B}(A \otimes_{\mathcal{R}} B))) && (5.40)\\ &= \dim_{\mathbb{F}_2}(\mathcal{H}_1(A \otimes_{\mathcal{R}} B)) && (5.41)\\ &= \dim_{\mathbb{F}_2}(\mathcal{H}_1(A) \otimes_{\mathcal{R}} \mathcal{H}_0(B)) + \dim_{\mathbb{F}_2}(\mathcal{H}_0(A) \otimes_{\mathcal{R}} \mathcal{H}_1(B)) && (5.42)\end{aligned}$$

Ideally, one would be able to relate the tensor product over $\mathcal{R}$ to the tensor product over $\mathbb{F}_2$, such that the dimension of the $\mathcal{R}$-tensor product of the homology groups can be calculated. Let us do so by restricting our attention to group algebras of finite groups, $\mathcal{R} = \mathbb{F}_2[G]$, and noting that for arbitrary $\mathcal{R}$-modules A, B:

$$A \otimes_{\mathcal{R}} B = \left(A \otimes_{\mathbb{F}_2} B\right)/V \tag{5.43}$$

where:

$$V := \langle ga \otimes gb - a \otimes b \, : \, a \in A, b \in B \rangle, g \in G \tag{5.44}$$

We note that, therefore, $A \otimes_{\mathcal{R}} B$ is equal to $\left(A \otimes_{\mathbb{F}_2} B\right)_G$, the so-called *G-coinvariants* of $A \otimes_{\mathbb{F}_2} B$. Given all of this, we now see that:

$$\begin{aligned} k &= \dim_{\mathbb{F}_2}\left(\left(\mathcal{H}_1(A) \otimes_{\mathbb{F}_2} \mathcal{H}_0(B)\right)_G\right) + \dim_{\mathbb{F}_2}\left(\left(\mathcal{H}_1(A) \otimes_{\mathbb{F}_2} \mathcal{H}_0(B)\right)_G\right) && (5.45)\\ &= \dim_{\mathbb{F}_2}\left(\left(\mathcal{H}_1(A) \otimes_{\mathbb{F}_2} \mathcal{H}_0(B)\right)/V\right) + \dim_{\mathbb{F}_2}\left(\left(\mathcal{H}_0(A) \otimes_{\mathbb{F}_2} \mathcal{H}_1(B)\right)/W\right) && (5.46)\\ &= \dim_{\mathbb{F}_2}\left(\mathcal{H}_1(A) \otimes_{\mathbb{F}_2} \mathcal{H}_0(B)\right) + \dim_{\mathbb{F}_2}\left(\mathcal{H}_0(A) \otimes_{\mathbb{F}_2} \mathcal{H}_1(B)\right) - \dim_{\mathbb{F}_2}(V) - \dim_{\mathbb{F}_2}(W) && (5.47)\\ &= \dim_{\mathbb{F}_2}(\mathcal{H}_1(A)) \cdot \dim(\mathcal{H}_0(B)) + \dim(\mathcal{H}_0(A)) \cdot \dim(\mathcal{H}_1(B)) - \dim_{\mathbb{F}_2}(V) - \dim_{\mathbb{F}_2}(W) && (5.48)\\ &= \dim_{\mathbb{F}_2}(\mathcal{H}_1(\mathbb{B}(A))) \cdot \dim(\mathcal{H}_0(\mathbb{B}(B))) + \dim(\mathcal{H}_0(\mathbb{B}(A))) \cdot \dim(\mathcal{H}_1(\mathbb{B}(B))) - \dim_{\mathbb{F}_2}(V) - \dim_{\mathbb{F}_2}(W) && (5.49)\\ &= \dim_{\mathbb{F}_2}\left(\mathcal{H}_1\left(\mathbb{B}(A) \otimes_{\mathbb{F}_2} \mathbb{B}(B)\right)\right) - \dim_{\mathbb{F}_2}(V) - \dim_{\mathbb{F}_2}(W) && (5.50)\\ & && (5.51)\end{aligned}$$

where:

$$V := \langle ga \otimes gb - a \otimes b \, : \, a \in H_1(A), b \in H_0(B), g \in G \rangle, \quad W := \langle ga \otimes gb - a \otimes b \, : \, a \in H_0(A), b \in H_1(B), g \in G \rangle \tag{5.52}$$

We thus see that the problem of determining the number of encoded qubits of the lifted product code reduces to the problem of doing so for the hypergraph product and, moreover, determining the dimension of V, W.

III

Spectral Sequences, Fibre Bundles and Twisted Product Codes

6

Fibre Bundles and Covering Spaces

In this chapter, we will first discuss covering space theory. Afterwards, we will give an overview of the theory of fibre bundles.

6.1. Covering Space Theory

Covering spaces are simply spaces that cover other spaces. Let us formalise this notion.

6.1.1. Ways of Covering a Circle

To gain some intuition for the concept of covering spaces, we start off by introducing an examplar of this theory: the circle S^1. One can easily cover the circle with another circle, by wrapping the second circle around it once, twice, or even more times. To cover it infinitely often, one could use a helix (which is homeomorphic to $\mathbb{R}$), as can be seen in this figure:

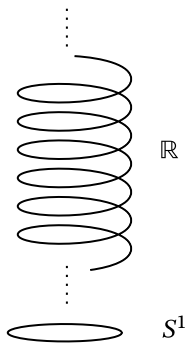


Figure 6.1: A circle S^1, together with a helix covering it. The helix is homeomorphic to $\mathbb{R}$.

Are there any more ways to cover S^1? The answer to this question, surprisingly, is no (at least, not up to isomorphism). To prove this, however, we need a better theoretic understanding of covering spaces. To this end, let us first introduce the formal definition of a covering space. [DK01, ST].

6.1.2. Covering Spaces and their Morphisms

Definition 6.1. Let $E, B \in$ **Top** and let $p : E \to B$ be a surjective map such that every $b \in B$ has an open neighbourhood $U \subseteq B$ such that:

1. $p^{-1}(U) = \bigsqcup V_\alpha$, with $V_\alpha \subseteq E$ open
2. For each α, $p|_{V_\alpha}$ is a homeomorphism $V_\alpha \cong U$.

Then E is a *covering space* of B with *covering map* p.

We see that our motivating examples indeed satisfy this definition: in the case of the helix, we take p to be the projection onto S^1, whereas in the case in which the cover is the circle, one can take the complex valued map $f_n : z \mapsto z^n$ as the covering map.

Another, quite different example of a covering space can be constructed as follows: Let G be a simply connected, abelian Lie group, and let $\mathfrak{g}$ be its Lie algebra in some point, say x. Then $\mathfrak{g}$ can be seen as a covering space of G. Here, the exponential map $\exp : \mathfrak{g} \to G$ serves as the covering map.

Given a covering space, we can easily find other covering spaces by restriction or by taking products.

Lemma 6.1. *Let $p : E \to B$ be a covering space. Given some subset $B_0 \subseteq B$. Then $E_0 := p^{-1}(B_0) \subseteq E$ is a covering space of B_0 with covering map $p|_{E_0}$.*

Lemma 6.2. *Let $p : E \to B$ and $p' : E' \to B'$ be covering spaces. Then $E \times E'$ is a covering space of $B \times B'$ with covering map $p \times p'$.*

Proof. The proof of these lemmata follows readily from elementary topological arguments, see e.g. [ST]. □

As a consequence of the last theorem, we see that the infinitely long cylinder $S^1 \times \mathbb{R}$ and the plane $\mathbb{R}^2$, are covers of the torus $\mathbb{T} \simeq S^1 \times S^1$.

We will mostly be interested in spaces which admit a covering space that is — in a certain sense — the largest possible covering space, much like how the helix is the largest covering space of S^1. This idea is captured in the notion of a universal cover.

Definition 6.2. Let $p : E \to B$ be a covering space. If E is simply connected, it is the *universal cover* of B.

One is guaranteed that a space possesses a universal cover if (and only if) it is a semilocally simply connected, locally path connected and path connected space ([ST, Hat01]). From this point forward, let us refer to such spaces as *"sufficiently well-behaved spaces"*, and let us assume that all the spaces we consider are sufficiently well-behaved.

Having presented these elementary definitions, we can now move forward to show which favourable properties covering spaces have, and turn to the question of how to classify the covering spaces of a given topological space. To do so, however, we first need to introduce the notion of an equivalence of covering spaces.

Definition 6.3. Let $p : E \to B$ and $p' : E' \to B$ be two covering spaces of B. A continuous mapping $Q : E \to E'$ is said to be a *covering space morphism* if the following diagram commutes:

$$\begin{tikzcd} E \arrow[rr, "Q"] \arrow[dr, "p"'] & & E' \arrow[dl, "p'"] \\ & B & \end{tikzcd}$$

if Q is, moreover, a homeomorphism, then Q is said to be an *isomorphism of covering spaces*. Lastly, if Q is an automorphism of some covering space E, then Q is called a *deck transformation*.

Returning to the example of the covering space $p : \mathbb{R} \to S^1$, one can see that mapping the coils of the helix onto the coils below them is an example of a deck transformation of the helix. Compositions of these maps are again deck transformations, and, furthermore, one can identify an inverse deck transformation as the map which maps each coil to the one above it:

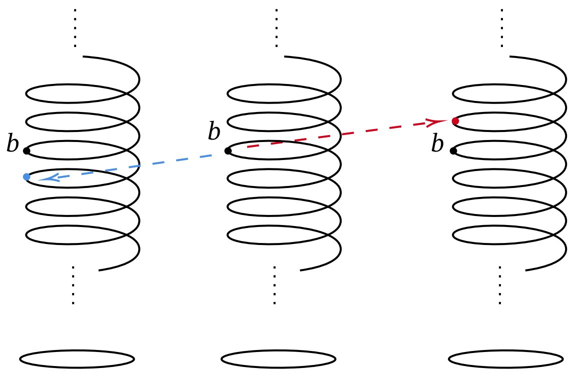


Figure 6.2: Three examples of the same covering of S^1 using the helix with basepoint b, along with the deck transformations between them. To move from the middle covering space to the leftmost one, one has to move down one winding of the helix, as indicated by the blue dashed arrow. To move from the middle covering space to the rightmost one, one has to move up one winding, as indicated by the red dashed arrow.

As such, one can see that the deck transformations actually form a group under composition. This is true in general, and the generated group is called the *automorphism group* of the covering space (see e.g. [Hat01]):

Theorem 6.3. *Let B be a sufficiently well-behaved space, and denote its universal cover by $\tilde{B}$. Then:*

$$\pi_1(B) \simeq \mathit{Aut}(\tilde{B}) \tag{6.1}$$

6.1.3. Lifting Properties of Covering Spaces

By definition, the open sets of B are lifted to open sets in E. This may lead us to wonder what additional structure of B might be lifted to its cover E. To this end, let us first return to the example of covering S^1 by the helix. One can easily see that any path on S^1 starting at some point b_0 can be lifted to a path on the helix, provided that we fix its starting point $e_0 \in p^{-1}(b_0)$ on the helix:

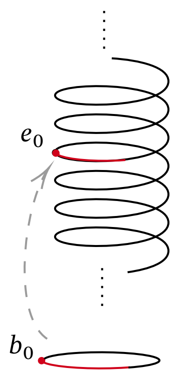


Figure 6.3: A path on the circle S^1 starting in the basepoint b_0 (in red), along with its lift to the covering space $\mathbb{R}$ that is based in the point e_0.

This is not merely a happy accident, but rather, is a property of every covering space.

Theorem 6.4 (Path Lifting Property)**.** *Let $p : E \to B$ be a covering space, and let $\gamma : [0,1] \to B$ be a path in B such that $\gamma(0) = b_0$. Given any $e_0 \in E$ such that $p(e_0) = b_0$, there is a unique path $\tilde{\gamma} : [0,1] \to E$ such that $\tilde{\gamma}(0) = e_0$ and $p \circ \tilde{\gamma} = \gamma$. In summary, we have the following commuting diagram:*

$$\begin{tikzcd}
{[0,1]} \arrow[r, "\exists!\,\tilde{\gamma}"] \arrow[dr, "\gamma"'] & (E, e_0) \arrow[d, "p"] \\
& (B, b_0)
\end{tikzcd}$$

Proof. Let $\mathcal{U} = \{U_i\}$ be an open cover of B, such that for each U_i, $p^{-1}(U_i)$ can be written as the disjoint union of open sets in E and such that p, restricted to each of these open sets, is a homeomorphism. By the Lebesgue number lemma, we can find a partition of $[0,1]$, say $\{[t_i, t_{i+1}, i = 1, \dots, N]\}$ such that each $\gamma([t_i, t_{i+1}])$ is contained a single open set in $\mathcal{U}$, say U_{t_i}. We can now define a proper lift for $\gamma|_{[t_0,t_1]}$, as there is but one open set V_0 in the preimage $p^{-1}(U_{t_0})$ with $e_0 \in V_0$, and, moreover, $p|_{V_0}$ is a homeomorphism $V_0 \cong U_{t_0}$. Hence, we can define $\tilde{\gamma}|_{[t_0,t_1]} = p|_{V_0}^{-1} \circ \gamma|_{[t_0,t_1]}$. This procedure also fixes the value for $\tilde{\gamma}(t_1)$. We can thus perform the same procedure to find the unique set V_1 in the preimage of U_{t_1} for which $\tilde{\gamma}(t_1) \in V_1$. Hence, we can define a lift for $\gamma|_{[t_1,t_2]}$, and, continuing inductively, we find lifts on each of the intervals $[t_i, t_{i+1}]$ such that $\tilde{\gamma}_{[t_i,t_{i+1}]}(t_{i+1}) = \tilde{\gamma}_{[t_{i+1},t_{i+2}]}(t_{i+1})$. By the gluing lemma, we conclude that $\tilde{\gamma} : [0,1] \to E$ is a continuous path.
Lastly, given two lifts $\tilde{\gamma}_1$ and $\tilde{\gamma}_2$, one must have that:

$$p|_{V_i} \circ \tilde{\gamma}_1|_{[t_i,t_{i+1}]} = \gamma|_{[t_i,t_{i+1}]} = p|_{V_i} \circ \tilde{\gamma}_2|_{[t_i,t_{i+1}]} \tag{6.2}$$

By induction on i, one can easily see that $\tilde{\gamma}_1|_{[t_i,t_{i+1}]}$ and $\tilde{\gamma}_2|_{[t_i,t_{i+1}]}$ are in the same path component of $p^{-1}(U_{t_0})$. As $p|_{V_i}$ is a homeomorphism, we now see that $\tilde{\gamma}_1|_{[t_i,t_{i+1}]} = \tilde{\gamma}_2|_{[t_i,t_{i+1}]}$. We thus conclude that lifts of paths are unique. □

Not just paths, but also homotopies between paths can be lifted to covering spaces.

Theorem 6.5 (Homotopy Lifting Property). *Let $p : E \to B$ be a covering space, and let $H : [0,1] \times [0,1] \to B$ be a continuous function such that $H(0,0) = b_0$. Given any $e_0 \in E$ such that $p(e_0) = b_0$, there is a unique homotopy $\tilde{H} : [0,1] \times [0,1] \to E$ such that $\tilde{H}(0,0) = e_0$ and $p \circ \tilde{H} = H$.*

Proof. The proof runs analogous to the proof of the path lifting property, with the only difference being that one must partition $[0,1]^2$ instead of $[0,1]$. □

These lifting results can be generalised to arbitrary maps, provided that some additional conditions are satisfied.

Theorem 6.6 (Lifting Criterion). *Let $p : (E, e_0) \to (B, b_0)$ be a covering map, and suppose X is a path connected and locally path connected topological space. A map $f : (X, x_0) \to (B, b_0)$ has a (unique) lift $\tilde{f} : (X, x_0) \to (E, e_0)$ if and only if $f_*(\pi_1(X, x_0)) \subseteq p_*(\pi_1(E, e_0))$.*

Proof. $\Leftarrow$. Let $x \in X$ arbitrary. Define a path γ from x_0 to x. We define $\tilde{f}(x) := \widetilde{(f \circ \gamma)}(1)$. We have to prove two things: firstly, that this expression is well-defined, and secondly, that f defined in this pointwise manner is unique. Let us start with the first of these: suppose that there is another path γ' from x_0 to x. We want to show that $\widetilde{(f \circ \gamma)}(1) = \widetilde{(f \circ \gamma')}(1)$. To this end, consider $\gamma' * \gamma^{-1}$. One can easily see that this is a loop in X. Furthermore, we see that the path $f \circ (\gamma' * \gamma^{-1})$ in B is also a loop. As its homotopy class is in the image of f_*, the assumption tells us that its homotopy class is also in the p_*-image of the fundamental group of (E, e_0), and hence, that $f \circ \widetilde{(\gamma' * \gamma^{-1})}$ is homotopic to a path that lifts to a loop in (E, e_0). By the path homotopy lifting property this means that $f \circ \widetilde{(\gamma' * \gamma^{-1})}$ itself also lifts to a path in (E, e_0) that is in the same homotopy class as a loop in E, and hence also a loop, i.e.:

$$\widetilde{f \circ \gamma'}(1) = f \circ \widetilde{(\gamma' * \gamma^{-1})}(1) = f \circ \widetilde{(\gamma' * \gamma^{-1})}(0) = \widetilde{f \circ \gamma^{-1}}(0) = \widetilde{f \circ \gamma}(1) \tag{6.3}$$

To prove continuity, we point to the following commuting diagram:

$$\begin{tikzcd}
(X, x_0) \arrow[r, "\bar{f}"] \arrow[dr, "f"'] & (E, e_0) \arrow[d, "p"] \\
 & (B, b_0)
\end{tikzcd}$$

Let $x \in X$, and consider an open neighbourhood U of $f(x)$. By continuity of f, there is a subset $V \subseteq X : f(V) \subseteq W$. Now $p^{-1}(U)$ is a disjoint union of open sets. Let W denote the subset of this disjoint union such that $\tilde{f}(x) \in W$. Now, as $p|_W$ is a homeomorphism $W \cong U$, we see that $\tilde{f}|_V = p_W^{-1} \circ f_V$, and hence, that $\tilde{f}|_V$ is continuous.
$\Rightarrow$. This follows immediately from applying the functor π_1 : **Top*** $\to$ **Grp** to the commutative diagram given above. □

The lifting criterion provides us with a tool for analysing the relationship between different covering spaces of the same space. For example, by taking $f : X \to B$ to be the universal cover of B and letting E be an arbitrary different cover, we find that, as $\pi_1(X)$ is trivial (by definition), f lifts to a cover of E. Hence, we conclude that the universal cover of B is also the universal cover of every other covering space of B.

This surprisingly profound yet simple consequence of the lifting criterion may lead one to speculate whether there is a deeper connection between the fundamental group of B and its covering spaces. This is indeed the case, as we will demonstrate in the next section.

6.1.4. Classification of Covering Spaces and Covering Space Morphisms

In this section, we will make the relationship between the homotopy of a space and its covering spaces explicit, and, in doing so, will fully classify the covering spaces of a given space.

To this end, let us first consider the example of the 2-fold covering of S^1. Let us consider two opposite points on S^1, say x_0 and x_1, that both project onto the same point y.

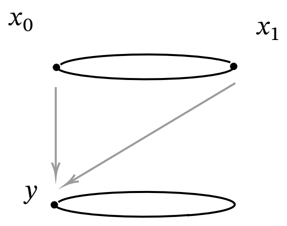


Figure 6.4: The 1-sphere with basepoint y, along with itself double cover. The preimage of y under the covering map consists of the points x_0 and x_1.

Inspired by the lifting criterion, one may ask how $p_*(\pi_1(S^1, x_0))$ and $p_*(\pi_1(S^1, x_1))$ relate to each other. Although the answer can be derived from elementary algebraic topological considerations, it is nevertheless instructive to consider whether one can arrive at this answer in a different way. One approach would be the following: let γ be the 1-loop starting and ending in y. By the path lifting property, we see that γ lifts to a path $\tilde{\gamma}$ from x_0 to x_1:

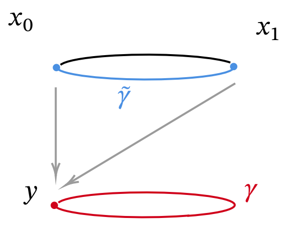


Figure 6.5: The closed loop γ that goes around the circle S^1 once and is based in y, along with its lift to the double cover of S^1. Its lift starts in x_0 and ends in x_1, and is therefore no longer a closed loop.

While conjugation by γ leaves $\pi_1(S^1, y)$ unaffected, we see that conjugation by $\tilde{\gamma}$ shifts paths starting in x_0 to paths starting in x_1 and vice versa. After projection, therefore, one expects the following expression to hold:

$$p_*(\pi_1(S^1, x_0)) = [\gamma]^{-1} p_*(\pi_1(S^1, x_1))[\gamma] \tag{6.4}$$

This is indeed true, even in a more general setting:

Theorem 6.7. *Let $p : E \to B$ and $p' : E' \to B$ be covering maps such that $p(e_0) = p'(e_0') = b_0$. There is an isomorphism of coverings $Q : E \to E'$ if and only if $p_*(\pi_1(E, e_0))$ is conjugate to $p_*(\pi_1(E, e_1))$ as a subgroup of $\pi_1(B, b_0)$.*

Proof. The proof to this theorem can be find in e.g. [Hat01, ST]. □

Let us return to the example of the n-fold covering S^1 by S^1: given a loop $\tilde{\gamma}(t)$ in S^1, we see that the covering map $f_n : z \mapsto z^n$ maps this loop its n-fold copy. Therefore, identifying $\pi_1(S^1)$ with $\mathbb{Z}$, we see that the covering map f_n induces a map $(f_n)_* : \pi_1(S^1) \to \pi_1(S^1)$ satisfying $(f_n)_*(\mathbb{Z}) = n\mathbb{Z}$. Conversely, we see that every non-trivial subgroup of $\mathbb{Z}$ is the image of some covering map $(f_n)_*$, while the trivial subgroup is the image of the universal covering map. This seems to imply that we can — in some sense — identify a covering space $p : E \to S^1$ with a subgroup of $\pi_1(S^1)$. As such, our initial claim that $f_n : S^1 \to S^1$ and $p : \mathbb{R} \to S^1$ are the only covering spaces of S^1 follows. This relationship between covering maps on the one hand and subgroups of the fundamental group on the other holds more generally:

Theorem 6.8. *Let B be a sufficiently well-behaved space. There is a one-to-one mapping between the set of isomorphism classes of cover spaces $p : E \to B$ and the conjugacy classes of subgroups of $\pi_1(B, b_0)$.*

Proof. We refer the reader to [ST] for the proof. □

Lastly, it is instructive to consider the relationship between maps between covering spaces on the on hand, and covering maps on the other. This relationship can be explicated as follows [ST]:

Theorem 6.9. *Let $E_0, E_1, B \in$ **Top** be path connected and locally path connected spaces, and let p_0, p_1 and q be maps such that the following diagram commutes:*

$$\begin{tikzcd}
E_1 \arrow[dr, "q"] \arrow[dd, "p_1"'] & \\
& E_0 \arrow[dl, "p_0"] \\
B &
\end{tikzcd}$$

If p_0 and p_1 are both covering maps, then so is q. Similarly, if p_0 and q are covering maps, then so is p_1. Moreover, given that B also admits a universal cover, If q and p_1 are both covering maps, then so is p_0.

We thus conclude that for semi-locally simply connected spaces, the notion of a covering map and a morphism of covering spaces coincide.

6.2. Fibre Bundle Theory

We present elementary notions from fibre bundle theory. Before doing so, we let us first try to explain why this theory was developed.

6.2.1. From Global To Local Descriptions

Not many spaces are easy to grasp in their entirety. The theory of manifolds, which are the spaces of differential geometers, deals with this problem by stepping away from the urge to describe spaces globally, and by describing them *locally* instead. In topology, this can be done using the notion of a fibre bundle. Let us try to illustrate this using the example of a Möbius strip.

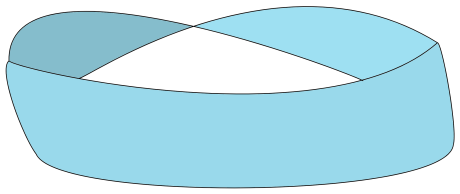


Figure 6.6: An artist's impression of the Möbius strip.

Locally, the Möbius band is not much different from any ordinary band. Yet, globally, it possesses an extra twist. That is, the strip, which can be seen as the product space $S^1 \times [-1, 1]$, can be understood locally as a product space too, i.e. as $(-1,1) \times [-1,1]$. The latter statement is also true for the Möbius strip, yet, globally, it is not a simple product space.

The idea behind fibre bundle theory is to exploit the fact that the Möbius strip is locally quite simple. To do so, we can break up the Möbius band into two overlapping strips:

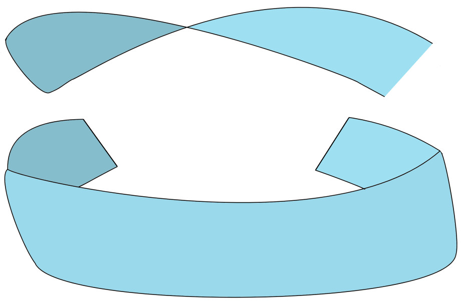


Figure 6.7: Two overlapping parts of the Möbius strip, each of which is itself homeomorphic to the regular, untwisted strip.

Formally, these local descriptions are given by *charts*, which are local homeomorphisms between the Möbius band and the strip $(-1,1) \times [-1,1]$. In order to distinguish the regular, untwisted band from the Möbius strip, we will need to specify how these different local descriptions of our space fit together. This information is encoded in the transition functions, which tells us how to move from one local description to the other via their overlap.

6.2.2. Fibre Bundles and Principal Bundles

After having presented the intuition behind the theory of fibre bundles, it is time to give an exposition of the formal theory. This section is primarily based on [DK01].

Definition 6.4 (Fibre Bundle)**.** Let $F, B, E \in$ **Top** and let $p : E \to B$ be a surjective map. (E, B, p, F) is called a *locally trivial bundle* if there is a collection of homeomorphisms called *charts* $\{\varphi : U \times F \to p^{-1}(U)\}$ for open sets U such that:

1. The following diagram commutes for each chart φ_U:

$$\begin{tikzcd} U \times F \arrow[rr, "\varphi_U"] \arrow[dr, "p_U"'] & & p^{-1}(U) \arrow[dl, "p"] \\ & U & \end{tikzcd}$$

2. For every $b \in B$: $\exists U \subseteq B$ open along with a chart φ_U such that $b \in U$.

A locally trivial bundle is called a *fibre bundle* if, moreover, there is a topological group G acting effectively[1] on F such that the following two conditions hold:

3. for every two charts $\varphi_U, \varphi_{U'}$ with $U \cap U' \neq \emptyset$, there exists a continuous function, called a *transition function*, $\theta_{\varphi_U \varphi_{U'}} : U \cap U' \to G$, such that for each $b \in U \cap U', f \in F$:

$$\varphi'(b, f) = \varphi(b, \theta_{\varphi_U \varphi_{U'}}(b) \cdot f) \tag{6.5}$$

We note that there is always a unique collection of charts that is maximal with respect to these conditions. Restricting our attention to this maximal collection ensures that for every φ_U, given an open subset $V \subseteq U$, there is a chart φ_V such that $\varphi_V = (\varphi_U)|_V$.

There are two other types of fibre bundles that one can consider.

Definition 6.5 (Principal Bundle)**.** Let $E, B \in$ **Top**, and let $G \in$ **Grp**. A fibre bundle $p : E \to B$ with fibre G is called a *principal G-bundle* if it has structure group G which acts on the fibre by left-translation.

[1]We say that G acts effectively on X if the map $G \to \text{Homeo(X)}$ is injective.

One can see that covering spaces are specific instances of fibre bundles. Morphisms of covering spaces are thus specific instances of morphisms between fibre bundles, which we will introduce later on. Similarly, the universal cover of X can be interpreted as a principal $\pi_1(X)$-bundle. More generally, every regular covering space is a principal $\pi_1(X)/p_*[\pi_1(E)]$-bundle (see e.g. [DK01]).

There is another class of fibre bundles which will prove to be important later in our discussion.

Definition 6.6 (Local Coefficient Systems). Let $A \in$ **Ab** be topologised with the discrete topology, and let G be a subgroup of Aut(A). A fibre bundle (E, B, p, A) with structure group G is called a *system of local coefficients* on B.

Note that systems of local coefficients are, in fact, also covering spaces.

Although the definition of a fibre bundle requires an explicit description of the corresponding charts, we can also describe a fibre bundle in terms of its transition functions alone ([DK01]):

Theorem 6.10 (Fibre Bundle Construction). *Let $B, F \in$ **Top** and let G be a topological group that acts continuously on F. Given an open cover $\{U_i\}$ of B and a set of continuous functions $\theta_{ij} : U_i \cap U_j \to G$ satisfying the so-called Čech cocycle condition on every domain $U_i \cap U_j \cap U_k$:*

$$\theta_{ij} = \theta_{ik}\theta_{kj} \tag{6.6}$$

Then there exists a fibre bundle (E, B, p, F) that is locally trivialisable over $\{U_i\}$ with transition functions θ_{ij}.

One of the powerful implications of this theorem is that, given a fibre bundle $p : E \to B$ with structure group G and fibre F, and suppose that there is another space $F' \in$ **Top** on which G acts effectively too. Then, we can construct a new fibre bundle $p' : E' \to B$ with fibre F' using the transition bundles of the first bundle. For every fibre bundle, therefore, one can construct a principal G-bundle with fibre $F' = G$. Using the so-called *Borel construction*, one can associate a fibre bundle to every principal G-bundle (see [DK01]):

Theorem 6.11. *Let $p : P \to B$ be a principal G-bundle, and let $F \in$ **Top** such that there is a left-action $G \times F \to F$. We define:*

$$P \times_G F := (P \times F)/\sim \tag{6.7}$$

where $(x, f) \sim (xg, g^{-1}f)$. One can now define a projection $P \times F \to B : (x, f) \mapsto p(x)$, which descends to a map $q : P \times_G F \to B$. Now $q : P \times_G F \to B$ is a fibre bundle with exactly the same transition functions as $p : P \to B$.

One can use this construction to characterise local coefficient systems.

Theorem 6.12. *Let (E, B, p, A) be a local coefficient system over a sufficiently well-behaved space B with universal cover $\tilde{B}$. Letting $\rho : \pi_1 B \to$ Aut(A), one finds that $p : E \to B$ is the fibre bundle associated to the principal $\pi_B/$ker(ρ)-bundle $p' : \tilde{B} \to B$ constructed using the Borel construction with action induced by the representation ρ.*

6.2.3. Morphisms of Fibre Bundles

Having discussed the basic notions related to fibre bundles, it is time to introduce the maps between them.

Definition 6.7 (Fibre Bundle Morphism). let (E, B, p, F) and (E', B', p', F') be two fibre bundles, and let $\beta : B \to B'$ and $\epsilon : E \to E'$ be two continuous maps. The pair (β, ϵ) is a *fibre bundle morphism* if the following diagram commutes:

$$\begin{tikzcd}
E \arrow[r, "\epsilon"] \arrow[d, "p"'] & E' \arrow[d, "p'"] \\
B \arrow[r, "\beta"] & B'
\end{tikzcd}$$

and, moreover, for every point $b \in B$, every open neighbourhood of b, $U \subseteq B$, and every open set $U' \subseteq B'$ such that $\beta(b) \in U'$, given two charts φ_U and $\varphi_{U'}$, there is a continuous map $\psi_{\varphi_U \varphi_{U'}} : U \cap f^{-1}(U') \to G$ such that the map:

$$\begin{tikzcd}
\{b\} \times F \arrow[d, "\varphi_U"'] & \{f(b)\} \times F \\
p^{-1}(b) \arrow[r, "\epsilon"] & (p')^{-1}(f(b)) \arrow[u, "\varphi_{U'}"']
\end{tikzcd}$$

is equivalent to the action induced by $\psi_{\varphi_U \varphi_{U'}}(b)$.

Lastly, we can introduce the notion of a fibre bundle equivalence, i.e. a fibre bundle isomorphism. We define a fibre bundle morphism (ϵ, β) to be an isomorphism of fibre bundles if there are maps (ϵ', β') going in the reverse direction such that the compositions of these maps are the identity.

7 Homology with Local Coefficients and Spectral Sequences

We previously saw a generalisation of the theory of homology that replaced the vector space structure of the chain complexes with the more general structure of a ring module. We now consider a different generalisation of the theory of singular homology, which aims to loosen the assumptions on the coefficient ring rather than the structure of the chain complexes. We first introduce this so-called *homology with local coefficients*. Afterwards, we introduce a technique that can be used to calculate the homology of fibre bundles, so-called *spectral sequences*. We finally use the theory of homology with local coefficients to construct a specific class of spectral sequences, the so-called *Leray-Serre spectral sequence*.

7.1. Homology with Local Coefficients

7.1.1. Local Coefficients

The key insight that motivates the introduction of homology with local coefficients is the fact that the homology theories we have considered thus far possess a certain *globality*. When considering fibre bundles, however, we move from a global description to a local description of space. Therefore, we would like to be able to do homology within such a local framework as well, and thus be able to make inferences about the homological structure of a fibre bundle by using its base space, as well of information about its fibre. Yet, one ought to be careful, as paths over the base space with the same endpoints no longer necessarily lift to paths with the same endpoints in the fibre bundle. Therefore, when determining the boundary of some path over the base space, one must take into account not only the endpoints of the path, but also the lifting properties of this path. To this end, we define a *local* system on our (base) space X: [Max13]:

Definition 7.1. Let $X \in$ **Top**. A *local system* $\underline{A} = \{A_x, \tau_\gamma\}$ assigns to each point $x \in X$ an abelian group A_x such that $A_x \simeq A_y$ for each other point $y \in X$. Furthermore, $\underline{A}$ assigns to each path $\gamma : [0, 1] \to X$ a isomorphism $A_{\gamma(1)} \to A_{\gamma_0}$ such that:

1. For the constant path $\gamma(t) = x$, $\tau_\gamma =$ id, and
2. For each $\sigma : \Delta^2 \to X$, one has the following identity:

$$\tau_{\sigma\circ\delta_0} \circ \tau_{\sigma\circ\delta_2} = \tau_{\sigma\circ\delta_1} \tag{7.1}$$

The two conditions in the definition of a local system ensure the regularity of our construction: the first condition makes sense, as we expect constant paths to lift to constant paths as well, while the second condition ensures that our spaces are locally well-behaved, in the sense that, two paths within a nicely behaved subspace with the same starting point and end point are required to behave similarly. One can indeed prove that this condition implies that for homotopic paths $\gamma \simeq \gamma'$, one has that $\tau_\gamma = \tau_{\gamma'}$ (see e.g. [Max13]).

This definition is, moreover, equivalent to the definition of a local coefficient system given in the last chapter: fixing some arbitrary point $x \in X$, we can take the group A_x as the fibre. The functions τ_γ then provide us with transition functions, where the second of the conditions of imposed on these functions is equivalent to the Čech cocycle condition. As such, by the fibre bundle construction theorem, our claim follows.

Each fibre bundle $p : E \to B$ induces a system of local coefficients. In fact, these will be the systems of local coefficients that we will be most interested in in what is to follow. Fixing some $q \in \mathbb{Z}$, we can define $A_x := \mathcal{H}_q(p^{-1}(x))$. One can show that in this case, there are induced isomorphisms τ_γ for every 1-simplex $\gamma : [0,1] \to B$. We refer the interested reader to [Max13] for the technical details of this construction.

There are two ways of constructing the homology groups with local coefficients of some given topological space: an algebraic approach and a topological approach [DK01, Geo78]. We present both of these. Before that, however, note that we will always assume that our space X is a semi-locally connected, path connected and locally path connected space. Hence, X admits a universal cover, which we will denote by $\tilde{X}$.

7.1.2. The Algebraic Construction

Suppose we are given an abelian group A. We have already seen how to define the singular homology groups of X with coefficients in A. Another way of obtaining these singular homology groups, is from the universal coefficient theorem, which, assuming that A is flat as a $\mathbb{Z}$-module, yields:

$$H_i(X;A) \simeq H_i(X;\mathbb{Z}) \otimes_{\mathbb{Z}} A \tag{7.2}$$

Now, let us assume moreover that we are given a representation $\rho : \pi_1(X) \to \text{Aut}(A)$. In line with the universal coefficient theorem, we would like to define the homology of X with coefficients in A, but then with an additional *twist*, as described ρ. As ρ acts on $\pi_1(X)$, which is isomorphic to the group of deck transformations of $\tilde{X}$, it is natural to consider the following definition:

Definition 7.2. Let X be a semi-locally, path connected, locally path connected space, and let A be an abelian group. Let $\rho : \pi_1(X) \to \text{Aut}(A)$ be a representation. We define the *homology of X with coefficients in A twisted by ρ, $H_n(X;A_\rho)$*, as the homology of the following chain complex:

$$C_n(X;A_\rho) := C_n(\tilde{X};\mathbb{Z}) \otimes_{\mathbb{Z}[\pi_1(X)]} A \tag{7.3}$$

We expect that taking a trivial twist in this construction brings us back to singular homology — let us check this. Firstly, note that $C_n(X;\mathbb{Z})$ is a free $\mathbb{Z}$-module. Let us denote its basis by $\{\sigma_i\}$. We can lift this basis to a set $\{\tilde{\sigma_i}\} \subseteq C_n(\tilde{X};\mathbb{Z})$. Acting on this set by all possible deck transformations, one finds a $\mathbb{Z}$-basis of $C_n(\tilde{X};\mathbb{Z})$. Therefore, this set is a $\mathbb{Z}[\pi_1(X)]$-basis of $C_n(\tilde{X};\mathbb{Z})$. Furthermore, as ρ acts trivially on A, we see that the following map is an isomorphism:

$$C_n(\tilde{X};\mathbb{Z}) \otimes_{\mathbb{Z}[\pi_1(X)]} A \to C_n(X;\mathbb{Z}) \otimes_{\mathbb{Z}} A : \tilde{\sigma_i} \otimes_{\mathbb{Z}[\pi_1(X)]} a \mapsto \sigma_i \otimes_{\mathbb{Z}} a \tag{7.4}$$

Moreover, this map is clearly also a chain map. We have found a chain isomorphism between these two chain complexes, and thus, we see that the homology with a trivial twist coincides with singular homology.

Let us consider what happens when A is a free $\mathbb{Z}[\pi_1(X)]$-module. To this end, it suffices to consider the case when $A = \mathbb{Z}[\pi_1(X)]$. We then see that:

$$C_n(\tilde{X};\mathbb{Z}) \otimes_{\mathbb{Z}[\pi_1(X)]} A = C_n(\tilde{X};\mathbb{Z}) \otimes_{\mathbb{Z}[\pi_1(X)]} \mathbb{Z}[\pi_1(X)] \simeq C_n(\tilde{X};\mathbb{Z}) \tag{7.5}$$

Such that:

$$H_n(X;A_\rho) = H_n(\tilde{X};\mathbb{Z}) \tag{7.6}$$

Taking the local coefficient system induced by the principal $\pi_1(X)$-bundle $p : \tilde{X} \to X$ with fibres isomorphic to $H_0(p^{-1}(x))$ for some $x \in X$, one can choose ρ to be the representation that identifies $\pi_1(X) \simeq \text{Aut}(\tilde{X})$. As the fibre of a local coefficient system is discrete, we have that $H_0(p^{-1}(X)) \simeq \mathbb{Z}[p^{-1}(X)] \simeq \mathbb{Z}[\pi_1(X)]$. Hence, by our last example, we see that, in this case, the homology with twisted coefficients coincides with the homology of the universal cover.

7.1.3. The Topological Construction

We now take a topological approach to constructing the twisted homology. To this end, let us consider a system of coordinates $p : E \to X$, whose fibre is the discrete abelian group A. Let us define sets $C_n(X; E)$, whose elements are taken to finite sums of the following form:

$$\sum_i a_i \sigma_i \tag{7.7}$$

where $\sigma_i : \Delta^n \to X$ is a singular n-complex and a_i is an element from the fibre $p^{-1}(\sigma_i((1, 0, 0, \dots, 0)))$. In the trivial case, the various fibres $p^{-1}(x)$ would be canonically isomorphic and this construction would therefore be equivalent to the construction of the free abelian group $\mathbb{A}[S_n(X)]$.

After noticing that this complex can be realised as the following direct sum, one can easily see that it has an abelian group structure:

$$C_n(X; E) \simeq \bigoplus_{x \in X} C_n(X; p^{-1}(x)) \tag{7.8}$$

Therefore, we can use these chain complexes to construct homology groups. One ingredient is missing, however: the differentials. Let us try to proceed as usual: we consider face maps $\delta_k : \Delta^{n-1} \to \Delta^n$, and consider the composition $\sigma_i \circ \delta_k$. For $k \geq 1$, we see that $\sigma_i \circ \delta_k(1, 0, \dots, 0) = \sigma_i(1, 0, \dots, 0)$. However, for $k = 0$, we see that $\sigma_i \circ \delta_0(1, 0, \dots, 0) = \sigma_i(0, 1, 0, \dots, 0)$, so precomposition with δ_0 does not lift the basepoint in Δ^{n-1} to the basepoint in Δ^n, and hence prohibits us from lifting elements a_i from $p^{-1}(\sigma_i(1, 0, \dots, 0))$ to $p^{-1}(\sigma_i \circ \delta_0(1, 0, \dots, 0))$. We can, however, use the local coordinate system to identify elements of different fibres. To this end, given some σ_i, let us consider the path $\gamma_{\sigma_i} : [0, 1] \to X : t \mapsto \sigma_i(t, 1 - t, 0, \dots, 0)$. As local coefficient systems are also covering spaces, we can use the path lifting property to find an isomorphism $p^{-1}(\sigma_i(1, 0, 0, \dots, 0)) \simeq p^{-1}(\sigma_i(0, 1, 0, \dots, 0))$. This isomorphism allows us to define the differential ∂ as the linear extension of the following map:

$$a_i \sigma_i \mapsto (\gamma_{\sigma_i})_*(a_i) \sigma \circ \delta_0 + \sum_{k=1}^{n} a_i \sigma \circ \delta_k \tag{7.9}$$

The homology of this newly defined chain complex can be shown to be equal to the homology of X with twist ρ, provided that the system of coordinates $p : E \to X$ is of the form $p : \tilde{X} \times_{\pi_1(X)} A$, where the action of $\pi_1(X)$ on A is given by the twist ρ (see e.g. [Geo78] for a proof).

7.1.4. Cellular Homology with Local Coefficients

Directly calculating the singular homology of spaces is already quite hard. One would therefore hope to have an equivalent of cellular homology when using local coefficients. This, fortunately, is indeed the case. We only sketch the construction here, and refer the interested reader to [Geo78, DK01].

Let X be a connected and sufficiently well-behaved CW-complex. Let us denote its universal cover by $p : \tilde{X} \to X$. Now consider an open n-cell in X, say E_j^n along with the homeomorphism $\chi : \overset{\circ}{D} \to E_j^n$. Let us now consider one of the path components of $\tilde{E} \subseteq p^{-1}(E_j^n)$. As both χ and $p|_{\tilde{E}}$ are homeomorphisms, we see that $\tilde{E}$ can actually be interpreted as an open n-cell of $\tilde{X}$: by defining $\tilde{X}_n$ as all the path components of all the open n-cells in X_n, we see that the $\tilde{X}_n$ form a CW complex, and, moreover, that $\tilde{X} = \bigcup_n \tilde{X}_n$. Furthermore, as the fundamental group of X is isomorphic to the group of deck transformations of $\tilde{X}$, we can actually interpret these results as saying that to every $\mathbb{Z}$-cellular complex $\mathcal{C}_n$ belonging to X, there is a $\mathbb{Z}[\pi_1(X)]$-cellular complex belonging to $\tilde{X}$. The differential of this new complex is generated by the old differential through the path lifting property.

7.2. Spectral Sequences

Spectral sequences are powerful tools that can be used to calculate the homology of spaces. Their power comes at the expense of great generality, which makes it hard to intuitively understand their workings. At a very basic level, however, they can be understood as sequences of lattices of abelian groups.

$$\begin{tikzcd}
 & \vdots & \vdots & \vdots & \\
\cdots & E^r_{0,1} & E^r_{1,0} & E^r_{1,1} & \cdots \\
\cdots & E^r_{0,1} & E^r_{0,0} & E^r_{0,1} & \cdots \\
\cdots & E^r_{-1,-1} & E^r_{-1,0} & E^r_{-1,1} & \cdots \\
 & \vdots & \vdots & \vdots &
\end{tikzcd}$$

Each such lattice is usually referred to as a *page*. Between the groups on each page, there are maps which very closely resemble boundary maps we have considered earlier.

$$\begin{tikzcd}
 & \vdots & \vdots & \vdots & \\
\cdots & E^2_{1,-1} \arrow[ul] & E^2_{1,0} \arrow[ul] & E^2_{1,1} \arrow[ul] & \cdots \\
\cdots & E^2_{0,-1} \arrow[ul] & E^2_{0,0} \arrow[ul] & E^2_{0,1} \arrow[ul] & \cdots \arrow[ul] \\
\cdots & E^2_{-1,-1} \arrow[ul] & E^2_{-1,0} \arrow[ul] & E^2_{-1,1} \arrow[ul] & \cdots \arrow[ul] \\
 & \vdots & \vdots & \vdots \arrow[ul] & {} \arrow[ul]
\end{tikzcd}$$

Given some page, the groups on the lattice of the next page are the homology groups with respect to the boundary-like maps between the groups on the previous page. The point, now, is that by choosing these groups and maps appropriately, the lattice will start to stabilise as we walk through the different pages. Hence, we can introduce a notion of convergence for such spectral sequences. By choosing the first page and the boundary maps wisely, the limit page of the spectral sequence can be shown to converge to the homology groups that one is interested in. We will formalise these ideas, after which we will introduce an important spectral sequence: the Leray-Serre spectral sequence. This discussion will primarily be based on [Max13, DK01, Hat01, Wei94].

7.2.1. Spectral Sequences of Filtered Complexes

We first formalise the idea of a lattice of groups.

Definition 7.3. A family of abelian groups E_{**} with indices in $\mathbb{Z}$ is called a *bigraded abelian group*. A family of homomorphisms f between two bigraded abelian groups A and B, such that $f_{p,q} : A_{p,q} \to B_{p+a,q+b}$ is a *bigraded map of degree* (a,b). One can define the kernel and image of such bigraded maps as the family of kernels and images of the respective maps $f_{p,q}$. Finally, we call a pair (E_{**}, ∂) consisting of a bigraded abelian group E_{**} together with a bigraded map $\partial : E_{**} \to E_{**}$ satisfying the condition $\partial^2 = 0$ a *differential bigraded abelian group*.

The condition $\partial^2 = 0$ invites us to consider the homology of such a differential bigraded abelian group.

Definition 7.4. Let (E_{**}, ∂) denote a differential bigraded abelian group, and suppose that ∂ is a map of degree (a,b). The *homology* of (E_{**}, ∂) is the bigraded abelian group defined as:

$$H_{p,q}(E_{**}, \partial) = \frac{\ker(\partial_{p,q})}{\operatorname{Im}(d_{p-a,q-b})} \tag{7.10}$$

We can now define spectral sequences.

Definition 7.5. Given a collection of differential bigraded abelian groups $\{E^r_{**}, \partial^r\}_{r\in\mathbb{N}}$ where each differential ∂^r belonging to *page* E^r_{**} is of bidegree $(-r, r-1)$. This collection is called a *spectral sequence* if the following holds:

$$E^{r+1}_{p,q} \simeq H_{p,q}(E^r, \partial^r) \tag{7.11}$$

At the beginning of this section, we mentioned walking through the different pages of the spectral sequence, until the spectral sequence had stabilised. The limit of this process is captured in the notion of the *limit page* of a sequence. We can explicitly construct this limit page by induction. First, note that we can define: $E^2_{p,q} = Z^1_{p,q}/B^1_{p,q}$, where $Z^1_{p,q} = \ker\left(\partial^1_{p,q}\right)$ and $B^1_{p,q} = \operatorname{Im}\left(\partial^1_{p+1,q}\right)$. Now, given that $\ker\left(d^2\right)_{p,q} \subseteq E^2_{p,q}$, we can write $\ker\left(d^2\right)_{p,q} := Z^2_{p,q}/B^1_{p,q}$. Similarly, we can write $\operatorname{Im}\left(\partial^2_{p+2,q-1}\right) := B^2_{p,q}/B^1_{p,q}$. Therefore, we have that:

$$E^3_{p,q} = (Z^2_{p,q}/B^1_{p,q})/(B^2_{p,q}/B^1_{p,q}) \simeq Z^2_{p,q}/B^2_{p,q} \tag{7.12}$$

Proceeding inductively, we find a sequence of sets $Z^i_{p,q}, B^i_{p,q}$ satisfying:

$$B^1_{p,q} \subseteq B^2_{p,q} \subseteq \ldots B^n_{p,q} \subseteq Z^n_{p,q} \subseteq \ldots \subseteq Z^2_{p,q} \subseteq Z^1_{p,q} \subseteq E^1_{p,q} \tag{7.13}$$

for each $n \in \mathbb{N}$, satisfying the property $E^{n+1}_{p,q} \simeq Z^n_{p,q}/B^n_{p,q}$. Hence, we can define $Z^\infty_{p,q} := \cap_{i=1}^\infty Z^i_{p,q}$ and $B^\infty_{p,q} := \cap_{i=1}^\infty B^i_{p,q}$. The limit page of the spectral sequence is now defined as the bigraded abelian group $E^\infty_{p,q}$ for which the following holds:

$$E^\infty_{p,q} := Z^\infty_{p,q}/B^\infty_{p,q} \tag{7.14}$$

For a technical discussion of the well-definedness of this limit page, we refer the reader to [Wei94].

We will primarily be interested in spectral sequences that are induced by chain complexes. To this end, we first introduce the notion of a filtration, which also allows us to understand the notion of convergence of spectral sequences.

Definition 7.6. A *filtration* on an abelian group A is a family $\{F_pA\}_{p\in\mathbb{Z}}$ of subgroups of A such that:

$$\ldots \subseteq F_{p-1}A \subseteq F_pA \subseteq F_{p+1}A \subseteq A \tag{7.15}$$

Each graded abelian group A_* for which one can find a filtration, there is a naturally induced graded abelian group, namely:

$$E^0_{p,q}(A_*, F_*) := F_pA_{p+q}/F_{p-1}A_{p+q} \tag{7.16}$$

With this in mind, we can now formalise the idea of convergence of a spectral sequence.

Definition 7.7. A spectral sequence $\{E^r_{**}, \partial^r\}$ is said to converge to the graded abelian group A_* if there is a filtration F_* such that there are isomorphisms $E^\infty_{p,q} \simeq E^0_{p,q}(A_*, F_*)$. We denote this by $E^1_{p,q} \to A_{p+q}$.

We can now finally introduce the notion of a filtered complex.

Definition 7.8. A *filtered chain complex* $(\mathcal{C}_*, \partial)$ consists of a chain complex together with a filtration $\{F_p\mathcal{C}\}_{p\in\mathbb{Z}}$ such that the boundary map preserves the filtration, i.e. such that:

$$\partial(F_p\mathcal{C}_n) \subseteq F_p\mathcal{C}_{n-1} \tag{7.17}$$

We note that by definition, for each p, a $(F_p\mathcal{C}_*, \partial)$ again forms a chain complex. Moreover, the inclusion $F_p\mathcal{C}_* \hookrightarrow \mathcal{C}_*$ induces a filtration on the homology of $\mathcal{C}_*$, which we denote by $F_p\mathcal{H}_*(\mathcal{C})$.

Assuming some technical conditions, a filtered chain complex induces a spectral sequence that converges to the homology of the chain complex.

Theorem 7.1. *Let $(\mathcal{C}_*, \partial)$ be a filtered chain complex with filtration F_*, define*

$$E^1_{p,q} = \mathcal{H}_{p+q}\left(\frac{F_p\mathcal{C}_*}{F_{p-1}\mathcal{C}_*}\right), \quad \partial^1 : [x + F_{p-1}\mathcal{C}_*] \mapsto [\partial(x) + F_p\mathcal{C}_*] \tag{7.18}$$

and assume, moreover, that the filtration is bounded in the following sense:

$$\forall n \in \mathbb{N} : \exists P \leq Q : \{0\} = F_P\mathcal{C}_n \subseteq \ldots \subseteq F_Q\mathcal{C}_n = \mathcal{C}_n \tag{7.19}$$

There now is a spectral sequence $\{E^r_{**}, \partial^r\}$ with first page E^1_{**} such that $E^1_{p,q} \Rightarrow H_{p+q}(\mathcal{C})$

Proof. We refer the reader to [Max13] for the proof. □

In particular, this result can be applied to double complexes. Recall that to each double complex $\mathcal{C}_{**}$, one can associate a total complex $\text{Tot}^{\oplus}(\mathcal{C})_*$. This total complex can now be filtered horizontally and vertically. We define its *column-wise filtration* as:

$$(F_p^{\uparrow}\text{Tot}^{\oplus}(\mathcal{C}))_n := \bigoplus_{i\leq p} C_{i,n-i} \tag{7.20}$$

And, similarly, its *row-wise filtration* as:

$$(F_p^{\leftarrow}\text{Tot}^{\oplus}(\mathcal{C}))_n := \bigoplus_{j\leq p} C_{n-j,j} \tag{7.21}$$

As these are clearly bounded (in the sense of Theorem 7.1), we see that there must exist a spectral sequence converging to the homology of the total complex. We can use result to derive the Leray-Serre spectral sequence.

7.2.2. The Leray-Serre Spectral Sequence

We now turn to the Leray-Serre spectral sequence, which can be used to find the homology of a fibre bundle. To this end, we present the construction of Dress[Max13] .
Given a fibre bundle $p : E \to B$ with fibre F and consider the following diagram:

$$\begin{tikzcd}
\Delta^p \times \Delta^q \arrow[r, "\sigma^E_{p,q}"] \arrow[d, "(t,s)\mapsto t"'] & E \arrow[d, "p"] \\
\Delta^p \arrow[r, "\sigma^B_p"'] & B
\end{tikzcd}$$

We define $S_{p,q}$ to be the set of all pairs $(\sigma^E_{p,q}, \sigma^B_p)$ for which this diagram commutes. We can then define the double complex $C_{p,q}$ as the free abelian group with basis $S_{p,q}$, and with horizontal and vertical boundary maps $\partial^{\uparrow}$, $\partial^{\leftarrow}$ for which, for each $p, q > 0$:

$$\partial^{\uparrow}_{p,q} : (\sigma^E_{p,q}, \sigma^B_p) \mapsto \sum_{i=0}^{p}(-1)^i(\sigma^E_{p,q} \circ (\delta_i \times \text{id}_{\Delta^q}), \sigma^B_p \circ \delta_i) \tag{7.22}$$

$$\partial^{\leftarrow}_{p,q} : (\sigma^E_{p,q}, \sigma^B_p) \mapsto \sum_{j=0}^{q}(-1)^{j+p}(\sigma^E_{p,q} \circ (\text{id}_{\Delta^p} \times \delta_j), \sigma^B_p) \tag{7.23}$$

Using the theorem for filtered chain complexes from the previous section and using this construction, one can prove the following theorem [Max13]:

Theorem 7.2. *Let $p : E \to B$ be a fibre bundle.Then there is a first quadrant spectral sequence $\{E^r, \partial^r\}_{r\geq 2}$, the so-called Leray-Serre spectral sequence, that converges to the homology of the total complex E, i.e. $E^2_{p+q} \Rightarrow \mathcal{H}_{p+q}(E)$. Moreover, the second page of the spectral sequence is given by the homology of B with coefficients in the local system $\underline{\mathcal{H}_*}(F)$, i.e.:*

$$E^2_{p,q} = \mathcal{H}_p(B; \underline{\mathcal{H}_q}(F)) \tag{7.24}$$

8

The Homology of the Fibre Bundles: Twists, Mayer-Vietoris and Spectral Sequences

An interesting example of a fibre bundle is the Möbius strip M. As alluded to before, the Möbius strip can be understood locally as a strip $(-1,1) \times [-1,1]$, much like the band $S^1 \times [-1,1]$, but differs from such a band globally due to its twist. We can therefore interpret the Möbius strip as a fibre bundle over S^1 with the interval $[-1,1]$ as its fibre. Following the Borel construction, we can also interpret it as the quotient space $M \simeq \mathbb{R} \times [-1,1]/\mathbb{Z}$, where the $\mathbb{Z}$-action of is given by $n \cdot (x,y) = (x+n, (-1)^n y)$.

Often, the homology of M is calculated by noting that there is a homotopy equivalence $M \simeq S^1$, and that therefore

$$\mathcal{H}_n(M,\mathbb{Z}) = \begin{cases} \mathbb{Z} & n = 0,1 \\ 0 & n > 1 \end{cases} \tag{8.1}$$

In this chapter, however, we will offer an exposition of three different techniques that can be used when tackling the general problem of calculating the homology of a given fibre bundle. To this end, let us consider the following more generic class of examples of fibre bundles:

$$\begin{tikzcd} F \arrow[r] & E \arrow[d, "p"] \\ & S^1 \end{tikzcd}$$

Given two points $u_1, u_2 \in S^1$, we can define the open sets $U_i = S^1 \setminus \{u_i\}$:

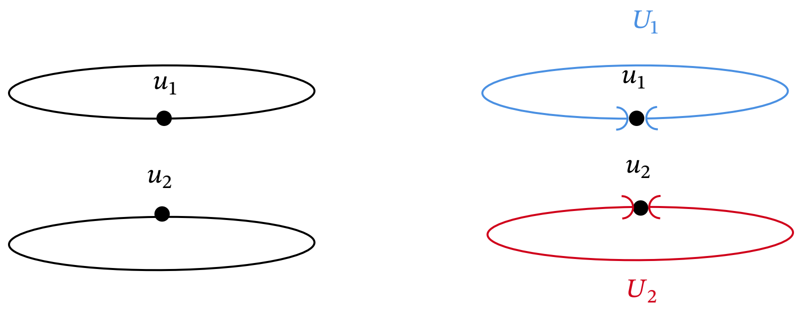


Figure 8.1: On the left: the circle S^1, along with two points u_1, u_2. One can use these points to define open sets $U_i = S^1 \setminus \{u_i\}$, which are shown on the right.

In order to define a transition function $\theta_i : U_1 \bigcap U_2 \to \mathrm{Aut}(F)$, we introduce the following definitions:

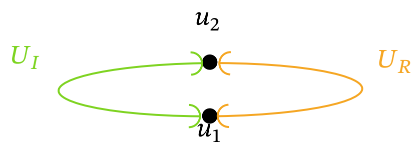


Figure 8.2: The points u_1 and u_2 on the circle S^1, together with the intersection of the two sets $U_i = S^1 \setminus \{u_i\}$, The path components of this set are dubbed U_I and U_R, respectively.

In the case of the Möbius strip, one would define the transition function θ_1 as:

$$\theta_1|_{U_I} = I, \quad \theta_1|_{U_R} = R \tag{8.2}$$

where $R : F \to F : f \mapsto -f$, and $\theta_2 = \theta_1^{-1}$. We generalise this construction to an arbitrary fibre F, and an arbitrary automorphism $R \in \text{Aut}(F)$.

Firstly, we will show how one can calculate the homology of these twisted bundles through direct computation. As these computations are rather cumbersome, we will only present them in the case of the Möbius strip. Afterwards, we will show how to calculate the homology of the more general class of twisted fibre bundles using the Mayer-Vietoris sequence, and spectral sequences.

8.1. Direct Computation Using Twists

We already noted that the Möbius strip can be defined as $\mathbb{R} \times [-1,1]/\mathbb{Z}$, where the $\mathbb{Z}$-action is defined by $n \cdot (x,y) = (x+n, (-1)^n y)$. Note that the quotient of $\mathbb{R}$ with $\mathbb{Z}$ is the base space S^1. One may wonder whether the we could retrieve the homological structure of the twisted bundle by considering the regular, untwisted Cartesian product of the base space and the fibre, but by encoding the twist into the boundary map of the tensor product chain complex. To this end, we consider the chain complexes associated to S^1 and $[-1,1]$, and take their tensor product:

$$C_*(S^1;\mathbb{Z}) \otimes C_*([-1,1];\mathbb{Z}) \tag{8.3}$$

Let us now consider how one can introduce a twist into the differential of these spaces. Given a path γ on the circle, and let f be a point in $[-1,1]$. We consider the element $(\gamma, f) \in S^1 \times [-1,1]$, which can be interpreted as a 1-complex on the space, and therefore ought to have a counterpart in the tensor product of the chain complexes of these respective spaces. If we are to consider this path as a path on the Möbius strip, its boundary points will depend on the loop γ: using the path lifting property to lift γ to the universal cover of S^1, $\mathbb{R}$, we see that if γ goes around an odd number of times (e.g. once), one of its endpoints is shifted due to the non-trivial $\mathbb{Z}$-action on the fibre $[-1,1]$. Hence, we propsose the following definition: given $\sigma_1 \in C_1(S^1;\mathbb{Z})$, the differential $\partial : C_1(S^1;\mathbb{Z}) \otimes C_n([-1,1];\mathbb{Z}) \to C_1(S^1;\mathbb{Z}) \otimes C_{n-1}([-1,1];\mathbb{Z}) \oplus C_0(S^1;\mathbb{Z}) \otimes C_n([-1,1];\mathbb{Z})$ is given by:

$$\partial(\sigma_1 \otimes f_n) = \sigma_1 \otimes \partial(f_n) \oplus (\sigma(1) \otimes \varphi(\sigma, \sigma(1)) f_n - \sigma(0) \otimes f_n) \tag{8.4}$$

where the *twist* φ is given by:

$$\varphi(\sigma, \sigma(1)) = (-1)^{w(\sigma_*)} \tag{8.5}$$

Here, $w(\cdot)$ returns the *winding number*.

In order to perform calculations, however, singular homology becomes rather complicated. We therefore take a cellular approach to the problem. We first consider the following cellulations:

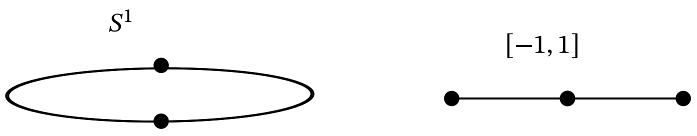


Figure 8.3: On the left: a cellulation of S^1 consisting of 2 0-cells and 2 1-cells. On the right: a cellulation of the interval $[-1.1]$ consisting of 3 0-cells and 2 1-cells.

The associated cellular complexes are therefore:

$$[-1,1]: \qquad \mathbb{Z}^2 \longrightarrow \mathbb{Z}^3$$

$$S^1: \qquad \mathbb{Z}^2 \longrightarrow \mathbb{Z}^2$$

Taking the tensor product of these spaces, we get the complex:

$$\mathbb{Z}^2 \otimes \mathbb{Z}^2 \xrightarrow{\partial_2} \mathbb{Z}^2 \otimes \mathbb{Z}^3 \oplus \mathbb{Z}^2 \otimes \mathbb{Z}^2 \xrightarrow{\partial_1} \mathbb{Z}^2 \otimes \mathbb{Z}^3$$

We now have to specify the boundary maps. We can use our previous considerations to see what this map must be:

$$\partial_2((1,0)\otimes f) = ((1,0)-(0,1)\otimes f) \oplus ((1,0)\otimes \partial(f)), \tag{8.6}$$
$$\partial_2((0,1)\otimes f) = ((0,1)\otimes f - (1,0)\otimes R(f)) \oplus ((0,1)\otimes \partial(f)) \tag{8.7}$$

And similarly for ∂_1, we find:

$$\partial_1(0 \oplus (1,0)\otimes f) = ((1,0)-(0,1))\otimes f \tag{8.8}$$
$$\partial_1(0 \oplus (0,1)\otimes f) = (0,1)\otimes f - (1,0)\otimes Rf \tag{8.9}$$
$$\partial_1(x\otimes f \oplus 0) = -x \otimes \partial(f) \tag{8.10}$$

where R is the map on the cellular complex induced by the twist φ. Indexing the 0-cells of $[-1,1]$ as f_0^0, f_1^0, f_2^0, and its 1-cells as f_0^1 and f_1^1, and taking $\partial(f_i^1) = f_i^0 - f_{i+1}^0$, that is, as follows:

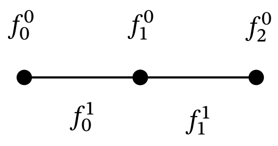


Figure 8.4: The cellulation of $[-1,1]$, along with a labelling of its cells.

We see that R maps f_1^0 to itself, and f_0^0 to f_2^0 and v.v., and, in order to commute with the differential, it maps f_0^1 to $-f_1^1$ and v.v. Note that we now indeed expect our complex to describe a cellular complex of a Möbius strip, as the twist in the boundary map changes the product complex as follows:

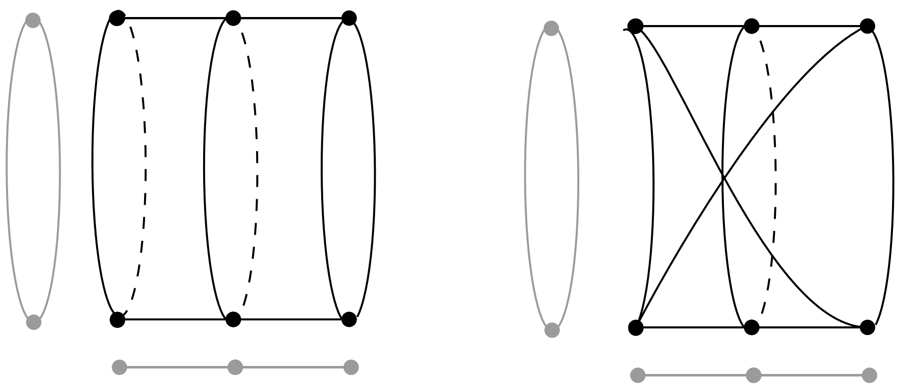


Figure 8.5: On the left: cellulation of the product space of S^1 and $[-1.1]$ (i.e. the closed strip) as induced by their respective cellulations. On the right: the cellular complex of their product space after applying the twist. One can easily see that this is, in fact, a cellulation of the Möbius strip.

We thus see that we could have constructed the twist differently, that is, as the transition function between the two charts on the base space. From this point of view, our cellular complex corresponds to the open cover on the base space on which we defined the chart.

One can now easily determine the homology of this complex. To this end, note that $\ker(\partial_2)$ is trivial, as some element $\sum_i x_i \otimes f_i$ in the kernel must satisfy $\sum_i x_i \otimes \partial f = 0$, but the boundary map on the fibre is injective.

Next, we determine the image of ∂_1. To this end, note that:

$$\partial_1(x \otimes \partial(f_i^1) \oplus 0) = -x \otimes (f_i^0 - f_{i+1}^0) \tag{8.11}$$

Furthermore, we note that:

$$\partial_1(0 \oplus (n,m) \otimes f_1^0) = (n-m, m-n) \otimes f_1^0 \tag{8.12}$$

$$\partial_1(0 \oplus (n,m) \otimes f_j^0) = (n-m, -n) \otimes f_j^0 + (0,m) \otimes f_{2-j}^0, \quad j = 0,2 \tag{8.13}$$

As such, we see that the following:

$$\begin{aligned}\partial_1((0,m) \otimes (f_0^1 + f_1^1) \oplus -(n,m) \otimes f_0^0) &= (n-m, -n+m) \otimes f_0^0 && (8.14)\\ &= (n-m, -n+m) \otimes f_1^0 + (n-m, -n+m) \otimes (f_0^0 - f_1^0) && (8.15)\\ &= (n-m, -n+m) \otimes f_1^0 + \partial_1(-(n-m, m-n) \otimes \partial_1(-f_1^1) \oplus 0) && (8.16)\end{aligned}$$

Such that:

$$\partial_1(0 \oplus (n,m) \otimes f_0^0) = \partial_1(0 \oplus (n,m) \otimes f_1^0) + \partial_1(-(n-m, m-n) \otimes \partial_1(-f_1^1) \oplus 0) \tag{8.17}$$

An analogous statement holds for $\partial_1(0 \oplus (n,m) \otimes f_0^2)$, and, as such, we see that the image of ∂_1 can be reduced to the image of vectors of the following form:

$$0 \oplus \mathbb{Z}^2 \otimes f_0^1, \quad \mathbb{Z}^2 \otimes f_i^1 \oplus 0\,(i = 0,1) \tag{8.18}$$

But note that a basis of the zeroth complex is given by:

$$(1,0) \otimes f_i^0, (0,1) \otimes f_i^0,\ i = 0,1,2 \tag{8.19}$$

Hence, we see that:

$$\mathcal{H}_0 = \mathbb{Z}[(1,0) \otimes f_i^0, (0,1) \otimes f_i^0]/\mathrm{Im}(\partial_1) = \mathbb{Z}[(0,1) \otimes f_1^0, (1,0) \otimes f_0^1]/\mathbb{Z}((1,-1) \otimes f_1^0) \simeq \mathbb{Z} \tag{8.20}$$

as expected. Finally, we turn to the computation of the first homology group.
To this end, we see that the image of ∂_2 is given by:

$$\mathrm{Im}(\partial_2) = \left\{(n, m-n) \otimes f_i^1 - (m,0) f_{1-i}^1 \oplus (n,m) \otimes (f_i^0 - f_{i+1}^0),\ n,m \in \mathbb{Z}, i = 0,1\right\} \tag{8.21}$$

We can now calculate the kernel of ∂_1:

$$\begin{aligned}&\partial_1\left((x_0 \otimes f_0^1 + x_1 \otimes f_1^1) \oplus \sum_{i=0}^{2}(n_i, m_i) \otimes f_i^0\right) && (8.22)\\ &= -x_0 \otimes (f_0^0 - f_1^0) - x^1 \otimes (f_1^0 - f_2^0) + (n_0 - m_0, m_2 - n_0) \otimes f_0^0 + (n_1 - m_1, m_1 - n_1) \otimes f_1^0 + (n_2 - m_2, m_0 - n_2) \otimes f_2^0 && (8.23)\\ &= (-x_0 + (n_0 - m_0, m_2 - n_0)) \otimes f_0^0 + (-x_1 + x_0 + (n_1 - m_1, m_1 - n_1)) \otimes f_1^0\,(x_1 + (n_2 - m_2, m_0 - n_2)) \otimes f_2^0 = 0 && (8.24)\end{aligned}$$

This yields the following set of equations:

$$\begin{cases} x_0 = (n_0 - m_0, m_2 - n_0) \\ x_0 - x_1 = -(n_1 - m_1, m_1 - n_1) \\ x_1 = -(n_2 - m_2, m_0 - n_2) \end{cases} = \begin{cases} x_0 = (n_0 - m_0, m_2 - n_0) \\ (0,0) = -(n_0 - m_0 + n_1 - m_1 + n_2 - m_2, m_1 - n_1 + m_2 - n_0 + m_0 - n_2) \\ x_1 = -(n_2 - m_2, m_0 - n_2) \end{cases} \tag{8.25}$$

where the middle two equations are linearly dependent, such that we can only deduce that:

$$m_2 = n_0 - m_0 + n_1 - m_1 + n_2 \tag{8.26}$$

As such, we can write the kernel as:

$$\left\{((n_0 - m_0, m_2 - n_0) \otimes f_0^1 - (n_2 - m_2, m_0 - n_2) \otimes f_1^1 \oplus \sum_{i=0}^{2} (n_i, m_i) \otimes f_i^0 \right\} \tag{8.27}$$

$$= \left\{((n_0 - m_0, -n_0) \otimes f_0^1 + (0, -m_0) \otimes f_1^1) \oplus (n_0, m_0) \otimes f_0^0 - (n_1, m_1) \otimes f_1^0 \right\} \cup \tag{8.28}$$

$$\left\{(0, m_2) \otimes f_0^1 + (n_2 - m_2, -n_2 \otimes f_1^1 \oplus (n_2, m_2) \otimes f_2^0 - (n_2, m_2) \otimes f_1^0 \right\} \cup \left\{0 \oplus \mathbb{Z}(1,1) \otimes f_1^0 \right\} \tag{8.29}$$

$$\simeq \mathrm{Im}\,(\partial_2) \oplus \mathbb{Z}[(1,1) \otimes f_1^0] \tag{8.30}$$

Hence, we see that:

$$\mathcal{H}_1 = \ker(\partial_1) / \mathrm{Im}\,(\partial_2) \simeq \mathbb{Z} \tag{8.31}$$

as expected.

Note that these calculations can be extended to the more general class of twisted bundles we introduced earlier, as these, too, can be defined as a quotient:

$$\mathbb{R} \times F / \mathbb{Z} \tag{8.32}$$

where now, the $\mathbb{Z}$-action is given by $n \cdot (x, f) = (x + n, Rf)$.

8.2. Computations using the Mayer-Vietoris Sequence

We now proceed to calculate the homology of our twisted bundles by making use of the Mayer-Vietoris sequence. To this end, let us define the following open sets $E_i = p^{-1}(U_i) \subseteq E$. Note that these sets E_i define an open cover on E, and hence, we can make use of the Mayer-Vietoris sequence to calculate the homology group of E in terms of their homology groups:

$$\begin{tikzcd} 0 \arrow[r] & C_n(E_1 \cap E_2) \arrow[r, "i_1 \oplus i_2"] & C_n(E_1) \oplus C_n(E_2) \arrow[r, "p_1 - p_2"] & C_n(E) \arrow[r] & 0 \end{tikzcd}$$

The point here, however, is to exploit the fibre bundle structure of E, and to make use of the local charts $\varphi_i : U_i \times F \to E_i$ to find another SES in terms of the homology groups of $U_i \times F$. With this in mind, we note that we have the following commuting square:

$$\begin{tikzcd} 0 \arrow[r] & C_n(E_1 \cap E_2) \arrow[r, "i_1 \oplus i_2"] \arrow[d, "\varphi_1"] & C_n(E_1) \oplus C_n(E_2) \arrow[r, "p_1 - p_2"] \arrow[d, "\varphi_1"] & C_n(E) \arrow[r] \arrow[d, "\mathrm{id}"] & 0 \\ 0 \arrow[r] & C_n((U_1 \cap U_2) \times F) \arrow[r, "\iota_n"] & C_n(U_1 \times F) \oplus C_n(U_2 \times F) \arrow[r, "\pi_n"] & C_n(E) \arrow[r] & 0 \end{tikzcd}$$

Where we have that:

$$\iota_n : \sigma \mapsto \sigma \oplus \varphi_2 \circ \varphi_1^{-1}(\sigma), \quad \pi_n : \sigma_1 \oplus \sigma_2 \mapsto \varphi_1^{-1}(\sigma_1) - \varphi_2^{-1}(\sigma_2) \tag{8.33}$$

This new SES is precisely the one we want, as it allows us to compute the homology of E in terms of the homology of the open cover of S^1. Let us proceed to do just that, by first considering the LES induced by this SES:

$$\begin{tikzcd} & H_1((U_1 \cap U_2) \times F) \arrow[d] & & & \\ & H_1(U_1 \times F) \oplus H_1(U_2 \times F) \arrow[r] & H_1(E) \arrow[r] & H_0((U_1 \cap U_2) \times F) \arrow[d] & \\ & & & H_0(U_1 \times F) \oplus H_0(U_2 \times F) \arrow[r] & H_0(E) \arrow[r] & 0 \end{tikzcd}$$

8.2.1. The Homology Groups of Order 0

Let us first calculate the zeroth homology group of E by applying the first isomorphism theorem and using the exacteness of the LES:

$$\begin{tikzcd}
H_0((U_1 \cap U_2) \times F) \arrow[r, "(\iota_0)_*"] & H_0(U_1 \times F) \oplus H_0(U_2 \times F) \arrow[r] \arrow[d] & H_0(E) \arrow[r] & 0 \\
& H_0(U_1 \times F) \oplus H_0(U_2 \times F)/\mathrm{im}\,((\iota_0)_*) \arrow[ru] & &
\end{tikzcd}$$

From this, we immediately deduce that:

$$H_0(E) \simeq H_0(U_1 \times F) \oplus H_0(U_2 \times F)/\mathrm{Im}\,((\iota_0)_*) \tag{8.34}$$

Here we can use the Künneth theorem, together with the fact that each U_i is a contractible space, to find that:

$$H_0(U_i \times F) \simeq H_0(U_i) \otimes H_0(F) \simeq H_0(F) \tag{8.35}$$

Moreover, we see that $U_1 \cap U_2 \times F = U_I \times F \sqcup U_R \times F$, therefore, again by the Künneth theorem, we find that:

$$H_0(U_1 \cap U_2 \times F) = H_0(U_I \times F) \oplus H_0(U_R \times F) \simeq H_0(F) \oplus H_0(F) \tag{8.36}$$

We can now proceed to calculate $(\iota_0)_*$:

$$(\iota_0)_* : H_0((U_1 \cap U_2) \times F) \to H_0(U_1 \times F) \oplus H_0(U_2 \times F) : [x] \mapsto [x] \oplus [\varphi_2 \circ \varphi_1^{-1}(x)] \tag{8.37}$$

By the Künneth theorem, we can understand this map as a map:

$$(\iota_0)_* : H_0(F) \otimes H_0(F) \to H_0(F) \oplus H_0(F) \tag{8.38}$$

Where we stick to the convention that the first term in the domain corresponds to $H_0(U_I \times F)$. We can thus determine the effect of the transition function:

$$(\iota_0)_* : [x] \oplus [0] \mapsto [x] \oplus [x] \tag{8.39}$$
$$[0] \oplus [x] \mapsto [x] \oplus [Rx] \tag{8.40}$$

We can now mod out the image of $H_0(F) \oplus 0$, which we dub $\Delta_{H_0(F)} := \{[x] \oplus [x], [x] \in H_0(F)\}$, from the image of $(\iota)_*$ using the first isomorphism theorem:

$$\begin{tikzcd}
\mathrm{im}\,((i_0)_*) \arrow[r, "{[x]\oplus[y]\mapsto[x]-[y]}"] \arrow[d] & (R-I)H_0(F) \\
\mathrm{im}\,((i_0)_*)\,/\Delta_{H_0(F)} \arrow[ru, "\simeq"] &
\end{tikzcd}$$

And similarly for its codomain:

$$\begin{tikzcd}
H_0(F) \oplus H_0(F) \arrow[r, "{[x]\oplus[y]\mapsto[x]-[y]}"] \arrow[d] & H_0(F) \\
(H_0(F) \oplus H_0(F))/\Delta_{H_0(F)} \arrow[ru, "\simeq"] &
\end{tikzcd}$$

Such that we can now use the third isomorphism theorem to find:

$$H_0(E) \simeq \left((H_0(F) \oplus H_0(F))/\Delta_{H_0(F)}\right) / \left(\mathrm{im}\,((i_0)_*)\,/\Delta_{H_0(F)}\right) \simeq H_0(F)/(R-I)H_0(F) \tag{8.41}$$

8.2.2. The First Order Homology Groups

Having calculated the zeroth order homology groups, let us take on the task of calculating the first order homology groups. To this end, we again apply the first isomorphism theorem to recover a SES from our initial LES:

$$\begin{tikzcd}
\cdots \arrow[r] & H_1((U_1 \cap U_2) \times F) \arrow[d, "(\iota_1)_*"] & & & H_0(U_1 \times F) \oplus H_0(U_2 \times F) \arrow[r] & \cdots \\
& H_1(U_1 \times F) \oplus H_1(U_2 \times F) \arrow[rr] \arrow[d] & & H_1(E) \arrow[r, "\delta_1"] \arrow[dr] & H_0((U_1 \cap U_2) \times F) \arrow[u] & \\
0 \arrow[r] & H_1(U_1 \times F) \oplus H_1(U_2 \times F)/\text{im}\,((\iota_1)_*) \arrow[urr] & & & \ker((\iota_0)_*) \arrow[r] & 0
\end{tikzcd}$$

We now have to calculate $\ker((\iota_0)_*)$ and $\text{Im}\,((\iota_1)_*)$. Let us start with the former:

$$\ker((\iota_0)_*) = \left\{ \sum n_i[x_i]_i \oplus \sum_j m_j[y_j] \, : \, \sum n_i[x_i]_i + \sum_j m_j[y_j] = 0 \wedge \sum n_i[x_i]_i \oplus \sum_j m_j R[y_j] = 0 \right\} \quad (8.42)$$

The first condition implies that we all the elements in the kernel are of the form $\sum n_i[x_i]_i \oplus \sum_j \sum -n_i[x_i]_i$. The second condition, then tells us that we must have that:

$$\sum n_i[x_i]_i = R(\sum n_i[x_i]_i) \quad (8.43)$$

But these are precisely the elements of $H_0(F)^R$.

In order to calculate the $\text{Im}\,((\iota_1)_*)$, we proceed in an analogous way as we did for the zeroth homology: first, by the Künneth formula and the fact that the U_i's are contractible, we see that:

$$H_1(U_i \times F) \simeq H_1(F), H_1(U_1 \cap U_2 \times F) \simeq H_1(F) \oplus H_1(F) \quad (8.44)$$

Furthermore, we note that the map induced by the inclusion behaves in precisely the same manner as in the zeroth order case:

$$(\iota_1)_* : [x] \oplus [0] \mapsto [x] \oplus [x] \quad (8.45)$$
$$[0] \oplus [x] \mapsto [x] \oplus [Rx] \quad (8.46)$$

Hence, we can proceed in an exactly analogous way by modding out the diagonal set $\Delta_{H_1(F)}$, and using the third isomorphism theorem to find that:

$$(H_1(F) \otimes H_1(F))/\text{Im}\,((i_1)_*) \simeq H_1(F)/(R - I)H_1(F) \quad (8.47)$$

Such that we obtain the following short exact sequence for $H_1(E)$:

$$\begin{tikzcd}
0 \arrow[r] & H_1(F)/\left((R - I)H_1(F)\right) \arrow[r] & H_1(E) \arrow[r] & H_0(E)^R \arrow[r] & 0
\end{tikzcd}$$

8.2.3. Higher Order Homology Groups

Having seen the similarity in the calculations for the zeroth and first order homology groups, one wonders whether these can be extended to determine the homology groups of arbitrary degree. To this end, let us try to reproduce these calculations for higher order groups. Fix $n \in \mathbb{N}_{\geq 1}$. We can again deduce a SES from the LES:

$$\begin{tikzcd}
\cdots \arrow[r] & H_n((U_1 \cap U_2) \times F) \arrow[d, "(\iota_n)_*"] & & & H_{n-1}(U_1 \times F) \oplus H_{n-1}(U_2 \times F) \arrow[r] & \cdots \\
& H_n(U_1 \times F) \oplus H_n(U_2 \times F) \arrow[rr] \arrow[d] & & H_n(E) \arrow[r, "\delta_n"] \arrow[dr] & H_{n-1}((U_1 \cap U_2) \times F) \arrow[u] & \\
0 \arrow[r] & H_n(U_1 \times F) \oplus H_n(U_2 \times F)/\text{im}\,((\iota_n)_*) \arrow[urr] & & & \ker((\iota_{n-1})_*) \arrow[r] & 0
\end{tikzcd}$$

Similar to the previous case, we will have to calculate ker$((\iota_{n-1})_*)$ and Im$((\iota_{n-1})_*)$. We see, however, that these calculations are completely similar! The Künneth formula tells us that:

$$H_n(U_i \times F) \simeq H_n(F), H_n(U_1 \cap U_2 \times F) \simeq H_n(F) \oplus H_n(F) \tag{8.48}$$

Now, we can again use the first and third isomorphism theorem by looking at the diagonal space $\Delta_{H_n(F)}$ to find:

$$0 \longrightarrow H_n(F)/((R-I)H_n(F)) \longrightarrow H_n(E) \longrightarrow H_{n-1}(E)^R \longrightarrow 0$$

8.3. Computations using Spectral Sequences

Let us consider try to again calculate the homology of the fibre bundle discussed in the previous section using the Leray-Serre spectral sequence. As our base space is now the circle S^1, $\mathcal{H}_p(S^1, \mathcal{H}_q(F)) = 0$ for $p \geq 2$. Hence, we obtain the following two-column spectral sequence:

$$\begin{tikzcd}
{} & {} & \vdots & \vdots & {} & {} \\
{} & 0 & E^2_{0,2} \arrow[ull] & E^2_{1,2} \arrow[ull] & 0 & {} \\
{} & 0 & E^2_{0,1} \arrow[ull] & E^2_{1,1} \arrow[ull] & 0 \arrow[ull] & {} \arrow[ull] \\
{} & 0 & E^2_{0,0} \arrow[ull] & E^2_{1,0} \arrow[ull] & 0 \arrow[ull] & {} \arrow[ull] \\
{} & {} & 0 & 0 & {} \arrow[ull] & {} \arrow[ull]
\end{tikzcd}$$

Note, however, that all the differentials in this setting have a trivial domain or codomain. Hence, we conclude that $E^2_{**} = E^3_{**}$. By induction, one easily sees that the spectral sequence actually stabilises at page 2, i.e. that $E^\infty_{**} = E^2_{**}$. Therefore, we know that such a spectral sequence induces the following SES (see e.g. [Wei94]):

$$0 \longrightarrow E^2_{p,q} \longrightarrow H_{p+q}(E) \longrightarrow E^2_{p+1,q-1} \longrightarrow 0$$

Which, by definition of the Leray-Serre spectral sequence, is equivalent to:

$$0 \longrightarrow H_p(S^1, \underline{H_q(B)}) \longrightarrow H_{p+q}(E) \longrightarrow H_{p+1}(S^1, \underline{H_{q-1}(B)}) \longrightarrow 0$$

Hence, we see that calculating the homology of a fibre bundle with a circle as its base is equivalent to calculating the homology with local coefficients of the circle.

8.3.1. The Homology with Local Coefficients of the Circle

We can use cellular homology with local coefficients to determine the homology groups of S^1. To this end, let us consider the following CW-complex on the circle:

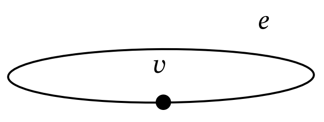


Figure 8.6: A cellular complex of the circle consisting of one 0-cell and one 1-cell.

This CW-complex lifts to a CW-complex of the universal cover of the circle, i.e.:

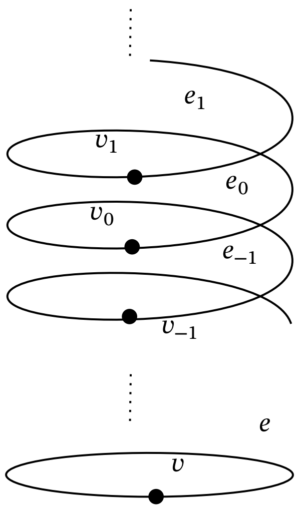


Figure 8.7: The cellular complex of the circle lifts to its universal cover as a cellular complex consisting of countably many 0-cells and countably many 1-cells (with respect to $\mathbb{Z}$).

Hence, we see that the cellular complex of the circle

$$\begin{tikzcd} 0 \arrow[r] & \mathbb{Z}[e] \arrow[r, "e\mapsto 0"] & \mathbb{Z}[v] \end{tikzcd}$$

lifts to a $\mathbb{Z}[\pi_1(X)]$-cellular complex of the universal cover:

$$\begin{tikzcd} 0 \arrow[r] & \mathbb{Z}[t,t^{-1}][e] \arrow[r, "\tilde{\partial}_1"] & \mathbb{Z}[t,t^{-1}][v] \end{tikzcd}$$

Where we see that, given the lift e_i of the edge e, its boundary is equal to $v_{i+1} - v_i$, which is the $(1-t)$-image of the vertex v of the cellular complex of the circle. Therefore, we see that $\tilde{\partial_1}(e) = (1-t)v$. Let us now proceed with the algebraic approach to the homology with local coefficients. We find the following chain complex:

$$\begin{tikzcd} 0 \arrow[r] & Z[t,t^{-1}] \otimes_{\mathbb{Z}[t,t^{-1}]} A \arrow[r, "\tilde{\partial}\otimes\mathrm{id}"] \arrow[d, "1\otimes a\mapsto a"] & Z[t,t^{-1}] \otimes_{\mathbb{Z}[t,t^{-1}]} A \arrow[r] \arrow[d, "1\otimes a\mapsto a"] & 0 \\ 0 \arrow[r] & A \arrow[r, "\tilde{\Delta}"] & A \arrow[r] & 0 \end{tikzcd}$$

Now as the maps $1 \otimes a \mapsto a$ are isomorphisms, we see that we can calculate the homology with coefficients in A by determining the induced map $\tilde{\Delta}$ and calculating its kernel and image. We now have:

$$\tilde{\Delta}(a) = (1\otimes a \mapsto a)\tilde{\partial_1}(1\otimes a) = (1\otimes a \mapsto a)(1\otimes a - t\otimes a) = (1\otimes a \mapsto a)(1\otimes a - 1\otimes Ra) = (I-R)a \tag{8.49}$$

Hence, we see that:

$$\ker(\tilde{\Delta}) = A^R, \operatorname{Im}(\tilde{\Delta}) = (I-R)A \tag{8.50}$$

And therefore, we conclude that:

$$H_0(S^1; \underline{A}) \simeq A/(I-R)A, \quad H_1(S^1 l \underline{A}) \simeq A^R \tag{8.51}$$

8.3.2. The Homology Groups of the Twisted Bundle

Taking $p = -1, q = 1$, and noting that $E^2_{p,q} = 0$, our SES immediately yields:

$$H_0(E) \simeq H_0(S^1, \underline{H_0(F)}) \simeq H_0(F)/(I-R)H_0(F) \tag{8.52}$$

Proceeding by taking $p = 0, q = 1$, we find the following SES:

$$\begin{tikzcd} 0 \arrow[r] & H_0(S^1, \underline{H_1}(F)) \arrow[r] & H_1(E) \arrow[r] & H_1(S^1, \underline{H_0(F)}) \arrow[r] & 0 \end{tikzcd}$$

which is equivalent to the following SES:

$$\begin{tikzcd} 0 \arrow[r] & H_1(F)/(I-R)H_1(F) \arrow[r] & H_1(E) \arrow[r] & H_0(F)^R \arrow[r] & 0 \end{tikzcd}$$

Proceeding inductively, we can take $p = n-1, q = 1$ for a given $n \in \mathbb{N}$ to find that the homology of the space E can be expressed in terms of the following LES:

$$\begin{tikzcd} 0 \arrow[r] & H_n(F)/(I-R)H_n(F) \arrow[r] & H_n(E) \arrow[r] & H_{n-1}(F)^R \arrow[r] & 0 \end{tikzcd}$$

9 Twisted Product Codes

In this chapter, we introduce two related product code constructions, which are inspired by the idea of *twisting* the product of two chain complexes.

9.1. Fibre Bundle Codes

In this section, we introduce the fibre bundle product construction as presented in [HHO21]. To this end, we first substantiate the construction of these codes theoretically using the theory from our discussion of fibre bundles as presented in the previous chapters, after which we formalise these results and present two theorems to determine the number of encoded qubits k. Before proceeding, we remark that no general way of determining the weight and the distance of such codes has currently been presented in the literature.

9.1.1. From Hypergraph Products to Fibre Bundle Products

Let us consider two length-1 chain complexes, corresponding to classical codes, say $\mathcal{F}$ and $\mathcal{B}$, which arise due to cellulations of two topological spaces, say $X_{\mathcal{F}}$ and $X_{\mathcal{B}}$. When constructing the hypergraph product, one takes the tensor product of these complexes. This corresponds to the chain complex of the Cartesian product of these spaces, i.e. of the space $X_{\mathcal{B}} \times X_{\mathcal{F}}$. If, for instance, we take $\mathcal{B}$ to arise from a cellular complex of a circle (i.e. a repetition code), and $\mathcal{F}$ from a cellular complex of a straight line (i.e. a repetition code with open boundaries), their hypergraph product yields a cellular complex of a strip $S^1 \times [-1, 1]$, as is displayed in Figure 9.1.

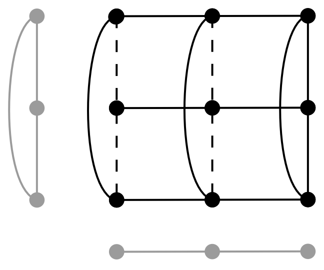


Figure 9.1: The hypergraph product of a repetition code arising from a cellulation of a circle, and a repetition code with open boundaries arising from a cellulation of a straight line. We see that this product yields a cellular complex of a string.

Our discussion of fibre bundles may lead one to wonder whether we could generalise this code construction, which considers just product spaces, to the class of fibre bundles. For instance, following our discussion in Chapter 8, we know that one introduce a *twist* to the differential of the tensor product complex, in order to define a cellular complex of the Möbius strip, as is shown in Figure 9.2.

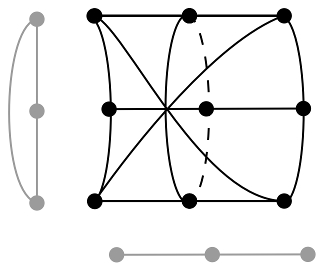


Figure 9.2: The twisted product a cellulation of a circle and a cellulation of a straight line. We see that this product yields a cellular complex of a Möbius strip. By interpreting the cellular complexes as a repetition code and a repetition code with open boundaries, respectively, we see that this product yields a new code, which differs from the hypergraph product of these two codes.

More formally, let us consider the following fibre bundle:

$$\begin{tikzcd} X_{\mathcal{F}} \arrow[r] & X_{\mathcal{F}} \times_{\varphi} X_{\mathcal{B}} \arrow[d] \\ & X_{\mathcal{B}} \end{tikzcd}$$

Where we use the (somewhat suggestive) notation $X_{\mathcal{F}} \times_{\varphi} X_{\mathcal{B}}$ to denote the total space. If $X_{\mathcal{B}}$ admits a universal cover, by the Borel construction (Theorem 6.11), we can interpret our fibre bundle as the principal $\pi_1(B)$-bundle $\tilde{X}_{\mathcal{B}} \times F/\pi_1(X_{\mathcal{B}})$. Assuming, moreover, that $\pi_1(X_{\mathcal{B}})$ is cyclic[1], we can actually find a *twist* $\varphi \in \text{Aut}(X_{\mathcal{F}})$ such that $g \cdot (x, f) = (gx, \varphi f)$, where g is the generator of $\pi_1(X_{\mathcal{B}})$. As we saw in the Chapter 8, one can use this twist to define a twisted differential on the tensor product complex of these two codes, which we denote by ∂^{φ}. The complex, together with this new differential, is denoted by $\mathcal{B} \otimes_{\varphi} \mathcal{F}$. This is the fundamental idea behind the fibre bundle code construction: by associating a so-called *base complex* to the base space, and a *fibre complex* to the fibre space, and by considering the twist induced by the associated bundle, one can create a new CSS code by considering the $\otimes_{\varphi}$-product complex of these two complexes.

9.1.2. Formal Construction of the Fibre Bundle Product

The fibre bundle construction considers two chain complexes: a so-called *base complex* $\mathcal{B}$ of length 1, and a *fibre complex* $\mathcal{F}$. The fibre complex is assumed to admit an automorphism group Aut($\mathcal{F}$), consisting of all the isomorphic chain maps from $\mathcal{F}$ to $\mathcal{F}$. As such, we can define a so-called *connection*, i.e. a map

$$\varphi : \{(b, a) : b, a \text{ cells s.t. } a \in \partial(b)\} \to \text{Aut}(\mathcal{F}) \tag{9.1}$$

which is said to associate *twists* to the pairs (b, a). Using this connection, we can define a different boundary map ∂ on $\mathcal{B} \otimes \mathcal{F}$ as the map induced by the billinear maps:

$$B_0 \times F_p \to (\mathcal{B} \otimes \mathcal{F})_p : (x, y) \mapsto x \otimes \partial_p^{\mathcal{F}}(y) \tag{9.2}$$

$$B_1 \times F_p \to (\mathcal{B} \otimes \mathcal{F})_{p+1} : (x, y) \mapsto x \otimes \partial_p^{\mathcal{F}}(y) + \sum_{a \in \partial x} a \otimes \varphi(x, a) y \tag{9.3}$$

One can prove that this indeed defines a proper differential. Furthermore, we see that when φ maps to the trivial automorphism, we retrieve the differential associated to the (non-twisted) tensor product. Lastly, it should be noted that this construction is more generic than the topologically motivated construction we presented in the last section, as the codes at hand do not necessarily arise from cellulations of topological spaces, and even if such topological spaces can be found, one is not guaranteed of the existence of a universal cover for the base space, nor of the cyclicity of its fundamental group.

9.1.3. Determining the Number of Encoded Qubits

In this section, we present two ways of determining the number of encoded qubits. Firstly, we prove the following theorem, which was first presented in [BE21a]:

[1]We make this assumption for reasons of clarity now. We will relax this assumption later on.

Theorem 9.1. *Let $\mathcal{B}$ denote the length 1 base complex, and let $\mathcal{F}$ denote the fibre complex. Given that φ acts trivially on the homology of $\mathcal{F}$, we have the following Künneth formula:*

$$H_n(\mathcal{B} \otimes_\varphi \mathcal{F}) \simeq \bigoplus_{p+q=n} H_p(\mathcal{B}) \otimes_{\mathbb{F}_2} H_q(\mathcal{F}) \tag{9.4}$$

Proof. As the $\otimes_\varphi$-complex arises from the regular $\mathbb{F}_2$-tensor product of chain complexes, we can consider the double complex $(\mathcal{B} \otimes \mathcal{F})_{**}$. By theorem 7.1, there is a spectral sequence E^r_{**} converging to the homology of the total complex, where E^2_{**} is obtained by first applying the homology with respect to the differentials $\mathrm{id} \otimes \partial^{\mathcal{F}}$, and then with respect to the twisted differentials. As $\mathcal{F}$ is a length-1 complex, the zeroth page actually contains just two columns. Hence, starting from the third page, the differentials vanish, causing the spectral sequence to stabilise at E^2_{**}. The homology with respect to $\mathrm{id} \otimes \partial^{\mathcal{F}}$ of $\mathcal{B}_* \otimes \mathcal{F}_*$ is simply $\mathcal{B}_* \otimes \mathcal{H}_*(\mathcal{F})$. Taking homology with respect to the twisted differential now yields $\mathcal{H}_*(\mathcal{B}) \otimes \mathcal{H}_*(\mathcal{F})$, as the twist has a trivial action on $\mathcal{H}_*(F)$. □

We now try to generalise this result by relaxing the condition that φ acts trivially on the homology of $\mathcal{F}$. By imposing additional structure on the chain complexes and the twist, we prove the following (new) generalisation of the Künneth theorem:

Theorem 9.2. *Let $\mathcal{B}$ be a length 1-complex which can be interpreted as the cellulation of a compact topological space $X_{\mathcal{B}}$ admitting a universal cover, and let $\mathcal{F}$ be an arbitrary chain complex which arises as a cellulation of a topological space $X_{\mathcal{F}}$. Denoting a twist by φ, the homology of the cellular complex relating to the code $\mathcal{B} \otimes_\varphi \mathcal{F}$ can be expressed as:*

$$\mathcal{H}_n\left(\mathcal{B} \otimes_\varphi \mathcal{F}\right) \simeq \mathcal{H}_1(\mathcal{B}; \underline{\mathcal{H}_{n-1}(\mathcal{F})}) \oplus \mathcal{H}_0(\mathcal{B}; \underline{\mathcal{H}_n(\mathcal{F})}) \tag{9.5}$$

Proof. Firstly, we note that as $X_{\mathcal{B}}$ is compact, its fundamental group $\pi_1(X_{\mathcal{B}})$ is finitely generated. For every generator $g \in \pi_1(X_{\mathcal{B}})$, define a $\pi_1(X_{\mathcal{B}})$-action on $X_{\mathcal{F}}$ which is induced by φ. Using this action, we can associate a principal $\pi_1(X_{\mathcal{B}})$-bundle to the space $\mathcal{B} \otimes_\varphi \mathcal{F}$, whose total space we denote by $X_{\mathcal{F}} \times_\varphi X_{\mathcal{B}}$. We will again apply the Serre spectral sequence to calculate the homology of the total space. Much like in our calculations from Chapter 8, we have that $\mathcal{H}_n(\mathcal{B}) = 0$ for $n \geq 2$. Therefore, we again find a two-column spectral sequence, and can thus simply repeat the arguments presented in Section 8.3 to find the following SES for the homology groups of the total space:

$$\begin{tikzcd} 0 \arrow[r] & \mathcal{H}_p(\mathcal{B}; \underline{\mathcal{H}_q(\mathcal{F})}) \arrow[r] & \mathcal{H}_{p+q}\left(X_{\mathcal{F}} \times_\varphi X_{\mathcal{B}}\right) \arrow[r] & \mathcal{H}_{p+1}(\mathcal{B}; \underline{\mathcal{H}_{q-1}(\mathcal{F})}) \arrow[r] & 0 \end{tikzcd}$$

Which, as we are working over the field $\mathbb{F}_2$, splits, such that:

$$\mathcal{H}_{p+q}\left(X_{\mathcal{F}} \times_\varphi X_{\mathcal{B}}\right) \simeq \mathcal{H}_{p+1}(\mathcal{B}; \underline{\mathcal{H}_{q-1}(\mathcal{F})}) \oplus \mathcal{H}_p(\mathcal{B}; \underline{\mathcal{H}_q(\mathcal{F})}) \tag{9.6}$$

Taking $q = n$ and $p = 0$ yields the desired result. □

9.2. Balanced Product Codes

9.2.1. From Twists to Actions

We have thus far seen how the fibre bundle product construction generalised the hypergraph product construction by introducing a twist in the differential, and we have seen how this shift can be motivated by considering classical codes that correspond to cellulations of the base space and fibre of a fibre bundle. Let us again consider two length-1 chain complexes $\mathcal{F}$ and $\mathcal{B}$, which arise due to cellulations of two topological spaces, which we call $X_{\mathcal{F}}$ and $X_{\mathcal{B}}$. We again assume that $X_{\mathcal{B}}$ is compact and has a universal cover $\tilde{X}_{\mathcal{B}}$, and let us now assume that we can associate a length 1 chain complex $\tilde{\mathcal{B}}$ to $\tilde{X}_{\mathcal{B}}$ as well. Now let us again consider the following fibre bundle:

$$\begin{tikzcd} X_{\mathcal{F}} \arrow[r] & X_{\mathcal{F}} \times_\varphi X_{\mathcal{B}} \arrow[d] \\ & X_{\mathcal{B}} \end{tikzcd}$$

While in the fibre bundle product construction, the twist was encoded in the differential, the idea behind the balanced product code is to encode the twist into the product of the two spaces using the Borel construction. For the fibre bundle at hand, this construction yields the associated $\pi_1(X_{\mathcal{B}})$-bundle:

$$\begin{tikzcd} X_{\mathcal{F}} \arrow[r] & (\tilde{X}_{\mathcal{B}} \times X_{\mathcal{F}})/\pi_1(X_{\mathcal{B}}) \arrow[d] \\ & X_{\mathcal{B}} \end{tikzcd}$$

Given an appropriate choice of the cellulations, the $\pi_1(X_{\mathcal{B}})$-action on $\tilde{X}_{\mathcal{B}}$ induces an action on the cellular complex $\tilde{X}_{\mathcal{B}}$, and similarly, the action on $X_{\mathcal{F}}$ induces an action on the cellular complex of $X_{\mathcal{F}}$. Hence, we see that the following holds:

$$C_*\left(\left(\tilde{X}_{\mathcal{B}} \times X_{\mathcal{F}}\right)/\pi_1(X_{\mathcal{B}})\right) \simeq C_*(\tilde{X}_{\mathcal{B}} \times \tilde{X}_{\mathcal{F}})/\pi_1(X_{\mathcal{B}}) \simeq \left(\left(\tilde{\mathcal{B}}_* \otimes \mathcal{F}_*\right)/\pi_1(X_{\mathcal{B}})\right)_* \tag{9.7}$$

Defining this newly constructed complex to be the balanced product complex, we see that the differentials of this complex are simply the maps obtained from $\partial^{\mathcal{B}} \otimes \mathrm{id}$ and $\mathrm{id} \otimes \partial^{\mathcal{F}}$ after descending to the quotient complex.

9.2.2. Formal Construction of the Balanced Product

The balanced product construction [BE21a] can be formalised as follows. Given two vector spaces V, W, and suppose that V has a right-action by the group G, while W has a left-action by this group. The *balanced product* of V and W, $V \otimes_G W$, is then defined as the *covariants of* $V \otimes W$ *under the group action of* G, i.e.:

$$V \otimes_G W := (V \otimes W)/\langle vg \otimes w - v \otimes gw \rangle \tag{9.8}$$

We can extend this notion to chain complexes of vector spaces: consider two chain complexes of vector spaces $\mathcal{C}$ and $\mathcal{D}$ along with a group G, and suppose that there is a linear left G-action on $\mathcal{D}$ and a linear right-action on $\mathcal{C}$ which restrict to an action on the bases of these chain complexes. We can then define the chain complex $(C \otimes_G D)_*$ as the complex given by:

$$(C \otimes_G D)_n := \bigoplus_{p+q=n} C_p \otimes_G D_q \tag{9.9}$$

The differentials are now simply the maps induced by the differentials $\mathrm{id} \otimes \partial$ and $\partial \otimes \mathrm{id}$.

9.2.3. Determining k for the Balanced Product

We first note that the following result has been derived by the authors in [BE21a]:

Theorem 9.3. *Let G be a finite group such that $|G|$ mod $2 = 1$. Then:*

$$\mathcal{H}_n(C \otimes_G D) \simeq \bigoplus_{p+q=n} \mathcal{H}_p(C) \otimes_G \mathcal{H}_q(D) \tag{9.10}$$

Based on the motivation for these codes, we see that we can prove a theorem similar to the new generalisation of the Künneth formula that we derived for fibre bundle product codes.

Theorem 9.4. *Let $X_{\mathcal{B}}$ be a compact topological space that admits a universal $\tilde{X}_{\mathcal{B}}$. Assume moreover that this space admits a CW-complex of degree 1, such that its universal cover has a cellulation given by the 1-complex $\tilde{\mathcal{B}}$. Moreover, suppose that we have another chain complex of arbitrary length representing a cellulation of a topological space, say $X_{\mathcal{F}}$, which we denote by $\mathcal{F}$. We then have the following formula for the homology of the balanced product code $\tilde{\mathcal{B}} \otimes_{\pi_1(X_{\mathcal{B}})} \mathcal{F}$:*

$$\mathcal{H}_n\left(\tilde{\mathcal{B}} \otimes_G \mathcal{F}\right) \simeq \mathcal{H}_1(\mathcal{B}; \underline{\mathcal{H}_{n-1}(\mathcal{F})}) \oplus \mathcal{H}_0(\mathcal{B}; \underline{\mathcal{H}_n(\mathcal{F})}) \tag{9.11}$$

Proof. We note that modding out the $\pi_1(X_{\mathcal{B}})$-action is equivalent to taking the total space $(\tilde{X}_{\mathcal{B}} \times X_{\mathcal{F}})/\pi_1(X_{\mathcal{B}})$. But this is the principal $\pi_1(X_{\mathcal{B}})$-bundle belonging to the fibre bundle with base space $X_{\mathcal{B}}$ and fibre $X_{\mathcal{F}}$. Hence, we can evoke the proof of Theorem 9.2 by taking the twist to be the action induced by the generators of $\pi_1(X_{\mathcal{B}})$ on the fibre. By compactness of $X_{\mathcal{B}}$, we note that $\pi_1(X_{\mathcal{B}})$ is finitely generated. □

Note that we can generalise this last theorem to arbitrary covers C of our base space, provided that we mod out the $\pi_1(B)/G_C$-action, where G_C is the subgroup corresponding to the covering space.

9.3. Relating Fibre Bundle Products, Balanced Products and Lifted Products

The final theorem outlines how we can relate fibre bundle products codes to balanced product codes. Given a length-1 complex $\mathcal{B}$, and an arbitrary complex $\mathcal{F}$ with twist φ, we see that we can interpret the fibre bundle product $\mathcal{B} \otimes_\varphi \mathcal{F}$ as the balanced product $\mathcal{C} \otimes_G \mathcal{F}$ under the following conditions: firstly, both $\mathcal{B}$ and $\mathcal{F}$ must be cellulations of topological spaces. Secondly, the space associated to $\mathcal{B}$ must admit a universal cover, and, thirdly, it must be compact. Fourthly, the twist φ must be the map on the fibre induced by the action of the fundamental group of the space corresponding to $\mathcal{B}$. In this case, given a covering space $X_\mathcal{C}$ of the space associated to the complex $\mathcal{B}$, and let G_C denote the subgroup corresponding to this covering space, one can take $\mathcal{C}$ to be the cellulation corresponding to this covering space. By then taking $G = \pi_1(B)/G_C$, we find that this fibre bundle product can be interpreted as a balanced product. The converse result holds true as well.

Furthermore, we argue that the balanced product construction can be seen as a generalisation of the lifted product (as was also noted in [BE21a]): given an abelian group G, we can define $\mathcal{R} := \mathbb{F}_2 G$. In deriving our formula for k for the lifted product, we obtained the following expression for two $\mathcal{R}$-bimodules A and B:

$$A \otimes_\mathcal{R} B \simeq (A \otimes_{\mathbb{F}_2} B)_G \tag{9.12}$$

Taking the balanced product with G of two chain complexes $\mathcal{A}$ and $\mathcal{B}$, per definition, yields:

$$(\mathcal{A} \otimes_G \mathcal{B})_n := (\mathcal{A} \otimes_{\mathbb{F}_2} \mathcal{B})_G \tag{9.13}$$

Hence, we see that indeed:

$$(\mathcal{A} \otimes_G \mathcal{B})_* = (A \otimes_\mathcal{R} B)_* \tag{9.14}$$

Conversely, we see that the balanced product taken with any finite, abelian group G can be interpreted as the lifted product with $\mathbb{F}_2$-algebra $\mathcal{R} := \mathbb{F}_2 G$.

10 The Toric Code, But With a Twist

At the very beginning of this thesis, we noted that the torus $\mathbb{T}$ can be understood as the Cartesian product of two circles, i.e. $\mathbb{T} \simeq S^1 \times S^1$. As such, we see that the torus can actually be interpreted as a trivial fibre bundle with base and fibre S^1. By the Eilenberg-Zilber theorem, we have that:

$$C(\mathbb{T})_* \simeq C(S^1 \times S^1)_* \simeq C(S^1)_* \otimes C(S^1)_* \tag{10.1}$$

Combining this with our earlier observation that a cellular complex of S^1 can be interpreted as the chain complex associated to the repetition code, we therefore conclude that the toric code can be interpreted as the hypergraph product of two repetition codes.

Given our previous discussion regarding twisted product codes, these considerations may lead one to wonder how applying a twist to the toric code alters its performance. We can rephrase this question in concrete terms using the formalism of the fibre bundle and balanced product codes. In this chapter, we will therefore use the formalism of these fibre bundle product codes to characterise the performance of the twisted toric code. Before explaining how we will do so, however, let us first present the twisted toric codes that we will study.

10.1. The Framework

We start off this section by recalling that the repetition code can be associated to a cellulation of the circle consisting of an equal number (say, n) of 0-cells and 1-cells. Such a cellulation can, in its turn, be interpreted as a (regular) polygon, as is shown in Figure 10.1.

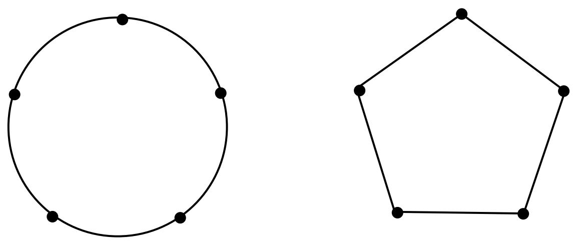


Figure 10.1: On the left: the cellulation of the circle corresponding to the repetition 5 code. On the right: up to continuous deformation, one can see that this cellular complex is also a cellular complex of the pentagon.

This allows us to determine the automorphism group quite simply: it is the *dihedral group* D_{2n}. For the purpose of our analysis, however, it is convenient to consider automorphisms that do not distinguish between different 0-cells. We therefore consider the more well-behaved subgroup of this automorphism group, which is generated by all the regular shifts. Such a shift acts on each vertex i by mapping it to the vertex $(i + k)$ mod n for some fixed k, and similarly for each edge, and is therefore isomorphic to the cyclic group C_n.

Now, let us apply the formalism from Section 9.1.2 by taking the repetition code $R_{n_{\mathcal{F}}}$ as our fibre $\mathcal{F}$, and the repetition code $R_{n_{\mathcal{B}}}$ as our base code $\mathcal{B}$. For convenience, we will assume that $n_{\mathcal{F}}$ and $n_{\mathcal{B}}$ are even throughout this entire chapter. We can then index the i-chains of $\mathcal{F}$ as f_j^i for $i = 0, 1$ and $j = 1, \dots, n_{\mathcal{F}}$, while for $\mathcal{B}$, we can index them as b_j^i in an analogous manner. As before, we take the parity check matrices of these codes to be the square matrix mapping the i^{th} edge to the i^{th} vertex plus the $i-1^{\text{th}}$ vertex (modulo n).

Having established our framework more precisely, we can move on to constructing the twisted toric code. To this end, we again note that the hypergraph product of these two codes must correspond to the (untwisted) toric code, as can be seen in Figure 10.2.

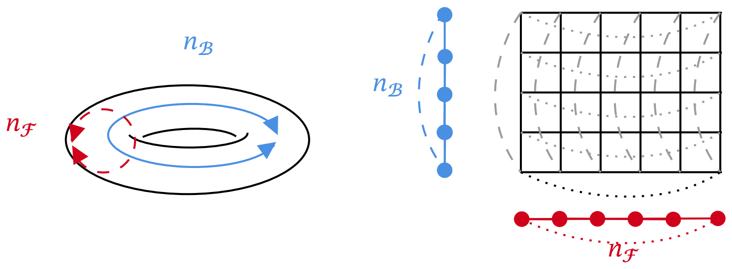


Figure 10.2: On the left: a torus upon which one can implement a toric code. On the right: the corresponding toric code, interpreted as the hypergraph product of two repetition codes with size $n_{\mathcal{B}}$ and $n_{\mathcal{F}}$, respectively. We note that qubits reside on the edges of the lattice. The dashed and dotted curved edges ensure the periodic boundary conditions. In this example, $n_{\mathcal{B}} = 5$ and $n_{\mathcal{F}} = 6$.

As such, we already know the vector spaces of which the chain complex of the twisted toric code is composed. We therefore turn our attention to establishing the last piece of the possible, which is constructing its differentials.

10.2. Defining the Twisted Differential

In order to construct the differentials of the toric code, let us first consider how applying a twist in the form of an *even* shift k alters the lattice. This is displayed in Figure 10.3.

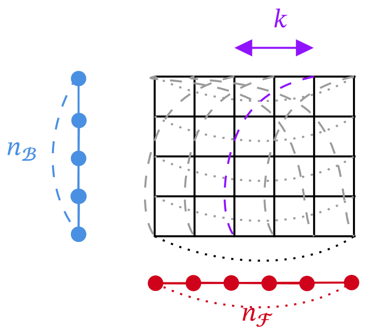


Figure 10.3: The twisted toric code with twist k when interpreted as the fibre bundle product of two repetition codes of size $n_{\mathcal{B}}$ and $n_{\mathcal{F}}$. Due to the twist, the upper end of all the edges connecting the upper and lower sides of the lattice is displaced.

We therefore see that we can describe this twist using the following map φ:

$$\varphi(b_i^1, b_j^0) f_l^k = \begin{cases} f^k_{(l+k \bmod n_{\mathcal{F}})} & i = n_{\mathcal{B}} \wedge j = -1 \bmod n_{\mathcal{B}} \\ f_l^k & \text{otherwise} \end{cases} \tag{10.2}$$

From the fibre bundle product construction (as presented in Section 9.1.2), we see that the twisted differential $\partial_1 : B_1 \otimes F_0 \oplus B_0 \otimes F_1 \to B_0 \otimes F_0$ is given by:

$$\partial_1(b_i^1 \otimes f_k^0 \oplus 0) = b_i^0 \otimes \varphi(b_i^1, b_i^0) f_k^0 + b_{i-1 \bmod n_{\mathcal{B}}}^0 \otimes \varphi(b_i^1, b_{i-1 \bmod n_{\mathcal{B}}}^0) f_k^0 \tag{10.3}$$

$$= \begin{cases} (b_i^0 - b_{i-1 \bmod n_{\mathcal{B}}}^0) \otimes f_k^0 & i \neq n_{\mathcal{B}} \\ b_i^0 \otimes f_k^0 + b_{i-1 \bmod n_{\mathcal{B}}}^0 \otimes f_{i+k \bmod n_{\mathcal{F}}}^0 & i = n_{\mathcal{B}} \end{cases} \tag{10.4}$$

This differential can be written in a clearer form by introducing the permutation matrices $P_k \in \mathcal{M}_{n\times n}(\mathbb{F}_2)$ with $(P_k)_{ij} = \delta_{j,i+k \bmod n}$, as well as matrices $U \in \mathcal{M}_{n\times n}(\mathbb{F}_2)$ for which $U_{ij} = \delta_{i,n}\delta_{j,1}$. We now see that the action of the twisted differential on $B_1 \otimes F_0$ can be written as:

$$\partial_1|_{B_1 \otimes F_0 \oplus 0} = (H_{\mathcal{B}} - U) \otimes I + U \otimes P_k \tag{10.5}$$

Such that the differential becomes:

$$\partial_1 = [(H_{\mathcal{B}} - U) \otimes I + U \otimes P_k | I \otimes H_{\mathcal{F}}^T] \tag{10.6}$$

where $H_{\mathcal{B}}$ is the parity check matrix of the repetition code with size $n_{\mathcal{B}}$, and, similarly, $H_{\mathcal{F}}$ is the parity check matrix of the repetition code with size $n_{\mathcal{F}}$.

10.3. The Effects of the Twist on the Code's Performance

As mentioned in the introduction, we would like to determine how the performance of this newly constructed twisted toric code compares to that of the regular, untwisted toric code. As this twisted code is a fibre bundle product code, and given that the authors in [HHO21] were able to improve upon the distance results from [TZ09] by applying twists, one would hope to see an increase in the distance of the toric code after applying a twist without causing a decrease in its encoding rate. We postulate that this is indeed the case for our choice of twists.

The fact that the number of encoded qubits (and hence, the rate) does not change can be seen quite easily: as the twist acts as the identity on the homology of the fibre, we see that the Künneth formula still holds. Similarly, the chain complex corresponding to the twisted toric code is still a cellulation of the torus, hence we see that the homology of the code remains unchanged after the twist. These two different arguments, therefore, each allow us to safely deduce that the rate remains unchanged.

Determining the distance of this twisted toric code, however, requires much more work. We will therefore tackle this question later on by first presenting our analytical derivation, and by verifying the validity of these results numerically.

Finally, when considering the regular, untwisted toric code, it is well known that the logical error rate of this code improves as one increases the distance of the code by increasing the lattice size, as can be seen in Figure 10.4.

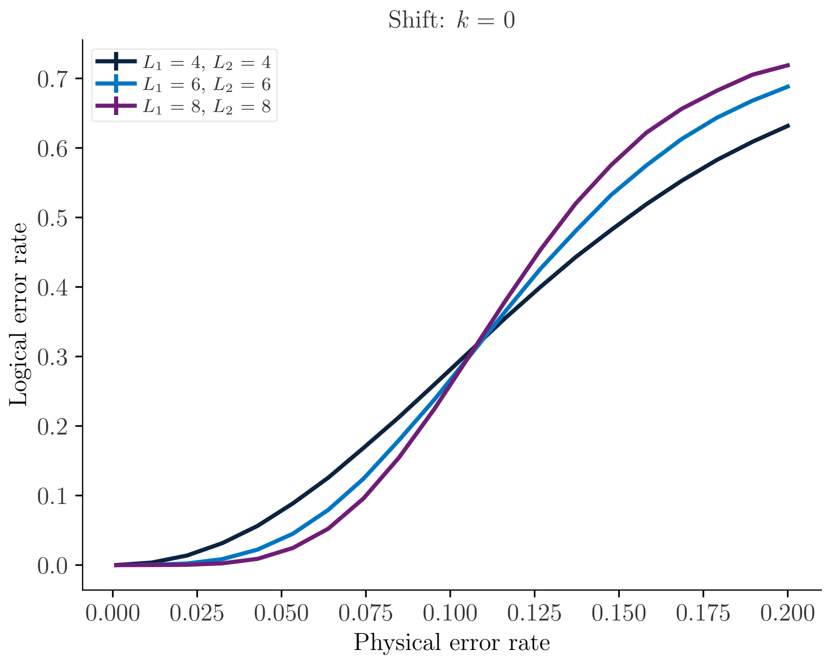


Figure 10.4: The logical error rate as a function of the physical error rate for the toric code defined on a square lattice of with sides of length 4, 6 and 8, respectively. One can see that below some threshold value of around $p \sim 10\%$, an increase in the code size (and hence, the distance) implies a decrease in the logical error rate. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates. The underlying error model was taken to be an i.i.d. Bernouilli distribution with parameter p (the physical error rate) for each of the physical qubits.

One would hope, therefore, that improving the distance of the toric code by applying a twist to the code would improve the logical error rate of the code. The distance of a code, however, is but one of its key performance indicators and previous research [BBKM19] has shown that a higher distance does not necessarily translate into an improved scaling of the logical error rate of the code. In [BBKM19], a comparison was made between the regular toric code and a "rotated" toric code. Moreover, based on the ideas of [DKLP02], analytic expressions for the logical error rate were sought and compared in order to be able to explain their numerical results. In the same vein, we will compare the twisted and the untwisted toric code by determining and comparing the scaling of the logical error rates of these codes, and by explaining this scaling behaviour for low values of p using analytical considerations.

10.4. The Distance of the Twisted Toric Code

In this section, we first determine the distance of the twisted toric code, after which we verify these results using numerical methods.

10.4.1. Analytical Derivation for the Distance

As seen in Chapter 2, the logical operators of the toric code correspond to closed loops on the torus. Let us, therefore, consider how applying a twist affects these closed loops on the torus. For now, let us assume that $k < n_{\mathcal{B}}, n_{\mathcal{F}}$. We can then first consider closed loops in the direction of the fibre, as displayed in Figure 10.5:

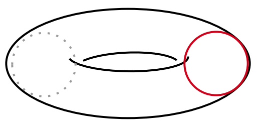


Figure 10.5: One loop (in red) on the torus. This loop corresponds to a minimum weight logical operator on the toric code. The dotted circle indicates where the twist is applied.

One can easily see that these loops are not affected by the twists. Therefore, one expects these loops to still correspond to logical operators of the twisted toric code. This should not be too surprising, as such loops correspond to the elements of $0 \oplus \ker\left(I \otimes H_{\mathcal{F}}^T\right)$, which is still a subspace of the kernel of the twisted differential.

The loops running in the direction of the base, however, do change, as the shift causes their endpoints to become detached. This is displayed in Figure 10.6.

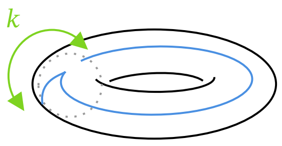


Figure 10.6: Due to the twist, the endpoints of some of the loops that were initially closed on the untwisted torus become detached. Therefore, these loops no longer correspond to logical operators once a twist is applied.

Logical operators, however, correspond to closed loops. Hence, in order to close these loops, one will have to compensate for the displacement of the endpoints by twisting one of them back. In principle, there are two ways to do so, which are shown in Figure 10.7.

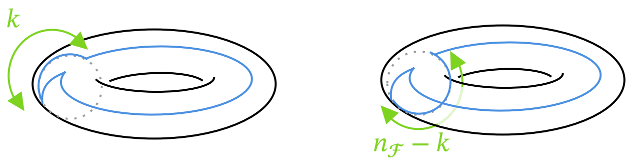


Figure 10.7: On the left: the loop is closed by moving in the direction opposite to the twist. On the right, this is done by moving in the direction of the twist. In the former case, the loop becomes k units longer, while in the latter case, the loop becomes $n_{\mathcal{F}} - k$ units longer.

As we are only interested in determining the minimum weight logical operators, only the shortest of these loops will be relevant for determining the minimum distance of the twisted toric code.

We can understand all of this quantitatively as well. In the untwisted case, these loops correspond to the elements of $\ker(H_{\mathcal{B}} \otimes I) \oplus 0 = \ker(H_{\mathcal{B}}) \otimes \mathbb{F}_2^{n_{\mathcal{F}}} \oplus 0$. The elements of $\ker(H_{\mathcal{B}})$ now, are precisely the closed loops $\sum_{i=1}^{n_{\mathcal{B}}} b_i^1$, such that the minimum weight loops are of the form $\sum_{i=1}^{n_{\mathcal{B}}} b_i^1 \otimes f$, for some $f \in \mathbb{F}_2^{n_{\mathcal{F}}}$. When introducing a twist, however, these elements are no longer in the kernel:

$$\begin{aligned}((H_{\mathcal{B}} - U) \otimes I + U \otimes P_k)\left(\sum_{i=1}^{n_{\mathcal{B}}} b_i^1 \otimes f_j^0\right) &= \sum_{i=2}^{n_{\mathcal{B}}} (b_i^0 + b_{i-1}) \otimes f_j^0 + b_1 \otimes f_j^0 + b_{n_{\mathcal{B}}}^0 \otimes f_{j+k \bmod n_{\mathcal{F}}}^0 \qquad (10.7)\\ &= b_{n_{\mathcal{B}}}^0 \otimes (f_j^0 + f_{j+k \bmod n_{\mathcal{F}}}^0) \qquad (10.8)\end{aligned}$$

This expresses quantitatively how the loops get detached. To mitigate the effects of the twist, one will have to compensate by twisting the final endpoint back. Here again, we see that there are two different ways of doing it, namely by either adding an element of the following form:

$$b_{n_{\mathcal{B}}}^0 \otimes \sum_{i=0}^{k} f_{j+i \bmod k}^0 \qquad (10.9)$$

Which is equivalent to moving in the direction opposite to the twist, or, one can add an element of the following form:

$$b_{n_{\mathcal{B}}}^0 \otimes \sum_{i=0}^{n_{\mathcal{F}}-k} f_{j-i \bmod k}^0 \qquad (10.10)$$

which corresponds to moving along with the twist.

We now want to work towards an explicit expression for the distance. To this end, let us first agree to denote a representative of the first class of logical operators by $\overline{Z}_1$, and the second class of logicals by $\overline{Z}_2$, as can be seen in Figure 10.8.

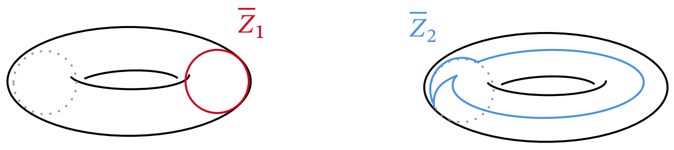


Figure 10.8: The red loops that remain unchanged by the twist are associated with the logical operator $\overline{Z}_1$, while the blue loops are associated with the logical operator $\overline{Z}_2$.

In this case, there are two possible loops that correspond to the logical $\overline{Z}_1\overline{Z}_2$:

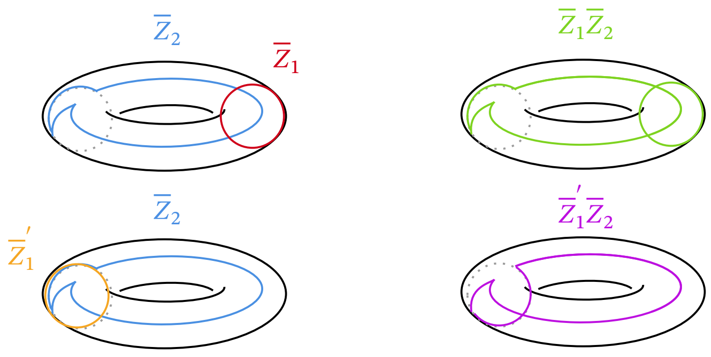


Figure 10.9: Top row: On the left, two loops corresponding to the logical operators $\overline{Z}_1$ and $\overline{Z}_2$ are given. On the right, their product $\overline{Z}_1\overline{Z}_2$ is given. This loop is clearly of length $n_{\mathcal{F}} + n_{\mathcal{B}} + k$. Bottom row: on the left, two loops corresponding to the logical operators $\overline{Z}_1'$ and $\overline{Z}_2$ are given. On the right, their product $\overline{Z}_1'\overline{Z}_2$ is given. This loop is clearly of length $n_{\mathcal{F}} + n_{\mathcal{B}} - k$.

Both of these classes of loops are of a higher weight than the loops corresponding to $\overline{Z}_1$ and $\overline{Z}_2$. Therefore, we can safely conclude that given an $n_{\mathcal{B}}$ by $n_{\mathcal{F}}$ lattice and some twist $k < \min\{n_{\mathcal{F}}, n_{\mathcal{B}}\} = n_{\mathcal{B}}$, the minimum distance of the twisted toric code is given by:

$$\min\{n_{\mathcal{F}}, \min\{n_{\mathcal{B}} + k, n_{\mathcal{B}} + (n_{\mathcal{F}} - k)\}\} \tag{10.11}$$

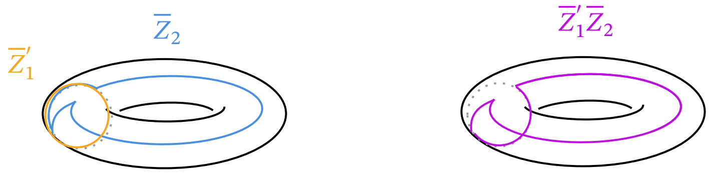
Let us consider a few cases. First, if we assume that we are working with a square lattice, i.e. $n_{\mathcal{F}} = n_{\mathcal{B}}$, we immediately see that the twist does not affect the minimum distance, i.e.:

$$d = n_{\mathcal{F}} = n_{\mathcal{B}} \tag{10.12}$$

Hence, if we want to improve the distance of a toric code by applying a twist, we must consider non-square lattices — specifically, lattices for which $n_{\mathcal{F}} \gg n_{\mathcal{B}}$. Restricting our attention to such lattices, and moreover assuming that $k < n_{\mathcal{F}} - k$, we see that the minimum distance of the code scales linearly with k:

$$d = n_{\mathcal{B}} + k \tag{10.13}$$

This derivation does not hold true, however, once we let go of the requirement that $k < \min\{n_{\mathcal{F}}, n_{\mathcal{B}}\} = n_{\mathcal{B}}$. For instance, if we assume that $n_{\mathcal{B}} < k < n_{\mathcal{F}}$, we see that the loops corresponding to $\overline{Z}_1\overline{Z}_2$ of the following form:



are of lower weight than $\overline{Z}_1$-loops. When calculating the minimum distance of the code, therefore, we have to take into account the length of such loops as well. Taking even larger twists complicates this even further, as products of $\overline{Z}_1$ and different loops corresponding to a $\overline{Z}_2$-error can then still be of lower weight than the loops corresponding to a $\overline{Z}_1$-error. As all of this *reduces* the minimum distance instead of increasing it, however, this is not a fruitful path to continue onto and therefore we do not pursue this analysis any further — from now on, we restrict our attention to values of k that are smaller than both $n_{\mathcal{F}}$ and $n_{\mathcal{B}}$. In this case, the distance of our code is given by:

$$d = \min\{n_{\mathcal{F}}, \min\{n_{\mathcal{B}} + k, n_{\mathcal{B}} + (n_{\mathcal{F}} - k)\}\} \tag{10.14}$$

We now note that there is a certain symmetry in the choice of twists, as twists $k = n_{\mathcal{F}}/2 \pm i$ (with $0 \leq i \leq n_{\mathcal{F}}/2$) yield the same result for the distance. Therefore, we can restrict our attention to twists $0 \leq k \leq n_{\mathcal{F}}/2$ without loss of generality.

In summary, requiring that $n_{\mathcal{F}} > n_{\mathcal{B}}$, and moreover requiring that $0 \leq k < \min\{n_{\mathcal{F}}/2, n_{\mathcal{B}}\}$, we find that the minimal distance is given by:

$$d = \min\{n_{\mathcal{F}}, n_{\mathcal{B}} + k\} \tag{10.15}$$

10.4.2. Numerical Verification

We now proceed with the numerical verification of these results. The Python scripts that were used for this purpose are built upon modules performing linear algebra over $\mathbb{F}_2$. These modules were used in [BVC+17] and were provided by the authors of this paper. We adapted their scripts for our own research purposes. All the scripts used can be found in Appendix B.

Let us first restrict our attention to a square lattice. The results of our numerical study can be found in Figure 10.10.

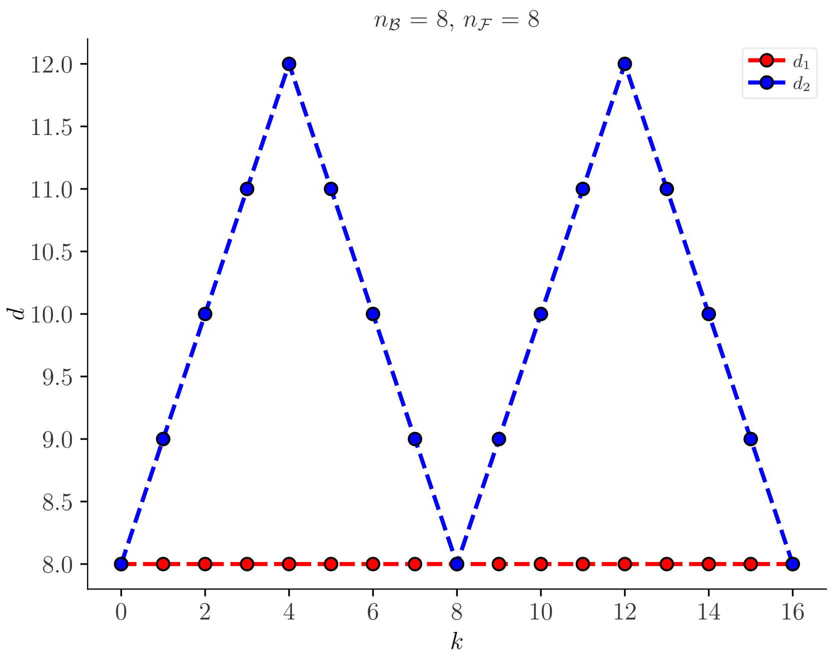


Figure 10.10: The weight of the minimum weight loops of the chain complex corresponding to the twisted toric code are given for an 8 by 8 lattice. d_1 (in red) corresponds to the logical $\overline{Z}_1$ (in red) in Figure 10.8, while d_2 (blue) corresponds to the logical $\overline{Z}_2$ (in blue) in Figure 10.8.

We indeed see that our analysis holds true: the loops corresponding to $\overline{Z}_1$ are unaffected by the twist, while the other loops become larger as on the domain $0 \leq k \leq n_{\mathcal{F}}/2$. By the symmetry of the choice of twists, we see that as $n_{\mathcal{F}}/2 \leq k < n_{\mathcal{F}}$, the correction for the twist can be performed in the opposite direction, and the distance starts decreasing, until k attains the value $n_{\mathcal{F}}$, which is mathematically equivalent to not performing a twist at all. The minimum distance of the code, nevertheless, remains $n_{\mathcal{F}}$.

Let us now restrict our attention to a non-square lattice, for example to a lattice for which $n_{\mathcal{F}} = 2 \cdot n_{\mathcal{B}}$. The results of our numerical study can now be found in Figure 10.11.

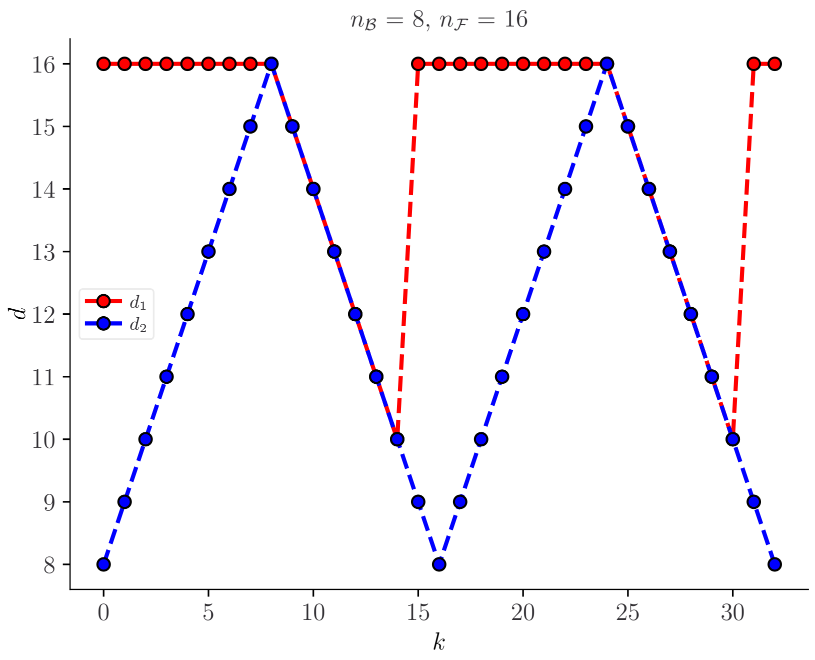


Figure 10.11: The weight of the minimum weight logicals of the twisted toric code on an 8 by 8 lattice are given. We see that the distance d_2 corresponds to the weight of the minimum weight loops corresponding to $\overline{Z}_2$ in Figure 10.8. The distance d_1, however, corresponds to the logical $\overline{Z}_1$ from Figure 10.8 for $k \leq 8$. For $8 < k \leq 16$, the weight of the loops corresponding to $\overline{Z}_1\overline{Z}_2$ becomes less than the weight of the loops corresponding to $\overline{Z}_1$. Therefore, for these values of k,d_1 reflects their weight instead of the weight of $\overline{Z}_1$.

If we restrict our attention to the domain $k < n_{\mathcal{B}}$, then these results are in line with our expectations, namely that the loops corresponding to $\overline{Z}_1$ possess the same weight, while the minimum weight of those corresponding to $\overline{Z}_2$ increases with k. Increasing the twist on this domain thus improves the minimum distance of the code. Going beyond this domain, however, we see that the minimum distance starts decreasing, as the weight of the loops corresponding to $\overline{Z}_1\overline{Z}_2$ becomes less than the weight of the loops corresponding to $\overline{Z}_1$. These numerical results therefore confirm that it suffices to restrict one's attention to relatively small values of k if one wants to maximise the minimum distance of the code.

10.5. The Logical Error Rate of the Twisted Toric Code

Having determined the rate and the distance of the twisted toric code, we now turn to the question of characterising the scaling of the logical error rate of the code as a function of the twist.

A lot of similar research has already been conducted for the class of surface codes (see e.g. [DKLP02]). In [BBKM19], a comparison was made between the performance of the toric code and a rotated toric code. In this paper, the researchers derived analytical approximations for the scaling of the logical error rates of each of these codes, and benchmarked these against numerical analyses. We can make use of their results for the toric code, and extend their analysis to the twisted toric code as well.

Before that, we note that an efficient decoder based on the minimum-weight perfect matching (MWPM) algorithm is already available for a wide class of codes, including the toric code. An efficient implementation of this decoder is given by the PyMatching Python package from [Hig22]. We can apply this decoder to the twisted toric code as well, and as such, we can run simulations for these codes and compare their performance. We will therefore start off our discussion with a numerical study of these codes. Afterwards, we will explain these numerical results in the low p regime by deriving analytical expressions for their logical error rates by generalising the results from [BBKM19].

10.5.1. Numerical Simulations

In order to perform a numerical study of the scaling of the logical error rate of the twisted and untwisted toric code, we first need to establish our assumptions. We will proceed by assuming that the physical errors, i.e. the errors on each qubit, are i.i.d. Bernoulli random variables with parameter p, which is commonly referred to as the physical error rate.

Furthermore, as the plaquette operators of the toric code are the vertex operators on its dual lattice, there is no qualitative difference between the Z and X-types of errors, and therefore, it suffices to consider just one of them. We thus restrict our attention to Z-errors.

We have written software to numerically estimate the logical error rate of the toric code for an arbitrary twist by calculating the fibre bundle product of a repetition code with parameter $n_{\mathcal{F}}$ as the fibre code and a repetition code with parameter $n_{\mathcal{B}}$ as the base code, along with the twist discussed earlier. The reader interested in the precise details of our implementation is referred to Appendix B, where all of our code can be found.

Having established the framework, let us now proceed with performing an analysis of our codes in the case in which they are defined on a square lattice. In order to compare their performance, we consider a small lattice size $n_{\mathcal{B}} = n_{\mathcal{F}}$, as this ensures that the effects due to the twist are relatively large.

We therefore first compare the performance of the untwisted toric code on a 4 by 4 lattice and the same code but with a twist of $k = 2$. The results of this analysis can be found in Figure 10.12 and in 10.13.

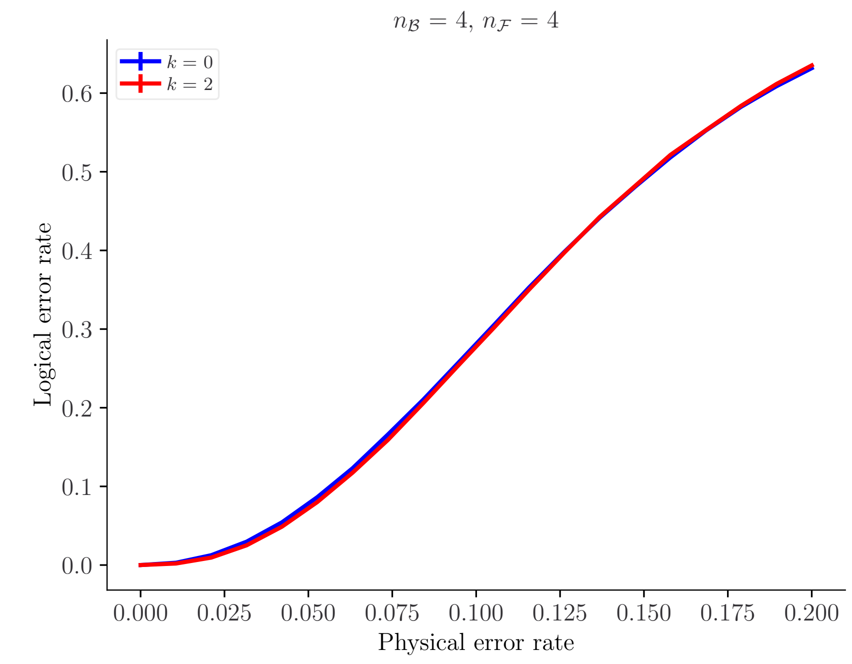


Figure 10.12: The logical error rate plotted as a function of the physical error rate for the twisted toric code defined on a 4 by 4 lattice for twists $k = 0$ and $k = 2$. For each of these values of k, the minimum distance of the code is 4. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

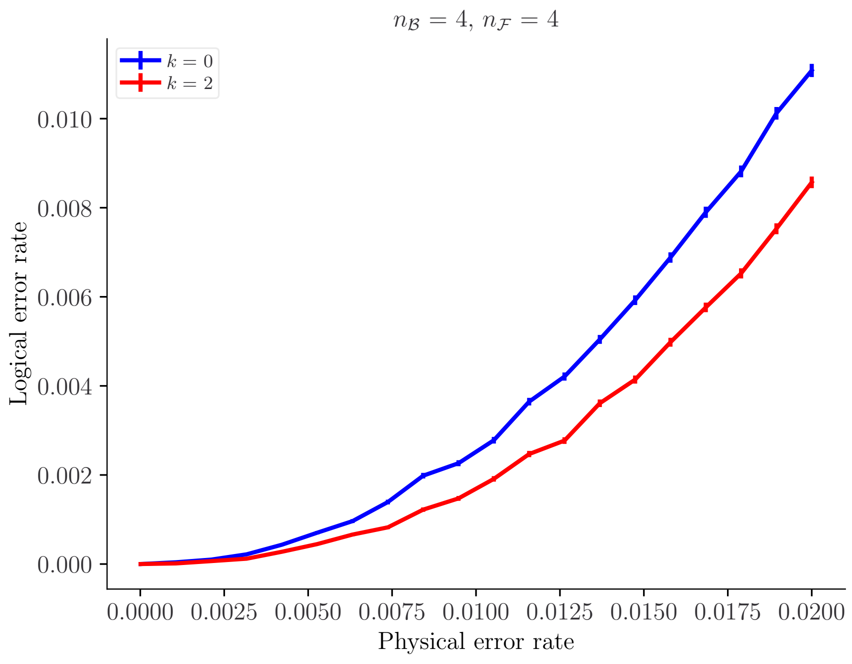


Figure 10.13: The results from Figure 10.12 considered on a smaller domain. The logical error rate is plotted as a function of the physical error rate for the twisted toric code defined on a 4 by 4 lattice for twists $k = 0$ and $k = 2$. For each of these values of k, minimum distance of the code is 4. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

These numerical simulations seem to imply that for low values of p (at least for $p < 0.02$), the logical error rate of a twisted toric code is lower than that of the untwisted toric code.

An interesting observation to make at this point is that the weight of the two independent minimum weight errors $\overline{Z}_1$ and $\overline{Z}_2$ when applying a twist of $k = 2$ is 4 and 6, respectively. This is also the case for the regular, untwisted toric code defined on a 4 by 6 lattice. Hence, we can compare how these two codes differ. The results of these simulations can be found in Figure 10.14.

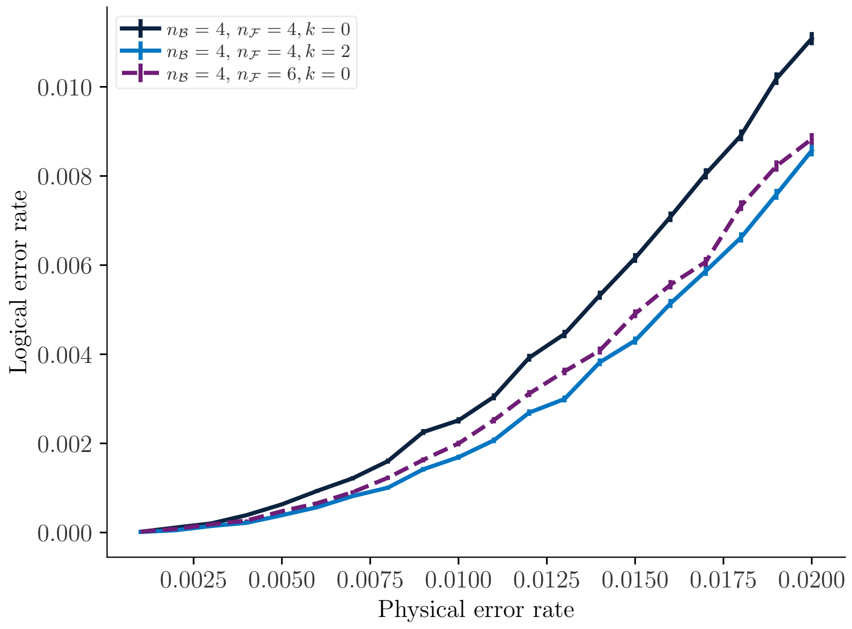


Figure 10.14: The logical error rate as a function of the physical error rate for the twisted toric code defined on a 4 by 4 lattice for twists $k = 0$ and $k = 2$, as well as for the toric code defined on a 4 by 6 lattice. In each of these cases, the minimum distance of the code is 4. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

It appears that the logical error rate of the twisted code and the toric code on a 4 by 6 lattice scale similarly in the low p regime. We will try to explain these results in the next section.

Let us now move on to codes defined on a non-square lattice, for example a lattice where $n_{\mathcal{F}} = 2 \cdot n_{\mathcal{B}}$. Here, too, we consider relatively small values of $n_{\mathcal{B}}$, such that the effects due to the twist are visible. The results of these simulations are presented in Figure 10.15 and Figure 10.16.

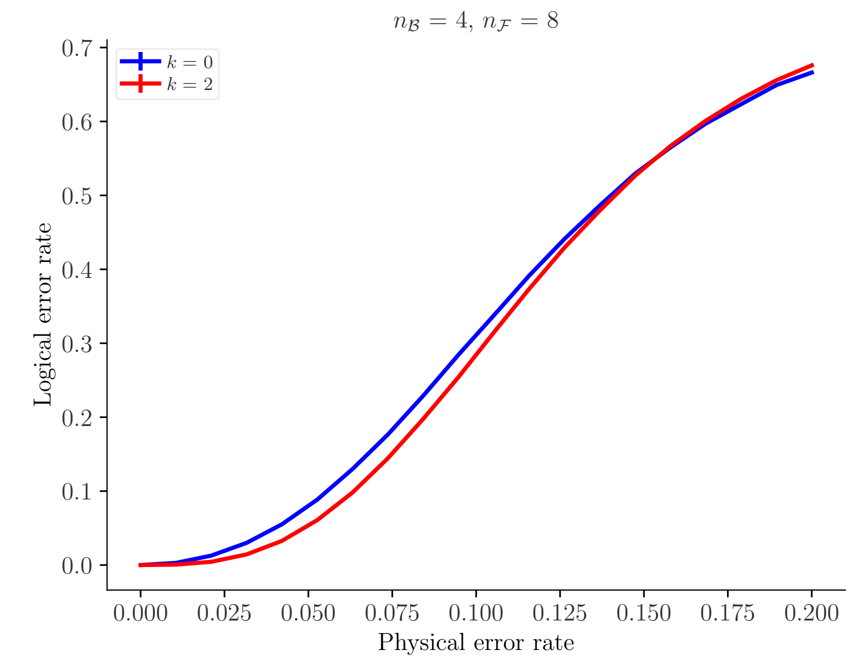


Figure 10.15: The logical error rate is plotted as a function of the physical error rate for the twisted toric code defined on a 4 by 8 lattice for twists $k = 0$ (with distance $d = 4$) and $k = 2$ (with distance $d = 6$). These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

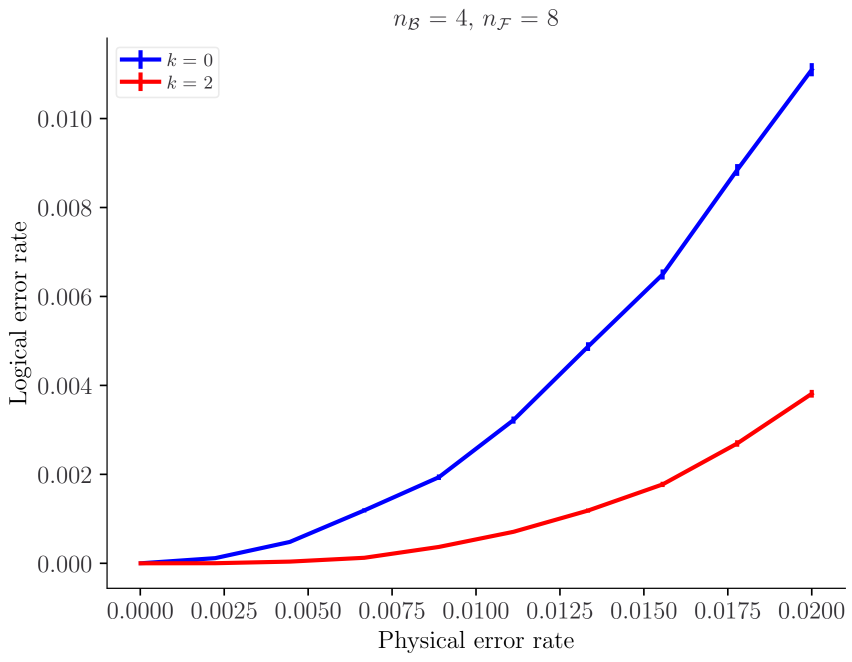


Figure 10.16: The results from Figure 10.15 considered on a smaller domain. The logical error rate is plotted as a function of the physical error rate for the twisted toric code defined on a 4 by 8 lattice for twists $k = 0$ (with distance $d = 4$) and $k = 2$ (with distance $d = 6$). These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

We will explain this behaviour using analytic methods in the next section.

10.5.2. The Logical Error Rate in The Low p Limit: The Untwisted Case

Given some error model, one can — in principle — calculate the logical error rate of the toric code. We proceed to do so, following the work of [BBKM19]: given a toric code defined on an $n_{\mathcal{F}}$ by $n_{\mathcal{B}}$ lattice, one can identify each string of Z-errors by their action on the qubits in the lattice. Such a string E has a weight, which we denote by $w(E)$. The logical error rate, now, is given by the sum of the probabilities of occurrence of each error string for which the correction produced is incorrect. Assuming that the Z-errors on the qubits can be modelled as i.i.d. Bernoulli variables, the probability of occurrence for an error string E is:

$$P(E) = (1-p)^{n-w(E)} p^{w(E)} \tag{10.16}$$

where $n := 2 \cdot n_{\mathcal{B}} \cdot n_{\mathcal{F}}$ is the total number of qubits on the lattice. The total logical error rate, $\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, p)$, is now given by:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, p) = \sum_{w \geq 0}^{n} \Omega(w)(1-p)^{n-w} p^{w} = \sum_{w \geq 0}^{n} (1-p)^{n} \Omega(w) \left(\frac{p}{1-p}\right)^{w} \tag{10.17}$$

where $\Omega(w)$ denotes the number of error strings with weight w that cannot be corrected. We note that $\Omega(w) = 0$ for $w < \frac{1}{2} \min\{n_{\mathcal{F}}, n_{\mathcal{B}}\}$ when using a minimum-weight decoder, as all strings with weight less than half the distance of our code can be corrected.

Before proceeding with our actual analysis, we note that one can interpret this framework in the language of statistical mechanics. To this end, one can apply the following substitutions:

$$\beta := -\log\left(\frac{p}{1-p}\right), \quad S(w) := \log(\Omega(w)) \tag{10.18}$$

By interpreting β as an inverse temperature and S as an entropy, we see that we can rewrite the logical error rate as:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, p) = (1-p)^{n} \sum_{w \geq 0}^{n} e^{-\beta F(w)} \tag{10.19}$$

where $F(w) := w - S(w)/\beta$ can be interpreted as a free energy. The goal of our analysis will be to derive an expression for the contribution of the minimum weight w_{min} error strings alone, we will approximate the logical error rate as follows:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, p) \simeq (1-p)^{n} e^{-\beta(w_{\text{min}} - S(w_{\text{min}})/\beta)} \tag{10.20}$$

Hence, we see that we can distinguish between the *entropic* contributions to the logical error rate, and *energetic* contributions. This point of view will be useful when interpreting our results.

Let us now start with our analysis. For the purpose of this analysis, we shift to the description of the toric code as the hypergraph product of two repetition codes. From this point of view, the minimum weight logicals correspond to vertical and horizontal loops over the lattice, as can be seen in Figure 10.17.

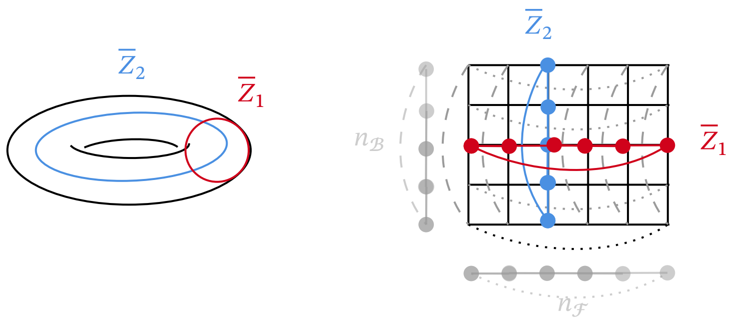


Figure 10.17: On the left, the two loops corresponding to minimum weight errors $\overline{Z}_1$ (red) and $\overline{Z}_2$ (blue). On the right, the toric code when interpreted as the hypergraph product of two repetition codes. The counterpart of the loops are shown on the lattice, where the lattice point on which they coincide is shifted slightly for illustrative purposes.

Let us first direct our attention to the square lattice, that is, let us assume for now that $n_{\mathcal{F}} = n_{\mathcal{B}}$. The loops of minimum weight are thus loops of weight $d/2 = n_{\mathcal{B}}/2$. We now claim that the number of minimum weight loops can be determined analytically using the following equation (as was proved in [BBKM19]):

$$\Omega\left(n_{\mathcal{B}}/2\right) = \frac{1}{2} \cdot \left(n_{\mathcal{B}} \cdot \binom{n_{\mathcal{F}}}{n_{\mathcal{F}}/2} + n_{\mathcal{F}} \cdot \binom{n_{\mathcal{B}}}{n_{\mathcal{B}}/2}\right) = \frac{2d}{2} \cdot \binom{d}{d/2} \tag{10.21}$$

To see why this is the case, let us decompose this equation. Firstly, we note that for each loop corresponding to an $\overline{Z}_2$-error (equivalently, for each vertical loop on the lattice as defined in Figure 10.17), there are

$$\binom{n_{\mathcal{B}}}{n_{\mathcal{B}}/2} \tag{10.22}$$

unique chains of errors of weight $n_{\mathcal{B}}/2$. An example of a few of such error chains can be seen in Figure 10.18.

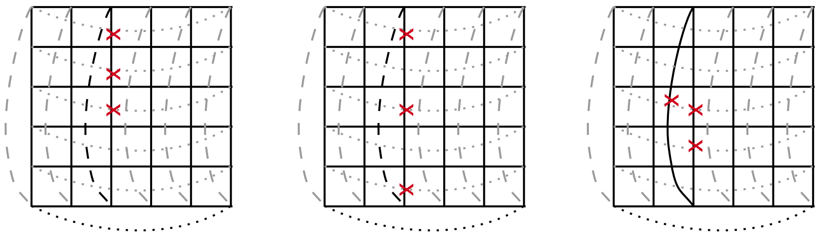


Figure 10.18: Three different minimum weight errors on the physical qubits (represented here as red crosses) are displayed whose correction can induce the logical error $\overline{Z}_2$ displayed in Figure 10.17.

Although most of these strings of errors produce a different syndrome, for each of these error strings, however, there is another error string produces the same syndrome. An example of this can be found in Figure 10.19.

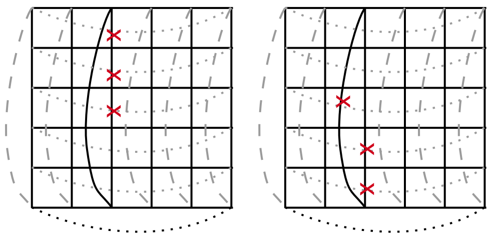


Figure 10.19: An example of two error strings (given by the red crosses) that result in the same syndrome.

The correction, however, for that syndrome, is always the same, hence, one of these strings gets corrected, while the correction extends the other string to a logical Z-error, as is illustrated in Figure 10.20.

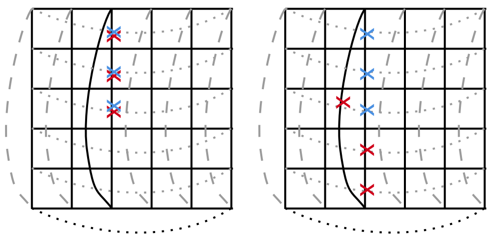


Figure 10.20: The two error strings from Figure 10.19, to which the same correction (the blue crosses) is applied. On the left, we see that the error is corrected, while on the right, we see that the correction actually applies a logical error.

Hence, only *half* of these error strings cause a logical error (explaining the factor 1/2). Finally, there are $n_{\mathcal{F}}$ different minimum weight loops corresponding to the $\overline{Z}_2$-logical operator (these are the different vertical lines on the lattice in Figure 10.17), which is accounted for by the factor of $n_{\mathcal{F}}$. The same analysis holds for $\overline{Z}_1$-errors (which correspond to the horizontal lines on the lattice), only now, there are $n_{\mathcal{B}}$ many of these. Hence, we conclude that the number of minimum weight errors is indeed

$$\Omega\left(n_{\mathcal{B}}/2\right) = \frac{2d}{2} \cdot \binom{d}{d/2} \tag{10.23}$$

Having presented the analysis for the case of a square lattice, we can now extend this analysis to the case where $n_{\mathcal{F}} > n_{\mathcal{B}}$. We note that this is fairly straightforward: we can simply split $\Omega(w)$ from our previous analysis up into $\Omega_{\mathcal{B}}(w)$, which counts the number of error strings causing a $\overline{Z}_2$-error, and $\Omega_{\mathcal{F}}(w)$, which counts the number of strings causing a $\overline{Z}_1$-error[1]. We then find the following expression for the number of minimum weight errors:

$$\Omega_{\mathcal{B}}\left(\frac{n_{\mathcal{B}}}{2}\right) = \frac{n_{\mathcal{F}}}{2} \cdot \binom{n_{\mathcal{B}}}{d/2}, \quad \Omega_{\mathcal{F}}\left(\frac{n_{\mathcal{B}}}{2}\right) = 0 \tag{10.24}$$

As we are interested in the behaviour of the logical error rate in the low p limit, the dominant errors will be the low weight errors. Therefore, for small values of p, we can approximate the logical error rate by only counting the contribution of the minimum weight logical errors. As such, we see that the logical error rate can be approximated as follows:

$$\overline{P}(n_{\mathcal{B}}, n_{\mathcal{F}}, p) \simeq (1-p)^n \cdot \left(\Omega_{\mathcal{B}}\left(\frac{d}{2}\right) + \Omega_{\mathcal{F}}\left(\frac{d}{2}\right)\right)\left(\frac{p}{1-p}\right)^{\frac{d}{2}} \tag{10.25}$$

We can now compare these approximations with our numerical results. We do so in Figure 10.21.

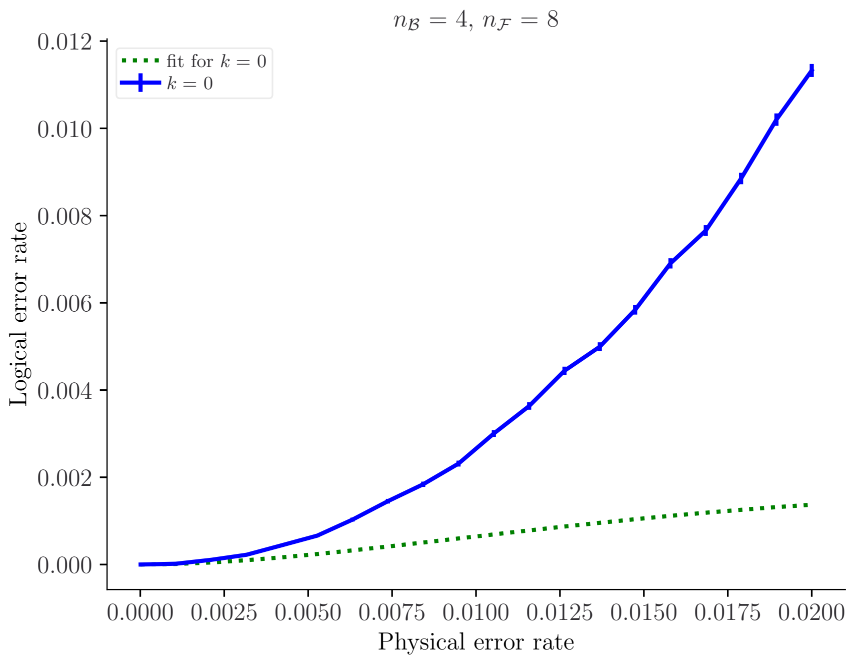


Figure 10.21: The logical error rate as a function of the physical error rate for the regular, untwisted toric code defined on a 4 by 8 lattice, along with the analytically derived approximation for the logical error rate derived above. The minimum distance of this code is 4. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

One can see that already for quite low values of p ($p \simeq 0.5\%$), this approximation becomes quite crude, which suggests that for the chosen parameters, the higher weight error strings start becoming dominant

[1]We note that we do not have to take into account $\overline{Z}_1\overline{Z}_2$-errors, as these cannot be of minimum weight.

for low values of p. We expect the minimum weight error strings' contribution to the logical error rate to become dominant as $p \downarrow 0$. We check this against our simulations in Figure 10.22.

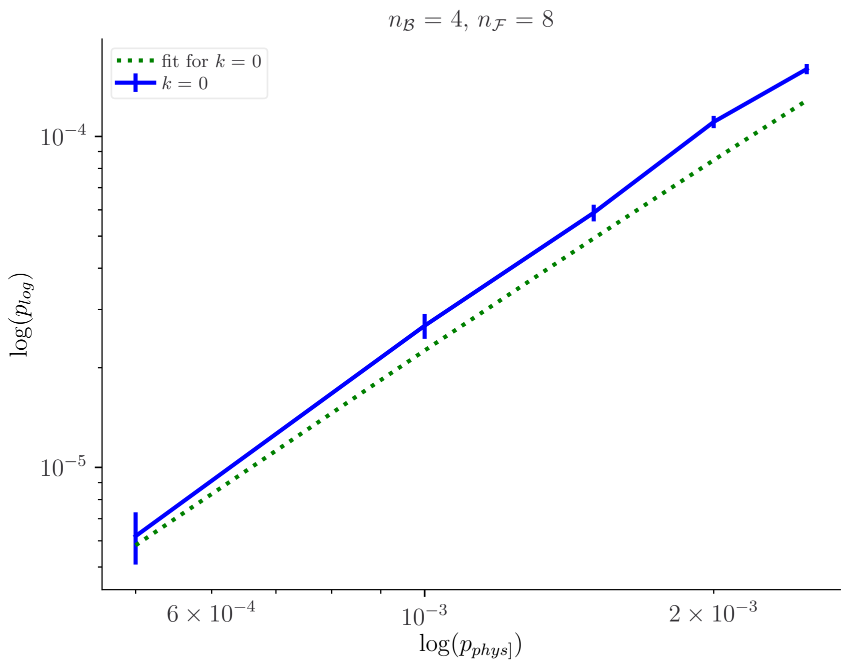


Figure 10.22: A logarithmic plot of the results from Figure 10.21. The logical error rate is plotted as a function of the physical error rate for the regular, untwisted toric code defined on a 4 by 8 lattice, along with the analytically derived approximation for the logical error rate derived above. The minimum distance of this code is 4. These simulations were performed by determining the decoding success rate for a total of 5,000,000 randomly drawn samples at each of the physical error rates.

These results indeed seem to verify the expected asymptotic behaviour of the logical error rate. We note, however, that to ensure that this is the case, one would need to take an even larger sample size. Yet, to get these accuracy results, we already had to increase the sampling rate by a factor of 10 to $5,000,000$ samples per value of the physical error rate. Increasing the sampling rate even further, unfortunately, lies beyond the reach of our current computational resources.

10.5.3. The Logical Error Rate in The Low p Limit: The Twisted Case

Let us now generalise the calculations of the logical error rate to the twisted toric code. To this end, let us again consider a $n_{\mathcal{F}}$ by $n_{\mathcal{B}}$ lattice, and let us apply the twist k introduced above. The logical error rate is now still given by the following expression:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, k, p) = \sum_{w \geq 0}^{n} (1-p)^n \left(\Omega_{\mathcal{B}}(w,k) + \Omega_{\mathcal{F}}(w,k)\right) \left(\frac{p}{1-p}\right)^w \tag{10.26}$$

We note that the loops on the torus can now again be represented by loops on the (twisted) lattice, as is displayed in Figure 10.23.

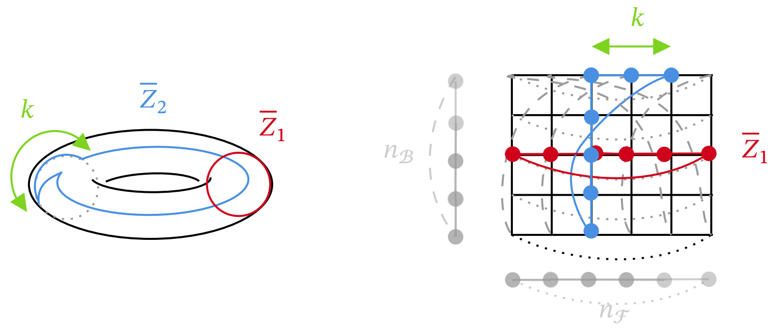


Figure 10.23: On the left, the two loops corresponding to minimum weight errors $\overline{Z}_1$ (red) and $\overline{Z}_2$ (blue) for the twisted toric code. On the right, the same twisted toric code when interpreted as the fibre bundle product of two repetition codes. The counterpart of the loops are shown on the lattice, where the lattice point on which they coincide is shifted slightly for illustrative purposes.

We immediately see that our previous analysis (for the untwisted toric code) is still valid when we want to calculate the number of horizontal ($\overline{Z}_1$) errors, as these loops are exactly the same loops as one would have in the untwisted case! This enables us to explain why for the square lattice, we see that the effect of a twist k on the scaling of the logical error rate is similar to the scaling of the logical error rate for the regular, untwisted toric code on a lattice with sides $n_{\mathcal{B}}$ and $n_{\mathcal{F}} + k$: in the low p regime, the dominant logical errors are those of minimum weight, and hence correspond to $\overline{Z}_1$, which remain unaltered by our twist. We thus conclude that, for a square lattice and low values of p, the effects of performing a twist are similar to increasing one side of the lattice by the twist, which is in line with our numerical results, as presented in Figure 10.14.

The number of error strings corresponding to a $\overline{Z}_2$-error, however, becomes substantially more complex to calculate. To determine these errors, we first recall that the loops corresponding to such errors are now of size $n_{\mathcal{B}} + k$. Yet, here already do we depart from our analysis of the untwisted situation, as all the possible minimum weight logical operators are not simply the vertical lines on the lattice, as is illustrated in Figure 10.24.

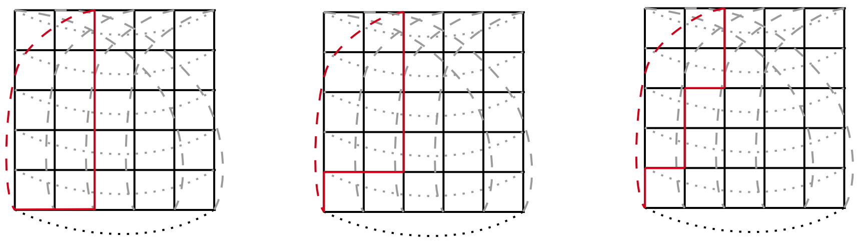


Figure 10.24: While in the untwisted case the minimum weight logical operators corresponded to vertical lines on the lattice, the minimum weight logical operators (in red) for the twisted code are much more diverse.

We want to count the number of different minimum weight error strings. To this end, we first start with the observation that some minimum weight error strings are perfectly correctable — see Figure 10.25 for two examples.

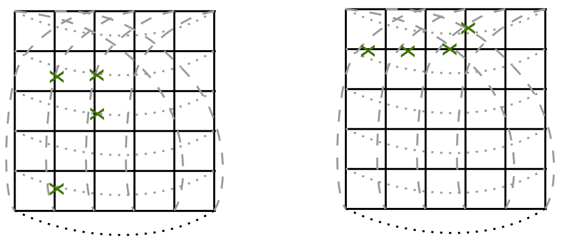


Figure 10.25: Two examples of minimum weight error strings on the twisted toric code that can be corrected by the PyMatching decoder.

What distinguishes uncorrectable from correctable minimum weight strings of errors, is that for the former class of strings, one can always construct a minimum weight loop encoding a $\overline{Z}_2$-logical that runs through all the edges of the qubits onto which the error acts. Hence, we can exploit the structure of these minimum weight logical operators to understand the possible minimum weight error strings.

To this end, let us start by investigating how one can construct specific minimum weight loops. We first fix a twisted edge of the lattice. We then see that all the possible minimum weight loops going through the lattice are bound to some sublattice, as is illustrated in Figure 10.26.

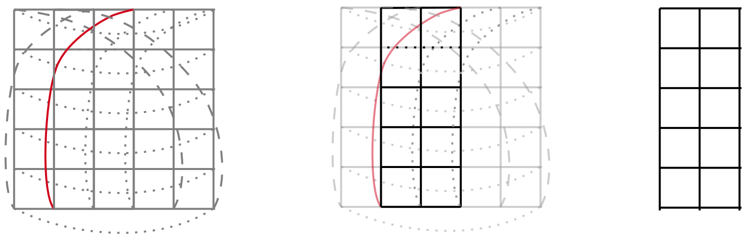


Figure 10.26: Choosing a twisted edge (in red), we see that the every minimum weight loop that runs through this edge is confined to a square lattice, which is shown in black in the second frame.

As such, we see that the problem of constructing a minimum weight loop that acts on a twisted edge is equivalent to the problem of choosing a path on a sublattice that starts in its upper right corner and ends in its bottom left corner by moving in only two directions: downwards and leftwards. This can be seen in Figure 10.27.

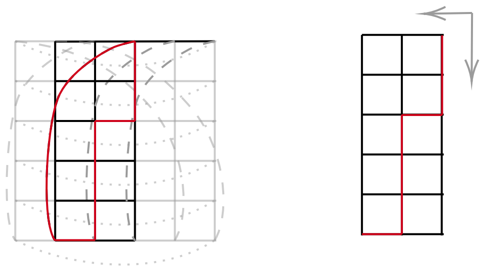


Figure 10.27: An example of a minimum weight loop going through the twisted edge (in red) and how the construction of this loop can be interpreted as a path between two ends of a lattice by only going downwards and leftward.

This has immediate implications for our counting problem, as this implies that one cannot just arbitrarily place errors on the lattice to obtain an uncorrectable error, but rather, that any relevant minimum weight error string can always be said to have a top right edge — this is illustrated in Figure 10.28.

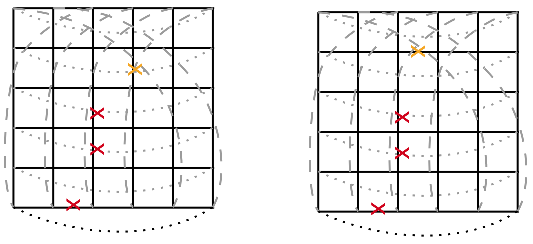


Figure 10.28: Two examples of minimum weight error strings, with the top right qubit given in orange, and the other errors in red.

Therefore, we can simplify our counting problem as follows. Let us focus on some specific section of the lattice, which is illustrated in Figure 10.29.

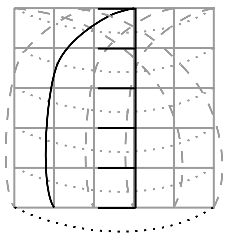


Figure 10.29: The section of the lattice (in black) that we will use in our counting problem. We adhere to the convention that the twisted edge is the topmost edge of this section.

For each edge on this section of the lattice, we can count the number of possible minimum weight error strings that lie on a minimum weight loop *and whose top right qubit is on this edge*. The sum of these numbers yields the number of possible minimum weight error strings whose top right qubit lies on this section. Let us denote it by $N(n_{\mathcal{B}}, k)$.

But now, we see that we can tile the whole lattice using $n_{\mathcal{F}}$ of these sections, and that, moreover, *all the error strings whose top right qubit lies on one of these sections are not counted on other sections*. But given that each error string has a top right qubit, we see that we can simply calculate the total number of possible minimum weight error strings that lie on a minimum weight loop. Noting that only half of these will lead to a logical error after error correction, we find that the total number of unique minimum weight errors that cannot be corrected is given by:

$$\Omega_{\mathcal{B}}\left(\frac{n_{\mathcal{B}+k}}{2}, k\right) = \frac{1}{2} n_{\mathcal{F}} \cdot N\left(n_{\mathcal{B}}, k\right) \tag{10.27}$$

We therefore move on to the problem of calculating $N\left(n_{\mathcal{B}}, k\right)$. We proceed inductively. First, note that by choosing some top right qubit, we are free to choose the next qubit within some sublattice, because of the very fact that this error string must lie on a minimum weight loop. This is illustrated in Figure 10.30.

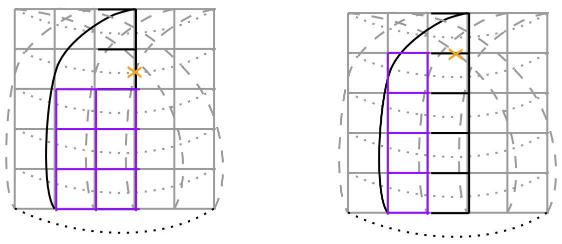


Figure 10.30: Two examples of how choosing a top right qubit (orange) allows one to choose the next qubit within a sublattice (in purple).

Suppose now that we have chosen the first two qubits. The third qubit can then, again, be chosen from some (smaller) sublattice, as can be seen in Figure 10.31

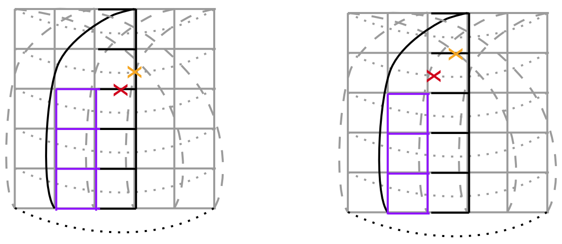


Figure 10.31: Two examples of how choosing a top right qubit (orange) and the first qubit (red) allows one to choose the next qubit within a smaller sublattice (in purple).

Hence, we see that there is a recurrence relation at play in our problem. Let us try to establish this relation.

To this end, consider an L_1 by L_2 lattice, and let us denote the number of ways in which we can pick m edges that lie on this lattice such that we can draw a path from the upper right corner to the bottom left corner by only going downwards or leftwards by $A(L_1, L_2, m)$. We claim that we have the following recurrence relationship for all $m \geq 2$:

$$A(L_1, L_2, m) = 2 \sum_{l_1=1}^{L_1-1} \sum_{l_2=1}^{L_2-1} A(l_1, l_2, m-1) + \sum_{l_2=1}^{L_2-1} A(L_1, l_2, m-1) + \sum_{l_1=1}^{L_1-1} A(l_1, L_2, m-1) \tag{10.28}$$

and that, furthermore:

$$A(L_1, L_2, 1) = 2L_1L_2 - L_1 - L_2 \tag{10.29}$$

We show that the equation for $A(L_1, L_2, m)$ holds by explaining each of the three terms. To this end, let us consider an L_1 by L_2 lattice, and let us distinguish three regions where we can choose the rightmost edge, that we will specify in Figure 10.32.

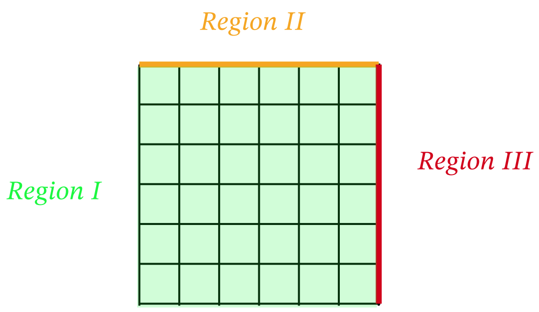


Figure 10.32: We distinguish three different regions on our lattice: Region I, corresponding to its interior, Region II, corresponding to its top boundary, and Region III, corresponding to its right boundary.

We argue that Region II corresponds to the second term in the summand, i.e. to the term

$$\sum_{l_2=1}^{L_2-1} A(L_1, l_2, m-1) \tag{10.30}$$

Indeed, this is the case, as choosing the l_1 th edge (we start counting from the right) in this region returns us to the problem of having to choose one less edge in a L_1 by l_2 sublattice, as is illustrated in Figure 10.33:

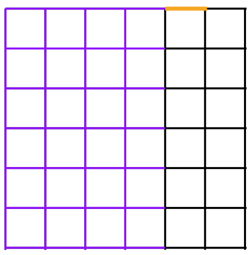


Figure 10.33: Choosing the orange edge, we see that the next edge can be chosen in the purple sublattice.

The fact that the third term corresponds to Region III follows from an exactly analogous line of reasoning.

Lastly, we show that Region I corresponds to the first term, i.e. to the term

$$2 \sum_{l_1=1}^{L_1-1} \sum_{l_2=1}^{L_2-1} A(l_1, l_2, m-1) \tag{10.31}$$

To see why this is the case, we first note that we can form pairs of edges in Region I, each of which yields the same sublattice to select the next edge from, as is illustrated in Figure 10.34.

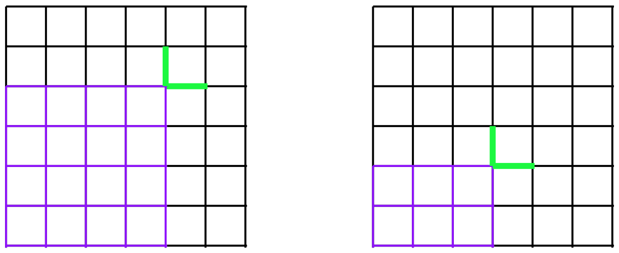


Figure 10.34: Two examples of pairs of edges (in green) that allow one to choose the next edge in the same sublattice (in purple).

Therefore, we see that for each vertex of the lattice in Region I, there are *two* edges that allow us to select the next qubit from the same sublattice. This explains the term 2 in front of the expression.

Moreover, we see that what defines these sublattices is their top right point, which is also the vertex in which the two edges that form a pair coincide:

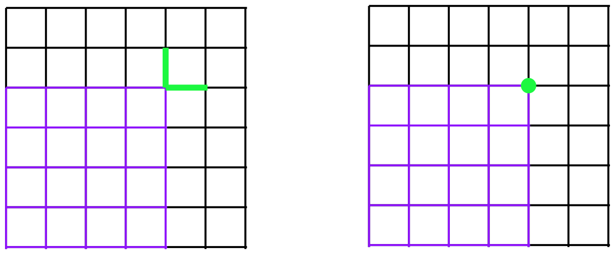


Figure 10.35: The vertex in which the pairs of edges (green) coincide is also the top right corner of the sublattice (in purple) within which one can choose the next qubit

Hence, by indexing the vertices on the lattice, where we take the bottom left vertex to be $(1,1)$, we see that for each vertex (l_1, l_2) where $1 \leq l_i \leq L_i - 1$, there are two edges that allow us to choose the next edge in a l_1 by l_2 sublattice, which explains the rest of the equation.

Finally, we have to prove the expression for $A(L_1, L_2, 1)$. But this is just the number of edges on an L_1 by L_2 lattice, which is well-known to satisfy the expression given. Hence, we are done.

We now return to our original problem of calculating $N(n_{\mathcal{F}}, k)$. We claim that the following holds:

$$N(n_{\mathcal{B}}, k) = \sum_{L=1}^{n_{\mathcal{B}}} A\left(L, k+1, (n_B + k)/2 - 1\right) + \sum_{L'=1}^{n_{\mathcal{B}}} A\left(L', k, (n_B + k)/2 - 1\right) \tag{10.32}$$

To understand this equation, we turn back to the sections of the lattice on which we choose the top rightmost edge. On these sections, the contribution of the vertical edges is given by the first sum in our proposed equation, whereas the contribution of the horizontal edges is given by the second sum.

We argue why this is the case. Firstly, note that by selecting the topmost possible horizontal edge as the top right qubit, one can choose the next edge in an $n_{\mathcal{B}}$ by k-sized lattice. Similarly, choosing the topmost possible vertical edge (which is the twisted edge) allows one to choose the next edge in an $n_{\mathcal{B}}$ by $k+1$ sized lattice. Choosing lower possible horizontal edges allows one to choose the next qubit in an L by k-sized lattice for $1 \leq L \leq n_{\mathcal{B}}$, and, similarly, choosing a lower possible vertical edge as the first qubit allows one to choose the next qubit from an L by $k+1$-sized lattice. Two examples of this are given in Figure 10.36.

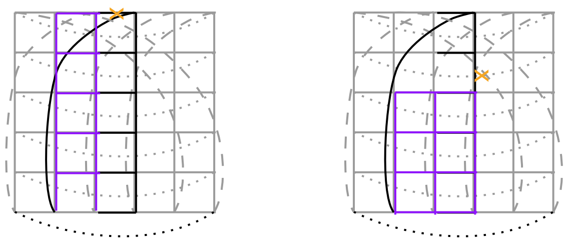


Figure 10.36: Choosing the top right qubit (in orange) to lie on a horizontal or a vertical edge determines whether the sublattice from which one can choose the next qubit will have width k or $k+1$ (here: $k=2$).

Lastly, we note that as the top right qubit is already fixed in this procedure, one can choose one less qubit than the weight of the error string, which is $(n_{\mathcal{B}}+k)/2$, and as such, the third argument of the functions is $(n_{\mathcal{B}}+k)/2-1$.

We have thus found a way to calculate the exact number of minimum weight error strings that lie on a minimum weight loop. Therefore, if we approximate the logical error rate of the twisted toric code in the low p region by only counting the contribution of such errors, we find that we can approximate the logical error rate as follows:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, k, p) \simeq (1-p)^n \cdot \Omega_{\mathcal{B}}\left(\frac{n_{\mathcal{B}}+k}{2}, k\right) \cdot \left(\frac{p}{1-p}\right)^{\frac{n_{\mathcal{B}}+k}{2}} \tag{10.33}$$

$$= \frac{n_{\mathcal{F}} \cdot (1-p)^n}{2} \cdot N\left(n_{\mathcal{B}}, k\right) \cdot \left(\frac{p}{1-p}\right)^{\frac{n_{\mathcal{B}}+k}{2}} \tag{10.34}$$

$$= \frac{n_{\mathcal{F}} \cdot (1-p)^n}{2} \cdot \sum_{L=1}^{n_{\mathcal{B}}} \left(A\left(L, k+1, \frac{n_{\mathcal{B}}+k}{2}-1\right) + A\left(L, k, \frac{n_{\mathcal{B}}+k}{2}-1\right)\right) \cdot \left(\frac{p}{1-p}\right)^{\frac{n_{\mathcal{B}}+k}{2}} \tag{10.35}$$

Having finally arrived at this equation, we can now compare it with our numerical approximations. We do so in Figure 10.37 below.

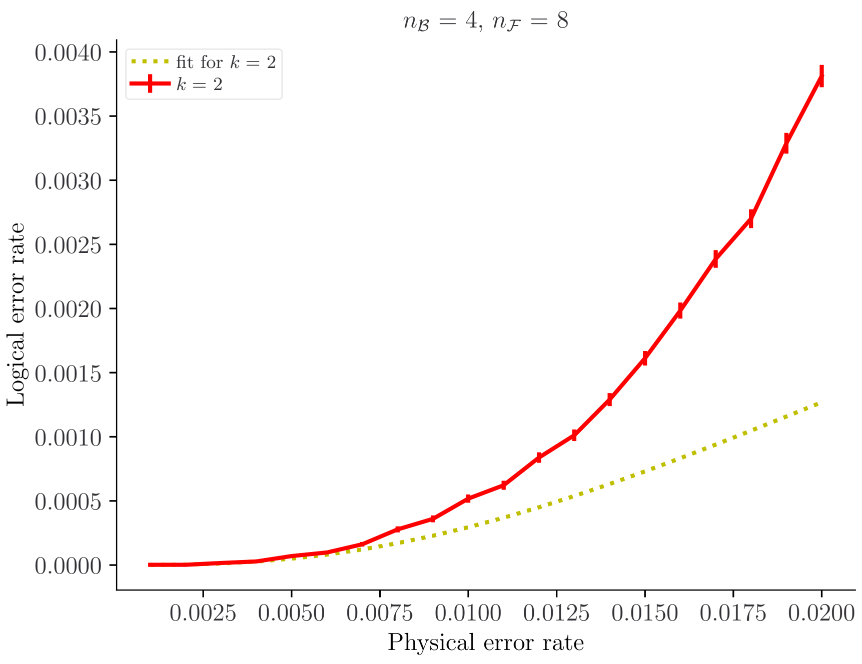


Figure 10.37: The logical error rate as a function of the physical error rate for the twisted toric code defined on a 4 by 8 lattice with a twist of $k = 2$, along with the analytically derived approximation for the logical error rate derived above. The minimum distance of this code is $d = 6$. These simulations were performed by determining the decoding success rate for a total of 500,000 randomly drawn samples at each of the physical error rates.

At first glance, the approximation seems to perform well for physical error rates $p < 0.5\%$, although it always underestimates the logical error rate (as one would expect). We expect this approximation to converge to the actual logical error rate as p converges to zero. To test this, we can zoom in on the domain $p < 0.5\%$. We plot these results on a logarithmic scale in Figure 10.38.

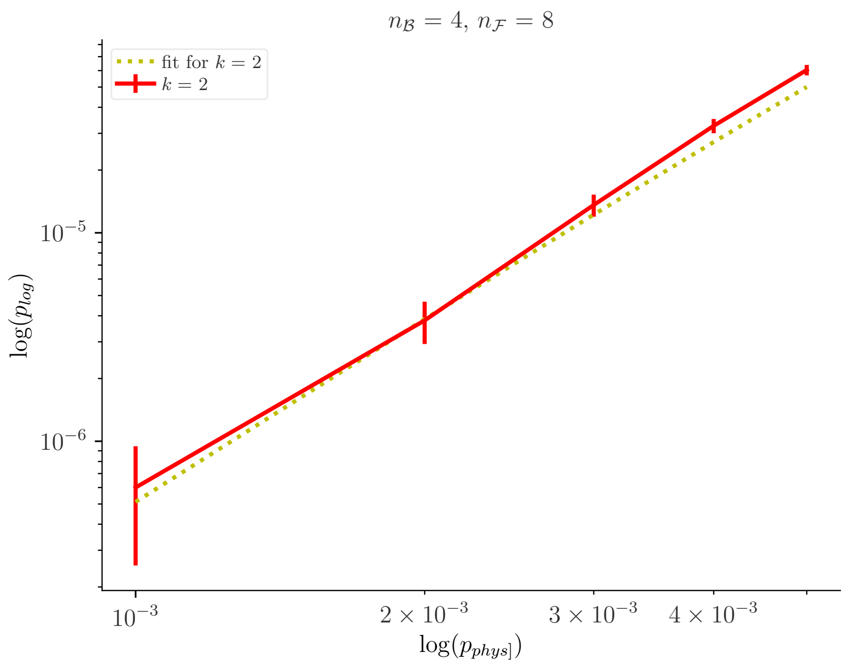


Figure 10.38: The logical error rate as a function of the physical error rate for the twisted toric code defined on a 4 by 8 lattice with a twist of $k = 2$, along with the analytically derived approximation for the logical error rate derived above plotted on a logarithmic scale. The minimum distance of this code is $d = 6$. These simulations were performed by determining the decoding success rate for a total of 5,000,000 randomly drawn samples at each of the physical error rates.

We see that our approximation seems to perform quite well in this regime, although we would need to perform more simulations at a higher sampling rate to confirm this. Unfortunately, however, this too lies beyond the possibilities with our current computational resources.

10.6. Reflection on the Analysis

Although we used our analytically derived approximations to explain the scaling of the twisted toric code for our specific choice of parameters ($n_{\mathcal{B}} = 4, n_{\mathcal{F}} = 8, k = 0, 2$), these approximations hold in full generality. We can thus use them to compare the relative asymptotic scaling of the logical error rate of the twisted toric code and the untwisted toric code on an arbitrary lattice. To this end, let us consider the relative scaling of the contributions of the minimum weight errors:

$$\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, k, p)/\overline{P}(n_{\mathcal{F}}, n_{\mathcal{B}}, 0, p) \simeq \frac{\Omega_{\mathcal{B}}\left(\frac{n_{\mathcal{B}}+k}{2}, k\right)}{\Omega_{\mathcal{B}}\left(\frac{n_{\mathcal{B}}}{2}, 0\right)} \cdot \left(\frac{p}{1-p}\right)^{k/2} = \mathcal{O}\left(\left(\frac{p}{1-p}\right)^{k/2}\right) \to 0 \tag{10.36}$$

Hence, we conclude that, although the entropic term is larger for the twisted toric code than for the untwisted one, this does not compensate for the energetic difference in the low p limit. That is to say, the twisted toric code on a non-square lattice *always* outperforms the untwisted toric code defined on the same lattice as $p \downarrow 0$.

However, we note that the fact that the entropic contribution of the minimum weight error strings is larger for the twisted toric code implies that there is some value for p for which the entropic difference become larger than the energetic difference, that is, for which the contribution to the logical error rate due to the minimum weight strings is actually larger for the twisted toric code than for the untwisted toric code. Yet, at or above this value for p, the contributions due to other, higher weight error strings can no longer be neglected, and hence, we cannot draw any conclusions in full generality based on our analysis alone. For future research, we suggest that this be explored by generalising the analytical derivations presented here such that they can be used to quantify the contributions of higher weight error strings to the logical error rate as well. We note that a first step in this direction would be to find an *explicit* expression for the number of minimum weight error strings that cannot be corrected by our decoder.

Lastly, we recall that the threshold value of the regular toric code, i.e. the lowest value of the physical error rate at which higher distance codes make more logical errors instead of fewer (see Figure 10.4), is well-known: in [DKLP02], an analytical approach to the question of determining this threshold is taken by interpreting the toric code in terms of a statistical mechanical model. In Figure 10.15, we see that our twisted toric code, too, seems to admit some threshold value. Moreover, given that we can also map the twisted toric code onto a statistical model, we speculate that the threshold can be determined by an argument along the lines of the argument presented in [DKLP02], and leave this, too, as a suggestion for future researchers to explore.

Discussion and Conclusion

Various product constructions for CSS codes have been defined and used in the search for a family of good qLDPC codes. In this thesis, we presented an overview of various homology theories, and used them to understand the different product code constructions in the literature and to analyse these. As our work was divided into three parts, let us consider these parts one by one to discuss and conclude our work and our findings.

In Part I, we provided an overview of the theory of singular homology and, more generally, of homology of chain complexes over vector spaces, and of three product code constructions that can be interpreted as the tensor product of two chain complexes: the hypergraph product from [TZ09], the EKZ product from [EKZ20], and their generalisation as presented in [ZP19]. In contrast to the original paper [TZ09], we determined an analytic expression for the number of encoded qubits k, the distance d and the weight w of the first two of these constructions without making any reference to their Tanner graph structure but rather in purely linear algebraic terms.

In Part II of this thesis, we introduced the theory of homology over ring modules. Next, we introduced the lifted product construction. While the authors took an algebraic approach to these constructions in their work [PK22b, PK22a], we took a homological approach to these codes instead by interpreting the construction as taking the tensor product of modules of a unital, associative and commutative $\mathbb{F}_2$-algebra, $\mathcal{R}$. While the authors were unable to find a general expression for the number of encoded qubits, k, of their construction, we derived a Künneth formula for the tensor product of these codes that allowed us to relate the homology of the product code to the $\mathcal{R}$-tensor product of the homology groups of the individual codes. Lastly, by restricting our attention to the case of a group algebra $\mathcal{R} := \mathbb{F}_2G$, we were able to relate the $\mathcal{R}$-tensor product of $\mathcal{R}$-modules to their $\mathbb{F}_2$-tensor product, namely as the G-coinvariants of the latter. As such, we were able to provide an expression for the number of encoded qubits in terms of the dimension of the codes that one would obtain by simply applying the regular hypergraph product, and the dimension of the subspaces of the tensor product space on which the G-action is non-trivial.

In Part III of the thesis, we first introduced the theory of covering spaces, of fibre bundles, of the homology with local coefficients, and lastly, of spectral sequences. Afterwards, we posed the question of how one could proceed with calculating the homology of fibre bundles. We gave an exposition of three techniques and performed calculations for a generalisation of the Möbius strip: by considering the tensor product complex of the base space and the fibre, but introducing a twist into the differential, where the twist is derived from the transition functions or, if the base admits a universal cover, from the action of the fundamental group of the base space on the fibre. Next, we saw how, using just the transition functions, one could use the Mayer-Vietoris sequence to calculate the homology. Lastly, we saw how one could use the Serre spectral sequence for this purpose as well.

From the discussion of the homology of fibre bundles, we could finally make the mathematical motivation of the fibre bundle products and balanced product explicit: by taking the base space to be a compact CW complex $\mathcal{B}$ with just 0 and 1-cells admitting a universal cover $\tilde{\mathcal{B}}$, the fibre bundle product of the cellular complex associated to this code with the cellular complex of some other space $\mathcal{F}$ could be interpreted as determining a cellular complex for the fibre bundle with fibre $\mathcal{F}$ and base space $\mathcal{B}$, where the twist φ represented the $\pi_1(\mathcal{B})$-action on the fibre one obtains in the associated bundle. As such, we see that this construction mirrors our first approach towards calculating the homology of the fibre bundle, which was based on introducing a twist in the differential of the tensor product complex of the base space and the fibre. In the same vein, we saw that we could interpret the balanced product of the complex $\tilde{B}$ and $\mathcal{F}$ as simply constructing the chain complex belonging to the total space of the principal $\pi_1(\mathcal{B})$-bundle associated to the fibre bundle. These considerations allowed us to do two things: firstly, they allowed us to determine the number of encoded qubits of the fibre bundle product of two codes in a novel manner, given that they arise as cellulations of the base and fibre space that we considered hitherto. Secondly, this allowed us to relate these two product code constructions, again given that the codes in consideration satisfy the properties outlined above. We note that this last result was (somewhat) too strict: as one can bijectively associate a subgroup of the fundamental group of the base space to each of its covers, the balanced product codes that derive from applying the Borel construction to different covering spaces of the same base space correspond to the same fibre bundle code.

Finally, we argued that all lifted product codes could be interpreted as balanced product codes, at least, as long as one assumes that the abelian group forming the basis of the group algebra is of finite order. Conversely, every balanced product construction with the action of an abelian group of finite order is a lifted product code. As such, we see that we can use these results to relate fibre bundle products to lifted product codes, and moreover, we see that this analysis implies that the balanced product code construction is of interest only when considering non-abelian group actions or actions of infinite groups.

Having considered these constructions, we proceeded by comparing the performance of the twisted toric code to that of the regular, untwisted toric code. While applying a twist does not alter the rate of the code, we did see that the minimum distance of the twisted toric code was altered due to the twist, provided that the code at hand was implemented on a non-square lattice. We determined an explicit expression for the distance of the twisted toric code, and identified the domains where the twist maximised the minimum distance. We verified these results using numerical simulations.

Afterwards, we proceeded to analyse the scaling of the logical error rate of these twisted toric codes. We implemented the PyMatching decoder for this code and used it to perform simulations that allowed us to compare the scaling of logical error rates. We saw that, when implemented on a square lattice, the twisted toric code performed better than the untwisted toric code. To understand why this is the case, we turned to analytical methods. While for the untwisted toric code the scaling of the logical error rate at low values of the physical error rate has already been explained in [BBKM19], we generalised their results by deriving an exact expression for the number of minimum weight error strings that cause a logical error to occur for the twisted toric code, and as such derived an approximation for the logical error rate of such codes for low values of the physical error rate. We verified the validity of these results by comparing them with our numerical study.

We suggested two possible avenues for further research. Firstly, we believe that it is worthwhile to extend our analysis by taking into account the contributions of higher weight error strings to the logical error rate as well, as this would allow us to better understand the scaling of the logical error rate of the twisted toric codes. Due to computational limitations, we were unable to extend our numerical simulations to larger toric codes, and as such, were also unable to study the scaling of the logical error rate as a function of the twist numerically. With better analytical methods, however, one would hope to be able to do so. A necessary step in this regard, however, is to find an *explicit* expression for the number of uncorrectable minimum weight error strings, as the implicit expression that we have derived makes it hard to analyse and compare the effects of increasing the twist.

Secondly, we pointed towards the fact that the twisted toric codes, much like the regular toric code, appeared to possess a threshold value for the logical error rate. The threshold of the toric code has been analysed by mapping the toric code to a statistical mechanical model [DKLP02]. As the twisted toric code can be interpreted using a similar statistical model, we speculated that the arguments presented in [DKLP02] could be extended in order to determine the critical behaviour of the twisted toric code.

Finally, we take a step back and make some more general comments on the state of affairs. Firstly, we would like to raise the suggestion for another product code construction: one could choose to twist not just the fibre complex, but also to twist the base complex. Doing so in the simple case of the toric code, this approach seems to yield a different code than the twisted toric code we considered in our work. It would be interesting to consider how such an additional twist would alter the performance of the code even further, and we speculate whether this construction could be used to at least improve upon the distance scaling of the codes constructed in [HHO21].

Secondly, we note that although the families of codes that were constructed in the literature all have the advantage of requiring a bounded number of checks per qubit and a bounded number of qubits involved in each check, and, at least for asymptotically good codes constructed in [PK22a], they are good in the sense of having a favourable scaling of the distance and the rate, one ought to consider other factors as well to determine the usefulness of these codes. For example, three of the useful properties of the surface code are the *locality* of the checks involved, the fact that the code can be embedded on a planar surface, and the existence of an efficient decoder, e.g. decoders using minimum weight perfect matching. In contrast, the asymptotically good codes constructed in [PK22a] have recently been shown to be interpretable as part of a tessellation of an 11-dimensional manifold, and as such, such codes are in no way guaranteed to be implementable on a planar surface. Next to that, although the qLDPC constructed have low weight, it is also not at all guaranteed that the checks involved can be implemented in a local manner. More generally, with all the constructions considered (with the exception of the balanced product construction) we were able to obtain their results using random code construction. It would therefore be interesting to consider whether these methods can be used to devise an explicit class of (asymptotically) good qLDPC codes instead.

A

On Abelian Categories

In this section, we provide an overview of the theory of abelian categories, which is mostly based on [Wei94, Rie16].

We work up towards the definition of abelian categories by slowly building up their defining structure. To this end, we start off with a more primitive type of categories: **Ab**-category. Generalising our scope to such categories ensures that we preserve the structure of the boundary maps (i.e. the ability to add and compose them unambiguously).

Definition A.1 (**Ab**-category)**.** A category $\mathcal{A}$ is called an ***Ab**-category* if $\forall A, B \in \mathcal{A}$: $\text{Hom}_{\mathcal{A}}(A, B)$ possesses an abelian group structure such that composition distributes over addition.

Given that the Hom-sets of **Ab**-categories posses an abelian group structure, one is naturally inclined to consider structure preserving maps between these Hom-sets. We therefore introduce the following notion:

Definition A.2. A functor $F : \mathcal{A} \to \mathcal{B}$ between **Ab**-categories is called *additive* if the induced map $\text{Hom}_{\mathcal{A}}(A, B) \to \text{Hom}_{\mathcal{B}}(F(A), F(B))$ is a group homomorphism for all $A, B \in \mathcal{A}$.

Taking a small side track, we note that the concept of a Cartesian product of sets and a direct sum of groups can be captured categorically in two dual notions:

Definition A.3 (product)**.** Let $A, B \in \mathcal{A}$. The *product* of A and B is an object $A \times B \in \mathcal{A}$ together with two projection maps $\pi_A : A \times B \to A$ and $\pi_B : A \times B \to B$ that possesses the following universal property:

$$\begin{tikzcd}
 & A \arrow[dl] \\
T & A \times B \arrow[l, dashed, "\exists!"'] \arrow[u, "\pi_A"'] \arrow[d, "\pi_B"] \\
 & B \arrow[ul]
\end{tikzcd}$$

The coproduct is the categorical dual notion of a product. In the category of chain complexes over vector spaces, direct product and the direct sum are the product and coproduct, respectively. We would like to retain this structure (cf. the additivity Eilenberg-Steenrod axiom), and moreover, we would like these two structures to coincide. To this end, we restrict our attention to additive categories:

Definition A.4 (additive category)**.** An **Ab**-category $\mathcal{A}$ is called an *additive category* if $\forall A, B \in \mathcal{A}$, the product $A \times B$ exists, moreover it possesses a zero object.

Lastly, in order to define a homology functor on such a category, we need to be able to speak of images and kernels.

Definition A.5. Given an additive category $\mathcal{A}$, a *kernel* of a morphism $f : B \to C$,ker(f), is a map $i : A \to B$ that is universal with respect to the property that $f \circ i = 0$. A *cokernel* is the categorical dual of a kernel. Furthermore, a map $i : A \to B$ is *monic* if for every map $g : A \to A'$: $g \circ i = 0$ implies that $g = 0$. An *epi* map is categorically dual to a monic map.

This, finally, brings us to the definition of an abelian category.

Definition A.6. A category $\mathcal{A}$ is called an *abelian category* if it is an additive category and moreover every map in $\mathcal{A}$ has a kernel and a cokernel, while for every monic map i: $i = \ker(\text{coker}(i))$ and for every epi map e, we have $e = \text{coker}(\ker(e))$.

One can now generalise the notions of a chain complex, of a chain complex morphism, and of an exact sequence to abelian categories. Similarly, one can define a homology functor $\mathcal{H}_* : \textbf{Ch}(\textbf{Ab}) \to \textbf{Ab}$ in precisely the same manner, that is, by defining:

$$\mathcal{H}_n(\mathcal{C}) = \ker(\partial_n)/\text{Im}(\partial_{n+1}) \tag{A.1}$$

An example of an abelian category is the category $\mathcal{R}$ – **Mod**. Surprisingly enough, the converse statement is also true — at least, for small abelian categories: every small abelian category can be embedded in the category of ring modules in a sense that we will now make precise:

Theorem A.1 (Freyd-Mitchell Embedding Theorem)**.** *Let $\mathcal{A}$ be a small abelian category. Then there exists a ring $\mathcal{R}$ along with a functor $F : \mathcal{A} \to \mathcal{R}$ – **Mod** that is fully faithful such that Hom$_{\mathcal{A}}(A,B) \simeq$ Hom$_{\mathcal{R}-\textbf{Mod}}(F(A),F(B))$, i.e. that embeds $\mathcal{A}$ as a full subcategory of $\mathcal{R}$ – **Mod**.*

The proof of this theorem follows from Yoneda's lemma, but is somewhat involved and hence omitted. Nevertheless, this theorem implies that it suffices to consider chain complexes of modules over rings to understand chain complexes of (small) abelian categories.

B

Python Scripts for Numerical Simulations

The Python scripts used to obtain the results in Chapter 10 are presented here.

B.1. Simulations and the Logical Error Rate

```python
"""
Author: Stephan M. Loor (QuTech/TU Delft)
Code partly based on the documentation on MWPM provided by Oscar Higgott, see: https://
    pymatching.readthedocs.io/en/latest/toric-code-example.html.
"""

import numpy as np
import math
import sys
np.set_printoptions(threshold=sys.maxsize)
import  matplotlib
import itertools
import matplotlib.pyplot as plt
import os
from scipy.sparse import hstack, kron, eye, csr_matrix, block_diag, vstack
from pymatching import Matching

matplotlib.use("pgf")

plt.style.use('plot_style.txt')

matplotlib.rcParams.update({
    "pgf.texsystem": "pdflatex",
    'font.family': 'serif',
    'text.usetex': True,
    'pgf.rcfonts': False,
})

matplotlib.rcParams['mathtext.fontset'] = 'stix'
matplotlib.rcParams['font.family'] = 'STIXGeneral'

def repetition_code(n):
    """
    INPUT: n, integer
    OUTPUT: csr_matrix
    Produces the parity check matrix of a repetition code mapping onto an n-bit space.
    """
    row_ind, col_ind = zip(*((i, j) for i in range(n) for j in (i, (i+1)%n)))
    data = np.ones(2*n, dtype=np.uint8)
    return csr_matrix((data, (row_ind, col_ind)))

def permutation_matrix(n,k):
    """
    n x n Permutation matrix for k-shifts.
    """
    row_ind, col_ind = zip(*((i, j) for i in range(n) for j in ((i-k)%n,)))
    data = np.ones(n, dtype=np.uint8)
    return csr_matrix((data, (row_ind, col_ind)))

def toric_code_twisted_X_stabilisers(L1,L2,twist_shift):
    """
    Sparse check matrix for the X stabilisers of a toric code with
    lattice size L1xL2, constructed as the hypergraph product of
    two repetition codes.
    """
    Hr1 = repetition_code(L1)
    Hr2 = repetition_code(L2)
    upper_bdry = csr_matrix(([1],([Hr1.shape[0]-1],[0])),shape=(Hr1.shape[0],Hr1.shape[1]))
    twisted_mat = (kron((Hr1+upper_bdry),eye(Hr2.shape[1])) + kron(upper_bdry,
    permutation_matrix(Hr2.shape[1],twist_shift)))
    H = hstack(
            [twisted_mat, kron(eye(Hr1.shape[0]), Hr2.T)],
            dtype=np.uint8
        )
    H.data = H.data % 2
    H.eliminate_zeros()

    return csr_matrix(H)

def toric_code_twisted_Z_stabilisers(L1,L2,twist_shift):
    Hr1 = repetition_code(L1)
    Hr2 = repetition_code(L2)
    upper_bdry = csr_matrix(([1],([Hr1.shape[0]-1],[0])),shape=(Hr1.shape[0],Hr1.shape[1]))
    twisted_mat = (kron((Hr1+upper_bdry),eye(Hr2.shape[1])) + kron(upper_bdry,
    permutation_matrix(Hr2.shape[1],twist_shift)))
    H = hstack(
            [kron(eye(Hr1.shape[0]), Hr2.T).T,twisted_mat.T],
            dtype=np.uint8
        )
    H.data = H.data % 2
    H.eliminate_zeros()
    return csr_matrix(H)

def toric_code_twisted_X_logicals(L1,L2,twist_shift):
    """
    Sparse binary matrix with each row corresponding to an X logical operator
    of a toric code with lattice size L1 x L2.
    """
    H1_L1 = csr_matrix(([1], ([0],[1])), shape=(1,L1), dtype=np.uint8)
    H1_L2 = csr_matrix(([1], ([0],[0])), shape=(1,L2), dtype=np.uint8)

    H0_L1 = csr_matrix(np.ones((1, L1), dtype=np.uint8))
    H0_L2 = csr_matrix(np.ones((1, L2), dtype=np.uint8))

    if (twist_shift % L2) == 0:
        H_twist_hor = np.zeros((1,L1*L2))
    elif (twist_shift % L2) <= L2 - (twist_shift % L2):
        row_ind_hor, col_ind_hor = zip(*((0, j) for j in (np.arange((L2-twist_shift+1)%L2,L2
    +1) % L2)))
        data_hor =  np.ones(len(col_ind_hor))
        H_twist_hor = csr_matrix((data_hor,(row_ind_hor,col_ind_hor)),shape = (1,L1*L2))
    else:
        row_ind_hor, col_ind_hor = zip(*((0, j) for j in np.arange(1,(L2-twist_shift+1)%L2) )
    )
        data_hor =  np.ones(len(col_ind_hor))
        H_twist_hor = csr_matrix((data_hor,(row_ind_hor,col_ind_hor)),shape = (1,L1*L2))

    H_twist = hstack([H_twist_hor, kron(H0_L1, H1_L2)])

    x_logicals = vstack([hstack([kron(H1_L1, H0_L2), csr_matrix(([0]),shape=(1,L1*L2))]),
    H_twist])
    x_logicals.data = x_logicals.data % 2
    x_logicals.eliminate_zeros()

    return csr_matrix(x_logicals)

def toric_code_twisted_Z_logicals(L1,L2,twist_shift):

    H1_L1 = csr_matrix(([1], ([0],[0])), shape=(1,L1), dtype=np.uint8)
    H1_L2 = csr_matrix(([1], ([0],[0])), shape=(1,L2), dtype=np.uint8)

    H0_L1 = csr_matrix(np.ones((1, L1), dtype=np.uint8))
    H0_L2 = csr_matrix(np.ones((1, L2), dtype=np.uint8))

    if (twist_shift % L2) == 0:
        H_twist_hor = np.zeros((1,L1*L2))
    elif (twist_shift % L2) <= L2 - (twist_shift % L2):
        row_ind_hor, col_ind_hor = zip(*((0, j) for j in range(L1*L2 - L2,L1*L2 - L2  + (
    twist_shift % L2))))
        data_hor =  np.ones(len(col_ind_hor))
        H_twist_hor = csr_matrix((data_hor,(row_ind_hor,col_ind_hor)),shape = (1,L1*L2))
    else:
        row_ind_hor, col_ind_hor = zip(*((0, j) for j in range(L1*L2 - (L2 - (twist_shift %
    L2)),L1*L2)))
        data_hor =  np.ones(len(col_ind_hor))
        H_twist_hor = csr_matrix((data_hor,(row_ind_hor,col_ind_hor)),shape = (1,L1*L2))

    H_twist = hstack([kron(H0_L1, H1_L2),H_twist_hor])
    z_logicals = vstack([hstack([csr_matrix(([0]),shape=(1,L1*L2)),kron(H1_L1, H0_L2)]),
    H_twist])

    z_logicals.data = z_logicals.data % 2
    z_logicals.eliminate_zeros()
    return csr_matrix(z_logicals)


def num_decoding_failures(H, logicals, p, num_trials):
    matching = Matching(H)
    num_errors = 0
    noise_list = []
    for trial in range(num_trials):
        noise = np.random.binomial(1, p, H.shape[1])
        syndrome = H@noise % 2
        correction = matching.decode(syndrome, num_neighbours=None)
        error = (noise + correction) % 2
        if np.any(H@error.T % 2): ##check wether we really end up back in code space
            print(f”Decoding mapped outside of the code space: the noise term is {noise}.\n
    The associated syndrome is {syndrome}. \n The induced error is {error}.”)
            print((H@error.T).toarray() % 2)

        if np.any(error@logicals.T % 2):
            num_errors += 1
            noise_list.append(np.sum(noise))
    return num_errors


def A(l1,l2,w):
    if l2 < 1 or l1 < 1:
        return 0

    if w > 1:
        x = 0

        ### Interior Points (Region I)
        for i in range(1,l1):
            for j in range(1,l2):
                x+= 2*A(i,j,w-1)

        ### Boundary Points
        ### Region III
        for i in range(1,l1):
            x+= A(i,l2,w-1)

        ### Region II
        for j in range(1,l2):
            x += A(l1,j,w-1)

        return x
```

```python

    if w == 1:
        return 2*l1*l2 - l1 - l2

def N(L1,k):
    w = int((L1 + k)/2)
    N = 0
    for i in range(1,L1+1):
        N += A(i,k+1,w-1)
        N += A(i,k,w-1)
    return N

def probability_errors_twisted(L1,L2,p,twist):
    n = 2*L1*L2
    num_loops = 1/2*L2*N(L1,twist)
    prob = ((1-p)**n) * num_loops * (p/(1-p))**((L1+twist)/2)
    return prob

def probability_errors_untwisted(L1,L2,p):
    n = 2*L1*L2
    num_loops = 1/2*L2*math.comb(L1,int(L1/2))
    prob = ((1-p)**n) * num_loops * (p/(1-p))**(L1/2)
    return prob


def test_logicals(L1_range,L2_range):
    no_Hx_errors = True
    no_Hz_errors = True

    for L1 in L1_range:
        for L2 in L2_range:
            for shift in np.arange(0,L1*L2):
                print(f"Testing for for L1 = {L1} L2 = {L2} k = {shift} ")

                Hz = toric_code_twisted_Z_stabilisers(L1,L2,shift)
                logX = toric_code_twisted_X_logicals(L1,L2,shift)

                for i in range(0,logX.shape[0]):
                    row = logX.getrow(i)
                    synd = (Hz@row.T).toarray() % 2
                    if int(np.sum(synd)) != 0:
                        print(f"There is an imcompatibility between the X-logicals and Z-
    stabilisers for L1 = {L1} L2 = {L2} k = {shift} ")
                        no_Hz_errors = False
                Hx = toric_code_twisted_X_stabilisers(L1,L2,shift)
                logZ = toric_code_twisted_Z_logicals(L1,L2,shift)

                for i in range(0,logZ.shape[0]):
                    row = logZ.getrow(i)
                    synd = (Hx@row.T).toarray() % 2
                    if int(np.sum(synd)) != 0:
                        print(f"There is an imcompatibility between the Z-logicals and X-
    stabilisers for L1 = {L1} L2 = {L2} k = {shift} ")
                        no_Hx_errors = False

    if no_Hx_errors:
        print(f"No incompatibility detected between the Z-logicals and X-stabilisers!")
    if no_Hz_errors:
        print(f"No incompatibility detected between the X-logicals and Z-stabilisers!")


def main_one_twist(num_trials,L_init,L_fin,L_step,L1_scale,L2_scale,p_init,p_fin,p_steps,
    twist_shift):
    """
    INPUT:
    OUTPUT:
    """
    Ls = range(L_init,L_fin,L_step)
    ps = np.linspace(p_init,p_fin,p_steps)

    np.random.seed(2)
    log_errors_all_L = []
    for L in Ls:
        L1 = L1_scale*L
        L2 = L2_scale*L
        print(f"Simulating L1={L1}, L2={L2}...")
        Hx = toric_code_twisted_X_stabilisers(L1,L2,twist_shift)
        logX = toric_code_twisted_X_logicals(L1,L2,twist_shift)
        log_errors = []
        for p in ps:
            print(f"Simulating p={p}...")
            num_errors = num_decoding_failures(Hx, logX, p, num_trials)
            log_errors.append(num_errors/num_trials)
        log_errors_all_L.append(np.array(log_errors))

    plot_folder_name = f"./Plots{num_trials}"
    if not os.path.isdir(plot_folder_name):
        os.makedirs(plot_folder_name)

    plt.figure()
    for L, logical_errors in zip(Ls, log_errors_all_L):
        L1 = L1_scale*L
        L2 = L2_scale*L
        std_err = (logical_errors*(1-logical_errors)/num_trials)**0.5
        plt.errorbar(ps, logical_errors, yerr=std_err, label=r"$L_1$ = {}, $L_2$ = {}".format
    (L1,L2))
    plt.title(f"Shift: $k = $ {twist_shift}")
    plt.xlabel("Physical error rate")
    plt.ylabel("Logical error rate")
    plt.legend(loc=2)
    plt.savefig(f"{plot_folder_name}/L1init={L_init}Shift={twist_shift}L1scale={L1_scale}
    L2scale={L2_scale}.pgf")
    plt.show()

def main_one_size(num_trials,L_lattice,L1_scale,L2_scale,p_init,p_fin,p_steps,shift_array,
    FITS=False):
    ps = np.linspace(p_init,p_fin,p_steps)
    np.random.seed(2)
    L1 = int(L1_scale*L_lattice)
    L2 = int(L2_scale*L_lattice)
    print(f"L1 = {L1}, L2 = {L2}...")
    log_errors_all_shifts = []
    log_twists = []

    for twist_shift in shift_array:
        print(f"Simulating shift={twist_shift}...")
        Hx = toric_code_twisted_X_stabilisers(L1,L2,twist_shift)
        logX = toric_code_twisted_X_logicals(L1,L2,twist_shift)
        log_errors = []

        for p in ps:
            print(f"Simulating p={p}...")
            num_errors = num_decoding_failures(Hx, logX, p, num_trials)
            log_errors.append(num_errors/num_trials)
        log_errors_all_shifts.append(np.array(log_errors))
        log_twists.append(twist_shift)


    plot_folder_name = f"./Plots{num_trials}"
    if not os.path.isdir(plot_folder_name):
        os.makedirs(plot_folder_name)

    plt.figure()
    colors_lst = ["b","g","r","y","c","m","b","g"]
    for twist_shift, logical_errors in zip(log_twists, log_errors_all_shifts):
        std_err = (logical_errors*(1-logical_errors)/num_trials)**0.5
        plt.errorbar(ps, logical_errors, yerr=std_err, label=r"$k = $ {}".format(twist_shift)
    ,c=colors_lst[twist_shift])
        if FITS:
            analytic_error_base_term = probability_errors_twisted(L1,L2,ps,twist_shift)
            plt.plot(ps,analytic_error_base_term,ls=":",label=r"fit for $k = $ {}".format(
    twist_shift),c=colors_lst[twist_shift+1])
    plt.title(r"$n_{{\mathcal{{B}}}}$ = {}, $n_{{\mathcal{{F}}}}$ = {}".format(L1,L2))
    plt.xlabel("Physical error rate")
    plt.ylabel("Logical error rate")

    if LOGLOG:
        plt.loglog(nonposx='clip', nonposy='clip')
        plt.xlabel(r'$\log(p_{phys]})$')
        plt.ylabel(r'$\log(p_{log})$')

    plt.legend(loc=2)
    if FITS:
        plt.savefig(f"{plot_folder_name}/Shiftarray={shift_array}L={L_lattice}L1={L1}L2={L2}
    Fitted.pgf")
    else:
       plt.savefig(f"{plot_folder_name}/Shiftarray={shift_array}L={L_lattice}L1={L1}L2={L2}.
    pgf")
    plt.show()


def main_multiple_twists_and_sizes(L1_L2_twist_array,num_trials,p_init,p_fin,p_steps):
    ps = np.linspace(p_init,p_fin,p_steps)
    np.random.seed(2)
    log_errors_all_diffs = []
    triple_lst = []
    for triple in L1_L2_twist_array:
        L1,L2,twist = triple
        triple_lst.append(np.array((L1,L2,twist)))
        print(f"L1 = {L1}, L2 = {L2}, k = {twist}...")
        Hx = toric_code_twisted_X_stabilisers(L1,L2,twist)
        logX = toric_code_twisted_X_logicals(L1,L2,twist)
        log_errors = []
        for p in ps:
            print(f"Simulating p={p}...")
            num_errors = num_decoding_failures(Hx, logX, p, num_trials)
            log_errors.append(num_errors/num_trials)
        log_errors_all_diffs.append(np.array(log_errors))

    plt.figure()
    colors_lst = ["b","g","r","y","c","m","b","g"]
    for triple, logical_errors in zip(triple_lst,log_errors_all_diffs):
        std_err = (logical_errors*(1-logical_errors)/num_trials)**0.5
        if triple[1] == 6:
            plt.errorbar(ps, logical_errors, yerr=std_err, label=r"$n_{{\mathcal{B}}}} = $
    {}, $n_{{\mathcal{F}}}} = {}, k = {}$".format(triple[0],triple[1],triple[2]),ls="--")
        else:
            plt.errorbar(ps, logical_errors, yerr=std_err, label=r"$n_{{\mathcal{B}}}} = $
    {}, $n_{{\mathcal{F}}}} = {}, k = {}$".format(triple[0],triple[1],triple[2]))
    plt.xlabel("Physical error rate")
    plt.ylabel("Logical error rate")
    plt.legend(loc=2)
    plot_folder_name = f"./Plots{num_trials}"
    if not os.path.isdir(plot_folder_name):
        os.makedirs(plot_folder_name)

    plt.savefig(f"{plot_folder_name}/multiplot_toric_code.pgf")
    plt.show()

if __name__ == "__main__":

    CHECK_DEFINITIONS = False
    ONE_SIZE = False
    ONE_TWIST = False
    FITS = False
    MULTI_SIZES_TWIST = True
    LOGLOG = False

    num_trials = 10000
    p_init = 0.0001
    p_fin = 0.02
    p_steps = 20

    L1_scale = 1
    L2_scale = 2


    if CHECK_DEFINITIONS:
        L1_range = np.arange(4,12)
        L2_range = np.arange(4,12)
        test_logicals(L1_range,L2_range)

    if ONE_SIZE:
        L_lattice = 4
        shift_array = np.array([0,])
        main_one_size(num_trials,L_lattice,L1_scale,L2_scale,p_init,p_fin,p_steps,shift_array
,FITS)

    if ONE_TWIST:
        L_init = 4
        L_fin = 9
        L_step = 2
        twist_shift = 0
        main_one_twist(num_trials,L_init,L_fin,L_step,L1_scale,L2_scale,p_init,p_fin,p_steps,
twist_shift)

    if MULTI_SIZES_TWIST:
        L1_L2_twist_array = [np.array((8,16,0)),np.array((8,16,2)),np.array((8,16,4))]
        main_multiple_twists_and_sizes(L1_L2_twist_array,num_trials,p_init,p_fin,p_steps)
```

B.2. Numerical Verification of the Distance

```
import sys
import networkx as nx
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import pyximport; pyximport.install()
from flinalg import *
from scipy import sparse
from scipy.sparse import hstack, kron, eye, csr_matrix


matplotlib.use("pgf")

plt.style.use('plot_style.txt')

matplotlib.rcParams.update({
    "pgf.texsystem": "pdflatex",
    'font.family': 'serif',
    'text.usetex': True,
    'pgf.rcfonts': False,
})

matplotlib.rcParams['mathtext.fontset'] = 'stix'
matplotlib.rcParams['font.family'] = 'STIXGeneral'



def get_coh1_reps(b1,b2):
    """find representatives of basis of H^1"""
    coB1ker = kernel(np.asarray(b2.T.todense()))
    coB0im = np.asarray(b1.T.todense())
    Q, W = quotient_basis(coB1ker, coB0im)
    return np.asarray(Q)


def doubled_graph(G,chain):
    """takes G and cuts all edges in the chain"""
    """then pastes in a copy of G called H"""
    VG = [(v,'G') for v in G.nodes()]
    EG = [((e[0],'G'),(e[1],'G')) for e in G.edges()]
    VH = [(v,'H') for v in G.nodes()]
    EH = [((e[0],'H'),(e[1],'H')) for e in G.edges()]

    D = nx.Graph()
    D.add_nodes_from(VG)
    D.add_nodes_from(VH)
    D.add_edges_from(EG)
    D.add_edges_from(EH)

    for e in chain:
        D.add_edge((e[0],'G'),(e[1],'H'))
        D.add_edge((e[0],'H'),(e[1],'G'))
        D.remove_edge((e[0],'G'),(e[1],'G'))
        D.remove_edge((e[0],'H'),(e[1],'H'))

    return D


def vertex_path_to_1chain(l,bound1):
    """
    INPUT: list of vertices
    OUTPUT: numpy array with support on edges of path
    """
    res = np.zeros(bound1.shape[0],dtype='uint8')
    for u,v in zip(l[:-1],np.roll(l[:-1],-1)):
        for i in range(bound1.shape[0]):
            if bound1[i,v] and bound1[i,u]:
                res[i] = 1
                break
    return res


def shortest_crossing(G,chain):
    """
    returns the shortest cycle in G which has odd support on chain
    G : NetworkX graph
    chain : list of edges of G
    """
    D = doubled_graph(G,chain)
    ret = sys.maxsize
    for v in G.nodes():
        try:
            pathlength = nx.shortest_path_length(D,source=(v,'G'),target=(v,'H'))
            if pathlength < ret:
                ret = pathlength
                rep = nx.shortest_path(D,source=(v,'G'),target=(v,'H'))
        except:
            pass
    assert ret < sys.maxsize
    rep = [x[0] for x in rep]
    return ret, rep


def systoles(b0,b1,b2):
    """
    INPUT: boundary operators as scipy sparse matrices
    OUTPUT: weight and minimal cycle
    """
    edges = []
    for i in range(b1.shape[0]):
        v,w = b1.getrow(i).nonzero()[1]
        edges.append((v,w))
    G = nx.Graph()
    G.add_edges_from(edges)

    coh1_reps = get_coh1_reps(b1,b2)

    ret = []
    for j,c in enumerate(coh1_reps):
        chain = []
        for i in range(len(edges)):
            if c[i]==1:
                chain.append(edges[i])
        weight, rep = shortest_crossing(G,chain)
        ret.append((weight,rep))
    return ret


def cosystoles(b0,b1,b2):
    cob0 = b1.T
    cob1 = b2.T
    cob2 = sparse.csr_matrix(np.ones((b2.shape[0],1),dtype='uint8'))
    return systoles(cob2,cob1,cob0)


def systole_lengths(b0,b1,b2):
    systo = systoles(b0,b1,b2)
    return [w for w,r in systo]


def cosystole_lengths(b0,b1,b2):
    systo = cosystoles(b0,b1,b2)
    return [w for w,r in systo]


###From here on: written by SML

def repetition_code(n):
    """
    Parity check matrix of a repetition code with length n.
    """
    row_ind, col_ind = zip(*((i, j) for i in range(n) for j in (i, (i+1)%n)))
    data = np.ones(2*n, dtype=np.uint8)
    return csr_matrix((data, (row_ind, col_ind)))

def permutation_matrix(n,k):
    """
    n x n Permutation matrix for k-shifts.
    """
    row_ind, col_ind = zip(*((i, j) for i in range(n) for j in ((i-k)%n,)))
    data = np.ones(n, dtype=np.uint8)
    return csr_matrix((data, (row_ind, col_ind)))

def HX(L1,L2,twist_shift):
    Hr1 = repetition_code(L1)
    Hr2 = repetition_code(L2)
    upper_bdry = csr_matrix(([1],([Hr1.shape[0]-1],[0])),shape=(Hr1.shape[0],Hr1.shape[1]))
    twisted_mat = (kron((Hr1+upper_bdry),eye(Hr2.shape[1])) + kron(upper_bdry,
    permutation_matrix(Hr2.shape[1],twist_shift)))
    H = hstack(
            [twisted_mat, kron(eye(Hr1.shape[0]), Hr2.T)],
            dtype=np.uint8
        )
    H.data = H.data % 2
    H.eliminate_zeros()
    return csr_matrix(H)

def HZ(L1,L2,twist_shift):
    Hr1 = repetition_code(L1)
    Hr2 = repetition_code(L2)
    upper_bdry = csr_matrix(([1],([Hr1.shape[0]-1],[0])),shape=(Hr1.shape[0],Hr1.shape[1]))
    twisted_mat = (kron((Hr1+upper_bdry),eye(Hr2.shape[1])) + kron(upper_bdry,
    permutation_matrix(Hr2.shape[1],twist_shift)))
    H = hstack(
            [kron(eye(Hr1.shape[0]), Hr2.T).T,twisted_mat.T],
            dtype=np.uint8
        )
    H.data = H.data % 2
    H.eliminate_zeros()
    return csr_matrix(H)

def Plotter(L1,L2,twist_array):
    L1_vals = []
    L2_vals = []
    for twist in twist_array:
        Hx = HX(L1,L2,twist)
        Hz = HZ(L1,L2,twist)
        w = systole_lengths(None,Hx.T,Hz)
        L1_vals.append(w[0])
        L2_vals.append(w[1])
    plt.figure()
    plt.plot(twist_array,L2_vals,"ro--",label=r"$d_1$")
    plt.plot(twist_array,L1_vals,"bo--",label=r"$d_2$")

    plt.title(r"$n_{{\mathcal{{B}}}}$ = {}, $n_{{\mathcal{{F}}}}$ = {}".format(L1,L2))
    plt.xlabel(r"$k$")
    plt.ylabel(r"$d$")
    plt.legend(loc=0)
    plt.savefig(f"./L1={L1}L2={L2}shift={twist_array[-1]}.pgf")
    plt.show()


if __name__ == "__main__":
    L1 = 4
    L2 = 8
    twist_array = range(0,2*max(L1,L2)+1,1)
    Plotter(L1,L2,twist_array)
```

Bibliography

[AAB$^+$19] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 10 2019.

[ABO08] Dorit Aharonov and Michael Ben-Or. Fault-Tolerant Quantum Computation with Constant Error Rate. *SIAM Journal on Computing*, 38(4):1207–1282, 1 2008.

[AC19] Benjamin Audoux and Alain Couvreur. On tensor products of CSS codes. *Annales de l'Institut Henri Poincaré D*, 6(2):239–287, 3 2019.

[BBKM19] Michael E Beverland, Benjamin J Brown, Michael J Kastoryano, and Quentin Marolleau. The role of entropy in topological quantum error correction. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(7):073404, 7 2019.

[BE21a] Nikolas P. Breuckmann and Jens N. Eberhardt. Balanced Product Quantum Codes. *IEEE Transactions on Information Theory*, 67(10):6653–6674, 10 2021.

[BE21b] Nikolas P. Breuckmann and Jens Niklas Eberhardt. Quantum Low-Density Parity-Check Codes. *PRX Quantum*, 2(4):040101, 10 2021.

[BK98] S. B. Bravyi and A. Yu. Kitaev. Quantum codes on a lattice with boundary. 11 1998.

[Bro14] Dan Browne. *Topological Codes and Computation*. 2014.

[BTL10] Sergey Bravyi, Barbara M Terhal, and Bernhard Leemhuis. Majorana fermion codes. *New Journal of Physics*, 12(8):083039, 8 2010.

[BVC$^+$17] Nikolas P Breuckmann, Christophe Vuillot, Earl Campbell, Anirudh Krishna, and Barbara M Terhal. Hyperbolic and semi-hyperbolic surface codes for quantum storage. *Quantum Science and Technology*, 2(3):035007, 9 2017.

[CRO$^+$19] Yudong Cao, Jonathan Romero, Jonathan P. Olson, Matthias Degroote, Peter D. Johnson, Mária Kieferová, Ian D. Kivlichan, Tim Menke, Borja Peropadre, Nicolas P. D. Sawaya, Sukin Sim, Libor Veis, and Alán Aspuru-Guzik. Quantum Chemistry in the Age of Quantum Computing. *Chemical Reviews*, 119(19):10856–10915, 10 2019.

[DK01] James Davis and Paul Kirk. *Lecture Notes in Algebraic Topology*, volume 35. American Mathematical Society, 1 edition, 2001.

[DKLP02] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 9 2002.

[EKZ20] Shai Evra, Tali Kaufman, and Gilles Zemor. Decodable Quantum LDPC Codes beyond the Square Root Distance Barrier using High Dimensional Expanders. In *Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS*, volume 2020-November, 2020.

[ES45] Samuel Eilenberg and Norman E. Steenrod. Axiomatic Approach to Homology Theory. *Proceedings of the National Academy of Sciences*, 31(4):117–120, 4 1945.

[FSG09] Austin G. Fowler, Ashley M. Stephens, and Peter Groszkowski. High-threshold universal quantum computation on the surface code. *Physical Review A*, 80(5):052312, 11 2009.

[Geo78] George W. Whitehead. *Elements of Homotopy Theory*. Springer, New York, NY, 1 edition, 1978.

[Got13] Daniel Gottesman. Fault-Tolerant Quantum Computation with Constant Overhead. *Quantum Information & Computation*, 14(15-16):1338–1372, 10 2013.

[Hat01] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 1 edition, 2001.

[HHO21] Matthew B. Hastings, Jeongwan Haah, and Ryan O'Donnell. Fiber Bundle Codes: Breaking the {\$}N{\^{}}{\{}1/2{\}} {\textbackslash}operatorname{\{}polylog{\}}(N){\$} Barrier for Quantum LDPC Codes. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1276–1288, New York, NY, USA, 6 2021. ACM.

[Hig22] Oscar Higgott. PyMatching: A Python Package for Decoding Quantum Codes with Minimum-Weight Perfect Matching. *ACM Transactions on Quantum Computing*, 3(3):1–16, 9 2022.

[Kit03] A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 1 2003.

[Len18] Lenny Taelman. *Modules and Categories*. University of Amsterdam, 2018.

[LO14] H. Lenstra and F. Oort. *Ringen en Lichamen*. Leiden, 2014.

[Max13] Maximilien Holmberg-Péroux. The Serre Spectral Sequence. *École Polytechnique Fédérale de Lausanne*, 2013.

[MLFA$^+$22] Lars S Madsen, Fabian Laudenbach, Mohsen Falamarzi Askarani, Fabien Rortais, Trevor Vincent, Jacob F F Bulmer, Filippo M Miatto, Leonhard Neuhaus, Lukas G Helt, Matthew J Collins, Adriana E Lita, Thomas Gerrits, Sae Woo Nam, Varun D Vaidya, Matteo Menotti, Ish Dhand, Zachary Vernon, Nicolás Quesada, and Jonathan Lavoie. Quantum computational advantage with a programmable photonic processor. *Nature*, 606(2):75, 2022.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2010.

[PK22a] Pavel Panteleev and Gleb Kalachev. Asymptotically good Quantum and locally testable classical LDPC codes. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 375–388, New York, NY, USA, 6 2022. ACM.

[PK22b] Pavel Panteleev and Gleb Kalachev. Quantum LDPC Codes With Almost Linear Minimum Distance. *IEEE Transactions on Information Theory*, 68(1):213–229, 1 2022.

[Rie16] Emily Riehl. Category theory in context. *Dover Books on mathematics*, 2016.

[Sag21] Steffen Sagave. *Algebraic Topology*. Nijmegen, 2021.

[SH22] Steffen Sagave and Gijs Heuts. *Algebraic Topology 2*. 2022.

[Sho94] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE Comput. Soc. Press, 1994.

[ST] Yan Shuo Tan. A Skeleton in the Category: The Secret Theory of Covering Spaces.

[Ter15] Barbara M. Terhal. Quantum error correction for quantum memories, 4 2015.

[TZ09] Jean-Pierre Tillich and Gilles Zemor. Quantum LDPC codes with positive rate and minimum distance proportional to n^{1/2}. *IEEE Transactions on Information Theory*, 3 2009.

[Wei94] Charles A. Weibel. *An Introduction to Homological Algebra*. Cambridge University Press, 4 1994.

[ZCC$^+$22] Qingling Zhu, Sirui Cao, Fusheng Chen, Ming-Cheng Chen, Xiawei Chen, Tung-Hsun Chung, Hui Deng, Yajie Du, Daojin Fan, Ming Gong, Cheng Guo, Chu Guo, Shaojun Guo, Lianchen Han, Linyin Hong, He-Liang Huang, Yong-Heng Huo, Liping Li, Na Li, Shaowei Li, Yuan Li, Futian Liang, Chun Lin, Jin Lin, Haoran Qian, Dan Qiao, Hao Rong, Hong Su, Lihua Sun, Liangyuan Wang, Shiyu Wang, Dachao Wu, Yulin Wu, Yu Xu, Kai Yan, Weifeng Yang, Yang Yang, Yangsen Ye, Jianghan Yin, Chong Ying, Jiale Yu, Chen Zha, Cha Zhang, Haibin Zhang, Kaili Zhang, Yiming Zhang, Han Zhao, Youwei Zhao, Liang Zhou, Chao-Yang Lu, Cheng-Zhi Peng, Xiaobo Zhu, and Jian-Wei Pan. Quantum computational advantage via 60-qubit 24-cycle random circuit sampling. *Science Bulletin*, 67(3):240–245, 2 2022.

[ZP19] Weilei Zeng and Leonid P. Pryadko. Higher-Dimensional Quantum Hypergraph-Product Codes with Finite Rates. *Physical Review Letters*, 122(23):230501, 6 2019.

[ZP20] Weilei Zeng and Leonid P. Pryadko. Minimal distances for certain quantum product codes and tensor products of chain complexes. *Physical Review A*, 102(6):062402, 12 2020.