THE GORDIAN KNOT FOR VLMS: DIAGRAMMATIC KNOT REASONING AS A HARD BENCHMARK

Anonymous authors
Paper under double-blind review

ABSTRACT

A vision–language model can fail a diagram question by misreading the diagram or by failing to operate on what it read. KnotBench pairs a corpus of 858,318 rendered diagrams of 1,951 prime-knot prototypes with 2,000 evaluation items in 14 tasks, labeled from the knot census and stored planar diagram (PD) codes. Its equivalence and move questions come as images and as PD codes, to separate these two sides of a perception–operation gap. We evaluate Claude Opus 4.7 and GPT-5 at two thinking settings each, under unequal output caps. Of the 56 (task, configuration) pairs, 22 (18 outside C1, whose baseline is 0) are at or below the uniform random baseline, and on 6 of 14 tasks no configuration scores above 65%. All configurations count crossings worse than a constant answer and match images to PD codes at chance. Pooled over the paired questions, they score 14.3 to 33.8 points higher on PD codes than on images. Thinking adds 2.0 points for Claude and 5.4 for GPT-5, mostly on three PD-code tasks, two of which text rules solve. The data establish a failure to read structure from our drawings; since real-step move labels also follow from crossing counts, whether the models simulate moves remains open.

1 INTRODUCTION

Vision–language models miscount overlapping circles, misjudge how many times two lines intersect (Rahmanzadehgervi et al., 2024), and lose accuracy in line tracing where the path first crosses itself (Petrova et al., 2026). A failure on a diagram question can arise at two steps: reading the structure off the pixels, or operating on the structure once read, for instance to decide whether two diagrams show the same object. We call the span between the two steps the *perception–operation gap*. An image question tests both steps at once, so a wrong answer does not say which step failed.

Knot diagrams let us pose one question with and without the reading step, with exact answers in both cases. A knot has many diagrams, related by Reidemeister moves, and each diagram has a planar diagram (PD) code, a list of integer tuples that records every crossing. Posing the same question on drawings and on PD codes holds the operation fixed. Asked which move turns one diagram into another, a model can try the moves or, when the two are one move apart, count crossings, because R1 changes the count by one, R2 by two and R3 not at all. Our hypothesis is that the models we test cannot operate on a diagram's structure even when it is given to them as a PD code. If so, accuracy should be low on both inputs unless a simpler property, such as the crossing count, gives the answer. A failure to read would lower accuracy on images only, even where a crossing count suffices.

KnotBench holds 858,318 images of diagrams derived by random walks of Reidemeister moves from 1,951 census knots and their mirror images, so each diagram has a known knot and stored PD code. Its 2,000 evaluation items form 14 tasks. Equivalence judgment (A0–A3) and move prediction (B0) pose five questions on images (-I) and, on separately sampled items, on PD codes (-S). Identification (C0, C1) asks for a drawing's crossing count or Dowker–Thistlethwaite (DT) code, and cross-modal grounding (D0, D1) asks whether a PD code describes an image or which of four does.

We evaluate Claude Opus 4.7 (Anthropic, 2026) and GPT-5 (OpenAI, 2025) at two thinking settings each. Of the 56 (task, configuration) pairs, 22 are at or below the uniform random baseline (18 without C1, whose baseline is 0; Fig. 1) and 36 at or below a constant answer or a text rule that follows

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

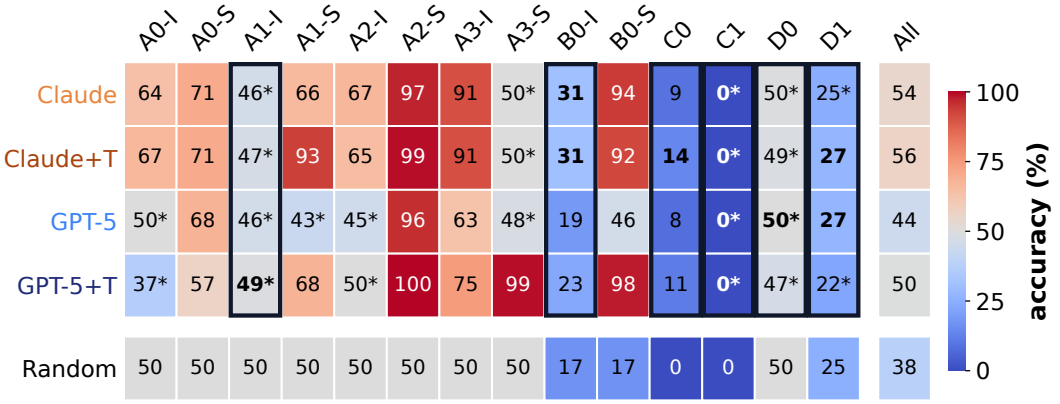


Figure 1: Accuracy (%) on the 14 tasks under the primary scoring: corrected labels, A0 scored as ambient isotopy and C1 by strict string match. Claude runs without thinking and GPT-5 at minimal reasoning effort; +T marks thinking at high effort. An asterisk marks accuracy at or below the task's uniform random baseline (Random row), compared exactly (22 of 56 pairs). Boxes mark the 6 tasks on which no configuration scores above 65%, with the best value in bold. All is item-weighted accuracy over the 2,000 items. The C0 and C1 baselines are 0 only because the answers are free-form: a constant count scores 20.0% on C0 and the most frequent C1 code 31.0%, more than every configuration, as do constant answers on A0-I, A0-S and A1-I (Table 1).

from how we built the items. Pooled over the five questions posed both ways, every configuration is 14.3 to 33.8 points more accurate on PD codes than on images, a difference we call the *PD-code advantage* (Fig. 2). All four count crossings less accurately than a constant answer and are at chance on D0 and D1. These results place the failure on the perception side of the gap, although D alone cannot separate misreading from a failed comparison with the code.

Thinking raises accuracy by 2.0 points for Claude and 5.4 for GPT-5 and widens the PD-code advantage, mainly through A1-S for Claude and A3-S and B0-S for GPT-5. Without these tasks, neither the lifts nor the widenings differ from zero. On A3-S and B0-S, rules that use only the order or the number of the tuples score 100.0 and 96.0%. Outside these two tasks and A2-S, whose answer follows from the tuple counts, the only PD-code score above 90% is Claude+T's 93.0% on A1-S, against 79.0% for the best non-visual rule we tested. This result counts against our hypothesis, although its margin over that rule rests on 10 ill-posed items (Section 5.3).

Contributions. (1) We release a 2,000-item evaluation set whose equivalence and move questions come both as images and as PD codes and whose grounding items have codes independent of the answer. (2) We evaluate four configurations of two models against random, constant and text-rule baselines. (3) We audited the items by computation after the evaluation, corrected 20 B0 and 5 A1 labels and relabeled 84 A0 items under ambient isotopy (Appendix J). We rebuilt D0 and D1 after finding that our renderer had switched every crossing of every corpus image, which leaves the A–C labels intact (Section 3). The cue rules we report show which tasks can support a claim about operating on structure (Appendix J.2). Sections 2 to 4 describe knot diagrams, the corpus and the tasks, and Sections 5 and 6 the results and what they establish.

2 Background: Knot Diagrams

A *knot* is a closed curve in space that does not intersect itself, and a *diagram* projects it onto a plane, recording at each *crossing* which strand passes over. Our images draw the lower strand with a gap, so a model must tell from the gaps which strand is on top. Our knots are *prime*, not two nontrivial knots tied in series.

By Reidemeister's theorem (Reidemeister, 1927), two diagrams show the same knot exactly when planar deformations and finitely many local *Reidemeister moves* turn one into the other. R1 adds

or removes a kink in one strand, R2 adds or removes a bigon where two strands overlap, and R3 slides one strand across a crossing of two others. Switching over and under at every crossing gives a diagram of the *mirror image*. A knot is *chiral* if no sequence of moves turns its diagrams into those of its mirror image, and *amphichiral* otherwise; the trefoil is chiral and the figure-eight knot amphichiral.

We write diagrams as *PD codes* in the convention of Regina (Burton et al., 1999–2025) and SnapPy (Culler et al., 2026). We orient the knot, number its arcs consecutively along the orientation, and write each crossing as the 4-tuple of its arc labels, read counterclockwise from the incoming under-strand. One three-crossing diagram of the left-handed trefoil in our corpus has the code $[(1, 4, 2, 5), (3, 6, 4, 1), (5, 2, 6, 3)]$. Given a PD code, a model reads every crossing, including which strand is on top, from the integers. Renumbering the arcs or reordering the tuples gives another code of the same diagram.

Regina's canonical signature (Burton et al., 1999–2025) fingerprints a diagram, not a knot, since a single R1 move changes it. We therefore take our knots, the *prototypes*, from the census of prime knots (Hoste et al., 1998; Burton, 2020), which lists each by a diagram with the fewest crossings the knot allows. That number of crossings is the knot's *crossing number*. Every diagram in our corpus descends by Reidemeister moves from the census diagram of a prototype or of its mirror image and inherits the knot of that starting diagram. The census also gives each prototype an alphabetical DT code (Dowker & Thistlethwaite, 1983), a string of letters that does not distinguish mirror images (`bca` for the trefoil). Some A0 negatives come from *Jones-collision pairs*, two prototypes whose knots differ but share the Jones polynomial (Jones, 1985). Such pairs need not be Conway mutants (Conway, 1970), and none of our 18 pairs is one, because SnapPy gives the two knots of every pair different volumes, which mutation preserves (Ruberman, 1987) (Appendix A).

3 THE KNOTBENCH CORPUS

We take 1,951 prototypes from the census of prime knots (Hoste et al., 1998; Burton, 2020) distributed with Regina (Burton et al., 1999–2025). They comprise all 801 prime knots with crossing number at most 11 and 100 to 200 at each crossing number from 12 to 19, sampled uniformly from the census files we used (Table 2), which at 18 and 19 crossings hold only alternating knots.

Walks. We call the census diagram of a prototype its *seed* and run 128 random walks from the seed and 128 from its mirror image. A walk proposes 80 to 160 Reidemeister moves and accepts a proposal that leaves at most 30 crossings with probability $\min(1, e^{-\Delta E})$ (Metropolis et al., 1953). The energy E penalizes crossings and, more heavily, the kinks and bigons that a drawing shows as small loops and narrow lenses (Appendix B). Only 38.9% of the 59,932,703 proposals change the diagram. We archive the diagram after every proposal and sample the B0 items from this archive. A topology guard, meant to catch errors in our move code, rejects a proposal whose canonical signature after simplification differs from that of the simplified seed. It cannot admit a wrong label but rejects valid moves, because the signature fingerprints a diagram and simplification is randomized; 6,604 walks of non-alternating prototypes never change their diagram (Appendix B).

Rendering. The corpus uses the last diagram of each walk, its *endpoint*; we drop 68,139 of the 499,456 walks whose endpoint repeats an earlier one of the same prototype and chirality. We lay out each endpoint with spherogram's orthogonal layout (Culler et al., 2026) and rasterize it at 512×512 pixels in a solid and a rope-twist texture that share color, rotation, scale, stroke width and gap width. After 52 images lost to a layout crash and 4,264 to a lint check for overlapping strands, the corpus holds 858,318 images of 431,291 endpoint diagrams with 3 to 30 crossings (Fig. 10). Our duplicate check misses 620 repeats among them.

Renderer convention error. After the evaluation we found that our layout step read spherogram's crossing tuples as over-then-under, whereas spherogram writes under-then-over. Every corpus image and every A–C item image therefore draws the diagram of its stored PD code with every crossing switched, the same outline with the mirror image of the knot (Appendix C). The A–C labels still hold. Switching every crossing keeps crossing counts and, applied to both images of an item, preserves whether they show the same knot, chirality or diagram and which move joins them. The C1 label

Table 1: The 14 tasks. Input is two rendered images for A and B and one for C0 and C1 (I), two PD codes (S), or one image with one or four PD codes (I+S). The -I and -S variants of a task pose the same question on separately sampled items. Questions are shortened from the prompts (Appendix E). The A1, A2 and A3 prompts also tell the model that the two diagrams show the same knot, the same knot with the same chirality, and the same knot with the same chirality and crossing count. The A1 premise counts a knot and its mirror image as the same knot (Appendix J.1). B0 is multiple choice among $R1^+$, $R1^-$, $R2^+$, $R2^-$, R3 and NOT-CONNECTED. N is the number of items. Random is the accuracy (%) of a uniform guess, 0 for the free-form answers of C0 and C1. Constant is the accuracy of always giving the most frequent label, which is printed in parentheses when no other label is as frequent. Best cue rule is the highest-scoring rule we tested that predicts the label from a property of the items without the operation the task was built to test (Section 4.1 and Appendix J.2); on D0 and D1 it is the accuracy that any rule reading only the codes has in expectation. On every -I task the best rule is an oracle rule (†), which we score with the true value of a quantity that must be read off the images, such as a crossing count. An oracle rule is therefore not a baseline. On A2 the same-crossing-count rule is the task itself, and we do not list it. Constant and rule scores use the primary scoring (Section 4.2).

Task	Question	Input	N	Random	Constant	Best cue rule	
A0-I	Same knot (topologically	I	200	50.0	70.5 (no)	same crossing count†	71.5
A0-S	equivalent)?	S	200	50.0	71.5 (no)	same crossing count	61.5
A1-I	Same chirality?	I	100	50.0	52.0 (no)	same writhe sign†	87.0
A1-S		S	100	50.0	51.0 (no)	same writhe sign	79.0
A2-I	Same number of crossings?	I	100	50.0	50.0	identical drawing†	67.0
A2-S		S	100	50.0	50.0	–	
A3-I	Same planar diagram up to	I	200	50.0	50.0	identical drawing†	100.0
A3-S	relabeling?	S	100	50.0	50.0	list order	100.0
B0-I	Which move turns D_t into	I	200	16.7	18.0	count difference†	95.0
B0-S	D_{t+1}?	S	100	16.7	20.0 (R3)	count difference	96.0
C0	How many crossings?	I	100	0.0	20.0	–	
C1	What is the alphabetical DT code of this diagram?	I	100	0.0	31.0 (`bcda`)	–	
D0	Does the PD code describe this image?	I+S	200	50.0	50.0	any text rule (expected)	50.0
D1	Which of four PD codes describes the image?	I+S	200	25.0	25.0	any text rule (expected)	25.0

does not distinguish mirror images. Against the images shown, every original D0 item had the answer no, so we withdrew D0 and D1 and rebuilt both with the corrected layout step.

4 Evaluation protocol

KnotBench poses 14 tasks in four families (Table 1). We build every item from the 261 test prototypes of a split (1,381 training, 309 validation) that keeps each Jones-collision pair together. We compare the image variant (-I) and the PD-code variant (-S) of A0–A3 and B0 only within these five task pairs, singly or pooled, and otherwise compare tasks only through their own baselines. The variants can admit different *cues*, properties of the items that predict the label without the operation the task was built to test. A cue can raise only its own variant. In the A2 pair only A2-I has one, so a higher A2-S score reflects the change of input. Each A3 variant has its own, so the A3 contrast does not, and the writhe-sign and count-difference cues of A1-S and B0-S can raise the PD-code side of those pairs. We call a rule that gives a task's most frequent label a *constant rule*, one that uses only the PD codes a *text rule*, and both *non-visual rules*.

4.1 TASKS

Family A: equivalence judgment. Each A task asks yes or no about two diagrams. **A0**: same knot, "i.e., topologically equivalent". Positives pair two diagrams of one prototype and negatives two different prototypes, of equal crossing number or, in 50 items per task, from the 18 Jones-collision pairs of the test split. Because we walked from each seed and from its mirror image, 41 A0-I and 43 A0-S positives pair a chiral knot with its mirror image. We take "topologically equivalent" to mean ambient isotopy (Appendix A) and label these items no. Answering no everywhere then scores 70.5 and 71.5%. **A1**: same chirality. We corrected 5 labels that came from a mislabeled list of amphichiral knots. An A1 label is yes exactly when the two diagrams are isotopic, as in A0. It is therefore yes on the 6 A1-I and 10 A1-S items that pair an amphichiral knot with its mirror image, which the prompt's definition of chirality would call no. We call these items *ill-posed* (Appendix J.1). Because mirroring negates the writhe, the same-writhe-sign rule scores 79.0% on A1-S. **A2**: same number of crossings, in a PD code the number of tuples; 17 A2-I positives show one layout in both textures. **A3**: same diagram up to relabeling. Every A3-I positive shows one layout in both textures, and every A3-S positive pairs a code with its tuples in ascending order with a relabeled copy listed out of order, against two ordered codes in every negative. An identical-drawing rule, given the true layouts, scores 100.0% on A3-I and a list-order rule 100.0% on A3-S, so the accuracy of neither variant can show that a model recognizes one diagram across layouts or labelings.

Family B: move prediction. **B0**: which Reidemeister move turns D_t into D_{t+1}, or NOT-CONNECTED if none does. Both diagrams come from one walk, drawn in independent layouts in B0-I. A real-step item pairs two consecutive, distinct diagrams, and a NOT-CONNECTED item two diagrams 5 to 24 proposals apart, which can still be one move apart. A search of all single moves found 10 of the 35 NOT-CONNECTED items of B0-I and 10 of the 20 of B0-S one move apart, and we relabeled them with that move. We assigned every real-step label from the change in crossing count, by which the prompt also glosses each move (Appendix E). A model that counts the crossings of both diagrams can therefore label every real-step item without simulating the move. The count-difference rule scores 96.0% on B0-S and, given the true counts, 95.0% on B0-I.

Family C: identification. **C0**: the number of crossings in one image. With 20 items for each of 8, 11, 14, 17 and 20 crossings, a constant answer scores 20.0%. **C1**: the alphabetical DT code of the drawn diagram, scored by strict string match, ignoring case, against the census code of the prototype (permissive tier in Appendix F). On 94 of the 100 items the drawing has more crossings than the label has letters, so C1 rewards the census code of the drawn knot and can credit a transcription of the drawing on at most 6 items. Answering `bcda`, the code of the figure-eight knot, scores 31.0%.

Family D: cross-modal grounding. **D0**: whether a PD code describes an image. **D1**: which of four codes does. The rebuilt items use the corrected renderer, a new design and a prompt paragraph that states the PD convention (Appendices C and E). A D0 item pairs an endpoint with a copy that has one crossing switched (66 items) or every crossing switched (68), or with another endpoint of the same prototype, chirality and crossing count (66). A D1 item offers two such endpoints and a one-switch copy of each. The image draws one member of the pair or of the options, chosen at random so that 100 D0 items are yes and each D1 letter is the key on 50 items. Every shown code is renumbered, reoriented and reordered at random, so the codes of an item are independent of its answer. Of 39 D0 and 43 D1 text, style and constant rules fixed in advance, only a cross-validated D0 classifier has an exact 95% interval that excludes the random baseline, and it lies below it (Appendix J.2).

4.2 CONFIGURATIONS AND SCORING

We queried Claude Opus 4.7 (Anthropic, 2026) and GPT-5 (OpenAI, 2025) through the vendors' APIs as `claude-opus-4-7` and `gpt-5`, with the same prompt for all four configurations (Appendix E). Our Claude requests omit the thinking parameter, and Claude+T requests adaptive thinking with effort `high`. Because the GPT-5 API has no setting that turns reasoning off, GPT-5 runs with reasoning effort `minimal` and GPT-5+T with `high`. In May 2026 we queried the A–C items with output caps, thinking included, of 2,048 (Claude), 16,000 (Claude+T) and 6,144 tokens (both GPT-5 configurations). We re-queried every empty or unparseable response of a thinking configuration with larger caps, which gave 43 of the stored Claude+T and 501 of the stored GPT-5+T

responses (Appendix K). In September 2026 we queried the D items with a cap of 64,000 tokens for both thinking configurations and the May caps otherwise. Every D response contains a parsed answer, and the API reported `claude-opus-4-7` and `gpt-5-2025-08-07` for all of them.

Each item has one response per configuration, sampled with the vendor's defaults. We take the answer from the last `ANSWER:` line or, if there is none, the last non-empty line (Appendix F), and count the 28 A–C responses without a parseable answer as wrong. We call the corrected labels, with ambient isotopy for A0 and strict matching for C1, the *primary* scoring (Table 13 gives others). We give Wilson 95% intervals (Wilson, 1927) for accuracies, Newcombe intervals for differences between the variants of a task pair, and paired bootstrap intervals and exact McNemar tests for thinking lifts. Per-task tests use Holm correction over each model's 14 tasks.

5 Results

Over all 2,000 items of the May and September runs, Claude, Claude+T, GPT-5 and GPT-5+T score 54.4, 56.4, 44.3 and 49.7% (Fig. 1 and Table 25). We do not rank the two models, whose output caps differed (Appendix M).

5.1 Accuracy against reference baselines

Under exact comparison, 22 of the 56 (task, configuration) pairs are at or below the uniform random baseline (Table 14). Apart from the zeros on C1, whose baseline is 0, these are 18 pairs on A0-I, A1-I, A1-S, A2-I, A3-S, D0 and D1. Their Wilson intervals contain the baseline except that of GPT-5+T on A0-I, which scores 36.5% [30.1, 43.4]. On 6 tasks, A1-I, B0-I, C0, C1, D0 and D1, no configuration scores above 65%.

On A0 every configuration scores at or below always answering no, with balanced accuracies from 50.5 to 57.2% (Appendix J.1), so the uniform baseline understates what non-visual rules achieve. We therefore also compare each pair with its task's constant rule or, on A3-S and B0-S, the text rule that follows from how we built the items, list order and count difference (Appendix J.2). Of the 56 pairs, 36 score at or below that rule, including all four configurations on A0-I, A0-S, A1-I, A3-S, C0, C1 and D0, and 38 if we add two further text rules, same crossing count and same writhe sign. On A2-S, whose answer follows from the tuple counts, the configurations score 96.0 to 100.0%, so they can count the tuples of a code as we present it.

5.2 Image versus PD-code input

Pooled over the five task pairs, the PD-code advantage ranges from 14.3 points [9.4, 19.1] for Claude to 33.8 [28.9, 38.4] for GPT-5+T (Newcombe intervals; accuracies in Fig. 2), or from 26.2 to 39.5 without A3. On A3, Claude, Claude+T and GPT-5 score higher on the image variant, but the two cues leave this contrast uninformative (Appendix M). Per task pair, the interval lies above zero for every configuration on A2 and B0. We interpret the A2 and B0 advantages as consistent with a failure to read structure from the image. The structure these -S variants require may, however, be no more than a count, since the A2-S answer and every real-step B0-S label follow from the tuple counts.

With thinking the PD-code advantage widens by 4.0 points for Claude [0.8, 7.3] and 16.5 for GPT-5 [10.6, 22.3] (paired bootstrap), while pooled -I accuracy changes by 0.5 and 1.6 points. The widening depends on a few task pairs: without A1, Claude's change is -0.4 points [-3.4, 2.7], and without A3 and B0, GPT-5's is 6.3 [-1.8, 14.0]. Claude+T wrote 50 output tokens or fewer on 994 of its 1,000 responses to the seven image-only A–C tasks. Its lift there of 0.9 points (exact McNemar $p = 0.368$) therefore says little about thinking on images (Appendix M).

5.3 Thinking lift

Thinking raises accuracy over the 14 tasks by 2.0 points for Claude [0.3, 3.6] and 5.4 for GPT-5 [3.0, 7.8], with the gains concentrated on -S tasks (Fig. 11). Of Claude's net gain of 39 items, 27 come from A1-S, and without A1-S the lift is 0.6 points (exact McNemar $p = 0.488$). Of GPT-5's 108, 51 come from A3-S and 52 from B0-S, the two tasks that text rules solve, and without them the lift is

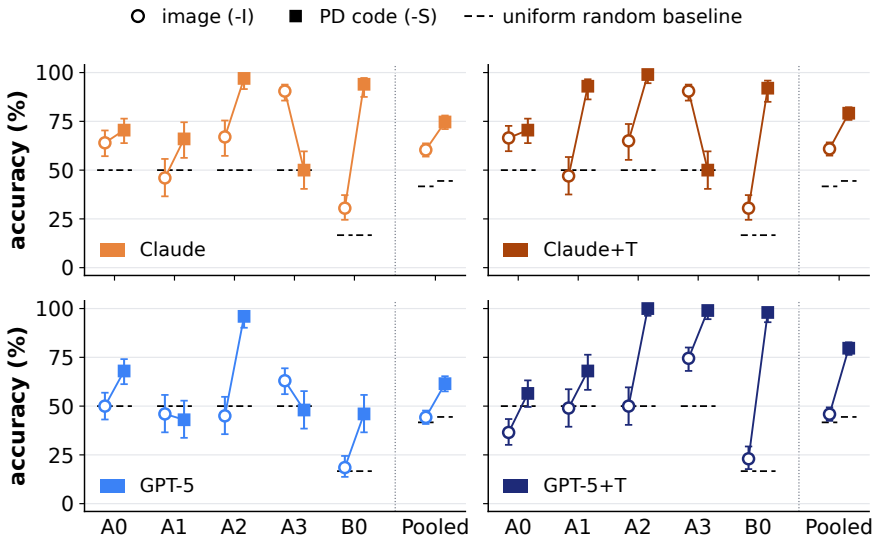


Figure 2: Accuracy on the image variant (-I, hollow circles) and the PD-code variant (-S, filled squares) of the five task pairs, one panel per configuration, under the primary scoring, with Wilson 95% intervals (200 items for A0-I, A0-S, A3-I and B0-I, 100 otherwise). Pooled: all 800 -I against all 600 -S items. Dashed: uniform random baseline (pooled: item-weighted mean).

0.3 points ($p = 0.866$). With Holm correction over each model's 14 tasks, the lift is significant for Claude only on A1-S, and for GPT-5 on A1-S, A3-S, B0-S and, as a drop, on A0-I (raw $p = 0.002$, Holm $p = 0.020$). On A0-I its rate of yes answers rises from 55.5 to 84.0%. GPT-5+T's changes on A0-S and A1-S include 12 and 5 responses without a parseable answer, which we score wrong; for 16 of them the re-query timed out (Appendix K). GPT-5+T agrees with the list-order rule on 99 of the 100 A3-S items and scores 98.0% [93.0, 99.5] on B0-S against 96.0% for the count-difference rule. GPT-5 does not return its reasoning text, so we cannot tell which route it took.

Apart from A2-S, only Claude+T on A1-S has a Wilson interval above every non-visual rule we tested (Table 19), 93.0% [86.3, 96.6] against 79.0% for the same-writhe-sign rule. With the 10 ill-posed items scored no, as the prompt's definition of chirality gives, it scores 87.0% [79.0, 92.2] against 85.0%.

5.4 Move prediction (B0)

B0-I accuracy ranges from 18.5 to 30.5%, between the 18.0% of the most frequent label and the 95.0% of the count-difference rule given the true counts, consistent with a failure to read the two diagrams. GPT-5 and GPT-5+T answer R3 on 165 and 147 of the 200 items, although only 36 are R3. On B0-S GPT-5 scores 46.0% (27.0% without the last-line fallback) and the others 92.0 to 98.0%, against 96.0% for the rule. Where the two crossing counts are equal, a counting solver must choose between R3 and NOT-CONNECTED, but 20 of the 22 such B0-S items are R3 (in B0-I 36 of 40). The best configuration on them, GPT-5+T on B0-S with 21 of 22 correct (Wilson interval 78.2 to 99.2%), cannot be told apart from always answering R3. As built, no B0 subset of useful size separates move simulation from counting and an answer prior (Appendix J.2).

5.5 Identification (C0, C1)

On C0 the configurations give the exact count on 8.0 to 14.0% of items, below the 20.0% of a constant answer, and on at most 3 of the 20 diagrams at each count of 11 or more (Fig. 5). Under strict matching no configuration writes the C1 label on any item, whereas always answering `bcda` scores 31.0%; the permissive tier credits 4 GPT-5+T answers, all `cdab` on figure-eight items. Because C1 rewards the census code, these zeros do not show a failure to transcribe the drawing (Appendix J.3).

5.6 CROSS-MODAL GROUNDING (D0, D1)

On D0 the configurations score 46.5 to 50.0% and on D1 22.0 to 27.0%, against uniform random baselines of 50.0 and 25.0% (Fig. 1), and none differs from its baseline (exact binomial test, smallest $p = 0.358$). Since the codes of a D item are independent of its answer, accuracy above the baseline could only come from the image. GPT-5 answers no on all 200 D0 items and Claude on 159. The D1 letters of every configuration depart from a uniform choice (χ^2 $p \leq 3.9 \times 10^{-4}$), GPT-5 choosing A on 2 of 200 items. No configuration scores above 57.6% on any of the three kinds of D0 item (Fig. 4). Thinking changes accuracy on the 400 D items by 0.8 points for Claude and -4.3 for GPT-5 (exact McNemar $p = 0.865$ and 0.181), although GPT-5+T wrote 6.6 million output tokens on them (Appendix K).

6 DISCUSSION AND LIMITATIONS

The failure that the data establish lies in reading structure from our drawings. The configurations count crossings less accurately than a constant answer (C0), score 45.0 to 67.0% on A2-I against 96.0 to 100.0% on A2-S, and match images to answer-independent codes at chance (D0, D1). D is the most direct of these tests, since its codes offer no route to the answer, but it does not tell misreading from a failure to match a correctly read diagram with a code up to renumbering, orientation and order. On A3-S, Claude, Claude+T and GPT-5 accept none of the 50 relabeled copies of a code and answer yes on at most 2 of the 100 items, so a failed comparison alone could keep D at chance.

Every -S score above 90% but one is on A2-S, B0-S or A3-S, where rules that use only the number or the order of the tuples score 96.0 to 100.0%. These scores therefore do not show an operation that the rules lack. The exception, Claude+T on A1-S, counts against our hypothesis unless we score the ill-posed items no (Section 5.3). Otherwise the pattern is consistent with our hypothesis (Section 1) but does not establish it. B0 cannot decide it (Section 5.4), and A0-S may be hard for any solver without a knot-identification program such as SnapPy. Since the A–C prompts do not state the PD convention, low -S scores may also reflect an unfamiliar notation.

Alternative accounts. If the configurations miss crossings at 512×512 pixels, accuracy should rise with larger renders or crops. They undercount on 54 to 85 of the 100 C0 diagrams, as this account predicts, but a failure to follow strands could also undercount. Images with every crossing marked would separate the two, since they remove the need to find crossings but not to trace strands. If the configurations misread the gaps that mark the lower strand, a worked example in the prompt should help at every image size. Answer priors predict one answer repeated across items, as with the R3 answers of GPT-5 on B0-I and its no answers on D0. A human baseline, which we lack, would show whether the drawings are legible.

Mental simulation. One mechanism that would produce the operation deficit of our hypothesis is the lack of a process that applies a Reidemeister move to an internal representation of one diagram and compares the result with the other. People do so when they mentally rotate drawn objects (Shepard & Metzler, 1971) or simulate the parts of a machine (Hegarty, 2004), though neither study involves knots. Such a process would separate R3 from NOT-CONNECTED on B0-S items with equal crossing counts, where counting cannot. A test would balance these labels, match the writhe of the two diagrams, and renumber, reorient and reorder both codes at random.

Limitations. We evaluated two proprietary models at two thinking settings each, with one response per item, so our intervals do not separate variation across items from variation across responses. They also treat items as independent, although the 200 B0-I items come from 9 prototypes and a few items repeat. The two variants of a task pair use separately sampled items, the 100 B0-S items coming from 25 prototypes, so each PD-code advantage also contains differences between the two samples. Claude+T had a larger May cap than Claude, and both thinking configurations had larger caps on re-queries and on the D items (Section 4.2 and Appendix K). Each thinking lift therefore includes a difference in output budget, and the two lifts were measured under different budgets. We queried the rebuilt D0 and D1, with their added prompt paragraph, more than four months after the other tasks and did not record which snapshots served the May requests. The 14-task scores therefore pool two runs whose snapshots may differ. We tested the A–C cue rules and corrected

labels after the evaluation, and other cues may remain. Under the original labels, 22 pairs are also at or below the uniform random baseline, and the thinking lifts over the 14 tasks are 3.2 and 7.2 points instead of 2.0 and 5.4, again mostly on -S tasks (Appendix M).

7 Related work

Failures on simple visual structure. On the classic perception tasks of BLINK the best model tested scores 51.3% against 95.7% for humans (Fu et al., 2024b), and vision–language models err on simple questions about the CLIP-blind image pairs of MMVP (Tong et al., 2024). In line tracing, which reading a knot diagram requires, models switch to nearby look-alike lines even when the crossings are removed (Hong et al., 2026). On CurveBench the best vision–language model recovers the full nesting of closed curves on 19.1% of the hard images (Mohseni et al., 2026). None of these studies also poses its questions on a symbolic description of the same structure. IsoBench poses math, science, algorithm and game problems both as images and as isomorphic text, and the models it tests score higher on the text (Fu et al., 2024a).

Knots in images and in symbols. The concurrent MindTopo asks, among other questions, whether a rendered rope is knotted or only looks tangled. All 14 models it evaluates score higher on its reasoning tasks than on its closed-loop planning tasks (Ge et al., 2026), a contrast of judging with acting rather than of drawings with codes. KnotGym asks an agent that sees only images to reshape a simulated rope until its Gauss code matches that of an exemplar (Chen & Artzi, 2025), and Nie & Yue (2026) classify photographs of tied knots. On drawn diagrams, Dranowski et al. (2025) regress the crossing number with supervised networks and report that GPT-5, prompted on 18 images, gives the correct count for 8. TACIT has an unknot-detection task but reports no model results (Medeiros, 2026). Most machine learning on knots takes symbolic input, such as hyperbolic invariants (Davies et al., 2021), braid words (Hughes, 2020; Gukov et al., 2021) or knot graphs (Jaretzki, 2023). Reinforcement-learning agents search for unknotting crossing changes (Applebaum et al., 2025) or apply Reidemeister moves to PD codes (Dranowski et al., 2026). KnotBench asks its equivalence and move questions of both kinds of input, drawings and PD codes.

Cues, baselines and thinking. Many multimodal benchmark items can be answered without the image (Chen et al., 2024), and text-only models outperform vision–language models on compositionality benchmarks with templated negative captions (Hsieh et al., 2023). In natural language inference, a classifier that reads only the hypothesis labels about 67% of SNLI correctly (Gururangan et al., 2018), and models trained on such data fail on items that defeat heuristics such as lexical overlap (McCoy et al., 2019). Our code-generated items contain such cues too (Table 1). On text benchmarks chain-of-thought prompting helps mainly on math and symbolic problems (Sprague et al., 2025), much as thinking helps our configurations mainly on PD-code tasks. On one visual task it lowered the accuracy of all six vision–language models tested (Liu et al., 2025), and long-reasoning models spend many tokens even on easy problems (Chen et al., 2025).

8 Conclusion

KnotBench asks questions about knot diagrams on rendered images and, for its equivalence and move tasks, on PD codes. Of the 56 (task, configuration) pairs, 22 (18 outside C1) are at or below the uniform random baseline. The configurations count crossings less accurately than a constant answer, a failure to read structure from our drawings, and match images to PD codes at chance. Thinking adds accuracy mainly on three PD-code tasks, two of which text rules solve. Claude+T's A1-S score counts against our hypothesis that the models also fail to operate on structure they are given. B0-S items with equal crossing counts and writhe, balanced R3 and NOT-CONNECTED labels and randomly renumbered codes would test whether they simulate Reidemeister moves.

AI use statement

We used generative AI tools to help debug the code for data generation, evaluation and analysis, to give feedback on the experimental design, and to assist with checking the items and results, literature search and writing. No label was produced by an AI model; labels come from the knot census and

the stored diagram data by the rules in Section 4. Claude Opus 4.7 is also one of the evaluated models. The authors take full responsibility for the content of this paper, including any text, code, data and claims produced with AI assistance.

Ethics statement

The dataset consists of computer-generated drawings of mathematical objects; it involves no human subjects and no personal data. The prototype diagrams come from the prime-knot tables distributed with Regina (Burton et al., 1999–2025; Burton, 2020). Model outputs were obtained through the vendors' public APIs under their terms of use. We release the evaluation items, the images shown to the models, the prompts and the model outputs under the CC BY 4.0 license (the outputs to the extent that we hold rights in them), and the scoring code under the MIT license.

Reproducibility statement

Section 3 and Appendix B describe the corpus construction, including the random seeds and the known sources of non-determinism. Section 4, Appendix E and Appendix F give the tasks, the verbatim prompts and the scoring rules; Appendix K lists the model identifiers, request parameters, output caps and re-query policy under which every stored response was produced. The anonymous supplementary material contains the 2,000 evaluation items with their images, the prompt templates, the scorer, the cue-rule scripts, the stored responses of all four configurations, a script that recomputes from these files the per-task scores, intervals, baselines, thinking lifts and output-cap counts that we report, and SHA-256 hashes of every file. It also holds the results of the D0 and D1 text-rule battery, whose recomputation needs Regina and corpus files that are not included. The full corpus of 858,318 images is not part of the supplement.

References

Anthropic. System card: Claude Opus 4.7, 2026. URL `https://www.anthropic.com/claude-opus-4-7-system-card`. Dated April 16, 2026.

Taylor Applebaum, Sam Blackwell, Alex Davies, Thomas Edlich, András Juhász, Marc Lackenby, Nenad Tomašev, and Daniel Zheng. The unknotting number, hard unknot diagrams, and reinforcement learning. *Experimental Mathematics*, 2025. doi: 10.1080/10586458.2025.2542174. Published online 18 August 2025; preprint arXiv:2409.09032.

Benjamin A. Burton. The next 350 million knots. In Sergio Cabello and Danny Z. Chen (eds.), *36th International Symposium on Computational Geometry (SoCG 2020)*, volume 164 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 25:1–25:17. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020. doi: 10.4230/LIPIcs.SoCG.2020.25.

Benjamin A. Burton, Ryan Budney, William Pettersson, et al. Regina: Software for low-dimensional topology. `http://regina-normal.github.io/`, 1999–2025. Version 7.4.1.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems*, volume 37, pp. 27056–27087, 2024. doi: 10.52202/079017-0850.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? On the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 9487–9499. PMLR, 2025.

Zizhao Chen and Yoav Artzi. Knot so simple: A minimalistic environment for spatial reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 70562–70585, 2025. doi: 10.52202/085713-2125. Datasets and Benchmarks Track.

John H. Conway. An enumeration of knots and links, and some of their algebraic properties. In John Leech (ed.), *Computational Problems in Abstract Algebra*, pp. 329–358. Pergamon Press, 1970. doi: 10.1016/B978-0-08-012975-4.50034-5.

Marc Culler, Nathan M. Dunfield, Matthias Goerner, and Jeffrey R. Weeks. SnapPy, a computer program for studying the geometry and topology of 3-manifolds. Available at `http://snappy.computop.org` (23/09/2026), 2026.

Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. Advancing mathematics by guiding human intuition with AI. *Nature*, 600(7887):70–74, 2021. doi: 10.1038/s41586-021-04086-x.

C. H. Dowker and Morwen B. Thistlethwaite. Classification of knot projections. *Topology and its Applications*, 16(1):19–31, 1983. doi: 10.1016/0166-8641(83)90004-4.

Anne Dranowski, Yura Kabkov, and Daniel Tubbenhauer. On knot detection via picture recognition. arXiv:2510.06284, 2025.

Anne Dranowski, Yura Kabkov, and Daniel Tubbenhauer. RL unknotter, hard unknots and unknotting number. arXiv:2603.07955, 2026.

Deqing Fu, Ruohao Guo, Ghazal Khalighinejad, Ollie Liu, Bhuwan Dhingra, Dani Yogatama, Robin Jia, and Willie Neiswanger. IsoBench: Benchmarking multimodal foundation models on isomorphic representations. In *First Conference on Language Modeling (COLM)*, 2024a. arXiv:2404.01266.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal large language models can see but not perceive. In *Computer Vision – ECCV 2024*, volume 15081 of *Lecture Notes in Computer Science*, pp. 148–166. Springer, 2024b. doi: 10.1007/978-3-031-73337-6_9.

Yunfei Ge, Anbang Liu, Qineng Wang, Johnalbert Garnica, Jianwen Lyu, Zihan Wang, Reuben Tan, Jianfeng Gao, Ruohan Zhang, Yining Hong, Jiajun Wu, and Manling Li. MindTopo: Can foundation models reason in topological space? arXiv:2609.11900, 2026.

Sergei Gukov, James Halverson, Fabian Ruehle, and Piotr Sułkowski. Learning to unknot. *Machine Learning: Science and Technology*, 2(2):025035, 2021. doi: 10.1088/2632-2153/abe91f.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 107–112. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-2017.

Mary Hegarty. Mechanical reasoning by mental simulation. *Trends in Cognitive Sciences*, 8(6): 280–285, 2004. doi: 10.1016/j.tics.2004.04.001.

Hyesoo Hong, Minsoo Kim, Wonje Jeung, Sangyeon Yoon, Dongjae Jeon, and Albert No. VLMs trace without tracking: Diagnosing failures in visual path following. arXiv:2605.15672, 2026.

Jim Hoste, Morwen Thistlethwaite, and Jeff Weeks. The first 1,701,936 knots. *The Mathematical Intelligencer*, 20(4):33–48, 1998. doi: 10.1007/BF03025227.

Cheng-Yu Hsieh, Jieyu Zhang, Zixian Ma, Aniruddha Kembhavi, and Ranjay Krishna. SugarCrepe: Fixing hackable benchmarks for vision-language compositionality. In *Advances in Neural Information Processing Systems*, volume 36, pp. 31096–31116, 2023. doi: 10.52202/075280-1355. Datasets and Benchmarks Track.

Mark C. Hughes. A neural network approach to predicting and computing knot invariants. *Journal of Knot Theory and Its Ramifications*, 29(3):2050005, 2020. doi: 10.1142/S0218216520500054.

Lennart Jaretzki. Geometric deep learning approach to knot theory. arXiv:2305.16808, 2023.

Vaughan F. R. Jones. A polynomial invariant for knots via von Neumann algebras. *Bulletin of the American Mathematical Society*, 12(1):103–111, 1985. doi: 10.1090/S0273-0979-1985-15304-2.

Shin'ichi Kinoshita and Hidetaka Terasaka. On unions of knots. *Osaka Mathematical Journal*, 9(2):131–153, 1957. URL https://projecteuclid.org/journals/osaka-mathematical-journal/volume-9/issue-2/On-unions-of-knots/ojm/1200689158.full.

Ryan Liu, Jiayi Geng, Addison J. Wu, Ilia Sucholutsky, Tania Lombrozo, and Thomas L. Griffiths. Mind your step (by step): Chain-of-thought can reduce performance on tasks where thinking makes humans worse. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 38489–38517. PMLR, 2025.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1334.

Daniel Nobrega Medeiros. TACIT benchmark: A programmatic visual reasoning benchmark for generative and discriminative models. arXiv:2603.00206, 2026.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114.

Amirreza Mohseni, Mona Mohammadi, Morteza Saghafian, and Naser Talebizadeh Sardari. CurveBench: A benchmark for exact topological reasoning over nested Jordan curves. arXiv:2605.14068, 2026.

Shiheng Nie and Yunguang Yue. Physical knot classification beyond accuracy: A benchmark and diagnostic study. arXiv:2603.23286, 2026.

OpenAI. GPT-5 system card, 2025. URL https://cdn.openai.com/gpt-5-system-card.pdf. Dated August 13, 2025.

Clara Petrova, Zhuo Chen, and Marin Soljačić. TraversalBench: Challenging paths to follow for vision language models. arXiv:2604.10999, 2026.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind. In *Computer Vision – ACCV 2024*, volume 15476 of *Lecture Notes in Computer Science*, pp. 293–309. Springer, 2024. doi: 10.1007/978-981-96-0917-8_17.

Kurt Reidemeister. Elementare Begründung der Knotentheorie. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 5(1):24–32, 1927. doi: 10.1007/BF02952507.

Robert Riley. Homomorphisms of knot groups on finite groups. *Mathematics of Computation*, 25 (115):603–619, 1971. doi: 10.1090/S0025-5718-1971-0295332-4.

Daniel Ruberman. Mutation and volumes of knots in S^3. *Inventiones Mathematicae*, 90(1):189–215, 1987. doi: 10.1007/BF01389038.

Roger N. Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971. doi: 10.1126/science.171.3972.701.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To CoT or not to CoT? Chain-of-thought helps mainly on math and symbolic reasoning. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=w6nlcS8Kkn.

The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.8)*, 2023. URL https://www.sagemath.org.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? Exploring the visual shortcomings of multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9568–9578, 2024. doi: 10.1109/CVPR52733.2024.00914.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

A KNOT-THEORETIC DEFINITIONS

Knots and diagrams. A *knot* K is a smooth embedding of the circle in $\mathbb{R}^3$. Two knots are *equivalent*, or the same knot, when a continuous family of homeomorphisms of $\mathbb{R}^3$, starting at the identity, carries one to the other (an ambient isotopy). A *diagram* of K is a generic projection of it to the plane that records which strand passes over at each *crossing*. The *crossing number* $c(K)$ is the smallest number of crossings in any diagram of K. By Reidemeister's theorem (Reidemeister, 1927), two diagrams show the same knot exactly when a finite sequence of R1, R2 and R3 moves and planar isotopies turns one into the other. A *flype* turns a tangle over by a half-turn, carrying an adjacent crossing to its other side, and also preserves the knot. Our walks include a move named flype, which we implement as an R3 move, so flype proposals are additional R3 proposals (Appendix B).

Mirror images. Switching every crossing of a diagram of K gives a diagram of its *mirror image* K^*. A knot is *amphichiral* if K^* is the same knot as K, as for the figure-eight knot, and *chiral* otherwise, as for the trefoil. No sequence of Reidemeister moves turns a diagram of a chiral knot into a diagram of its mirror image, so a walk never changes chirality. We read "topologically equivalent" in the A0 prompt as ambient isotopy, the equivalence defined above, and score A0 accordingly. An equivalence that also admits orientation-reversing homeomorphisms of $\mathbb{R}^3$ identifies K with K^*. The generation-time A0 labels follow that convention (Appendix J).

PD and DT codes. A *PD code* numbers the $2n$ arcs between consecutive crossings in order along an orientation and writes each crossing as the 4-tuple of its arcs, listed counterclockwise from the incoming under-strand. All our codes follow this convention, which is Regina's. Starting the numbering at another arc, numbering along the opposite orientation or listing the crossings in another order gives another PD code of the same diagram. A PD code determines the diagram on the sphere, but not which region of a planar drawing is the outer one. Shifting every label of the left-handed trefoil $[(1,4,2,5),(3,6,4,1),(5,2,6,3)]$ by one, with 6 becoming 1, gives $[(2,5,3,6),(4,1,5,2),(6,3,1,4)]$, the code we store for the trefoil prototype. A *DT code* (Dowker & Thistlethwaite, 1983) numbers the $2n$ passages through crossings along the knot and lists, for each odd number, the even number met at the same crossing, with a sign for over or under. Its alphabetical form writes $2, 4, 6, \ldots$ as `a`, `b`, `c`, ... and negative entries in upper case, so the trefoil is `bca` and the figure-eight knot `bcda`. A diagram and its reflection in a line, which is a diagram of the mirror image, have the same DT codes, so a DT code does not distinguish a knot from its mirror image.

The canonical signature. Regina's *canonical signature* is the string that `knotSig()` (Burton et al., 1999–2025) returns with its default settings. It is the same for two diagrams exactly when one is obtained from the other by some combination of relabeling the crossings, reversing the orientation, reflecting the diagram in a line and turning it over. Because reflection is among these operations, the canonical signature identifies a diagram up to mirror image. Adding one R1 kink to the trefoil above changes the string, so the canonical signature is not a knot invariant. Regina's `simplify()` removes that kink, but it is a greedy heuristic with random choices and can stop above $c(K)$. The word "signature" has two senses in knot theory; where this paper writes it without qualification, it means this string.

We therefore take the knot of every stored diagram from the seed of its walk, the census diagram of a prototype (Hoste et al., 1998; Burton, 2020) or its reflection in a line, a diagram of the mirror image. Every image except those of the rebuilt D0 and D1 items draws its stored diagram with all crossings switched and so shows the mirror image of that knot (Appendix C). Every proposal of a walk is one

of Regina's R1, R2 and R3 moves and preserves the knot. The guard of Section 3 compares the canonical signatures of the simplified proposal and the simplified seed. Since no proposal changes the knot, the guard cannot admit a wrong one and can only reject valid proposals (Appendix B). The permissive tier of C1 is the only scoring rule that compares canonical signatures (Appendix F).

Jones-collision pairs and mutation. We compute four knot invariants of every prototype with SageMath (The Sage Developers, 2023): the Jones polynomial V_K (Jones, 1985), the Alexander polynomial Δ_K, the classical signature $\sigma(K)$ and the determinant $|\Delta_K(-1)| = |V_K(-1)|$. The classical signature is an integer computed from a Seifert matrix and is unrelated to the canonical signature. Two prototypes with identical Jones polynomials form a *Jones-collision pair*. We compare the polynomials of the census diagrams and do not count the 44 pairs whose polynomials agree only after one knot is replaced by its mirror image. The corpus contains 114 Jones-collision pairs. In 39 of them the two prototypes also agree on Δ_K and σ, and hence on all four invariants. A0 uses the 18 pairs with both prototypes in the test split, 4 of which agree on all four invariants.

A *Conway mutation* (Conway, 1970) cuts a knot along a sphere that meets it in four points and glues the inside back after a half-turn that swaps the four points in pairs. Mutants share all four invariants, and a mutant of a hyperbolic knot is hyperbolic with the same volume (Ruberman, 1987). The Conway knot (Conway, 1970) and the Kinoshita–Terasaka knot (Kinoshita & Terasaka, 1957), K11n34 and K11n42 in the naming of SnapPy (Culler et al., 2026), are mutants, and Riley (1971) first proved them distinct. Our prototypes of the two, both in the training split, agree on all four invariants, and SnapPy gives each the volume 11.219118. Of the 39 pairs that agree on all four invariants, 13 have equal volumes and are mutant candidates, as are 3 of the pairs that collide only up to mirror image.

Our sampler draws each diagram of an A0 item from the walks of either chirality. An item can therefore show one knot of a pair beside the mirror image of the other, and the Jones polynomial of a mirror image K^* is $V_K(t^{-1})$. In 21 of the 50 A0-I and 24 of the 50 A0-S Jones-collision negatives, the two knots shown have different Jones polynomials for this reason. The renderer error of Appendix C switches the crossings of both images of an A0-I item and does not change which items these are. In every one of the 18 test pairs the two knots differ in volume. A knot and its mirror image have the same volume, so no Jones-collision negative of A0 shows two mutants.

B Corpus construction details

We describe the corpus pipeline stage by stage, in the order in which it runs and with the parameters of the run that built the corpus. Section 3 summarizes the same pipeline.

Prototypes. We take the prototypes from the census tables of prime knots that Regina distributes (Burton et al., 1999–2025; Burton, 2020). Hoste et al. (1998) enumerated the prime knots up to 16 crossings. The tables list each prime knot once, counting a knot and its mirror image as one entry, and give its name, its canonical signature and an alphabetical DT code. They are split into files by crossing number, by whether the knot is alternating, and by geometric type (hyperbolic, torus, satellite). For each crossing number c we pool the rows of all census files we downloaded. Up to $c = 11$ we take every row. For $c \geq 12$ we draw a uniform sample without replacement, with a single Python generator, seeded with 42 and used in increasing order of c. The files we downloaded at $c = 17$ contain only the hyperbolic knots, and those at $c = 18$ and 19 only the alternating hyperbolic knots, so every prototype at 18 and 19 crossings is alternating although the sampling is uniform over the pooled rows. Torus knots occur only up to $c = 11$ (7 prototypes), and satellite knots not at all. Table 2 gives the counts. The seed diagram of a prototype is the diagram that Regina builds from its DT code, and it has exactly c crossings for all 1,951 prototypes. The prototype record holds the census name, signature and DT code, the seed PD code, c, the geometric type and the alternating flag.

Random walks. We walk each prototype in two chiralities, from its seed diagram and from the planar reflection of the seed (Regina's `reflect()`), which is a diagram of the mirror image. Each (prototype, chirality) is the seed of 128 independent random walks. We draw the length L of a walk uniformly from $\{80, \ldots, 160\}$, and the walk makes L proposal steps. At each step we draw a move type with the weights of Table 3 and ask Regina to apply it at a random site of a copy of the current

Table 2: Prototypes per crossing number c. "Census" counts the prime knots in the census files we sampled from, a knot and its mirror image counting once; "Prototypes" counts the knots we took. Up to $c = 11$ we take every knot; for $c \geq 12$ we sample uniformly from the pooled files. †The census files we downloaded hold only the hyperbolic knots at $c = 17$ and only the alternating hyperbolic knots at $c = 18, 19$.

	Census		Prototypes	
c	all	alternating	all	alternating
3	1	1	1	1
4	1	1	1	1
5	2	2	2	2
6	3	3	3	3
7	7	7	7	7
8	21	18	21	18
9	49	41	49	41
10	165	123	165	123
11	552	367	552	367
12	2,176	1,288	200	129
13	9,988	4,878	200	106
14	46,972	19,536	150	53
15	253,293	85,263	150	44
16	1,388,705	379,799	150	30
17†	8,053,363	1,769,978	100	23
18†	8,400,285	8,400,285	100	100
19†	40,619,384	40,619,384	100	100
total			1,951	1,148

Table 3: Move types proposed by the random walk. At each step the walk draws a type with the given weight and applies it with Regina at a random site. Regina has no flype primitive, and the type labeled "flype" performs an R3 move, so R3 is proposed with total weight 0.45.

Move	Weight	Regina call and site search
R3	0.40	`r3(arc, side)`, random arc and side, up to 80 tries
$R2^+$	0.20	`r2(arc, side, arc, side)`, two random arcs, up to 80 tries
$R2^-$	0.15	`r2(crossing)`, crossings in random order
$R1^+$	0.10	`r1(arc, side, sign)`, random arc, up to 30 tries
$R1^-$	0.10	`r1(crossing)`, crossings in random order
"flype"	0.05	as R3, on diagrams with at least 4 crossings, up to 40 tries

diagram D. If the site search listed there finds no valid site, the diagram stays as it is. We reject the proposal D' if it has more than 30 crossings or fails the topology guard described below. Otherwise we accept it with probability $\min\{1, e^{-\beta \Delta E}\}$, where $\Delta E = E(D') - E(D)$, $\beta = 1$ and

$$E(D) \;=\; \lambda_{\text{size}}\,|D| \;+\; \lambda_{\text{tiny}}\,\big(N_1(D) + 0.5\,N_2(D)\big). \tag{1}$$

Here $|D|$ is the number of crossings and $N_k(D)$ the number of faces of D bounded by k edges, so that N_1 counts kinks and N_2 bigons. We set $\lambda_{\text{size}} = 0.05$ and $\lambda_{\text{tiny}} = 1$. The size term pulls walks toward smaller diagrams. The face terms discourage kinks and bigons, which appear in a drawing as small loops and narrow lenses. The acceptance rule has the form of Metropolis et al. (1953), but $R2^+$ and $R2^-$ have different weights, and $R1^+$, $R2^+$ and R3 try a bounded number of random sites while $R1^-$ and $R2^-$ try every crossing. The proposals are therefore not symmetric, and a walk is a Metropolis-filtered random walk that does not sample from $e^{-\beta E}$. The walk never simplifies the diagram it continues from; `simplify()` runs only on the copies that the topology guard compares.

Topology guard. To catch errors in our move code, the walk simplifies a copy of the proposal with Regina's `simplify()` before the Metropolis test and compares the canonical signature of the result with that of the simplified seed. If the two differ, it rejects the proposal and the walk continues. Every proposal is a Reidemeister move and preserves the knot, so the guard cannot make

a label wrong. It can still reject proposals, because the signature identifies a diagram and not a knot, and `simplify()` is a randomized heuristic that need not return the diagram it returns for the seed. The walks did not log guard rejections, so we probed the guard after the corpus was built, at the seed diagrams and at later states of archived walks, with proposals that found a valid site. At the seed diagram it rejected 17,306 of 26,610 such proposals for the 446 non-alternating prototypes whose seed admits an R3 move. It rejected none for the other prototypes, and no alternating seed admits an R3 move. Averaged over its walks, each of these prototypes accepts between 0.5% and 37.8% of its proposals, against 37.0% to 55.7% for the other 1,505 prototypes. The two ranges overlap, since 11 of them lie above the lowest rate of the others, so the acceptance rate alone does not identify them. In all, 6,604 walks of non-alternating prototypes never changed their diagram, while every walk of an alternating prototype did. Away from the seed the guard also rejects proposals of the other prototypes. At later walk states it rejected 1,582 of 34,750 probe proposals on 344 of the 1,148 alternating prototypes. On the other 357 non-alternating prototypes it rejected 775 of 10,968, on 149 of them.

Trajectory archive. For every walk we store the seed and the diagram after every proposal step, $L + 1$ PD codes in all, whether or not the step changed the diagram. Over the 499,456 walks this gives 60,432,159 stored diagrams for 59,932,703 proposal steps. Of the steps, 61.1% leave the diagram unchanged, through a rejection or for lack of a valid site, and 23,320,789 change it. The archive holds one JSONL file per prototype with its 256 walks, 128 in each chirality. We sample the B-family items from it (Section 4.1 and Appendix D).

Endpoints and de-duplication. The last diagram of a walk is its endpoint, and we render only endpoints as corpus images. Within each (prototype, chirality) we merge endpoints whose PD codes coincide once the arcs are renumbered in order of appearance. In all, 68,139 of the 499,456 walks (13.6%) end on the same diagram as an earlier walk of the same prototype and chirality, which leaves 431,317 distinct endpoints. Our renumbering key depends on the order in which the crossings are listed, so it misses some duplicates. Comparing endpoints by the canonical signature computed with reflection and turning over disallowed, we found that 620 of the 431,291 diagrams in the corpus repeat another diagram of the same prototype and chirality, in 526 groups across 372 prototypes. We left them in the corpus.

Layout. For each prototype we write the distinct endpoints to a JSONL file and start one SageMath Python process, which reads the file and writes the layouts to a second JSONL file. This layout step computes spherogram's orthogonal layout of each PD code (Culler et al., 2026). The layout consists of vertex positions on an integer grid, the strand pieces between them, and a tuple for each crossing that names the pieces passing over and under. The layout step rescales x and y separately to the interval $[0.05, 0.95]$, so every drawing is stretched to fill a square. It also inserts each crossing point into both strand pieces that meet there. The layout step read the crossing tuples in the wrong order, so every corpus image shows its diagram with all crossings switched. Appendix C gives the cause, the labels that the error leaves unchanged, and the fix. The layout failed on 26 diagrams of one prototype, whose 52 images were never produced.

Rasterization. We draw each laid-out diagram as an SVG on a white 512×512 canvas and rasterize it with cairosvg to an 8-bit RGB PNG. We place the layout inside a margin of 12% of the canvas on each side, then rotate it by θ and scale it by s about the canvas center. The angle θ is uniform on $[-180°, 180°]$, and s is uniform on $[0.62, 0.78]$, small enough for any rotation to fit. Strands are horizontal and vertical segments with rounded bends. Each strand piece gets one corner radius, uniform between 1.5 and 3 stroke widths and capped at half the shorter adjacent segment. We do not use smoothed curves, because smoothing moves the crossing points off the strands and misaligns the gaps. The stroke width is uniform on $[4.5, 6.5]$, and the gap cut into the under-strand at a crossing is uniform on $[20, 32]$. Both are measured before scaling and come to 2.8 to 5.1 and 12.4 to 25 pixels on the canvas. The color is one of seven dark colors, chosen uniformly.

We draw each distinct diagram in two textures. A *solid* image is the SVG as drawn. We draw a *rope-twist* image the same way but with corner radii of its own, and then take every pixel from a position shifted by a smooth random offset. Two fields of Gaussian noise with standard deviation 6 pixels, each blurred with a Gaussian of standard deviation 2.8 pixels, give the horizontal and vertical shifts. The blurring leaves shifts with a root-mean-square size of 0.6 pixel along each axis, and the shifted

positions are truncated to whole pixels. The pipeline chooses color, rotation, scale, stroke width and gap once per diagram, from a seed derived from the diagram, so the two images of a diagram share them. This holds for all 427,027 diagrams that have both images. Every A3-I positive shows one diagram in its two textures, so a cue rule that answers "yes" exactly when two images share these parameters answers 200 of the 200 A3-I items correctly (Appendix J).

Visual lint. Some layouts give unreadable images although the PD code is valid, for instance two strands drawn on top of each other or running side by side. We score each image with two measures on its ink mask, the pixels with gray value below 128. The *overlap ratio* is the maximum of the mask's distance transform divided by its 99th percentile. It grows where strokes stack into a thicker blob. The *parallel-close score* is the fraction of skeleton pixels that lie within 5 pixels of ink belonging to a different connected piece of the drawing. We discard an image if its overlap ratio exceeds 1.5, if its parallel-close score exceeds 0.05, or if it cannot be scored. We removed 4,264 images at this step, 4,260 of them rope-twist. Together with the 52 images that were never laid out, they make up the 4,316 planned images that are missing for reasons other than duplication.

Totals. With 128 walks, two chiralities and two textures, a prototype can have at most 512 images, or 998,912 over all prototypes. The corpus has 858,318 images of 431,291 endpoint diagrams (620 of them repeats, above). Of the 140,594 images short of the maximum, 136,278 (96.9%) are the images that duplicate endpoints would have produced, and 4,316 were removed by lint or never laid out. A prototype has between 16 and 512 images (mean 439.9, median 470). All 123 prototypes with fewer than 256 images are non-alternating.

Style coverage. Over all 858,318 images, each color takes between 14.2% and 14.3% of the images, against 14.3% for a uniform draw. No color deviates from uniform by more than 0.1 percentage point. The manifest records rotation to $0.01°$, and all 36,001 values in $[-180°, 180°]$ occur. Each of twelve $30°$ bins holds between 8.3% and 8.4% of the images. The images are 50.2% solid and 49.8% rope-twist, and 50.0% original and 50.0% mirror chirality. Each of the four combinations of chirality and texture is within 0.149 percentage point of 25%. These marginal shares show that no single style value is over-represented. They say nothing about the joint distribution. The styles of different images are also not independent, because the two images of a diagram share color, rotation, scale, stroke width and gap.

Seeds. Every random choice that the pipeline makes itself comes from one of the following seeded generators. We discuss the random choices made inside Regina and spherogram under the reproducibility limits below.

- Census sample: Python's `random.Random(42)`, used in increasing order of crossing number.
- Walk $w \in \{0, \ldots, 127\}$ of prototype p in chirality χ: `random.Random`$(b + 9973\,w)$, where b is the 8-byte BLAKE2b digest of p, χ and the string `walk`, each followed by a zero byte, read as a big-endian integer. The walk length is the generator's initial draw, and every later move type, site and acceptance draw of the walk comes from the same generator.
- Style: a NumPy generator seeded with the BLAKE2b digest of the diagram's identifier (prototype, chirality, crossing count and index of the endpoint among those with that count), the index 0 and the base seed 42. The corner radii and the rope-twist shifts use generators seeded with the digest of the image's identifier, which includes the texture, combined by exclusive or with `0x5A5A5A` and `0x7E7E7E`, respectively.

Reproducibility limits. From the seed schedule we can recompute the census sample, prototype for prototype, and the lengths of all 499,456 walks. In general the diagrams cannot be recomputed. The guard calls Regina's `simplify()`, which draws from Regina's own random generator, and the pipeline never seeds that generator, so the guard's verdict on a proposal can differ between runs. A changed verdict changes the diagram from which the walk continues, and every later step can differ. We measured the effect in two tests. In the first test we re-ran the walk code in memory, with the seeds of the schedule above, for 8 prototypes, 128 walks per chirality, and compared every regenerated diagram with the archived one. Only the trefoil prototype reproduced all 256 walks. The alternating prototypes `6a_1`, `6a_2` and `7a_6` reproduced every walk in the original chirality but only 36, 63 and 64 of 128 in the mirror chirality; `11a_1` reproduced 38 and 55 of 128 in its two

Table 4: Steps 1 to 10 of walk 0 of the trefoil prototype, original chirality. D' is the proposed diagram, N_1 and N_2 its numbers of kinks and bigons, $p = \min\{1, e^{-\Delta E}\}$ the acceptance probability and u the uniform draw, which is made only when $\Delta E > 0$. A rejected step, and a step without a valid site, leave the diagram unchanged. The "flype" proposal performs an R3 move. Row 0 is the seed diagram D_0.

Step	Move	$\lvert D'\rvert$	N_1	N_2	$E(D')$	ΔE	p	u	Outcome
0	seed	3	0	3	1.65				
1	R2$^+$	5	0	3	1.75	0.1	0.9048	0.444	accept
2	R2$^+$	7	0	3	1.85	0.1	0.9048	0.2595	accept
3	R3	7	1	2	2.35	0.5	0.6065	0.3378	accept
4	R3	7	1	1	1.85	-0.5	1.0		accept
5	R2$^+$	9	1	2	2.45	0.6	0.5488	0.9201	reject
6	"flype"	7	1	2	2.35	0.5	0.6065	0.4712	accept
7	R2$^-$	5	2	1	2.75	0.4	0.6703	0.5795	accept
8	R1$^-$	4	1	2	2.2	-0.55	1.0		accept
9	R3								no valid site
10	R1$^+$	5	2	2	3.25	1.05	0.3499	0.461	reject

chiralities, and the three non-alternating prototypes between 28 and 58 of 128 per chirality. In the second test, run inside one process, we took the 344 alternating prototypes on which the guard had rejected a probe proposal, drew new proposals at archived walk states, and simplified each proposal ten times. On 343 of these prototypes at least one proposal received different verdicts across the ten calls. Since the verdicts vary within a single process, a difference in library versions cannot be the whole explanation.

Spherogram's orthogonal layout of a PD code can also differ from run to run. In the validation of Appendix C, a single run of the layout step reproduced the stored image for 532 of the 773 images, and recovering all of them took up to 264 runs. For every one of these images some run gave a layout whose rendering is pixel-identical to the stored image, so rasterization is deterministic once the layout is fixed. We did not record library versions for the corpus run. The evaluation set uses the stored images, and the manifest records the PD code and style parameters of each. Evaluating on KnotBench therefore does not require re-running any of these stages.

A worked walk. Table 4 lists the first ten steps of walk 0 of the trefoil prototype in its original chirality, a walk of length $L = 135$. We replayed the walk with its seed from the schedule above and logged every step. The replayed diagrams match the archived trajectory at all $L + 1$ positions. The archive stores the diagram after each step but not the proposals that were rejected, so the values for rejected proposals come from the replay. The seed diagram has no kinks and 3 bigons, so $E(D_0) = 0.05 \cdot 3 + 0.5 \cdot 3 = 1.65$. Over its 135 steps the walk accepts 74 proposals. The Metropolis test rejects 37, 24 proposals find no valid site, and the guard rejects none.

C Renderer convention error

Every corpus image, and every image in the evaluation items other than the rebuilt D0 and D1 items, shows the diagram of its stored PD code with all crossings switched. We found the error when we audited the benchmark after the evaluation runs. The error leaves the labels of every task except D0 and D1 correct for the images shown; we withdrew the original D0 and D1 items and rebuilt both tasks.

Cause. Our layout step (Appendix B) takes the geometry of a diagram from the method `plink_data()` of spherogram's orthogonal layout (Culler et al., 2026). The corpus run recorded no library versions; we describe spherogram 2.4.1, the version installed in the Sage environment that ran the layout step. The method returns vertex positions, the strand pieces between them, and one tuple per crossing. Its documentation describes the tuple as (piece over, piece under). The code that builds the tuple emits (piece under, piece over, virtual flag, label), and plink, the link editor whose

Table 5: The diagram read back from the pixels of an image, compared with the diagram D of the record's PD code and with $s(D)$, the same diagram with every crossing switched. Comparisons are between oriented diagrams, by the canonical signature with reflection, turning over and reversal of the orientation disallowed; the reading ignores the layout's crossing tuples. "Stored": the corpus images. "Same layout, corrected": the layout that produced each stored image, rendered again with the corrected tuple order. "New layout, corrected": a fresh run of the corrected layout step.

Images	read	undecided crossings	drawing $= D$	drawing $= s(D)$
Stored	773	0	0	773
Same layout, corrected	773	0	773	0
New layout, corrected	773	0	773	0

input format this is, reads it in that order. Our layout step followed the documentation and took the leading entry to be the over-strand, so the renderer cut the gap into the strand that in fact passes over.

Effect. Nothing else in our layout step depends on the order of the two entries. The strand geometry and the crossing positions are correct, and only the over/under decision is reversed, at every crossing. An image therefore shows $s(D)$, where D is the diagram of the record's PD code and s switches every crossing (Regina's `changeAll()`). The diagram $s(D)$ has the same shadow as D (the same curve in the plane, without over/under information) and is a diagram of the mirror image of the knot of D, which for a chiral knot is a different knot. The error is not a reflection of the picture. A reflection would move the strands in the plane and reverse the cyclic order of the four strand ends at every crossing, whereas in the stored images every strand runs where it should. The error concerns the images only. It does not touch the PD codes stored in the records, from which the -S prompts are built, or any label computed from them.

Evidence. One trefoil record has the PD code $[(1,4,2,5),(3,6,4,1),(5,2,6,3)]$, for which Regina computes writhe -3 and SnapPy agrees (-3), so the code describes a left-handed trefoil. All three crossings of its image are positive, as in a right-handed trefoil.

We wrote a program that recovers the drawn diagram from an image without using the layout's crossing tuples. It takes the course of the strands from the layout geometry and decides over and under at each crossing twice. The SVG reading asks which strand is continuous through the crossing point in the SVG that was rasterized. The pixel reading checks in the PNG which strand is interrupted, that is, shows background pixels on both sides of the other strand. From either reading, the program rebuilds a PD code in the viewer's frame, with x to the right, y upward and tuples read counterclockwise as seen. We compare rebuilt and stored codes by the canonical signature of Appendix A, computed with neither reflection nor turning over allowed (turning a drawing over reflects the picture and switches every crossing). A comparison of oriented diagrams also disallows reversing the orientation. Our sample holds all 391 distinct images of the original D0 and D1 items and 382 corpus images, stratified by crossing count, chirality, alternation and texture. In all it has 773 images of 576 diagrams, with 3 to 30 crossings per image and 11,378 crossings in all. For each image we first recovered the layout that produced it, by repeating the original layout step until the rendering was pixel-identical to the stored image. This succeeded for all 773 images (Appendix B).

Table 5 gives the result. The SVG and pixel readings agreed at every crossing of every image, and the pixel reading left no crossing of a stored image undecided. Every stored image shows $s(D)$ as an oriented diagram, and none shows D. When reversal of the orientation is allowed, 17 stored drawings also match D, because for these diagrams D and $s(D)$ differ only in orientation. The writhe read from the image is the negative of the record's writhe on all 690 images with non-zero writhe. Even with the orientation free, the stored drawing equals the planar reflection of D on only 69 of the 773 images. To isolate the code change, we ran the original and the corrected layout step on the same spherogram layout of each of the 576 diagrams. The two return identical strand geometry and crossing positions, and the two entries of every crossing tuple are exchanged. We also rebuilt, from each of the 400 layouts behind the B0-I images, the diagram that the renderer drew, and found $s(D)$ in all 400 cases.

Fix. Exchanging the two entries of the tuple is the only change we made to the layout step. When we render each recovered layout again with the corrected order, the drawing equals D in all 773 cases. Each such image differs from the stored one at every crossing, and every pixel difference lies at a crossing or at a bend of a strand. The bends differ because each piece of strand between two gaps draws its own corner radius, and moving the gaps changes the pieces. A fresh run of the corrected layout step gives D in 773 of the 773 cases by the SVG reading and in 773 by the pixel reading. Of the 773 corrected images of each kind, 772 and 772 pass the visual lint of Appendix B. Reflecting the stored images top to bottom would restore the knot type but not the diagram. The reflection of $s(D)$ is D turned over, a different diagram of the same knot, which in general has a different PD code in the viewer's frame.

Labels that the error leaves unchanged. The same map s acts on every image. It keeps the shadow and is one-to-one on diagrams. It sends a diagram of a knot K to a diagram of the mirror image K^*, and two knots K_1 and K_2 are the same exactly when K_1^* and K_2^* are. A label is therefore unchanged whenever it depends only on the shadow, on a relation between two images that s preserves, or on the knot up to mirror image.

- **A0 and A1.** The two images of an item show $s(D_1)$ and $s(D_2)$. Their knots are equal, or mirror images of each other, exactly when those of D_1 and D_2 are. This holds for the primary A0 labels, under which a chiral knot and its mirror image are different knots, and for A0 as designed, which identifies them.
- **A2 and C0.** The map s keeps every crossing, so crossing counts do not change.
- **A3.** Because s is one-to-one on diagrams, two images show the same diagram exactly when the two records do.
- **B0.** If one Reidemeister move turns D_t into D_{t+1}, the same move with every crossing switched turns $s(D_t)$ into $s(D_{t+1})$. That move is of the same type and changes the crossing count by the same amount. A pair that no single move connects stays unconnected, because s is its own inverse.
- **C1.** The label is the census DT code of the prototype. The census lists a knot and its mirror image as one entry, and an alphabetical DT code fixes a diagram only up to reflection, so the drawn knot K^* has the same label. The permissive tier compares canonical signatures with reflection allowed.
- **The -S variants** contain no image.

D0 and D1 ask whether a PD code describes the drawing, a relation between the record's diagram and its image that s does not preserve. Judged against its drawing, with the orientation free and reflection and turning over disallowed, every original D0 item has the answer "no". In 195 of the 200 original D1 items, no option describes the drawing. We therefore withdrew the original D items and report no model result from them.

Rebuilding D0 and D1. We drew the new D images with the corrected layout step; the rest of the renderer is the one used for the corpus. We also changed the D prompts. The original prompts stated no PD convention, so for a code of the reflected drawing the question had no defined answer. We added one paragraph to each D prompt that states the convention, with the tuples read counter-clockwise as seen in the image, and kept every other sentence (Appendix E). A code describes an image if it equals the code of the drawn diagram after renumbering the arcs, choosing the orientation and reordering the crossings. A code of the reflected drawing, of the drawing turned over, or of the drawing with any crossing changed does not describe it unless it is also a code of the drawn diagram.

We built the new items from stored walk endpoints of the test prototypes (Section 4). A D0 item starts from a pair of diagrams with the same crossing count: an endpoint P and P with one crossing switched (66 items), P and $s(P)$ (68), or P and another endpoint of the same prototype, chirality and crossing count (66). The prompt gives the code of one member of the pair. The image draws that member on the 100 yes items and the other member on the 100 no items. The code of a no item does not describe the drawing even if the picture is reflected, so no label depends on the frame in which the tuples are read. A D1 item offers two endpoints P and Q of one prototype, chirality and crossing count and the copies P' and Q' with one crossing switched, four diagrams that stay distinct when reflection and turning over are allowed. Across items the image draws P, P', Q and Q' equally often, and each letter is the key on 50 items. An image that draws P with one or all crossings switched (D0), or P' or Q' (D1), shows a diagram that is not in the corpus, as on half of

the D1 items. We draw the image from the pair or from the four options, instead of always drawing P, because with P always drawn the knot type of a shown code predicts the answer. P and Q are diagrams of one knot, while a single switch usually changes the knot. Every shown code is renumbered along its orientation, reversed in orientation with probability one half, shifted cyclically in its arc numbers and listed in random order. We read each of the 400 new images back from its pixels and from its SVG as above. On every item the oriented drawing equals the diagram the item records as drawn, the D0 code describes the drawing exactly on the yes items, and the D1 key is the only option that describes it. Appendix J.2 gives the scores of the text-only rules on the new items. Appendix K gives the dates and settings of the queries on the rebuilt items.

Metadata for the drawn diagram. We did not redraw the corpus, so its images keep the error. In the released evaluation set, the record of every image of an A, B or C item holds the PD code of the diagram it shows, next to the stored PD code to which the label refers. The drawn code uses the same arc labels. It replaces each crossing tuple (a, b, c, d) by (d, a, b, c) when the over-strand enters at d and by (b, c, d, a) when it enters at b. Which case holds follows from the orientation that runs along the under-strands from a to c. The transformation needs no library. It agrees with Regina's `changeAll()` on all 773 validation images and with the diagram read from the pixels of all 773 stored images. Work that pairs a corpus image with a PD code should apply this transformation first. The rebuilt D images show the diagrams of their records' codes and need no such mapping.

D REIDEMEISTER MOVES AND B0 LABELS

B0 shows two diagrams D_t and D_{t+1} and asks which Reidemeister move (Section 2) turns the first into the second. R1 adds or removes a small kink, which the B0 prompt calls a self-loop, and changes the number of crossings by one. R2 adds or removes a bigon and changes it by two, and R3 leaves it unchanged. We give R1 and R2 a direction, $+$ for the move that adds crossings and $-$ for the move that removes them. R3 needs no direction, because the inverse of an R3 move is again an R3 move. The five move labels R1$^+$, R1$^-$, R2$^+$, R2$^-$ and R3, together with NOT-CONNECTED for a pair that no single move relates, are the six answer options of the B0 prompt (Appendix E).

Real-step items. We build B0 from the trajectory archive of the test-split prototypes (Appendix B). The archive stores the diagram after every proposal step of a walk, whether or not the proposal was accepted, so two consecutive stored diagrams are either identical or one accepted move apart. A *real-step* item pairs two consecutive stored diagrams that differ, with D_t the earlier one. We visit the test prototypes in a seeded random order and collect every such pair in which both diagrams have at most 20 crossings and D_t has at least 8. We draw B0-I and B0-S from two such scans with different seeds, so the two variants share no item. We assign the label from the change in crossing count ($+1$ for R1$^+$, -1 for R1$^-$, $+2$ for R2$^+$, -2 for R2$^-$, 0 for R3). We file each pair in one of four strata by the crossing count of D_t, 8–10, 11–13, 14–16 or 17–20, and stop the scan as soon as every stratum holds more pairs of every move class than a quarter of that class's items. We then draw 33 items per move class for B0-I and 16 for B0-S, about a quarter from each stratum. Because the scan stops early, the 200 B0-I items come from 9 prototypes and the 100 B0-S items from 25. After evaluation we searched every B0 item, with the procedure of Appendix J.1, for a single R1, R2 or R3 move, in either direction, that turns one of its diagrams into the other. The search confirms the label of every real-step item in both variants.

NOT-CONNECTED items. NOT-CONNECTED items come from the walks of the prototypes that supplied pairs to the same scan. For each walk with at least 10 stored diagrams, we draw three pairs of stored diagrams that lie 5 to 24 proposal steps apart, apply the same crossing limits, drop identical pairs and label the rest NOT-CONNECTED without a further check. We fill the remaining 35 slots of B0-I and 20 of B0-S from these pairs, again stratified by the crossing count of D_t. Such a pair can still be one move apart, because most proposal steps leave the diagram unchanged (61.1%) and a walk can return to an earlier diagram. The single-move search finds 20 of the 55 NOT-CONNECTED items one move apart, 10 of 35 in B0-I and 10 of 20 in B0-S. For the primary scores we label these items with the move the search found. Table 6 gives their moves and the class counts before and after the correction, and Appendix J lists the items. Our sampler visits the prototypes for NOT-CONNECTED pairs in an order that depends on Python's string hashing, which is randomized

per process unless fixed, so running it again can draw a different set of NOT-CONNECTED items. The released item file fixes the set we evaluated.

Table 6: B0 labels by class. *Original*: labels assigned by the sampler. *Relabeled*: NOT-CONNECTED items that a single move of the given class relates, moved to that class. *Primary*: labels used for the primary scores. A dash means none. In the NOT-CONNECTED row, Relabeled gives the number of items moved out of the class. Relabeling leaves the totals unchanged, and the Total row has no Relabeled entry.

	B0-I			B0-S		
Label	Original	Relabeled	Primary	Original	Relabeled	Primary
$R1^+$	33	2	35	16	–	16
$R1^-$	33	1	34	16	1	17
$R2^+$	33	3	36	16	2	18
$R2^-$	33	1	34	16	3	19
R3	33	3	36	16	4	20
NOT-CONNECTED	35	−10	25	20	−10	10
Total	200		200	100		100

Crossing counts and labels. Every move changes the crossing count by a fixed amount, and the B0 prompt states that amount in its gloss of each move option (Appendix E). We assigned the real-step labels from this change, and each relabeled item is one move apart, so the two crossing counts decide the label of every item whose label is a move. We call the rule that labels a pair from its two crossing counts as the sampler did, and answers NOT-CONNECTED for any other change, the *crossing-count rule*. Applied to the PD codes, where the crossing count is the number of 4-tuples, it scores 96 of 100 on B0-S. Given the true crossing counts of both images, it would score 190 of 200 on B0-I. Under the original labels the two scores are 86 and 180. The rule errs only on NOT-CONNECTED items whose crossing count changes by at most two, which are 4 of the 10 in B0-S and 10 of the 25 in B0-I. Counting cannot separate R3 from NOT-CONNECTED when the two counts are equal. This is the case for 22 B0-S items (20 R3, 2 NOT-CONNECTED) and 40 B0-I items (36 R3, 4 NOT-CONNECTED), which Section 5.4 reports separately. Answering R3 on every such item scores 20 of 22 and 36 of 40, so within this subset only the 2 and 4 NOT-CONNECTED items separate a check for a move from a default answer of R3.

Images. For B0-I we lay out D_t and D_{t+1} separately with the orthogonal layout of Section 3 and draw both with the same style parameters and the solid texture. The two pictures of an item therefore share a style but not a layout, and the move does not appear as a local difference between them. B0-S shows the two PD codes as they are stored in the archive. Like the corpus images, the B0 images show every crossing switched relative to the stored PD code (Appendix C). Switching every crossing of both diagrams turns a move of each class into a move of the same class, so no B0 label changes.

E Prompts

We print below the system message and the user-message template of every task as sent. Each request holds one system message, passed as the `system` parameter in the Claude requests and as a message with role `system` in the GPT-5 requests, and one user message made of text blocks and image blocks in a fixed order. Our prompt builder takes no input from the configuration, so all four configurations received the same prompts, and the re-queries of empty or unparseable first responses used the same builder. The stored input token counts agree with this on the A–C items. They are identical for every item across the two configurations of each model, and every re-query that returned a response has the same count as its first attempt. The September harness stores a hash of the prompt with every D response, and the hash is the same for all four configurations on every D item; the input-token counts of the two configurations of each model agree there as well. Every image is a 512×512 PNG file, sent as base64 data of type `image/png` without a file name or identifier. In the GPT-5 requests we left the image `detail` parameter at its default. No prompt

names the knot or gives its prototype identifier. The two diagrams of a pair are called DIAGRAM A and DIAGRAM B, or DIAGRAM_T and DIAGRAM_T1 in B0, and the tasks with one image refer to it only as a knot diagram or the image. For each A and B task the image and PD-code variants share the first text block and differ only in the blocks that follow it. For C0 and C1 we send one text block followed by the image.

In the boxes, typewriter text is sent exactly as printed, including repeated spaces. A new line in a box is a line break in the prompt, and an empty line in a box is an empty line in the prompt. A prompt line too long for the box continues on indented lines. The dash in the system message is a single em-dash character. Italic text in angle brackets marks an insertion. An image block is printed as ⟨*image*⟩ on a line of its own, and the line break before it is not part of the prompt. An item's PD code is inserted on one line as a bracketed list of 4-tuples. The diagram of the left-handed trefoil in Section 2, for example, is inserted as

```
[[1, 4, 2, 5], [3, 6, 4, 1], [5, 2, 6, 3]]
```

Every text block after the first begins with a line break. The first text block of every template ends with the format of the final answer line, and the scorer takes the last such line of the response (Appendix F).

System message, all tasks

```
You are a topology expert evaluating knot diagrams. Read each question
   carefully. Always finish your answer with a line in the EXACT format
   requested — do not add explanation after the answer line.
```

A0, first text block

```
You are given two knot diagrams, labeled DIAGRAM A and DIAGRAM B.

Are DIAGRAM A and DIAGRAM B drawings of the SAME knot (i.e.,
   topologically equivalent)?

Answer ONLY 'yes' or 'no' on the LAST line, exactly. Format your final
   line as:
ANSWER: yes  or  ANSWER: no
```

A1, first text block

```
You are given two knot diagrams, labeled DIAGRAM A and DIAGRAM B. Both
   diagrams are guaranteed to be drawings of the same knot.

DIAGRAM A and DIAGRAM B are drawings of the same knot. Are they drawn
   with the SAME chirality (handedness)? Two diagrams have the same
   chirality iff one is NOT the mirror image of the other.

Answer ONLY 'yes' or 'no' on the LAST line, exactly. Format your final
   line as:
ANSWER: yes  or  ANSWER: no
```

A2, first text block

```
You are given two knot diagrams, labeled DIAGRAM A and DIAGRAM B. Both
   diagrams represent the same knot, drawn with the same chirality.

DIAGRAM A and DIAGRAM B depict the same knot drawn with the same
   chirality. Do they have the SAME number of crossings?

Answer ONLY 'yes' or 'no' on the LAST line, exactly. Format your final
   line as:
ANSWER: yes  or  ANSWER: no
```

A3, first text block

```
You are given two knot diagrams, labeled DIAGRAM A and DIAGRAM B. Both
   diagrams represent the same knot, with the same chirality, with the
   same crossing count.

DIAGRAM A and DIAGRAM B are drawings of the same knot, drawn with the
   same chirality, with the same number of crossings. Do they represent
   the SAME canonical planar diagram (i.e., are they identical as
   diagrams up to relabeling of strands)?

Answer ONLY 'yes' or 'no' on the LAST line, exactly. Format your final
   line as:
ANSWER: yes  or  ANSWER: no
```

A0-I to A3-I, remaining blocks

```
DIAGRAM A:
⟨image of diagram A⟩
DIAGRAM B:
⟨image of diagram B⟩
```

A0-S to A3-S, remaining blocks

```
DIAGRAM A (PD code):
⟨PD code of diagram A⟩
DIAGRAM B (PD code):
⟨PD code of diagram B⟩
```

B0, first text block

```
You are given two knot diagrams, DIAGRAM_T and DIAGRAM_T1, that are
   claimed to be consecutive steps of a Reidemeister-move sequence on
   the same underlying knot.

Which Reidemeister move (if any) transforms DIAGRAM_T into DIAGRAM_T1?

Choices:
  R1+    add one self-loop (one new crossing)
  R1-    remove one self-loop (delete one crossing)
  R2+    add one bigon (two new crossings)
  R2-    remove a bigon (delete two crossings)
  R3    triangle move (3 strands swap, no crossing-count change)
  NOT-CONNECTED   the two diagrams are NOT connected by a single R-move

Answer ONLY one of {R1+, R1-, R2+, R2-, R3, NOT-CONNECTED} on the LAST
   line, exactly as:
ANSWER: <choice>
```

B0-I, remaining blocks

```
DIAGRAM_T:
⟨image of DIAGRAM_T⟩
DIAGRAM_T1:
⟨image of DIAGRAM_T1⟩
```

B0-S, remaining blocks

```
DIAGRAM_T (PD code):
⟨PD code of DIAGRAM_T⟩
DIAGRAM_T1 (PD code):
⟨PD code of DIAGRAM_T1⟩
```

C0, all blocks

```
Count the number of crossings in this knot diagram.

Answer with a single integer on the LAST line:
ANSWER: <integer>
⟨image⟩
```

C1, all blocks

```
This is a knot diagram. Provide the alphabetical Dowker-Thistlethwaite
    (DT) code for it.

Format: a string of lowercase letters of length n (where n is the number
    of crossings). Use the standard alphabetical DT convention.

Answer with the DT string ONLY on the LAST line:
ANSWER: <dt-string>
⟨image⟩
```

Prompts of the rebuilt D items. We queried the rebuilt D0 and D1 items (Appendix C) in the September run with the prompts below. We printed them from the prompt code whose hash the run metadata record, and every stored D response carries the version tag of that code. The prompts of the original D items did not say how a PD code is read against a picture, so a code that describes the planar reflection of the drawing had no defined answer. After the first sentence the new prompts insert one paragraph that states the PD convention, followed by an empty line, and they keep every other sentence and block in the same order. Removing that paragraph and its empty line gives the original prompt. The paragraph is the same in D0 and D1 and is printed once.

D0 and D1, PD-convention paragraph

```
PD code convention (as in KnotTheory and Regina): orient the knot and
    number its arcs 1, 2, 3, ... consecutively along that orientation,
    where an arc is the piece of the curve between two consecutive
    crossing points (passing over or under). Each crossing is written as
    a 4-tuple [a, b, c, d] of the labels of the four arcs that meet at
    it, listed counterclockwise as seen in the image, starting from the
    incoming under-strand (so the under-strand runs from arc a to arc c).
    A PD code describes a diagram only up to renumbering of the arcs and
    reordering of the crossings.
```

D0, all blocks

```
You are given an image of a knot diagram and a PD code.

⟨PD-convention paragraph⟩

Question: Does the PD code describe THIS image?

Answer ONLY 'yes' or 'no' on the LAST line, exactly:
ANSWER: yes  or  ANSWER: no
Image:
⟨image⟩
PD code to verify:
⟨PD code⟩
```

D1, all blocks

```
You are given an image of a knot diagram and 4 candidate PD codes
   labeled A, B, C, D.

⟨PD-convention paragraph⟩

Question: Which PD code correctly describes the diagram in the image?

Option A:
⟨PD code of option A⟩

Option B:
⟨PD code of option B⟩

Option C:
⟨PD code of option C⟩

Option D:
⟨PD code of option D⟩

Answer ONLY one letter A, B, C, or D on the LAST line, exactly:
ANSWER: <letter>
⟨image⟩
```

Prompt wording and labels. The A0 prompt asks whether the two diagrams show the same knot, "i.e., topologically equivalent". We score A0 by that definition, read as isotopy, so a diagram of a chiral knot and a diagram of its mirror image show different knots, and 41 A0-I and 43 A0-S positives become negatives (Section 4.1). The A1 prompt calls such a pair "drawings of the same knot", the convention under which we generated the items, so A0 and A1 use "same knot" in different senses. Under the A1 definition of chirality, the 6 A1-I and 10 A1-S items that pair a diagram of an amphichiral knot with a diagram of its mirror image have the answer no, although their two diagrams are isotopic. We keep them with the label yes and report in Appendix J how the scores change if we remove them or score them no. The C1 prompt asks for a code of the drawn diagram in lower-case letters, one per crossing, but the label is the DT code of the prototype's minimal diagram. That code has fewer letters than the drawing has crossings on 94 of the 100 items, and 8 labels contain the upper-case letters that the alphabetical DT code uses for negative entries (Appendix F). The B0 prompt glosses each move by its change in crossing count, the quantity from which we assigned the real-step labels (Appendix D).

F SCORING RULES

We score each response with a fixed function of the response text and the item's label, so no human or model judges an answer. The same code scored all four configurations in the May run (Appendix K), and the September harness for D0 and D1 uses verbatim copies of its parsing functions.

Re-running the scorer on the stored responses of the May run reproduces the stored parsed answer and correctness flag on every row. Scores under the corrected labels (Appendix J) compare the same stored parsed answers with the corrected labels.

Answer extraction. Every prompt asks the model to finish with a line of the form `ANSWER: <answer>` (Appendix E). The scorer takes the last line that begins with `ANSWER:`, ignoring case and leading white space, and keeps the text after the colon. If a response has no such line, the scorer takes its last non-empty line, and a response with no non-empty line gives an empty answer, which is wrong on every task. Among the non-empty A–C responses only 39, all from GPT-5 without thinking, lack an answer line (37 on B0-S and 2 on B0-I). Each is a single line holding a move label and nothing else, and under the corrected labels 20 of them are correct. Had we required an answer line, GPT-5 would score 44.4% instead of 45.7% on the 12 A–C tasks and 27.0% instead of 46.0% on B0-S. The GPT-5 thinking lift over these tasks, 7.8 points as scored and mostly from A3-S and B0-S (Appendix H), would then be 9.1 points.

Normalization. The scorer reads the extracted answer with a rule for each task, and an answer the rule cannot read is scored wrong. For the yes/no tasks (A0–A3 in both variants, and D0) the scorer lower-cases the answer and strips periods, commas, question marks and exclamation marks at either end. It reads `yes`, `y`, `true`, `same` and `matching` as yes, and `no`, `n`, `false`, `different` and `not` as no. Any other answer that begins with `yes` or `no` is read as that word, which would turn "not sure" into no and "yes or no" into yes. No stored A–C answer reaches this last rule, because every yes/no answer is one of the listed words or empty.

For B0 the scorer upper-cases the answer, strips the same punctuation from either end, removes spaces and parentheses, and accepts exactly `R1+`, `R1−`, `R2+`, `R2−`, `R3` and `NOT−CONNECTED`, reading `NOTCONNECTED` and `NOT_CONNECTED` as `NOT−CONNECTED`. Every stored B0 answer normalizes to one of these six tokens. For C0 the scorer reads the first integer in the answer, with an optional minus sign, and compares it with the number of crossings in the drawing. Every stored C0 answer is a bare integer.

For C1 the strict rule lower-cases both the answer and the label, strips white space, the same punctuation, quotation marks and backticks from either end of the answer, and requires the two strings to be equal, so it ignores case. The alphabetical DT code writes negative entries as upper-case letters, and 8 labels contain them. The C1 prompt asked for lower-case letters, so on these items a correct answer could not be written in the requested format. The strict rule would instead accept the all-lower-case string, which encodes a different diagram. No configuration scores on any of these items under either tier. Strict matching also marks as wrong a different valid DT string of the labeled diagram, such as one written from another starting point.

For D1 the scorer upper-cases the answer, strips surrounding punctuation and parentheses, and takes the first character if it is one of A to D, so a word such as "Both" would be read as B.

For D0 and D1 the September harness also records whether each answer came from an answer line, together with a stricter second parse. That parse rejects hedged yes/no answers, a bare `not`, and any D1 answer other than a single option letter, optionally written as `(A)` or `Option A`. Neither changes a D score. Every D answer line holds a bare yes, no or option letter, and the second parse agrees with the first on every row. Only 1 D response lacks an answer line, a GPT-5 response to D1 whose last line is a single letter, and it is wrong under both parses.

C1 permissive tier. The permissive tier credits any string that encodes the labeled diagram. A separate pass takes the first answer line whose text is a single run of letters, optionally in quotation marks, or failing that the last line of the response that consists only of letters. It lower-cases the string, decodes it with Regina's `Link.fromDT` (Burton et al., 1999–2025), and compares the `knotSig()` of the decoded diagram with the stored signature of the prototype's minimal diagram, the diagram that the label encodes. Because of the lower-casing, a correct answer to an item whose label contains upper-case letters cannot score under this tier. With its default options the signature does not change when the crossings of a diagram are relabeled, its orientation is reversed, or it is reflected in a line or turned over (Appendix A). The pass does not simplify the decoded diagram, so an answer scores only if it describes the minimal diagram itself. On the 94 items whose drawing has more crossings than the label has letters, a correct transcription of the drawing fails under both

tiers. Under the permissive tier GPT-5+T scores on 4 items and the other three configurations on none. Every such answer is `cdab`, and every such item has the label `bcda`, the DT code of the figure-eight knot, drawn with 4 to 6 crossings. On the drawing with 4 crossings, `cdab` is also a DT code of the drawn diagram, so strict matching marks a correct transcription wrong. We report the permissive tier as its own scenario in Table 13. Unless marked as permissive, C1 numbers in the paper use the strict rule.

Responses without an answer. The stored responses include 23 empty ones (4 from Claude+T and 19 from GPT-5+T) and 5 GPT-5+T responses to C1 that end in an `ANSWER:` line with nothing after it. All 28 rows belong to A–C tasks. We score them wrong in every scenario and trace each to its cause in Appendix K. If we dropped these rows instead, Claude+T would score 61.2% instead of 61.0% on the A–C tasks, and GPT-5+T 54.3% instead of 53.5%.

Statistics. Accuracy is the share of items scored correct. Overall accuracy weights every item equally, so tasks with more items weigh more, and the macro mean averages the task accuracies. Intervals for a single accuracy are Wilson score intervals at 95% (Wilson, 1927). A thinking lift compares the two configurations of one vendor on the same items. We test it with the exact two-sided McNemar test on the discordant items, those gained and those lost with thinking, and correct the per-task tests with Holm's procedure across the tasks of each vendor. Bootstrap intervals use 10,000 resamples that draw items with replacement within each task, so every resample keeps the task sizes, and report the 2.5th and 97.5th percentiles. Both configurations of a vendor receive the same resampled items, which makes the lift intervals paired. The gap between the -S and -I variants of a family compares different items, so its interval comes from Newcombe's hybrid score method and from a bootstrap that resamples the two variants independently. Resampling indices come from NumPy's default generator with seed 20260923 for the A–C tasks and 20260924 for the rebuilt D tasks. Every comparison with a baseline uses exact fractions, and "at or below" allows no tolerance.

The intervals and tests treat items as independent, although a few items repeat. In A2-I, items 0001 and 0003 are identical, and so are 0016 and 0020. Items 0136 and 0139 of B0-I are built from the same pair of diagrams, drawn separately, and items 0065 and 0067 of B0-S show the same pair of PD codes. Items 0002 and 0003 of C0, and 0061 and 0074 of C1, draw the same diagram in the two textures. A2-I items 0005, 0006 and 0052 draw the diagrams whose PD codes A2-S items 0000, 0001 and 0054 show. We score all of these items as released.

G PER-TASK BREAKDOWNS

We break the score of each task down by kind of item, by true class or by true count, and list the answers that the configurations gave. We count under the primary labels (Appendix J) and use the stored responses, from the May run for the A–C tasks and from the September run for D0 and D1 (Appendix K). Table 12 gives the accuracies with their intervals, and Appendix J.2 gives the constant and cue rules of each task.

A0. On A0-I Claude is right on 13 of the 59 positives and GPT-5+T on 50. GPT-5+T is right on 7 of the 50 negatives that we sampled as two different knots of equal crossing number (Table 8). Claude, Claude+T and GPT-5 are right less often on the 50 Jones-collision negatives, which reuse 18 pairs, than on the negatives of equal crossing number. Claude is right on 78.0 against 88.0% and GPT-5 on 32.0 against 52.0%. GPT-5+T is right more often, on 18.0 against 14.0%. On A0-S every configuration answers yes less often than on A0-I, from 5.0% of the items for Claude to 42.0% for GPT-5+T (Table 7), and Claude is right on 4 of the 57 positives. GPT-5+T gives no answer on 12 A0-S items, which we score as wrong (Table 22).

A1. On A1-I Claude and Claude+T answer yes on 28.0 and 35.0% of the items and GPT-5 and GPT-5+T on 86.0 and 91.0%, so the two models err in opposite directions. No configuration scores above the constant answer no (52.0%). On A1-S Claude is right on 49 of the 50 pairs of a chiral knot with one diagram mirrored but on 13 of the 30 pairs of equal chirality, and Claude+T on all 30.

Table 7: Answers on the yes/no tasks of family A under the primary labels. Labels gives the number of items labeled yes and no. For each configuration, Yes is the percentage of items it answered yes, and Bal. is its balanced accuracy (%), the mean of its accuracy on the items labeled yes and its accuracy on the items labeled no. Giving the same answer to every item scores a balanced accuracy of 50. An empty response (Appendix F) counts as neither answer and as wrong.

Task	Labels		Claude		Claude+T		GPT-5		GPT-5+T	
	yes	no	Yes	Bal.	Yes	Bal.	Yes	Bal.	Yes	Bal.
A0-I	59	141	19.5	51.8	17.0	53.6	55.5	52.7	84.0	50.5
A0-S	57	143	5.0	51.4	14.5	57.2	20.5	56.5	42.0	55.9
A1-I	48	52	28.0	45.1	35.0	46.4	86.0	47.4	91.0	50.6
A1-S	49	51	17.0	65.4	50.0	93.0	50.0	43.0	67.0	68.4
A2-I	50	50	39.0	67.0	33.0	65.0	39.0	45.0	84.0	50.0
A2-S	50	50	49.0	97.0	49.0	99.0	48.0	96.0	50.0	100.0
A3-I	100	100	59.5	90.5	59.5	90.5	29.0	63.0	71.5	74.5
A3-S	50	50	0.0	50.0	0.0	50.0	2.0	48.0	49.0	99.0

A2. GPT-5+T answers yes on 84.0% of the A2-I items and the other three configurations on 33.0 to 39.0%; 50 of the 100 pairs have equal counts. On A2-S, where the answer is whether the two codes have the same number of tuples, every configuration scores at least 96.0%.

A3. Claude and Claude+T are right on all 100 A3-I positives, each of which shows one drawing in two textures, and answer yes on 19 of the 100 negatives. GPT-5 is right on 42 of the positives. On A3-S Claude and Claude+T answer no on every item and GPT-5 answers yes on 2 items, whereas GPT-5+T gives the answer of the list-order rule on 99 of the 100 items.

B0. GPT-5 answers R3 on 165 of the 200 B0-I items and is right on 6 of the 164 items of other classes, and GPT-5+T answers R3 on 147 (Table 9). Claude+T answers NOT-CONNECTED on 66 B0-I items; 25 have that label. On B0-S, Claude, Claude+T and GPT-5+T are right on at least 86 of the 90 real-step items. GPT-5 answers NOT-CONNECTED on 29 B0-S items and is right on 5 of the 10 that have this label.

C0. At 8 crossings the configurations give the exact count on 5 to 13 of the 20 items. At 11 or more crossings they do so on at most 3 of the 20 items of any one count (Table 10). Most wrong answers are too low, with undercounts on 85, 81, 54 and 70 of the 100 items for Claude, Claude+T, GPT-5 and GPT-5+T.

C1. No configuration gives the label string on any of the 100 items. Under the permissive tier GPT-5+T is credited on 4 figure-eight items, each time for the answer `cdab` (Appendix J.3). Apart from empty answers and words such as `unknown`, GPT-5 answers with a string of letters on 1 item and GPT-5+T on 58 (Appendix L).

D0 and D1. Table 11 splits the D0 items into those whose code matches the drawing and, for the others, the kind of pair each was built from. It breaks D1 down by the kind and the letter of the option each configuration chose, and Fig. 4 plots the same counts as shares. GPT-5 answers no on all 200 D0 items, and the other three configurations answer yes on 20.5, 53.5 and 47.5% of them (Claude, Claude+T, GPT-5+T). None of the three answers yes more often when the code matches the drawing than when it differs from the drawing at one crossing. Claude answers yes on 20 of the 100 matching codes and on 10 of the 33 codes with one crossing switched, Claude+T on 52 and 19, and GPT-5+T on 44 and 19. On D1 the configurations choose the key or its twin with one crossing switched on 48.5 to 57.0% of the items, against 50% for a uniform choice (Claude+T at the upper end, exact binomial $p = 0.056$), and within that pair they choose the key on 45.4 to 55.7%. The letters they choose are far from uniform (χ^2 test against equal shares, $p \leq 3.9 \times 10^{-4}$ for every configuration), and GPT-5 chooses A on 2 items, although A is the key on 50.

Table 8: Correct answers by kind of item on the A tasks, under the primary labels, with the label of each kind in parentheses. n is the number of items of the kind in the image (-I) and PD-code (-S) variant, and a dash marks a kind that the variant does not have. A0 negatives pair two knots of equal crossing number, or the two knots of a Jones-collision pair (Section 2), strict when they also share the Alexander polynomial, the classical signature and the determinant. A1 pairs two diagrams of one knot. Listed knots are those on the list of amphichiral prototypes that the sampler used, which holds chiral knots as well (Appendix J.1). Rows marked * are the ill-posed A1 items, which we label yes. The A0 row of a chiral knot and its mirror image and the A1 rows of a listed chiral knot and of an unlisted amphichiral knot carry corrected labels. A2 pairs two diagrams of one knot and one chirality, sorted by the difference of their crossing counts. A3 pairs two diagrams with the same prototype, chirality and crossing count; a positive of A3-I shows one drawing in two textures, and a positive of A3-S gives two codes of one diagram, the second with its arcs renumbered and its crossings reordered (Appendix J.2).

	-I					-S				
Kind of item (label)	n	Claude	Claude+T	GPT-5	GPT-5+T	n	Claude	Claude+T	GPT-5	GPT-5+T
A0: same knot?										
One knot (yes)	59	13	13	35	50	57	4	15	17	31
Chiral knot and its mirror image (no)	41	32	32	23	7	43	40	32	28	22
Two knots of equal crossing number (no)	50	44	47	26	7	50	50	48	48	32
Jones-collision pair, strict (no)	10	7	8	3	1	9	9	9	8	6
Jones-collision pair, Jones only (no)	40	32	33	13	8	41	38	37	35	22
A1: same chirality?										
Chiral knot, same chirality (yes)	30	5	5	24	28	30	13	30	17	29
Chiral knot, one diagram mirrored (no)	49	34	31	6	4	50	49	46	22	25
Listed knot, same chirality (yes)	12	3	7	10	11	9	1	8	2	8
Listed chiral knot, one diagram mirrored (no)	3	1	1	0	1	1	1	1	0	0
Listed amphichiral knot, one diagram mirrored* (yes)	5	2	2	5	4	10	2	8	2	6
Unlisted amphichiral knot, one diagram mirrored* (yes)	1	1	1	1	1	–	–	–	–	–
A2: same number of crossings?										
Equal counts (yes)	50	28	24	17	42	50	48	49	47	50
Counts differ by 1 (no)	20	15	17	7	1	19	18	19	18	19
Counts differ by 2 (no)	14	10	10	9	3	12	12	12	12	12
Counts differ by 3 (no)	5	4	4	2	1	8	8	8	8	8
Counts differ by 4 (no)	3	2	2	3	1	3	3	3	3	3
Counts differ by 5 (no)	3	3	3	3	1	5	5	5	5	5
Counts differ by 6 (no)	1	1	1	1	0	1	1	1	1	1
Counts differ by 7 (no)	2	2	2	2	1	1	1	1	1	1
Counts differ by 8 (no)	2	2	2	1	0	1	1	1	1	1
A3: same diagram?										
One diagram (yes)	100	100	100	42	96	50	0	0	0	49
Two diagrams (no)	100	81	81	84	53	50	50	50	48	50

Table 9: B0 under the corrected labels. (a) Correct answers by true class, with the number of items n of each class in B0-I and B0-S. Equal counts are the items whose two diagrams have the same number of crossings, R3 or NOT-CONNECTED, which the count-difference rule (Appendix J.2) cannot tell apart. (b) Number of items on which each configuration gave each answer. Every B0 response gave one of the six answers.

(a) Correct answers by true class

	B0-I					B0-S				
Class	n	Claude	Claude+T	GPT-5	GPT-5+T	n	Claude	Claude+T	GPT-5	GPT-5+T
R1$^+$	35	11	11	0	5	16	15	16	3	16
R1$^-$	34	15	16	0	4	17	17	17	1	17
R2$^+$	36	6	4	0	2	18	18	18	7	18
R2$^-$	34	11	8	2	3	19	19	19	13	19
R3	36	9	9	31	29	20	19	16	17	20
NOT-CONNECTED	25	9	13	4	3	10	6	6	5	8
Equal counts	40	10	11	31	29	22	20	17	17	21

(b) Answers given

	B0-I				B0-S			
Answer	Claude	Claude+T	GPT-5	GPT-5+T	Claude	Claude+T	GPT-5	GPT-5+T
R1$^+$	23	22	0	14	15	16	4	16
R1$^-$	35	32	2	10	17	17	1	17
R2$^+$	16	11	0	11	20	20	16	19
R2$^-$	32	26	10	11	20	20	25	19
R3	54	43	165	147	20	17	25	21
NOT-CONNECTED	40	66	23	7	8	10	29	8

Table 10: C0 answers by the true number of crossings of the drawing, 20 items per count. Exact: answers equal to the count. ± 1: answers that differ from it by at most one, exact answers included. Under: answers below it. Every C0 response gave an integer.

Crossings	Claude			Claude+T			GPT-5			GPT-5+T		
	Exact	± 1	Under	Exact	± 1	Under	Exact	± 1	Under	Exact	± 1	Under
8	9	15	7	13	16	4	5	11	7	7	11	8
11	0	4	18	1	3	17	0	5	10	0	9	12
14	0	0	20	0	0	20	1	2	13	3	3	15
17	0	0	20	0	0	20	1	1	11	0	6	16
20	0	0	20	0	0	20	1	1	13	1	1	19
All	9	19	85	14	19	81	8	20	54	11	30	70

Table 11: D0 and D1 of the September run. (a) D0 by the kind of pair the item was built from: the code shown matches the drawing (label yes), or the drawing is the other member of a pair made of a stored diagram and the same diagram with one crossing switched, with every crossing switched, or another stored diagram of the same knot, chirality and crossing count (label no). For each configuration, Corr. counts correct answers and Yes the answers yes. (b) D1 by the kind of option each configuration chose, named relative to the drawing. (c) D1 by the option letter chosen.

(a) D0

		Claude		Claude+T		GPT-5		GPT-5+T	
Kind of item	n	Corr.	Yes	Corr.	Yes	Corr.	Yes	Corr.	Yes
Code matches the drawing (yes)	100	20	20	52	52	0	0	44	44
One crossing switched (no)	33	23	10	14	19	33	0	14	19
All crossings switched (no)	34	26	8	16	18	34	0	13	21
Other diagram of the knot (no)	33	30	3	15	18	33	0	22	11

(b) D1, kind of option chosen

Option	Claude	Claude+T	GPT-5	GPT-5+T
Correct option	49	54	54	44
Drawing, one crossing switched	57	60	43	53
Other diagram	50	42	49	48
Other diagram, one crossing switched	44	44	54	55

(c) D1, letter chosen

Letter	Claude	Claude+T	GPT-5	GPT-5+T
A	31	29	2	40
B	39	37	39	32
C	75	68	78	57
D	55	66	81	71

H RESULTS UNDER EVERY SCENARIO

Table 12 gives the accuracy of each configuration on each task under the primary scoring, with corrected labels, A0 scored as ambient isotopy and C1 scored by strict string match (Appendix J). The primary scoring keeps the ill-posed A1 items, which pair an amphichiral knot with its mirror image, with the label yes. Per-task intervals and the interval of the item-weighted mean are Wilson intervals (Appendix F), and the macro mean has a stratified bootstrap interval, as in Table 25. The D0 and D1 rows come from the September run, so means and counts over all 14 tasks pool responses from two runs with different output caps (Appendix K). Only the means over the 12 A–C tasks compare the configurations within one run.

Table 12: Accuracy (%) per task and configuration, with Wilson 95% intervals. N is the number of items and Random the accuracy of a uniform guess. The two means over the 12 A–C tasks and over all 14 tasks are defined as in Table 25, and labels and scoring follow Table 1.

Task	N	Random	Claude	Claude+T	GPT-5	GPT-5+T
A0-I	200	50.0	64.0 [57.1, 70.3]	66.5 [59.7, 72.7]	50.0 [43.1, 56.9]	36.5 [30.1, 43.4]
A0-S	200	50.0	70.5 [63.8, 76.4]	70.5 [63.8, 76.4]	68.0 [61.3, 74.1]	56.5 [49.6, 63.2]
A1-I	100	50.0	46.0 [36.6, 55.7]	47.0 [37.5, 56.7]	46.0 [36.6, 55.7]	49.0 [39.4, 58.7]
A1-S	100	50.0	66.0 [56.3, 74.5]	93.0 [86.3, 96.6]	43.0 [33.7, 52.8]	68.0 [58.3, 76.3]
A2-I	100	50.0	67.0 [57.3, 75.4]	65.0 [55.3, 73.6]	45.0 [35.6, 54.8]	50.0 [40.4, 59.6]
A2-S	100	50.0	97.0 [91.6, 99.0]	99.0 [94.6, 99.8]	96.0 [90.2, 98.4]	100.0 [96.3, 100.0]
A3-I	200	50.0	90.5 [85.6, 93.8]	90.5 [85.6, 93.8]	63.0 [56.1, 69.4]	74.5 [68.0, 80.0]
A3-S	100	50.0	50.0 [40.4, 59.6]	50.0 [40.4, 59.6]	48.0 [38.5, 57.7]	99.0 [94.6, 99.8]
B0-I	200	16.7	30.5 [24.5, 37.2]	30.5 [24.5, 37.2]	18.5 [13.7, 24.5]	23.0 [17.7, 29.3]
B0-S	100	16.7	94.0 [87.5, 97.2]	92.0 [85.0, 95.9]	46.0 [36.6, 55.7]	98.0 [93.0, 99.5]
C0	100	0.0	9.0 [4.8, 16.2]	14.0 [8.5, 22.1]	8.0 [4.1, 15.0]	11.0 [6.3, 18.6]
C1	100	0.0	0.0 [0.0, 3.7]	0.0 [0.0, 3.7]	0.0 [0.0, 3.7]	0.0 [0.0, 3.7]
D0	200	50.0	49.5 [42.7, 56.4]	48.5 [41.7, 55.4]	50.0 [43.1, 56.9]	46.5 [39.7, 53.4]
D1	200	25.0	24.5 [19.1, 30.9]	27.0 [21.3, 33.5]	27.0 [21.3, 33.5]	22.0 [16.8, 28.2]
Means over the 12 A–C tasks						
Item-weighted			58.8 [56.3, 61.1]	61.0 [58.6, 63.4]	45.7 [43.3, 48.1]	53.5 [51.1, 55.9]
Macro			57.0 [55.0, 59.0]	59.8 [58.0, 61.8]	44.3 [42.1, 46.5]	55.5 [53.6, 57.3]
Means over the 14 tasks						
Item-weighted			54.4 [52.2, 56.6]	56.4 [54.2, 58.5]	44.3 [42.1, 46.4]	49.7 [47.5, 51.8]
Macro			54.2 [52.4, 56.0]	56.7 [54.9, 58.4]	43.5 [41.5, 45.4]	52.4 [50.7, 54.1]

We give the aggregate numbers under five further scenarios in Table 13, so that a reader can see which conclusions depend on a decision about labels or scoring. Four of the scenarios change one decision of the primary scoring, and the fifth, original labels, undoes every correction. Every decision concerns an A–C task, so the table covers the 12 A–C tasks, and the D scores are the same in every scenario. The number of A–C pairs at or below the uniform random baseline is 16 of 48 in every scenario except C1 permissive. There the 4 permissive credits of GPT-5+T raise its C1 score above zero, and the count falls to 15. With A0 scored as designed, so that a knot and its mirror image count as one knot, the best configuration scores at most 65% on both A0 tasks. The number of tasks at or below that level then rises from 4 to 6. The number of pairs at or below the best tier-1 rule of their task (Appendix J.2) falls from 30 to 24. The item-weighted accuracies of GPT-5+T and Claude, 53.5 and 58.8% under the primary scoring, become 55.1 and 55.0%, or 881 and 880 correct items. Appendix J traces these changes to the relabeled A0 items.

Under the primary scoring Claude's thinking lift comes mostly from A1-S and that of GPT-5 from A3-S and B0-S (Fig. 11). Without A1-S, Claude's lift over the other 11 A–C tasks is 0.6 points (unadjusted exact McNemar $p = 0.452$), and without A3-S and B0-S the lift of GPT-5 over the other 10 is 1.6 points ($p = 0.303$). On the D items the lift is 0.8 points for Claude ($p = 0.865$) and -4.3 for GPT-5 ($p = 0.181$). Claude's lift is significant at the 5% level (unadjusted exact McNemar test) in every scenario except A1 amphichiral scored no, which gives the ill-posed A1 items the label no ($p = 0.058$). The lift of GPT-5 lies between 7.3 and 10.5 points, with p at most 4.6×10^{-7} in all six scenarios (Table 13).

Table 13: Accuracy, baseline counts and thinking lifts over the 12 A–C tasks under six labeling and scoring scenarios. The scenarios do not affect D0 and D1. Primary uses the corrected labels, which relabel 20 B0 NOT-CONNECTED items, 4 A1-I items and 1 A1-S item. It scores A0 as topological equivalence, under which a chiral knot and its mirror image are different knots, so 41 A0-I and 43 A0-S positives become negatives. It scores C1 by strict string match, and it keeps the 6 A1-I and 10 A1-S items that pair an amphichiral knot with its mirror image, with the label yes (Appendix J.1). Original labels uses the labels the sampler assigned and scores A0 as designed, counting a knot and its mirror image as the same knot. A0 as designed is Primary with A0 scored as in Original labels. C1 permissive also credits a C1 answer that decodes to a diagram with the same Regina signature as the label's diagram. A1 amphichiral dropped removes those A1 items, and A1 amphichiral scored no gives them the label no. Panel (a) gives item-weighted accuracy (%) and the tasks on which the best configuration scores at most 65%. Its two counts are the numbers of the 48 (task, configuration) pairs at or below the uniform random baseline and at or below the best tier-1 rule of their task (Appendix J.2). Panel (b) gives the thinking lift in points with its paired bootstrap 95% interval and exact McNemar p.

(a) Accuracy and baseline counts

Scenario	Claude	Claude+T	GPT-5	GPT-5+T	At or below		Best $\leq 65\%$
					random	tier 1	(number: tasks)
Primary	58.8	61.0	45.7	53.5	16	30	4: A1-I, B0-I, C0, C1
Original labels	54.2	57.9	44.3	54.3	16	25	6: A0-I, A0-S, A1-I, B0-I, C0, C1
A0 as designed	55.0	58.3	44.6	55.1	16	24	6: A0-I, A0-S, A1-I, B0-I, C0, C1
C1 permissive	58.8	61.0	45.7	53.8	15	30	4: A1-I, B0-I, C0, C1
A1 amphichiral dropped	59.0	60.9	45.6	53.4	16	30	4: A1-I, B0-I, C0, C1
A1 amphichiral scored no	59.1	60.6	45.7	52.9	16	30	4: A1-I, B0-I, C0, C1

(b) Thinking lift

	Claude+T − Claude		GPT-5+T − GPT-5	
Scenario	points	p	points	p
Primary	+2.3 [0.8, 3.7]	0.004	+7.8 [5.3, 10.4]	4.8×10^{-8}
Original labels	+3.8 [2.3, 5.3]	1.3×10^{-6}	+10.0 [7.4, 12.6]	1.9×10^{-12}
A0 as designed	+3.3 [1.8, 4.7]	2.3×10^{-5}	+10.5 [7.9, 13.1]	1.5×10^{-13}
C1 permissive	+2.3 [0.8, 3.7]	0.004	+8.1 [5.4, 10.7]	2.0×10^{-8}
A1 amphichiral dropped	+1.9 [0.4, 3.3]	0.015	+7.7 [5.1, 10.3]	8.8×10^{-8}
A1 amphichiral scored no	+1.5 [0.0, 2.9]	0.058	+7.3 [4.6, 9.9]	4.6×10^{-7}

Table 14 lists the pairs at or below the uniform random baseline under the primary scoring, 22 of the 56 pairs over all 14 tasks and 18 outside C1. On A1-I, C1 and D0 all four configurations are at or below it, on A3-S all but GPT-5+T, and on D1 Claude and GPT-5+T. GPT-5+T on A0-I is the only pair whose whole Wilson interval lies below the baseline. No D score differs from its baseline by an exact binomial test (smallest $p = 0.358$, GPT-5+T on D0), so the D pairs above the baseline give no evidence of a score above it. On A0, C0 and C1 a constant answer scores well above the uniform random baseline. Answering no scores 70.5 and 71.5% on A0-I and A0-S, where the primary labels are unbalanced. Answering any one count scores 20.0% on C0, and answering `bcda` scores 31.0% on C1, although the uniform random baseline of both is 0; C0 and C1 lie above that baseline only because it is 0. We therefore also count the pairs at or below the best tier-1 rule of their task, 30 of the 48 A–C pairs and 36 of all 56. On D0 and D1 the best tier-1 rule is a constant answer, which scores the uniform random baseline.

Table 14: The 22 of the 56 (task, configuration) pairs over the 14 tasks whose accuracy is at or below the uniform random baseline, with no tolerance. Accuracy in % with a Wilson 95% interval. The random baseline of C0 and C1 is 0, so a configuration with no correct C1 answer is counted. * marks a pair whose whole interval lies below the baseline. Labels and scoring follow Table 1.

Task	Random	Configuration	Accuracy and interval	
A0-I	50.0	GPT-5	50.0	[43.1, 56.9]
		GPT-5+T	36.5*	[30.1, 43.4]
A1-I	50.0	Claude	46.0	[36.6, 55.7]
		Claude+T	47.0	[37.5, 56.7]
		GPT-5	46.0	[36.6, 55.7]
		GPT-5+T	49.0	[39.4, 58.7]
A1-S	50.0	GPT-5	43.0	[33.7, 52.8]
A2-I	50.0	GPT-5	45.0	[35.6, 54.8]
		GPT-5+T	50.0	[40.4, 59.6]
A3-S	50.0	Claude	50.0	[40.4, 59.6]
		Claude+T	50.0	[40.4, 59.6]
		GPT-5	48.0	[38.5, 57.7]
C1	0.0	Claude	0.0	[0.0, 3.7]
		Claude+T	0.0	[0.0, 3.7]
		GPT-5	0.0	[0.0, 3.7]
		GPT-5+T	0.0	[0.0, 3.7]
D0	50.0	Claude	49.5	[42.7, 56.4]
		Claude+T	48.5	[41.7, 55.4]
		GPT-5	50.0	[43.1, 56.9]
		GPT-5+T	46.5	[39.7, 53.4]
D1	25.0	Claude	24.5	[19.1, 30.9]
		GPT-5+T	22.0	[16.8, 28.2]

I SUPPLEMENTARY FIGURES

The first six figures plot accuracy against the uniform random baseline on all 14 tasks (Fig. 3), D0 and D1 by kind of item and kind of answer (Fig. 4), crossing counts by the true count (Fig. 5), move prediction by true class (Fig. 6), the output tokens of every A–C response (Fig. 7), and the cost and output tokens per correct A–C answer (Fig. 8). The last two describe the corpus of Section 3 (Figs. 9 and 10). All score figures use the primary scoring, with the stored May responses for the A–C tasks and the September responses for D0 and D1. Tables 23 and 24 give the output tokens and cost of the September run.

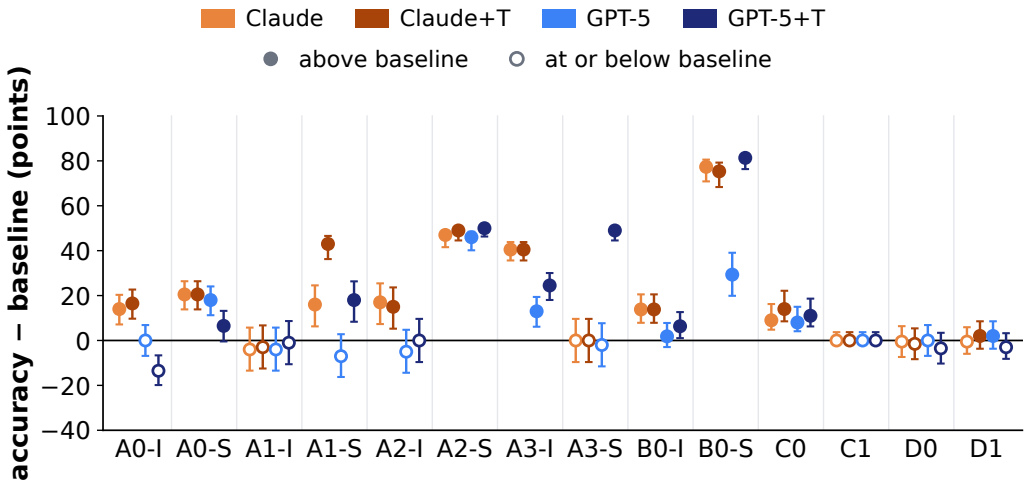


Figure 3: Accuracy minus the uniform random baseline, in percentage points, for each of the 56 (task, configuration) pairs over the 14 tasks under the primary scoring. Error bars are Wilson 95% intervals of the accuracy, shifted by the baseline (Table 12 gives the number of items of each task). Baselines are 1/2 for the yes/no tasks (A0 to A3 and D0), 1/6 for B0, 1/4 for D1 and 0 for C0 and C1, so on C0 and C1 no configuration can fall below its baseline. Hollow markers: accuracy at or below the baseline, compared exactly (22 pairs, 18 of them outside C1); filled markers: above it. C0 lies above its baseline only because the baseline is 0: answering one constant count scores 20.0% on C0, above every configuration, and always answering the most frequent C1 code (`bcda`) scores 31.0%, against 0 for every configuration. On A0-I and A0-S, whose primary labels are unbalanced, answering no to every item scores 70.5 and 71.5%, also above every configuration (Table 19).

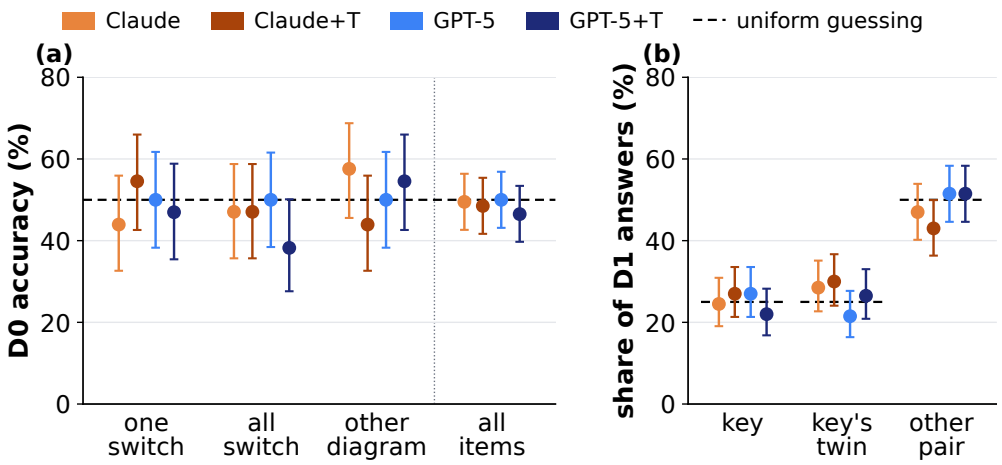


Figure 4: D0 and D1 of the September run by kind of item and kind of answer, under the primary scoring. (a) D0 accuracy on the items built from each kind of pair, the drawn diagram with a copy that has one crossing switched, with a copy that has all crossings switched, or with another endpoint of the same knot, chirality and crossing count (66, 68 and 66 items), and on all 200 items. Half of the items of each kind give the code of the drawing and half the code of its partner, so a constant answer scores 50%. (b) Share of the 200 D1 answers that name the key, the key's twin with one crossing switched, or one of the two options of the other pair. Dashed lines: uniform guessing (50% in (a); 25, 25 and 50% in (b)). Error bars are Wilson 95% intervals. No accuracy on D0 or D1 differs from the uniform baseline by an exact binomial test (smallest $p = 0.358$, GPT-5+T on D0). The largest share of D1 answers inside the key's pair is 57.0% (Claude+T, $p = 0.056$ against 50%). GPT-5 answers no on every D0 item.

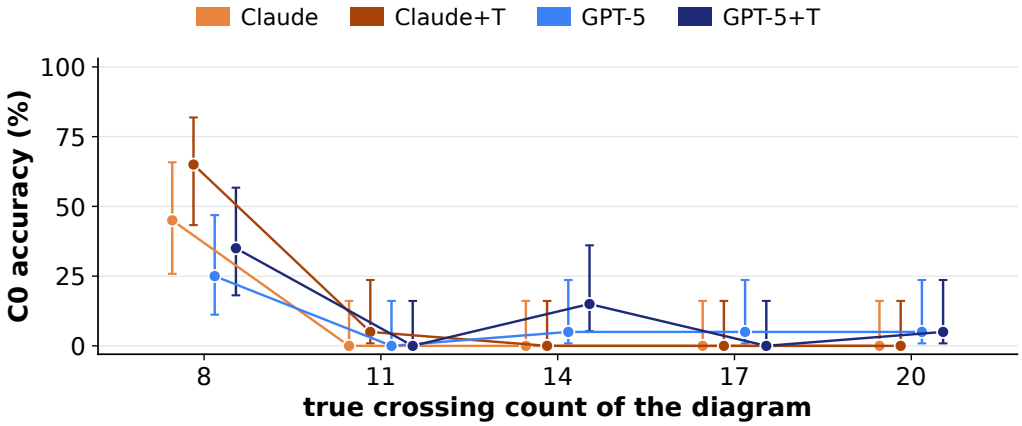


Figure 5: C0 accuracy (exact integer match) by the true crossing count of the diagram, for each configuration, 20 items per count and 100 in total. Error bars are Wilson 95% intervals. At 8 crossings the configurations score 45.0, 65.0, 25.0 and 35.0% (Claude, Claude+T, GPT-5, GPT-5+T); at 11 or more crossings no configuration answers more than 3 of the 20 items of any count correctly. The uniform random baseline of an open-ended count is 0, and answering any one of the five counts 8, 11, 14, 17 and 20 on every item scores 20.0%, more than any configuration scores on C0.

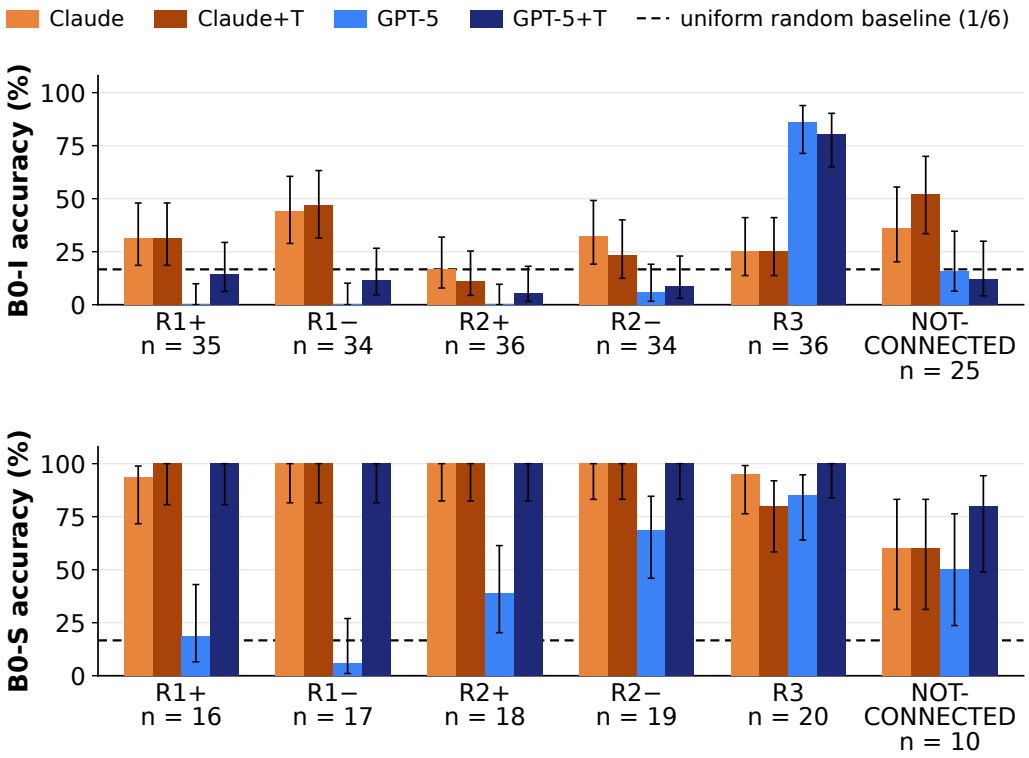


Figure 6: B0 accuracy by true class under the corrected labels, for B0-I (top, 200 items) and B0-S (bottom, 100 items); n under each class is its number of items. The classes are the five move labels $R1^+$, $R1^-$, $R2^+$, $R2^-$ and R3 (Appendix D) and NOT-CONNECTED. After we relabeled 10 B0-I and 10 B0-S NOT-CONNECTED items with their true move, 25 B0-I items and 10 B0-S items keep the label NOT-CONNECTED (Appendix J.1). Error bars are Wilson 95% intervals; the dashed line is the uniform random baseline of 1/6.

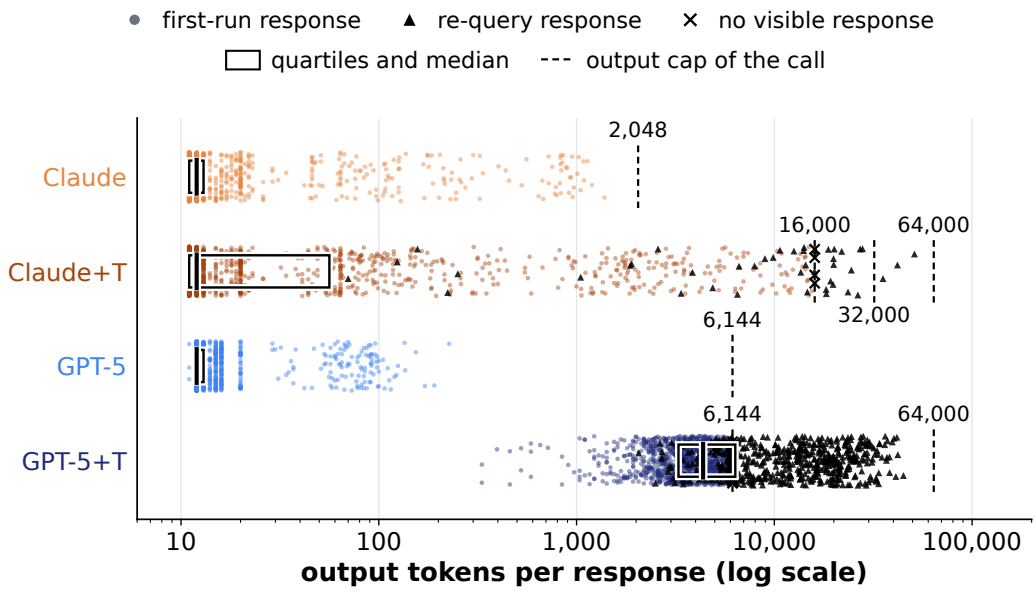


Figure 7: Output tokens per response on the 1,600 A–C items of each configuration (log scale), as reported by the API, reasoning tokens included. Each point is one response, jittered vertically. Circles are first-run responses and triangles are responses to re-queries of items whose first response was empty or had no parseable answer. Crosses are responses with no visible text (4 Claude+T responses at 16,000 tokens and 19 GPT-5+T responses at 6,144, all scored wrong). Boxes span the quartiles, with the median as a thick line. Dashed vertical lines mark the output-token caps of the requests that produced the responses (Table 20), 2,048 for Claude and 6,144 for GPT-5. For Claude+T they are 16,000 for the first run and 32,000 or 64,000 for its re-queries, and for GPT-5+T 6,144 for the first run and 64,000 for its re-queries. Table 21 gives the number of responses at each cap. The first-run responses include the 4 Claude+T and 24 GPT-5+T responses whose re-query gave no usable response (Appendix K).

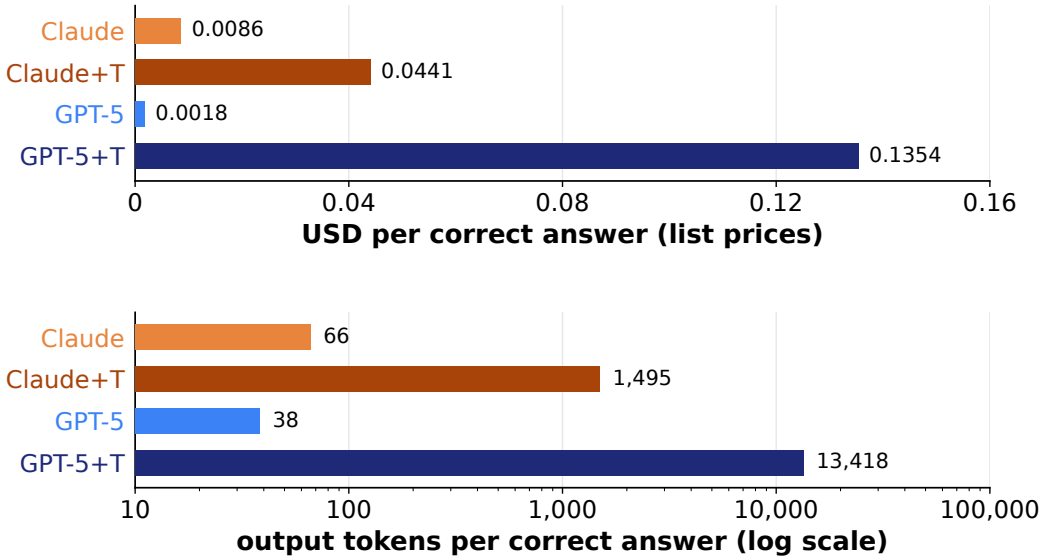


Figure 8: Cost and output tokens per correct answer on the 1,600 A–C items of each configuration, primary scoring. Top: list-price cost of the stored input and output tokens divided by the number of correct answers, in US dollars. The list prices per million input and output tokens are 5 and 25 US dollars for Claude Opus 4.7 and 1.25 and 10 for GPT-5. Bottom: output tokens, reasoning tokens included, divided by the number of correct answers (log scale). We count only the stored May responses to the A–C items. The plotted cost therefore leaves out the first-run responses that re-queries replaced (43 of Claude+T and 501 of GPT-5+T) and the other requests listed in Appendix K, and it is a lower bound on what the run cost.

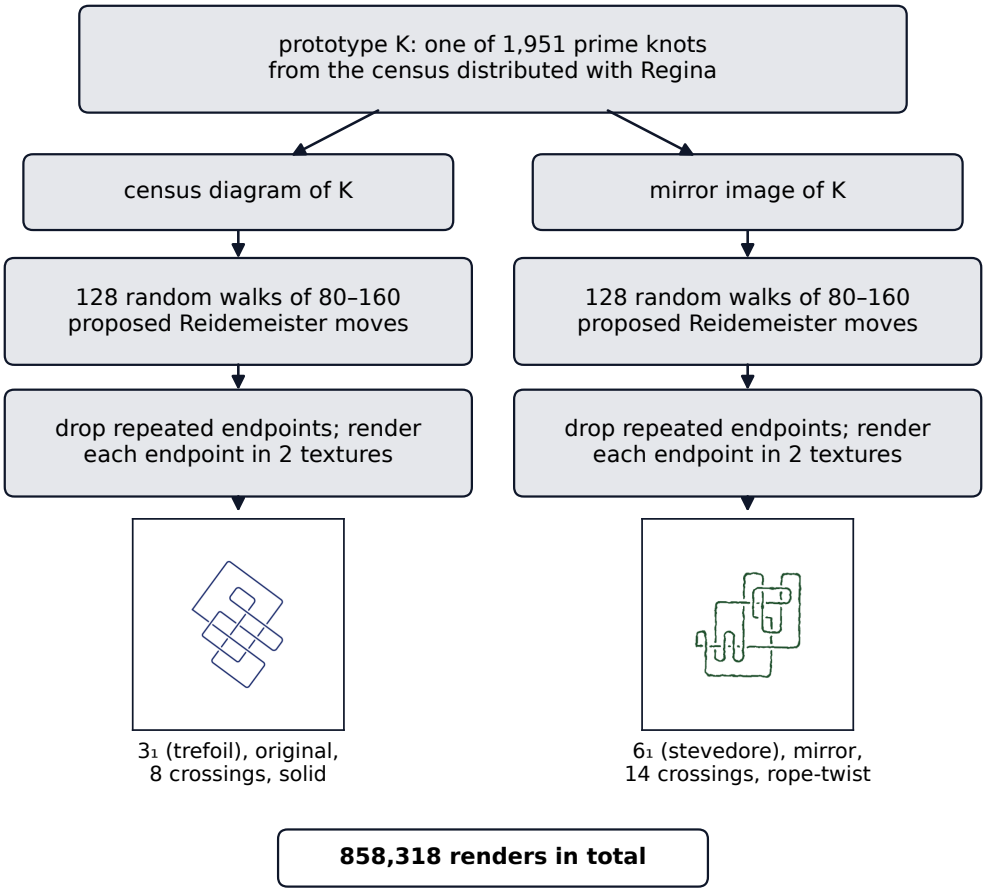


Figure 9: Corpus construction for one prototype K. For each of the 1,951 prototypes we run 128 random walks from the census diagram and 128 from its mirror image. A walk takes 80 to 160 proposal steps, and only 38.9% of the steps change the diagram (23,320,789 of 59,932,703); 6,604 walks of non-alternating prototypes never change their diagram. We drop walks whose endpoint repeats an earlier endpoint of the same prototype and chirality by our renumbering key (68,139 of 499,456 walks); the key misses 620 repeated diagrams (Appendix B). We render each remaining endpoint in the solid and the rope-twist texture, and the corpus keeps the 858,318 renders that pass the lint check and the layout step (Section 3). The two insets are corpus images. The solid one comes from a walk seeded with the census diagram of the trefoil (census 3a_1), the rope-twist one from a walk seeded with the mirror image of the census diagram of the stevedore knot (census 6a_3). Like every corpus image, each draws its stored PD code with every crossing switched (Section 3).

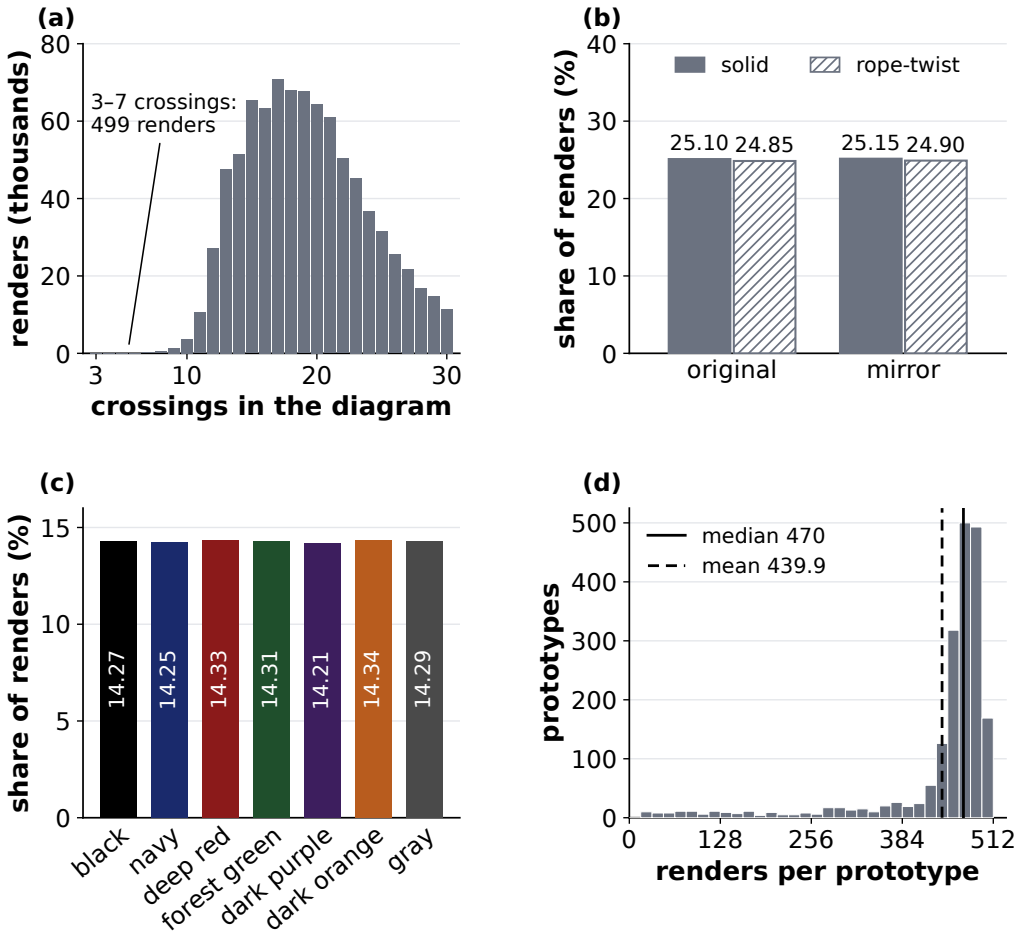


Figure 10: Composition of the 858,318 corpus renders. (a) Crossings in the rendered diagram (3 to 30; 499 renders have 7 or fewer). (b) Share of renders by chirality of the walk's seed (the census diagram or its mirror image) and texture. The chirality names the stored record; every image draws that record with all crossings switched, a diagram of the mirror image (Section 3). (c) Share of renders by line color, each bar drawn in the color the renderer uses; a uniform draw gives 1/7 of the renders to each color. (d) Renders per prototype over the 1,951 prototypes (bins of 16; median 470, mean 439.9, at most 512 = 2 chiralities $\times$ 128 walks $\times$ 2 textures).

Table 15: B0 items labeled NOT-CONNECTED whose two diagrams are one Reidemeister move apart, with the move from D_t to D_{t+1} that the single-move search finds. The primary scores use this move as the label.

B0-I item	Move	B0-S item	Move
B0_I__0166	R2$^+$	B0_S__0083	R3
B0_I__0168	R2$^+$	B0_S__0085	R2$^-$
B0_I__0170	R2$^+$	B0_S__0086	R2$^+$
B0_I__0171	R1$^+$	B0_S__0087	R1$^-$
B0_I__0175	R3	B0_S__0089	R2$^+$
B0_I__0180	R1$^-$	B0_S__0092	R3
B0_I__0182	R2$^-$	B0_S__0093	R2$^-$
B0_I__0186	R3	B0_S__0095	R3
B0_I__0193	R3	B0_S__0097	R3
B0_I__0197	R1$^+$	B0_S__0099	R2$^-$

J LABEL CORRECTIONS AND CUE RULES

After the evaluation we checked every label against the stored PD codes of its item with Regina 7.4.1 and SnapPy 3.3.2 (Burton et al., 1999–2025; Culler et al., 2026). We also asked of every task what else in the items predicts the label. The first check finds wrong labels, which make a reported score wrong, and we correct them (Appendix J.1). It also finds 84 A0 labels that depend on how the prompt's "topologically equivalent" is read, and we relabel them under ambient isotopy. We call the labels that the sampler assigned the original labels and the corrected set the primary labels. The main text reports scores under the primary labels, and Table 13 gives them under the original labels as well. The second check finds cues. A cue is a property of the items that predicts the label without the operation the task was built to test. It leaves every label correct but opens a second route to the answer, and for each cue we report a rule that takes that route (Appendix J.2). C1 has a defect of a third kind. Its labels are correct, but they record something other than what its prompt asks the model to write (Appendix J.3).

J.1 LABEL CORRECTIONS

B0 items labeled NOT-CONNECTED. We built a real-step B0 item from two consecutive, distinct stored diagrams of one walk, and a NOT-CONNECTED item from two stored diagrams of one walk at least five stored diagrams apart. Such a pair can still be a single move apart, because the walk archive stores a rejected proposal as a repeat of the previous diagram and a walk can return to an earlier diagram. For every B0 item we enumerated every R1$^\pm$, R2$^\pm$ and R3 move that applies to D_t or to D_{t+1} and compared each result with the other diagram. We compared diagrams twice, by a canonical form of the PD code and by Regina's canonical signature with reflection disallowed. A second search, which removes R1 kinks and R2 bigons directly in the PD code and calls Regina only for R3, agrees with the enumeration on all 300 B0 items. As a check on completeness, our own face tracing counts the same number of available R1$^-$, R2$^-$ and R3 moves as Regina on all 600 diagrams in B0.

The search finds the labeled move on every real-step item (165 in B0-I and 80 in B0-S), so the real-step labels stand. It also finds a single connecting move for 20 of the 55 NOT-CONNECTED items, namely 10 of 35 in B0-I and 10 of 20 in B0-S. Table 15 lists them with the move found, and we use that move as their label in the primary scores.

Under the original labels the configurations scored between 30.0 and 40.0% on the NOT-CONNECTED class of B0-S. On the 10 items that remain NOT-CONNECTED they score between 50.0 and 80.0%. On the 10 relabeled B0-S items they had named the move found on 10, 9, 7 and 10 items, answers that the original labels scored as wrong. Most of the low NOT-CONNECTED accuracy on B0-S therefore came from the wrong labels. B0-S accuracy rises for every configuration, and B0-I accuracy falls for Claude+T and rises for the other three (Table 18).

A1 labels and the list of amphichiral knots. To build A1 the sampler needs to know which test prototypes are amphichiral, and it reads this from a hard-coded list of 19 prototype identifiers. We copied the amphichiral knots of the Rolfsen table up to ten crossings into the list, except 10_{123}, but wrote each Rolfsen index as a corpus identifier. The corpus identifiers follow the Hoste–Thistlethwaite numbering of the census tables (Hoste et al., 1998; Burton, 2020), which differs from Rolfsen's, so most entries name a different knot. The entry written for 6_3, for example, is the prototype that SnapPy identifies as 6_1. By SnapPy's symmetry groups, 14 of the 19 listed prototypes are chiral. Of these, 12 already fail a necessary condition for amphichirality (classical signature 0 and a Jones polynomial invariant under $t \mapsto t^{-1}$). The remaining chiral entries, written for 10_{79} and 10_{118}, are 10_{48} and 10_{104}, which meet that condition.

The list gives the following A1 items a wrong label. `A1_I__0032`, `A1_I__0040`, `A1_I__0046` and `A1_S__0034` pair a diagram of 6_1 with a diagram of its mirror image. Because its identifier is on the list, the sampler labeled them yes, but 6_1 is chiral. Every isometry that SnapPy finds between the complements of the two diagrams reverses orientation and their Jones polynomials differ, so the answer is no. `A1_I__0064` pairs a diagram of 10_{79} with a diagram of its mirror image. 10_{79} is amphichiral but missing from the list, so the sampler labeled the item no, whereas SnapPy finds an orientation-preserving isometry between the two complements and the answer is yes. We checked all 200 A1 items in the same way. With these corrections the label of every A1 item is yes exactly when its two diagrams are isotopic, and A1 scores change by at most 2 points (Table 18). The list misses other amphichiral knots up to ten crossings as well, but of those in the test split only 10_{79} appears in an A1 item.

A0 positives that pair a chiral knot with its mirror image. The A0 prompt asks whether the two diagrams are drawings of the same knot, "i.e., topologically equivalent". We built each A0 positive from two renders of one prototype. Because we walked every prototype both as given and as its mirror image, a positive can pair a diagram of a knot with a diagram of its mirror image. Read as ambient isotopy (Appendix A), topological equivalence makes a chiral knot and its mirror image different knots. Among the positives, 43 A0-I and 46 A0-S items pair a render of the prototype with a render of its mirror image. SnapPy's symmetry groups, together with the fact that torus knots are chiral, show the knot to be chiral in 41 and 43 of them, so under this reading the answer is no (Table 17 lists these items). The others are amphichiral knots and keep the label yes.

We score A0 with these relabeled items in the primary scenario, which leaves A0 unbalanced. A0-I has 59 positives and 141 negatives and A0-S has 57 and 143. Answering no on every item scores 70.5% on A0-I and 71.5% on A0-S. Table 13 also scores A0 under the convention the items were built with, in which a knot and its mirror image count as the same knot (row A0 as designed). The A1 prompt assumes that convention, since its premise that the two diagrams are "drawings of the same knot" also covers pairs in which one is the mirror image of the other. The A0 and A1 prompts, as issued, disagree about mirror images. We score A0 by the question its prompt asks, and A1 by whether its two diagrams are isotopic, which agrees with the A1 prompt on every item except the 16 discussed below.

The relabeling turns positives into negatives, so a configuration gains accuracy on each relabeled item it answers no and loses on each it answers yes. A0 accuracy rises for every configuration except GPT-5+T on A0-I, which answers yes on 34 of the 41 relabeled A0-I items and falls from 50.0 to 36.5%. Claude answers yes on 9 of them and rises from 52.5 to 64.0%. Under the primary labels no configuration scores above the always-no rate on either A0 task, and on each A0 task the configurations rank in the order of how often they answer no (Table 16; Claude and Claude+T tie on A0-S). Balanced accuracy, which does not reward a bias toward no, lies between 50.5 and 57.2%, so under the primary labels A0 accuracy mostly reflects how often a configuration answers no and says little about how well it judges equivalence.

A1 items that pair an amphichiral knot with its mirror image. The A1 prompt defines chirality through the diagrams: two diagrams have the same chirality "iff one is NOT the mirror image of the other". For an amphichiral knot the mirror image is isotopic to the knot itself. A diagram of such a knot paired with a diagram of its mirror image shows a knot and its mirror image, so the prompt's definition gives no. The pair also shows two diagrams of one knot that no comparison of handedness can tell apart, so any such comparison gives yes. The question has no single answer on these items, and we call them ill-posed. Of the A1 items, 16 are ill-posed, 6 in A1-I and 10 in A1-S (Table 17).

Table 16: A0 under the primary labels, 200 items per task. Answers yes: percentage of items on which the configuration answers yes. Balanced accuracy: mean of the accuracies (%) on positives and on negatives.

Task		Claude	Claude+T	GPT-5	GPT-5+T
A0-I	answers yes	19.5	17.0	55.5	84.0
	balanced accuracy	51.8	53.6	52.7	50.5
A0-S	answers yes	5.0	14.5	20.5	42.0
	balanced accuracy	51.4	57.2	56.5	55.9

Table 17: Items affected by the two conventions about mirror images, given by number (0003 in row A0-I is `A0_I__0003`). A0: the positives that pair a chiral knot with its mirror image, which the primary labels score as negatives. A1: the ill-posed items, which pair an amphichiral knot with its mirror image and which the primary labels keep as yes.

Task	Items
A0-I	0003, 0004, 0006, 0009, 0012, 0014, 0016, 0019, 0025, 0027, 0028, 0029, 0030, 0032, 0033, 0042, 0043, 0044, 0045, 0047, 0048, 0052, 0057, 0058, 0060, 0069, 0070, 0071, 0073, 0074, 0077, 0079, 0081, 0082, 0083, 0085, 0086, 0091, 0095, 0096, 0099
A0-S	0004, 0007, 0008, 0010, 0014, 0017, 0018, 0021, 0023, 0024, 0025, 0026, 0027, 0029, 0032, 0038, 0039, 0040, 0041, 0044, 0047, 0048, 0053, 0054, 0055, 0056, 0057, 0058, 0059, 0064, 0068, 0070, 0074, 0075, 0079, 0082, 0083, 0086, 0087, 0092, 0094, 0095, 0096
A1-I	0030, 0033, 0045, 0047, 0049, 0064
A1-S	0031, 0032, 0033, 0035, 0039, 0041, 0043, 0044, 0048, 0049

They show 8_3 (10 items), 4_1 (5) and 10_{79} (1). We keep them in the primary scores with the label yes, because their two diagrams are isotopic and the primary scores decide A0 by isotopy as well. Of the 16, 15 already had that label. The configurations answer yes on 5, 11, 8 and 11 of them.

If we remove these items or score them no, some A1 scores change by several points (Table 18). Claude on A1-S goes from 66.0 to 72.0% when we score them no, and no A1 score changes by more. The number of pairs at or below the uniform random baseline is 16 with the items removed and 16 with them scored no, the same count as under the primary scores. Claude's thinking lift over the 12 A–C tasks, most of which comes from A1-S (Appendix H), is +2.3 points (unadjusted exact McNemar $p = 0.004$) with the items kept, +1.9 ($p = 0.015$) with them removed and +1.5 ($p = 0.058$) with them scored no. Whether Claude's lift is significant at the 5% level therefore depends on how we score these items. GPT-5's lift is at least +7.3 points in all three cases, with McNemar p at most 4.6×10^{-7}.

Effect on the aggregate numbers. Table 18 gives the effect of each correction per task and Table 13 the aggregates over the 12 A–C tasks under each scenario. The B0 and A1 corrections alone (row A0 as designed) raise item-weighted accuracy for all four configurations, for Claude from 54.2 to 55.0% and for GPT-5+T from 54.3 to 55.1%. The A0 relabeling accounts for most of the difference between the original and the primary aggregates. It raises Claude from 55.0 to 58.8% and lowers GPT-5+T from 55.1 to 53.5%, which moves Claude above GPT-5+T. It also raises the best score on A0-I and on A0-S above 65%, so the number of tasks whose best configuration scores at or below 65% falls from 6 to 4. It lowers Claude's thinking lift from +3.3 to +2.3 points and GPT-5's from +10.5 to +7.8. Under the primary labels Claude's lift comes mostly from A1-S and that of GPT-5 from A3-S and B0-S (Appendix H); with A0 as designed, Claude also gains 16 items with thinking on A0-S. The number of (task, configuration) pairs at or below the uniform random baseline is 16 of 48 under the primary labels and 16 under the original labels.

Table 18: Accuracy (%) on the six tasks whose labels changed, under the original labels and under the primary labels. The last four rows remove the 16 ill-posed A1 items, which pair an amphichiral knot with its mirror image, or score them no. The primary labels keep them as yes.

Task	Labels	Claude	Claude+T	GPT-5	GPT-5+T
A0-I	original	52.5	55.0	47.5	50.0
	primary	64.0	66.5	50.0	36.5
A0-S	original	52.0	60.0	61.5	55.5
	primary	70.5	70.5	68.0	56.5
A1-I	original	46.0	47.0	48.0	49.0
	primary	46.0	47.0	46.0	49.0
A1-S	original	65.0	92.0	44.0	69.0
	primary	66.0	93.0	43.0	68.0
B0-I	original	29.5	32.5	17.0	21.0
	primary	30.5	30.5	18.5	23.0
B0-S	original	84.0	84.0	41.0	88.0
	primary	94.0	92.0	46.0	98.0
A1-I	items removed	45.7	46.8	42.6	46.8
	items scored no	46.0	47.0	40.0	45.0
A1-S	items removed	71.1	94.4	45.6	68.9
	items scored no	72.0	87.0	49.0	63.0

J.2 Constant and cue rules

We compare every task with three kinds of reference rule. A constant rule gives the most frequent label of the task. A text rule reads only the PD codes of an item, those of an -S item or those shown with a D item. An oracle rule needs a quantity read correctly off the images, such as the crossing counts of the two diagrams in a B0-I item. We call constant and text rules non-visual rules and sort them into two tiers. Tier 1 holds the constant rules and the text rules that follow from how we built the items, namely the A3-S list order and the B0-S crossing-count difference, from which we assigned the real-step labels. Tier 2 holds generic heuristics computed from the codes, same number of crossings and same sign of the writhe. We report oracle rules to show what the images make available, but we do not treat them as baselines, since reading that quantity is itself part of the task. Table 19 lists the rules with their scores under the primary labels next to the four configurations. We looked for the A–C rules after the evaluation, by reading the code that built the items and testing each candidate on the items, and cues we did not think of may remain. For the rebuilt D items we fixed a battery of text rules before running it on the final items, and the table gives its best score.

Label priors. On A0, C0 and C1 a constant answer scores well above the uniform random baseline. After the A0 relabeling, answering no is the best non-visual rule on both A0 tasks, and every configuration is at or below it. The same holds on C0, where each of the crossing counts 8, 11, 14, 17 and 20 labels 20 items, and on C1, where 31 labels are `bcda`, the DT code of the figure-eight knot. On B0-I, $R2^+$ and R3 tie as the most frequent labels with 36 items each, so a constant answer scores 18.0%, and all four configurations are above that, GPT-5 at 18.5%.

A3-S: list order. The A3-S positives pair a PD code whose crossing tuples are listed in ascending order with a copy whose arc labels are permuted and whose crossings are listed in random order. The negatives pair two codes whose tuples are in ascending order. A rule that answers yes exactly when a crossing list is out of ascending order scores 100.0%. DIAGRAM B's list is out of ascending order on 50 of the 50 positives and on 0 of the 50 negatives, and DIAGRAM A's list never is. The permuted labels also leave DIAGRAM B's arcs numbered out of sequence along the knot, which the consecutive numbering of Section 2 never produces. Claude and Claude+T answer no on every A3-S item and score the always-no rate of 50.0%. GPT-5 answers yes on 2 items and scores 48.0%. GPT-5+T scores 99.0% and agrees with the list-order rule on 99 of the 100 items. Its stored responses contain little beyond the answer line, so we cannot tell whether it compared the two diagrams or checked the order of the lists, and its A3-S score cannot show that it recognizes one diagram under relabeling. The cue cannot explain the other three scores, since Claude, Claude+T and GPT-5 answer

Table 19: Reference rules per task, scored under the primary labels (%), next to the four configurations. Kind: C, constant rule; T1 and T2, text rules of tier 1 and tier 2; O, oracle rule that needs a quantity read correctly off the images; for A2-S the rule is the task itself; B, the best of the 39 D0 and 43 D1 text and constant rules that we fixed in advance. By construction every text rule has an expected D score equal to the uniform random baseline, so we do not use the best of the battery as a baseline. The uniform random baseline is 50.0% for the A tasks, 16.7% for B0, 0% for C0 and C1, 50.0% for D0 and 25.0% for D1. D scores come from the September run.

Task	Rule	Kind	Score	Claude	Claude+T	GPT-5	GPT-5+T
A0-I	always no	C	70.5	64.0	66.5	50.0	36.5
	same crossing count	O	71.5				
	identical drawing	O	71.5				
A0-S	always no	C	71.5	70.5	70.5	68.0	56.5
	same crossing count	T2	61.5				
A1-I	always no	C	52.0	46.0	47.0	46.0	49.0
	same crossing count	O	39.0				
	same writhe sign	O	87.0				
	identical drawing	O	52.0				
A1-S	always no	C	51.0	66.0	93.0	43.0	68.0
	same crossing count	T2	49.0				
	same writhe sign	T2	79.0				
A2-I	either constant	C	50.0	67.0	65.0	45.0	50.0
	same crossing count	O	100.0				
	identical drawing	O	67.0				
A2-S	either constant	C	50.0	97.0	99.0	96.0	100.0
	same crossing count	task	100.0				
A3-I	either constant	C	50.0	90.5	90.5	63.0	74.5
	same crossing count	O	50.0				
	identical drawing	O	100.0				
A3-S	either constant	C	50.0	50.0	50.0	48.0	99.0
	list order	T1	100.0				
	same crossing count	T2	50.0				
B0-I	always $R2^{+}$ or R3	C	18.0	30.5	30.5	18.5	23.0
	count difference	O	95.0				
B0-S	always R3	C	20.0	94.0	92.0	46.0	98.0
	count difference	T1	96.0				
C0	any one count	C	20.0	9.0	14.0	8.0	11.0
C1	always `bcda`	C	31.0	0.0	0.0	0.0	0.0
D0	either constant	C	50.0	49.5	48.5	50.0	46.5
	best text rule	B	57.0				
D1	any one letter	C	25.0	24.5	27.0	27.0	22.0
	best text rule	B	29.5				

yes on none of the 50 positives. On A3-S they show no sign of recognizing one diagram under relabeling.

A3-I and A2-I: identical drawings. Every A3-I positive shows one diagram twice, with the same layout, in the two textures, and every negative shows two different diagrams. We drew the style parameters once per diagram, so all 427,027 pairs of texture variants in the corpus share color, rotation, scale, stroke and crossing gap. A rule that answers yes exactly when the two images show the same drawing scores 100.0% on A3-I. The rule needs the images only to detect a near duplicate, so A3-I accuracy (Claude 90.5%) cannot separate the operation that the A3 question names, recognizing one diagram under relabeling, from near-duplicate detection. The comparison of A3-I with A3-S is therefore not a modality contrast, because each variant has its own cue. The same rule scores 67.0% on A2-I, where some positives are also one drawing in two textures, and no configuration scores above it there (Claude 67.0%).

B0: crossing-count difference. We assigned every real-step label from the change in the number of crossings between D_t and D_{t+1}: $+1$ for R1$^+$, -1 for R1$^-$, $+2$ for R2$^+$, -2 for R2$^-$ and 0 for R3. The rule that applies this map, and answers NOT-CONNECTED for any other change, scores 96.0% on B0-S from the lengths of the two codes and 95.0% on B0-I if both counts are read correctly. It errs only on the NOT-CONNECTED items whose counts happen to differ by two or less (10 in B0-I, 4 in B0-S). Claude, Claude+T and GPT-5 score below the rule on B0-S and GPT-5+T scores above it, so B0-S accuracy cannot separate a configuration that simulates the candidate moves from one that compares the lengths of the two codes.

These NOT-CONNECTED items are the only B0 items on which counting is not enough, since on every other item the count change gives the label. Among them, 2 in B0-S and 4 in B0-I have equal counts, which leave a solver that counts a choice between R3 and NOT-CONNECTED. R3 dominates the items with equal counts, 36 of 40 in B0-I and 20 of 22 in B0-S. On these B0-S items the rule, which answers R3, is right on the R3 items, and the four configurations are right on 20, 17, 17 and 21. As built, B0 has no subset of useful size on which move simulation can be measured apart from counting and an answer prior. A version of B0 in which the NOT-CONNECTED pairs have the count changes of the real moves, in matched numbers, would provide one, since counting would then leave a choice between a real move and NOT-CONNECTED on every item.

On B0-I the four configurations score between 18.5 and 30.5%, against 95.0% for the rule with true counts. On C0 the configurations report the exact crossing count of a single diagram on 8.0 to 14.0% of items, which suggests that the counting route is not open to them from the images either. Low B0-I accuracy is consistent with a failure to read the diagrams and does not by itself bear on move simulation.

A1: writhe sign. Mirroring a diagram switches every crossing and negates its writhe, so the sign of the writhe tends to differ between a render of a chiral prototype and a render of its mirror image. The rule that answers yes exactly when the two PD codes have writhes of the same sign (zero counting as a sign of its own) scores 79.0% on A1-S. The rule does not test chirality, since an R1 move changes the writhe of a diagram without changing the knot. It is a partial computation on the codes, and only Claude+T (93.0%) scores above it. Applied to crossing signs read off the images, the same rule scores 87.0% on A1-I, where all four configurations are at or below the constant answer.

D0 and D1. The D items as first built admitted two cue rules. In D1, a rule that reads only the four option codes matched the answer key on all 200 items, and in D0 a rule based on the texture matched the label on 122 of 200 items. We retired those items for the reason given in Appendix C and designed the rebuilt items so that neither rule applies. Our specification for them asks for balanced D0 labels, a D0 texture chosen independently of the label, and D1 distractors that leave every text rule at the level of a uniform guess.

On the rebuilt items the specification holds. The labels are balanced (100 yes and 100 no in D0; each letter the key on 50 D1 items), so every constant answer scores the uniform random baseline. Texture is balanced within each D0 label, and the code shown with a D0 item always has the crossing count of the drawing, so the rule that compares the two counts scores 50.0%. We also fixed, before running it on the final items, a battery of 39 D0 and 43 D1 rules that give a constant answer or read only the shown codes or the style of the image. The rules of the second kind use the order of the tuples, the arc labels, the writhe, alternation, the number of tuples, sites for R1 and R2 moves, the crossings left after Regina's `simplify()`, the Jones polynomial, the D1 rule of the retired items, the style parameters, or a classifier trained by cross-validation on all text features or on the style parameters. The rules score between 40.5 and 57.0% on D0 and between 21.3 and 29.5% on D1. The exact 95% interval contains the baseline for every D1 rule and for all but 1 of the D0 rules. The exception is the classifier on all text features, whose interval lies below the baseline, not above it. Its score falls within the range the same procedure produces on shuffled labels, and with folds stratified by class the classifier's interval contains the baseline. The same battery, run on a design whose image always draws P, finds rules whose interval excludes the baseline on both tasks, which is why the image is drawn from the pair or the options (Appendix C).

Pairs at or below the best non-visual rule. A score at or below a non-visual rule contains no evidence of the operation beyond what the rule obtains without it, whether or not the configuration used the rule. Over the 48 (task, configuration) pairs of families A to C under the primary labels,

30 are at or below the best tier-1 rule of their task (29 strictly below), against 16 at or below the uniform random baseline. Adding the tier-2 rules gives 32. The tier-1 count is 25 under the original labels and 24 with only the B0 and A1 corrections (A0 as designed), so the rise comes from the A0 relabeling. On A0-I, A0-S, A1-I, A3-S, C0 and C1 all four configurations are at or below the tier-1 rule. On D0 and D1 the tier-1 rule is a constant answer at the uniform random baseline, and all four configurations on D0 and Claude and GPT-5+T on D1 are at or below it. Over all 14 tasks 36 of the 56 pairs are at or below the best tier-1 rule (34 strictly below).

Which tasks can show an operation on structure. A task can show that a configuration operates on structure only if the configuration scores above every rule that skips the operation. A3-I and A3-S fail this test, since the identical-drawing and list-order rules solve them, and so does A2-S, whose answer is the number of tuples. On A2-I no configuration scores above the identical-drawing rule, which needs the images only to detect a near duplicate. B0-S can show move simulation only on its 22 items with equal crossing counts, of which 20 are R3. A1-S can show it through the margin over the writhe-sign rule, and that margin rests on the ill-posed items (Appendix J.1). A0-S has no text rule above the constant answer, so a score above that answer would show an operation, but every configuration is at or below it. On A0-I, A1-I, B0-I and C0 a score also depends on reading the drawings, so a low score there does not show by itself which step failed. D0 and D1 ask for no move or equivalence, only a match between an image and a code, and every text rule has the uniform random baseline as its expected score, but a score at that baseline cannot separate a misread image from a failed comparison with the code. C1 as scored asks for recognition of the knot as well as transcription (Appendix J.3).

J.3 C1 LABELS AND THE DRAWN DIAGRAM

C1 shows one rendered diagram and asks for "the alphabetical Dowker-Thistlethwaite (DT) code for it", as "a string of lowercase letters of length n (where n is the number of crossings)". The label is the census DT code of the prototype's minimal diagram. The rendered diagram comes from a random walk, and on 94 of the 100 items it has more crossings than the label has letters. A model that follows the prompt and writes the DT code of the drawn diagram produces a string longer than the label on those items. Strict string matching can therefore credit a correct transcription only on the 6 items whose drawing has as many crossings as the label has letters. Our permissive tier decodes the answer and compares Regina's canonical signature of the decoded diagram, without simplifying it, with the stored signature of the minimal diagram, so it has the same limit. It also lower-cases the answer before decoding, which loses the item among these whose label contains upper-case letters (Appendix F). Strict matching has a second limit, because a diagram has several valid DT strings and our scorer accepts only the stored one. On `C1__0017`, drawn with 4 crossings, GPT-5+T answered `cdab`, a valid DT string of the drawn diagram, whose label is `bcda`. Strict matching marks the answer wrong, and the permissive tier credits it.

The other permissive credits are the same answer from GPT-5+T on figure-eight items drawn with more crossings (Appendix F), where it is the code of the knot's minimal diagram and not a transcription of the drawing. C1 as scored therefore asks for more than transcription. A model must recognize the knot in a drawing that is usually not minimal and write the census code of its minimal diagram. The strict score of 0.0% for all four configurations is a result about that combined task. It does not show on its own that the configurations cannot transcribe the drawn diagram, which C1 can test on at most 6 items.

K MODELS, REQUEST PARAMETERS AND OUTPUT BUDGETS

We queried Claude Opus 4.7 (Anthropic, 2026) and GPT-5 (OpenAI, 2025) through the vendors' APIs, each in two configurations (Table 20). The May run, on 9 and 10 May 2026, covered the A–C items and the original D items, which we later retired (Appendix C). The September run covered the rebuilt D0 and D1 items in one session on 24 September 2026 (UTC), with a 64,000-token cap for both thinking configurations from the first request. Both runs sent the model identifiers `claude-opus-4-7` and `gpt-5`, and `gpt-5` is an alias that the vendor resolves to a dated snapshot. We did not store the model identifier returned with each May response, so we do not know which snapshots produced the May rows. The September harness stores it for every row. All 400

Table 20: Request parameters of the four configurations. A cap is the output-token limit of a request (`max_tokens` for Claude, `max_completion_tokens` for GPT-5) and includes thinking or reasoning tokens. Claude+T uses adaptive thinking at effort `high`. Both first-run GPT-5 caps are 2,048 tokens for the answer plus 4,096 for reasoning. May values apply to the A–C items and September values to the rebuilt D0 and D1. Concurrency is the largest number of simultaneous requests for one configuration. September values are those recorded in the run metadata and in every stored row.

	Claude	Claude+T	GPT-5	GPT-5+T
Model identifier sent	`claude-opus-4-7`		`gpt-5`	
Identifier returned, September	`claude-opus-4-7`		`gpt-5-2025-08-07`	
Thinking or reasoning	none	adaptive	minimal	high
Cap, May first run	2,048	16,000	6,144	6,144
Cap, May re-queries	–	32,000, 64,000	–	64,000
Cap, September	2,048	64,000	6,144	64,000
Streaming, May	no	re-queries	no	no
Streaming, September	no	yes	yes	yes
Concurrency, May	8	8	24	16
Concurrency, September	8	8	8	32
Client timeout, May	600 s, two automatic retries (SDK defaults, not recorded)			
Client timeout, September	3,600 s, SDK retries off, at most 2 logged retries			

September rows of each Claude configuration report `claude-opus-4-7`, and all 400 rows of each GPT-5 configuration report `gpt-5-2025-08-07`. The four months between the runs leave open whether the May rows came from the same snapshots.

We store one response per item and configuration (the re-queries are described below). We set no temperature, top-p or seed for either model, so each vendor's default sampling applies; Claude Opus 4.7 does not accept sampling parameters. We sent each image as a base64-encoded PNG file inside the user message, with the fixed system prompt of Appendix E. The two configurations of a vendor received identical inputs, and their input-token counts agree on every A–C item and every rebuilt D item.

Thinking settings. Claude+T requests adaptive thinking with effort `high`. The Claude configuration omits the thinking parameter, and Opus 4.7 then answers without thinking. An adaptive-thinking request sets no thinking budget, and the number of thinking tokens varies between requests, from none upward. GPT-5 has no setting that turns reasoning off: the GPT-5 configuration uses `reasoning_effort minimal` and GPT-5+T uses `high`. Both APIs bill thinking and reasoning tokens as output and count them against the output cap. We have no thinking or reasoning text. GPT-5 does not return it, and Opus 4.7 leaves the thinking text empty unless a summary is requested, which neither run did. The 64,000-token cap is our choice and lies below the 128,000-token output limit of Claude Opus 4.7.

The May run and its re-queries. In the first run we sent every item to the four configurations with the first-run caps of Table 20. The two vendors ran in parallel, and the two configurations of one vendor ran one after the other. In this run every thinking response that reached its cap came back without visible text. Across all items of the May run we therefore re-queried each thinking response that was empty or had no parseable answer, that is, no answer the task's rule can read. On C1 any non-empty answer string counts as parseable, including refusals such as `unknown`. For Claude+T these were 47 responses, sent first with a 32,000-token cap, which recovered 33, and then, for the 14 still empty, with 64,000, which recovered 10. For GPT-5+T they were 793 responses, sent with a 64,000-token cap, of which 769 returned a parseable answer. We had stopped an earlier GPT-5+T pass with a 96,000-token cap before it finished and did not use its responses. A re-query replaced the first-run response only if it contained visible text with a parseable answer, and otherwise we kept the first-run response. We did not re-query the non-thinking configurations, and none of their A–C responses is empty or unparseable. Table 21 gives the source and cap of every stored response; its last block counts the 400 D responses of each configuration in the September run, all at the

September cap. Most stored rows that come from a re-query fall in A0-S, A1-S and A3-S (Table 23). For GPT-5+T they make up 184 of the 200 A0-S rows, 93 of the 100 A1-S rows and 79 of the 100 A3-S rows, so its scores on these tasks come mostly from the 64,000-token re-queries.

Table 21: Stored responses per configuration, by the request that produced them and its output-token cap (Table 20). A–C rows count the 1,600 items of the 12 A–C tasks. All rows count the 2,000 items of the May run, including the original D0 and D1 items that we withdrew (Appendix C). In the May run we re-queried every item whose first Claude+T or GPT-5+T response was empty or unparseable, 47 for Claude+T and 793 for GPT-5+T. The Claude+T re-query at the smaller of its two caps gave a usable response on 33 items, and we sent the other 14 again at the larger cap. When the re-query gave no usable response either, we kept the first-run row ("first run, kept"). These kept rows are the 23 empty responses and the 5 responses whose ANSWER line has nothing after it. For 16 of the empty responses, all from GPT-5+T, the re-query ended in a client timeout.

Configuration	Stored row from	Output cap	A–C rows	All rows
Claude	first run	2,048	1,600	2,000
Claude+T	first run	16,000	1,553	1,953
	first run, kept	16,000	4	4
	re-query	32,000	33	33
	re-query	64,000	10	10
GPT-5	first run	6,144	1,600	2,000
GPT-5+T	first run	6,144	1,075	1,207
	first run, kept	6,144	24	24
	re-query	64,000	501	769
D0 and D1 (September 2026 run)				
Claude	D run	2,048	400	–
Claude+T	D run	64,000	400	–
GPT-5	D run	6,144	400	–
GPT-5+T	D run	64,000	400	–

Selection effect of the re-query policy. An item keeps its first-run response when that response gave a parseable answer within the first-run cap. Otherwise we store the re-query, which may itself be short, or keep the first-run response if the re-query also failed (Table 22). Compared with a single run at 64,000 tokens, the stored GPT-5+T rows therefore over-represent short reasoning. If a response to some item runs past 6,144 tokens with probability q, a single run stores a long response for that item with probability q and our policy with probability q^2. Of the 501 GPT-5+T re-query rows among the A–C items, 94 finished within 6,144 tokens. These short re-queries are correct on 41.5% of their items, against 65.8% for the 407 longer ones. If the short ones had scored like the long ones, GPT-5+T would gain 1.4 points on the A–C tasks (0.8 points by the same calculation on all items of the May run under the original labels). This pooled estimate reflects which tasks the short re-queries fall in. They cluster in C1, C0, B0-S and the image variants of the A and B tasks. When we compare short and long re-queries within each task, the estimate is -0.07 points, and 17 short re-queries lie in tasks that have no long re-query to compare against. On these data we cannot tell whether the over-representation of short reasoning lowered the GPT-5+T score. For Claude+T a short re-query is one that finished within the 16,000-token first-run cap. Its 43 re-query rows among the A–C items all fall in A0-S and A1-S, and its within-task estimate of -0.08 points rests on too few rows to be informative.

Client timeouts. In May we left the client at the SDK defaults, a 600 s timeout with up to two automatic retries, and did not record the retries. For a streamed request the timeout bounds only the wait between chunks, so a streamed request can run past 600 s. The longest Claude+T request, a streamed re-query, took 622.0 s. The first-run Claude+T requests were not streamed and took at most 220.2 s, so none of them reached the timeout. For a non-streamed GPT-5+T request the timeout bounds the wait for the whole response, so the client abandoned any request that had not returned within 600 s and sent it again. Of the 793 GPT-5+T re-queries, 16 (11 on A0-S and 5 on A1-S) timed out on all three attempts, and for these items the empty first-run response is stored and scored wrong. Of all GPT-5+T responses stored from the May run, including those to the retired D items, 156 took longer than 600 s and were therefore answered on a second or third attempt. Of these, 24 took longer

Table 22: A–C responses without an answer, by cause. We score all of them wrong. In every case the stored row is the first-run response.

Configuration	Task	Rows	Cause
Claude+T	A0-S	3	Empty at the 16,000-token cap and again after the 32,000 and 64,000 re-queries.
Claude+T	A1-S	1	As above.
GPT-5+T	A0-S, A1-S	16	Empty at the 6,144-token cap; the 64,000 re-query timed out on all three attempts (11 A0-S, 5 A1-S).
GPT-5+T	A0-S	1	Empty at the 6,144-token cap; the re-query used all 64,000 output tokens without visible text.
GPT-5+T	C1	2	Empty at the 6,144-token cap; the re-query ended in an empty `ANSWER:` line, so the empty response was kept.
GPT-5+T	C1	5	First run and re-query both ended in an empty `ANSWER:` line; the first-run responses used 2,251 to 5,323 output tokens.

than 1,200 s, and the longest took 1790.5 s. For GPT-5+T, response length was limited by the 600 s timeout as well as by the token cap. Had the timed-out items been answered at the accuracy of the items answered by a re-query in their task, GPT-5+T would gain 0.6 points, reaching 54.1% on the A–C tasks.

Causes of responses without an answer. Table 22 traces each of the 28 A–C rows without an answer to its cause. The stored token counts of these rows are those of the first run. By the cost-ledger arithmetic of our harness audit, each of the 4 empty Claude+T rows used the full cap on its 64,000-token re-query; the stored rows record only the first-run 16,000 tokens.

The September run. For the rebuilt D items we used the same four configurations and the same answer-line format. In September we set an explicit 3,600 s read timeout and turned the SDKs' own retries off. The harness retries a request at most twice, only after a timeout, a transport error, a rate limit or a server error and never after a completed response, and it logs every attempt. GPT-5 requests stream when the vendor allows it and otherwise fall back to non-streamed requests. Every stored row records the returned model identifier, the request identifier, the stop reason, the token counts including reasoning tokens where the API reports them, the elapsed time, the number of attempts and the cap. All GPT-5 requests of the run were streamed. No stored D response is empty, stopped at its cap or without a parseable answer, and none needed a re-query. Of the GPT-5+T requests, 3, all to D1 items, ended in an HTTP 504 gateway time-out returned by the vendor's endpoint and were answered on the second attempt; every other request was answered on its first. The longest answered GPT-5+T attempt took 866.7 s and the longest Claude+T request 162.3 s. Each failed attempt ran longer than any answered one before the endpoint returned the 504, and no attempt reached the client timeout. The 504 thus marks a time limit of the vendor's gateway that lies below our client timeout, so a GPT-5+T response that would have taken longer than this limit could not be observed.

Output tokens and thinking engagement. Table 23 gives output-token statistics per task and Table 24 per configuration. Claude+T produced 50 output tokens or fewer on 1,189 of its 1,600 A–C responses, and on 994 of the 1,000 responses to the seven image-only A–C tasks. The median Claude response without thinking has 12.0 output tokens, and we read a Claude+T response of at most 50 output tokens as one with little or no thinking. Claude+T responses with more than 50 output tokens come almost entirely from A0-S, A1-S and B0-S (Table 23). On the Claude+T responses of 50 tokens or fewer, the parsed answer equals Claude's on 951 of 1,189 items. Of Claude's thinking lift of 36 items, 11 come from those items (McNemar $p = 0.266$) and 25 from the others ($p = 0.003$). GPT-5+T produced at least 332 output tokens on every A–C response, while GPT-5 at minimal effort stayed at 50 tokens or fewer on 1,498 of 1,600. Thinking multiplies total output tokens by 23.5 for Claude and by 410.2 for GPT-5 on the A–C items.

Table 23: Output tokens per task for the thinking configurations, and stored rows that come from a re-query. "At most 50" counts Claude+T responses with 50 output tokens or fewer. Medians include reasoning tokens. The A–C rows come from the May run and the D rows from the September run, which had no re-queries.

		Claude+T		GPT-5+T	Rows from a re-query	
Task	Items	At most 50	Median	Median	Claude+T	GPT-5+T
A0-I	200	200	11.0	3,532	0	10
A0-S	200	3	255.5	18,924	26	184
A1-I	100	100	11.0	4,396	0	13
A1-S	100	0	3,153	16,844	17	93
A2-I	100	100	11.0	3,980	0	0
A2-S	100	99	12.0	1,612	0	0
A3-I	200	200	12.0	3,340	0	2
A3-S	100	91	11.0	9,036	0	79
B0-I	200	199	13.0	3,373	0	5
B0-S	100	2	83.0	4461.5	0	37
C0	100	99	12.0	4,589	0	17
C1	100	96	17.5	5,773	0	61
All A–C	1,600	1,189	12.0	4,365	43	501
D0	200	53	411.5	10,892	–	–
D1	200	113	11.0	21,612	–	–

Table 24: Tokens and cost of the stored responses to the 1,600 A–C items of the May run (top) and to the 400 D items of the September run (bottom). Token counts are as reported by the APIs, with thinking and reasoning tokens counted as output. Prices are list prices in US dollars per million tokens, the same for both runs, and costs are in US dollars. Cost per correct answer uses the corrected labels. Neither cost includes requests whose response was not stored (Appendix K).

	Claude	Claude+T	GPT-5	GPT-5+T
Input tokens, total	1,302,436	1,302,436	843,536	843,536
Output tokens, total	62,202	1,459,597	28,003	11,485,817
Output tokens, median	12.0	12.0	12.0	4,365
Output tokens, largest	1,384	51,118	227	44,940
Price, input tokens	5	5	1.25	1.25
Price, output tokens	25	25	10	10
Cost of the stored responses	8.1	43.0	1.3	115.9
Cost per correct answer	0.0086	0.0441	0.0018	0.1354
D0 and D1, September run				
Input tokens, total	464,324	464,324	350,524	350,524
Output tokens, total	6,743	359,956	4,953	6,603,362
Output tokens, largest	340	14,854	43	38,860
Cost of the stored responses	2.49	11.32	0.49	66.47

On the D items Claude+T produced 50 output tokens or fewer on 166 of its 400 responses (53 on D0 and 113 on D1). On D0, which pairs the image with a code, it thus stayed at 50 tokens or fewer on 26.5% of responses, against at least 96.0% on each image-only task of the May run. GPT-5+T produced 6,603,362 output tokens on the D items, 6,598,528 of them reasoning tokens, against 4,953 for GPT-5, and its accuracy on D0 and D1 is no higher (Table 12).

Cost. At list prices, with reasoning tokens billed as output, the stored A–C responses cost 168.32 US dollars (Table 24), and all stored responses of the May run, including those to the retired D items, 204.96 dollars. This A–C cost leaves out the responses to the retired D items, the first-run responses that re-queries replaced, the failed re-queries, the stopped 96,000-token pass, and the requests that

the OpenAI SDK re-sent after timeouts, which our harness did not record. The logged requests of the May run, including a pilot, cost 293.4 dollars at list prices. Adding 13.3 dollars for the stopped pass and an estimated 78.6 to 146.6 dollars for the unlogged retries gives 385 to 453 dollars for the May run. This estimate does not count the Claude+T requests of a first re-query attempt that we stopped early and whose cost we did not record.

The stored responses of the September run cost 80.77 dollars at list prices, 66.47 of them for the 6.6 million output tokens of GPT-5+T. The 3 GPT-5+T attempts that ended in a gateway time-out reported no usage, so any charge for them is not included, and this cost is a lower bound as well.

L QUALITATIVE EXAMPLES

We quote five stored responses of the May run, in four examples. Neither model returned the text of its thinking (Appendix K), so what we can read is the visible response, and on the image tasks that is usually the answer line alone. Of the 1,000 responses of each configuration to the seven image-only tasks, Claude wrote text beyond the answer line in 1 and Claude+T in 0. GPT-5 and GPT-5+T did so in 114 and 49, of which 99 and 45 were responses to C1. We chose the examples by hand, after reading the responses that contain text beyond the answer, to show the route that a response states, which a score does not reveal. They are not a random sample. With each example we give a count, over all items of its task, of how often its pattern occurs, and we claim nothing about how typical it is beyond that count. We quote every response in full, with the model's own formatting marks.

Counting crossings in B0-S. Item `B0_S__0006` has the label $R1^{+}$, so DIAGRAM_T1 has one crossing more than DIAGRAM_T.

B0_S__0006, Claude, answer NOT-CONNECTED (wrong)

```
DIAGRAM_T has 13 crossings, DIAGRAM_T1 has 13 crossings. Same count, so
it's R3 or NOT-CONNECTED.

Looking at T1: crossing [22, 22, 23, 21] has arc 22 appearing twice as
the under-strand entries — this is a self-loop, indicating a kink
(R1-style crossing).

T has 13 crossings with no obvious self-loop. T1 has 13 crossings with a
self-loop at [22,22,23,21].

Same crossing count rules out R1±, R2±. The presence of a new self-loop
with no removal elsewhere isn't an R3.

ANSWER: NOT-CONNECTED
```

B0_S__0006, Claude+T, answer $R1^{+}$ (correct)

```
Looking at the crossing counts:
- DIAGRAM_T: 13 crossings
- DIAGRAM_T1: 14 crossings (one more)

DIAGRAM_T1 contains crossing [22, 22, 23, 21], which has a repeated arc
label (22) — a self-loop / kink. The rest of the diagram corresponds
(with relabeled arcs) to DIAGRAM_T.

This is the addition of one self-loop (one new crossing).

ANSWER: R1+
```

Both responses start from the number of crossings in each code, the quantity from which we assigned the real-step labels (Appendix J.2) and by which the B0 prompt glosses each move (Appendix E). Claude's response gives the same count for both codes and rules out $R1^{\pm}$ on the counts, although it names the new kink. Claude+T's response gives both counts correctly and names the kink. Claude's text states the crossing count of each code, separately or as one count for both, in 95 of its 100 B0-S responses, and a stated count is wrong in 2 of them. They are this item and `B0_S__0094`, on which Claude answers $R2^{+}$ from a difference of two although the label is NOT-CONNECTED. Both responses state the route of the count-difference rule; neither shows whether its configuration also simulates the move.

Comparing writhes in A1-S. `A1_S__0000` gives two codes of a chiral knot with the same chirality, and its label is yes.

A1_S__0000, Claude+T, answer yes (correct)

```
Looking at the writhes:

**Diagram A:** Computing crossing signs (positive if d-b≡1 mod 26,
negative if b-d≡1):
- Positive crossings: 5, Negative crossings: 8
- Writhe = -3

**Diagram B:**
- Positive crossings: 4, Negative crossings: 9
- Writhe = -5

Both diagrams have negative writhe. Diagram B contains a Reidemeister-I
kink at crossing [13,13,14,12] (a negative kink), which accounts for
some of the writhe difference. The overall handedness — indicated by
the consistent sign of writhe across both presentations — is the same.

ANSWER: yes
```

The response gives the signs of the two writhes as the reason for its answer, which is what the same-writhe-sign rule of Appendix J.2 computes. Its numbers of positive and negative crossings are those of Regina's convention with the two signs exchanged, which leaves the comparison unchanged. The writhe is not an invariant of the knot, since an R1 move changes it by one, and the response itself attributes part of the difference to a kink. Of the 100 A1-S responses of Claude+T, 95 mention the writhe, and its answers agree with the rule on 82 items. The rule is wrong on 21 items, 8 of them ill-posed (Appendix J.1). Claude+T answers 16 of the 21 correctly, including 10 of the 13 that are not ill-posed, so its answers do not come from the sign of the raw writhe alone. A rule that also discounts kinks, as this response does, is one we did not score. Claude+T scores 93.0% on A1-S against 79.0% for the rule under the primary labels, and 87.0 against 85.0% if the ill-posed items are labeled no.

Counting before writing a DT code. `C1__0000` shows a diagram of the figure-eight knot. Its label is `bcda`, the census DT code of the knot's prototype.

C1__0000, GPT-5+T, answer abcde (wrong)

```
Based on the diagram, I count five crossings and compute the
alphabetical DT code accordingly.

ANSWER: abcde
```

The count in the response is right, and the answer has one letter per counted crossing, as the prompt asks. Apart from empty answers and words such as `unknown`, GPT-5+T answers with a string of letters on 58 of the 100 C1 items. Of these strings, 11 have one letter per crossing of the drawing. The drawing has more crossings than the label has letters, as on 94 of the 100 C1 items, so no transcription of this drawing could have matched the label (Appendix J.3). The answer is not a transcription either, since Regina decodes `abcde` to a diagram of the unknot in which every crossing is a kink. The visible text is one sentence, although the stored row, from a re-query with a cap of 64,000 tokens, reports 8,606 output tokens.

A refusal on C1. `C1__0008` shows another diagram of the figure-eight knot, with as many crossings as the drawing of `C1__0000`.

C1__0008, GPT-5+T, answer unknown (wrong)

```
I can't determine a unique DT code from this image because the crossings
don't indicate which strand goes over or under at each intersection. The
DT code depends on that information. If you can provide a version with
over/under markers (e.g., small gaps at underpasses) or a PD/Gauss code,
I can compute it precisely.

If you just want the unsigned alphabetical DT (ignoring crossing signs),
here's how to generate it from a clearly drawn oriented diagram:
- Number the visits to crossings along the oriented knot 1,2,...,2n.
- For each odd label 1,3,5,..., record the even label paired at that
crossing; this gives the unsigned numeric DT list.
- Convert even numbers 2k to letters: a=2, b=4, c=6, etc. Example:
[4,6,2] -> bca.

Please provide a clarified diagram or the numeric DT list and I'll
return the alphabetical DT string.

ANSWER: unknown
```

The response says that the crossings do not show which strand passes over, and it asks for small gaps at underpasses. The image draws the lower strand with a gap at every crossing (Section 2). In place of a code, GPT-5+T answers with a word on 35 of the 100 C1 items, 25 times `unknown`. GPT-5 does so on 99, 82 times `unknown`. We did not code the reasons these responses give. A response that asks for information the image contains is consistent with a failure to read from the gaps which strand passes over at each crossing. C0 asks only for the number of crossings and does not test this reading.

Table 25: Accuracy of three reference rules and of the four configurations. The rules are a uniform guess, the most frequent label of each task, and the best tier-1 rule of each task. The last is that label except where a rule that reads only the PD codes scores higher (list order on A3-S and count difference on B0-S, Appendix J.2). Claude and Claude+T are Claude Opus 4.7 with thinking off and with adaptive thinking at high effort. GPT-5 and GPT-5+T are the GPT-5 model with reasoning effort set to minimal and to high. Brackets give 95% intervals. Accuracy is item-weighted, with a Wilson interval. Macro is the unweighted mean of the per-task accuracies, with a bootstrap interval (items resampled within each task). Lift is the item-weighted accuracy of Claude+T minus that of Claude, and of GPT-5+T minus that of GPT-5, in percentage points, with a paired bootstrap interval. The p value is from an exact two-sided McNemar test on the items that exactly one of the two configurations answers correctly. Claude+T ran with a larger output-token cap than Claude, and we re-queried the empty or unparseable first responses of both thinking configurations with larger caps (Table 21), so each lift includes these differences. We score a response correct when its last ANSWER line, or its last non-empty line if it has none, matches the label after normalization (Appendix F). Empty counts the responses with no text. We score these as wrong, as we do the 5 GPT-5+T responses to C1 whose ANSWER line has nothing after it. In the September run of D0 and D1 both thinking configurations ran at the largest cap of the May re-queries from the start (Table 20), and every response has a parsed answer. Labels and scoring follow Table 1, Table 12 gives the per-task scores, and Table 13 gives these numbers under the original labels and other scoring conventions.

Configuration	Accuracy (%)	Macro (%)	Lift (points)	p	Empty
12 A–C tasks, 1,600 items per configuration					
Uniform guess	37.5	36.1			
Most frequent label	46.5	44.5			
Best tier-1 rule	54.4	55.0			
Claude	58.8 [56.3, 61.1]	57.0 [55.0, 59.0]			0
Claude+T	61.0 [58.6, 63.4]	59.8 [58.0, 61.8]	+2.3 [0.8, 3.7]	0.004	4
GPT-5	45.7 [43.3, 48.1]	44.3 [42.1, 46.5]			0
GPT-5+T	53.5 [51.1, 55.9]	55.5 [53.6, 57.3]	+7.8 [5.3, 10.4]	4.8×10^{-8}	19
All 14 tasks, 2,000 items per configuration					
Uniform guess	37.5	36.3			
Most frequent label	44.7	43.5			
Best tier-1 rule	51.0	52.5			
Claude	54.4 [52.2, 56.6]	54.2 [52.4, 56.0]			0
Claude+T	56.4 [54.2, 58.5]	56.7 [54.9, 58.4]	+2.0 [0.3, 3.6]	0.025	4
GPT-5	44.3 [42.1, 46.4]	43.5 [41.5, 45.4]			0
GPT-5+T	49.7 [47.5, 51.8]	52.4 [50.7, 54.1]	+5.4 [3.0, 7.8]	3.2×10^{-5}	19

M EXTENDED RESULTS

Table 25 gives accuracy and thinking lifts pooled over all 14 tasks and over the 12 A–C tasks of the May run.

The two models. Claude scores above GPT-5 and Claude+T above GPT-5+T over all items (Table 25), but the two models ran under different output caps (Section 4.2), and when we score A0 as designed, counting a knot and its mirror image as the same knot, Claude+T scores 54.2% [52.0, 56.3] and GPT-5+T 50.9% [48.7, 53.1], with overlapping Wilson intervals. We therefore do not rank the two models.

Original labels over the 14 tasks. Table 13 scores the 12 A–C tasks under six labeling and scoring scenarios; the D items are the same in all of them. Over the 14 tasks, the original labels give accuracies of 50.8, 53.9, 43.1 and 50.3% for Claude, Claude+T, GPT-5 and GPT-5+T, 22 pairs at or below the uniform random baseline, 8 tasks on which no configuration scores above 65%, and thinking lifts of 3.2 points for Claude [1.5, 4.8] and 7.2 for GPT-5 [4.8, 9.6], with 45 of Claude's 63 net items and 115 of GPT-5's 143 on -S tasks. Scoring A0 as designed with the corrected labels

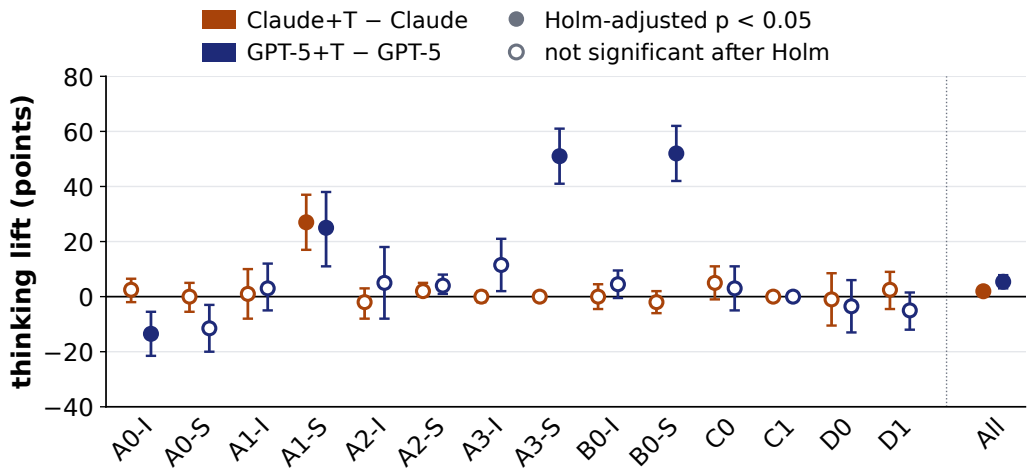


Figure 11: Thinking lift per task: accuracy of the thinking configuration minus accuracy of the same model without thinking on the same items, in percentage points (Claude+T − Claude; GPT-5+T − GPT-5), under the primary scoring. Error bars are 95% paired percentile bootstrap intervals (10,000 resamples of the task's items). Filled markers: two-sided exact McNemar test on the items that change, significant at 0.05 after Holm correction over the 14 tasks within each model; hollow markers: not significant. All: item-weighted lift over all 2,000 items, with a task-stratified bootstrap interval and an unadjusted exact McNemar test: Claude 2.0 [0.3, 3.6], GPT-5 5.4 [3.0, 7.8] points. Claude's lift comes from A1-S: without it the lift is 0.6 points ($p = 0.488$). GPT-5's comes from A3-S and B0-S, which a text rule solves: without them it is 0.3 points ($p = 0.866$). GPT-5+T's scores on A0-S and A1-S include 12 and 5 responses without a parseable answer, scored wrong.

leaves 22 pairs at or below the uniform random baseline and gives 8 tasks on which no configuration scores above 65%, adding A0-I and A0-S to the 6 of the primary scoring.

The ill-posed A1 items over the 14 tasks. Scoring the 16 A1 items that pair an amphichiral knot with its mirror image as no, as the prompt's definition of chirality gives, lowers Claude's thinking lift over the 14 tasks to 1.4 points [−0.3, 3.0] (exact McNemar $p = 0.125$), and removing them gives 1.7 [0.1, 3.3] ($p = 0.056$). Whether Claude's lift over the 14 tasks is significant at the 5% level therefore depends on how we score these items, as over the 12 A–C tasks (Appendix J.1). GPT-5's lift is 5.0 [2.6, 7.4] and 5.3 [2.9, 7.7] points in the two cases.

Image versus PD-code input per pair. Figure 2 shows the two variants of each task pair. The pooled PD-code advantage is 14.3 [9.4, 19.1] points for Claude, 18.3 [13.5, 22.9] for Claude+T, 17.3 [12.0, 22.4] for GPT-5 and 33.8 [28.9, 38.4] for GPT-5+T (Newcombe 95% intervals). These pools weight the five tasks differently, since B0 contributes 200 of the 800 -I items and 100 of the 600 -S items. The unweighted means of the five per-pair advantages are 15.9, 21.0, 15.7 and 37.7 points. The PD-code advantage of a single pair has a Newcombe interval above zero for every configuration on A2 and B0 and for all but GPT-5 on A1. On A0 only the intervals of GPT-5 and GPT-5+T lie above zero, and A0 accuracy mostly reflects how often a configuration answers no. Claude and Claude+T score 90.5% on A3-I, where the identical-drawing rule scores 100.0% given the true layouts, and 50.0% on A3-S, where they answer no on every item. GPT-5 answers yes on 2 A3-S items, none of them a positive. These three configurations thus reject every relabeled copy on A3-S, which the list-order cue would accept, while on A3-I every positive shows one layout twice. The A3 contrast therefore cannot separate the input format from the two cues (Appendix J.2).

Thinking on the image-only tasks. Claude+T wrote 50 output tokens or fewer on 994 of its 1,000 responses to the seven image-only A–C tasks, the five -I variants, C0 and C1 (Appendix K). Its scores on these tasks therefore say little about whether thinking helps on images. On the D items it wrote 50 output tokens or fewer on 53 of its 200 D0 responses and 113 of its 200 D1 responses. GPT-5+T wrote at least 1,292 output tokens on each of its responses to the seven image-only A–C

tasks, and its lift over GPT-5 on them is 1.6 points [−1.6, 4.8], with exact McNemar $p = 0.370$. When we score A0 as designed, this lift rises to 4.8 points [1.5, 8.0], with $p = 0.005$. Thinking changes pooled accuracy on the five -I variants by 0.5 points for Claude and 1.6 for GPT-5, with exact McNemar $p = 0.720$ and 0.459.

The 12 A–C tasks. The 14-task numbers of the main text pool the May run of the A–C items with the September run of the rebuilt D items (Section 4.2). Restricted to the 12 A–C tasks, 16 of the 48 (task, configuration) pairs are at or below the uniform random baseline and 30 at or below the best tier-1 rule of their task (Appendix J.2), and on 4 of the 12 tasks no configuration scores above 65%. The thinking lifts are 2.3 points for Claude [0.8, 3.7] and 7.8 for GPT-5 [5.3, 10.4] (Table 25). Without A1-S the lift for Claude is 0.6 points ($p = 0.452$), and without A3-S and B0-S the lift for GPT-5 is 1.6 points ($p = 0.303$). Holm correction over the 12 tasks marks the same tasks as significant as over 14, and gives GPT-5's drop on A0-I $p = 0.016$.