Dehn surgeon school: Nathan's HW 1 Wednesday, July 17, 2019.

References and links for the second computational session, which demonstrated a wide variety of tools for studying Dehn surgery, hyperbolic geometry, Floer homology, foliations, character varieties, and Heegaard splittings, all of which can be used together via Python/SageMath via the computop/sage Docker image.

- http://snappy.computop.org
- http://www.sagemath.org
- http://bitbucket.org/t3m/sagedocker
- http://doi.org/10.7910/DVN/LCYXPO
- http://github.com/bzhan/bfh_python
- http://regina-normal.github.io

1. Get SageMath and SnapPy working together on your laptop, for example using the computop Docker image. Alternatively, from any of the physical ICERM terminals you can access it via http://icerm2.icerm.brown.edu:8888.

2. You can get a knot of 14 or fewer crossings in SnapPy by doing:

   ```
   knots = snappy.HTLinkExteriors(cusps=1)
   E = knots.random()
   ```

   Use verified computation as described here: http://snappy.computop.org/verify.html to prove it is hyperbolic and to compute its volume to a provably correct 250 decimal places. By Mostow rigidity this number is an invariant of the knot exterior and hence of the knot itself. (There are a handful of non-hyperbolic links in this range, so you're very unlike to pick one of them and so be unable to complete this problem!)

3. Look at the documentation for HTLinkExteriors by typing

   ```
   ?snappy.HTLinkExteriors
   ```

   to figure out how to pick a random **10 crossing** knot. Download the software of http://doi.org/10.7910/DVN/LCYXPO and use it to find coorientable taut/Reebless foliations on at least one Dehn surgery of your random knot.

4. Python programming practice:

   Use http://snappy.computop.org/spherogram.html to write a Python function to produce a link projection of the $(a_1, a_2, \ldots, a_k)$ pretzel link. For the $(-2,3,7)$ pretzel knot, write a procedure that searches for the two slopes of the two len space Dehn surgeries discovered by Fintushel-Stern. Use Regina to determine which lens spaces these are. Can you find lens space surgeries on other pretzel knots?

5. Look at the list of software that is part of the computop Docker image. See if you can compute something interesting with one of them.

6. The webpage http://computop.org lists a wide variety of computational tools in low-dimensional topology. Find one that is relevant to your own work and try to get it working in your Docker container.

Gordon. Exercises 3.

(1) Let $\mathcal{E}$ be a parallel family of positive edges in Γ, joining vertices $a \neq b$. Let I_a (resp. I_b) be the set of labels at the endpoints at a (resp. b) of the edges in $\mathcal{E}$. Show that if $I_a \cap I_b \neq \emptyset$ then $\mathcal{E}$ contains a Scharlemann cycle.

(2) Let Γ_1, Γ_2 be the intersection graphs coming from essential 2-spheres in $M(\alpha_1) \times M(\alpha_2)$. Show that if Γ_1 contains a parallel family of n_2 negative edges then Γ_2 contains a Scharlemann cycle. What about $(n_2 - 1)$ parallel negative edges?

(3) Let $(M; \alpha_1, \alpha_2)$ be an exceptional triple where $M(\alpha_1)$ is reducible and $M(\alpha_2)$ contains an

incompressible torus T (with the minimality assumptions of Lecture 2). Let σ be a $(1,2)$-Scharlemann cycle in Γ_1, and let $X \subset T$ be the union of the edges of σ and the flat vertices V_1 and V_2.

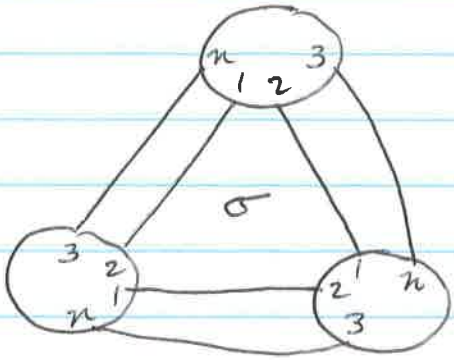
(i) Show that X does not lie in a disk in T.

(ii) Show that if σ has order 3 then X lies in an annulus in T.

(iii) What if σ has order > 3?

(4) Let Γ_1, Γ_2 be the intersection graphs coming from essential 2-spheres S_1, S_2 in $M(a_1), M(a_2)$ with the usual minimality assumptions. An <u>extended Scharlemann cycle</u> in Γ_1 (say) is a (say) $(1,2)$ - SC with digon faces adjacent

to each edge. Thus the bigons have corners $(n,1)$ and $(2,3)$; see figure. (Recall that $n = n_2 \geq 4$.)



Show that T_1 cannot contain an extended Scharlemann cycle.

Hint. Recall that the edges of σ cut $S_2 - V_1 \cup V_2$ into disks $D_1, \ldots, D_k$. Consider the disk D_i that contains V_n and V_3.