COMPUTING THE ALEXANDER POLYNOMIAL FROM SEIFERT SURFACES

These are some rough notes with no proofs I wrote on computing the Seifert matrix and using it to compute the Alexander polynomial from a Seifert surface. Beware that they are written for a high school audience (and all pictures were drawn by hand on a train!). For better 3D pictures generated by the students[1] the notes were aimed at that make illustrating the computations much clearer, can be found at `https://github.com/zoemarschner/math-seminar`.

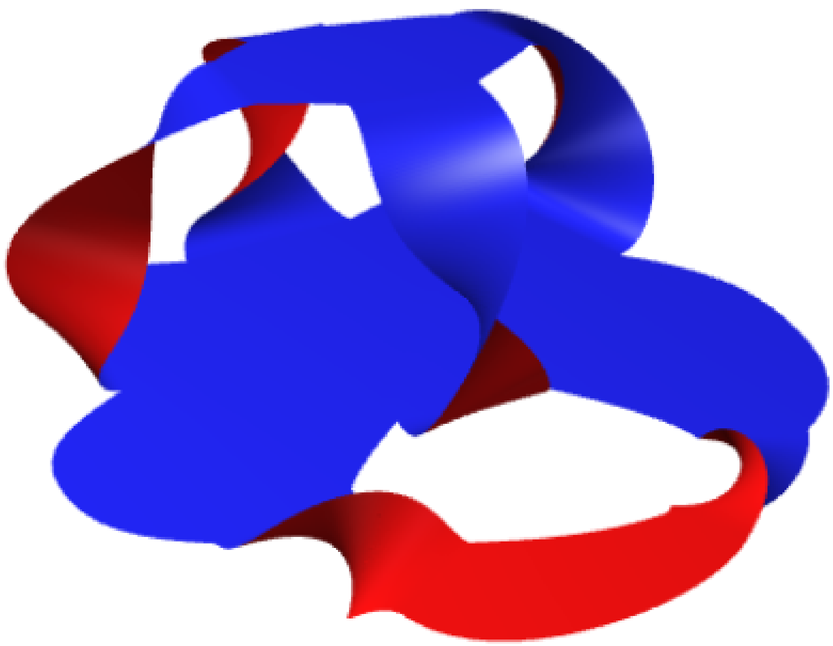


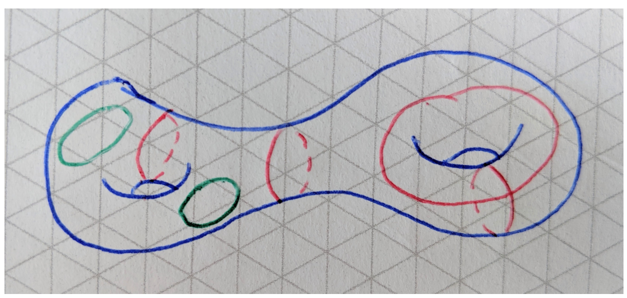
FIGURE 1. Seifert surface for the knot 6_2 generated by the program

[1]Zoe Marschner, Dora Kassabova, and Kasia Fadeeva

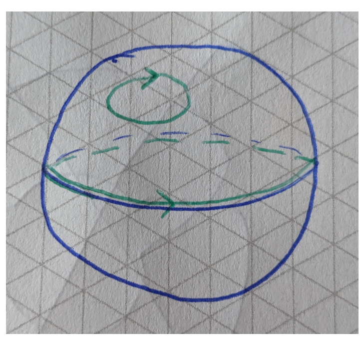
The idea is to construct a matrix of size $2g \times 2g$, called the **Seifert matrix**, where g is the genus of your Seifert surface. The entries of the matrix are given by considering linking numbers of loops in and over your surface (recall that the linking number is the number of times a loop winds around another loop, and can be computed as $\frac{p-n}{2}$ where p is the number of positive crossings and the n is the number of negative crossings, so the loops need to be oriented). Defining these loops is where it gets a little more complicated - we'll call them the "generators" or "generating loops". In total you will need 2g loops, and each of these should not be "equivalent" to any of the other loops, where "equivalent" means that you can pull one loop through your surface until it is the second loop. You also don't want any of the loops to be "trivial", where trivial means you can shrink the loop down to a point. Basically a loop will not be trivial if it loops around a hole in your surface, and two loops will not be equivalent if they loop around different holes in your surface.

Remember that we are only considering the (two-dimensional) surface, the "inside" of the shapes is empty, and not part of our space.

Example 1. In this example all the red loops are non-trivial and the green loops are trivial. A trivial loop is the boundary of (up to some stretching or pushing) a disk. Note that none of the red loops can be pulled to be the boundary of a disk, since they're restricted by the holes that they encircle.

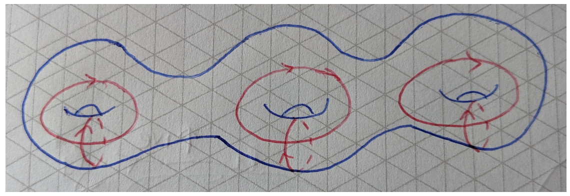


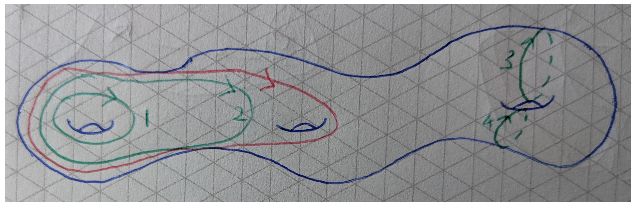
Example 2.
On a sphere, all loops are trivial.



Example 3. On this genus 3 surface, all loops are non-equivalent (you'll note that there are 2g of them, which is exactly what we wanted).

Example 4.
On this surface, the green loops 1 and 2 are equivalent, as are loops 3 and 4, but the red loop is not equivalent to any other loops.

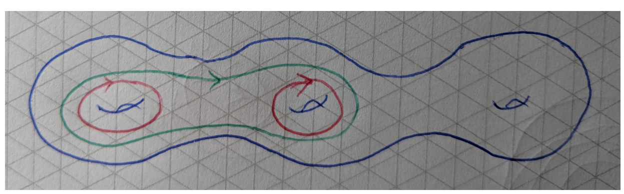




A couple of points: there are lots of different choices of $2g$ pairwise non-equivalent generating loops on the surface, each of these choices will give you slightly different matrices, but will not affect the final polynomial invariant. Note also that all of the loops are oriented.

I swept a very important point under the rug, which is that we can endow the set of loops on a surface with an algebraic structure, by joining loops together and traversing them in the opposite direction. It's not worth getting into the details at the moment (this set is actually a group, called the first homology group of the surface), but this means that some loops that seem to be non-equivalent to other loops in the surface might actually be equivalent to a join of two other loops, and hence should not be used as a loop to define the Seifert matrix. This group given by joining generating loops is also an invariant of the Seifert surface (and more generally of any topological space), which makes it clearer why the Alexander polynomial should be an invariant.

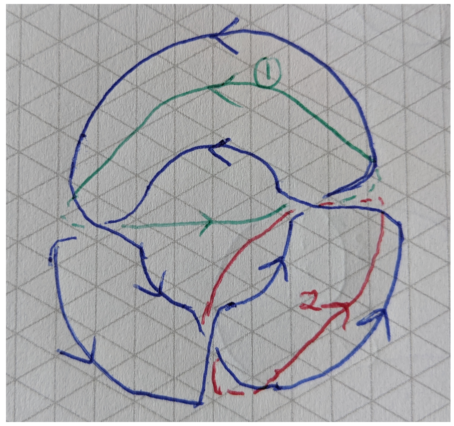
Example 5. In the surface below, the green loop should not be considered as a generating loop, since it can be obtained by joining the two red loops.



As a consequence, you need to be slightly careful when choosing loops in your Seifert surface.

Example 6. For the example of the trefoil, the following two loops are the generating loops for our set (recall that the genus of this surface is 1). I don't need a third loop in the surface, because any other loop can be obtained from the first two by joining.

More generally, your non-trivial loops that will be easiest to work with will be loops that go around two consecutive twisted bands (the bands joining the disks that are bounded by Seifert circles given by the Seifert algorithm). These bands should have at least one and at most two loops traversing them to ensure that you have a generating set of loops (recall also that there should be $2g$ of them).



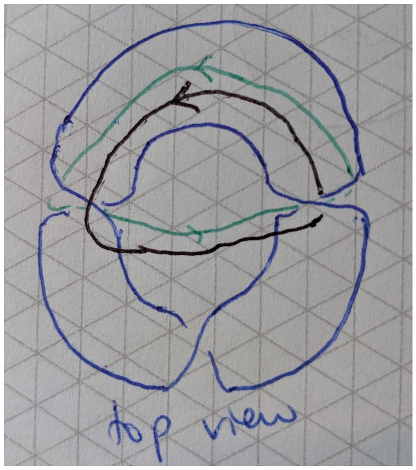
Label each of your loops as l_1 through l_{2g} to form the matrix V. The (i, j)th entry of the matrix will be given by the following formula:

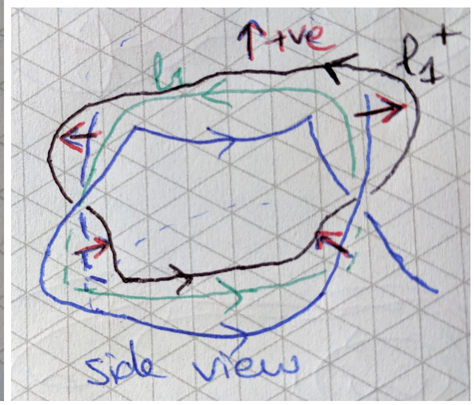
$$V(i, j) = lk(l_i, l_j^+)$$

where $lk(L, M)$ is the linking number of the oriented loops L and M and L^+ is the "positive push off" of L. This push off is a loop that runs parallel to L (in the same direction) just above the Seifert surface. The concept of "above" makes sense because the surface is orientable, so we have a concept of top and bottom and hence of above and below. Choose the orientation of your Seifert surface that is consistent with the orientation of the knot that bounds it (use the right hand rule). When I say "just", I mean that the amount you push the loop off the surace should be small in comparison to the size of your Seifert surface, and to the loops themselves. Otherwise your linking number will always be zero. The places to be careful when computing this push off are at the twisted bands, this is what will determine the linking number. It's a little bit difficult for me to draw these push off loops in a way that you can really see what is going on, but I'll give it a go.

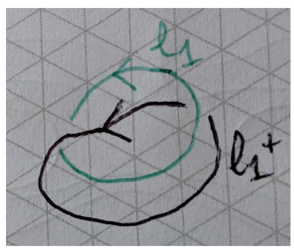
Example 7. In the example of the trefoil above, we have the generating loops 1 and 2 for the Seifert surface. The orientation of my Seifert circles is such that the positive normal direction (the "top") of the Seifert surface comes out of the page.

The Seifert matrix for this surface is a 2×2 matrix. To compute the $(1, 1)$-entry, we want to find the linking number of l_1 with l_1^+, drawn below, where the green loop is the original loop l_1 and the black loop is the pushed off loop l_1^+.





Without the Seifert surface in the way, we have the following picture for l_1 and l_1^+.



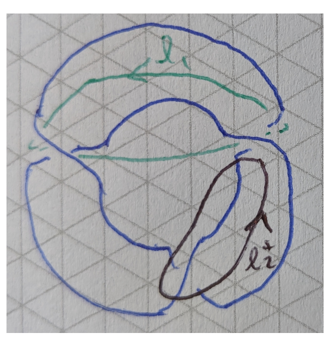
Therefore, we see that

$$V(1,1) = lk(l_1, l_1^+) = \frac{2-0}{2} = 1$$

Since all the crossings are the same, we see that the computation for the $(2,2)$ entry is exactly the same, namely

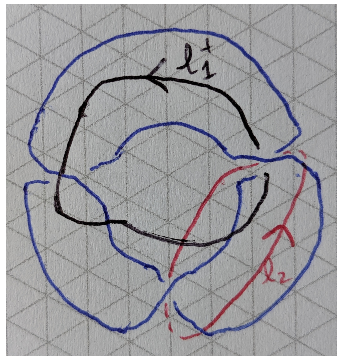
$$V(2,2) = lk(l_2, l_2^+) = 1$$

For the off-diagonal entries, we now compute the linking numbers of l_1 with l_2^+ and l_2 with l_1^+. For the $(1,2)$ entry, we have the following picture:



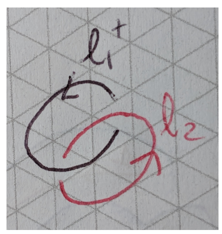
The loops are clearly disjoint, so we get the following

$$V(1,2) = lk(l_1, l_2^+) = 0$$



For the final entry, we have the following drawing of the loops:

Or without the Seifert surface in the way:



Hence we have

$$V(2,1) = \operatorname{lk}(l_2, l_1^+) = \frac{0-2}{2} = -1$$

Therefore our Seifert matrix is:

$$V = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

To compute the Alexander polynomial $A(t)$ from the Seifert matrix, we have the following formula:

$$A(t) = \det(V - tV^T)$$

Where V^T is the transpose matrix of V, i.e. $V^T(i,j) = V(j,i)$ for all $i, j = 1, \dots, 2g$. Again, this is up to some power of t.

Hence, in the example above, the Alexander polynomial for the trefoil is:

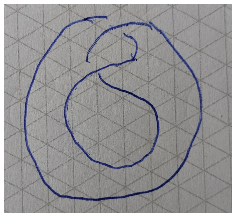
$$A(t) = \det\left(\begin{bmatrix} 1-t & t \\ -1 & 1-t \end{bmatrix}\right) = (1-t)^2 + t = 1 - t + t^2$$

as we expected.

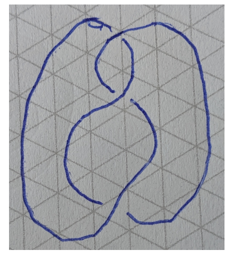
There are some checks that you can do to verify that the Alexander polynomial is indeed an invariant, by checkig certain properties of the determinant and the Seifert matrix, but for now let's just take it as a theorem.

Exercise 8. Using this method, compute the Alexander polynomial of the figure-eight knot.

Exercise 9. Using this method, compute the Alexander polynomial of the following family of knots K_n. Note that K_0 is the unknot and K_1 is the trefoil.



(A) K_0



(B) K_1

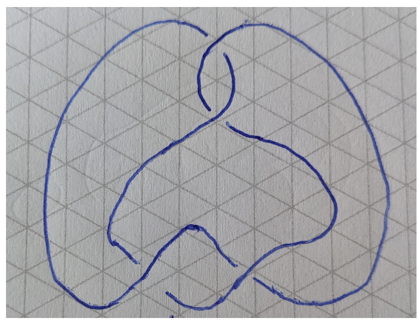


FIGURE 3. K_2

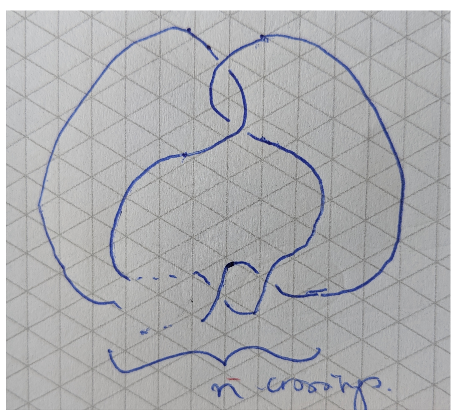


FIGURE 4. K_n

As a hint, you should obtain a genus 1 surface for each K_n and therefore only need two loops again.