A NEW COMPUTATIONAL TOOL FOR KHOVANOV COBORDISM MAPS

ZSOMBOR FEHÉR

Abstract. We describe a Python module that we developed to calculate cobordism maps induced on Khovanov homology. As applications of our program, we compute these maps for all incompressible Seifert surfaces for prime knots up to 10 crossings, and distinguish many ribbon disks arising from symmetries of the boundary knots.

Introduction

Khovanov homology assigns an algebraic object $\mathrm{Kh}(L)$ to every oriented link $L \subseteq S^3$. Computer programs [KJob, Sage] are available to calculate this invariant, which can be used to distinguish links.

To any smoothly embedded surface $\Sigma \subseteq S^3 \times [0,1]$ between two links L_0 and L_1, Khovanov homology assigns a map $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(L_0) \to \mathrm{Kh}(L_1)$. These maps provide a method to distinguish surfaces up to smooth isotopy. However, despite the combinatorial nature of their definition, no computational tool was available for calculating these maps. We developed a program to compute them, as detailed in Section 2. This Python module is available on GitHub [KhCC].

In Section 3, we present two applications. We calculate the cobordism maps for all incompressible Seifert surfaces for prime knots up to 10 crossings (Table 5), and distinguish many ribbon disks arising from symmetries of the boundary knots (Table 6). For instance, we present five distinct ribbon disks for the knot 10_{123}, obtained via a rotational symmetry, which are distinguished by Khovanov homology (Figure 1).

1. Background

We give a brief overview of the definition of Khovanov homology, following [HS24]. Our focus is on the combinatorial details that are crucial for implementing the computer program. For a broader discussion of the context and philosophy underlying Khovanov homology, see [BN05, Kho00]. Throughout, we work with $\mathbb{Z}$ coefficients.

1.1. Khovanov homology of links.

Let $L \subseteq S^3$ be an oriented link. Choose a diagram D of L, and an enumeration $X = (x_1, \dots, x_n)$ of its crossings. We define a chain complex $(\mathcal{C}\mathrm{Kh}(D,X), d)$ associated to the data (D,X) as follows.

For any binary sequence $\sigma = (\sigma_1, \dots, \sigma_n) \in \{0,1\}^n$, we consider the *smoothing* D_σ, which is obtained by replacing each crossing x_i  with a 0-smoothing  or 1-smoothing  , depending on σ_i. We call the connected components of D_σ *loops*. A *labelled smoothing* α_σ is a labelling of the loops of D_σ with a 1 or x. Let $\mathcal{C}\mathrm{Kh}(D,X)$ be the free $\mathbb{Z}$-module generated by all the labelled smoothings α_σ of D.

For simplicity, we describe the differential map d using the Morse induced chain maps that are defined in Table 1 (taken from [HS24]). Let α_σ be a labelled smoothing with $\sigma = (\sigma_1, \dots, \sigma_n)$. For an i with $\sigma_i = 0$, let m_i denote the chain map induced by the

Morse move	Ornament	Definition of chain map
birth		$1 \mapsto 1$
death		$1 \mapsto 0$ $x \mapsto 1$
saddle		$1\,1 \mapsto 1$ $1\,x \mapsto x$ $x\,1 \mapsto x$ $x\,x \mapsto 0$
		$1 \mapsto 1\,x + x\,1$ $x \mapsto x\,x$

TABLE 1. Chain maps induced by a Morse move.

Morse saddle move $\asymp$ that turns the 0-smoothing of x_i into the 1-smoothing. (Thus, $m_i(\alpha_\sigma)$ is a sum of 0, 1, or 2 labelled smoothings.) We define the differential map $d\colon \mathcal{C}\mathrm{Kh}(D,X) \to \mathcal{C}\mathrm{Kh}(D,X)$ on the generators by

$$d(\alpha_\sigma) = \sum_{\substack{i \\ \sigma_i=0}} (-1)^{\xi_i} m_i(\alpha_\sigma), \qquad \xi_i = \sum_{\substack{j \\ j<i}} \sigma_j,$$

and extend it linearly to $\mathcal{C}\mathrm{Kh}(D,X)$. Then it can be verified that $d^2 = 0$, showing that $(\mathcal{C}\mathrm{Kh}(D,X), d)$ is indeed a chain complex. Its homology is denoted by $\mathrm{Kh}(D,X)$. It turns out that up to isomorphism, $\mathrm{Kh}(D,X)$ is unaffected by our particular choices of D and X [Kho00], and thus, it can be denoted by $\mathrm{Kh}(L)$, the *Khovanov homology* of the link L.

The chain complex $\mathcal{C}\mathrm{Kh}(D,X)$ is bigraded by homological grading h and quantum grading q. Let n_+, n_- be the number of positive and negative crossings in D. For a labelled smoothing α_σ, let $|\sigma|$ be the number of 1-smoothings in σ, and let $v_+(\alpha_\sigma), v_-(\alpha_\sigma)$ be the number of 1-labels and x-labels in α_σ. The grading (h,q) of α_σ is defined by

$$\begin{aligned} h(\alpha_\sigma) &= |\sigma| - n_-, \\ q(\alpha_\sigma) &= v_+(\alpha_\sigma) - v_-(\alpha_\sigma) + h(\alpha_\sigma) + n_+ - n_-, \end{aligned}$$

and $\mathcal{C}\mathrm{Kh}^{h,q}(D,X)$ is defined as the submodule generated by such α_σ. Then the differential map is also bigraded, $d\colon \mathcal{C}\mathrm{Kh}^{h,q}(D,X) \to \mathcal{C}\mathrm{Kh}^{h+1,q}(D,X)$. Hence, the bigrading descends to homology, and we have that $\mathrm{Kh}(L) = \bigoplus_{h,q} \mathrm{Kh}^{h,q}(L)$ is a bigraded $\mathbb{Z}$-module.

1.2. **Crossing order change.** We explicitly construct the isomorphism of $\mathcal{C}\mathrm{Kh}(D,X)$ for a different choice of X, as this will be essential for computations. Consider two enumerations $X = (x_1,\dots,x_n)$ and $X' = (x_{\pi(1)},\dots,x_{\pi(n)})$ for some permutation π. Then any labelled smoothing $\alpha_\sigma \in \mathcal{C}\mathrm{Kh}(D,X)$ with $\sigma = (\sigma_1,\dots,\sigma_n)$ naturally corresponds to a labelled smoothing $\alpha'_{\sigma'} \in \mathcal{C}\mathrm{Kh}(D,X')$ that has $\sigma' = (\sigma_{\pi(1)},\dots,\sigma_{\pi(n)})$ and the same labels. By definition, the differential d' on $\mathcal{C}\mathrm{Kh}(D,X')$ is given by

$$d'(\alpha'_{\sigma'}) = \sum_{\substack{i\\ \sigma_i=0}} (-1)^{\xi'_i} m_i(\alpha'_{\sigma'}), \qquad \xi'_i = \sum_{\substack{j\\ \pi^{-1}(j)<\pi^{-1}(i)}} \sigma_j.$$

We define the isomorphism $\Phi\colon \mathcal{C}\mathrm{Kh}(D,X) \to \mathcal{C}\mathrm{Kh}(D,X')$ on the generators by

$$\Phi(\alpha_\sigma) = (-1)^{\pi_\sigma}\alpha'_{\sigma'}, \qquad \pi_\sigma = \sum_{\substack{j,k\\ j<k\\ \pi^{-1}(j)>\pi^{-1}(k)}} \sigma_j\sigma_k,$$

and extend it linearly. Then it can be checked that $\Phi(d(\alpha_\sigma)) = d'(\Phi(\alpha_\sigma))$. Therefore, $(\mathcal{C}\mathrm{Kh}(D,X),d)$ and $(\mathcal{C}\mathrm{Kh}(D,X'),d')$ are isomorphic chain complexes, and $\mathrm{Kh}(D,X) \cong \mathrm{Kh}(D,X')$.

We also provide a more intuitive perspective on this isomorphism, by describing the differential map in a way that is independent of the crossing order, as follows. For a labelled smoothing α_σ, let $i_1 < \dots < i_k$ represent all indices i where $\sigma_i = 1$, and replace α_σ with the formal expression $\alpha_\sigma \cdot \omega_\sigma$, where $\omega_\sigma = dx_{i_1} \wedge \dots \wedge dx_{i_k}$. By substituting each labelled smoothing (α_σ and every $m_i(\alpha_\sigma)$) in this way within the definition of d, the differential map can be written formally as

$$d = m_1 \cdot dx_1 + \dots + m_n \cdot dx_n,$$

where $m_i \cdot dx_i$ acts on a labelled smoothing as

$$(m_i \cdot dx_i)(\alpha_\sigma \cdot \omega_\sigma) = m_i(\alpha_\sigma) \cdot (dx_i \wedge \omega_\sigma).$$

Indeed, the summation restricts to the indices i with $\sigma_i = 0$ because $dx_i \wedge dx_i = 0$, and the signs $(-1)^{\xi_i}$ in our original definition of d come from $dx_i \wedge dx_j = -dx_j \wedge dx_i$, when we change $dx_i \wedge \omega_\sigma$ to have increasing indices (as $m_i(\alpha_\sigma)$ has a new 1-smoothed crossing x_i).

With this order-independent view of d, it is clear what happens when we change the order of crossings to X'. The new $\omega'_{\sigma'}$ that uses the ordering X' has $\omega'_{\sigma'} = \pm\omega_\sigma$, where the $\pm$ sign depends on the parity of the permutation of $dx_{i_1},\dots,dx_{i_k}$ in $\omega'_{\sigma'}$. In the above definition of Φ, α_σ gets this sign, whose explicit definition is $(-1)^{\pi_\sigma}$. For example, if $X = (x_1,x_2,x_3)$ and $X' = (x_1,x_3,x_2)$, then

$$\Phi(\alpha_{110} + \alpha_{101} + \alpha_{011}) = \alpha'_{101} + \alpha'_{110} - \alpha'_{011}.$$

1.3. **Khovanov homology of cobordisms.** Let L_0, L_1 be links. A *cobordism* from L_0 to L_1 is a smoothly embedded compact surface $\Sigma \subseteq S^3 \times [0,1]$ whose boundary is $L_0 \times \{0\} \cup L_1 \times \{1\}$. Any cobordism can be represented as a *movie*: a sequence of diagrams $D_0,\dots,D_N$, where D_{i+1} is obtained from D_i by an *elementary move* (Reidemeister or

Morse move), and D_0, D_N represent the links L_0, L_1, respectively. We often write the cobordism as a function $\Sigma\colon L_0 \to L_1$.

Given a cobordism $\Sigma\colon L_0 \to L_1$, we will associate to Σ a group homomorphism $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(L_0) \to \mathrm{Kh}(L_1)$. Moreover, if Σ is orientable, with oriented boundary $\partial\Sigma = (L_0 \times \{0\}) \cup (L_1^r \times \{1\})$ (where L_1^r denotes L_1 with reversed orientation), then $\mathrm{Kh}(\Sigma)$ will be a bigraded map:

$$\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}^{h,q}(L_0) \to \mathrm{Kh}^{h,q+\chi(\Sigma)}(L_1).$$

We now give the definition of this map, using the movie $D_0, \dots, D_N$. We do this by defining a chain map $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ associated to every Reidemeister and Morse move, and $\mathcal{C}\mathrm{Kh}(\Sigma)$ will be the composition of these maps.

Morse moves. There are three types of Morse moves: *births*, *deaths*, and *saddles*. A birth ⌑ adds a small new loop to the diagram, a death ⊗ removes a small loop, and a saddle ⨝ merges or splits loops, by changing)(to ⋌⋋ locally.

Let X_i be an enumeration of the crossings of D_i. If D_{i+1} is obtained from D_i by a Morse move, then the list of crossings does not change. We let $X_{i+1} = X_i$, and we define the chain map $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ on labelled smoothings according to Table 1.

Reidemeister moves. There are three types of Reidemeister moves: R1 removes or adds 1 crossing, R2 removes or adds 2 crossings, and R3 manipulates the diagram around 3 crossings. We define the Reidemeister induced chain maps using the Morse induced chain maps defined previously. In addition, we will use ⨟ to denote two saddle moves on an arc (one splitting, then one re-merging). Then $\frac{1}{2}$ ⨟ sends a 1-labelled loop to an x-labelled loop, and an x-labelled loop to 0.

Let X_i be an enumeration of the crossings of D_i. If D_{i+1} is obtained from D_i by an R1 move that adds a crossing x, then add x to the enumeration in the last position: $X_{i+1} = (X_i, x)$. We define the chain map $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ on labelled smoothings according to Table 2. If the R1 move removes a crossing x, then consider first an enumeration X_i' that has x as its last element, and apply the isomorphism $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_i, X_i')$ described in Section 1.2. Then we define $\mathcal{C}\mathrm{Kh}(D_i, X_i') \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ according to the same Table, where $X_i' = (X_{i+1}, x)$.

If D_{i+1} is obtained from D_i by an R2 move that adds the crossings x_1, x_2, then let $X_{i+1} = (X_i, x_1, x_2)$, and define the chain map $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ on labelled smoothings according to Table 3. If the R2 move removes two crossings x_1, x_2, then first apply an isomorphism $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_i, X_i')$ that moves x_1, x_2 to the last two positions in X_i', then define $\mathcal{C}\mathrm{Kh}(D_i, X_i') \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ according to the same Table, where $X_i' = (X_{i+1}, x_1, x_2)$.

Finally, assume that D_{i+1} is obtained from D_i by an R3 move. The chain maps are presented in table form (Table 4), interpreted as follows. (These tables are adapted from [HS24], but with corrected signs.) The top-left cell of each table depicts the Reidemeister move, showing the starting diagram with crossings x_1, x_2, x_3 in the left corner, and the ending diagram with crossings y_1, y_2, y_3 in the right corner. As usual, we first apply an isomorphism $\mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_i, X_i')$, which reorders the crossings x_1, x_2, x_3 to the last three positions. Write $X_i' = (X, x_1, x_2, x_3)$, and let $X_{i+1} = (X, y_1, y_2, y_3)$. The map $\mathcal{C}\mathrm{Kh}(D_i, X_i') \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ is then defined as follows. The left column of the table lists the smoothings of the starting diagram, while the top row lists the smoothings of the ending diagram. The image of a labelled smoothing is obtained by adding together

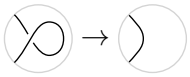
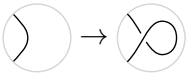
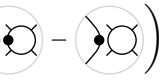
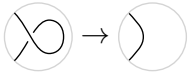
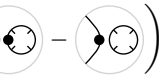
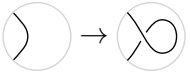
Reidemeister move	Smoothing	Induced map
		
		0
		
		
		0
		

TABLE 2. Chain maps induced by a Reidemeister 1 move.

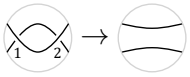
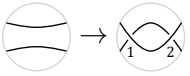
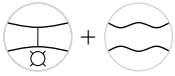
Reidemeister move	Smoothing	Induced map
		
		
		0
		0
		

TABLE 3. Chain maps induced by a Reidemeister 2 move.

the non-empty cells in its corresponding row, where I denotes an isotopy of the labelled smoothing.

This completes the definition of the maps induced by elementary moves. Now, let $\Sigma\colon L_0 \to L_1$ be a cobordism with movie $D_0, \dots, D_N$, and let X_0 and X'_N be enumerations of the crossings of D_0 and D_N, respectively. We obtain enumerations $X_1, \dots, X_N$ and maps $\phi_i\colon \mathcal{C}\mathrm{Kh}(D_i, X_i) \to \mathcal{C}\mathrm{Kh}(D_{i+1}, X_{i+1})$ from the description above. We define

$$\mathcal{C}\mathrm{Kh}(\Sigma, X_0, X'_N) = \Phi \circ \phi_{N-1} \circ \cdots \circ \phi_0\colon \mathcal{C}\mathrm{Kh}(D_0, X_0) \to \mathcal{C}\mathrm{Kh}(D_N, X'_N),$$

	000	100	010	001	110	101	011	111
000	I							
100		I		I				
010			I					
001								
110					I			
101							$-I$	
011								
111								

	000	100	010	001	110	101	011	111
000								
100			I					
010								
001				I				
110							I	
101						I		
011							I	
111								I

TABLE 4. Chain maps induced by a Reidemeister 3 move.

where $\Phi\colon \mathcal{C}\mathrm{Kh}(D_N, X_N) \to \mathcal{C}\mathrm{Kh}(D_N, X'_N)$ is the isomorphism induced by reordering the crossings. It can be checked that any ϕ_i is a chain map. Therefore, $\mathcal{C}\mathrm{Kh}(\Sigma, X_0, X'_N)$ is also a chain map, which induces a map on homology: $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(L_0) \to \mathrm{Kh}(L_1)$.

The power of using Khovanov homology for cobordisms comes from the fact that the map $\mathrm{Kh}(\Sigma)$ is invariant (up to a global sign) under smooth isotopy of Σ relative to the boundary [Jac04]. In fact, the same holds if we only look at a diffeomorphism of the ambient space:

Theorem 1 ([LS22, Proposition 3.7])**.** *Let $\Sigma \subseteq S^3 \times [0,1]$ be a cobordism from L_0 to L_1. Then the map $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(L_0) \to \mathrm{Kh}(L_1)$ is invariant up to multiplication by ± 1 under any diffeomorphism of $S^3 \times [0,1]$ that restricts to the identity on the boundary.*

As often is the case, if $\Sigma \subseteq D^4$ is a slice surface bounded by the link $L \subseteq S^3$, then Σ can be viewed as a cobordism $\Sigma\colon L \to \emptyset$, or as $\Sigma^\star\colon \emptyset \to L$. Thus, to distinguish the surfaces $\Sigma_0, \Sigma_1 \subseteq D^4$ smoothly, it suffices to distinguish the maps $\mathrm{Kh}(\Sigma_i)$, or the maps $\mathrm{Kh}(\Sigma_i^\star)$. Recent examples in the literature have demonstrated the use of this method to distinguish surfaces smoothly or exotically [SS22, HS24, HKM+22, Hay23].

If $\Sigma\colon \emptyset \to L$ is a cobordism, then the map $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(\emptyset) \to \mathrm{Kh}(L)$ is entirely determined by the image of $1 \in \mathbb{Z} = \mathrm{Kh}(\emptyset)$. We call this homology class $\mathrm{Kh}(\Sigma)(1)$ the *Khovanov–Jacobsson class* of Σ.

To determine whether two Khovanov–Jacobsson classes are the same up to sign, part (b) of the following well-known statement often provides a simple diagrammatic proof that a chain element is non-zero in homology.

Proposition 2. *Let $\alpha \in \mathcal{C}\mathrm{Kh}(D, X)$ be a chain element, given as a linear combination $\alpha = n_1\alpha_1 + \cdots + n_s\alpha_s$ of distinct labelled smoothings α_i, $0 \neq n_i \in \mathbb{Z}$.*

(a) *If, for every α_i, each 0-smoothed crossing in α_i connects two distinct x-labelled loops, then $d\alpha = 0$, i.e. α is a cycle.*
(b) *If there exists an α_i where each 1-smoothed crossing in α_i connects two distinct 1-labelled loops, then $\alpha \neq d\beta$ for any β, i.e. α is not a boundary.*

Proof. This is straightforward from the definition of d and the Morse saddle induced chain maps (Table 1). □

1.4. **Mirroring.** Let $-L$ denote the mirror of the link L, and $-D$ the mirror of the diagram D, obtained by swapping overstrands and understrands at every crossing. Then there is an isomorphism of chain complexes $\mathcal{C}\mathrm{Kh}(-D, X) \cong \mathcal{C}\mathrm{Kh}(D, X)^\star$, where $C^\star = \mathrm{Hom}(C, \mathbb{Z})$ is the dual of the $\mathbb{Z}$-module C [Kho00, Proposition 32]. This isomorphism Ψ_D is given by $\alpha_\sigma \mapsto (\beta_{-\sigma})^\star$, where $(-\sigma)_i = 1 - \sigma_i$ is the same smoothing on the mirrored diagram, $\beta_{-\sigma}$ is obtained from α_σ by swapping 1-labels and x-labels, and for a $\mathbb{Z}$-module C freely generated by $c, c_1, c_2, \ldots \in C$, the homomorphism $c \mapsto 1, c_i \mapsto 0$ is denoted by $c^\star \in C^\star$.

If $f\colon C \to B$ is a map, then the dual $f^\star\colon B^\star \to C^\star$ is defined by $\phi \mapsto \phi \circ f$ for $\phi \in B^\star$. Then it is straightforward to check that $\Psi_D \circ d_{-D} = d_D^\star \circ \Psi_D$, showing the isomorphism of chain complexes $(\mathcal{C}\mathrm{Kh}(-D, X), d_{-D})$ and $(\mathcal{C}\mathrm{Kh}(D, X)^\star, d_D^\star)$. Taking homology, this shows that $\mathrm{Kh}(-L)$ is related to $\mathrm{Kh}(L)$ in the usual way that homology is related to cohomology. If Tor denotes the torsion subgroup, this means that, since Ψ multiplies both gradings by -1,

$$\mathrm{Kh}^{h,q}(-L) \otimes \mathbb{Q} \cong \mathrm{Kh}^{-h,-q}(L) \otimes \mathbb{Q},$$
$$\mathrm{Tor}(\mathrm{Kh}^{h,q}(-L)) \cong \mathrm{Tor}(\mathrm{Kh}^{1-h,-q}(L)).$$

Let $\Sigma\colon L_0 \to L_1$ be a cobordism. The reverse of the cobordism is $\Sigma^\star\colon L_1 \to L_0$, and the mirror of the cobordism is $-\Sigma\colon -L_0 \to -L_1$. Then the cobordism maps induced on Khovanov homology are compatible with the previous isomorphism. Specifically, the following diagram commutes up to chain homotopy[1]:

$$\begin{array}{ccc} \mathcal{C}\mathrm{Kh}(-D_0, X_0) & \xrightarrow[\Psi_{D_0}]{\cong} & \mathcal{C}\mathrm{Kh}(D_0, X_0)^\star \\ \Big\downarrow{\scriptstyle \mathcal{C}\mathrm{Kh}(-\Sigma)} & & \Big\downarrow{\scriptstyle \mathcal{C}\mathrm{Kh}(\Sigma^\star)^\star} \\ \mathcal{C}\mathrm{Kh}(-D_1, X_1) & \xrightarrow[\Psi_{D_1}]{\cong} & \mathcal{C}\mathrm{Kh}(D_1, X_1)^\star \end{array}$$

2. The computer program

We have developed a Python module that can calculate the map $\mathrm{Kh}(\Sigma)$ for any cobordism Σ (given as a movie). This addresses an important gap in the existing computational tools for Khovanov homology. The source code is available in the file `khovanov.py` on GitHub [KhCC], and is also included as an ancillary file in the arXiv version of this paper.

As a basic example, we use the following code to show that the two annuli A_0, A_1, into which the torus link $T_{4,6}$ divides the torus, are not smoothly isotopic relative to the boundary in D^4 [Feh23, Theorem 1].

```
1  from khovanov import *
2
3  L = Link(braid_closure = [1, 2, 3] * 6)
4  c, d = L.crossings[0], L.crossings[3]
5  L.orient((c, 0), (c, 1))
6
7  A0 = Cobordism(L)
8  A0.morse_saddle((c, 0), (c, 3))
9  A0.finish()
10
11 A1 = Cobordism(L)
12 A1.morse_saddle((d, 0), (d, 3))
13 A1.finish()
14
15 compare(A0.KJ_class(), A1.KJ_class())
```

Lines 3–5 define the link $L = T_{4,6}$ using a braid closure, denote two of its crossings c and d, and define its orientation (needed only when the link has more than one component). Lines 7–9 define the annulus A_0 combinatorially, viewed as a cobordism from L to $\emptyset$. Its movie starts with a Morse saddle move between two strands of the diagram that are identified by the 0th and 3rd strands around the crossing c. The next moves of the movie would then be added as subsequent function calls (e.g. `A0.morse_birth()`, `A0.reidemeister_2((d, 3), (d, 0))`, `A0.reidemeister_1_up((c, 2))`). In this example, however, the resulting link after the first move is the unknot, so we use the convenient `finish()` function to automatically finish the rest of the cobordism. Lines 11–13 define the cobordism A_1 similarly, starting with a different Morse saddle move. In

[1] It appears that a proof of this statement does not appear in the literature. It is stated in [Hay25, Sun22], but without a proper reference.

line 15, we calculate chain representatives of the Khovanov–Jacobsson classes of $A_0^\star, A_1^\star$, and compare them in homology. In this case, the program outputs that both their sum and difference are trivially outside the image of the differential map d (Proposition 2 (b)), giving a proof that A_0, A_1 are not smoothly isotopic in D^4 by Theorem 1.

2.1. **Classes and methods.** We provide a brief overview of the program's classes, highlighting key properties and methods that might be useful to users. The program builds on SnapPy's `Link` class [Snap], with modifications. This class represents a link diagram combinatorially as a graph in which each vertex has degree 4 or 2.

Crossing. A `Crossing` represents a crossing in the link diagram, modelled as a degree-4 vertex. The strands are labelled 0, 1, 2, and 3 in positive orientation around the vertex, with 0 and 2 corresponding to the understrand. Its properties:

- `adjacent` specifies the connection for each strand.
- `label` is used to identify a crossing within a `Link` (usually an integer).

For example, `a = Crossing(7)` creates a `Crossing` object with `label` 7, and connecting the strand `(a, 3)` to `(b, 0)` can be achieved by writing `a[3] = b[0]`.

Strand. A `Strand` is similar to a `Crossing`, but represents a degree-2 vertex. While SnapPy automatically fuses these and removes unknotted unlinked components, we retain them as they are essential for representing cobordism movies.

Link. A `Link` represents a link diagram with an enumeration of its crossings. Its property:

- `crossings` provides a list of `Crossing` and `Strand` objects that form the vertices of the graph.

Links can be created using PD codes, braid closures, or a list of `Crossing` and `Strand` objects, e.g. `L = Link([a, b, c])`. Its methods:

- `L.orient(cs0, cs1, ...)`: if the link L has multiple components, it may be necessary to define its orientation to set its `n_plus` and `n_minus` properties (representing the number of positive and negative crossings).
- `L.differential_matrix(h, q)` returns a matrix of the differential map d in grading $(h, q) \to (h + 1, q)$.
- `L.homology_with_generators(h, q)` calculates the Khovanov homology $\mathrm{Kh}^{h,q}(L)$, expressing it as a direct sum of cyclic groups and returning chain representatives for the generators of each group (this method needs Sage [Sage]).

For computing Khovanov homology without generators, significantly faster programs exist, such as KnotJob [KJob], which implements Bar-Natan's efficient algorithm [BN07].

CrossingStrand. A `CrossingStrand` represents one of the four strands around a `Crossing` (or one of the two strands around a `Strand`). Its properties:

- `crossing` references the associated `Crossing` or `Strand` object.
- `strand_index` indicates the strand's position, e.g. 0, 1, 2, or 3.

For example, `cs = CrossingStrand(a, 2)` sets `cs.crossing` to the object `a`, and `cs.strand_index` to 2.

`CrossingStrand` objects are crucial for defining cobordisms, as they allow precise specification of strands involved in an elementary move. Several methods are available to navigate within a `Link`:

- `cs + 1`, `cs.next()`, and `cs - 1` give the adjacent strands around the same crossing.
- `cs.opposite()` returns the strand located at the opposite end of the edge connected to `cs`.

It is important to understand that while `cs` and `cs.opposite()` lie on the same edge of the underlying graph of the link diagram, they represent distinct portions of the edge. Care must be taken to use the appropriate one when specifying strands for defining an elementary move.

SmoothLink. A `SmoothLink` represents a smoothing of a link diagram. Its properties:

- `link` refers to the underlying `Link`.
- `smoothing` is a dictionary mapping each `Crossing` to 0 or 1.
- `loops` is a list of loops, where each loop is represented as a tuple of `CrossingStrand` objects (two at each crossing) traversed along the loop.

LabelledSmoothing. A `LabelledSmoothing` represents a labelled smoothing with a coefficient. Its properties:

- `smooth_link` refers to the underlying `SmoothLink`.
- `labels` is a dictionary mapping each loop to `"1"` or `"x"`.
- `coefficient` is an integer that will represent the coefficient in a linear combination.

There are several methods for examining the relationship between a `LabelledSmoothing` `LS` and the image of the differential map (without computing the entire differential matrix):

- `LS.differential()` returns its differential as a list of `LabelledSmoothing` objects, interpreted as a linear combination.
- `LS.is_outside()` checks if every 1-smoothed crossing connects two distinct 1-labelled loops. If so, then `LS` lies outside the image of the differential by Proposition 2 (b).
- `LS.row_size()` and `LS.reverse_differential()` give how many and which elements contain `LS` in their differential.

CKhElement. A `CKhElement` represents a chain element of $\mathcal{C}\mathrm{Kh}(D, X)$. It is a list of `LabelledSmoothing` objects, interpreted as a linear combination using their `coefficient` properties.

Each of the elementary moves can be applied to a `CKhElement` `CKH` by calling the corresponding method and specifying the location with appropriate `CrossingStrand` objects (see the source code for more details and pictures):

- `CKH.morse_birth()` adds a new loop.
- `CKH.morse_death(cs)` removes a loop containing `cs`.
- `CKH.morse_saddle(cs0, cs1)` adds a saddle move to connect `cs0` with `cs1`.
- `CKH.reidemeister_1(cs)` removes a twist containing `cs`.
- `CKH.reidemeister_1_up(cs, True)` adds a positive twist to the left of `cs`.
- `CKH.reidemeister_2(cs0, cs1)` removes the common crossing of `cs0` and `cs1`, and the common crossing of `cs0.opposite()` and `cs1.opposite()`.
- `CKH.reidemeister_2_up(cs0, cs1)` moves `cs0` to the right and over `cs1`.
- `CKH.reidemeister_3(cs0, cs1, cs2)` moves `cs0` over `cs1`, `cs2`, and the common crossing of `cs1.opposite()` and `cs2.opposite()`.

These change the underlying `LabelledSmoothing`, `SmoothLink`, and `Link` objects as well by the elementary move. Further methods:

- `CKH.fuse(strand)` fuses a `Strand`, combining the two edges from it.
- `CKH.differential()` returns the differential.
- `CKH.reorder_crossings(new_order)` reorders the crossings (see Section 1.2).
- `CKH.replace_link(link, flipping)` might be useful if `CKH` is on a different copy of the same link.
- `CKH.simplify()` simplifies the linear combination by combining equal terms and removing 0-coefficient elements.
- `CKH0 + CKH1` and `CKH0 - CKH1` calculate the sum and difference of two chain elements and simplify the result.
- `compare(CKH0, CKH1)` returns whether two chain elements are the same up to sign in homology.

Cobordism. A `Cobordism` represents a movie of a cobordism between two links. Its properties:

- `movie` contains a list of elementary moves and their locations.
- `links` gives the list of links corresponding to each stage of the movie.

A new cobordism $C\colon L \to L$ with empty movie can be created by `C = Cobordism(L)`, and moves can be added by subsequent calls of methods. Crossing labels can also be used in arguments instead of `CrossingStrand` objects, e.g. `C.morse_death((7, 1))`. In addition to the above methods of `CKhElement`, there are two convenient methods for adding multiple elementary moves at once: `band_move(...)` and `finish()`.

- `C.band_move(n, cs0, (cs1, True), (cs2, False), ..., csk)` adds a band with n half-twists, starting at `cs0`, passing over `cs1`, under `cs2`, and so on, before ending at `csk`.
- `C.finish()` tries to simplify the last link using Reidemeister 1–3 moves (similar to SnapPy's `simplify("level")` algorithm), then does a Morse death on each crossingless component.
- `C.reverse()` returns the reversal of the cobordism.
- `C.mirror()` returns the mirror of the cobordism.
- `C.chi()` returns the Euler characteristic of the cobordism.
- `C.map(CKH)` applies the map induced by the cobordism to `CKH` and changes it in place.
- `C.KJ_class()` returns the Khovanov–Jacobsson class of `C` or `C.reverse()`, if one of the ends of `C` is the empty link.
- `C.matrix(h, q)` calculates the cobordism map induced on homology in grading $(h, q) \to (h, q + \chi(C))$ in matrix form (this method needs Sage).

All classes support the standard `print(...)` method to display information about an object. For instance, `print(L)` outputs the adjacency structure of a `Link`, `print(C)` displays the sequence of moves in a `Cobordism` movie, and `print(CKH)` represents a `CKhElement` by showing the labels of each loop in every element of the linear combination. Additionally, `CKH.print_short()` provides a concise representation that omits printing the loops.

3. Applications

As two applications of our program, we compute cobordism maps for Seifert surfaces and ribbon disks bounded by small knots.

3.1. **Seifert surfaces up to 10 crossings.** For a prime knot L with at most 10 crossings, there is a unique incompressible Seifert surface Σ up to smooth isotopy in D^4 (see [Kak05] for the classification up to isotopy in S^3). Viewing Σ as a cobordism $L \to \emptyset$, the induced map on Khovanov homology is $\mathrm{Kh}(\Sigma)\colon \mathrm{Kh}(L) \to \mathrm{Kh}(\emptyset)$. As $\mathrm{Kh}(\emptyset) \cong \mathbb{Z}$ is supported in grading $(0,0)$, and $\mathrm{Kh}(\Sigma)$ shifts grading by $(0,\chi)$, where $\chi = \chi(\Sigma)$, the only non-zero part of this cobordism map is $\mathrm{Kh}^{0,-\chi}(L) \to \mathrm{Kh}^{0,0}(\emptyset)$. For all knots L considered, $\mathrm{Kh}^{0,-\chi}(L) \cong \mathbb{Z}^k$ is torsion-free. When $k \geq 1$, this map is represented by a $1 \times k$ matrix. As the isomorphism with $\mathbb{Z}^k$ is not canonical, the only information contained in this matrix is the greatest common divisor of its entries. We now calculate this non-negative integer for all such knots L and their mirror.

If D is an alternating diagram of L, then Seifert's algorithm produces a minimal genus Seifert surface for L [Gab86]. This algorithm constructs the surface from the orientation-preserving smoothing σ by attaching a half-twisted band at each crossing to D_σ. This method provides a straightforward way to algorithmically generate incompressible Seifert surfaces for most prime knots with at most 10 crossings.

For non-alternating knots, Seifert's algorithm often still yields minimal genus Seifert surfaces. Only 18 knots out of 249 required additional adjustments. For these cases, we repeatedly perturbed the diagram using Reidemeister moves (occasionally increasing the crossing number by 1 or 2) until we successfully obtained a minimal genus Seifert surface using Seifert's algorithm in every instance. The PD codes and minimal genus data were sourced from KnotInfo [KInfo].

Using this data of Seifert surfaces, we computed the cobordism maps in matrix form for both Σ and $-\Sigma$, the mirror of the surface (bounded by the mirror of the knot). Table 5 shows the resulting matrices, where an empty cell means $k = 0$, and an entry such as 2 0 0 indicates $k = 3$.

Viewing the cobordism Σ in the opposite direction, denoted $\Sigma^\star$, we obtain a map $\mathrm{Kh}^{0,0}(\emptyset) \to \mathrm{Kh}^{0,\chi}(L)$, which is determined by the Khovanov–Jacobsson class of $\Sigma^\star$. However, these maps $\mathrm{Kh}(\Sigma^\star)$ provide no new information, since $\mathcal{C}\mathrm{Kh}(-\Sigma)$ and $\mathcal{C}\mathrm{Kh}(\Sigma^\star)^\star$ are canonically related as described in Section 1.4.

There have been studies on the Khovanov–Jacobsson class of surfaces obtained via Seifert's algorithm [SS22, Ell10, Ell09], but only on the chain level rather than homology. These studies help interpret some of the results in Table 5, using the state graph Γ_σ of the Seifert smoothing σ of a diagram. This graph's vertices represent the loops of D_σ, while its edges correspond to the crossings of the link diagram connecting two loops. The edges are coloured red or blue based on whether the crossing is positive or negative (i.e. 0-smoothed or 1-smoothed in σ). The Khovanov–Jacobsson chain element's structure only depends on the Betti numbers b_0, b_1 of the red subgraph of Γ_σ [SS22].

3.2. **Ribbon disks from symmetries.** Let L be a ribbon knot with ribbon disk Σ and suppose that L has a non-trivial symmetry: a diffeomorphism $\varphi\colon S^3 \to S^3$ such that $\varphi(L) = L$ (disregarding orientation of L) and φ is not isotopic to the identity through diffeomorphisms that fix L setwise. Viewing Σ as an immersed surface in S^3 with ribbon singularities, consider $\varphi(\Sigma)$. This is also a ribbon disk for L, which often differs from Σ (up to smooth isotopy relative to the boundary in D^4). For instance, the two exotically

L	Σ	$-\Sigma$													
3_1	1		9_{16}	1		10_{17}			10_{67}	2 0		10_{117}			
4_1	2	2	9_{17}			10_{18}	2 0 0 0		10_{68}			10_{118}			
5_1	1		9_{18}	1		10_{19}			10_{69}			10_{119}			
5_2	1		9_{19}			10_{20}	2		10_{70}			10_{120}	1		
6_1	2	2 0	9_{20}	2 0		10_{21}	2 0		10_{71}			10_{121}			
6_2	2		9_{21}		2 0 0	10_{22}			10_{72}	2 0		10_{122}			
6_3			9_{22}			10_{23}			10_{73}			10_{123}			
7_1	1		9_{23}	1		10_{24}	2 0		10_{74}	2 0 0		10_{124}	1		
7_2	1		9_{24}			10_{25}	2 0		10_{75}			10_{125}			
7_3	1		9_{25}	2 0		10_{26}			10_{76}	2		10_{126}			
7_4	1		9_{26}			10_{27}			10_{77}			10_{127}	2		
7_5	1		9_{27}			10_{28}			10_{78}	2 0 0		10_{128}	1		
7_6	2 0		9_{28}			10_{29}			10_{79}			10_{129}			
7_7			9_{29}			10_{30}	2 0 0		10_{80}	1		10_{130}			
8_1	2	2 0	9_{30}			10_{31}			10_{81}			10_{131}	2		
8_2	2		9_{31}			10_{32}			10_{82}			10_{132}		2	
8_3	2 0	2 0	9_{32}			10_{33}			10_{83}			10_{133}	2		
8_4		2 0	9_{33}			10_{34}			10_{84}			10_{134}	1		
8_5	2		9_{34}			10_{35}			10_{85}			10_{135}			
8_6	2		9_{35}	1		10_{36}	2 0		10_{86}			10_{136}		0 0	
8_7			9_{36}		2 0	10_{37}			10_{87}			10_{137}			
8_8			9_{37}			10_{38}	2 0		10_{88}			10_{138}			
8_9			9_{38}	1		10_{39}	2 0		10_{89}			10_{139}	1		
8_{10}			9_{39}		2 0 0	10_{40}			10_{90}			10_{140}			
8_{11}	2 0		9_{40}			10_{41}			10_{91}			10_{141}			
8_{12}			9_{41}			10_{42}			10_{92}	2 0 0		10_{142}	1		
8_{13}			9_{42}		0	10_{43}			10_{93}			10_{143}			
8_{14}	2 0		9_{43}	2		10_{44}			10_{94}			10_{144}	2 0		
8_{15}	1		9_{44}			10_{45}			10_{95}			10_{145}		1	
8_{16}			9_{45}		2	10_{46}	2		10_{96}			10_{146}			
8_{17}			9_{46}	2 0	2	10_{47}			10_{97}	2 0 0		10_{147}	2 0 0		
8_{18}			9_{47}			10_{48}			10_{98}	2 0 0		10_{148}			
8_{19}	1		9_{48}		2 0 0	10_{49}	1		10_{99}			10_{149}	2		
8_{20}			9_{49}	1		10_{50}	2 0		10_{100}			10_{150}	2 0		
8_{21}	2		10_1	2	2 0	10_{51}			10_{101}	1		10_{151}			
9_1	1		10_2	2		10_{52}			10_{102}			10_{152}	1		
9_2	1		10_3	2 0	2 0 0	10_{53}	1		10_{103}			10_{153}			
9_3	1		10_4		2 0 0	10_{54}			10_{104}			10_{154}	1		
9_4	1		10_5			10_{55}	1		10_{105}			10_{155}			
9_5	1		10_6	2		10_{56}	2 0		10_{106}			10_{156}			
9_6	1		10_7	2 0		10_{57}			10_{107}			10_{157}		2	
9_7	1		10_8	2 0		10_{58}			10_{108}			10_{158}			
9_8	2 0 0		10_9			10_{59}			10_{109}			10_{159}			
9_9	1		10_{10}			10_{60}			10_{110}			10_{160}	2 0		
9_{10}	1		10_{11}	2 0 0		10_{61}	2 0		10_{111}	2 0 0		10_{161}	1		
9_{11}		2 0	10_{12}			10_{62}			10_{112}			10_{162}		2 0	
9_{12}	2 0		10_{13}			10_{63}	1		10_{113}			10_{163}			
9_{13}	1		10_{14}	2 0		10_{64}			10_{114}			10_{164}			
9_{14}			10_{15}			10_{65}			10_{115}			10_{165}		2	
9_{15}		2 0	10_{16}	2 0 0		10_{66}	1		10_{116}						

TABLE 5. Khovanov cobordism maps induced by the unique incompressible Seifert surfaces $\Sigma\colon L \to \emptyset$ and their mirror $-\Sigma\colon -L \to \emptyset$.

knotted ribbon disks in [HS24, Figure 1] and the annuli in [Feh23, Theorem 1] are both related by symmetry and distinguished smoothly by Khovanov homology.

In this section, we study ribbon disks obtained via symmetries for all prime ribbon knots with at most 9 crossings and for certain 10-crossing ribbon knots, using Khovanov homology.

The group of diffeomorphisms of the pair (S^3, L) up to isotopy is called the *symmetry group* of L. If L is hyperbolic, this group is finite, and can be computed from the hyperbolic triangulation of the knot complement. KnotInfo [KInfo] provides data on these symmetry groups, summarized in Table 6. Here, D_n denotes the dihedral group of $2n$ elements, corresponding to the symmetry group of a regular n-gon. The table also lists the number of different ribbon disks we identified, distinguished by Khovanov homology.

Knot	6_1	8_8	8_9	8_{20}	9_{27}	9_{41}	9_{46}	10_3	10_{123}
Symmetry group	D_2	D_2	D_4	D_1	D_2	D_3	D_2	D_2	D_{10}
Ribbon disks found	2	2	4	1	2	2	2	2	5

TABLE 6. Ribbon disks distinguished by Khovanov homology.

Knowing the symmetry group of a knot does not always make it straightforward to identify its symmetries. A knot L is called *reversible*, if there exists a symmetry φ that reverses the orientation of L. For any reversible prime knot with at most 10 crossings, there exists a *symmetric diagram*, where an orientation-reversing symmetry φ is realized as a 180° rotation, as listed in [Sak86, Lam22]. Every ribbon knot up to 10 crossings that admits a non-trivial symmetry is reversible [KInfo].

Eisermann–Lamm [EL11] provide band diagrams for one ribbon disk for each ribbon knot up to 10 crossings (given as symmetric union diagrams). Using KLO [KLO], we manually transformed these into symmetric diagrams to find additional ribbon disks derived from the symmetries. In certain cases (8_9 and 9_{27}), further manual manipulations were performed to search for the remaining symmetries. These ribbon disks were then compared using our Khovanov homology program. The source code for these computations is provided in the ancillary file `ribbon_disks.py` and is also available on GitHub [KhCC].

Figure 1 shows the band diagrams for ribbon disks obtained via symmetries, layered over the same knot diagram. With the exception of 10_{123}, each band diagram consists of a single band. Solid lines represent ribbon disks that are distinguished by Khovanov homology. Dotted lines indicate disks that are isotopic to others, while dashed lines represent disks that cannot be distinguished by Khovanov homology (both these disks and their mirrors share the same Khovanov–Jacobsson class with another disk in the figure). Disks sharing the same Khovanov homology are shown in the same colour.

- 6_1, 8_8, 9_{46}, 10_3: Likely all four symmetries were identified, yielding two distinct ribbon disks. The other two disks are equivalent to the first two via band isotopy.
- 8_9: Four of eight symmetries were found, resulting in four distinct ribbon disks.
- 8_{20}: Both symmetries were identified, but they produce the same ribbon disk.
- 9_{27}: Likely all four symmetries were found, yielding at least two distinct ribbon disks. The other two are not distinguished by Khovanov homology but are not evidently the same band diagram.

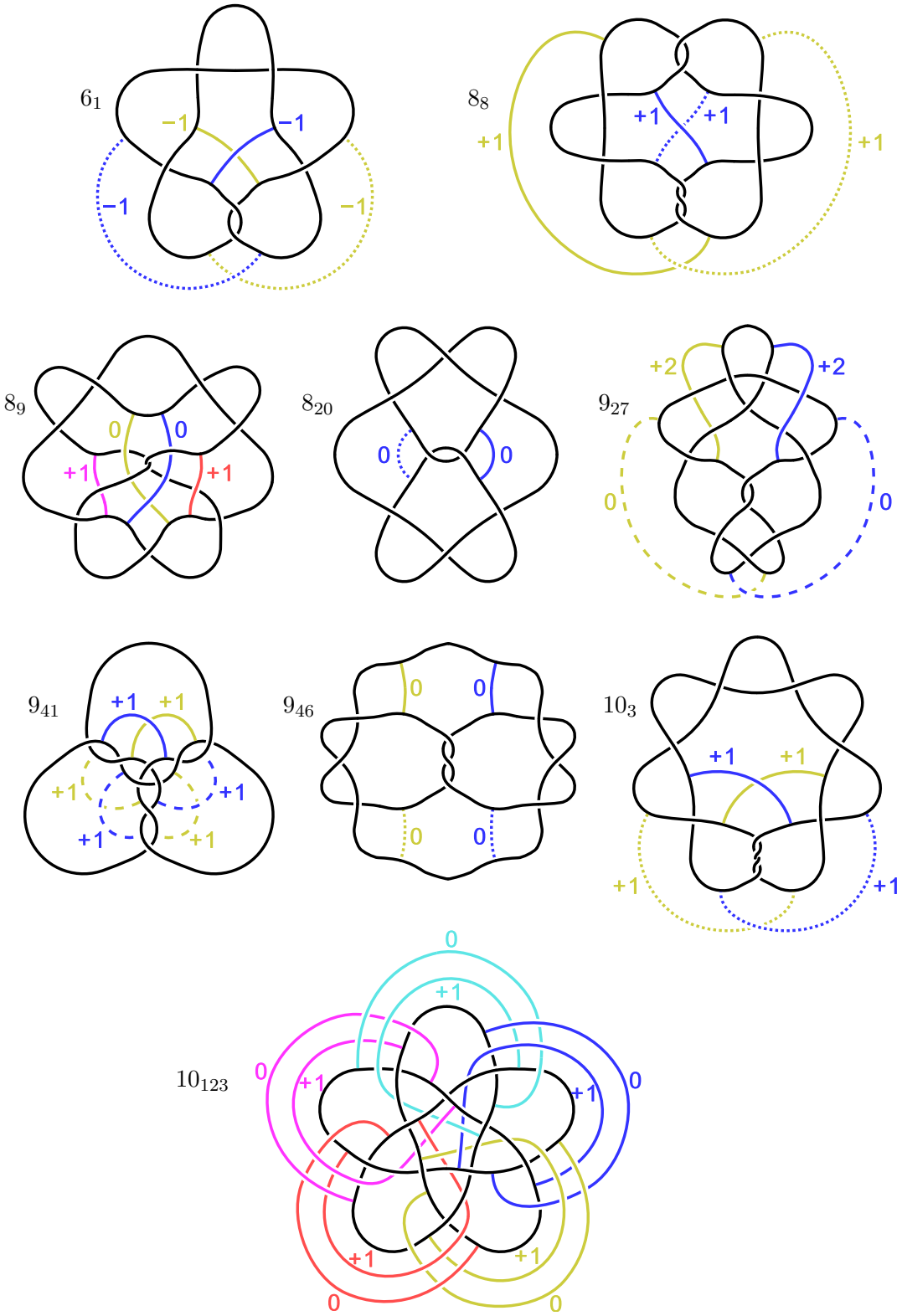


FIGURE 1. Band diagrams for ribbon disks obtained by symmetry. Solid lines of different colours represent non-isotopic disks distinguished by Khovanov homology.

- 9_{41}: Likely all six symmetries were identified, resulting in at least two distinct ribbon disks. The other four are not distinguished by Khovanov homology.
- 10_{123}: Chosen for its high number of symmetries, likely all 20 were identified, yielding at least five distinct ribbon disks. The remaining disks are not distinguished by Khovanov homology.

Some of these ribbon disks have already been observed to be non-isotopic. For instance, see [Fox61, Examples 10–11] for 6_1 and [MP19, SS22] for 9_{46}.

References

[BN05] Dror Bar-Natan, *Khovanov's homology for tangles and cobordisms*, Geom. Topol. **9** (2005), 1443–1499. MR2174270

[BN07] ———, *Fast Khovanov homology computations*, J. Knot Theory Ramifications **16** (2007), no. 3, 243–255. MR2320156

[EL11] Michael Eisermann and Christoph Lamm, *A refined Jones polynomial for symmetric unions*, Osaka J. Math. **48** (2011), no. 2, 333–370. MR2831977

[Ell09] Andrew Elliott, *Graphical methods establishing nontriviality of state cycle Khovanov homology classes* (2009), available at arXiv:0907.0396.

[Ell10] ———, *State cycles, quasipositive modification, and constructing H-thick knots in Khovanov homology*, ProQuest LLC, Ann Arbor, MI, 2010. Thesis (Ph.D.)–Rice University. MR2782376

[Feh23] Zsombor Fehér, *Non-isotopic seifert surfaces in the 4-ball* (2023), available at arXiv:2304.12113.

[Fox61] R. H. Fox, *A quick trip through knot theory*, Topology of 3-manifolds and related topics (Proc. The Univ. of Georgia Institute, 1961), 1961, pp. 120–167. MR140099

[Gab86] David Gabai, *Genera of the alternating links*, Duke Math. J. **53** (1986), no. 3, 677–681. MR860665

[Hay23] Kyle Hayden, *An atomic approach to Wall-type stabilization problems* (2023), available at arXiv:2302.10127.

[Hay25] ———, *Lecture notes on link homologies and knotted surfaces* (2025), available at arXiv:2507.15305.

[HKM+22] Kyle Hayden, Seungwon Kim, Maggie Miller, JungHwan Park, and Isaac Sundberg, *Seifert surfaces in the 4-ball* (2022), available at arXiv:2205.15283.

[HS24] Kyle Hayden and Isaac Sundberg, *Khovanov homology and exotic surfaces in the 4-ball*, J. Reine Angew. Math. **809** (2024), 217–246. MR4726569

[Jac04] Magnus Jacobsson, *An invariant of link cobordisms from Khovanov homology*, Algebr. Geom. Topol. **4** (2004), 1211–1251. MR2113903

[Kak05] Osamu Kakimizu, *Classification of the incompressible spanning surfaces for prime knots of 10 or less crossings*, Hiroshima Math. J. **35** (2005), no. 1, 47–92. MR2131376

[KhCC] Zsombor Fehér, *Khovanov Cobordism Calculator*, software, 2024. Available at https://github.com/ZsomborFeher/khovanov-cobordism.

[Kho00] Mikhail Khovanov, *A categorification of the Jones polynomial*, Duke Math. J. **101** (2000), no. 3, 359–426. MR1740682

[KInfo] Charles Livingston and Allison H. Moore, *KnotInfo: Table of Knot Invariants*, 2025. Available at https://knotinfo.org.

[KJob] Dirk Schütz, *KnotJob*, software, 2023. Available at https://www.maths.dur.ac.uk/users/dirk.schuetz/knotjob.html.

[KLO] Frank Swenton, *KLO (Knot-Like Objects) software (version 0.978 alpha)*, 2024. Available at https://www.klo-software.net.

[Lam22] Christoph Lamm, *Symmetric diagrams for all strongly invertible knots up to 10 crossings* (2022), available at arXiv:2210.13198.

[LS22] Robert Lipshitz and Sucharit Sarkar, *A mixed invariant of nonorientable surfaces in equivariant Khovanov homology*, Trans. Amer. Math. Soc. **375** (2022), no. 12, 8807–8849. MR4504654

[MP19] Allison N. Miller and Mark Powell, *Stabilization distance between surfaces*, Enseign. Math. **65** (2019), no. 3-4, 397–440. MR4113047

[Sage] W. A. Stein et al., *Sage Mathematics Software (Version 9.5)*, The Sage Development Team, 2022. Available at http://www.sagemath.org.

[Sak86] Makoto Sakuma, *On strongly invertible knots*, Algebraic and topological theories (Kinosaki, 1984), 1986, pp. 176–196. MR1102258

[Snap] Marc Culler, Nathan M. Dunfield, Matthias Goerner, and Jeffrey R. Weeks, *SnapPy, a computer program for studying the geometry and topology of* 3*-manifolds*, 2024. Available at http://snappy.computop.org.

[SS22] Isaac Sundberg and Jonah Swann, *Relative Khovanov-Jacobsson classes*, Algebr. Geom. Topol. **22** (2022), no. 8, 3983–4008. MR4562563

[Sun22] Isaac Sundberg, *Khovanov homology & uniqueness of surfaces in the 4-ball*, ProQuest LLC, Ann Arbor, MI, 2022. Thesis (Ph.D.)–Bryn Mawr College. Available at https://repository.brynmawr.edu/dissertations/225/.

Mathematical Institute, University of Oxford, Andrew Wiles Building, Radcliffe Observatory Quarter, Woodstock Road, Oxford, OX2 6GG, UK

Email address: zsombor012@gmail.com