Topological Data Analysis - TD5 (Midterm)

Alexis de La Fournière et Valentin Villecroze

18 octobre 2019

1 Explications et analyse de l'implémentation

1.1 Description du code

Courte description des différents fichiers créés (nous avons choisi de suivre un paradigme de programmation orientée objet) :

`main.py` Fichier à exécuter.

`simplex.py` Définit la classe `Simplex` avec un ordre permettant de trier les simplexes d'une filtration en respectant la contrainte d'une face arrivant toujours avant un simplexe dans lequel elle apparait.

`complex.py` Définit la classe `Complex` qui génère une liste de simplexes à partir du fichier d'une filtration.

`column.py` Définit l'objet `Column` qui est une représentation creuse d'une colonne (liste des indices non nuls) sur laquelle nous avons implémenté l'addition sur $\mathbb{Z}/2\mathbb{Z}$.

`matrix.py` Définit la classe `Matrix` sur laquelle nous avons implémenté l'algorithme de réduction à l'aide d'un pivot de Gauss, ainsi que la génération d'un code barre à partir de la matrice réduite. Pour accélérer la recherche de pivots lors de la réduction, nous gardons en mémoire pour chaque ligne l'indice de la colonne contenant le pivot s'il y en a un, et -1 sinon.

`utils.py` Implémente une recherche dichotomique dans une liste triée.

`plot.py` Permet d'afficher le code barre.

`data/` Contient les filtrations (celles des sphères et des boules sont dans `balls`).

`results/` Contient les codes barres calculés.

1.2 Création de la matrice

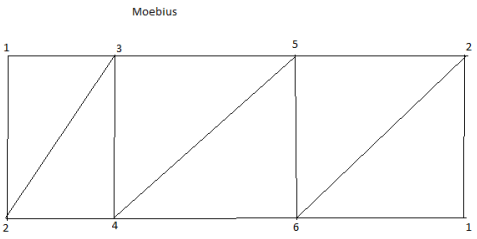
Afin de créer la *boundary matrix*, nous avons utilisé une table de hachage afin de trouver pour chaque simplex les indices de ses faces dans la liste triée selon l'ordre imposé par la filtration. Cet étape s'effectue ainsi en complexité linéaire (selon le nombre de simplexes) et il suffit ensuite de créer la matrice en parcourant la liste des simplexes et en créant les colonnes adéquates, ce qui se fait donc en complexité $O(dn)$ (avec d la dimension maximale des simplexes) ce qui est donc une complexité linéaire dans la plupart des cas étudiés.

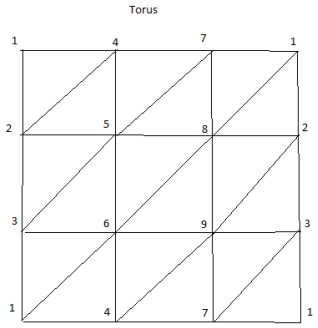
1.3 Complexité de la réduction

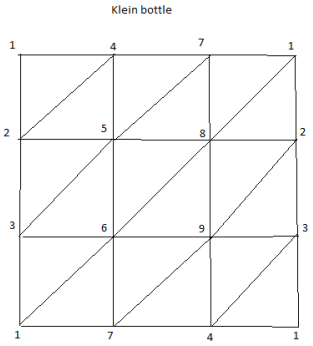
La réduction de la matrice est effectivement en $O(m^3)$. En effet, on itère sur chaque colonne et sur chacune, on itère sur le plus haut coefficient non nul, ce qui donne au plus m opérations. Enfin, à chacun de ces $O(m^2)$ tours de boucle, on effectue une addition, qui prend au plus $O(m)$ opération. Notons, que cette dernière approximation est très grossière. Chaque colonne contient au départ $d+1$ coefficients non nul (où d est la dimension du simplexe codé par la colonne), et l'on peut considérer que nombre ne va pas beaucoup augmenter tout au long des opérations. Or la représentation creuse fait que le nombre d'opérations effectuées lors d'une addition entre colonnes est au plus égale à la somme du nombre d'indices non nuls. On voit donc que l'on peut s'attendre à une complexité plus proche d'être quadratique que cubique.

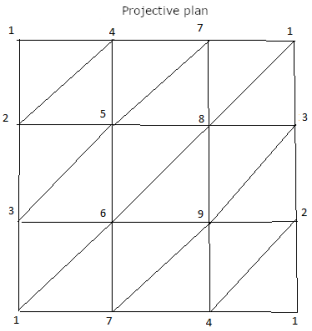
2 Filtrations des formes usuelles

Les filtrations choisies pour les formes sont les suivantes :









Pour chacune, nous avons pris la convention de mettre la valeur égale à la dimension, ce qui assure la consistance de la filtration. Les fichiers `.txt` sont disponibles dans le dossier `data`. Pour les sphères et les boules, celles-ci ont été générées grâce au scipt `ballgen.py` également présent dans ce dossier.

On retrouve les nombres attendus pour les exemples connus que sont les sphères, les boules et le tore.

3 Analyse sur les données fournies

3.1 Performance

filtration A

Création de la matrice : 43 secondes

Réduction de la matrice : 18 minutes 55 secondes

filtration B

Création de la matrice : 10 secondes

Réduction de la matrice : 50 secondes

filtration C
Création de la matrice : 20 secondes
Réduction de la matrice : 2 minutes 3 secondes

filtration D
Création de la matrice : 51 secondes
Réduction de la matrice : 50 secondes

3.2 Analyse

filtration A On peut imaginer que la structure originelle est celle d'une sphère. Il faut un certain temps pour que tous les points fusionnent ensemble, d'où l'apparition de nombreuses composantes connexes et de petits cycles. Une fois tous les points liés et les faces apparues, on peut observer la structure de sphère par la large barre sur H2. Enfin, la boule finit par se remplir, d'où la fin de la barre.

filtration B On voit plusieurs phénomènes : d'abord 8 trous d'ordre 2 faisant penser à 8 petites sphères, puis l'apparition de 5 trous de dimension 1 et enfin, au moment où ils disparaissent, l'apparition d'un trou de dimension 2. Cela fait penser à la structure que l'on obtiendrait en disposant 8 sphères dont les centres seraient les sommets d'un cube et qui auraient chacune un point en commun avec chacune de ses voisines. Il y aurait donc un trou par face, mais le sixième est linéairement dépendant des 5 autres, d'où l'apparition de 5 barres. Au moment où ces 5 cycles disparaissent, cela signifie que les faces sont maintenant pleines. Mais au centre du cube, un espace n'est pas encore atteint, d'où l'apparition d'un trou d'ordre 2. La forme sous-jacente est donc celle de 8 sphères placées au sommet d'un cube de côté égal au diamètre des sphères (le cube ne sert qu'à se représenter la disposition, il n'est pas réellement tracé).

filtration C et D Les filtrations C et D ont toutes les deux des diagrammes similaires, avec deux principaux trous d'ordre 1 et d'ordre 2. On reconnait là la structure d'un tore. De plus, le fait qu'un trou disparait en même temps que le vide de dimension 2 correspond au moment où le tore devient plein pour prendre une structure "d'anneau plein" (de cercle). La différence entre les deux provient du fait que le C est beaucoup plus fin que le D.