universität wien

# MASTERARBEIT | MASTER'S THESIS

Titel | Title

Grid Diagrams of Fibered Knots

verfasst von | submitted by

Paul Leon Itzlinger BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

| | |
|---|---|
| Studienkennzahl lt. Studienblatt \| Degree programme code as it appears on the student record sheet: | UA 066 821 |
| Studienrichtung lt. Studienblatt \| Degree programme as it appears on the student record sheet: | Masterstudium Mathematik |
| Betreut von \| Supervisor: | Assoz. Prof. Vera Vértesi Ph.D. |

I would like to thank my advisor, Professor Vera Vértesi, for the continuous support and the invaluable suggestions throughout this thesis.

# Abstract [DE]

Diese Masterarbeit befasst sich mit der Existenz spezieller Gitterdiagramme von gefaserten Knoten (fibered knots). Mithilfe von Gitterdiagrammen lässt sich eine Invariante für Knoten definieren, die sogennante Gitterhomologie (grid homology). Dies ist eine spezielle Form der Knoten-Floer-Homologie. Paulo Ghiggini und Yi Ni haben in [Ghi08] und [Ni07] gezeigt, dass Knoten-Floer-Homologie gefaserte Knoten erkennt. Aus ihren Resultaten folgt insbesondere, dass gewisse Gitterdiagramme nur für gefaserte Knoten existieren können, jedoch ist es unklar ob jeder gefaserte Knoten ein solches Gitterdiagramm besitzt, vgl. [OSS15, Open Problem 17.1.5]. Unser Hauptaugenmerk liegt auf diesem offenen Problem. Wir beginnen damit die notwendigen Grundlagen der Knotentheorie, der homologischen Algebra und der Gitterhomologie zu formulieren. Danach definieren wir Gitterdiagramme, die eine noch strengere Bedingung erfüllen als jene in [OSS15, Open Problem 17.1.5]. Anschließend entwickeln wir eine effiziente Methode, um zu testen, ob ein gegebenes Gitterdiagramm diese Kriterien erfüllt, und implementieren dies in einem Python-Programm. Im letzten Abschnitt der Arbeit verwenden wir unser Python-Programm um geeignete Gitterdiagramme für 5384 der 5397 gefaserten Primknoten mit Kreuzungszahl $\leq 13$ zu finden.

# Abstract [EN]

This thesis investigates the existence of special grid diagrams for fibered knots. Grid diagrams are combinatorial representations of knots that are used to define grid homology, a variant of knot Floer homology. Paulo Ghiggini and Yi Ni showed that knot Floer homology detects fibered knots [Ghi08], [Ni07]. In particular this implies that certain grid diagrams can only exist for fibered knots. Whether every fibered knot admits such a special diagram remains an open question, cf. [OSS15, Open Problem 17.1.5]. Our main point of focus lies on this open problem. After reviewing the necessary background in knot theory, homological algebra and grid homology, we define grid diagrams satisfying even stricter conditions than those considered in open problem 17.1.5. We then develop an efficient method for testing whether a given grid diagram meets these criteria and implement this method in a Python package. Using this package we find suitable grid diagrams for 5384 of the 5397 fibered prime knots with crossing number $\leq 13$.

# Contents

# Introduction

This thesis focuses on grid diagrams of fibered knots. Grid diagrams are special knot diagrams that are used to define a combinatorial version of knot Floer homology, called grid homology. Knot Floer homology can be used to compute other knot invariants. For example, Paolo Ghiggini and Yi Ni proved that knot Floer homology detects fibered knots [Ghi08], [Ni07]. Using these theorems it is not hard to show that certain special grid diagrams can only exist for fibered knots. It is however an open question whether all fibered knots do admit such special grid diagrams, cf. [OSS15, Open Problem 17.1.5]. The main part of this thesis focuses on this open question.

We first recall all the necessary definitions from knot theory, homological algebra and grid homology in Chapters 1 to 4. Chapter 1 recalls basic knot-theoretical notions such as knot isotopies, Reidemeister moves, knot genus, prime knots and fibered knots. Readers already familiar with these definitions can thus skip Chapter 1 altogether. Chapter 2 focuses on grid diagrams of knots, in particular toroidal grid diagrams and grid moves. This chapter closely follows Chapter 3 from the book *Grid Homology for Knots and Links* [OSS15]. Readers that are familiar with these notions should still read Section 2.1 as our definitions of grid moves differ slightly from those given in the source text. Chapter 3 is very short, it gives the necessary background from homological algebra in the setting of bigraded modules. Here we follow [OSS15, Appendix A]. Finally in Chapter 4 we define fully blocked and simply blocked grid homology for knots in the three sphere. This chapter can be seen as a condensed version of Chapters 4 and 5 from [OSS15] only containing those parts that are needed for our subsequent discussion, for example we omit the definition of unblocked grid homology.

After these preparations we come to the main part of this thesis, consisting of Chapters 5 and 6. In Chapter 5 we directly adress open problem 17.1.5. We define a class of grid diagrams that satisfy an even stronger condition than previously considered. In Section 5.2, we develop an efficient method to check whether a given grid diagram satisfies these conditions. Chapter 6 explains how these results can be implemented in a Python package designed to search for such grid diagrams for fibered knots. This approach succesfully finds suitable diagrams for almost all fibered prime knots of crossing number $\leq 13$. Only 12 of the 5397 fibered prime knots remain unsolved, cf. Table 6.1. The thesis concludes with a discussion about the existence of such grid diagrams for fibered knots and where to go from here.

# 1 Knot Theory

In this chapter we define the basic notions from knot theory needed throughout the text, drawing guidance from [Cro04], [OSS15, Chapter 2], and several other sources referenced in this chapter. We work in the smooth setting. To define a knot properly we have to recall the definition of an embedding between two manifolds.

**Definition 1.1.** A map $f : M \to N$ between two manifolds is called a *smooth embedding* if:

- The map $f$ is smooth and a homeomorphism onto its image
- For all $p \in M$ the tangent map $T_p f : T_p M \to T_{f(p)} N$ is injective.

**Definition 1.2.** A smooth embedding $K : S^1 \to \mathbb{R}^3$ is called a *knot* in $\mathbb{R}^3$.

Even though knots are defined as maps between manifolds, we often think of $K$ as the image of the underlying map. The simplest example of a knot is the circle $S^1$ canonically embedded into $\mathbb{R}^3$, we call this object the *unknot*. We want to consider two knots as equivalent if one can be deformed into the other. To make this mathematically precise we define an equivalence relation on the set of all knots that describes the physical deformation of a knot by pulling on its threads.

**Definition 1.3.** Two knots $K, K'$ in $\mathbb{R}^3$ are called *equivalent* if there exists a smooth map $H : S^1 \times [0,1] \to \mathbb{R}^3$ such that for all $t \in [0,1]$ the map $H_t = H(\cdot, t) : S^1 \to \mathbb{R}^3$ is a knot in $\mathbb{R}^3$ and $H_0 = K, H_1 = K'$.

It is easy to see that this definition defines an equivalence relation. Furthermore, the assumption that each $H_t$ is a knot ensures that we can not pass parts of the knot through itself during the deformation process. The smoothness assumption ensures that the deformation is not allowed to pull the knot infinitely tight as this would describe an equivalence between any knot and the unknot. We sometimes consider oriented knots, in this case we require that the map $H$ also preserves the orientation. Knots are a special case of the following object.

**Definition 1.4.** A collection of knots $K_1, \dots, K_n$ such that the images $K_i(S^1)$ are all disjoint is called a *link* in $\mathbb{R}^3$.

A link can also be given as a smooth embedding $L : \bigcup_{i=0}^{n} S^1 \to \mathbb{R}^3$ and the definition of equivalent knots extends readily to links.

---

**Convention:** From now on, whenever we speak of a knot or link, we refer to the whole equivalence class and not some fixed smooth embedding.

---

To look at examples of knots and links we first mention some facts about visualizing them on the plane.

**Theorem 1.5. ([Cro04, Theorem 3.2.1])** *Every link $L$ can be drawn on the plane by a smooth projection that only intersects itself transversally at finitely many points and there exist no three points of $L$ that are mapped to the same point under the projection.*

A projection of this type is called *regular*. For smooth knots, the proof of this theorem uses tools from differential topology that are out of scope for what we need in this text.

**Definition 1.6.** If we indicate the undercrossings of a regular projection by broken segments this unambiguously describes the underlying link. We call this object a *link diagram*, cf. Figure 1.1.

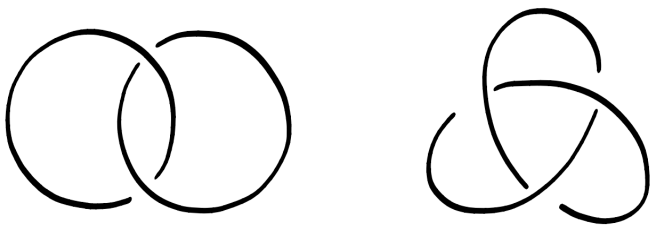


**Figure 1.1.** A diagram of the Hopf link (left) and a diagram of the trefoil knot (right).

**Definition 1.7.** We call a continuous deformation of a link diagram that does not change any crossing a *planar isotopy*.

Planar isotopies are clearly not enough to create all possible link diagrams for a given link. However, only three simple modifications on link diagrams are needed in addition as the next theorem shows.

**Theorem 1.8. ([Ros04])** *Two link diagrams represent the same link if and only if they differ by a finite sequence of planar isotopies and* Reidemeister moves*, cf. Figure 1.2.*

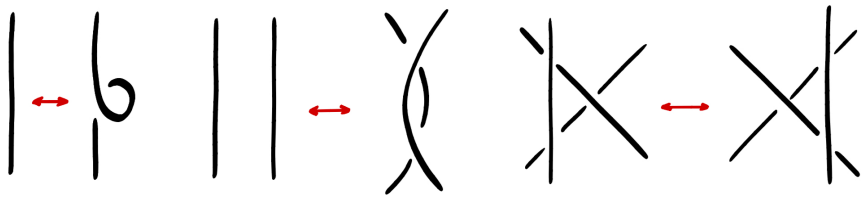


**Figure 1.2.** From left to right: The three Reidemeister moves R1, R2, R3.

This proof again uses tools from differential topology. Originally, this theorem was proven by Reidemeister for polygonal knots, cf. [Rei13, §3]. While this is a powerful theorem, it does not provide us with an algorithm to verify if two different diagrams represent the same link, in general this is very hard. In Chapter 6 we use use an online database consisting of a large list of knots to investigate an open question in knot theory. Most databases of knots consist of prime knots only. To define these, we first look at a way of factorising knots.

**Definition 1.9.** Let $S$ be a sphere in $\mathbb{R}^3$ that intersects a knot $K$ transversally in only two points. Let $a$ be any arc inside $S$ that connects the two points of intersection and denote by $U_1$ and $U_2$ the two components of $\mathbb{R}^3 \setminus S$. After smoothing the vertices, the diagrams $K_i = (K \cap U_i) \cup a$ define two new knots, called *factors* of $K$, cf. Figure 1.3.

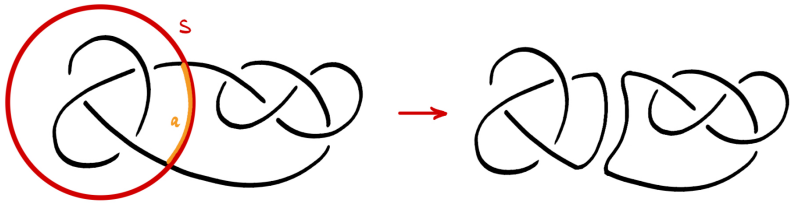



**Figure 1.3.** Factoring a knot $K$: The red circle represents $S$, the orange arc represents one possible choice for $a$.

**Definition 1.10.** A knot $K$ is called *prime* if it is not the unknot and $K$ has no non-trivial factors, i.e. every factor is either $K$ or the unknot.

The idea of factorising knots has many similarities to the prime factor decomposition in number theory. Indeed, a knot has a finite number of factors which are uniquely determined up to order. Proofs and a more thorough discussion can be found in [Cro04, Section 4.3]. There exists a partial inverse operation to factoring a knot.

**Definition 1.11.** Let $K$ and $K'$ be two knots inside the same ambient space $\mathbb{R}^3$ which are separated by a plane. We obtain a new knot as follows. Choose a rectangular disk $R$ in the ambient space whose boundary consist of four arcs $r_1, r_2, r_3, r_4$ such that $K \cap R = r_1$ and $K' \cap R = r_3$. Furthermore, the plane that separates $K$ and $K'$ intersects the boundary of $R$ only in one point of $r_2$ and one point of $r_4$ we define the *connected sum* $K\#K'$ as the knot whose image is obtained from $(K \setminus r_1) \cup r_2 \cup (K' \setminus r_3) \cup r_4$ after smoothing all vertices, cf. Figure 1.4.

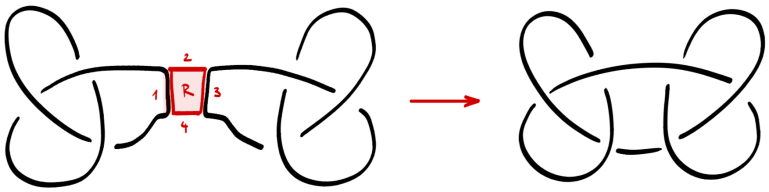



**Figure 1.4.** The connected sum of two knots. Creating a new knot from two given knots and a connecting rectangle $R$. The boundary edges of $R$ are labelled by 1 to 4.

So far all knots and links have been embedded in $\mathbb{R}^3$. It is however more common to consider knots and links inside the three sphere $S^3$. While this may sound unusual, the main motivation behind this fact is that $S^3$ is a compact space. In fact we simply think of $S^3$ as $\mathbb{R}^3 \cup \{\infty\}$. This identification can be made rigorous using stereographic projection. From this viewpoint it makes no difference if we think of a link in $\mathbb{R}^3$ or $S^3$, indeed any link in $S^3$ can be assumed to not contain the point $\infty$ and all the previous statements carry over to links in $S^3$. For a more thorough discussion see [Cro04, Section 2.1].

---

**Convention:** From now on, all knots and links are embedded in $S^3$.

---

## 1.1 Knot Invariants

Objects that are invariant under different representations of the same knot are called knot invariants. Some knot invariants have a very complex definition while others can be stated in one sentence. Often knot invariants that are easy to define are hard to calculate, conversely more complex invariants are sometimes easier to calculate and are thus important tools to classify knots. In Chapter 4 we define one such example, called *simply blocked grid homology*, cf. Definition 4.23. In this section we define two related invariants, to introduce the first we follow [Cro04, Section 2.6].

**Definition 1.12.** We call a connected orientable surface whose boundary is the link $L$ a *Seifert surface* for $L$.

**Theorem 1.13. ([Cro04, Section 5.2])** *There exists a Seifert surface for every link.*

*Proof.* The idea behind the proof is to construct a Seifert surface from a given link diagram. If the diagram is not oriented we first choose an orientation. Afterwards, the following local modification is performed at each crossing. Remove the crossing from the diagram, then reconnect the resulting ends in the only way as to not reintroduce a crossing so that the orientation is respected. After this modification, the diagram becomes a set of disjoint simple loops. We assign to each loop $l$ an index $h(l)$ that represents the number of loops that contain $l$. To construct the Seifert surface, we take for each loop $l$ a disc in $\mathbb{R}^3$ at height $h(l)$ such that the boundary of the disc represents $l$. We imagine the discs to be two-colored, red on one and blue on the other side. If the boundary of a disc is oriented clockwise from above, it is colored blue on the upper side, otherwise the upper side is colored red. To finish the surface, twisted bands are inserted at those points of the boundaries where originally a crossing was in the link diagram. Which kind of half twist is chosen, depends on the original type of crossing in order to respect the orientation. By construction, the two-coloring of the discs can be extended to the twisted bands. Thus, the resulting surface is indeed two-sided. If the link diagram was disconnected, the surface will also be disconnected. In this case, components can be connected by pipes. See Figure 1.5 for a Seifert surface of the trefoil knot. □

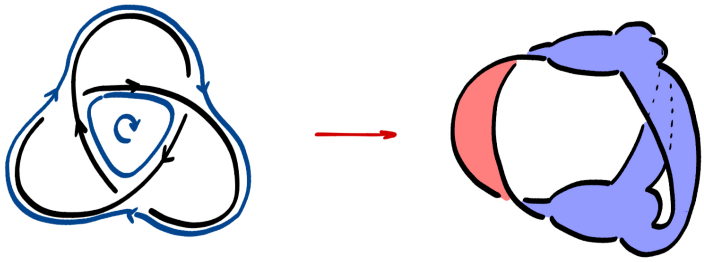


**Figure 1.5.** Creating a Seifert surface for the trefoil knot, as explained in Theorem 1.13.

Connected orientable surfaces have two intrinsic properties than can be used to distinguish them.

- The number of boundary components
- The genus

Indeed, those two properties already specify the surface.

**Theorem 1.14. ([DH10])** *If two connected orientable surfaces have the same number of boundary components and the same genus, then they are homeomorphic.*

The existence of Seifert surfaces thus allows us to associate an invariant to knots from properties that are a priori defined for surfaces.

**Definition 1.15.** For a knot $K$, the *Seifert genus* $g(K)$ is defined as the minimal genus of all Seifert surfaces of $K$.

As we are taking the minimum over all possible surfaces that span $K$, this automatically defines a knot invariant. On the downside, it is very hard to compute $g(K)$. In Chapter 4 we define *simply blocked grid homology* for a knot, cf. Definition 4.23, this knot invariant is computable for a large class of knots and it can be used to calculate the genus of a knot, we come back to this fact in Chapter 5.

The second knot invariant of interest to us has a rather involved definition which we will formulate later. Intuitively, we are interested in knots for which there exists a family of Seifert surfaces such that the following holds.

- For every element in $S^1$ there exist exactly one associated Seifert surface.
- The family of Seifert surfaces together with the knot cover all points of $S^3$.
- The interiors of any two Seifert surfaces do not intersect.

We call such knots *fibered*. As a first easy example we can look at the unknot.

**Example 1.16.** The unknot $U$ is fibered. To see this, recall that we visualize $S^3$ as $\mathbb{R}^3 \cup \{\infty\}$. The usual disk $D^2$ embedded in $\mathbb{R}^3 \subset S^3$ clearly is a Seifert surface for $U$. We can continuously deform $D^2$ as visualized in Figure 1.6. This deformation process describes a family of seifert surfaces of $U$ parametrized by $S^1$.

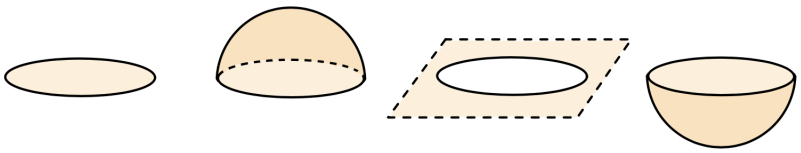


**Figure 1.6.** A family of Seifert surfaces of the unknot parametrized by $t \in S^1$. At time $t = \pi$ (third step in figure) the Seifert surface looks like a punctured plane when visualized in $\mathbb{R}^3$. In $S^3$ this also describes a disk.

As a less trivial example we next look at a whole class of knots.

**Definition 1.17.** A knot $T$ that lives on an unknotted torus in $S^3$ is called a *torus knot*. More precisely, a *p q torus knot* $T_{p,q}$ is defined by the following set of points that lie on a standard torus in $S^3 \subset \mathbb{C}^2$

$$\{(z_1, z_2) \in \mathbb{C}^2 : z_1\overline{z_1} + z_2\overline{z_2} = 1, z_1^p + z_2^q = 0\} \subset S^3.$$

The simplest example of a torus knot is the trefoil knot $T_{2,3}$, cf. Figure 1.7. We want to show that the trefoil knot is fibered, here we follow [ser].

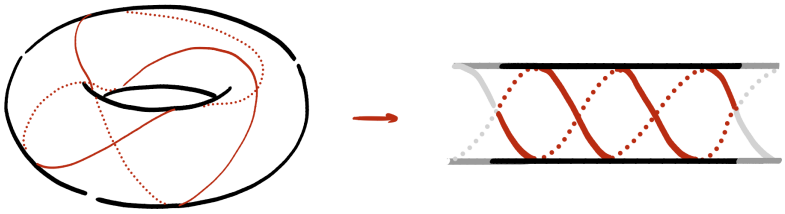


**Figure 1.7.** The trefoil knot can be drawn on a torus. When cutting the torus along a meridian, the trefoil looks like a double helix (right).

**Example 1.18.** The trefoil knot $T_{2,3}$ is fibered. We first look for a family of surfaces that cover all of the points inside the torus on which $T_{2,3}$ lies. To cover the

rest of $S^3$ we then use a similar approach as for the unknot. To this end, it is easiest to cut the torus open and view it as a tube, cf. Figure 1.7. One surface that lives inside the torus is obtained by connecting the double helix that represents the torus by twisted bands. This surface has as its boundary the trefoil knot and two additional longitudes of the torus on the top and bottom. We can rotate this surface along its boundary longitudes as depicted in Figure 1.8. This rotating motion enables us to cover all points inside the torus. To complete the surface into an actual Seifert surface we attach domes at both boundary longitudes, cf. Figure 1.9. When rotating the surface, these domes deform as in the example for the unknot, thus covering all points outside of the torus.

**Figure 1.8.** Rotating the inside of a Seifert surface for the trefoil knot, as explained in Example 1.18

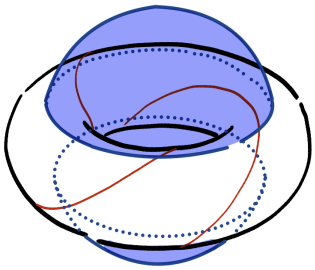
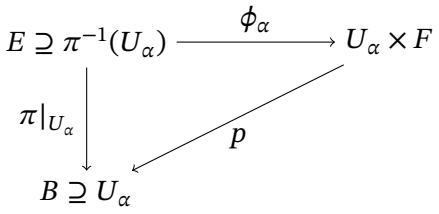


**Figure 1.9.** Obtaining a Seifert surface for the trefoil by attaching domes outside of the torus, as explained in Example 1.18.

A more rigorous but less visual proof can only be formulated after stating the precise definition of fibered knots. For this we need to recall what a locally trivial fibration is.

**Definition 1.19.** Let $E, B$ and $F$ be topological spaces, a *locally trivial fibration* consists of a continuous map $\pi : E \to B$ and an open cover $\{U_\alpha\}$ of $B$ such that for every open set $U_\alpha \subset B$ there exists a homeomorphism $\phi_\alpha : \pi^{-1}(U_\alpha) \to U_\alpha \times F$.

Furthermore, if $p$ is the projection $U_\alpha \times F \to U_\alpha$ then $\phi_\alpha$ should satisfy $p \circ \phi_\alpha^{-1} = \pi|_{U_\alpha}$.




We call $F$ the *fiber*, $B$ the *base space* and $E$ the *total space*.

Intuitively the total space $E$ of a locally trivial fibration $\pi$ with base space $B$ and fiber $F$ is a topological space that locally looks like the product space $B \times F$. Furthermore, for all points $p \in B$, the fiber $\pi^{-1}(p)$ is homeomorphic to $F$.

**Definition 1.20.** A knot $K$ is *fibered* if the complement $S^3 \setminus K$ is the total space of a locally trivial fibration $\pi$ with base space $S^1$. Additionally, for all $t \in S^1$ the closure of the fiber $\overline{\pi^{-1}(t)} = S_t$ should be a Seifert surface for $K$ such that $S_t \cap S_t' = K$ for $t \neq t'$.

The isotopy extension theorem guarantees that if $K$ and $K'$ are two different smooth embeddings that represent the same knot, the complements $S^3 \setminus K$ and $S^3 \setminus K'$ are diffeomorphic. Therefore the notion of fibered knot does not depend on the chosen smooth embedding. See also [Sav11] for a more thorough discussion of fibered knots.

A recipe to rigorously prove that any torus knot $T_{p,q}$ is fibered works by proving that the map $f : S^3 \setminus K \to S^1$ defined by $f(z_1, z_2) = (z_1^p + z_2^q)/\|z_1^p + z_2^q\|$ is a locally trivial fibration over $S^1$ that satisfies all the necessary conditions. Even though it is possible to directly show that torus knots are fibered, it is in general very hard to see if an arbitrary knot is fibered. Just as for the genus, we can detect fiberedness for a large class of knots using simply blocked grid homology from Definition 4.23. We come back to this fact in Chapter 5.

# 2 Grid Diagrams

In this chapter we introduce a special way of describing knots and links by so called grid diagrams, these object were first defined by Cromwell, see [Cro95]. They are of central importance for the rest of the text. We closely follow Chapter 3 from the book *Grid Homology for Knots and Links* [OSS15].

**Definition 2.1.** A *planar grid diagram* $\mathbb{G}$ of *grid size* $n$ is a square consisting of $n$ rows and $n$ columns of smaller squares such that in each row and in each column of $\mathbb{G}$ exactly one of the smaller squares is marked with an $X$ and one other smaller square is marked with an $O$. We denote the set of $X$-markings by $\mathbb{X}$ and the set of $O$-markings by $\mathbb{O}$.

**Remark 2.2.** A planar grid diagram always defines an oriented link. Connect the $X$-markings to the $O$-markings in each column by a vertical line. Then, connect the $O$-markings to the $X$-markings in each row by a horizontal line. If the horizontal line crosses a vertical line, the horizontal line passes under the vertical line. The resulting link is oriented such that we always travel from $X$- to $O$-markings on vertical lines, cf. Figure 2.1.

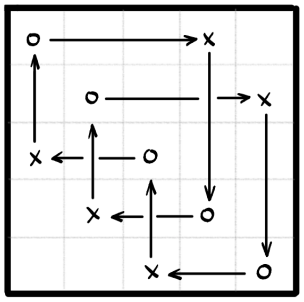


**Figure 2.1.** A planar grid diagram of the trefoil knot.

Not only does every planar grid diagram define an oriented link, it is also true that every oriented link has such a representation.

**Proposition 2.3. ([Cro95, p. 42])** *Every oriented link can be represented by a planar grid diagram.*

*Proof.* We first represent the oriented link by a piecewise linear projection with the property that it only consists of horizontal and vertical segments. A priori, this projection does not necessarily satisfy the condition that at each crossing the horizontal segment lies under the vertical segment. To ensure this we can apply the modification visualized in Figure 2.2.[1] We then stretch the resulting projection so that no two segments lie on the same line. This specifies a planar grid diagram that represents the oriented link by marking the vertices by $X$'s and $O$'s with vertical segments pointing from $X$ to $O$ and horizontal segments pointing from $O$ to $X$. □

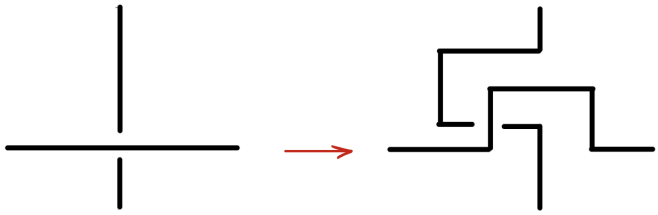


**Figure 2.2.** Turning a horizontal overcrossing into a vertical overcrossing.

## 2.1 Grid Moves

A representation of an oriented link by a planar grid diagram is clearly not unique. We now define moves - i.e. modifications of planar grid diagrams - such that the underlying oriented link is not changed. Similar to Reidemeister moves we want to define a set of moves so that any two planar grid diagrams that represent the same oriented link only differ by a finite sequence of such moves. The first type arises by interchanging two neighbouring columns/rows of a planar grid diagram.

If we think of $\mathbb{G}$ as embedded in $\mathbb{R}^2$, then for any column/row of $\mathbb{G}$ the vertical/horizontal segment that connects the $X$- and $O$-markings as described in Remark 2.2 describes an open interval of real numbers. If we interchange two neighbouring columns/rows we have to distinguish three cases.

- The open intervals are disjoint
- One of the open intervals is fully contained in the other
- The open intervals intersect non-trivially

[1] Like all other illustrations, Figure 2.2 is hand-drawn. In this case, [OSS15, Figure 3.2] served as a reference.

For a move of type 1 or 2 it is easy to see that the resulting grid diagram represents the same oriented link. A move of type 3 however does not preserve the link, indeed it is not hard to see that a move of type 3 corresponds to a crossing change.

**Definition 2.4.** Two planar grid diagrams $\mathbb{G}$ and $\mathbb{G}'$ are said to differ by a *column commutation* if $\mathbb{G}'$ is obtained from $\mathbb{G}$ by interchanging two neighbouring columns such that the two open intervals that correspond to the vertical segments are either disjoint or one is fully contained in the other[2], i.e. the column change corresponds to a move of type 1 or 2 from above, cf. Figure 2.3. *Row commutations* are defined analogously.

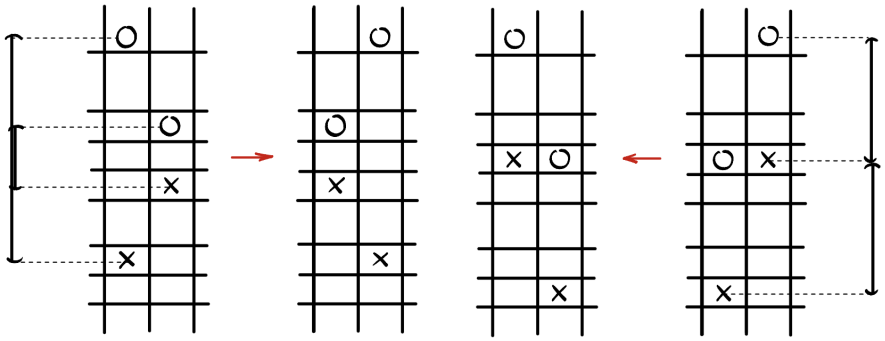


**Figure 2.3.** Two examples of column commutations.

There clearly exist planar grid diagrams of different grid size that represent the same oriented link. Commutation moves alone are thus not enough. We now introduce a second type of move that changes the grid size.

**Definition 2.5.** Let $\mathbb{G}$ be a planar grid diagram with grid size $n$. The following modification of $\mathbb{G}$ is called a *stabilization*. Choose any marked square of $\mathbb{G}$ and remove its entry as well as the entries of the marked squares in the same column and in the same row. Now split the row and column of the chosen square into two. There are four different ways of filling in $X$- and $O$-markings into the new grid so that it again becomes a planar grid diagram which is now of grid size $(n+1)$. Any grid diagram $\mathbb{G}'$ obtained this way is called *a stabilization of* $\mathbb{G}$ and conversely, $\mathbb{G}$ is called a *destabilization* of $\mathbb{G}'$.

For any stabilization the chosen square gets subdivided into four squares of which exactly three contain a marking. The fourth empty square can be in all four possible positions, northwest NW, northeast NE, southeast SE and southwest SW. If the chosen square originally contained an $X$-marking then the four possible stabilizations are denoted by $X$-NW, $X$-NE, $X$-SE and $X$-SW, cf. Figure 2.4.[3] Similarly,

[2] This definition differs slightly from the source text, cf. Remark 2.6.
[3] For drawing Figure 2.4, [OSS15, Figure 3.8] served as a reference.

if the chosen square contained an $O$-marking, the four possible stabilizations are denoted by $O$-NW, $O$-NE, $O$-SE and $O$-SW. It is easy to see that all stabilizations preserve the underlying link.

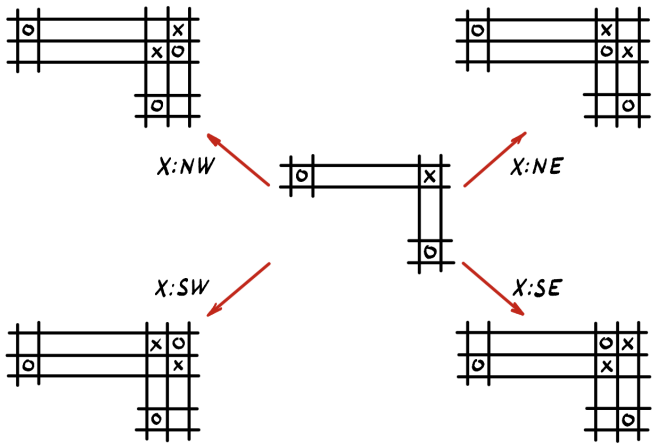



**Figure 2.4.** The four possible stabilizations at an $X$-marking. Here the stabilization of type $X$-SW corresponds to a Reidemeister move R1, the other stabilizations correspond to planar isotopies.

**Remark 2.6.** According to Definition 2.4 it may happen that a commutation is performed on two neighbouring columns/rows that have an $X$- and an $O$-marking in the same row/column. This convention differs from [OSS15], there the authors consider moves of this type separately and call them *switches*. It is an easy exercise to show that a switch can be expressed as a sequence of commutations, stabilizations and destabilizations. We do not need to distinguish between commutations and switches in this text.

**Definition 2.7.** Commutations, stabilizations and destabilizations are called *grid moves*.

The following theorem, due to Cromwell, proves that grid moves achieve what we want.

**Theorem 2.8. ([Cro95, p. 45])** *Two planar grid diagrams represent the same link if and only if there is a finite sequence of grid moves that transforms one into the other.*

## 2.2 Toroidal Grid Diagrams

To define simply blocked grid homology in Chapter 4 we have to transfer grid diagrams to the torus. This originates from the fact that simply blocked grid homology is a combinatorial version of a knot invariant that is defined from Heegaard

diagrams of manifolds. We only focus on the combinatorial theory in this text. Nevertheless, transferring grid diagrams to the torus is useful in its own right.

**Definition 2.9.** We can transfer planar grid diagrams to the torus by identifying the top and bottom boundary edges as well as the left and right boundary edges. When we do this, the vertical and horizontal grid lines become vertical and horizontal circles, cf. Figure 2.5. We call these objects *toroidal grid diagrams*.

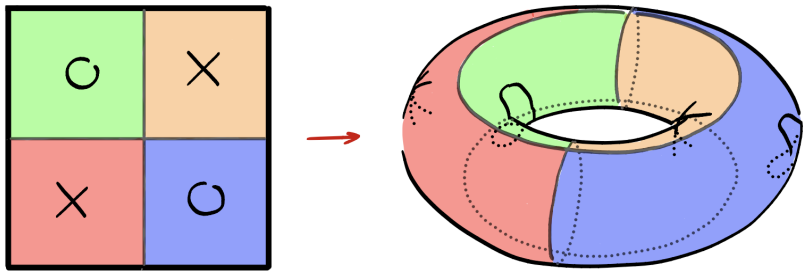


**Figure 2.5.** Transferring a planar grid diagram of the unknot to the torus.

**Definition 2.10.** We can also go from a toroidal grid diagram to a planar grid diagram by choosing one vertical circle and one horizontal circle as the boundary edges of the planar grid diagram. A planar grid diagram obtained this way is called a *planar realization* of the toroidal grid diagram.

A planar grid diagram of grid size $n$ consists of $(n+1)$ vertical and $(n+1)$ horizontal grid lines. Due to the identification on the boundary, the corresponding toroidal grid diagram consist of $n$ vertical and $n$ horizontal circles. The grid size of the toroidal grid diagram is thus given by the number of horizontal or vertical circles. It also follows that a toroidal grid diagram of grid size $n$ determines $n^2$ different planar realizations.

**Proposition 2.11.** *All planar realizations of a toroidal grid diagram represent the same link.*

*Proof.* If $\mathbb{G}$ and $\mathbb{G}'$ are two planar realizations of a toroidal grid diagram we see that $\mathbb{G}$ can be obtained from $\mathbb{G}'$ by a sequence of cyclic permutations of the rows and columns of the grid. It is possible to express any cyclic permutation as a sequence of commutations, stabilizations and destabilizations, cf. [OSS15, Lemma 3.2.4]. By Theorem 2.8 it follows that $\mathbb{G}$ and $\mathbb{G}'$ represent the same link.

Intuitively this statement is also clear. When we think of two different planar realizations of a toroidal grid diagram as embedded in the torus we see that the corresponding links at most differ by an isotopy that moves some arcs from one part of the torus to the "opposite" part. For example, the four different planar realizations of the toroidal grid diagram of the unknot given in Figure 2.5 are visualized in Figure 2.6. □

By introducing toroidal grid diagrams we have thus managed to collect $n^2$ different planar grid diagrams of size $n$ of the same link into one object.

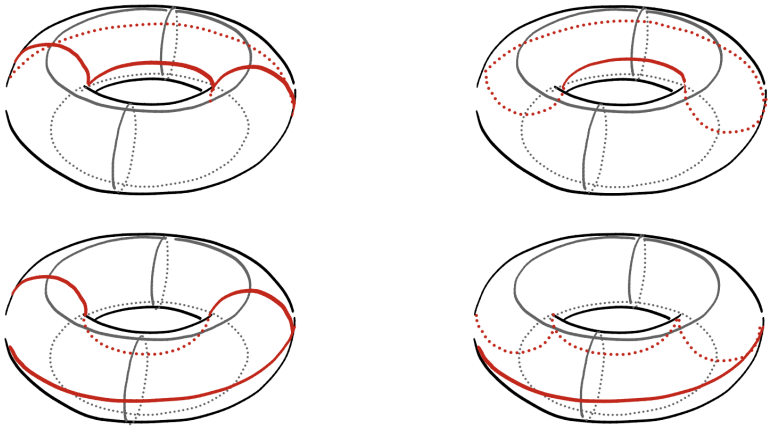


**Figure 2.6.** The four different planar realizations of the toroidal grid diagram from Figure 2.5 embedded in the torus.

---

**Convention:** From now on, we call toroidal grid diagrams simply grid diagrams. When drawing a planar realization of a grid diagram we identify the top and bottom boundary as well as the left and right boundary. We also call the vertical and horizontal grid lines of a planar realization horizontal and vertical circles.

---

This convention makes clear that we always think of the grid diagram as living on the torus, even when we draw our pictures on the plane. Furthermore, two grid diagrams are said to differ by a commutation/stabilzation if they have planar realizations that differ by a commutation/stabilization. Finally we also mention the following fact here which are needed in the final parts of Section 4.3.

**Proposition 2.12. ([OSS15, Cor. 3.2.3])** *Any stabilization of a grid diagram can be achieved as a stabilization of type X-SW combined with a sequence of commutations.*

Proving this is an easy exercise that makes use of the fact that a stabilization creates a segment of length one on the link. Such a segment commutes with all rows/-columns.

# 3 Homological Algebra

We want to construct a knot invariant using grid diagrams which detects the Seifert genus and fibered knots. Due to the combinatorial nature of grid diagrams such an invariant is comparatively easy to calculate and is thus a powerful tool to investigate topological properties of knots. The construction uses several tools from homological algebra. Here we recall the most important notions and facts that we need going forward. We only consider the field $\mathbb{Z}/2\mathbb{Z}$ which we denote by $\mathbb{K}$ and we denote the polynomial ring $\mathbb{K}[V_1, \dots, V_n]$ by $\mathcal{R}$. We follow [OSS15, Appendix A].

**Definition 3.1.** An $\mathcal{R}$-module $C$ is called a *chain complex*, if there exists a map $\partial : C \to C$ of $\mathcal{R}$-modules such that $\partial^2 = 0$, i.e. $\operatorname{im}\partial \subset \ker\partial$.

**Definition 3.2.** A *subcomplex* of a chain complex $(C, \partial)$ is an $\mathcal{R}$-submodule $C' \subset C$ such that $\partial(C') \subset C'$. A *quotient complex of $C$ by the subcomplex $C'$* is the quotient module $C/C'$ with the differential induced from $C$, it is a well defined chain complex since $C'$ is a subcomplex.

**Definition 3.3.** The *homology* $H(C, \partial)$ of a chain complex is the quotient $\mathcal{R}$ - module $\ker\partial / \operatorname{im}\partial$.

Most chain complexes are $\mathbb{Z}$-graded and the differential map reduces this grading by one, i.e. $C$ decomposes into a direct sum of submodules $C = \bigoplus_{d\in\mathbb{Z}} C_d$ and the differential map restricts to $\partial|_{C_d} : C_d \to C_{d-1}$. Due to $\operatorname{im}(\partial) \cap C_d \subset \ker(\partial) \cap C_d$ the homology inherits a grading

$$H_d(C) = \frac{\ker(\partial) \cap C_d}{\operatorname{im}(\partial) \cap C_d}.$$

The chain complexes we define are additionally equipped with a bigrading $C = \bigoplus_{d,s\in\mathbb{Z}} C_{d,s}$ such that:

- The differential restricts to $\partial|_{C_{d,s}} : C_{d,s} \to C_{d-1,s}$.
- The maps $V_i : C \to C$, which act by multiplying elements with the variable $V_i \in \mathcal{R}$, restrict to $V_i|_{C_{d,s}} : C_{d,s} \to C_{d-2,s-1}$.

**Definition 3.4.** A *bigraded $\mathcal{R}$-module $M$* is an $\mathcal{R}$-module that has a bigrading as a $\mathbb{K}$ vector space $M = \bigoplus_{d,s\in\mathbb{Z}} M_{d,s}$ and the multiplication maps $V_i$ for $1 \leq i \leq n$ restrict to $V_i|_{M_{d,s}} : M_{d,s} \to M_{d-2,s-1}$. A morphism of $\mathcal{R}$-modules $f : M \to N$ is called *homogeneous of degree* $(k,t)$ if it restricts to $f|_{M_{d,s}} : M_{d,s} \to N_{d+k,s+t}$. If $f$ is homogeneous of degree $(0,0)$ we call it a *bigraded morphism of $\mathcal{R}$-modules*.

**Definition 3.5.** A *bigraded chain complex* $(C,\partial)$ *over* $\mathcal{R}$ is a bigraded $\mathcal{R}$-module together with a morphism of $\mathcal{R}$-modules $\partial : C \to C$ which is homogeneous of degree $(-1,0)$.

The homology of a bigraded chain complex inherits a bigrading

$$H_{d,s}(C) = \frac{\ker(\partial) \cap C_{d,s}}{\operatorname{im}(\partial) \cap C_{d,s}}.$$

**Definition 3.6.** A *chain map* between bigraded chain complexes $f : (C,\partial) \to (C',\partial')$ over $\mathcal{R}$ is a morphism of $\mathcal{R}$-modules such that $\partial' \circ f = f \circ \partial$. If $f$ is additionally a bigraded morphism we call it a *bigraded chain map*. An *isomorphism of bigraded chain complexes* is a bigraded chain map $f : (C,\partial) \to (C',\partial')$ such that there exists another bigraded chain map $g : (C',\partial') \to (C,\partial)$ satisfying $f \circ g = \mathrm{id}_{C'}, g \circ f = \mathrm{id}_C$. In this case we write $(C,\partial) \cong (C',\partial')$.

It is easy to check that a chain map $f : (C,\partial) \to (C',\partial')$ induces a well defined map on homology $H(f) : H(C,\partial) \to H(C',\partial')$. As $\partial$ is an $\mathcal{R}$-module morphism it follows that $V_i \circ \partial = \partial \circ V_i$. Thus the multiplication maps $V_i : C \to C$ are chain maps. Given two bigraded chain complexes $(C,\partial), (C',\partial')$ and a chain map $f : (C,\partial) \to (C',\partial')$ the image $\operatorname{im}(f)$ is a subcomplex of $C'$, the quotient complex $C'/\operatorname{im}(f)$ is therefore well defined and it is easily checked that $C'/\operatorname{im}(f)$ is again a bigraded chain complex.

---

**Convention:** To simplify the notation, we denote the quotient complex $C/\operatorname{im}(V_i)$ by $C/V_i$.

---

**Definition 3.7.** Two chain maps $f, g : (C,\partial_C) \to (D,\partial_D)$ of bigraded chain complexes over $\mathcal{R}$ that are homogeneous of degree $(k,t)$ are called *chain homotopic* if there exists a morphism of bigraded $\mathcal{R}$-modules $p : C \to D$ that is homogeneous of degree $(k+1,t)$ such that $f - g = \partial_D \circ p + p \circ \partial_C$.

A direct calculation shows that chain homotopic maps induce the same map on homology.

**Definition 3.8.** A chain map $f : C \to D$ between chain complexes $C, D$ is called a *chain homotopy equivalence* if there exists another chain map $g : D \to C$ such that $g \circ f$ is chain homotopic to $\mathrm{id}_C$ and $f \circ g$ is chain homotopic to $\mathrm{id}_D$.

A chain homotopy equivalence $f : C \to D$ induces an isomorphism of the homology groups $H(C) \cong H(D)$. If the chain homotopy preserves the bigrading, then the isomorphism on the homology groups also preserves the bigrading.

**Proposition 3.9. ([OSS15, Prop. 4.5.6])** *Let $C, D$ be bigraded chain complexes over $\mathcal{R} = \mathbb{K}[V_1, \dots, V_n]$. A chain map $f : C \to D$, homogeneous of degree $(k, t)$, induces a chain map on the quotient complex $[f] : C/V_i \to D/V_i$ which is again homogeneous of degree $(k, t)$. If $g$ is also a chain map of the same degree as $f$ and there exists a chain homotopy between them, then there exists an induced chain homotopy between $[f]$ and $[g]$.*

*Proof.* The differential $\partial_C$ on $C$ induces the differential $[\partial_C]$ on $C/V_i$. For $v \in C$ denote by $[v]$ the corresponding element in $C/V_i$, then $[\partial_C]([v]) = [\partial_C(v)]$ is well defined since $\partial_C$ is an $\mathcal{R}$-module morphism. The chain maps $f, g$ induce in the same way well defined chain maps $[f], [g] : C/V_i \to D/V_i$. If $f, g$ are homogeneous of degree $(k, t)$ then it is easy to check that $[f], [g]$ are homogeneous of the same degree. Finally, if $h : C \to D$ is the chain homotopy between $f$ and $g$, then the induced morphism $[h]$ inherits the property $[\partial_D] \circ [h] + [h] \circ [\partial_C] = [f] - [g]$ from $h$. □

**Remark 3.10.** The previous proposition shows in particular, that the chain maps $V_j : C \to C$ induce chain maps of the same degree on the quotient complex $C/V_i \to C/V_i$. If $i = j$ then the morphism induced by $V_i$ on $C/V_i$ becomes the zero morphism which can be regarded as homogeneous of any degree.

**Theorem 3.11. ([OSS15, Lemma A.2.1.])** *Given three bigraded chain complexes $C, D, E$ over $\mathcal{R}$ as well as a chain map $f : C \to D$ of degree $(k, t)$ and a chain map $g : D \to E$ of degree $(0, 0)$ such that there exists a short exact sequence*

$$0 \to C \overset{f}{\hookrightarrow} D \overset{g}{\twoheadrightarrow} E \to 0.$$

*Then there exists an $\mathcal{R}$-module morphism $\delta : H(E) \to H(C)$, homogeneous of degree $(-k-1, -t)$ that induces a long exact sequence in homology.*

$$\cdots \to H_{d-k,s-t}(C) \xrightarrow{H(f)} H_{d,s}(D) \xrightarrow{H(g)} H_{d,s}(E) \xrightarrow{\delta} H_{d-k-1,s-t}(C) \to \dots$$

This is a well known theorem in homological algebra, the proof uses a standard diagram chase. We also do not have to require $g$ to be of degree $(0, 0)$, but this is the case that we need.

# 4 Grid Homology

In this chapter we define two homology theories for knots using the tools and definitions from the previous two chapters. We follow [OSS15, Chapter 4 and 5].

## 4.1 Grid States

As a first step, we introduce objects on grid diagrams that are used to define the generators of the chain complex.

**Definition 4.1.** Let $\mathbb{G}$ be a grid diagram of grid size $n$. A *grid state* $x$ of $\mathbb{G}$ is a one-to-one correspondence between the vertical and horizontal circles of $\mathbb{G}$. A grid state can be visualized by $n$ lattice points $\{x_1, \dots, x_n\}$ on the torus where each horizontal and vertical circle contains exactly one of the points, cf. Figure 4.1. We denote the set of grid states of $\mathbb{G}$ by $S(\mathbb{G})$.

While we do not need to choose planar realizations of grid diagrams in this chapter, the following definition is convenient later.

**Definition 4.2.** If we fix a planar realization of a grid diagram we can label the vertical circles from left to right, and the horizontal circles from top to bottom. A grid state $x$ can be specified on a planar realization by a permutation $\sigma_x$ such that the point of $x$ on the $i$-th vertical circle lies at the intersection with the $\sigma_x(i)$-th horizontal circle. We call $\sigma_x$ the *grid state permutation* of $x$.

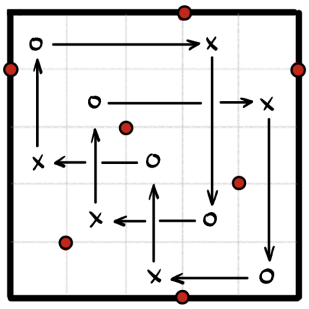


**Figure 4.1.** A planar realization of a grid diagram with red dots representing one possible grid state. Recall that we are identifying top and bottom edges as well as left and right edges.

The differential on the chain complex counts so called rectangles that connect one grid state to another. The next definition makes this concept precise.

**Definition 4.3.** Given $x, y \in S(\mathbb{G})$, a *rectangle* $r$ connecting $x$ and $y$ is an embedded rectangle in the torus with boundary edges inside the horizontal and vertical circles of the grid such that the points of $x$ and $y$ overlap in $n-2$ points and the four remaining points are the four corners of $r$. The orientation on the torus induces an orientation on $r$, we say that $r$ goes from $x$ to $y$ if the horizontal edges of the rectangle go from points in $x$ to points in $y$. The set of rectangles from $x$ to $y$ is denoted by $\mathrm{Rect}(x, y)$.

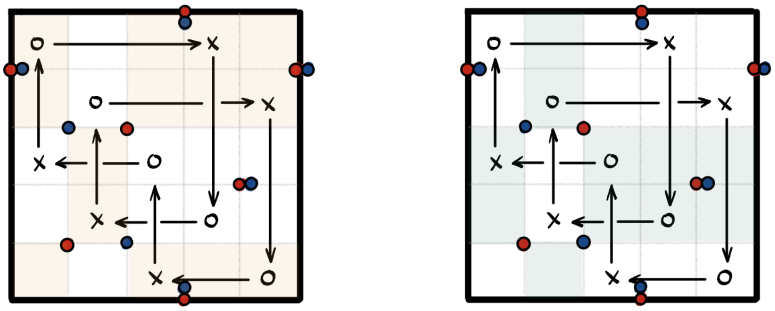


**Figure 4.2.** A grid diagram with two grid states $x$ and $y$ visualized in red and blue.
*Left:* The orange areas represent two elements of $\mathrm{Rect}(x, y)$, one rectangle of size $1 \times 2$ and one bigger rectangle of size $4 \times 3$. In this planar realization the bigger rectangle is split into four pieces.
*Right:* Similarly the blue areas represent two elements of $\mathrm{Rect}(y, x)$ of size $1 \times 3$ and $4 \times 2$.

For any two grid states $x$ and $y$, the set $\mathrm{Rect}(x,y)$ is either empty or consists of two elements, as can be seen in Figure 4.2.

**Remark 4.4.** When speaking of a rectangle on a grid diagram, we also include the data of the grid states $x$ and $y$. Thus, if another grid state $x'$ exists that is connected by a rectangle to the grid state $y$ it is always viewed as a different rectangle, even if the geometric subsets on the torus are the same.

By definition, if $r \in \mathrm{Rect}(x,y)$ includes points of the grid state $x$ in its interior then these points also belong to $y$, i.e. $x \cap \mathrm{Int}(r) = y \cap \mathrm{Int}(r)$.

**Definition 4.5.** If $r \in \mathrm{Rect}(x,y)$ satisfies $x \cap \mathrm{Int}(r) = y \cap \mathrm{Int}(r) = \emptyset$ then $r$ is called an *empty rectangle*. We denote the subset of empty rectangles from $x$ to $y$ by $\mathrm{Rect}^\circ(x,y)$.

We want to define two integer-valued functions on grid states that induce a bigraded structure of the chain complex. To this end we first define winding numbers.

**Definition 4.6.** Let $\mathbb{G}$ be a grid diagram of the knot $K$, let $p$ be a point on $\mathbb{G}$ that does not lie on the knot projection. The *winding number* $w(p)$ is an integer that describes how often the knot projection travels around $p$ counterclockwise. After choosing a planar realization of $\mathbb{G}$, the winding number at $p$ can be calculated by drawing a ray $\gamma$ emerging out of $p$ and then counting the number of intersections of $\gamma$ with the knot projection. An intersection where the knot projection travels counterclockwise increases $w(p)$ by 1, clockwise intersections decrease $w(p)$ by 1, cf Figure 4.3.

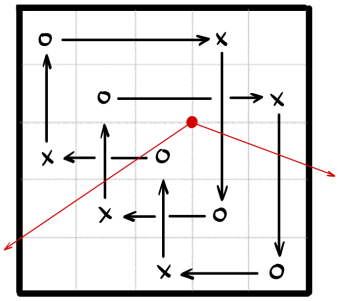


**Figure 4.3.** For this grid diagram, the winding number at the red point $p$ is -2. To see this, one can choose any ray emerging out of $p$ and count its signed intersection with the knot projection, as explained in Definition 4.6. Here two possible rays are drawn, both intersect the knot projection twice in clockwise direction.

With this we can define the function that induces the first grading.

**Definition 4.7.** Let $\mathbb{G}$ be a grid diagram of size $n$, then there are $2n$ squares on the grid that are marked with either an $X$ or an $O$. Counting all four cornerpoints of all $2n$ squares gives a tuple $(p_1, \dots, p_{8n})$ of $8n$ lattice points on the grid where it is possible that some points are counted more than once. The *Alexander function* on a grid state $x = \{x_1, \dots, x_n\}$ is given by the equation

$$A(x) = -\sum_i w(x_i) + \frac{1}{8}\sum_{j=1}^{8n} w(p_j) - \left(\frac{n-1}{2}\right). \tag{4.1}$$

**Example 4.8.** To calculate the Alexander grading of the grid state given in Figure 4.1 we first calculate the winding numbers of the points of the grid state. Counting the vertical circles of the grid diagram from left to right, only the red points on the third and fifth circle lie inside the knot projection. For the point on the third circle the winding number is -2, for the point on the fifth circle the winding number is -1. As all other winding numbers are zero, we have $-\sum_i w(x_i) = 3$. The grid size is $n = 5$, we thus have $-(n-1)/2 = -2$ and it only remains to calculate $\frac{1}{8}\sum_{j=1}^{8n} w(p_j)$. Starting at the square on the top left, containing an $O$-marking, only the bottom right cornerpoint is inside the knot projection and has a winding number of -1. I.e. the value associated to this square is -1. We have to calculate this value for all squares containing either an $X$- or $O$-marking and then divide the sum by 8. For example, the value associated to the square in the fourth column containing an $O$ is -5, three cornerpoints have a winding number of -1, the upper left cornerpoint has a winding number of -2. Continuing like this, we see that $\sum_{j=1}^{8n} w(p_j) = -24$. Putting everything together it follows that the Alexander grading of this grid state is -2.

For the Alexander function to define a grading it has to be of the form $S(\mathbb{G}) \to \mathbb{Z}$. Indeed, this is the case.

**Proposition 4.9. ([OSS15, Prop. 4.3.3])** *The Alexander function is interger valued.*

The following proposition guarantees the existence of the function that induces the second grading.

**Proposition 4.10. ([OSS15, Prop. 4.3.1])** *For any grid diagram $\mathbb{G}$ there exists a function $M : S(\mathbb{G}) \to \mathbb{Z}$ that is uniquely defined by the following two conditions.*

- *$M(x^{NWO}) = 0$ where $x^{NWO}$ denotes the grid state with points on the north-west corners of squares on the grid that are marked with an O.*
- *Let $x, y$ be connected by $r \in \mathrm{Rect}(x, y)$. Denote by $\#(r \cap \mathbb{O})$ the number of O-markings inside $r$, by $\#(x \cap Int(r))$ the number of points of $x$ in the interior of $r$. Then the function satisfies*

$$M(x) - M(y) = 1 - 2\#(r \cap \mathbb{O}) + 2\#(x \cap Int(r)). \tag{4.2}$$

**Definition 4.11.** We call the function $M : S(\mathbb{G}) \to \mathbb{Z}$ from Proposition 4.10 the *Maslov function*.

Note that $M$ is independent of the placement of the $X$-markings on the grid.

## 4.2 Fully Blocked Grid Homology

We can now define the first version of grid homology. Again we denote the field $\mathbb{Z}/2\mathbb{Z}$ by $\mathbb{K}$.

**Definition 4.12.** Given a grid diagram $\mathbb{G}$, we define the *fully blocked grid chain complex* $\widetilde{\mathrm{GC}}(\mathbb{G})$ as a $\mathbb{K}$ vector space whose basis are the elements of $S(\mathbb{G})$. The differential $\tilde{\partial}_{\mathbb{O},\mathbb{X}}$ is defined on basis elements $x$ as

$$\tilde{\partial}_{\mathbb{O},\mathbb{X}}(x) = \sum_{y \in S(\mathbb{G})} \#\{r \in \mathrm{Rect}^{\circ}(x,y) : r \cap \mathbb{O} = r \cap \mathbb{X} = \emptyset\} \cdot y.$$

As $\widetilde{\mathrm{GC}}(\mathbb{G})$ is a $\mathbb{K}$ vector space, $\#\{\cdot\}$ is understood as the number of elements in the set modulo 2. The subscript $_{\mathbb{O},\mathbb{X}}$ indicates that we count only those empty rectangles that are disjoint from $\mathbb{O}$ and $\mathbb{X}$. To show that this map really defines a differential of a chain complex we have to show $\tilde{\partial}^2_{\mathbb{O},\mathbb{X}} = 0$. Here we give a sketch of this proof, a more general fact is proven in [OSS15, Lemma 4.6.7].

**Proposition 4.13.** *The differential* $\tilde{\partial}_{\mathbb{O},\mathbb{X}} : \widetilde{\mathrm{GC}}(\mathbb{G}) \to \widetilde{\mathrm{GC}}(\mathbb{G})$ *satisfies* $\tilde{\partial}^2_{\mathbb{O},\mathbb{X}} = 0$.

*Proof.* A grid state $z \in S(\mathbb{G})$ only appears as a summand in $\tilde{\partial}^2_{\mathbb{O},\mathbb{X}}(x)$ if there exists a grid state $y$ as well as rectangles $r_1 \in \mathrm{Rect}^{\circ}(x,y), r_2 \in \mathrm{Rect}^{\circ}(y,z)$ such that $r_i \cap \mathbb{O} = r_i \cap \mathbb{X} = \emptyset$ for $i \in \{1,2\}$. Thus, $x$ and $z$ can differ by at most four points on the grid diagram. If $r_2$ connects $y$ to $z$ on two points of $y$ that also belong to $x$ then $x$ and $z$ differ by exactly four points. If one of the corner points of $r_2$ is a point of $y$ that does not belong to $x$ then only two things can happen. Either the diagonal corner point belongs to $x$ or it is exactly the only other point of $y$ that does not belong to $x$, in this case we have $x = z$. Hence we only need to consider three cases.

- The grid states $x$ and $z$ differ by 4 points.
- The grid states $x$ and $z$ differ by 3 points.
- The grid states coincide, $x = z$.

In cases one and two there exist exactly two different grid states $y, y'$ which connect $x$ to $z$, as seen in Figure 4.4.[1] As we are counting modulo 2 it follows that summands of this form vanish. In the third case the two rectangles $r_1, r_2$ share two edges and thus make up an annulus on the torus, a contradiction to the assumption $r_i \cap \mathbb{O} = r_i \cap \mathbb{X} = \emptyset$ for $i \in \{1, 2\}$. □

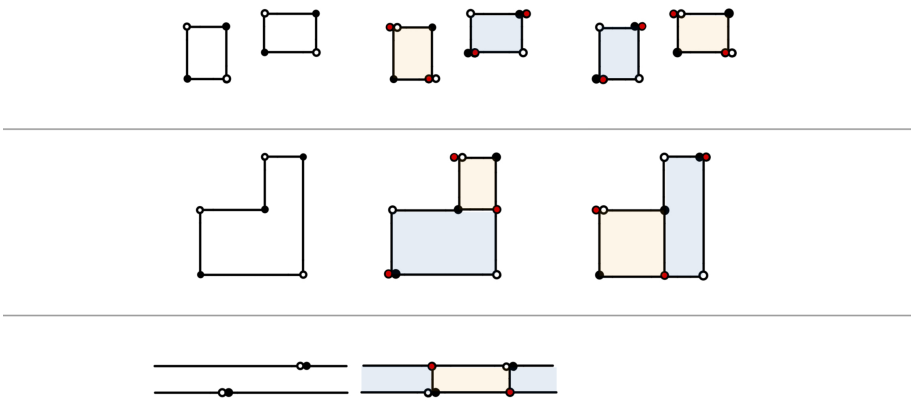


**Figure 4.4.** The three cases of Proposition 4.13 from top to bottom. The black dots represent the initial grid state $x$, the white dots represent the final grid state $z$ and red dots represent possible choices for intermediate grid states $y, y'$. The orange rectangle goes from $x$ to $y$ or $y'$, the blue rectangle goes from $y$ or $y'$ to $z$.

Now we can show that the fully blocked chain complex is a bigraded chain complex (here we are in the case $\mathcal{R} = \mathbb{K}$). We denote by $\widetilde{\mathrm{GC}}_d(\mathbb{G}, s)$ the subspace of $\widetilde{\mathrm{GC}}(\mathbb{G})$ that is generated by grid states $x$ with $M(x) = d$ and $A(x) = s$.

**Proposition 4.14.** *The differential $\tilde{\partial}_{\mathbb{O},\mathbb{X}}$ maps $\widetilde{\mathrm{GC}}_d(\mathbb{G}, s)$ into $\widetilde{\mathrm{GC}}_{d-1}(\mathbb{G}, s)$.*

*Proof.* Given $x \in S(\mathbb{G})$, let $y$ be a grid state such that there exists an empty rectangle $r \in \mathrm{Rect}^\circ(x, y)$ with $r \cap \mathbb{O} = r \cap \mathbb{X} = \emptyset$ it follows quickly from equation 4.2, that $M(y) = M(x) - 1$. Furthermore $A(x) = A(y)$ follows immediately from the more general fact, that the Alexander function satisfies

$$A(x) - A(y) = \#(r \cap \mathbb{X}) - \#(r \cap \mathbb{O}) \tag{4.3}$$

for any $x, y \in S(\mathbb{G})$ and $r \in \mathrm{Rect}(x, y)$, cf. [OSS15, Prop. 4.3.3]. □

We can now define the first version of grid homology.

[1] For drawing Figure 4.4, [OSS15, Figure 4.4] served as a reference.

**Definition 4.15.** *Fully blocked grid homology* $\widetilde{\mathrm{GH}}(\mathbb{G})$ is the homology of the chain complex $(\widetilde{\mathrm{GC}}(\mathbb{G}), \tilde{\partial}_{\mathbb{O},\mathbb{X}})$. Fully blocked grid homology inherits a bigrading from $\widetilde{\mathrm{GC}}(\mathbb{G})$

$$\widetilde{\mathrm{GH}}_d(\mathbb{G}, s) = \frac{\ker(\tilde{\partial}_{\mathbb{O},\mathbb{X}}) \cap \widetilde{\mathrm{GC}}_d(\mathbb{G}, s)}{\mathrm{im}(\tilde{\partial}_{\mathbb{O},\mathbb{X}}) \cap \widetilde{\mathrm{GC}}_d(\mathbb{G}, s)}. \tag{4.4}$$

As an example, we compute fully blocked grid homology for a grid diagram of the unknot.

**Example 4.16.** For the grid diagram $\mathbb{G}$ of the unknot given in Figure 2.5 there exist exactly two gridstates, we denote these by $q$ and $p$, cf Figure 4.5. The points of $p$ lie at the north-west corners of squares that are marked with an $X$. Proposition 4.10 thus implies $M(p) = 0$. There exists a rectangle $r$ from $q$ to $p$ such that $\#(r \cap \mathbb{O}) = 1$ and $\#(x \cap \mathrm{Int}(r)) = 0$. Using Equation 4.2 we deduce

$$M(q) = M(q) - M(p) = 1 - 2\#(r \cap \mathbb{O}) + 2\#(x \cap \mathrm{Int}(r)) = -1.$$

A direct calculation, using Equation 4.1, implies furthermore that $A(p) = 0$ and $A(q) = -1$. Thus, $\widetilde{\mathrm{GC}}(\mathbb{G})$ is a two dimensional vector space over $\mathbb{K}$ with one generator of bigrading $(0, 0)$ and the other generator of bigrading $(-1, -1)$. The differential $\tilde{\partial}_{\mathbb{O},\mathbb{X}}$ preserves the Alexander grading and drops the Maslov grading by one, as $\widetilde{\mathrm{GC}}(\mathbb{G})$ has no non-trivial elements of bigrading $(-1, -2)$ or $(0, -1)$ it follows that $\tilde{\partial}_{\mathbb{O},\mathbb{X}} = 0$, another way to see this is to observe that there exist no rectangles that do not contain $X$- or $O$-markings. Due to $\tilde{\partial}_{\mathbb{O},\mathbb{X}} = 0$ we deduce $\widetilde{\mathrm{GC}}_d(\mathbb{G}, s) = \widetilde{\mathrm{GH}}_d(\mathbb{G}, s)$ for all $d, s \in \mathbb{Z}$, i.e.

$$\widetilde{\mathrm{GH}}_{-1}(\mathbb{G}, -1) \cong \mathbb{K} \cong \widetilde{\mathrm{GH}}_0(\mathbb{G}, 0)$$
$$\widetilde{\mathrm{GH}}(\mathbb{G}) \cong \mathbb{K} \oplus \mathbb{K}.$$

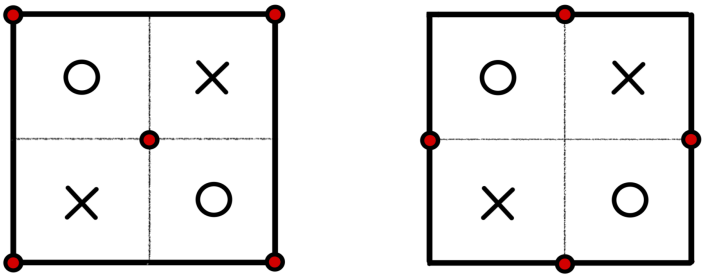


**Figure 4.5.** The two grid states $p$ (left) and $q$ (right) of the grid diagram of the unknot from Figure 2.5.

In the next section we show in Corollary 4.30 that the dimension of $\widetilde{\mathrm{GH}}(\mathbb{G})$ is divisible by $2^{n-1}$ where $n$ denotes the grid size of $\mathbb{G}$. We also give an outline of the

following theorem which shows that fully blocked grid homology leads to an integer valued knot invariant.

**Theorem 4.17. ([OSS15, Section 5.3])** *Let $\mathbb{G}$ and $\mathbb{G}'$ be grid diagrams of a knot $K$ of size $n$ and $m$. It follows that*

$$\dim_{\mathbb{K}}(\widetilde{\mathrm{GH}}(\mathbb{G}))/2^{n-1} = \dim_{\mathbb{K}}(\widetilde{\mathrm{GH}}(\mathbb{G}')/2^{m-1} \in \mathbb{Z}.$$

## 4.3 Simply Blocked Grid Homology

Instead of simply working over the base field $\mathbb{Z}/2\mathbb{Z} = \mathbb{K}$ we now define a bigraded chain complex over the polynomial ring $\mathcal{R} = \mathbb{K}[V_1, \dots, V_n]$ where $n$ is the size of the underlying grid diagram. Our goal is to define a differential on a suitable chain complex that counts empty rectangles that may contain $O$-markings. To this end we label the $O$-markings $\mathbb{O} = \{O_i\}_{i=1}^n$ which induces a bijective correspondence to the variables $V_i$.

**Definition 4.18.** For two grid states $x, y$ on a grid diagram that are connected by a rectangle $r \in \mathrm{Rect}(x, y)$, the *multiplicity* $O_i(r)$ of $r$ at the marking $O_i$ is 1 if $r$ contains $O_i$ and 0 otherwise.

**Definition 4.19.** Let $\mathbb{G}$ be a grid diagram of size $n$. The *unblocked grid chain complex* $\mathrm{GC}^-(\mathbb{G})$ is the $\mathcal{R} = \mathbb{K}[V_1, \dots, V_n]$-module freely generated by $S(\mathbb{G})$. The differential $\partial^-_{\mathbb{X}}$ is defined on grid states $x$ by

$$\partial^-_{\mathbb{X}}(x) = \sum_{y \in S(\mathbb{G})} \sum_{\{r \in \mathrm{Rect}^\circ(x,y) : r \cap \mathbb{X} = \emptyset\}} V_1^{O_1(r)} \cdots V_n^{O_n(r)} \cdot y. \tag{4.5}$$

The unblocked grid chain complex is a $\mathbb{K}$ vector space with basis

$$\{V_1^{k_1} \dots V_n^{k_n} \cdot x \mid x \in S(\mathbb{G}), k_i \in \mathbb{Z}^{\geq 0}\}.$$

The Maslov and Alexander functions are extended to this basis by

$$\begin{aligned} M(V_1^{k_1} \dots V_n^{k_n} \cdot x) &= M(x) - 2k_1 - \cdots - 2k_n \\ A(V_1^{k_1} \dots V_n^{k_n} \cdot x) &= A(x) - k_1 - \cdots - k_n. \end{aligned} \tag{4.6}$$

If we let $\mathrm{GC}^-_d(\mathbb{G}, s)$ be the $\mathbb{K}$ vector space spanned by basis elements of $\mathrm{GC}^-(\mathbb{G})$ having Maslov grading $d$ and Alexander grading $s$ we get a bigrading on $\mathrm{GC}^-(\mathbb{G})$ as a $\mathbb{K}$ vector space

$$\mathrm{GC}^-(\mathbb{G}) = \bigoplus_{d,s \in \mathbb{Z}} \mathrm{GC}^-_d(\mathbb{G}, s).$$

**Proposition 4.20.** *The unblocked grid chain complex* $\mathrm{GC}^-(\mathbb{G})$ *is a bigraded chain complex.*

*Proof.* From equation 4.6 it follows immediately that the multiplication maps $V_i$ on $\mathrm{GC}^-(\mathbb{G})$ are homogeneous of degree $(-2,-1)$. It remains to show that $\partial^-_{\mathbb{X}} \circ \partial^-_{\mathbb{X}} = 0$ and that the differential is homogeneous of degree $(-1,0)$. The first statement is proven similarly to Proposition 4.13, see [OSS15, Lemma 4.6.7] for a detailed proof. To show the second statement, let $V_1^{O_1(r)} \cdots V_n^{O_n(r)} \cdot y$ be a summand of $\tilde{\partial}_{\mathbb{O},\mathbb{X}}(x)$. Then there exists a rectangle $r \in \mathrm{Rect}^\circ(x,y)$ such that $r \cap \mathbb{X} = \emptyset$ and $\#(r \cap \mathbb{O}) = O_1(r) + \cdots + O_n(r)$. Thus equations 4.2 and 4.6 imply

$$M(V_1^{O_1(r)} \cdots V_n^{O_n(r)} \cdot y) = M(y) - 2\#(r \cap \mathbb{O}) = M(x) - 1,$$

equations 4.3 and 4.6 imply

$$A(V_1^{O_1(r)} \cdots V_n^{O_n(r)} \cdot y) = A(y) - \#(r \cap \mathbb{O}) = A(x). \qquad \square$$

Next we show that the unblocked grid chain complex is a generalization of the fully blocked grid chain complex.

**Lemma 4.21.** *Let* $(\widehat{\mathrm{GC}}(\mathbb{G}), \hat{\partial}_{\mathbb{X},O_i})$ *be the* $\mathbb{K}$ *vector space with basis*

$$\{V_1^{k_1} \cdots V_{i-1}^{k_{i-1}} \cdot V_{i+1}^{k_{i+1}} \cdots V_n^{k_n} \cdot x \mid k_{j\neq i} \in \mathbb{Z}^{\geq 0}, x \in S(\mathbb{G})\}$$

*Let* $\hat{\partial}_{\mathbb{X},O_i} : \widehat{\mathrm{GC}}(\mathbb{G}) \to \widehat{\mathrm{GC}}(\mathbb{G})$ *be defined on grid states by*

$$\hat{\partial}_{\mathbb{X},O_i}(x) = \sum_{y \in S(\mathbb{G})} \sum_{\{r \in \mathrm{Rect}^\circ(x,y) : r \cap \mathbb{X} = \emptyset, O_i(r) = 0\}} V_1^{O_1(r)} \cdots V_{i-1}^{O_{i-1}(r)} \cdot V_{i+1}^{O_{i+1}(r)} \cdots V_n^{O_n(r)} \cdot y.$$

*Then* $(\widehat{\mathrm{GC}}(\mathbb{G}), \hat{\partial}_{\mathbb{X},O_i})$ *is a bigraded chain complex over* $\mathbb{K}$ *and there exists an isomorphism of bigraded* $\mathbb{K}$ *vector spaces* $\frac{\mathrm{GC}^-(\mathbb{G})}{V_i} \cong \widehat{\mathrm{GC}}(\mathbb{G})$.

*Proof.* The quotient complex $\mathrm{GC}^-(\mathbb{G})/V_i$ inherits a bigrading

$$\frac{\mathrm{GC}^-(\mathbb{G})}{V_i} = \bigoplus_{d,s} \frac{\mathrm{GC}^-_d(\mathbb{G},s)}{\mathrm{im}(V_i) \cap \mathrm{GC}^-_d(\mathbb{G},s)},$$

each summand is isomorphic to the $\mathbb{K}$ vector space spanned by the subset of elements in the basis of $\widehat{\mathrm{GC}}(\mathbb{G})$ that have Maslov grading $d$ and Alexander grading $s$, which we denote by $\widehat{\mathrm{GC}}_d(\mathbb{G},s)$. Under this identification the differential induced from $\partial^-_{\mathbb{X}}$ on the quotient complex is represented by $\hat{\partial}_{\mathbb{X},O_i}$. It follows that $(\widehat{\mathrm{GC}}(\mathbb{G}), \hat{\partial}_{\mathbb{X},O_i})$ inherits all the properties of a bigraded chain complex over $\mathbb{K}$. $\square$

**Corollary 4.22.** *The bigraded chain complex* $\mathrm{GC}^-(\mathbb{G})$ *generalizes* $\widetilde{\mathrm{GC}}(\mathbb{G})$. *More precisely, there exists an isomorphism of bigraded chain complexes over* $\mathbb{K}$

$$\frac{\mathrm{GC}^-(\mathbb{G})}{V_1 = \cdots = V_n} \cong \widetilde{\mathrm{GC}}(\mathbb{G})$$

*where the right hand side denotes the quotient complex of* $\mathrm{GC}^-(\mathbb{G})$ *obtained after quotienting by the multiplication maps* $V_i$ *for all* $1 \leq i \leq n$.

The unblocked grid chain complex gives rise to the second homology theory for grid diagrams.

**Definition 4.23.** Fix $1 \leq i \leq n$, the *simply blocked grid chain complex* of a grid diagram $\mathbb{G}$ is the quotient complex $\mathrm{GC}^-(\mathbb{G})/V_i$. Lemma 4.21 gives a representation as a bigraded chain complex over $\mathbb{K}$ by $(\widehat{\mathrm{GC}}(\mathbb{G}), \hat{\partial}_{\mathbb{X},O_i})$. *Simply blocked grid homology* $\widehat{\mathrm{GH}}(\mathbb{G})$ is defined as the homology $H(\mathrm{GC}^-(\mathbb{G})/V_i)$ of the quotient complex.

Again we look at an example computation.

**Example 4.24.** Let $\mathbb{G}$ be the grid diagram of the unknot given in Figure 2.5. Using the same notation as in Example 4.16, we see that the unblocked grid chain complex $\mathrm{GC}^-(\mathbb{G})$ is the $\mathbb{K}[V_1, V_2]$ module, freely generated by the grid states $p$ and $q$. Since there exists no rectangle from $p$ to $q$ that does not contain an $X$-marking, we have $\partial^-_{\mathbb{X}}(p) = 0$. However, there do exist two rectangles from $q$ to $p$ that do not contain an $X$-marking. Equation 4.5 thus implies $\partial^-_{\mathbb{X}}(q) = V_1 p + V_2 p$. From Lemma 4.21 we see that the simply blocked grid chain complex $\widehat{\mathrm{GC}}(\mathbb{G})$ can be represented as the quotient $\mathrm{GC}^-(\mathbb{G})/V_2$. With this representation, the differential $\hat{\partial}_{\mathbb{X},O_2}$ on $\widehat{\mathrm{GC}}(\mathbb{G})$ evaluates to $\hat{\partial}_{\mathbb{X},O_2}(q) = V_1 p$ and $\hat{\partial}_{\mathbb{X},O_2}(p) = 0$. An arbitrary element in $\mathrm{GC}^-(\mathbb{G})/V_2$ is represented as $V_1^k p + V_1^j q$ whith $k, j \in \mathbb{Z}^{\geq 0}$. Computing $\hat{\partial}_{\mathbb{X},O_2}(V_1^k p + V_1^j q) = V_1^{j+1} p$ we see that, as a $\mathbb{K}$ vector space, $\ker \hat{\partial}_{\mathbb{X},O_2}$ is generated by $\{V_i^k p \mid k \in \mathbb{Z}^{\geq 0}\}$. Similarly, $\operatorname{im} \hat{\partial}_{\mathbb{X},O_2}$ is the subspace generated by $\{V_i^k p \mid k \in \mathbb{Z}^{\geq 1}\}$. We conclude that

$$\widehat{\mathrm{GH}}(\mathbb{G}) \cong \ker \hat{\partial}_{\mathbb{X},O_2} / \operatorname{im} \hat{\partial}_{\mathbb{X},O_2} \cong \mathbb{K}.$$

Simply blocked grid homology is a powerful tool as it is a combinatorial description of a more general homology theory for knots, called *knot Floer homology*, that detects fibered knots and the knot genus. We use simply blocked grid homology to investigate fibered knots in the next chapter. However, right now many things are yet to be shown. We dont know if $\widehat{\mathrm{GH}}(\mathbb{G})$ is independent of the chosen index $i$ let alone if simply blocked grid homology is independent of the chosen grid diagram. In the remaining part of this chapter we prove the first fact, and give a sketch proof of the second statement. Along the way we show that $\widehat{\mathrm{GH}}(\mathbb{G})$ is a finite dimensional vector space, more precisely we will see that

$$2^{n-1} \cdot \dim \widehat{\mathrm{GH}}(\mathbb{G}) = \dim \widetilde{\mathrm{GH}}(\mathbb{G})$$

where $n$ denotes the grid size of $\mathbb{G}$. The proof of Theorem 4.17 then is a corollary of the fact that $\widehat{\mathrm{GH}}(\mathbb{G})$ is a knot invariant.

As a first step we have to proof the following technical proposition.

**Proposition 4.25. ([OSS15, Lemma 4.6.9])** *Let $\mathbb{G}$ be a grid diagram of a knot. Then the multiplication maps $V_i, V_j$ on $\mathrm{GC}^-(\mathbb{G})$ are chain homotopic for all $i, j \in \{1, \dots, n\}$.*

*Proof.* Since $V_i, V_j$ are homogeneous of degree $(-2,-1)$ we need to find a morphism $p\colon \mathrm{GC}^-(\mathbb{G}) \to \mathrm{GC}^-(\mathbb{G})$ that is homogeneous of degree $(-1,-1)$ satisfying the relation $p\circ\partial^-_{\mathbb{X}} + \partial^-_{\mathbb{X}}\circ p = V_j - V_i$. To this end we define a map that counts rectangles between grid states which contain the $i$-th $X$-marking $X_i$, i.e. the $X$-marking that is in the same row as $O_i$. As usual let $x \in S(\mathbb{G})$ and define

$$h_i(x) = \sum_{y\in S(\mathbb{G})} \sum_{\{r\in \mathrm{Rect}^\circ(x,y)\colon r\cap\mathbb{X}=X_i\}} V_1^{O_1(r)} \cdots V_n^{O_n(r)} \cdot y.$$

Similarly to Proposition 4.20 we can show that $h_i$ is homogenous of degree $(-1,-1)$. The rest of the proof now uses a similar argument to Proposition 4.13. A grid state $z \in S(\mathbb{G})$ only appears as a summand in $h_i\circ\partial^-_{\mathbb{X}}(x)$ if there exists another $y \in S(\mathbb{G})$ as well as $r_1 \in \mathrm{Rect}^\circ(x,y), r_2 \in \mathrm{Rect}^\circ(y,z)$ such that $r_1 \cap \mathbb{X} = \emptyset, r_2 \cap \mathbb{X} = X_i$. For a summand in $\partial^-_{\mathbb{X}}\circ h_i(x)$ the roles of $r_2$ and $r_1$ are swapped. If $z$ is a summand of $[h_i\circ\partial^-_{\mathbb{X}} + \partial^-_{\mathbb{X}}\circ h_i](x)$ that differs from $x$ by 4 or 3 points on the grid it again vanishes modulo 2. In the case $x = z$ we again deduce that the rectangles $r_1, r_2$ share two edges on the torus and thus make up an annulus. This annulus has $X_i$ in its interior but no other $X$-markings, i.e. its width is the length of one square of the grid diagram. Let $O_k$ be the $O$-marking that is on the same column as $X_i$, we conclude that $[h_i\circ\partial^-_{\mathbb{X}} + \partial^-_{\mathbb{X}}\circ h_i](x) = V_k x + V_i x$, cf. Figure 4.6. As we are working modulo 2, $V_k x - V_i x$ is the same as $V_k x + V_i x$. If $j = k$ we can thus take $p = h_i$. Otherwise, since $\mathbb{G}$ is a grid diagram of a knot, there exists a sequence of variables $V_i = V_{m_1}, \dots, V_{m_l} = V_j$ such that in each step $X_{m_i}$ is in the same column as $O_{m_{i+1}}$ in this case one easily checks that $p = h_{m_1} + \cdots + h_{m_l}$ works. □

Observe that this proof relies on the fact that $\mathbb{G}$ represents a knot. For an arbitrary link, this statement is false.

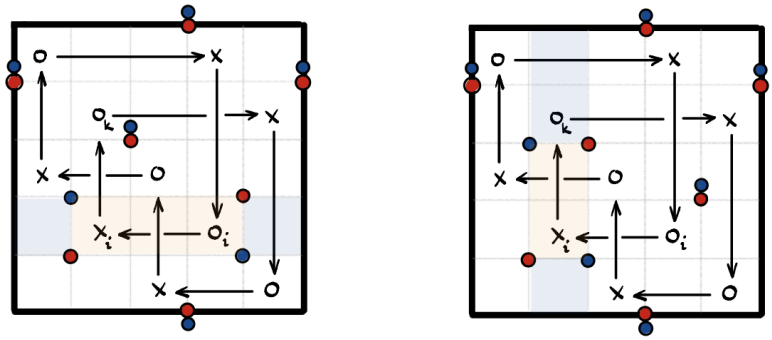


**Figure 4.6.** A visual explanation of Proposition 4.25. The red dots represent a given grid state $x$, the blue dots represent the only two choices of intermediate grid states $y$, that do not vanish under $[h_i \circ \partial^-_{\mathbb{X}} + \partial^-_{\mathbb{X}} \circ h_i](x)$.

To show that the homologies $H(\mathrm{GC}^-(\mathbb{G})/V_i), H(\mathrm{GC}^-(\mathbb{G})/V_j)$ are isomorphic as bigraded vector spaces we use an auxiliary two-dimensional bigraded vector space, $W$ over $\mathbb{K}$, with one generator $\alpha$ in bigrading $(-1,-1)$ and one generator $\beta$ in bigrading $(0,0)$. Additionally we have to define the tensor product of bigraded vector spaces.

**Definition 4.26.** Let $V = \bigoplus_{d,s\in\mathbb{Z}} V_{d,s}$ and $U = \bigoplus_{d,s\in\mathbb{Z}} U_{d,s}$ be two bigraded vector spaces. The tensor product inherits a bigrading $V \otimes U = \bigoplus_{d,s\in Z}(V\otimes U)_{d,s}$ in the following way

$$(V \otimes U)_{d,s} = \bigoplus_{\substack{d_1+d_2=d \\ s_1+s_2=s}} V_{d_1,s_1} \otimes U_{d_2,s_2}. \tag{4.7}$$

The following notation is convenient later.

**Definition 4.27.** Let $V$ be a bigraded vector space, and let $a$ and $b$ be fixed integers. The *shift of* $V$ is denoted by $V[[a,b]]$, it is a bigraded vector space given by $V[[a,b]]_{d,s} = V_{d+a,s+b}$.

**Example 4.28.** By definition $W = W_{-1,-1} \oplus W_{0,0} = \mathbb{K}\alpha \oplus \mathbb{K}\beta$. For an arbitrary bigraded vector space $V$ it follows that

$$V \otimes W \cong \bigoplus_{d,s\in\mathbb{Z}} (V_{d,s} \otimes \mathbb{K}\alpha) \oplus \bigoplus_{d,s\in\mathbb{Z}} (V_{d,s} \otimes \mathbb{K}\beta) \cong V[[1,1]] \oplus V$$

Similarly one sees that

$$V \otimes W^{\otimes 2} \cong V \oplus V[[1,1]] \oplus V[[1,1]] \oplus V[[2,2]].$$

We use the notion of tensor products between bigraded vector spaces together with the previous technical proposition to prove the following proposition which will finally enable us to show that simply blocked grid homology is finite dimensional.

**Proposition 4.29. ([OSS15, Prop. 4.6.15])** *Let $\mathbb{G}$ be a grid diagram of grid size $n$ of a knot. Let $W$ be as above the two-dimensional vector space over $\mathbb{K}$ with one generator in bigrading $(-1,-1)$ and one generator in bigrading $(0,0)$. There exists an isomorphism of bigraded vector spaces*

$$\widetilde{\mathrm{GH}}(\mathbb{G}) \cong \widehat{\mathrm{GH}}(\mathbb{G}) \otimes W^{\otimes(n-1)}$$

*where $\widehat{\mathrm{GH}}(\mathbb{G})$ is defined as the quotient complex $H(\mathrm{GC}^-(\mathbb{G})/V_i)$ for any $1 \leq i \leq n$.*

*Proof.* By Corollary 4.22 it is enough to prove

$$H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_j}\right) \cong H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1}\right) \otimes W^{\otimes(j-1)}$$

for all $1 \leq j \leq n$. Indeed, this implies $\widetilde{\mathrm{GH}}(\mathbb{G}) \cong H(\mathrm{GC}^-(\mathbb{G})/V_1) \otimes W^{\otimes(n-1)}$ and by relabelling the variables $V_i$ the statement follows. We want to use induction for the proof, to this end we interpret the 0-th tensor product $W^{\otimes 0}$ as the one-dimensional vector space over $\mathbb{K}$ with generator in bigrading $(0,0)$, this ensures that $W^{\otimes j} \otimes W \cong W^{\otimes(j+1)}$ holds for all $j \geq 0$. The base case $j = 1$ is automatically true. Assuming the equation holds for $j-1$ we look at the short exact sequence

$$0 \to \frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}} \xhookrightarrow{[V_j]} \frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}} \xrightarrow{q} \frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_j} \to 0$$

where the second arrow is induced from the injective multiplication map $V_j$ and the third arrow is the quotient map by the image. Both maps are chain maps and the quotient map is naturally of degree $(0,0)$ we can thus use Theorem 3.11 to obtain the induced long exact sequence in homology. By Proposition 4.25 the multiplication maps $V_j, V_1$ are chain homotopic. Using Proposition 3.9 and Remark 3.10 we obtain a chain homotopy between $[V_j]$ and the zero map $0 = [V_1]$. This implies that the map on homology $H([V_j])$ induced from $[V_j]$ is the zero map. The long exact sequence therefore reduces to a short exact sequence

$$0 \to H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}}\right) \xhookrightarrow{H(q)} H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_j}\right) \xrightarrow{\delta} H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}}\right) \to 0.$$

Since $H(q)$ is homogeneous of degree $(0,0)$ and $\delta$ is homogeneous of degree $(-1,-1)$ it follows that this short exact sequence of bigraded vector spaces induces the isomorphism of bigraded vector spaces

$$H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_j}\right) \cong H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}}\right) \oplus H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}}\right)[[1,1]] \tag{4.8}$$

From Example 4.28 we see that Equation 4.8 can be rewritten as

$$H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_j}\right) \cong H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_1=\cdots=V_{j-1}}\right) \otimes W.$$

The inductive hypothesis now concludes the proof. □

Since $\dim \widetilde{\mathrm{GC}}(\mathbb{G})$ equals the cardinality of the set $S(\mathbb{G})$, it follows that both $\widetilde{\mathrm{GC}}(\mathbb{G})$ and $\widetilde{\mathrm{GH}}(\mathbb{G})$ are finite dimensional. From the previous proposition we immediately arrive at the following

**Corollary 4.30.** *The vector space $\widehat{\mathrm{GH}}(\mathbb{G})$ is finite dimensional. More precisely, if $n$ is the grid size of $\mathbb{G}$ then*

$$2^{n-1} \cdot \dim \widehat{\mathrm{GH}}(\mathbb{G}) = \dim \widetilde{\mathrm{GH}}(\mathbb{G}).$$

Before continuing with the proof of the independence of grid homology we use proposition 4.29 to calculate simply blocked grid homology for the trefoil knot.

**Example 4.31.** We want to calculate simply blocked grid homology of the right-handed trefoil knot. To this end we use the grid diagram $\mathbb{G}$ depicted in Figure 2.1. The size of $\mathbb{G}$ is 5 leading to $5! = 120$ different grid states, a direct calculation of $\widehat{\mathrm{GH}}(\mathbb{G})$ would thus be very cumbersome. Instead, we can use Proposition 4.29 which implies

$$\widetilde{\mathrm{GH}}(\mathbb{G}) \cong \widehat{\mathrm{GH}}(\mathbb{G}) \otimes W^{\otimes 4}. \tag{4.9}$$

Recall that $W = W_{-1,-1} \oplus W_{0,0}$ with $W_{-1,-1} \cong W_{0,0} \cong \mathbb{K}$. Using Equation 4.7 an easy direct calculation shows

$$W^{\otimes 4} = W^{\otimes 4}_{-4,-4} \oplus W^{\otimes 4}_{-3,-3} \oplus W^{\otimes 4}_{-2,-2} \oplus W^{\otimes 4}_{-1,-1} \oplus W^{\otimes 4}_{0,0},$$

where the non-zero summands are

$$\begin{aligned} W^{\otimes 4}_{-4,-4} &\cong W^{\otimes 4}_{0,0} \cong \mathbb{K} \\ W^{\otimes 4}_{-3,-3} &\cong W^{\otimes 4}_{-1,-1} \cong \mathbb{K}^{\oplus 4} \\ W^{\otimes 4}_{-2,-2} &\cong \mathbb{K}^{\oplus 6}. \end{aligned}$$

Thus, Equation 4.9 can be written as

$$\widetilde{\mathrm{GH}}_d(\mathbb{G}, s) \cong \bigoplus_{i=0}^{4} \widehat{\mathrm{GH}}_{d+i}(\mathbb{G}, s+i)^{\oplus x_i} \tag{4.10}$$

for all $d, s \in \mathbb{Z}$ and $(x_0, x_1, x_2, x_3, x_4) = (1, 4, 6, 4, 1)$. In particular this implies that whenever $\widetilde{\mathrm{GH}}_d(\mathbb{G}, s) = 0$ we also have

$$\widehat{\mathrm{GH}}_d(\mathbb{G}, s) = \widehat{\mathrm{GH}}_{d+1}(\mathbb{G}, s+1) = \cdots = \widehat{\mathrm{GH}}_{d+4}(\mathbb{G}, s+4) = 0.$$

We can thus try to find possible generators of $\widetilde{\mathrm{GH}}(\mathbb{G})$ to calculate $\widehat{\mathrm{GH}}(\mathbb{G})$. The first step is to check in which gradings grid states of $\mathbb{G}$ may appear. Using Equation 4.1 is not hard to see that the grid states depicted in Figure 4.7 are the two unique grid states with minimal and maximal Alexander grading. The maximal Alexander grading for $\mathbb{G}$ thus is 1, while the minimal Alexander grading is $-5$. We see that $\widetilde{\mathrm{GH}}_d(\mathbb{G}, s)$ can only be nonzero for $s \in \{-5,-4,-3,-2,-1,0,1\}$ and using our observation this implies that $\widehat{\mathrm{GH}}_d(\mathbb{G}, s)$ can only be nonzero for $s \in \{-1,0,1\}$. Let us begin by determining $\widehat{\mathrm{GH}}_d(\mathbb{G}, 1)$ for all $d$. As there exist no grid states with Alexander grading greater than 1 we know that $\widetilde{\mathrm{GH}}_d(\mathbb{G}, s) = 0$ for all $s > 1$. From Equation 4.10 we conclude that $\widehat{\mathrm{GH}}_d(\mathbb{G}, 1) \cong \widetilde{\mathrm{GH}}_d(\mathbb{G}, 1)$ for all $d$. Let $x_{\max}$ be the unique grid state with Alexander grading 1 and let $d'$ be its Maslov grading, then $\widetilde{\mathrm{GC}}_d(\mathbb{G}, 1) = 0$ for all $d \neq d'$ and $\widetilde{\mathrm{GC}}_{d'}(\mathbb{G}, 1) \cong \mathbb{K}$. Recall also that $\tilde{\partial}_{\mathbb{O},\mathbb{X}}$ preserves the Alexander grading but drops the Maslov grading by one, using the uniqueness of $x_{\max}$ we see from Equation 4.4 that $\widehat{\mathrm{GC}}_{d'}(\mathbb{G}, 1) \cong \widetilde{\mathrm{GH}}_{d'}(\mathbb{G}, 1)$. Putting everything together we have $\widehat{\mathrm{GH}}_d(\mathbb{G}, 1) = 0$ for all $d \neq d'$ and $\widehat{\mathrm{GH}}_{d'}(\mathbb{G}, 1) \cong \mathbb{K}$. Similarly we can determine $\widehat{\mathrm{GH}}_d(\mathbb{G}, -1)$ for all $d$. As $\widetilde{\mathrm{GH}}_d(\mathbb{G}, s) = 0$ for all $s < -5$, Equation 4.10 implies $\widetilde{\mathrm{GH}}_d(\mathbb{G}, -5) \cong \widehat{\mathrm{GH}}_{d+4}(\mathbb{G}, -1)$ for all $d$. Let $x_{\min}$ be the unique grid state with Alexander grading $-5$ and let $d''$ be its Maslov grading. The same argument as before shows that $\widehat{\mathrm{GH}}_d(\mathbb{G}, -1) = 0$ for all $d \neq d'' + 4$ and $\widehat{\mathrm{GH}}_{d''+4}(\mathbb{G}, -1) \cong \mathbb{K}$. It now only remains to calculate $\widehat{\mathrm{GH}}_d(\mathbb{G}, 0)$ for all $d$. Again we can use Equation 4.10 to deduce that $\widetilde{\mathrm{GH}}_d(\mathbb{G}, 0) \cong \widehat{\mathrm{GH}}_d(\mathbb{G}, 0) \oplus (\widehat{\mathrm{GH}}_{d+1}(\mathbb{G}, 1))^{\oplus 4}$. We thus have $\widetilde{\mathrm{GH}}_d(\mathbb{G}, 0) \cong \widehat{\mathrm{GH}}_d(\mathbb{G}, 0)$ for all $d \neq d' - 1$ and $\widetilde{\mathrm{GH}}_{d'-1}(\mathbb{G}, 0) \cong \widehat{\mathrm{GH}}_{d'-1}(\mathbb{G}, 0) \oplus \mathbb{K}^{\oplus 4}$. Furthermore, there exist exactly five grid states for $\mathbb{G}$ whose Alexander grading equals 0, cf. Figure 4.8. Observe that for all of these five grid states, there exists a rectangle of size $1 \times 1$ connecting it to $x_{\max}$ that contains one $O$-marking. Using Equation 4.2 we deduce that the Maslov grading for all of these five grid states is $d' - 1$. Hence, all grid states with Alexander grading 0 have the same Maslov grading. Again using Equation 4.4 we see that $\widetilde{\mathrm{GH}}_d(\mathbb{G}, 0) = 0$ for all $d \neq d' - 1$ and $\widetilde{\mathrm{GH}}_{d'-1}(\mathbb{G}, 0) \cong \mathbb{K}^{\oplus 5}$ which finally implies that $\widehat{\mathrm{GH}}_d(\mathbb{G}, 0) = 0$ for all $d \neq d' - 1$ and $\widehat{\mathrm{GH}}_{d'-1}(\mathbb{G}, 0) \cong \mathbb{K}$. To conclude we only have to calculate $d'$ and $d''$. Recall from Proposition 4.10 that the grid state whose points lie on the north-west corners of squares that are marked with an $O$ has Maslov grading 0. Here, this is exactly $x_{\max}$ and thus $d' = 0$. To calculate $d''$ we have to transform $x_{\min}$ into a grid state whose Maslov grading we already know using only rectangles. Figure 4.9 depicts one way of transforming $x_{\max}$ into $x_{\min}$ by a sequence of grid states $x_{\max} = x_0, x_1, x_2, x_3, x_4 = x_{\min}$ such that $x_i$ is connected to $x_{i+i}$ through some rectangle $r_i \in \mathrm{Rect}(x_i, x_{i+1}) \cup \mathrm{Rect}(x_{i+1}, x_i)$. A repeated application of Equation 4.2 shows that $M(x_1) = -1, M(x_2) = -4, M(x_3) = -5$ and finally $M(x_{\min}) = -6$. In summary, we have shown

$$\widehat{\mathrm{GH}}_d(\mathbb{G}, s) = \begin{cases} \mathbb{K} & \text{if } (d, s) \in \{(0,1), (-1,0), (-2,-1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

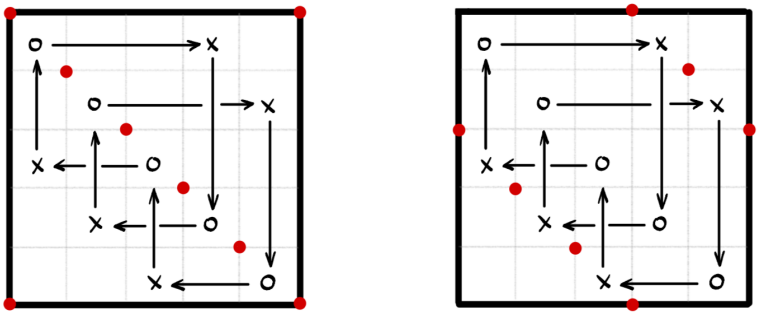


**Figure 4.7.** The two unique grid states of the right-handed trefoil knot with minimal (right) and maximal (left) Alexander grading.

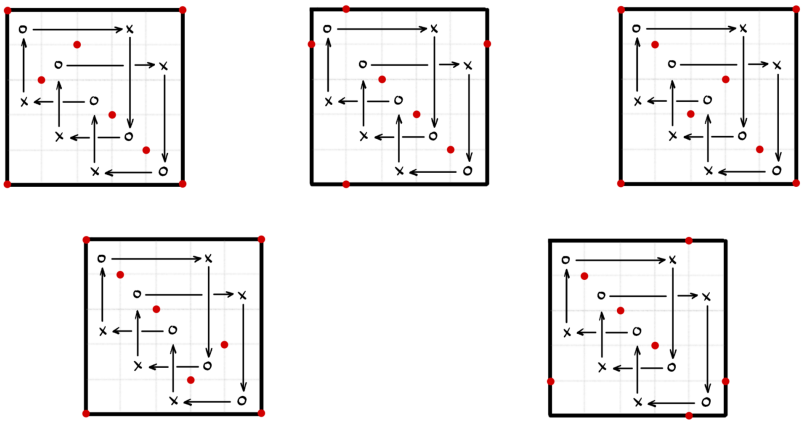


**Figure 4.8.** The five grid states of the right handed trefoil with Alexander grading 0.

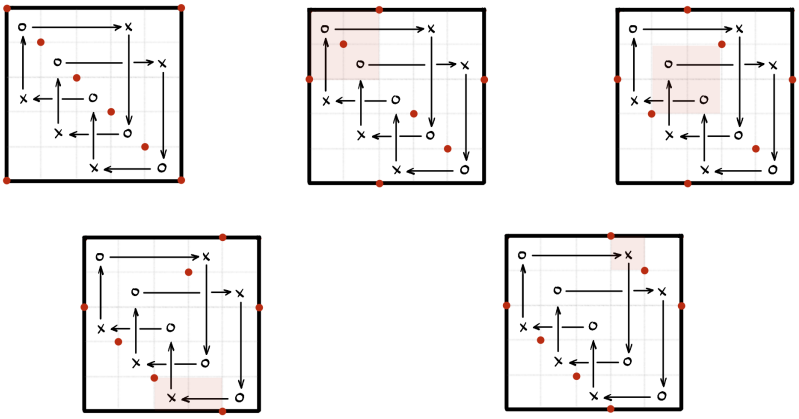


**Figure 4.9.** Transforming $x_{\max}$ (top left) into $x_{\min}$ (bottom right) by a sequence of grid states $x_{\max} = x_0, x_1, x_2, x_3, x_4 = x_{\min}$ such that $x_i$ is connected to $x_{i+i}$ by a rectangle $r_i \in \mathrm{Rect}(x_i, x_{i+1}) \cup \mathrm{Rect}(x_{i+1}, x_i)$ (colored in red).

Let us now come back to proving that simply blocked grid homology is independent of the chosen index. For this we need the following easy Lemma.

**Lemma 4.32.** *Let $V, U$ be finite dimensional bigraded vector spaces over $\mathbb{K}$. As before, let $W$ be the two-dimensional vector space over $\mathbb{K}$ with one generator in bigrading $(-1,-1)$ and the other in bigrading $(0,0)$. If there exists an isomorphism of bigraded vector spaces $V \otimes W \cong U \otimes W$, then $V$ and $U$ are isomorphic as bigraded vector spaces.*

*Proof.* As $U$ and $V$ are finite dimensional it is enough to show $\dim V_{d,s} = \dim U_{d,s}$ for all $d, s \in \mathbb{Z}$. The isomorphism $V \otimes W \cong U \otimes W$ implies $\dim(V \otimes W)_{d,s} = \dim(U \otimes W)_{d,s}$ for all $d, s \in \mathbb{Z}$. From Example 4.28 it follows that $(V \otimes W)_{d,s} = V_{d,s} \oplus V_{d+1,s+1}$. We therefore have to show that $\dim(V_{d,s} \oplus V_{d+1,s+1}) = \dim(U_{d,s} \oplus U_{d+1,s+1})$ for all $d, s \in \mathbb{Z}$ already implies $\dim V_{d,s} = \dim U_{d,s}$ for all $d, s \in \mathbb{Z}$. There exist integers $d', s'$ such that $U_{d,s} = 0 = V_{d,s}$ whenever $|d| > d'$ or $|s| > s'$. For a pair $(d, s)$ lying on the boundary of this rectangle, it is easy to show that $\dim V_{d,s} = \dim U_{d,s}$ holds. The assumption $\dim(V_{d,s} \oplus V_{d+1,s+1}) = \dim(U_{d,s} \oplus U_{d+1,s+1})$ then implies that $\dim V_{d+k,s+k} = \dim U_{d+k,s+k}$ for all $k \in \mathbb{Z}$. As $(d, s)$ was an arbitrary index element on the boundary the claim follows. □

The independence of simply blocked grid homology can now be phrased as a corollary.

**Corollary 4.33.** *Simply blocked grid homology $\widehat{\mathrm{GH}}(\mathbb{G}) = H(\mathrm{GC}^-(\mathbb{G})/V_i)$ is independent of $1 \leq i \leq n$.*

*Proof.* Proposition 4.29 in particular implies the isomorphism of bigraded vector spaces

$$H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_i}\right) \otimes W^{\otimes(n-1)} \cong H\left(\frac{\mathrm{GC}^-(\mathbb{G})}{V_j}\right) \otimes W^{\otimes(n-1)}$$

for $i, j \in \{1, \dots, n\}$. Applying the previous Lemma $(n-1)$-times, it follows that the homologies $H(\mathrm{GC}^-(\mathbb{G})/V_i)$ and $H(\mathrm{GC}^-(\mathbb{G})/V_j)$ are isomorphic bigraded vector spaces. □

The remaining part of this chapter focuses on giving an outline of the proof that simply blocked grid homology only depends on the underlying knot $K$ and not and the chosen grid diagram. It is therefore valid to speak of *the* simply blocked grid homology $\widehat{\mathrm{GH}}(K)$ of a knot $K$ without specifying a grid diagram of $K$.

**Theorem 4.34. ([OSS15, Chapter 5])** *The isomorphism type of simply blocked grid homology $\widehat{\mathrm{GH}}(\mathbb{G})$ only depends on the underlying unoriented knot $K$. More precisely, if $\mathbb{G}$ and $\mathbb{G}'$ are two grid diagrams representing $K$ then there exists an isomorphism of bigraded vector spaces over $\mathbb{K}$*

$$\widehat{\mathrm{GH}}(\mathbb{G}) \cong \widehat{\mathrm{GH}}(\mathbb{G}').$$

We first show the invariance for grid diagrams of $K$ for a fixed orientation. Due to Theorem 2.8 it is sufficient to show the above isomorphism between grid states $\mathbb{G}$ and $\mathbb{G}'$ that differ either by a commutation move, or a stabilization.

**Proposition 4.35. ([OSS15, Prop. 5.1.7])** *Let $\mathbb{G}$ and $\mathbb{G}'$ by two grid diagrams of a knot $K$ that differ by a commutation move, then the simply blocked grid homologies $\widehat{\mathrm{GH}}(\mathbb{G})$ and $\widehat{\mathrm{GH}}(\mathbb{G}')$ are isomorphic as bigraded vector spaces over $\mathbb{K}$.*

*Sketchproof.* If $\mathbb{G}$ and $\mathbb{G}'$ differ by a commutation the invariance is proven by constructing maps on the unblocked grid chain complexes

$$P\colon\ \mathrm{GC}^-(\mathbb{G}) \to \mathrm{GC}^-(\mathbb{G}') \quad \text{and} \quad P'\colon\ \mathrm{GC}^-(\mathbb{G}') \to \mathrm{GC}^-(\mathbb{G})$$

and then verifying that these maps satisfy all the conditions of being bigraded chain homotopy equivalences. I.e. one has to show that $P, P'$ are bigraded chain maps and that there exists a map $H\colon\ \mathrm{GC}^-(\mathbb{G}) \to \mathrm{GC}^-(\mathbb{G})$ that is a homotopy from the bigraded chain map $P \circ P'$ to the identity. These maps induce a bigraded chain homotopy equivalence $\widehat{\mathrm{GC}}(\mathbb{G}) \to \widehat{\mathrm{GC}}(\mathbb{G}')$ and therefore $\widehat{\mathrm{GH}}(\mathbb{G}) \cong \widehat{\mathrm{GH}}(\mathbb{G}')$. Similar to how we interpret the differential maps as counting rectangles that connect different grid states, we also have a visual interpretation for the map $P$ when $\mathbb{G}$ and $\mathbb{G}'$ differ by a commutation. Indeed, in this case it is possible to draw both diagrams on one torus, cf Figure 4.10. The map $P$ can then be interpreted as counting suitable pentagons that connect a grid state $x \in S(\mathbb{G})$ with grid states in $S(\mathbb{G}')$. Proving

that $P$ is a chain map is once again done by a similar argument as in Proposition 4.13. The map $P'$ is defined analogously to $P$ and the chain map $H$ counts suitable hexagons that connect grid states in $S(\mathbb{G})$. □

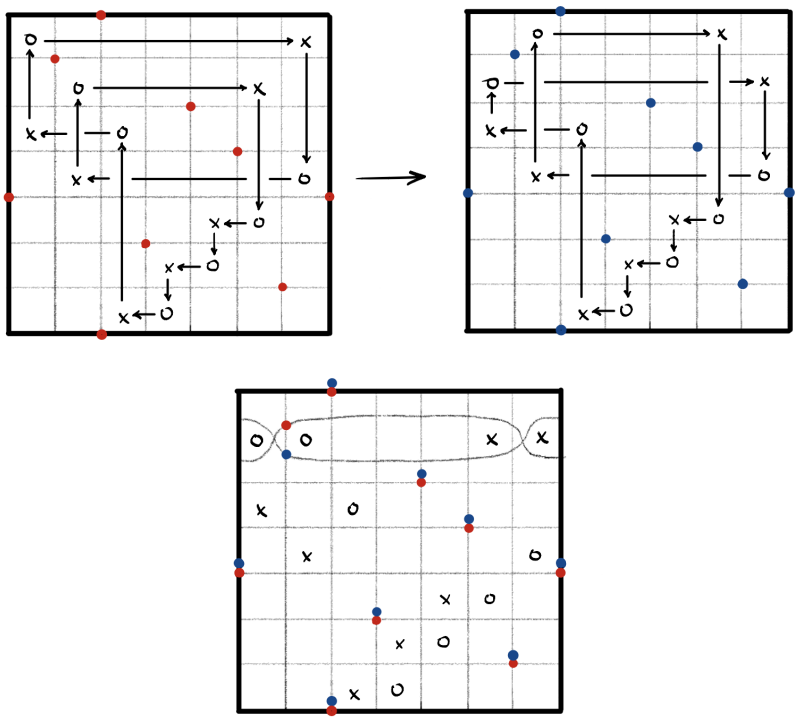


**Figure 4.10.** *Top:* A row commutation move of some grid diagram.
*Bottom:* Expressing the same row commutation move in one modified grid diagram by drawing the vertical line that lies between the two commuted rows as two curved lines.

**Proposition 4.36. ([OSS15, Cor. 5.2.3])** *Let $\mathbb{G}$ and $\mathbb{G}'$ by two grid diagrams of a knot $K$ such that $\mathbb{G}'$ is obtained from $\mathbb{G}$ after performing one stabilization, then then the simply blocked grid homologies $\widehat{\mathrm{GH}}(\mathbb{G})$ and $\widehat{\mathrm{GH}}(\mathbb{G}')$ are isomorphic as bigraded vector spaces over $\mathbb{K}$*

*Sketchproof.* The easiest approach to prove the invariance is to show the following isomorphism of bigraded vector spaces

$$\widetilde{\mathrm{GH}}(\mathbb{G}') \cong \widetilde{\mathrm{GH}}(\mathbb{G}) \otimes W \tag{4.11}$$

where $W$ is as before defined as the two-dimensional bigraded vector space over $\mathbb{K}$ with one generator in bigrading $(-1,-1)$ and the other in $(0,0)$. If $n$ is the grid

size of $\mathbb{G}$, then $\mathbb{G}'$ has grid size $n+1$, Proposition 4.29 therefore implies

$$\widehat{\mathrm{GH}}(\mathbb{G}') \cong \widetilde{\mathrm{GH}}(\mathbb{G}') \otimes W^{\otimes(n+1)} \cong \widetilde{\mathrm{GH}}(\mathbb{G}) \otimes W^{\otimes n} \cong \widehat{\mathrm{GH}}(\mathbb{G}).$$

The isomorphism 4.11 is proven in several steps. Observe that it is enough to show the invariance for a stabilization of type X:SW, due to Proposition 2.12. We then split the set of grid states of $\mathbb{G}'$ into two suitably chosen disjoint sets $I, N$. A grid state $x \in S(\mathbb{G}')$ is in $I$ if a point of $x$ lies on the intersection of the horizontal and vertical circles of the torus that were introduced by the stabilization. This induces a splitting of the fully blocked grid chain complex $\widetilde{\mathrm{GC}}(\mathbb{G}') = \tilde{I} \oplus \tilde{N}$ where $\tilde{I}$ and $\tilde{N}$ denote the spans of $I$ and $N$. With respect to this splitting the differential $\tilde{\partial}_{\mathbb{O},\mathbb{X}}$ on $\widetilde{\mathrm{GC}}(\mathbb{G}')$ splits into four maps $\partial_{I,I}, \partial_{N,N}, \partial_{I,N}$ and $\partial_{N,I}$. The last map $\partial_{N,I} : \tilde{N} \to \tilde{I}$ is zero, indeed if $r \in \mathrm{Rect}(x,y)$ is a rectangle that connects $x \in N$ to $y \in I$ then $r$ must contain one of the two $X$-markings that were introduced from the stabilization, cf Figure 4.11. Thus $\tilde{N}$ is a subcomplex with differential map $\partial_{N,N}$ and a direct calculation shows $\partial_{I,I} \circ \partial_{I,I} = 0$, i.e. $(\tilde{I}, \partial_{I,I})$ is also a chain complex. The splitting of $\widetilde{\mathrm{GC}}(\mathbb{G}')$ can thus be expressed as a short exact sequence of chain complexes

$$0 \to (\tilde{N}, \partial_{N,N}) \hookrightarrow (\widetilde{\mathrm{GC}}(\mathbb{G}')), \tilde{\partial}_{\mathbb{O},\mathbb{X}}) \twoheadrightarrow (\tilde{I}, \partial_{I,I}) \to 0.$$

The proof continues by showing that the induced map $\delta : H(\tilde{I}) \to H(\tilde{N})$ of the long exact sequence in homology coincides with the map that is induced from $\partial_{I,N} : \tilde{I} \to \tilde{N}$. Finally, one can even show that $\partial_{I,N}$ induces the trivial map on homology. It follows that the long exact sequence in homology reduces to a short exact sequence of bigraded $\mathbb{K}$ vector spaces.

$$0 \to H(\tilde{N}) \hookrightarrow \widehat{\mathrm{GH}}(\mathbb{G}') \twoheadrightarrow H(\tilde{I}) \to 0.$$

Since any short exact sequence of vector spaces is split exact this implies the isomorphism of bigraded vector spaces

$$\widehat{\mathrm{GH}}(\mathbb{G}') \cong H(\tilde{N}) \oplus H(\tilde{I}).$$

To conclude one has to proof that $H(\tilde{N})$ is isomorphic as a bigraded vector space to $\widetilde{\mathrm{GH}}(\mathbb{G})$ and $H(\tilde{I})$ is isomorphic to $\widetilde{\mathrm{GH}}(\mathbb{G})[[1,1]]$. □

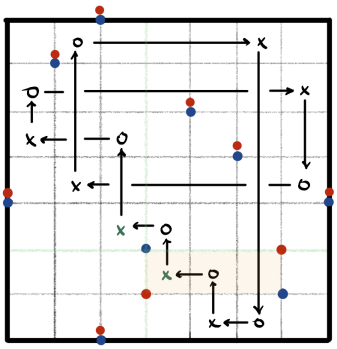


**Figure 4.11.** A grid diagram on which a stabilization of type X:SW has been performed. The newly introduced horizontal and vertical circles of the torus are colored green. The blue dots represent a grid state $y$ in $I$, the red dots represent a grid state $x$ in $N$ and the orange rectangle represents an element $r$ in $\mathrm{Rect}(x,y)$. Wee see that $r$ contains one of the newly introduced $X$-markings which are colored green.

The previous two propositions show that it is valid to speak of the simply blocked grid homology $\widehat{\mathrm{GH}}(K^+)$ of an oriented knot $K^+$. The independence of the chosen orientation now follows from the following.

**Proposition 4.37. ([OSS15, Prop. 5.3.2])** *Let $K^+$ and $K^-$ be the two oriented knots of the underlying unoriented knot $K$. Then there exists an isomorphism of bigraded vector spaces between $\widehat{\mathrm{GH}}(K^+)$ and $\widehat{\mathrm{GH}}(K^-)$.*

*Sketchproof.* If $\mathbb{G}^+$ is a grid diagram representing $K^+$ then we obtain a grid diagram $\mathbb{G}^-$ for $K^-$ by reflecing $\mathbb{G}^+$ through a diagonal. This reflection induces a bijection $\varphi : S(\mathbb{G}^+) \to S(\mathbb{G}^-)$. From Definition 4.7 it is easily seen that $\varphi$ preserves the Alexander grading. While we have not constructed the Maslov function explicitly, it is also not hard to show that $\varphi$ preserves the Maslov grading. This reflection also describes a one-to-one correspondence between $\mathrm{Rect}^\circ(x,y)$ and $\mathrm{Rect}^\circ(\varphi(x),\varphi(y))$. Thus the bigraded chain complexes $\mathrm{GC}^-(\mathbb{G}^+)$ and $\mathrm{GC}^-(\mathbb{G}^-)$ are isomorphic which is an even stronger statement than the claim $\widehat{\mathrm{GH}}(K^+) \cong \widehat{\mathrm{GH}}(K^-)$. □

# 5 Grid Diagrams of Fibered Knots

In the previous chapters we have gathered all the necessary definitions from knot theory and grid homology. We can now start to work on our central question in this chapter. We first provide some background. Simply blocked grid homology $\widehat{\mathrm{GH}}(K)$ of a knot $K$ in $S^3$ is a combinatorial version of a more general homology theory for knots, called *knot Floer homology*,

$$\widehat{\mathrm{HFK}}(K) = \bigoplus_{d,s} \widehat{\mathrm{HFK}}_d(K,s).$$

**Theorem 5.1. ([MOS09])** *Let $K$ be a knot in $S^3$, then there exists a bigraded isomorphism $\widehat{\mathrm{GH}}(K) \cong \widehat{\mathrm{HFK}}(K)$.*

Knot Floer homology is a powerful tool to investigate topological properties of knots due to the following two theorems.

**Theorem 5.2. ([OS04])** *Let $K$ be a knot in $S^3$. The Seifert genus of $K$ is equal to the maximal integer $s$ for which $\widehat{\mathrm{HFK}}_*(K,s) = \bigoplus_d \widehat{\mathrm{HFK}}_d(K,s)$ is non-zero.*

**Theorem 5.3. ([Ni07], [Ghi08])** *Let $K$ be a knot in $S^3$ with Seifert genus $g$. Then $\dim \widehat{\mathrm{HFK}}_*(K,g) = 1$ if and only if $K$ is fibered.*

We are interested in grid diagrams that admit grid states satisfying the following condition.

**Definition 5.4.** Let $\mathbb{G}$ be a grid diagram and let $x \in S(\mathbb{G})$ have Alexander grading $A(x) = s$. We call $x$ *maximal and unique* if

$$\max_{y \in S(\mathbb{G})} \{A(y)\} = s \quad \text{and} \quad \{y \in S(\mathbb{G}) \mid A(y) = s\} = \{x\}.$$

The previous discussion implies the following proposition.

**Proposition 5.5.** *Let $\mathbb{G}$ be a grid diagram of a knot $K$ with maximal and unique grid state $x$ and let $A(x) = s$. Then $K$ is fibered and has Seifert genus $s$.*

*Proof.* We use the representation of the simply blocked grid chain complex $\widehat{\mathrm{GC}}(\mathbb{G})$ from Lemma 4.21. Let $n$ be the grid size of $\mathbb{G}$ and fix $i = n-1$ to simplify the notation. Recall the action of $\hat{\partial}_{\mathbb{X},O_i}$ on $x$

$$\hat{\partial}_{\mathbb{X},O_i}(x) = \sum_{y \in S(\mathbb{G})} \sum_{\{r \in \mathrm{Rect}^\circ(x,y) : r \cap \mathbb{X} = \emptyset, O_n(r) = 0\}} V_1^{O_1(r)} \cdots V_{n-1}^{O_{n-1}(r)} \cdot y.$$

As the differential preserves Alexander grading, a summand $V_1^{O_1(r)} \cdots V_{n-1}^{O_{n-1}(r)} \cdot y$ of $\hat{\partial}_{\mathbb{X},O_i}(x)$ satisfies

$$A(x) = A(V_1^{O_1(r)} \cdots V_{n-1}^{O_{n-1}(r)} \cdot y) = A(y) - \#(r \cap \mathbb{O}).$$

This implies $A(y) \geq A(x)$, by assumption no grid state other than $x$ itself satisfies this condition. Since there can not exist a rectangle connecting $x$ with itself it follows that $\hat{\partial}_{\mathbb{X},O_i}(x) = 0$. The basis of $\widehat{\mathrm{GC}}_*(\mathbb{G},s) = \bigoplus_d \widehat{\mathrm{GC}}_d(\mathbb{G},s)$ are elements of the form $V_1^{k_1} \cdots V_{n-1}^{k_{n-1}} \cdot y$ such that $A(V_1^{k_1} \cdots V_{n-1}^{k_{n-1}} \cdot y) = s$. We conclude that $\widehat{\mathrm{GC}}_*(\mathbb{G},s)$ is generated by the single element $x$ and that the differential $\hat{\partial}_{\mathbb{X},O_i}$ is the zero map when restricted to $\widehat{\mathrm{GC}}_*(\mathbb{G},s)$. For $\widehat{\mathrm{GH}}_*(K,s) = \bigoplus_d \widehat{\mathrm{GH}}_d(K,s)$ it follows that

$$\widehat{\mathrm{GH}}_*(K,s) = \frac{\ker \hat{\partial}_{\mathbb{X},O_i} \cap \widehat{\mathrm{GC}}_*(\mathbb{G},s)}{\mathrm{im}\, \hat{\partial}_{\mathbb{X},O_i} \cap \widehat{\mathrm{GC}}_*(\mathbb{G},s)} = \widehat{\mathrm{GC}}_*(\mathbb{G},s).$$

In particular, $\dim(\widehat{\mathrm{GH}}_*(K,s)) = 1$ and $\widehat{\mathrm{GH}}_*(K,s') = 0$ for all $s' > s$. Due to $\widehat{\mathrm{GH}}_*(K,s) \cong \widehat{\mathrm{HFK}}_*(K,s)$ we conclude from the previous theorems that $K$ is fibered and $s$ is the Seifert genus of $K$. □

It is natural to be interested in the converse statement of Proposition 5.5, this is exactly our central question.

**Question 5.6.** Let $K$ be a fibered knot. Does there always exist a grid diagram of $K$ with a maximal and unique grid state?

## 5.1 Alexander Function Upper Bound

As a first step we establish an upper bound of the Alexander function.

**Definition 5.7.** Let $\mathbb{G}$ be a grid diagram and let $x = \{x_1, \dots, x_n\} \in S(\mathbb{G})$. The *winding function* $A' : S(\mathbb{G}) \to \mathbb{Z}$ is defined as

$$A'(x) = -\sum_{i=1}^{n} w(x_i).$$

When we recall the definition of the Alexander function in terms of winding numbers from Definition 4.7 we immediately see that the Alexander grading of $x$ differs from $A'(x)$ only by the term $\frac{1}{8}\sum_{j=1}^{8n} w(p_j) - (n-1)/2$ which is independent of $x$. In other words, the lower the winding number sum of a grid state, the higher is its Alexander grading.

**Definition 5.8.** The *winding matrix* of a planar realization of a grid diagram $\mathbb{G}$ of size $n$ is an $n \times n$ matrix, whose entries are the winding numbers at the lattice points of $\mathbb{G}$, cf. Figure 5.1.

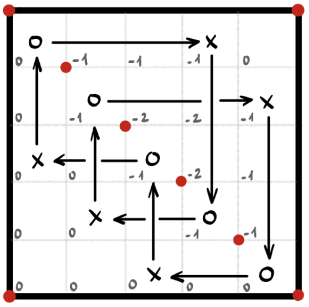


**Figure 5.1.** A grid diagram of the trefoil with a grid state visualized by red dots. Entries of the winding matrix of this planar realization are written in grey.

In Definition 4.2 we explained how to specify a grid state $x$ by a permutation $\sigma_x$ after choosing a planar realization of the grid diagram. The winding number of the point of $x$ in the $i$-th vertical circle of the planar realization is therefore the $\sigma_x(i)$-th entry of the $i$-th column of the corresponding winding matrix. We see that every row and every column of the winding matrix contains exactly one winding number of $x$.

**Definition 5.9.** Given a grid diagram $\mathbb{G}$, let $M$ be the winding matrix of any planar realization of $\mathbb{G}$. Denote by $r_i$ the minimum of the $i$-th row of $M$, by $c_i$ the minimum of the $i$-th column of $M$. Then we call $r = -\sum_i r_i$ the *row number* of $\mathbb{G}$ and $c = -\sum_i c_i$, the *column number* of $\mathbb{G}$.

As any two winding matrices of the same grid diagram only differ by a sequence of cyclic permutations of the rows and columns we see that $r$ and $c$ are independent of the chosen planar realization. The row- and column-numbers therefore give rise to the following upper bounds.

$$A'(x) \leq \min\{r, c\}$$
$$A(x) \leq \min\{r, c\} + \frac{1}{8}\sum_{j=1}^{8n} w(p_j) - (n-1)/2.$$

**Definition 5.10.** We call a grid state $x$ *perfect* if it realizes the upper bound for the Alexander function, i.e. if $A'(x) = \min\{r, c\}$. We call $x$ *row-perfect* if $A'(x) = r$ and *column-perfect* if $A'(x) = c$.

Row- and column-numbers enable us to quickly check if certian grid states are maximal. In Proposition 5.15 of the next section we show that there exists an efficient method of analyzing if a grid diagram has a unique perfect grid state.

**Example 5.11.** The grid state in Figure 5.1 is the unique perfect grid state of this diagram. Indeed, for the winding matrix in the figure, we have $r = c = 6$. The grid state $x$ given by the purple dots realizes this upper bound since $A'(x) = 6$. To see the uniqueness, note that any maximal grid state of this grid diagram must have its corresponding entry in the third row of the winding matrix $M$ positioned at the same spot since this is the only point where the entry is minimal for this row of $M$. This forces the corresponding entry in the second and forth rows to also be at the same spot as in the given grid state. Continuing like this, also the entries in the first and finally the last row of $M$ are fixed.

## 5.2 Perfect Grid States

For the grid state in Example 5.11 it was rather easy to see that it was perfect and unique. In this chapter we formulate an efficient method of analizing if a grid diagram has a unique perfect grid state using similar ideas. We start with a technical definition.

**Definition 5.12.** Let $\mathbb{G}$ be a grid diagram of size $n$, let $M$ be the winding matrix of a fixed planar realization of $\mathbb{G}$ with columns $c_1, \dots, c_n$. Let $x$ be a grid state with grid state permutation $\sigma$ as defined in 4.2. We say that $(M, \sigma)$ *admits a loop* if there exist columns $c_{i_1} \dots c_{i_k}$ of $M$ and for each column an index $j_{i_1} \dots j_{i_k}$ such that $j_{i_r} \neq \sigma(i_r)$ and $M_{j_{i_r}, i_r} = M_{\sigma(i_r), i_r}$ for all $1 \leq r \leq k$. Additionally we require that the sequence of matrix indices

$$(\sigma(i_1), i_1), (j_{i_1}, i_1), (\sigma(i_2), i_2), (j_{i_2}, i_2), \dots, (\sigma(i_k), i_k), (j_{i_k}, i_k)$$

describes a loop in $M$, i.e. $j_{i_r} = \sigma(i_{r+1})$ for all $1 \leq r \leq k-1$ and $j_{i_k} = \sigma(i_1)$.

**Example 5.13.** The following $5 \times 5$ matrix $M$ admits a loop for the permutation $\sigma = (1, 3, 4, 2, 5)$. The first three columns contain suitable indices $j_1, j_2, j_3$. For $1 \leq i \leq 3$, the entries $M_{\sigma(i),i}$ are red and $M_{j_i,i}$ are blue.

$$\begin{pmatrix} -1 & 0 & -1 & 2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ -1 & -2 & 1 & 1 & -2 \\ -1 & -2 & -1 & 3 & -2 \\ 1 & -1 & 3 & 1 & -3 \end{pmatrix}$$

The notion of a loop is useful due to the following fact.

**Lemma 5.14.** *Let $\mathbb{G}$ be a grid diagram of size $n$ and let $M$ be the winding matrix for a fixed planar realization. Let $x \in S(\mathbb{G})$ be described by the permutation $\sigma$. If $(M, \sigma)$ admits a loop then there exists another grid state $x \neq y \in S(\mathbb{G})$ such that $A(y) = A(x)$.*

*Proof.* Let $c_{i_1}, \dots, c_{i_k}$ be the columns of $M$ with indices $j_{i_1}, \dots, j_{i_k}$ that create the loop for $(M, \sigma)$. To simplify the notation assume $i_1 = 1, \dots, i_k = k$. We can then define the sequence

$$\tau = (j_1, \dots, j_k, \sigma(k+1), \dots, \sigma(n)).$$

By construction, $\tau$ is a permutation which is different from $\sigma$. Indeed the looping condition implies that $j_1, \dots, j_k$ are all different and $j_r \neq \sigma(r')$ for $1 \leq r \leq k$ and $r' \in \{k+1, \dots, n\}$. Let $y$ be the grid state of $\mathbb{G}$ with grid state permutation $\tau$. The claim $A(y) = A(x)$ then follows from the assumption $M_{j_r,r} = M_{\sigma(r),r}$. □

We can now formulate and prove an important Proposition about the existence of unique perfect grid states.

**Proposition 5.15.** *Let $\mathbb{G}$ be a grid diagram that has a unique column-perfect grid state. Let $M$ be the winding matrix of a fixed planar realization of $\mathbb{G}$. Then there exists a column of $M$ that has a unique minimal entry for this column. The analogous statement holds for row-perfect grid states.*

*Proof.* We formulate the proof for column-perfect grid states, the row-perfect case works similarly. Let $n$ be the grid size of $\mathbb{G}$ and assume contrapositively that all columns of $M$ have more than one entry that is minimal for this column. If the diagram does not admit a column-perfect grid state we are done, otherwise there exists a permutation $\sigma$ that represents a column-perfect grid state $x$. Additionally, for all columns $1 \leq i \leq n$ of $M$ there exists an index $j_i \neq \sigma(i)$ such that $M_{j_i,i} = M_{\sigma(i),i}$. If we manage to create a permutation $\tau$ different from $\sigma$ such that at each

index $i$ we have $\tau(i) \in \{\sigma(i), j_i\}$ the claim follows. This can be achieved using Algorithm 1.

The Algorithm can be visualized as follows. We start by connecting the point of $x$ that corresponds to $\sigma(1)$ by a vertical line to the point on the grid diagram that corresponds to $j_1$. We then connect this point to the only point of $x$ on the same horizontal line, i.e. to the point represented by $\sigma(t)$ for some $1 \leq t \leq n, t \neq 1$ and again connect this point by a vertical line to the point corresponding to $j_t$. Continuing like this, two things can happen. If the piecewise linear curve meets a point of $x$ twice, a loop is created which specifies a new column-perfect grid state as explained in Lemma 5.14. Otherwise, the modified version of $\sigma$ is at some point again a permutation that is different from the original permutation, cf. Figure 5.2. □

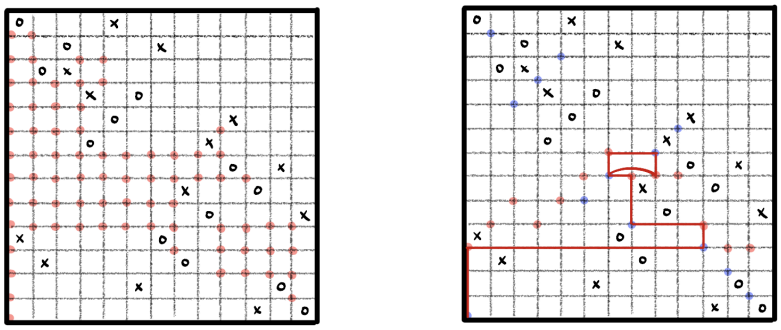


**Figure 5.2.** A visual explanation of Proposition 5.15.
*Left*: An example of a grid diagram where no column of the winding matrix has a unique minimal value. The red dots represent entries of the winding matrix that are minimal for this column.
*Right*: For each column, we choose two minimal entries (blue and red). The blue dots represent a column perfect grid state. Then we draw a curve (red) until a point of the grid state is visited twice. The closed loop which is created from this curve tells us which blue dots can be switched to red dots to create a new column perfect grid state.

The previous proposition can be extended to an efficient method that checks if a grid diagram $\mathbb{G}$ admits a unique perfect grid state as follows. We fix a planar realization of $\mathbb{G}$ with winding matrix $M$ and compute $r$ and $c$. If one of the values is strictly bigger we know that there can only exist one type of perfect grid state, say column-perfect grid states. We then check if there exists a column of $M$ that has a unique minimal entry for this column. If not, we immediately know from Proposition 5.15 that $\mathbb{G}$ does not have a column-perfect grid state and thus no perfect grid state at all. If such a column does exist we save the unique minimal entry,

**Algorithm 1** New Perfect Grid State

**Require:** A permutation $\sigma$ of length $n$ and a list $J$ of the same length, such that $J[i] \neq \sigma[i]$ for all $1 \leq i \leq n$.

**Ensure:** A new permutation $\tau$ different from $\sigma$, such that at each index $i$ the new permutation has the value $\sigma(i)$ or $J(i)$.

```
n ← length(σ)
S ← copy(σ)
S[1] ← J[1]                          ▷ Replace first element with value from J
i ← 1                                ▷ Index of most recent change in S
changes ← [ i ]                      ▷ List of modified indices.
while true do
    duplicates ← duplicate elements of S
    if duplicates = ∅ then           ▷ No duplicate found: the permutation is valid
        τ ← copy(S)
        return τ
    else
        Let t be the unique element in duplicates with t ≠ i
    end if
    if t ∈ changes then              ▷ Loop detected: modify only indices between t
        start ← position of t in changes
        loop ← elements of changes from position start + 1 onward
        τ ← copy(σ)
        for all k ∈ loop do
            τ[k] ← J[k]
        end for
        return τ
    end if
    S[t] ← J[t]                      ▷ Update S at t with the value from J
    i ← t                            ▷ Update current index to recently changed index
    Append i to changes
end while
```

and then delete the column and row of $M$ in which the unique entry is positioned. For the new reduced matrix three things can happen:

- There are columns that now no longer have their original minimum as an entry. In that case we know that $\mathbb{G}$ does not admit a column-perfect grid state.
- For all remaining rows there does not exists a column that only has one entry with minimal value. If this happens, we know that $\mathbb{G}$ does not admit a unique column-perfect grid state by the same argument as in Proposition 5.15. It could be that $\mathbb{G}$ admits several perfect grid states or none.
- There again exists at least one column that has a unique entry with the original minimal value of the column. Here we can continue as before, by extracting the original position of the new unique minimal entry and again reducing the matrix.

This process either stops after finding a unique column-perfect grid state or it stops if after some iteration case 1. or 2. happens. In the case that $r = c$ we do have to perform this method for columns as well as rows. A downside of this procedure is that it can not tell grid diagrams that admit more than one column-perfect grid state apart from grid diagrams that admit no column-perfect grid state. This would be interesting to know, since if $\mathbb{G}$ admits no column-perfect grid state, it may still have a maximal and unique grid state. An example of this is given in Figure 5.3.

**Example 5.16.** The grid state in Figure 5.3 is maximal und unique but not perfect. Indeed, the grid state $x$ given by red dots is not perfect since $A'(x) = 13$ whereas $r = 14 = c$. Note that the 5th and 6th rows of the corresponding winding matrix have their unique minimal entry at the same column, it is thus impossible to have a row-perfect grid state. Analogously the 3rd and 4th column have their unique minimal entry at the same row. Therefore column-perfect grid states also cannot exist and we see that $x$ is maximal. Any other maximal grid state must have their corresponding values on the winding matrix of the 3rd or 4th column on the 5th row and the entries of the 2nd, 5th, 6th, 7th and 8th column are forced to be in the same spot as for $x$. It is now easy to see that $x$ is a unique maximal grid state.

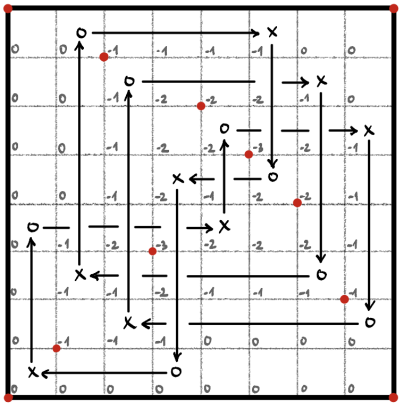


**Figure 5.3.** An example of a grid diagram with a grid state that is maximal and unique but not perfect.

While our method to prove the uniqueness of maximal grid states is efficient in the case that the grid state is perfect, we can see from the previous example, that it is much more tedious to show uniqueness when the grid state is not perfect. In Figure 5.3 the grid state $x$ is almost perfect in the sense that $A'(x)+1 = \min\{r, c\}$. In general, the farther away a grid state is from reaching this upper bound, the less practical our method becomes. We thus focus on grid diagrams of knots that admit a unique perfect grid state. For convenience we introduce the following definition.

**Definition 5.17.** Given a grid diagram $\mathbb{G}$ of a knot $K$, we call $\mathbb{G}$ *nice*, if there exists a unique perfect grid state for $\mathbb{G}$.

Instead of working on Question 5.6 we reformulate it into:

**Question 5.18.** Let $K$ be a fibered knot. Does there always exist a nice grid diagram of $K$?

# 6 Programming

In the previous chapter we have seen how Proposition 5.15 leads to an efficient method for checking if a grid diagram is nice. The aim of this last chapter is to develop a Python package based on these ideas. The package should be able to take as input a knot, given as a knot name or as a grid diagram, for which it then tries to find a nice grid diagram by creating a large list of candidate grid diagrams that represent the input knot using commutations and optionally stabilizations. On each of these the program should calculate the winding matrix and then apply the method from Chapter 5 of finding a unique perfect grid state. While this approach is very brute force, most of the required calculations can be implemented very efficiently. Our goal is to investigate Question 5.18 by finding a large list of fibered knots that admit a nice grid diagram.

---

**Disclaimer:** The code provided in this chapter was annotated using Claude 4. Claude and ChatGPT were also used to design the rough structure of the algorithms in 6.7 and 6.9.

---

## 6.1 Importing and pre-processing

We will use the knot database provided from *knotinfo.math.indiana.edu* [LM24]. It contains 5397 fibered prime knots and, conveniently, all of the knots in this database are already equipped with a minimal grid diagram representation. A grid diagram is given in so called *grid-notation*, this is a list of tuples that specify the positions of the $X$- and $O$-markings. For example, the diagram of size 5 from Figure 2.1 would be given by a sequence of 10 tuples

$$(1,1),(1,3),(2,2),(2,4),(3,3),(3,5),(4,1),(4,4),(5,2),(5,5).$$

In this notation the orientation of the grid diagram is not specified. In order to perform column-commutations efficiently it is much more practical to describe

grid diagrams by vertical segments. Again using Figure 2.1 as an example, we can specify it by 5 tuples that represent oriented vertical intervals

$$(3,1),(4,2),(5,3),(1,4),(2,5).$$

We call this the *vertlist-notation*. The first step is to transform grid-notation into vertlist-notation, we do this in two steps. The first function transforms grid-notation into an intermediate format *gridlist-notation*, the second function transforms this intermediate format into vertlist-notation.

```
    def gridnotation_to_gridlist(gridnotation: GridNotation
        ) -> GridList:
    """
    Convert grid notation to grid list representation.

    Parameters
    ----------
    gridnotation : List[List[int]]
        Grid notation specifying a grid diagram

    Returns
    -------
    List[int]
        Grid list representation
    """
    if not gridnotation:
        raise ValueError("Grid notation cannot be empty")

    temp = [gridnotation[0][1]]
    current_tuple = gridnotation[0]

    while len(temp) < len(gridnotation):
        if len(temp) % 2 == 1:
            # Look for matching first coordinate
            for segment in gridnotation:
                if segment[1] == temp[-1] and segment[0] !=
                    current_tuple[0]:
                    temp.append(segment[0])
                    current_tuple = segment
                    break
            else:
                raise ValueError("Invalid grid notation: no
                    matching segment found")
        else:
            # Look for matching second coordinate
            for segment in gridnotation:
                if segment[0] == temp[-1] and segment[1] !=
                    current_tuple[1]:
                    temp.append(segment[1])
```

```
                    current_tuple = segment
                    break
            else:
                raise ValueError("Invalid grid notation: no
                    matching segment found")

    # Convert to 0-indexed
    return [x - 1 for x in temp]

    def vlist(gridlist: GridList) -> VertList:
    """
    Convert grid list to vertical segment list.

    Parameters
    ----------
    gridlist : List[int]
        Grid list representation.

    Returns
    -------
    List[Tuple[int, int]]
        List of tuples representing oriented vertical
            segments.
    """
    extended_grid = gridlist.copy()
    extended_grid.append(gridlist[0])
    n = len(extended_grid)
    x = n + 1

    vsegments: List[Optional[Tuple[int, int]]] = [None] *
        (2 * n + 1)
    vsegments[x] = (extended_grid[0], extended_grid[2])

    for i in range(2, n - 2, 2):
        x = x + extended_grid[i + 1] - extended_grid[i - 1]
        if 0 <= x < len(vsegments):
            vsegments[x] = (extended_grid[i], extended_grid
                [i + 2])
        else:
            raise IndexError("Calculated index is out of
                bounds!")

    return [seg for seg in vsegments if seg is not None]
```

**Code 6.1.** Converting grid diagrams in grid-notation to vertlist-notation.

The orientation that is chosen during this transformation process does not matter. As reversing the placings of the $X$- and $O$-markings on a grid diagram amounts to reversing the orientation of the underlying knot, we can switch quickly be-

tween both orientations of the grid diagram. We will use this approach later to look for unique perfect grid states in both orientations. In vertlist-notation, a simple python function is enough to look for possible column-commutations on the grid diagram.

```
    def can_commute(t1: Tuple[int, int], t2: Tuple[int, int
        ]) -> bool:
    """
    Check if two neighbouring segments can be commuted.

    Parameters
    ----------
    t1, t2 : Tuple[int, int]
        Two segments to check.

    Returns
    -------
    bool
        True if segments can be commuted, False otherwise.
    """

    a, b = t1
    c, d = t2
    max1, min1 = max(a, b), min(a, b)
    max2, min2 = max(c, d), min(c, d)

    return (
        (max1 <= min2) or
        (min1 >= max2) or
        (max1 >= max2 and min1 <= min2) or
        (max2 >= max1 and min2 <= min1)
    )
```

**Code 6.2.** Checking if two neighbouring segments can be commuted.

To also perform row-commutations efficiently we describe the grid diagram using its vertical segments instead. We call this the *horzlist-notation*. Another simple Python function is enough to transform vertlist- into horzlist-notation. And we can use Function 6.2 to check if neighbouring rows can be commuted. The next step is to take a grid diagram given as a vertlist and compute a list of all grid diagrams obtainable after one row- or column commutation.

```
    def c_move(input_list: Union[VertList, HorzList]) ->
        List[Union[VertList, HorzList]]:
    """
    If input is vertical segment list, generates all column
        commutation moves.
```

```
    If input is horizontal segment list, generates all row
       commutation moves.

    Parameters
    ----------
    input_list : List[Tuple[int, int]]
        Vertical or horizontal segment list.

    Returns
    -------
    List[List[Tuple[int, int]]]
        List of all possible configurations after one
           commutation.
    """

    result = []
    seen: Set[Tuple[Tuple[int, int], ...]] = set()
    n = len(input_list)

    def add_to_result(lst: List[Tuple[int, int]]) -> None:
        """Add configuration to result if not seen before.
           """
        tpl = tuple(lst)
        if tpl not in seen:
            seen.add(tpl)
            result.append(lst)

    # Try adjacent commutations
    for i in range(n - 1):
        if can_commute(input_list[i], input_list[i + 1]):
            swapped_list = input_list.copy()
            swapped_list[i], swapped_list[i + 1] =
               swapped_list[i + 1], swapped_list[i]
            add_to_result(swapped_list)

    # Try wrap-around commutation
    if can_commute(input_list[0], input_list[-1]):
        swapped_list = input_list.copy()
        swapped_list[0], swapped_list[-1] = swapped_list
           [-1], swapped_list[0]
        add_to_result(swapped_list)

    return result

    def knot_commute(vertlist: VertList) -> List[VertList]:
    """
    Generate all possible knot diagrams obtainable by one
       commutation.
```

```
    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
        Vertical segment list.

    Returns
    -------
    List[List[Tuple[int, int]]]
        List of all possible vertical lists after one
            commutation.

    Notes
    -----
    This considers both vertical and horizontal
       commutations.
    """

    # Get vertical commutations
    v_commutations = c_move(vertlist)

    # Get horizontal commutations and convert back to
       vertical
    h_list = v_to_h(vertlist)
    h_commutations = c_move(h_list)
    h_to_v_commutations = [h_to_v(horzlist) for horzlist in
        h_commutations]

    # Combine and remove duplicates
    c_set = set(tuple(lst) for lst in v_commutations +
       h_to_v_commutations)
    c_list = [list(tpl) for tpl in c_set]

    return c_list
```

**Code 6.3.** Creating a list of all grid diagrams obtainable from one commutation.

As we are thinking of the grid diagram embedded on the torus, we allow for column commutations between the first and last vertical segment as well as row commutations between the highest and lowest row. Another benefit of describing grid diagrams in vertlist-notation is that calculating the winding matrix is also straighforward.

```
    def w_matrix(vertlist: VertList) -> WindingMatrix:
    """
    Calculate the winding matrix for a knot grid diagram.

    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
```

```
        Vertical segment list.

    Returns
    -------
    np.ndarray
        Winding matrix of grid diagram represneted by
            Vertical segment list.

    Notes
    -----
    The winding number increases by 1 when crossing an
       upward segment
    and decreases by 1 when crossing a downward segment.
    """

    size = len(vertlist)
    result = []

    for i in range(size):
        row = [0]  # Start with winding number 0 for first
            column

        for j in range(size - 1):
            tail, head = vertlist[j]

            if tail <= i < head:  # Upward segment
                row.append(row[-1] + 1)
            elif head <= i < tail:  # Downward segment
                row.append(row[-1] - 1)
            else:  # No crossing
                row.append(row[-1])

        result.append(row)

    return np.array(result, dtype=np.int32)
```

**Code 6.4.** Computing the winding matrix of a grid diagram given in vertlist-notation.

This function is inspired by the winding matrix function inside Gridlink, a computer program to work with grid diagrams of knots, [Cul].

## 6.2 Breadth-first search algorithms

On a matrix of a knot we can look for unique perfect grid states using the method derived from Proposition 5.15. This is the heart of our python package.

```
    def h_type_0_permutation(matrix: WindingMatrix) ->
        Union[Permutation, str]:
    """
    Find a unique horizontal type-0 permutation (i.e. a
        unique row-perfect grid state) if it exists.

    Parameters
    ----------
    matrix : np.ndarray
        Winding matrix.

    Returns
    -------
    List[int] or str
        Permutation if unique one exists, error message
            otherwise.
    """

    n = len(matrix)

    # Find column indices of minimum values in each row
    min_indices = [
        set(i for i, val in enumerate(row) if val == min(
            row))
        for row in matrix
    ]

    result: List[Optional[int]] = [None] * n

    while any(s is not None for s in min_indices):
        # Find a singleton set
        singleton_index = next(
            (j for j, s in enumerate(min_indices) if s is
                not None and len(s) == 1),
            None
        )

        if singleton_index is None:
            return "No unique h-type-0 permutation exists."

        # Extract the single element
        singleton_set = min_indices[singleton_index]
        x = next(iter(singleton_set))
        result[singleton_index] = x
        min_indices[singleton_index] = None

        # Remove x from all other sets
        for s in min_indices:
            if s is not None:
```

```
                s.discard(x)

        # Check for empty sets
        if any(s is not None and len(s) == 0 for s in
            min_indices):
             return "No unique h-type-0 permutation exists."

    # Ensure all positions are filled
    if None in result:
        return "This should never happen, invalid input or
            problem in code!"

    return [x for x in result if x is not None]
```

**Code 6.5.** Searching for unique row-perfect grid states on a winding matrix.

Similarly we define `v_type_0_permutation()` to search for unique column-perfect grid states on a winding matrix. According to Proposition 5.5, the Alexander grading of a unique perfect grid state coincides with the Seifert genus of the knot. We therefore define a function to calculate the Alexander grading of a grid state to later verify if the grid state the code finds satisfies this condition.

```
def a_grading(vertlist: VertList, matrix: WindingMatrix
    , permutation: Permutation) -> int:
"""
Calculate the Alexander grading for a grid state.

Parameters
----------
vertlist : List[Tuple[int, int]]
    Vertical segment list.
matrix : np.ndarray
    Winding matrix.
permutation : List[int]
    Permutation representing the grid state.

Returns
-------
int
    Alexander grading of the grid state.
"""

n = len(vertlist)
m = (n - 1) / 2

a_sum = 0

for tpl in vertlist:
    col = vertlist.index(tpl)
```

```
        upper_row = min(tpl)
        lower_row = max(tpl)

        if upper_row == 0 and col != vertlist.index(
            vertlist[-1]):
             upper_sum = matrix[0][col] + matrix[0][col + 1]

        elif upper_row != 0 and col != vertlist.index(
            vertlist[-1]):
             upper_sum = matrix[upper_row-1][col] + matrix[
                upper_row-1][col+1] + matrix[upper_row][col]
                 + matrix[upper_row][col + 1]

        elif upper_row == 0 and col == vertlist.index(
            vertlist[-1]):
             upper_sum = matrix[0][col]

        elif upper_row != 0 and col == vertlist.index(
            vertlist[-1]):
             upper_sum = matrix[upper_row-1][col] + matrix[
                upper_row][col]


        if col != vertlist.index(vertlist[-1]):
             lower_sum = matrix[lower_row-1][col] + matrix[
                lower_row-1][col+1] + matrix[lower_row][col]
                 + matrix[lower_row][col + 1]

        elif col == vertlist.index(vertlist[-1]):
             lower_sum = matrix[lower_row-1][col] + matrix[
                lower_row][col]

        a_sum = a_sum + upper_sum + lower_sum


    a_sum = a_sum/8
    w_sum = 0

    for i in permutation:
        j = permutation.index(i)
        w_sum = w_sum + matrix[j][i]

    return int(-w_sum + a_sum - m)
```

**Code 6.6.** Calculating the Alexander grading of a grid state.

We also need to define `rev()`, a simple function that reverses the orientation of grid diagrams given in vertlist- or horzlist-notation, we will use this function to search for unique perfect grid states on both orientations of the knot. Additionally

we also define `vperm_to_hperm()`, this function is used to transform the notation of grid states found from `v_type_0_permutation()` into the notation of grid states found from `h_type_0_permutation()`. Combining everything, the first breadth-first search algorithm of our python package looks as follows.

```
    def try_permutations(vertlist: VertList) -> Optional[
        Tuple[VertList, str, Permutation, WindingMatrix, int
        ]]:
    """
    Try to find a unique perfect grid state for a diagram.

    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
        Vertical segment list.

    Returns
    -------
    Optional[Tuple]
        If found: (vertlist, type, permutation, matrix,
            alexander_grading)
        If not found: None

    Notes
    -----
    This function tries both the original and reversed
        orientation,
    checking for both horizontal and vertical perfect grid
        states.
    """

    # Try original orientation
    matrix = w_matrix(vertlist)
    rowsum = np.sum(np.min(matrix, axis=1))
    colsum = np.sum(np.min(matrix, axis=0))

    # Check based on row/column sums
    if rowsum > colsum:
        # Only horizontal perfect grid state possible
        h_perm = h_type_0_permutation(matrix)
        if not isinstance(h_perm, str):
            return (vertlist, "h_type_0", h_perm, matrix,
                a_grading(vertlist, matrix, h_perm))

    elif colsum > rowsum:
        # Only vertical perfect grid state possible
        v_perm = v_type_0_permutation(matrix)
        if not isinstance(v_perm, str):
            h_perm = vperm_to_hperm(v_perm)
```

```
            return (vertlist, "v_type_0", h_perm, matrix,
                a_grading(vertlist, matrix, h_perm))

    else:  # rowsum == colsum
        # Try both types
        h_perm = h_type_0_permutation(matrix)
        if not isinstance(h_perm, str):
            return (vertlist, "h_type_0", h_perm, matrix,
                a_grading(vertlist, matrix, h_perm))

        v_perm = v_type_0_permutation(matrix)
        if not isinstance(v_perm, str):
            h_perm = vperm_to_hperm(v_perm)
            return (vertlist, "v_type_0", h_perm, matrix,
                a_grading(vertlist, matrix, h_perm))

    # Try reversed orientation
    vertlist_rev = rev(vertlist)
    matrix_rev = w_matrix(vertlist_rev)
    rowsum_rev = np.sum(np.min(matrix_rev, axis=1))
    colsum_rev = np.sum(np.min(matrix_rev, axis=0))

    if rowsum_rev > colsum_rev:
        h_perm = h_type_0_permutation(matrix_rev)
        if not isinstance(h_perm, str):
            return (vertlist_rev, "h_type_0", h_perm,
                matrix_rev,
                    a_grading(vertlist_rev, matrix_rev,
                        h_perm))

    elif colsum_rev > rowsum_rev:
        v_perm = v_type_0_permutation(matrix_rev)
        if not isinstance(v_perm, str):
            h_perm = vperm_to_hperm(v_perm)
            return (vertlist_rev, "v_type_0_rev", h_perm,
                matrix_rev,
                    a_grading(vertlist_rev, matrix_rev,
                        h_perm))

    else:  # rowsum_rev == colsum_rev
        h_perm = h_type_0_permutation(matrix_rev)
        if not isinstance(h_perm, str):
            return (vertlist_rev, "h_type_0", h_perm,
                matrix_rev,
                    a_grading(vertlist_rev, matrix_rev,
                        h_perm))

        v_perm = v_type_0_permutation(matrix_rev)
        if not isinstance(v_perm, str):
```

```
            h_perm = vperm_to_hperm(v_perm)
            return (vertlist_rev, "v_type_0_rev", h_perm,
                matrix_rev,
                    a_grading(vertlist_rev, matrix_rev,
                        h_perm))

    return None

    def gridstate_finder_commute(vertlist: VertList, n: int
        ) -> Optional[Dict]:
    """
    Find unique perfect grid states through commutation
        moves.

    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
        Initial vertical segment list.
    n : int
        Maximum number of commutation iterations.

    Returns
    -------
    Optional[Dict]
        Dictionary containing grid state information if
            found, None otherwise.

    Notes
    -----
    We use a breadth-first search approach to explore
        commutation space.
    """

    # Check initial state
    perm_result = try_permutations(vertlist)
    if perm_result:
        vlist_out, perm_type, perm, matrix, alex =
            perm_result
        return {
            "vlist": vlist_out,
            "matrix": matrix.tolist(),  # Convert to list
                for JSON serialization
            "type": perm_type,
            "gridstate": perm,
            "alexander-grading": alex
        }

    # BFS through commutation space
    current_states = {tuple(vertlist)}
```

```
    visited_states = set(current_states)

    for iteration in range(n):
        new_states = set()

        for state in current_states:
            commuted_states = knot_commute(list(state))

            for commuted in commuted_states:
                commuted_tuple = tuple(commuted)

                if commuted_tuple not in visited_states:
                    visited_states.add(commuted_tuple)

                    # Try to find perfect grid state
                    perm_result = try_permutations(commuted
                        )
                    if perm_result:
                        vlist_out, perm_type, perm, matrix,
                            alex = perm_result
                        return {
                            "vlist": vlist_out,
                            "matrix": matrix.tolist(),
                            "type": perm_type,
                            "gridstate": perm,
                            "alexander-grading": alex
                        }

                    new_states.add(commuted_tuple)

        current_states = new_states

        if not current_states:
            break  # No new states to explore

    return None
```

**Code 6.7.** A BFS-algorithm to search for nice diagrams of a knot.

This algorithm tries to find a nice grid diagram by creating a large list of grid diagram representations using only commutation moves. If the algorithm is successful, the resulting grid diagram will be minimal as the input also was minimal and commutation moves do not change the grid size. At $n = 100$ commutation iterations this approach succesfully finds nice grid diagrams for 4346 of the 5397 fibered prime knots.

For the remaining knots we try a similar algorithm after performing a stabilization. From Proposition 2.12 we know that any stabilization can be obtained by a stabilization of type X-SW followed by a sequence of commutations. Analogously

it is also enough to perform only stabilizations of type X-NW in combination with commutations. The following function performs a X-NW stabilization of a grid diagram in vertlist-notation on a specified segment.

```
    def x_nw(vertlist: VertList, loc: Tuple[int, int]) ->
        VertList:
    """
    Perform an X-NW stabilization at specified segment.

    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
        Vertical segment list.
    loc : Tuple[int, int]
        Segment where to perform stabilization.

    Returns
    -------
    List[Tuple[int, int]]
        Stabilized vertical list.
    """

    if loc not in vertlist:
        raise ValueError(f"Segment {loc} not in vertical
            list")

    k = vertlist.index(loc)
    temp = [list(tpl) for tpl in vertlist]

    # Shift indices greater than loc[0]
    for segment in temp:
        for j in range(len(segment)):
            if segment[j] > loc[0]:
                segment[j] += 1

    # Insert new segment and modify existing one
    temp.insert(k + 1, [loc[0], loc[0] + 1])
    temp[k][0] = loc[0] + 1

    return [tuple(segment) for segment in temp]
```

**Code 6.8.** Performing a stabilization of type X-NW on a grid diagram in vertlist-notation.

The second BFS-algorithm, to find nice diagrams of knots after performing one stabilization, looks as follows.

```
    def gridstate_finder_stab(vertlist: VertList, n: int)
        -> Optional[Dict]:
```

```
    """
    Find unique perfect grid states by trying
       stabilizations.

    Maintains a global visited set across all stabilization
        attempts
    to avoid recomputing states.

    Parameters
    ----------
    vertlist : List[Tuple[int, int]]
        Initial vertical segment list.
    n : int
        Maximum number of commutation iterations after
           stabilization.

    Returns
    -------
    Optional[Dict]
        Dictionary containing grid state information if
           found, None otherwise.
    """

    # Global visited set for all stabilization attempts
    global_visited = {tuple(vertlist)}

    # Try each stabilization
    for segment in vertlist:
        try:
            stab_vertlist = x_nw(vertlist, segment)
            stab_tuple = tuple(stab_vertlist)

            # Skip if we've seen this stabilized state
               before
            if stab_tuple in global_visited:
                continue

            # Custom BFS that uses the global visited set
            result = _gridstate_finder_commute_with_visited
               (
                stab_vertlist, n, global_visited
            )

            if result:
                # Add stabilization info
                result["stabilizations"] = 1
                return result

        except Exception:
```

```
            continue

    return None


def _gridstate_finder_commute_with_visited(
    vertlist: VertList,
    n: int,
    global_visited: Set[Tuple[Tuple[int, int], ...]]
    ) -> Optional[Dict]:
    """
    Helper function: gridstate_finder_commute that respects
        a global visited set.
    """

    # Check initial state
    perm_result = try_permutations(vertlist)
    if perm_result:
        vlist_out, perm_type, perm, matrix, alex =
            perm_result
        return {
            "vlist": vlist_out,
            "matrix": matrix.tolist(),
            "type": perm_type,
            "gridstate": perm,
            "alexander-grading": alex
        }

    # BFS with shared visited set
    current_states = {tuple(vertlist)}

    for iteration in range(n):
        new_states = set()

        for state in current_states:
            commuted_states = knot_commute(list(state))

            for commuted in commuted_states:
                commuted_tuple = tuple(commuted)

                if commuted_tuple not in global_visited:
                    global_visited.add(commuted_tuple)

                    perm_result = try_permutations(commuted
                        )
                    if perm_result:
                        vlist_out, perm_type, perm, matrix,
                            alex = perm_result
                        return {
```

```
                                "vlist": vlist_out,
                                "matrix": matrix.tolist(),
                                "type": perm_type,
                                "gridstate": perm,
                                "alexander-grading": alex
                            }

                        new_states.add(commuted_tuple)

            current_states = new_states

            if not current_states:
                break

    return None
```

**Code 6.9.** A BFS-algorithm to search for nice diagrams of a knot after one stabilization.

This algorithm is noticeably slower than the commutation-only algorithm. Most likely, this is caused by the fact that a stabilization introduces a segment of length one that can be commuted with everything, thus increasing the number of possible commutations drastically. Nevertheless it succesfully found nice grid diagrams for 1039 knots of the remaining knots. This only leaves 12 fibered prime with crossing number $\leq 13$ as unsolved.

| **Crossing number** | **Unsolved knots** |
|---|---|
| $\leq 11$ | $\emptyset$ |
| 12 | $12n_{79}$, $12n_{168}$ |
| 13 | $13n_{282}$, $13n_{917}$, $13n_{1279}$, $13n_{1281}$, $13n_{1413}$, $13n_{1826}$, $13n_{2915}$, $13n_{3089}$, $13n_{3904}$, $13n_{3932}$ |

**Table 6.1.** Unsolved knots by crossing number

Coincidentally, for all unsolved knots the minimal grid size coincides with the crossing number. In the full dataset 35% of the knots have a minimal grid size of 14 or 15. For all of those knots nice grid diagrams have been found, cf. Figure 6.1 for two examples of nice grid diagrams of grid size 16, i.e. these are examples of knots of minimal grid size 15 where the code only found a nice grid diagram after stabilizing once. For the unsolved knots, the introduction of a second stabilization did not lead to any nice grid diagrams. However, to properly investigate grid diagrams where more than one stabilization has been performed, it would be necessary to make the alogrithms much more efficient. One possibility is to use some form of multiprocessing. Within the project, multiprocessing has already been used to find nice grid diagrams for several knots simultaneously. Nevertheless, for

the most time consuming knots a parallelized version of the BFS algorithms would be much faster. It remains an interesting task to create more efficient versions of our code. As of right now, the functionalities of the code are also limited, some of the missing features are:

- The code is only able to perform stabilizations of type $X-NW$. While this is sufficient in theory, introducing more stabilization types may decrease the number of necessary commutations drastically.
- The code does not destabilize diagrams.
- We are not looking at 2 or more stabilizations.

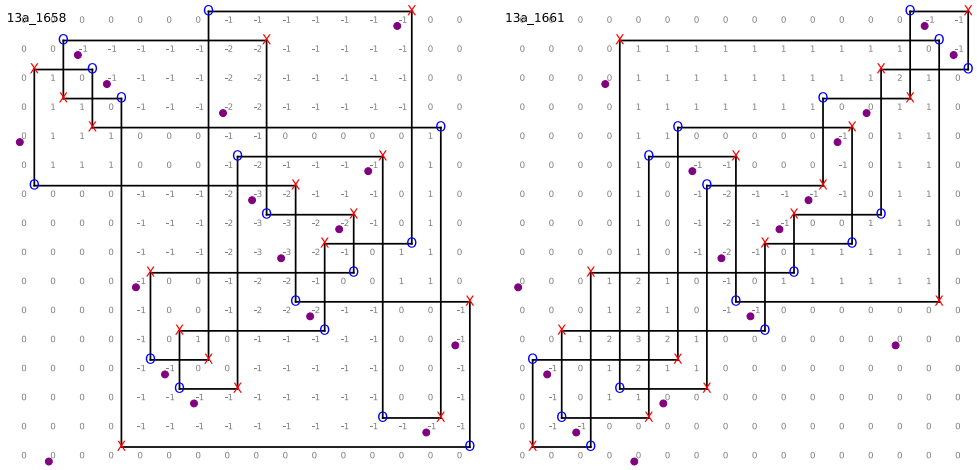



**Figure 6.1.** Two examples of nice grid diagrams for fibered prime knots of minimal grid size 15 where one stabilization has been performed. The purple dots represent the corresponding perfect grid state.

## 6.3 Final Remarks and Questions

Even without the introduction of destabilizations and using at most one stabilization on a minimal grid diagram, this simple python package was able to find nice diagrams of 5384 of the 5397 fibered prime knots from the database. This suggests the following question.

**Question 6.1.** Does every knot that admits a grid diagram with unique maximal grid state, also admit a nice grid diagram?

This project has lead to many more interesting questions, for example:

- For those knots where nice grid diagrams where found only after stabilizing, do there exist nice grid diagrams of minimal grid size? We did not introduce destabilizations in the package. This may be related to the question if all grid diagrams of a knot of the same size are connected using commutations only, or if it is necessary to stabilize and destabilize. In [Dyn06] Dynnikov proved that any grid diagram of the unknot can be reduced to the simplest diagram without stabilizing, however in the appendix of [Men06] an example of a grid diagram is given that can not be reduced without stabilizing.
- Assuming a knot has a minimal grid diagram with a unique maximal grid state as well as a nice grid diagram, does there exist a minimal nice grid diagram? So far the code did not find a minimal nice grid diagram for $9_{42}$, but Figure 5.3 is an example of a minimal grid diagram for $9_{42}$ with a unique maximal grid state.

We end this project with a list of remarks.

- Now that nice grid diagrams have been found for a large list of fibered prime knots it might be interesting to use machine learning methods in order to find more efficient methods of finding nice grid diagrams. As of right now, there is no measurement of how far away a grid diagram is from being nice, maybe ML-methods can even help at finding such a metric.
- The package checks if the Alexander grading of the unique perfect grid state coincides with the 3-genus of the input knot. According to Lemma 5.5 this has to be the case. For all knots where a nice grid diagram has been found, the grading of the grid state coincided with the genus of the knot. This is a very good sign that the code is producing correct results. Additionally one can compare the HFK-ranks of the resulting grid diagram with the original knot using for example Gridlink [Cul], to further ensure that the calculations are correct. This has been done

Thank you.

# Bibliography


[DH10] Max Dehn and Poul Heegaard. “Analysis situs”. In: *Encyklopädie der Mathematischen Wissenschaften mit Einschluss ihrer Anwendungen: Dritter Band: Geometrie*. Springer, 1910, pp. 153–220.

[Cro95] Peter R Cromwell. “Embedding knots and links in an open book I: Basic properties”. In: *Topology and its Applications* 64.1 (1995), pp. 37–58.

[Cro04] Peter R Cromwell. *Knots and links*. Cambridge university press, 2004.

[OS04] Peter Ozsváth and Zoltán Szabó. “Holomorphic disks and genus bounds”. In: *Geometry & Topology* 8.1 (2004), pp. 311–334.

[Ros04] Dennis Roseman. “Elementary moves for higher dimensional knots”. eng. In: *Fundamenta Mathematicae* 184.1 (2004), pp. 291–310. URL: http://eudml.org/doc/283078.

[Dyn06] Ivan Dynnikov. “Arc-presentations of links: monotonic simplification”. In: *Fundamenta Mathematicae* 1.190 (2006), pp. 29–76.

[Men06] William W Menasco. “An addendum on iterated torus knots”. In: *arXiv preprint math/0610566* (2006).

[Ni07] Yi Ni. “Knot Floer homology detects fibred knots”. In: *Inventiones mathematicae* 170.3 (2007), pp. 577–608.

[Ghi08] Paolo Ghiggini. “Knot Floer homology detects genus-one fibred knots”. In: *American journal of mathematics* 130.5 (2008), pp. 1151–1169.

[MOS09] Ciprian Manolescu, Peter Ozsváth, and Sucharit Sarkar. “A combinatorial description of knot Floer homology”. In: *Annals of Mathematics* (2009), pp. 633–660.

[Sav11] Nikolai Saveliev. *Lectures on the topology of 3-manifolds: an introduction to the Casson invariant*. Walter de Gruyter, 2011.

[Rei13] Kurt Reidemeister. *Knotentheorie*. Vol. 1. Springer-Verlag, 2013.

[OSS15] Peter S Ozsváth, András I Stipsicz, and Zoltán Szabó. *Grid homology for knots and links*. Vol. 208. American Mathematical Soc., 2015.

[LM24] Charles Livingston and Allison H. Moore. *KnotInfo: Table of Knot Invariants*. 2024. URL: knotinfo.math.indiana.edu (visited on 12/15/2024).

[Cul] Marc Culler. *Gridlink: A graphical tool for working with grid diagrams of knots*. URL: https://github.com/3-manifolds/gridlink (visited on 12/15/2024).

[ser] seriousmathsandunicorns. *Fun drawing and fibred knots*. URL: https://seriousmathsandunicorns.wordpress.com/2016/04/15/fun-drawing-and-fibred-knots/ (visited on 07/12/2025).