Week3

February 6, 2022

```
In [1]: from week3 import *
        ## Assumes that the following files are in your directory:
        ## the header file week3.py and
        ## knot_examples.py
        ## knot_utilities.py
        ## alexander1.py
        ## seifert.py
        ## knot_signature.py
        ## knotfloer1.py

        ## You can also import just some of the modules, for example :
        ## from seifert import *
        ## for this you will need to have the file
        ## seifert.py and its dependencies,
        ## (in this case knot_utilities.py and alexander1.py)
```

1 Examples

The following functions are included in knot_examples:

Unknot(), LHT(), RHT(). Basic examples: unknot, left-handed trefoil, right-handed-trefoil.
Torus(p,q): Torus knots, p,q is relatively prime.
Pretzel(a,b,c): Three stranded pretzel knots, at most one of a, b, c is even.
Conway(): The Conway knot.
KT(): The Kinoshita-Terasaka knot.
Mirror(K): Changes all the crossings in the projection. It represents the mirror of the knot.
Reverse(K): Adds 1 more crossing close to the global minimum. It represents the reverse of the knot.

```
In [52]: K1 = Conway() #The Conway knot
         print(K1)
```

```
(101, 101, 2, 2, 102, 106, 1, 1, -3, -3, -3, -5, -5, 7, 7, 7, -102, -102, -102, -101)
```

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

```
Out[3]: ((0, 1),)
```

```
In [4]: PrintAlexander(K1)
```

```
Out[4]: '1'
```

```
In [5]: K2 = KT() # The Kinoshita-Terasaka knot
        print(K2)
```

```
(101, 101, 2, 2, 102, 106, 1, 1, -3, -3, -3, 5, 5, 5, -7, -7, -102, -102, -102, -101)
```

```
In [6]: K3 = Pretzel(2,3,5)
        PrintAlexander(K3)
```

```
Out[6]: '-T^{-4}+3T^{-3}-4T^{-2}+5T^{-1}-5+5T^{1}-4T^{2}+3T^{3}-T^{4}'
```

```
In [7]: K4 = Torus(3,4)
        KnotDeterminant(K4)
```

```
Out[7]: -3
```

```
In [8]: KnotSignature(K4)
```

```
Out[8]: -6
```

```
In [9]: Torus(7,2)
```

```
Out[9]: (101, 102, 1, 1, 1, 1, 1, 1, 1, -102, -101)
```

```
In [10]: Mirror(Torus(7,2))
```

```
Out[10]: (101, 102, -1, -1, -1, -1, -1, -1, -1, -102, -101)
```

```
In [11]: IsAlternatingProjection(Torus(7,2))
```

```
Out[11]: True
```

```
In [12]: IsAlternatingProjection(Torus(7,3))
```

```
Out[12]: False
```

```
In [13]: IsKnot(Pretzel(-3,5,7))
```

```
Out[13]: True
```

```
In [14]: IsKnot(Pretzel(2,3,5))
```

```
Out[14]: True
```

```
In [15]: IsKnot(Pretzel(2,4,7))
```

```
Out[15]: False
```

```
In [16]: Alexander(Pretzel(2,4,7))
```

```
Not a knot
```

2 The Alexander polynomial

The following functions are included in alexander1:

Alexander(K): The symmetric Alexander polynomial, computed using Kauffman states.

PrintAlexander(K): The answer in string format

KnotDeterminant(K): Defined asAlexander(K) evaluated at -1.

```
In [17]: KnotDeterminant(Pretzel(3,5,7))

Out[17]: -71

In [18]: KnotDeterminant(Pretzel(-3,5,7))

Out[18]: 1

In [19]: KnotDeterminant(Torus(7,4))

Out[19]: -7

In [20]: KnotDeterminant(Torus(7,5))

Out[20]: 1

In [21]: PrintAlexander(Torus(4,5))

Out[21]: 'T^{-6}-T^{-5}+T^{-2}-1+T^{2}-T^{5}+T^{6}'

In [22]: Alexander(Torus(4,5))

Out[22]: ((-6, 1), (-5, -1), (-2, 1), (0, -1), (2, 1), (5, -1), (6, 1))

In [23]: Alexander(Pretzel(-2,3,5))

Out[23]: ((-4, 1), (-3, -1), (-1, 1), (0, -1), (1, 1), (3, -1), (4, 1))

In [24]: PrintAlexander(Pretzel(-2,3,5))

Out[24]: 'T^{-4}-T^{-3}+T^{-1}-1+T^{1}-T^{3}+T^{4}'
```

3 Seifert surfaces

The following functions are included in seifert:

GenusFromSeifertAlgorithm(K): Implements the Seifert algorithm and computes genus of the resulting surface.

GenusBounds(K): Uses the Alexander polynomial and the Seifert algorithm to get lower and upper bounds on g(K).

```
In [25]: GenusFromSeifertAlgorithm(Torus(3,7))

Out[25]: 6
```

```
In [26]: GenusBounds(Torus(3,7))
         #The Seifert genus is 6

Out[26]: [6, 6]

In [27]: GenusFromSeifertAlgorithm(KT())

Out[27]: 4

In [28]: GenusBounds(KT())
         # The Seifert genus is somewhere between 0 and 4. (In fact it is equal to 2)

Out[28]: [0, 4]

In [29]: GenusBounds(Pretzel(11,13,15))

Out[29]: [1, 1]

In [30]: GenusBounds(Pretzel(2,5,7))

Out[30]: [6, 6]

In [31]: GenusBounds(Pretzel(-2,3,7))

Out[31]: [5, 5]

In [32]: GenusBounds(Pretzel(-3,5,7))
         # As soon as we show that this is a non-trivial knot it follows that g = 1.

Out[32]: [0, 1]

In [33]: GenusBounds(Pretzel(1,-1,1))
         # This is the unknot, so genus is in fact 0.

Out[33]: [0, 1]
```

4 Signature of knots

The following functions are included in knot_signature:

KnotSignatureForAlternating(K): This function assumes that K is an alternating diagram.

KnotSignature(K): Computes the signature for a general projection.

```
In [34]: KnotSignature(Conway())

Out[34]: 0

In [35]: KnotSignature(Torus(5,7))

Out[35]: -16

In [36]: KnotSignature(Mirror(Torus(5,7)))
```

```
Out[36]: 16

In [37]: KnotSignature(Torus(-5,7))

Out[37]: 16

In [38]: KnotSignatureForAlternating(Pretzel(17,17,2))

Out[38]: -32

In [39]: KnotSignature(Pretzel(17,17,2))

Out[39]: -32

In [40]: KnotSignature(Pretzel(-17,-17,-2))

Out[40]: 32

In [41]: KnotSignatureForAlternating(Pretzel(2,3,-5))

Not an alternating projection


In [42]: KnotSignature(Pretzel(2,3,-5))

Out[42]: 2
```

5 A first look at Knot Floer Homology

The following functions are included in knot_floer1:

KnotFloerGenerators(K): Computes the bigraded Kauffman states

GenusBounds2(K): Computes a slightly improved upper bound for the genus using Kauffman states.

```
In [43]: KnotFloerGenerators(Torus(2,3))

Out[43]: ((-1, -2, 1), (0, -1, 1), (1, 0, 1))

In [44]: KnotFloerGenerators(Torus(-2,3))

Out[44]: ((-1, 0, 1), (0, 1, 1), (1, 2, 1))

In [45]: KnotFloerGenerators(Torus(3,4))

Out[45]: ((-3, -6, 1),
          (-2, -5, 2),
          (-2, -4, 1),
          (-1, -4, 3),
          (-1, -3, 4),
          (-1, -2, 1),
```

```
          (0, -3, 4),
          (0, -2, 10),
          (0, -1, 6),
          (0, 0, 1),
          (1, -2, 3),
          (1, -1, 4),
          (1, 0, 1),
          (2, -1, 2),
          (2, 0, 1),
          (3, 0, 1))
```

```
In [46]: KnotFloerGenerators(KT())
```

```
Out[46]: ((-2, -2, 1),
          (-2, -1, 1),
          (-1, -1, 5),
          (-1, 0, 5),
          (0, 0, 13),
          (0, 1, 12),
          (1, 0, 1),
          (1, 1, 19),
          (1, 2, 18),
          (2, 1, 2),
          (2, 2, 19),
          (2, 3, 17),
          (3, 2, 2),
          (3, 3, 12),
          (3, 4, 10),
          (4, 3, 1),
          (4, 4, 4),
          (4, 5, 3))
```

```
In [47]: GenusBounds(KT())
```

```
Out[47]: [0, 4]
```

```
In [48]: GenusBounds2(KT())
```

```
Out[48]: [0, 2]
```

```
In [49]: KnotFloerGenerators(Pretzel(-3,5,7))
```

```
Out[49]: ((-1, -1, 6), (-1, 0, 6), (0, 0, 18), (0, 1, 17), (1, 1, 12), (1, 2, 12))
```

```
In [50]: KnotFloerGenerators(Pretzel(3,-5,-7))
```

```
Out[50]: ((-1, -2, 12), (-1, -1, 12), (0, -1, 17), (0, 0, 18), (1, 0, 6), (1, 1, 6))
```