Computational Topology (Spring 2018) — Homework 4

- You **must email your submission** as a **PDF file** to bkrishna@math.wsu.edu. You are welcome to write answers by hand, and scan or take photos of the writings. Put all the images on a PDF file, though.
- Your file name should identify you. For instance, if you are Napolean Dynamite, you should name your submission NapoleanDynamite_Hw4.pdf. **Please start your name in this format. If you want to add more bits to the title, e.g., Math574, you could name it NapoleanDynamite_Math574_Hw4.pdf, for instance. Also, please avoid white spaces in the file name :-).**
- **This homework is due before midnight on Thursday, March 1**.

1. (25) Let the simplicial complex K consist of a d-simplex σ and its faces. Justify your answers to both the following questions.
   (a) How many d-simplices does $\operatorname{Sd} K$, the barycentric subdivision of K, have?
   (b) What is the Euler characteristic $\chi(K)$?
2. (20) Let $S = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ be a set of points in $\mathbb{R}^d$. The *furthest point Voronoi cell* of a point $\mathbf{v}_j \in S$ is defined as
$$F_{\mathbf{v}_j} = \{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x} - \mathbf{v}_j\| \geq \|\mathbf{x} - \mathbf{v}_i\| \ \ \forall\, \mathbf{v}_i \in S\}.$$
   The collection of $F_{\mathbf{v}_j}$ for all $\mathbf{v}_j \in S$ is called the *furthest point Voronoi diagram* of S. Draw the furthest point Voronoi diagram of a set S with at least *eight* points in $\mathbb{R}^2$. On the same diagram, also draw the nerve of this furthest point Voronoi diagram, i.e., the furthest point Delaunay triangulation of S.
3. (30) Let S be a finite set of points in $\mathbb{R}^d$, and let $\text{Alpha}(r), \check{\text{C}}\text{ech}(r)$, and Del denote the alpha, Čech, and Delaunay complexes of S, with the first two complexes defined for radius r of balls centered at each $\mathbf{v}_j \in S$. Either prove each of the following two subset relationships, or give counterexamples violating them.
   (a) $\text{Alpha}(r) \ \subseteq \ \check{\text{C}}\text{ech}(r) \cap \text{Del}$.
   (b) $\check{\text{C}}\text{ech}(r) \cap \text{Del} \ \subseteq \ \text{Alpha}(r)$.
4. (40) Create a dataset S of *at least* ten points in $\mathbb{R}^2$ or $\mathbb{R}^3$. You should be able to perform *both* the following computations on the chosen dataset.
   (a) Pick a appropriate radius r at which $\check{\text{C}}\text{ech}_S(r) \subset \text{VR}_S(r)$, i.e., the Čech complex is a strict subcomplex of the Vietoris-Rips complex. Produce a visualization showing both complexes **simultaneously**. One option is to use TetView, and set a different *attribute* for those extra triangles that are present only in $\text{VR}_S(r)$ (the attribute values are given in the last (fifth column) in a face file).
   (b) Choose an appropriate radius at which at least *two* tetrahedra are included in $\text{VR}_S(r)$. Produce visualization(s) of both the Čech and Vietoris-Rips complexes at this radius, using TetView or another software package.

5. (50) This exercise asks you to create a filtration of a Delaunay complex. You could use Octave to do most of the computations. The visualizations could be generated in *TetView* (or in Octave itself).

   (a) Let the set S contain the eight vertices of the unit cube in $\mathbb{R}^3$ sitting in the nonnegative orthant, along with ten (10) extra points lying in the interior of the cube (you could choose these points in any way you want—just describe how you locate them). Find a Delaunay tetrahedralization of S – you could use the function `delaunay` in Octave, or another similar program (TetGen, for instance). Denote this complex by Del_S.

   (b) Rather than finding alpha complexes at various radii, we will use pairwise distance cutoffs similar to those used in defining Vietoris-Rips complexes. For a given radius r, define the Beta complex of S as a subcomplex of its Delaunay complex as follows.

   $$\mathrm{Beta}_S(r) = \{\sigma \in \mathrm{Del}_S \mid \mathrm{diam}(\sigma) \leq 2r\}.$$

   Write some scripts in Octave (or another language) which can run through the list of simplices in Del_S to identify $\mathrm{Beta}_S(r)$. To carry out these computations efficiently, notice that you need to compute the lengths of all edges in Del_S *only once*, and do comparisons with $2r$ as needed.

   (c) Generate $\mathrm{Beta}_S(r)$ for $r = 1/2,\ 1/\sqrt{2}$, and $\sqrt{3}/2$. Create visualizations showing the three complexes using *TetView* or a similar program. You are welcome to try Octave for this task, but TetView might prove easier. Notice that there is an option to display just the outline of a 2- or 3-complex in TetView.

   Describe each $\mathrm{Beta}_S(r)$ complex, and include snapshot(s) of each complex.