```
SetDirectory["~/AcademicPensieve/Projects/HigherRank/DunfieldKnots"];
<< ../../KnotTheory/KnotTheory/init.m
<< ../Rot.m
```

```
Get: Cannot open KnotTheory`.
Loading Rot.m from http://drorbn.net/AP/Projects/HigherRank to compute rotation numbers.
```

(Alt) In[ ]:=

```
CCF[ℰ_] := ExpandDenominator@ExpandNumerator@Together[ℰ];
CCF[ℰ_] := Factor[ℰ];
CF[ℰ_List] := CF /@ ℰ;
CF[ℰ_] := Module[{vs = Cases[ℰ, (x | p | π | g)_, ∞] ∪ {x, p, ϵ}, ps, c},
```

Data

(from Talks/Beijing-2407/theta.nb)

(Alt) In[ ]:=

```
R1[1, i_, j_] = CF[
    1 / 2 - T3 g1ji g2ji - g3ii + g2jj g3ii + T1 (T3 - 1) g1ji g3ji +
     T2 (T3 - 1) g2ji g3ji - T2 g2ji g3jj + (g1jj g2ii + (T3 - 1) g1jj g2ji -
```

(Alt) In[ ]:=

```
R1[-1, i_, j_] = CF[
    -1 / 2 - T1^-1 g1ji g2ii - (1 - T1^-1 - T2^-1) g1ji g2ji - g1jj g2ji - g1ji g2jj + g3ii +
     T1^-1 g1ji g3ii - (1 - T2^-1) g2ji g3ii - g2jj g3ii + (1 - T3^-1) g1ji g3ji - (1 - T3^-1) g2ii g3ji +
```

(Alt) In[ ]:=

```
θ[{1, i0_, j0_}, {1, i1_, j1_}] =
   -T1 (T3 - 1) g1,j1,i0 g2,i1,i0 g3,j0,i1 + (T3 - 1) g1,j1,j0 g2,i1,i0 g3,j0,i1 +
```

(Alt) In[ ]:=

```
θ[{1, i0_, j0_}, {-1, i1_, j1_}] =
   (T3 - 1) g1,j1,i0 g2,i1,i0 g3,j0,i1 - T1^-1 (T3 - 1) g1,j1,j0 g2,i1,i0 g3,j0,i1 -
```

(Alt) In[ ]:=

```
θ[{-1, i0_, j0_}, {1, i1_, j1_}] = CF[
    T1^-1 T2^-1 (T3 - 1) (g1,j1,i0 g2,i1,i0 g3,j0,i1 -
```

(Alt) In[ ]:=

```
θ[{-1, i0_, j0_}, {-1, i1_, j1_}] = CF[
    (1 - T3^-1) (-T1^-1 g1,j1,i0 g2,i1,i0 g3,j0,i1 +
```

(Alt) In[ ]:=

```
Γ1[φ_, k_] = -φ / 2 + φ g3,k,k;
```

The Programs

```
θ[K_] := Module[{Cs, φ, n, A, s, i, j, k, Δ, G, ν, α, β, gEval, c, z},
   {Cs, φ} = Rot[K]; n = Length[Cs];
   A = IdentityMatrix[2 n + 1];
                                                   -T^s  T^s - 1
   Cases[Cs, {s_, i_, j_} :→ (A[[{i, j}, {i + 1, j + 1}]] += (                 ))];
                                                    0     -1
   Δ = T^((-Total[φ]-Total[Cs[[All,1]]])/2) Det[A];
   G = Inverse[A]; gEval[ℰ_] := Factor[ℰ /. g_{ν_,α_,β_} :→ (G[[α, β]] /. T → T_ν)];
   z = gEval[Sum[θ[Cs[[k1]], Cs[[k2]]], {k1, n}, {k2, n}]];
```

(Alt) In[•]:=

```
PolyPlot[0] = Graphics[{}];
PolyPlot[p_] := Module[{crs, m1, m2, maxc, minc, s, hex},
   crs = CoefficientRules[T_1^(m1=-Exponent[p,T_1,Min]) T_2^(m2=-Exponent[p,T_2,Min]) p, {T_1, T_2}];
   maxc = N@Log@Max@Abs[Last /@ crs];
   minc = N@Log@Min@Select[Abs[Last /@ crs], # > 0 &];
   If[minc == maxc, s[_] = 0, s[c_] := s[c] = (maxc - Log@c) / (maxc - minc)];
   hex = Table[{Cos[α], Sin[α]} / Cos[2 π / 12] / 2, {α, 2 π / 12, 2 π, 2 π / 6}];
   Graphics[crs /. ({x1_, x2_} → c_) :→ {
       If[c == 0, White, Lighten[If[c > 0, Red, Blue], 0.88 s[Abs@c]]]
```

Testing

In[•]:=

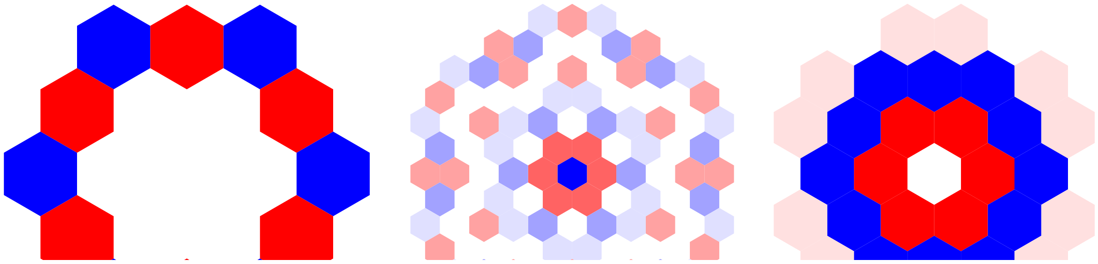
```
GraphicsRow[PolyPlot[θ[Knot[#]][[2]]] &
   /@ {"3_1", "K11n34", "K11n42"}]
```

KnotTheory: Loading precomputed data in PD4Knots`.

KnotTheory: Loading precomputed data in DTCode4KnotsTo11`.

KnotTheory: The GaussCode to PD conversion was written by Siddarth Sankaran at the University of Toronto in the summer of 2005.

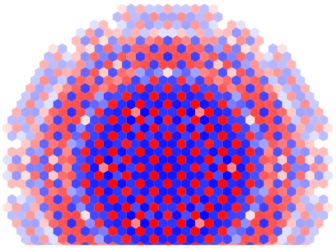
Out[•]=



In[•]:= AbsoluteTiming@

$$\text{PolyPlot}\Big[\theta\Big[\text{EPD}\Big[X_{14,1}, \overline{X}_{2,29}, X_{3,40}, X_{43,4}, \overline{X}_{26,5}, X_{6,95}, X_{96,7}, X_{13,8}, \overline{X}_{9,28}, X_{10,41}, X_{42,11}, \overline{X}_{27,12},$$
$$X_{30,15}, \overline{X}_{16,61}, \overline{X}_{17,72}, \overline{X}_{18,83}, X_{19,34}, \overline{X}_{89,20}, \overline{X}_{21,92}, \overline{X}_{79,22}, \overline{X}_{68,23}, \overline{X}_{57,24}, \overline{X}_{25,56}, X_{62,31},$$

Out[•]=

{13.1716,  }

Run

(Alt) In[•]:=

```
DunfieldKnots =
  ReadList["../../../People/Dunfield/nmd_random_knots"] /. k_Integer :→ k + 1;
```

(Alt) In[•]:=

```
DKString[n_] := StringDrop[ToString[1000 + n], 1]
```

In[•]:= DKString[76]

Out[•]=

```
076
```

(Alt) In[•]:=

```
Do[
 If[FileExistsQ[from = "D" <> ToString[n] <> ".m"],
  RenameFile[from, "D" <> DKString[n] <> ".m"]];
 If[FileExistsQ[from = "PP" <> ToString[n] <> ".png"],
```

```
Clear[at, pp];
Do[
 If[(n = k) > 1000, Abort[]];
 If[Not@FileExistsQ["D" <> DKString[n] <> ".m"],
  Print["Working on ", n];
  Put[
   ({at, th} = AbsoluteTiming[θ[DK[n]]]) /. {T_1 → T1, T_2 → T2},
   "D" <> DKString[n] <> ".m"];
  Print@Export["PP" <> DKString[n] <> ".png", pp = PolyPlot[th〚2〛]];
```

```
Join[mon, {n}]
```