

Efficient computation of quantum knot polynomials

Shana Yunsheng Li

Department of Mathematics, University of Illinois Urbana-Champaign · yl202@illinois.edu · [shana-y-li.github.io](https://github.com/shana-y-li)

Introduction

Since the discovery of the Jones polynomial and the establishment of the Reshetikhin–Turaev functor, a plethora of quantum knot polynomials have been defined. However, the ability of computing these polynomials has been limited, as the Jones polynomial is already #P-hard to compute; a relatively naive implementation would struggle to compute the 10-colored Jones polynomial even for the trefoil. In this work, we show that a natural algorithm of the computation is provably efficient and give its implementation in **SnapPy** , with which it takes only a few minutes to compute the 30-colored Jones polynomial for an 11-crossing knot.

Background

Reshetikhin–Turaev functor

Given R-matrix $R \in \text{Aut}(V \otimes V)$ and ribbon element $h \in \text{Aut}(V)$:

$$\begin{array}{ccccccc} \uparrow & \mapsto & \uparrow_V & \begin{array}{c} V \\ \swarrow \searrow \\ V \otimes V \end{array} & \mapsto & \begin{array}{c} V^* \otimes V \\ \uparrow_u \\ \mathbb{F} \end{array} & \begin{array}{c} V \otimes V^* \\ \swarrow \searrow \\ \mathbb{F} \end{array} \\ \downarrow & \mapsto & \uparrow_{V^*} & \begin{array}{c} V \otimes V \\ \swarrow \searrow \\ V \otimes V \end{array} & \mapsto & \begin{array}{c} \mathbb{F} \\ \uparrow_n \\ V \otimes V^* \end{array} & \begin{array}{c} \mathbb{F} \\ \swarrow \searrow \\ V^* \otimes V \end{array} \end{array}$$

where

$$u(1) = \sum_i e_i^* \otimes h(e_i), \quad u'(1) = \sum_i e_i \otimes e_i^*,$$

$$n(x \otimes f) = f(h^{-1}(x)), \quad n'(f \otimes x) = f(x), \quad \forall x \in V, f \in V^*.$$

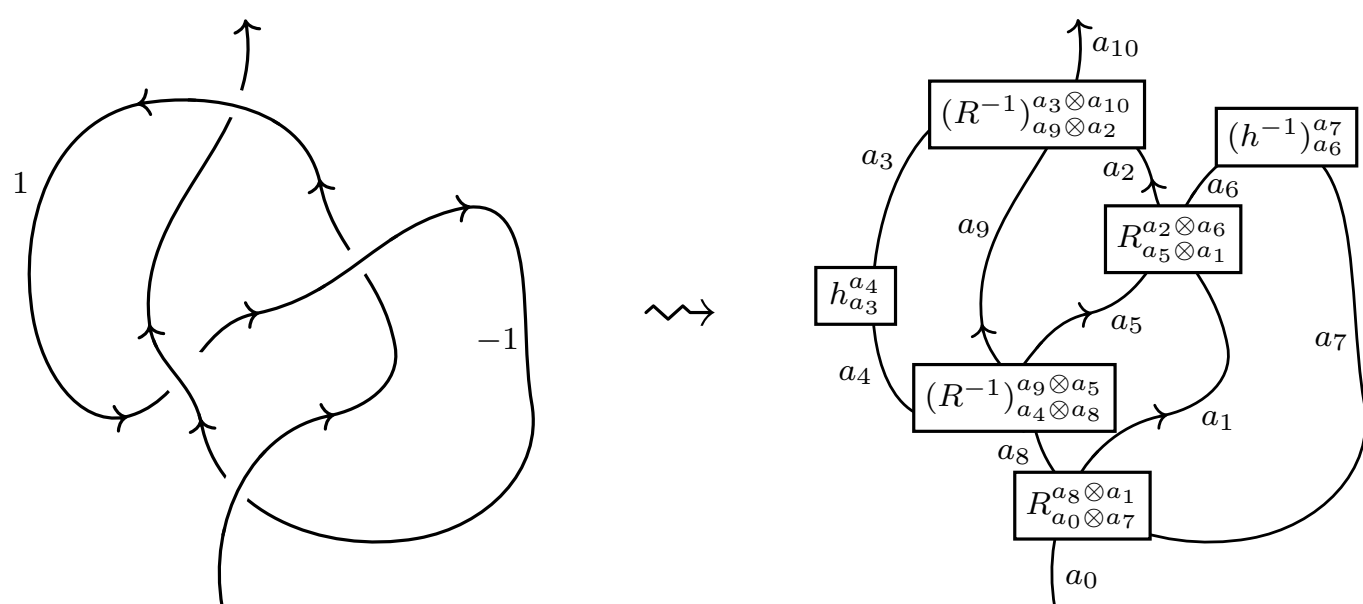
Writing R and h in matrix form with respect to a basis $\mathcal{B} = \{e_i\}$ of V :

$$\begin{array}{ccc} \begin{array}{c} a_k \\ \swarrow \searrow \\ a_i \end{array} & \mapsto & \begin{array}{c} a_k \\ \swarrow \searrow \\ a_i \end{array} \begin{array}{c} a_l \\ \swarrow \searrow \\ a_j \end{array} \\ \begin{array}{c} a_k \\ \swarrow \searrow \\ a_i \end{array} & \mapsto & \begin{array}{c} a_k \\ \swarrow \searrow \\ a_i \end{array} \begin{array}{c} a_l \\ \swarrow \searrow \\ a_j \end{array} \end{array}$$

Tensor contractions

The computation amounts to evaluating sums of products of tensor entries:

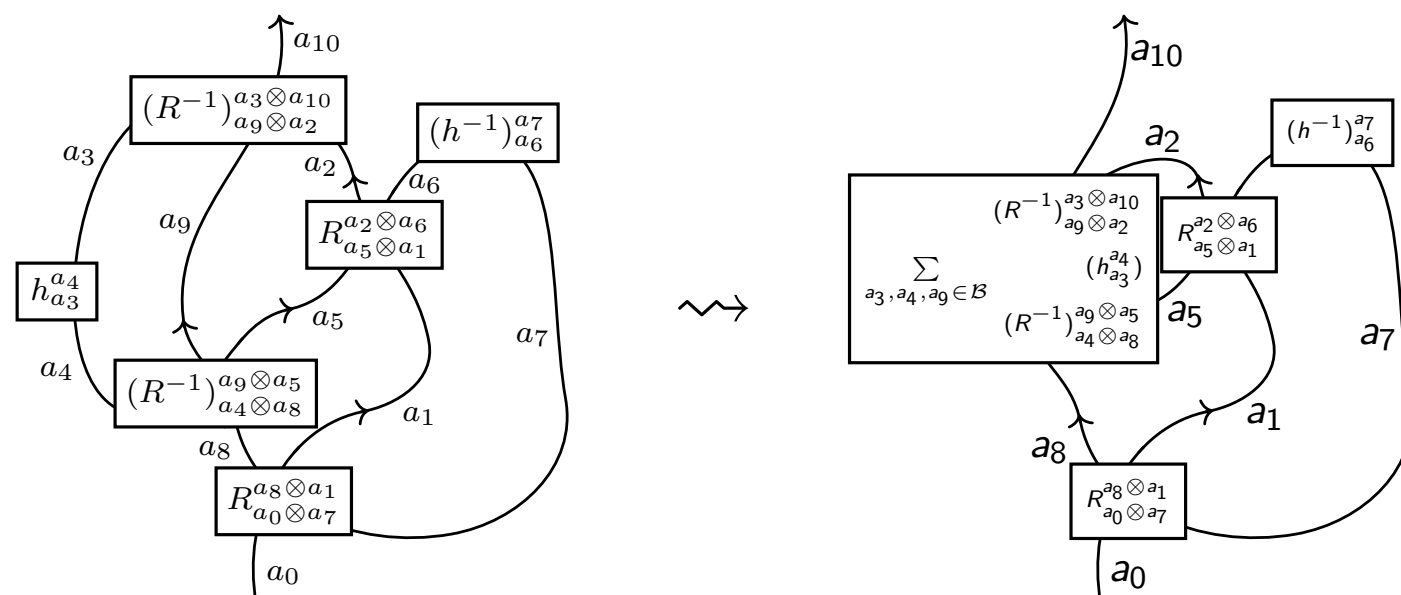
Example. The figure-eight knot



The knot polynomial is the finite sum

$$\sum_{\substack{a_1, \dots, a_9 \in \mathcal{B} \\ a_0 = a_{10} = e_1}} R_{a_9 \otimes a_7}^{a_8 \otimes a_1} \cdot h_{a_3}^{a_4} \cdot (R^{-1})_{a_4 \otimes a_8}^{a_9 \otimes a_5} \cdot R_{a_5 \otimes a_1}^{a_2 \otimes a_6} \cdot (h^{-1})_{a_6}^{a_7} \cdot (R^{-1})_{a_9 \otimes a_2}^{a_3 \otimes a_{10}}.$$

This can be achieved step-by-step; each step is a *tensor contraction*:



Method

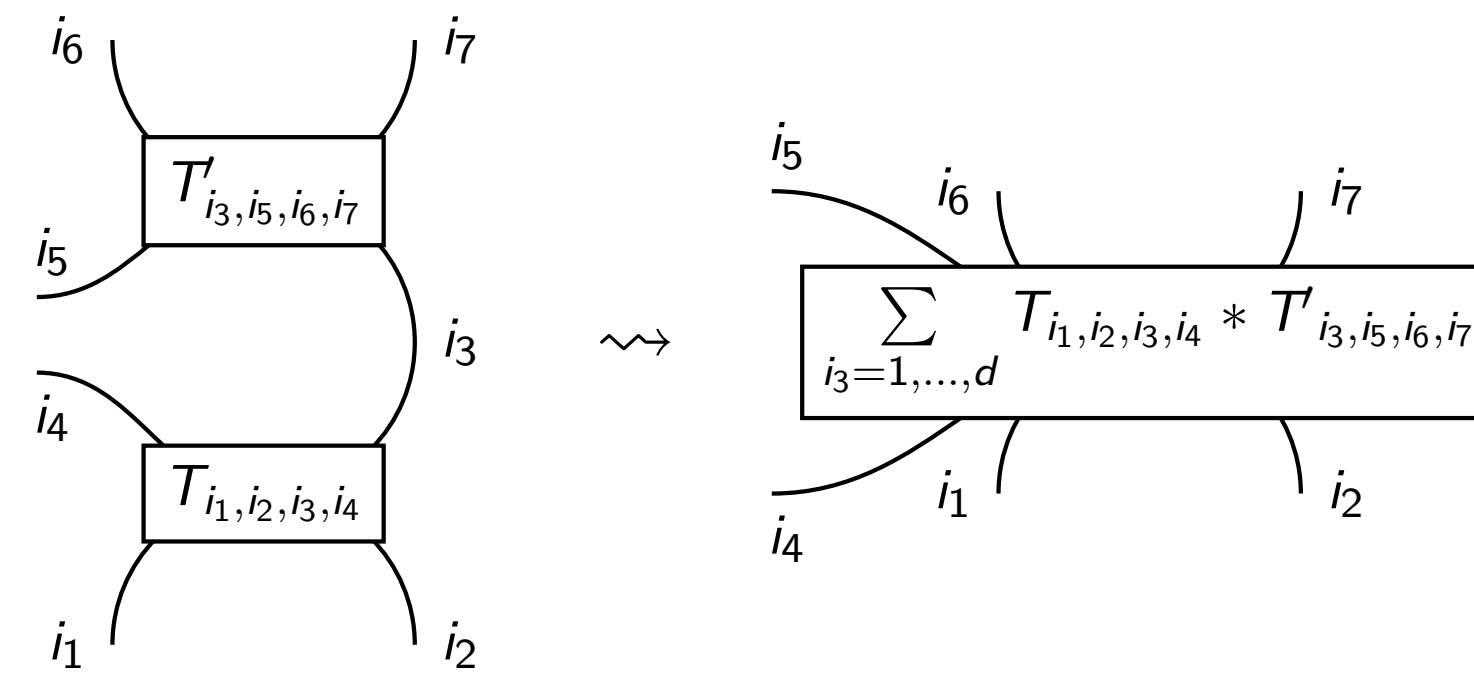
Contraction width

Assume all indices have dimension d . Computing a contraction of c tensors summing along s indices is

$$O(c \cdot d^s) \cdot d^{N-2s} = O(c \cdot d^{N-s}),$$

where N is the total number of indices of the c tensors.

Example. Contraction of two tensors



A contraction of 2 tensors along 1 index of dimension d , resulting in a tensor with 6 indices.

Let $\mathfrak{S}(\Gamma)$ denote the set of all contraction sequences on a tensor network Γ :

$$\min_{\mathfrak{S} \in \mathfrak{S}(\Gamma)} \left(\underbrace{\max_{i=1, \dots, |\mathfrak{S}|} \underbrace{(N_i - s_i)}_{\text{local contraction-width of the } i\text{-th contraction}}}_{\text{contraction-width of } \mathfrak{S}} \right)$$

Main Results

Theorem (L. 2026)

Computing a k -variable Reshetikhin–Turaev knot polynomial on an n -crossing knot diagram by contracting a contraction sequence with contraction-width w is

$$O((\dim V)^w n^{k+2} (\log n)^3),$$

Lemma

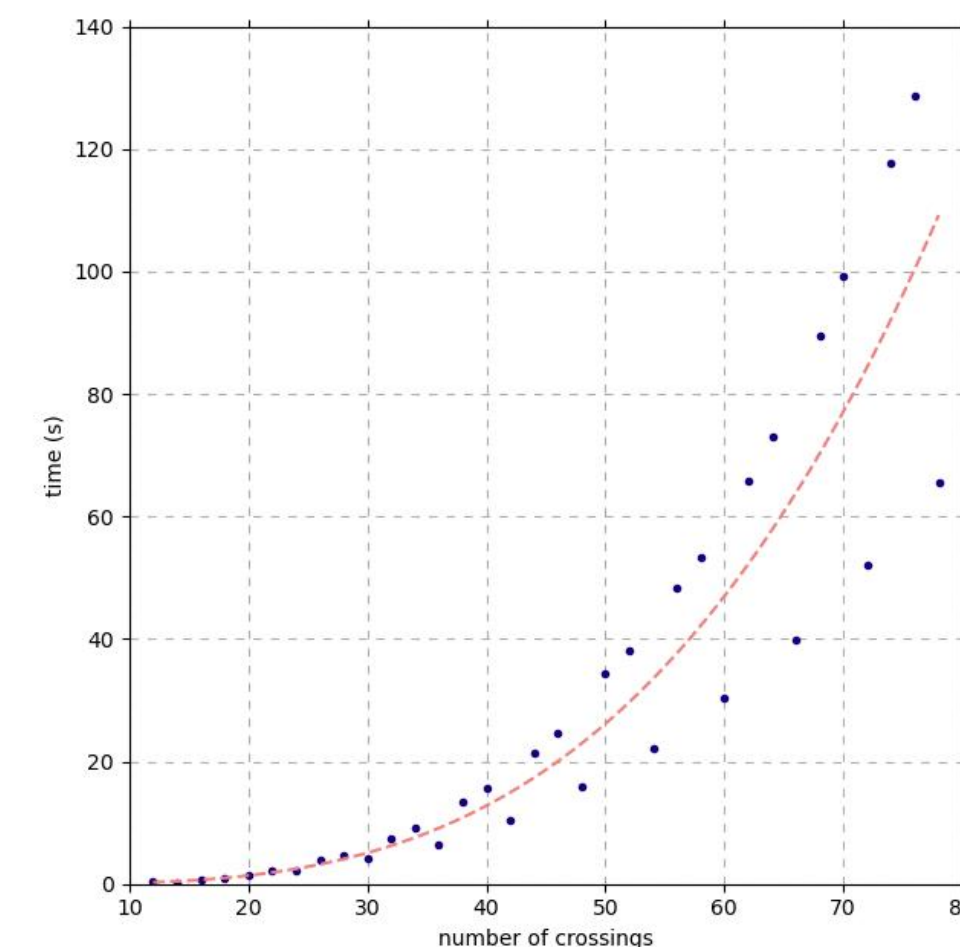
Let $w(\Gamma)$ and $tw(\Gamma)$ be the contraction-width and tree-width of Γ respectively,

$$w(\Gamma) \leq \Delta(\Gamma) \cdot (tw(\Gamma) + 1) = O(\Delta(\Gamma) \cdot \sqrt{n}),$$

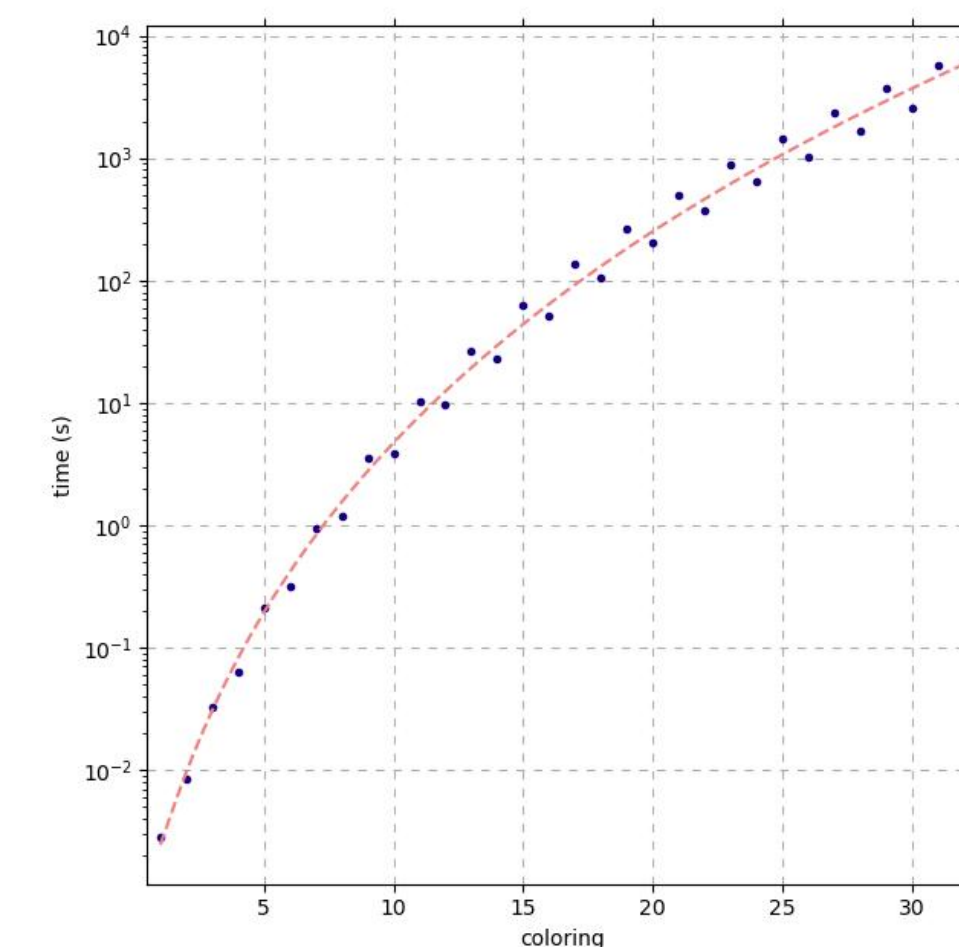
where $\Delta(\Gamma)$ is the maximum degree of Γ .

Corollary. Computing Reshetikhin–Turaev knot polynomials is $e^{O(\sqrt{n})}$.

Benchmarks



The 2-colored Links–Gould polynomial for $(3, i)$ -torus links as i ranges from 6 to 39.



The n -colored Jones polynomial for $11n34$ as n ranges from 1 to 32, plotted in log scale.

Applications

Patterns of the FK3 knot polynomial

The FK3 knot polynomial is obtained from the smallest sporadic Nichols algebra following the construction of [GK26]; its computation is about as expensive as that of the 11-colored Jones polynomial. With the implementation presented here, we were able to compute the FK3 knot polynomial for all prime knots with at most 16 crossings. With the computed data, experimentally we found:

- ▶ $\text{FK3}_K(t) = \text{FK3}_K(t^{-1}) = \text{FK3}_{\bar{K}}(t)$.
- ▶ FK3 is invariant under mutation.
- ▶ $\deg_t \text{FK3}_K(t) \leq 8 \cdot \text{genus}(K)$.
- ▶ FK3 is likely not related to other quantum knot polynomials coming from representation theory of simple Lie (super)algebras.
- ▶ FK3 is not determined by the knot Floer homology.
- ▶ FK3 is not determined by the HOMFLY-PT polynomial.
- ▶ FK3 is not determined by the Θ invariant.
- ▶ The FK3 polynomials of the following six knots with a wild range of number of crossings are nontrivial and identical:
 $8n1, 10n29, 14n18212, 15n41340, 15n45749, 16n675897$.

Question. What do these knots have in common?

Independence of knot polynomials

Efficient computation allows us to explore relationships between different quantum knot polynomials much more easily. In particular, if two knots have the same polynomial P while having different polynomial Q , then Q carries information that is not captured by P .

The colored Jones polynomial comes from representations of the quantum group $U_q(\mathfrak{sl}_2)$, while the Links–Gould polynomial LG comes from the 4-dimensional representation of the quantum supergroup $U_q(\mathfrak{sl}_{2|1})$. It is thus natural to ask whether one determines the other. By direct computation on cabled mutant knots, we were able to show that

Theorem (Garoufalidis & L. 2026)

The colored Jones polynomial does not determine the Links–Gould polynomial.

We were also able to do the same for a related family, the Kashaev–Mangazeev polynomial KM_n , for which we have $\text{KM}_2(t/q, q^2) = \text{LG}(t, q)$:

Theorem (Garoufalidis & L. 2026)

The colored Jones polynomial does not determine the KM_3 polynomial.

References

- [CDGW] Marc Culler, Nathan Dunfield, Matthias Goerner, and Jeffrey Weeks, *SnapPy, a computer program for studying the geometry and topology of 3-manifolds*, Available at <http://snappy.computop.org>.
- [GK26] Stavros Garoufalidis and Rinat Kashaev, *Multivariable Knot Polynomials from Braided Hopf Algebras with Automorphisms*, Publ. Res. Inst. Math. Sci. **62** (2026), no. 1, 75–114.
- [GL26a] Stavros Garoufalidis and Shana Yunsheng Li, *The colored Jones polynomial does not determine the Links–Gould polynomial*, 2026, arXiv:2610.07198.
- [GL26b] ———, *The FK3 knot polynomial*, 2026, arXiv:2609.01506.
- [Li26] Shana Yunsheng Li, *Fixed-parameter tractable computation of Reshetikhin–Turaev knot polynomials via tensor networks*, 2026, arXiv:2606.29662.