Topological Data Analysis **École Polytechnique**
INF556 **October 26, 2017**

TD 5 - Topological Persistence

Lucas Lugão Guimarães – lucas.lugao-guimaraes@polytechnique.edu
Alexandre Ribeiro João Macedo – alexandre.macedo@polytechnique.edu

Q1. The code of this implementation is attached in the final submission of this TD.

Q2. The reduction algorithm already have the complexity requested in question 3 where we give an explanation for this complexity.

Q3. The reduction algorithm can be seen bellow:

Algorithm 1 Reduction

```
1: let $R$ be a list of $m$ ordered sets representing the $\partial$ boundary matrix
2: let $P$ be an array of size $m$
3: $P \leftarrow [-1, \dots, -1]$
4: for $j \leftarrow 0$ to $m-1$ do
5:     while $R[j]$ is not empty do
6:         $pivot \leftarrow max(R[j])$        the max function is $O(1)$ since $R[j]$ represents a ordered set.
7:         if $pivot \geq 0$ then
8:             $R[j] \leftarrow R[j] \ominus R[pivot]$
9:         else
10:            break
11:        end if
12:    end while
13:    if $R[j]$ is empty then
14:        $pivots[max(R[j])] \leftarrow j$
15:    end if
16: end for
```

In this implementation we can see two explicit loops in the lines 4 and 5 and an implicit one in the symmetrical difference at line 8. The first one obviously finishes after m steps. The second one will continue to run while there is still ones in the j-th column to be eliminated, that means that it will run for at most m steps (usually this number is much smaller due to the matrix sparsity). The last loop is the symmetrical difference of two given columns, this operation takes $O(m)$ steps as well. Hence, the given algorithm takes $O(m^3)$ to finish as requested.

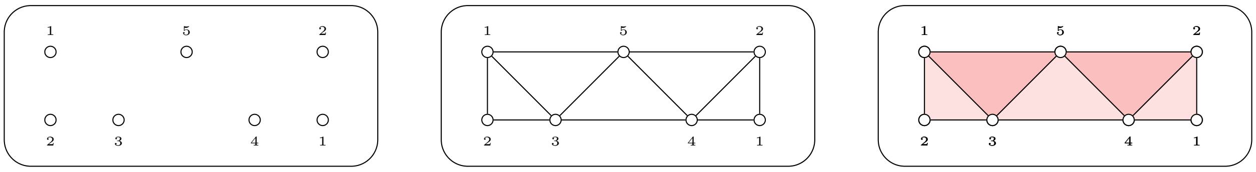
Q4. The function used to calculate the barcode is explained in the pseudocode below:

Algorithm 2 Barcode

```
1:  let $F$ be an array of simplices representing the filtration
2:  let $R$ be an array of $m$ ordered sets representing the reduced $\partial$ boundary matrix
3:  let $P$ be an array of size $m$
4:  let $B$ be a list representing the barcode
5:  $P \leftarrow [false, \ldots, false]$
6:  for $j \leftarrow 0$ to $m - 1$ do
7:      if $R[j]$ is not empty then
8:          $P[max(R[j])] \leftarrow true$
9:      end if
10: end for
11: for $j \leftarrow 0$ to $m - 1$ do
12:     if $R[j]$ is not empty then
13:         $birth \leftarrow max(R[j])$
14:         $death \leftarrow j$
15:         $B.add([F[birth].dim, F[birth].val, F[death].val])$
16:     else
17:         if not $P[j]$ then
18:             $B.add([F[j].dim, F[j].val, \infty])$
19:         end if
20:     end if
21: end for
```

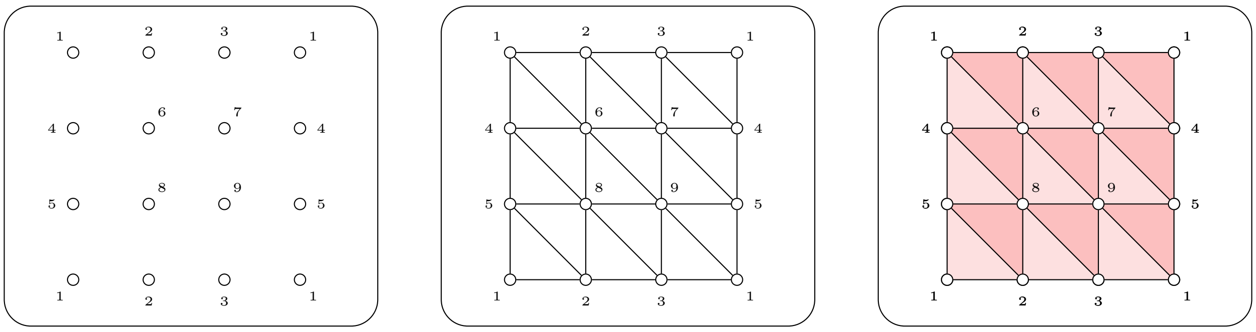
As in its implementation in C++ it has complexity $O(m)$ in time and space.

Q5. The figures below show one possible filtration for the Moebius band, the torus, the Klein bottle, the projective plane, the d-sphere and the d-ball. For the d-ball, we only show the triangulations for d ≤ 3 and for the d-sphere, we only show the triangulation for d ≤ 2 due to the complexity or impossibility of representing the drawing for higher values of d. In question 6, we show the bar codes for some of the filtrations.



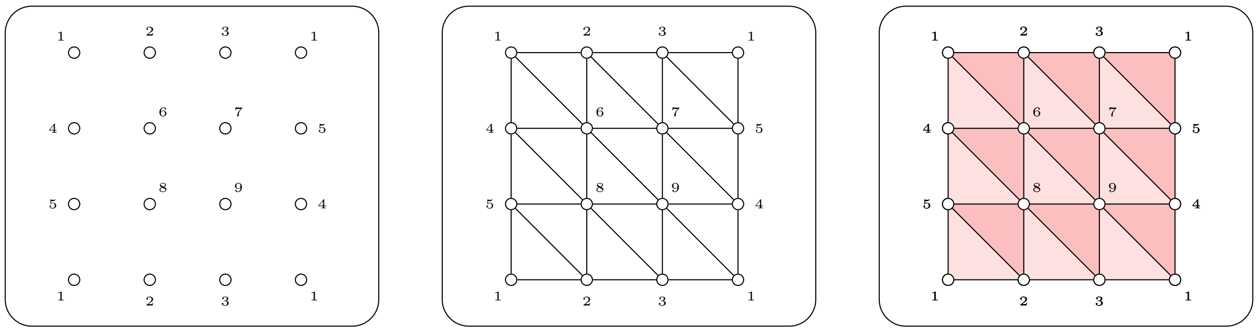
(a) 0-simplexes (b) 1-simplexes (c) 2-simplexes

Figure 1: One possible filtration for the Moebius band.



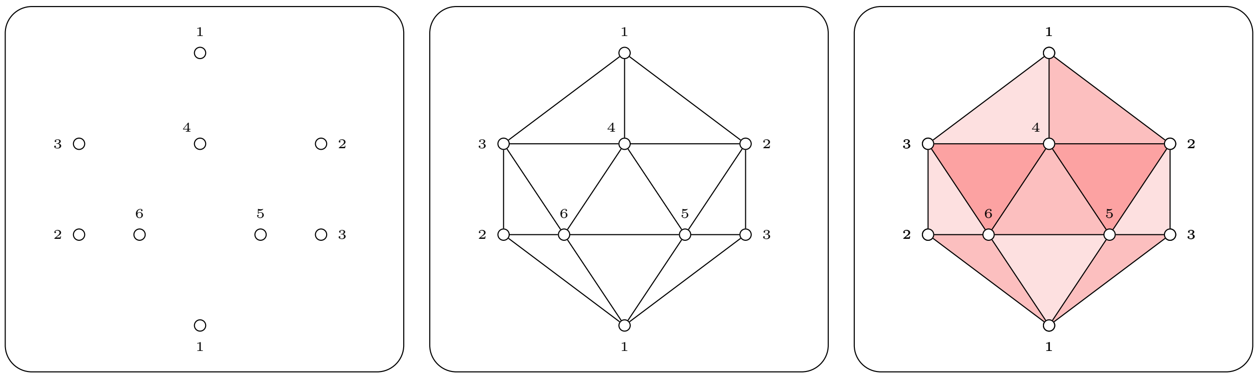
(a) 0-simplexes (b) 1-simplexes (c) 2-simplexes

Figure 2: One possible filtration for the torus.



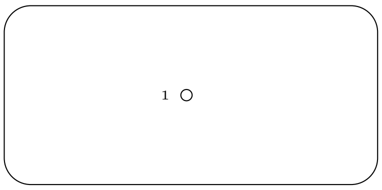
(a) 0-simplexes (b) 1-simplexes (c) 2-simplexes

Figure 3: One possible filtration for the Klein bottle.

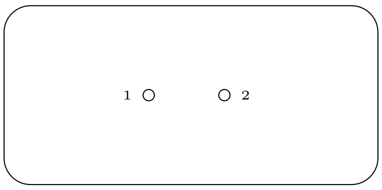


(a) 0-simplexes (b) 1-simplexes (c) 2-simplexes

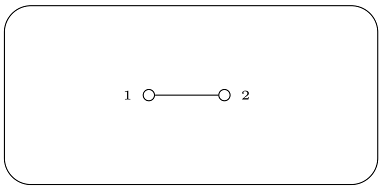
Figure 4: One possible filtration for the projective plane.



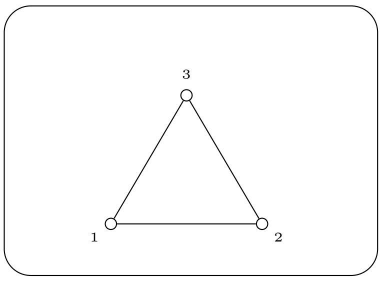
(a) 0-ball. 100



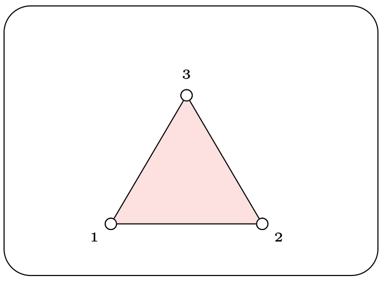
(b) 0-sphere. 200



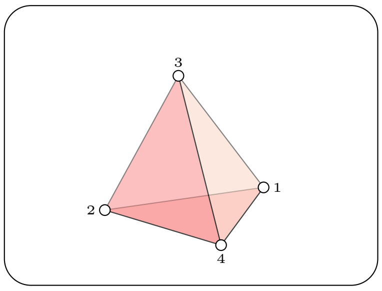
(c) 1-ball. 100



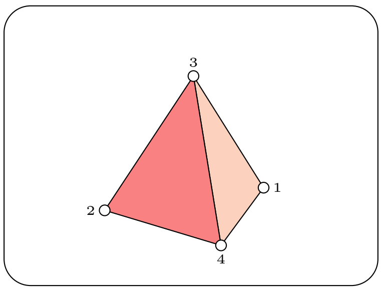
(d) 1-sphere. 110



(e) 2-ball. 100



(f) 2-sphere. 101



(g) 3-ball. 100

Figure 5: D-ball and d-sphere triangulations. The Betti numbers are given in the captions.

Q6. For the computed barcodes, we can validate our results. For example, for the torus we saw that the number of Betti was 121, as we spected it to be. For the d-balls and d-spheres, we saw that it started by creating the points and the final Betti number was for the ball 100...0 and for the sphere 100...1.

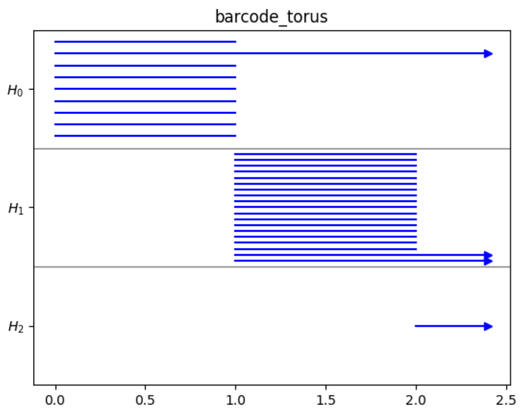


Figure 6: Barcode for the torus.

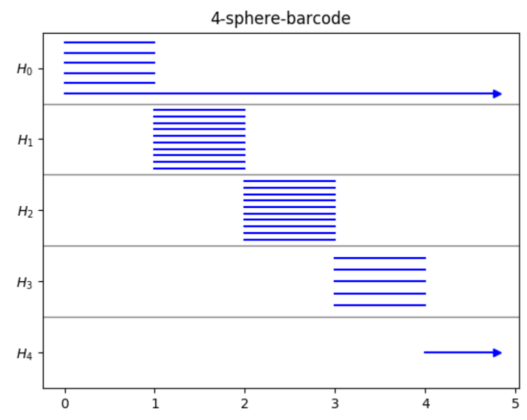
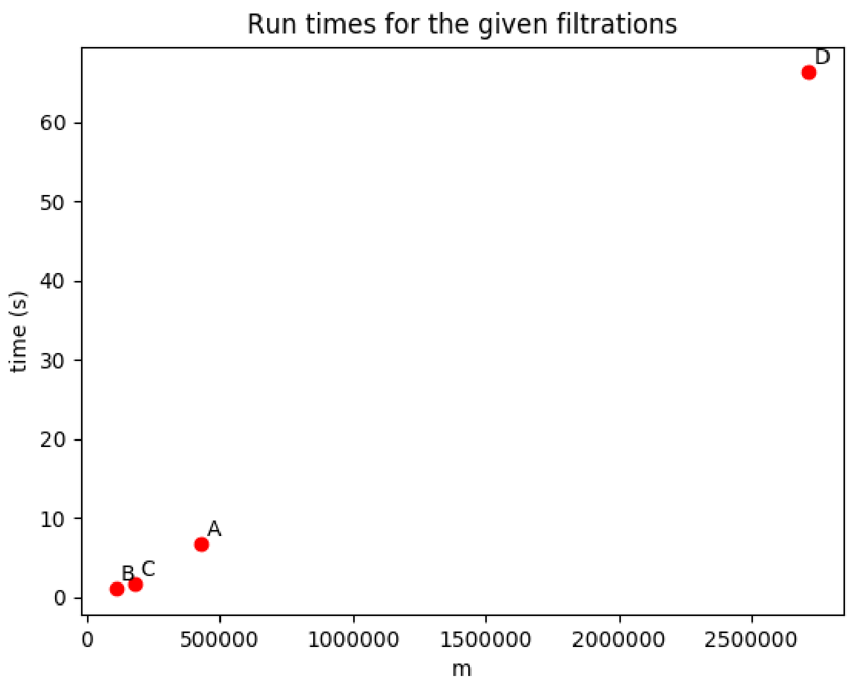


Figure 7: Barcode for the 4-sphere.

Q7.



As can be seen in the picture above, the timings of the algorithm for each data set increases in a sub $O(m^3)$ this can be explained due to the sparse matrix representation. Anyway, if a dense approach had been chosen the timings would reflect this time complexity.

$$m_A = 428643, \quad t_A = 6.732$$
$$m_B = 108161, \quad t_B = 1.033$$
$$m_C = 180347, \quad t_C = 1.736$$
$$m_D = 2716431, \quad t_D = 66.260$$

Q8.

A) The filtration A might show a 2-sphere. When the H_1 goes to zero H_2 goes to one. Then it keeps this H_2 for a while. When H_2 goes to zero, it means the filtration is "saturated".

B) For this filtration. We might think that it is composed by tetrahedra. As the number of H_2 goes down as the time passes as they are filled. The H_1s represents triangles.

C) The filtration B shows at times between 0 and 1 betti numbers (1,2,1) which corresponds possibly to a 2-torus or a klein bottle. Given that it then changes to a configuration with betti numbers (1,1,0), the one of a simple strip, it could indicate that in fact the original topological feature was of an torus. The noise in the filtration could indicate a Čech filtration of an point cloud generated from a 2-torus.

D) This filtration shows some similarities whit the Klein bottle and the torus. Therefore, we can think of it as a filtration for a triangulation for one of these two.

References

[1] Edelsbrunner, Herbert. *A Short Course in Computational Geometry and Topology.* Springer, 2014.

[2] *Simplical Homology.* Available here.

[3] *The Real Projective Plane Triangulated.* Available here.