Alexander

January 30, 2022

```
In [1]: from alexander import *
```

1 Bridge positions

We will represent knots by bridge positions and use the following conventions:

Assume that all crossings, local minimums and maximums have a different height (y coordinate on the plane).

We start at the global maximum for the y coordinate.

If the $y = C$ line intersects the knot transversally in $2k$ points, the strands are labeled from left to right on this line, from 1 to $2k$.

Maximums where the new cap is between strands t and $t+1$ are represented by $100+t$.

Minimums where the cup is between strands t and $t+1$ are represented by $-100-t$.

Right-handed crossings between strands t and $t+1$ are represented by t.

Left-handed crossings between strands t and $t+1$ are represented by $-t$.

An example: $K = [101, 102, 104, 1, 3, 5, -102, -102, -101]$. This is the left handed trefoil presented as the three stranded pretzel knot $P(1,1,1)$.

Unknot = $[101, -101]$ (First a maximum between strands 1 and 2, then a minimum between strands 1 and 2)

Right handed trefoil = $[101, 103, 2, 2, 2, -101, -101]$

We orient the knot so that the last minimum is oriented from left to right.

Functions that are included: IsKnot(K), Pretzel(a,b,c), Torus(p,q), Mirror(K).

```
In [2]: K1 = [101,102,104,1,3,5,-102,-102,-101] #This is the Left-handed trefoil
```

```
In [3]: IsKnot(K1)
```

```
Out[3]: True
```

```
In [4]: K2 = [101, -101]
        IsKnot(K2)
```

```
Out[4]: True
```

```
In [5]: K3 = Pretzel(-3,4,7)
        # A three stranded pretzel knot, positive values indicate right handed twists
        K3
```

```
Out[5]: [101, 102, 104, -1, -1, -1, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, -102, -102, -101]
```

```
In [6]: print(Mirror(K3))
```

```
[101, 102, 104, 1, 1, 1, -3, -3, -3, -3, -5, -5, -5, -5, -5, -5, -5, -102, -102, -101]
```

```
In [7]: L1 = Pretzel(-2,2,7)
        IsKnot(L1) # This is a two component link, not a knot
```

```
Out[7]: False
```

```
In [8]: K4 = Torus(3,4)
        print(K4)
```

```
[101, 102, 103, 2, 1, 2, 1, 2, 1, 2, 1, -103, -102, -101]
```

```
In [9]: K5 = Torus(3,-4)
        print(K5)
```

```
[101, 102, 103, -2, -1, -2, -1, -2, -1, -2, -1, -103, -102, -101]
```

```
In [10]: L2 = Torus(3,6) #The 3, 6 Torus Link
         IsKnot(L2)
```

```
Out[10]: False
```

```
In [11]: Mirror(K5)
```

```
Out[11]: [101, 102, 103, 2, 1, 2, 1, 2, 1, 2, 1, -103, -102, -101]
```

2 Kauffman states and the Alexander polynomial

We use a state sum formula on the Kauffman states to compute the Alexander polynomial.

The algorithm uses an inductive computations for portions of the Bridge presentation.

This amounts to computing upper Kauffman states for the part of the knot that lies in the half-plane $y \geq C$.

The key step is to compute what happens as C decreases and the $y = C$ line goes through a crossing or a maximum or a minimum.

In terms of a bridge presentation K this is is equivalent to computing for some function F the value of $F(K[0:t])$ in terms of $F(K[0:t-1])$ and $K[t-1]$.

In later lectures we will see how this can be done also for the Jones polynomial, Knot Floer homology, and Khovanov homology.

Functions:

NumberOfKauffmanStates(K)

Alexander(K) returns the Alexander polynomial of K as a tuple of values $((a_1, b_1), \cdots, (a_k, b_k))$ where (a, b) contributes $b \cdot T^a$.

PrintAlexander(K) gives the answer in a string format.

```
In [12]: NumberOfKauffmanStates(K1)
```

```
Out[12]: 3

In [13]: Alexander(K1)

Out[13]: ((-1, 1), (0, -1), (1, 1))

In [14]: PrintAlexander(K1)

Out[14]: 'T^{-1}-1+T^{1}'

In [15]: Alexander(K1) == Alexander(Mirror(K1))

Out[15]: True

In [16]: Alexander(Torus(3,4))

Out[16]: ((-3, 1), (-2, -1), (0, 1), (2, -1), (3, 1))

In [17]: Alexander(Torus(4,5))

Out[17]: ((-6, 1), (-5, -1), (-2, 1), (0, -1), (2, 1), (5, -1), (6, 1))

In [18]: PrintAlexander(Torus(4,5))

Out[18]: 'T^{-6}-T^{-5}+T^{-2}-1+T^{2}-T^{5}+T^{6}'

In [19]: PrintAlexander(Pretzel(-2,5,7))

Out[19]: 'T^{-6}-T^{-5}+T^{-3}-2T^{-2}+3T^{-1}-3+3T^{1}-2T^{2}+T^{3}-T^{5}+T^{6}'

In [20]: Alexander(Pretzel(-3,3,3))

Out[20]: ((-1, -2), (0, 5), (1, -2))

In [21]: PrintAlexander(Torus(5,6))

Out[21]: 'T^{-10}-T^{-9}+T^{-5}-T^{-3}+1-T^{3}+T^{5}-T^{9}+T^{10}'

In [22]: NumberOfKauffmanStates(Torus(5,6))

Out[22]: 261725

In [23]: NumberOfKauffmanStates(Torus(5,21))

Out[23]: 11035990217440195921

In [24]: PrintAlexander((101,-101)) #The Unknot

Out[24]: '1'

In [25]: PrintAlexander(Pretzel(-3,5,7))
         # A nontrivial knot with the same Alexander polynomial as the unknot
```

Out[25]: '1'

In [26]:
```
for i in range(-15,-1): #Looking for other such examples
    for j in range(2,31):
        for k in range(j,31):
            K = Pretzel(i,j,k)
            if IsKnot(K) and PrintAlexander(Pretzel(i,j,k))== '1':
                print(i,j,k)
```

```
-13 25 27
-11 21 23
-9 13 29
-9 17 19
-7 11 19
-7 13 15
-5 7 17
-5 9 11
-3 5 7
```