Knot theory in R

Robin K. S. Hankin
Auckland University of Technology

Abstract

In this short article I introduce the **knot** package, which creates two dimensional knot diagrams optimized for visual appearance.

The **knot** package is a systematic R-centric suite of software for the creation of production-quality artwork of knot diagrams.

Keywords: Knot theory, R.

1. Introduction

A *mathematical knot* is a smooth, unoriented embedding of a circle $\mathbb{S}^1$ into $\mathbb{R}^3$ (Manturov 2004). Two knots are said to be equivalent if one can be continuously deformed in to the other; if so, there is a homeomorphism $h\colon \mathbb{R}^3 \longrightarrow \mathbb{R}^3$ which takes one embedded circle to the other.

It is common to present a knot using diagrams such as Figure 1, in which a two-dimensional projection of the knot is given with broken lines indicating where one strand passes over another.

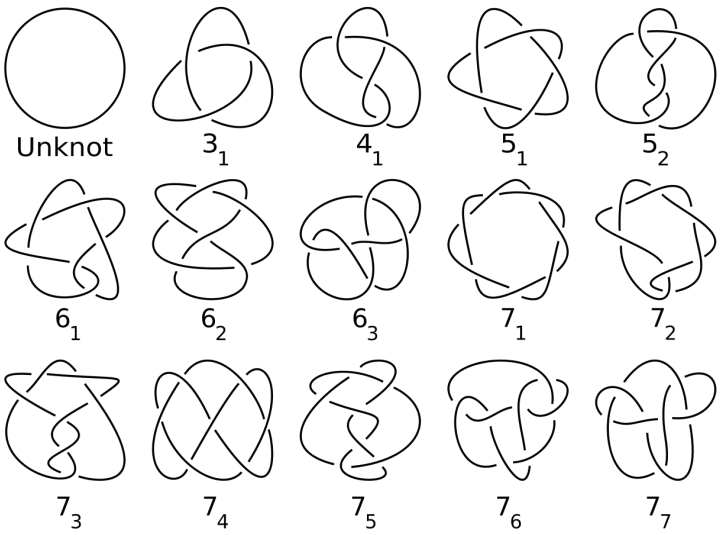

Figure 1: A table of prime knots up to seven crossings, labels following Alexander and Briggs (1926). Image taken from Wikipedia (2016).

Consider Figure 1 from an æsthetic perspective; the diagrams are representative of those available under a free license. However, these diagrams are not suitable for high-quality artwork such as posters: they are not vectorized. Many of the underlying knots possess a line of symmetry (at least, the diagrams do if the breaks are ignored), which is not present in the visual representation. Also, several of the strands cross at acute angles. The diagram for knot 7_3, for example, contains ugly kinks and abrupt changes of curvature.

Such considerations suggest that knot diagrams might be produced by minimization of some objective function that quantifies the visual inelegance of a knot diagram. The precise nature of such an objective function is, of course, a subjective choice but one might require the following desiderata:

- Curvature to be as uniform as possible
- Strands to cross at right angles
- Crossing points to be separate from one another
- Any symmetry present in the knot should be enforced exactly, and be visually apparent

Knot diagrams may be created using vectorized graphics software such as inkscape (Kirsanov 2009): one specifies a sequence of control points, then interpolates between these points to create a knot diagram with the appropriate topology. One way of smoothly interpolating between specified points is Bezier curves (Olsen 2014). A Bezier curve is a visually pleasing polynomial path that can be used to specify the path of a knot projection.

The package presented here allows one to specify a knot in terms of its Bezier control points within inkscape, import the object into R, and then to use numerical optimization techniques to improve the visual appearance of the knot.

2. The package in use

In this section, I give workflows for creating two simple knots: firstly 7_6, followed by the figure-of-eight knot 4_1 which requires imposition of symmetry constraints.

The first step is to create a closed curve in inkscape that shows the rough outline of the knot (Figure 2 shows a screenshot of `7_6_first_draft.svg`, supplied with the package). Note that this file contains only the knot *path*; the over and under information is to be added later.

Here, knot paths are required to have Bezier handles that are symmetrically placed with regard to nodes. Although radius of curvature is not necessarily matched at either side of a node, visual continuity of the strand is ensured. In the diagram, strand crossing points are far from nodes, as this ensures visual continuity of strands at crossing points, especially the understrand.

The knot shown in Figure 2 is clearly suboptimal: even though the nodes are connected by Bezier curves which are individually smooth, the path as a whole is ugly and unattractive as its path has unsightly regions where the radius of curvature changes abruptly.

Although it is possible in principle to improve the visual appearance of the knot path by hand in inkscape, this is a surprisingly difficult and frustrating task. In order to remedy the

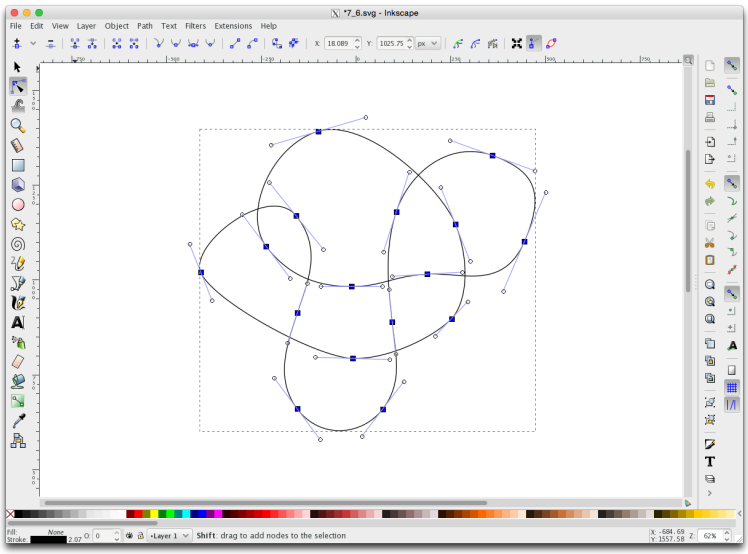

Figure 2: Screenshot of inkscape setup for knot 7_6. Nodes are shown as squares, handles as small circles, symmetrically placed on either side of nodes

flaws of the diagram using an automated system, we first read the `.svg` file into R using the `reader()` function:

```
> k76 <- reader(system.file("7_6_first_draft.svg",package="knotR"))
> head(k76)
```

```
              x         y
[1,]  -98.81963 339.81898
[2,] -223.87754 303.35366
[3,] -299.84521 121.06064
[4,] -236.36319  36.35340
[5,] -172.88118 -48.35384
[6,]  -92.86186 -69.78212
```

Object `k76` is stored as an object of class `inkscape`: a two-column matrix with rows corresponding to the node and handle positions of the inkscape path. This representation has a certain amount of redundancy as knot paths have handles which are symmetrically placed with respect to nodes; also, the first node is the same as the last for the loop is closed. The package can coerce inkscape objects to other forms, specifically `minobj` objects, which contain no redundancy (the position of each node, as well as one of the handles, is stored); or `controlpoints` objects, which allow for easy construction of Bezier interpolation between nodes.

To beautify it we need to specify a function of the path that quantifies its displeasingness, and then minimize this objective function using numerical methods.

Two examples of desiderata for such an objective function might be to keep the strands crossing at right angles, and the overall bending energy. These are evaluated in the package by functions `total_crossing_angles()` and `total_bending_energy()` respectively:

```
> par(oma=c(0, 0, 0, 0))
> par(mar=c(0, 0, 0, 0))
> par(plt=c(0, 1, 0, 1))
> k76_rough <- reader(system.file("7_6_first_draft.svg",package="knotR"))
> knotplot2(k76_rough, seg=TRUE)
```

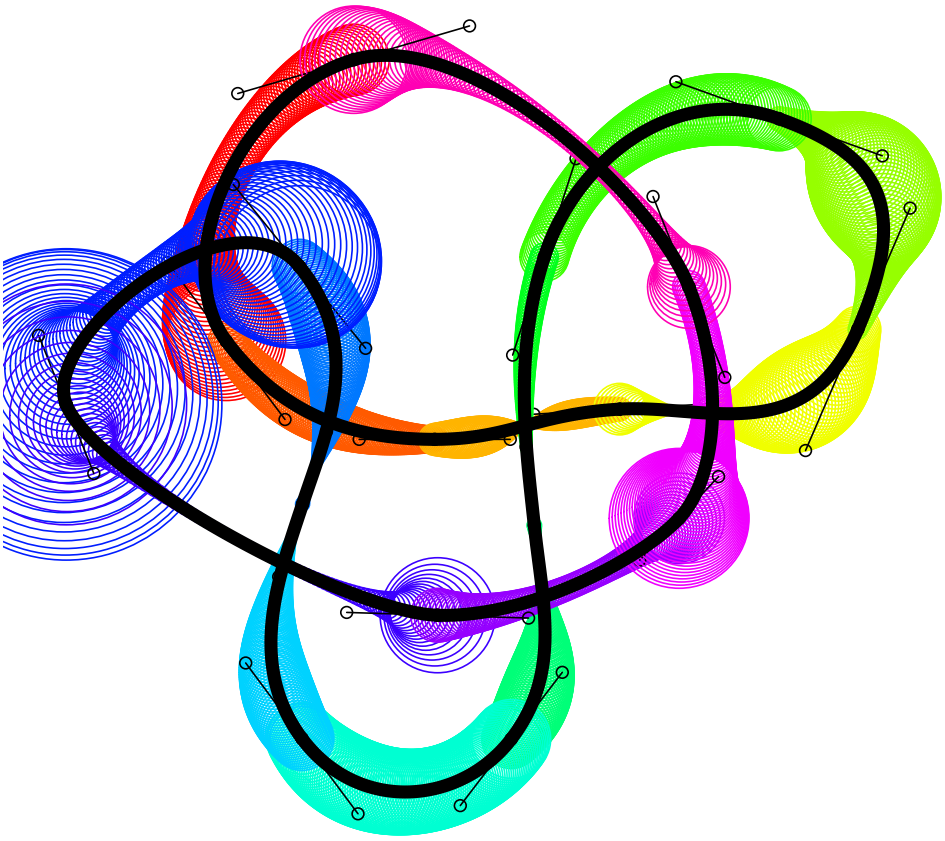

Figure 3: The *path* of knot 7_6, showing Bezier control points. Coloured circles have a radius proportional to the curvature (that is, the reciprocal of the radius of curvature) along the strand

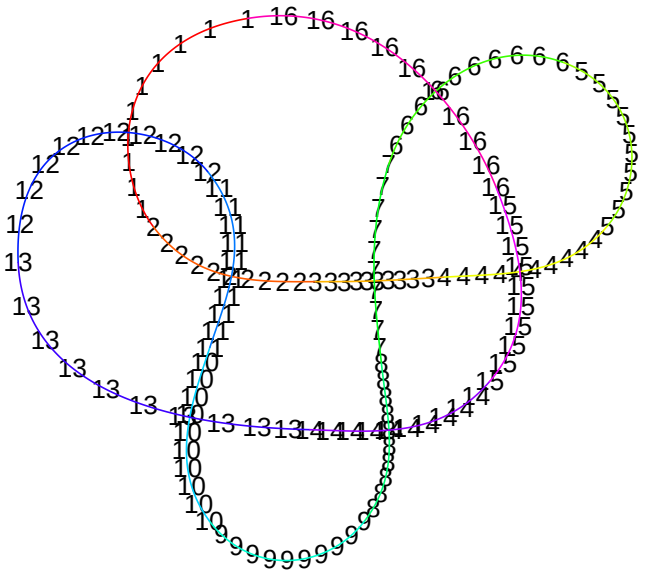

Figure 4: Knot 7_6 with strands numbered to guide creation of over and under information

```
> b <- as.controlpoints(k76_rough)
> total_crossing_angles(b)

[1] 0.3145033

> total_bending_energy(b)

[1] 0.1276257
```

The knots supplied in the package minimize a weighted sum of these and other badnesses[1], as evaluated by function `badness()`:

```
> badness(k76_rough)

[1] 6.16279
```

This function may be minimized by numerical optimization:

```
> o <- nlm(badness, as.knotvec(k76_rough))
> k7_6 <- as.minobj(o$estimate)
> badness(k7_6)
```

[1]Function `badness()` includes various "housekeeping" badnesses which are used to make sure that the minimum found by `nlm()` is topologically identical to the starting configuration. Function `non_crossing_strand_close_approach_badness()`, for example, makes non-crossing strands "repel" one another so as not to introduce spurious intersections.

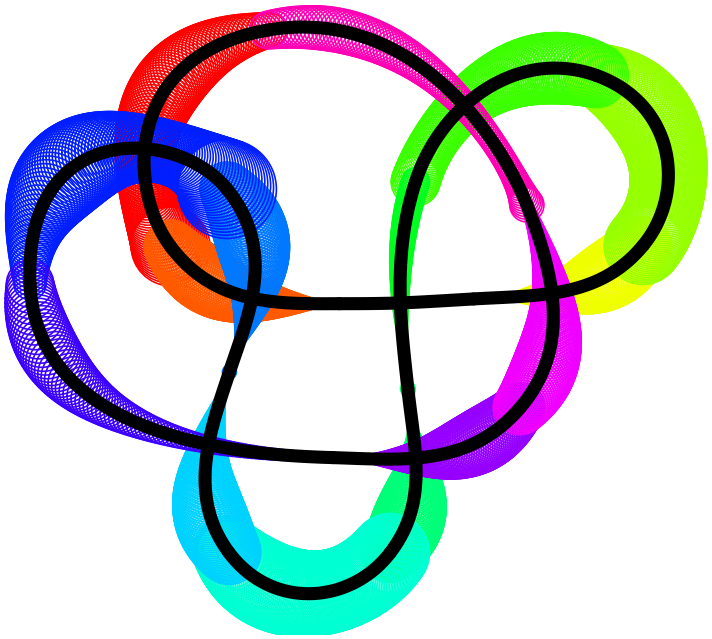

Figure 5: Knot 7_6, post-optimization

```
[1] 3.550152
```

(the above takes about an hour of CPU time: it is optimizing function of 64 real variables, and the objective function takes a few seconds to evaluate). However, the result is much nicer (Figure 5).

To specify the senses of the knot's crossings, we create an overunder object which is a two-column matrix:

```
> ou76 <- matrix(c(
+     12,01,
+     02,11,
+     07,03,
+     04,15,
+     16,06,
+     14,08,
+     10,13
+     ),byrow=TRUE,ncol=2)
```

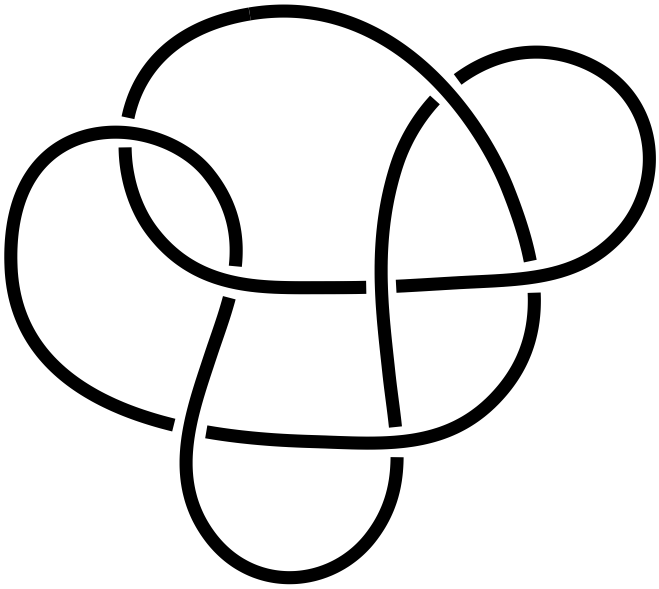

Figure 6: Knot 7_6, post-optimization with breaks indicating underpassing strands

With reference to Figure 4, each row of `ou76` corresponds to a crossing; the first element gives the overstrand and the second the understrand; thus strand 12 passes over strand 1, strand 2 passes over strand 11, and so on. The result is shown in figure 6.

2.1. Symmetry

Many of the knots in Figure 1 have an axis of symmetry, or possess rotational symmetry. The package has the ability to impose symmetry relations on knots, and to optimize the resulting symmetrical knot. Minimizing the badness is not entirely straightforward on account of the induced redundancy, which is characterized using a symmetry object specific to the knot under consideration. However, symmetry constraints do reduce the dimensionality of the optimization problem.

I will consider the figure-of-eight knot 4_1 as an example. Using Figure 7, top left, as reference, the appropriate symmetry object is defined as follows:

```
> fig8 <- reader(system.file("4_1_first_draft.svg",package="knotR"))
> Mver8 <- matrix(c(
```

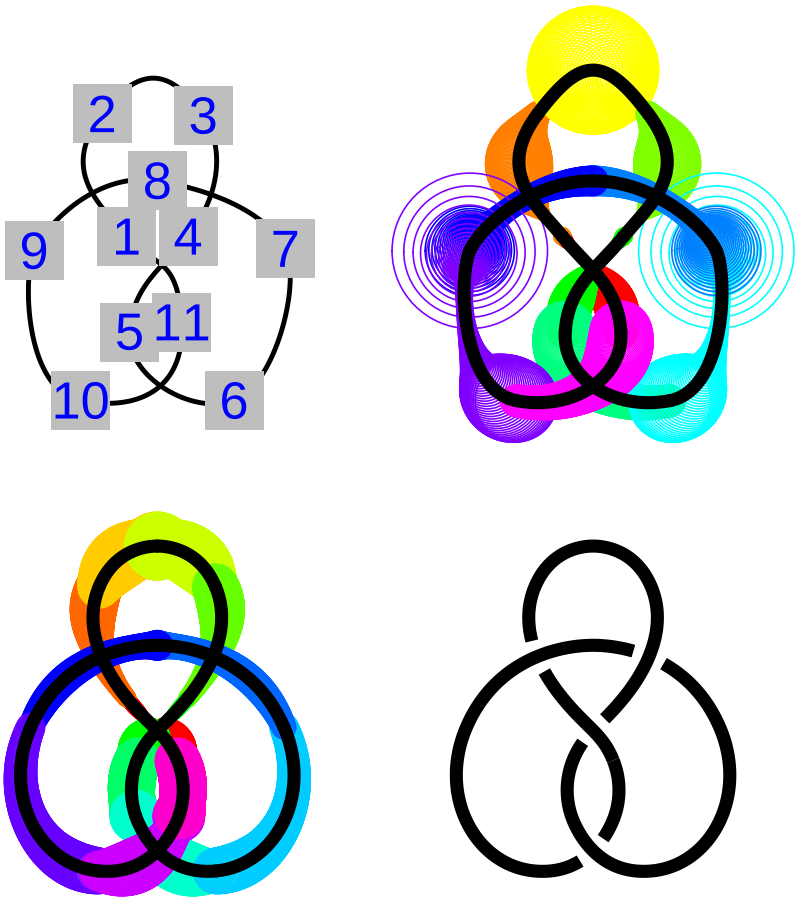

Figure 7: Figure eight knots drawn using different plotting methods. top left, knot path with node numbers shown in order to facilitate definition of the symmetry object; top right, result of symmetrizing the rough path; lower left, the optimized knot with imposed vertical symmetry, with curvature plotted; lower right, knot plotted with overstrand and understrand indicated using line breaks

```
+     02,03,
+     09,07,
+     05,11,
+     10,06
+     ),ncol=2,byrow=TRUE)
> sym8 <- symmetry_object(fig8, Mver=Mver8, xver=8)
```

(Matrix `Mver8` specifies that nodes 2 and 3 are symmetric, as are nodes 9 and 7, and so on; `xver=8` forces node 8 to be on the axis of symmetry). The results are shown in Figure 7.

2.2. Rotational symmetry

Consider knot 5_1. This knot has fivefold rotational symmetry, in addition to a vertical line of symmetry. The package includes functionality to impose appropriate symmetry constraints.

Using Figure 8 as reference, we have:

```
> knotplot2(k5_1,node=TRUE,width=FALSE)
```

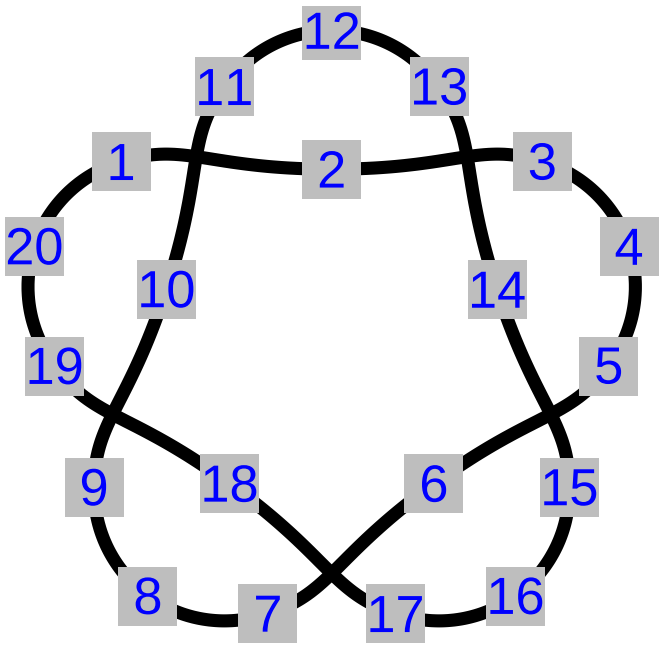

Figure 8: Knot 5_1 shown with node numbers

```
> sym51 <- symmetry_object(k5_1,
+                          Mver = cbind(11,13),
+                          xver = c(2,12),
+                          Mrot = rbind(
+                              c(12,04,16,08,20),
+                              c(13,05,17,09,01),
+                              c(11,03,15,07,19),
+                              c(02,14,06,18,10)
+                          ))
```

Thus, using the same notation as before, nodes 11 and 13 are symmetrical about the vertical axis, nodes 2 and 12 are on the vertical axis. The `Mrot` argument specifies sets of nodes

that map to themselves under rotation. The top line of `Mrot` indicates that nodes 12,4,16, 8, and 20 are concyclic. An example of a rotationally symmetric knot is given in Figure 10.

3. Conclusions and further work

The **knot** package allows the user to create rough diagrams of knots using the inkscape suite of software, and subsequently polish up such diagrams in terms of a customizable objective function using numerical optimization techniques. Further work might include functionality to deal with links.

4. Gallery

There now follows a selection of pleasing knot diagrams taken from datasets provided with the package.

```
> par(oma=c(0, 0, 0, 0))
> par(mar=c(0, 0, 0, 0))
> par(plt=c(0, 1, 0, 1))
> par(pty='m')
> plot(NULL,xlim=c(-700,2200),ylim=c(-1800,200),asp=1,box=FALSE)
> jjA <- as.inkscape(perko_A)*1.7
> jjB <- as.inkscape(perko_B)*1.7
> oA <- perko_A$overunder
> oB <- perko_B$overunder
> knotplot(jjA,ou=oA,setup=FALSE,gap=50,lwd=4)
> jjB <- sweep(as.inkscape(jjB),2,c(1500,-600),"+")
> knotplot(jjB,ou=oB,setup=FALSE,gap=50,lwd=4)
```

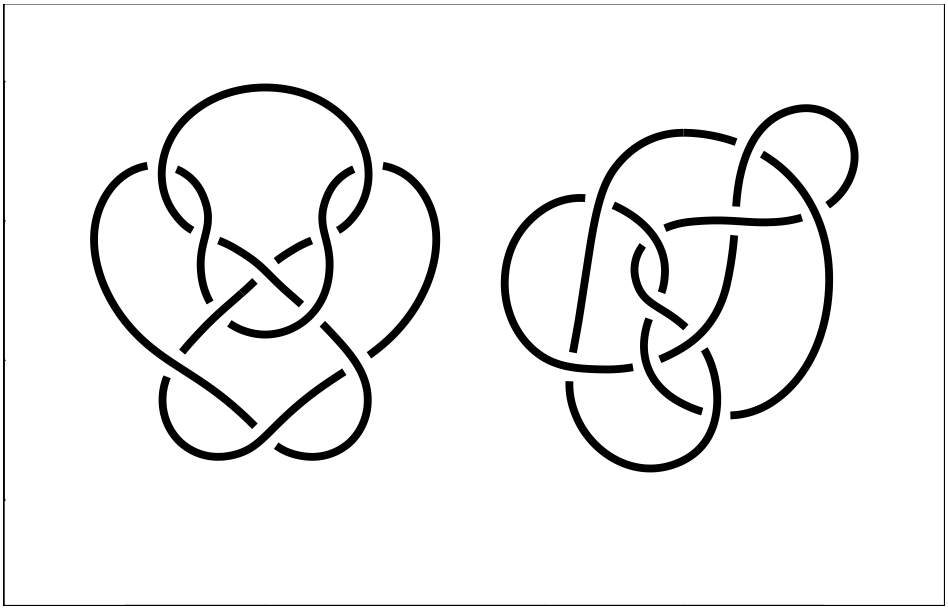

Figure 9: Two representations of knot 10_{125}, known as the Perko Pair

References

Alexander JW, Briggs GB (1926). "On Types of Knotted Curves." *Annals of Mathematics*, **28**(1/4), 562–586. ISSN 0003-486X. doi:10.2307/1968399. URL http://www.jstor.org/stable/1968399.

Kirsanov D (2009). *The Book of Inkscape: The Definitive Guide to the Free Graphics Editor*. 1 edition edition. No Starch Press, San Francisco. ISBN 978-1-59327-181-7.

```
> par(oma=c(0, 0, 0, 0))
> par(mar=c(0, 0, 0, 0))
> par(plt=c(0, 1, 0, 1))
> par(pty='m')
> knotplot(ornamental20,gap=20)
```

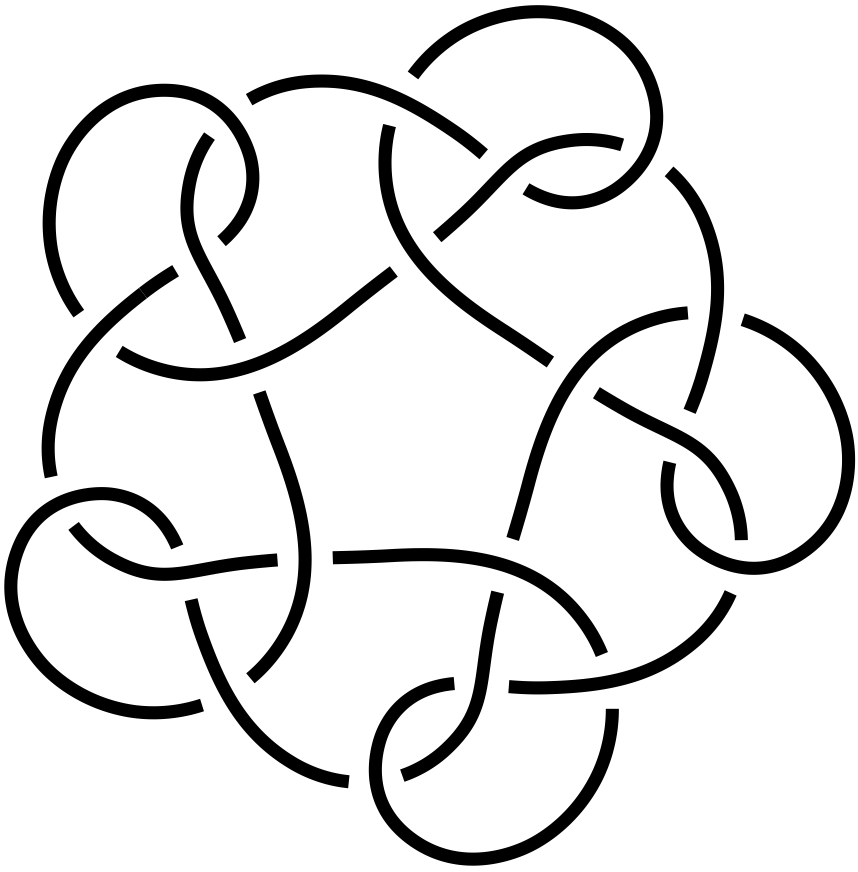

Figure 10: An ornamental knot exhibiting fivefold rotational symmetry; note the absence of mirror symmetry

```
> par(oma=c(0, 0, 0, 0))
> par(mar=c(0, 0, 0, 0))
> par(plt=c(0, 1, 0, 1))
> par(pty='m')
> knotplot(k10_123,gap=20)
```

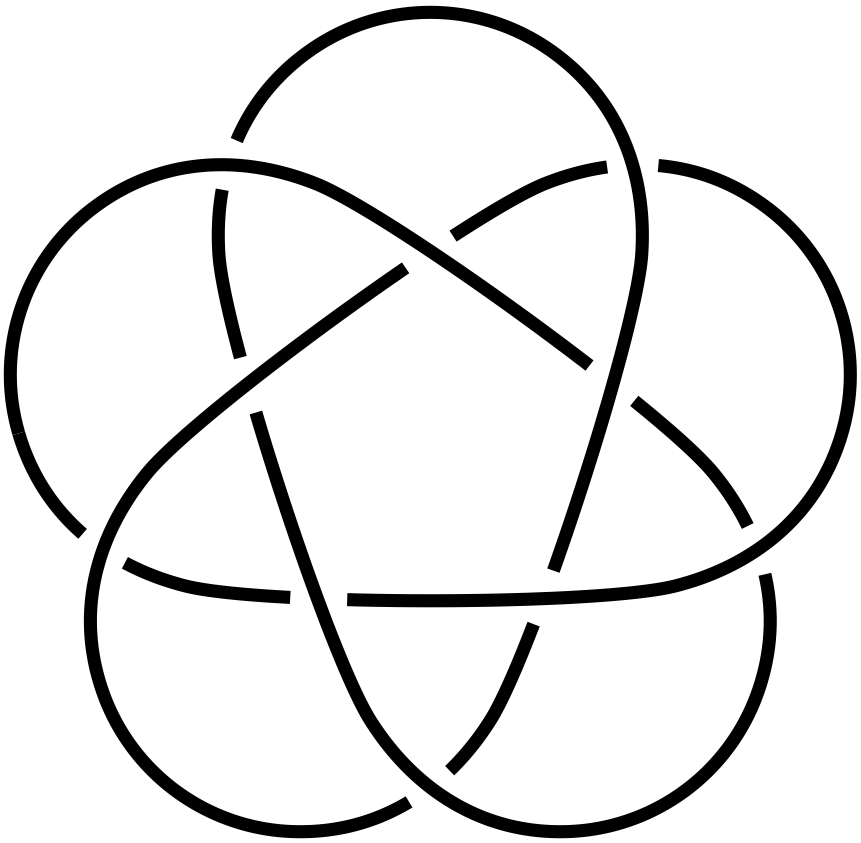

Figure 11: Knot 10_{123}, exhibiting fivefold symmetry

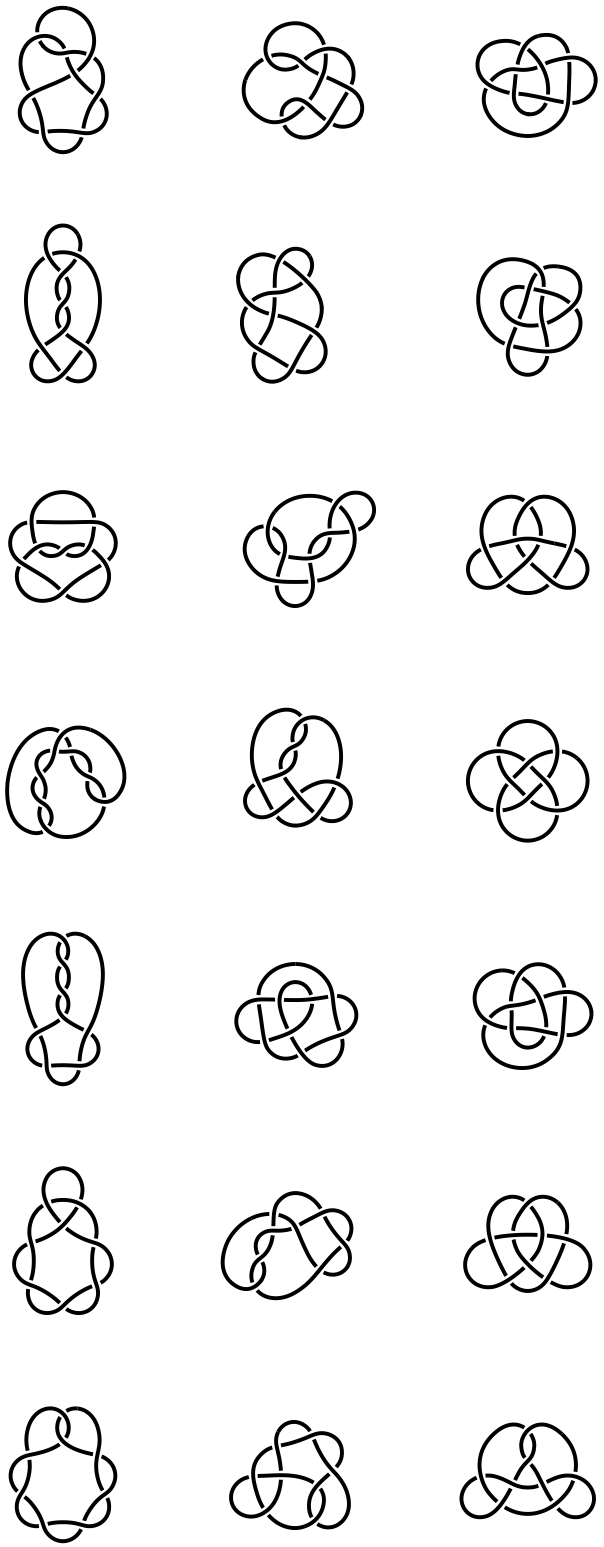

Figure 12: All prime eight-crossing knots, following Rolfsen

Manturov V (2004). *Knot Theory*. Chapman and Hall.

Olsen A (2014). *bezier: Bezier Curve and Spline Toolkit*. R package version 1.1, URL https://CRAN.R-project.org/package=bezier.

Wikipedia (2016). "Knot theory — Wikipedia, The Free Encyclopedia." [Online; accessed 17-June-2016], URL https://en.wikipedia.org/w/index.php?title=Knot_theory&oldid=724939634.

Affiliation:

Robin K. S. Hankin
Auckland University of Technology
New Zealand