WXML Final Report: Nerves of Simplicial Complexes

Maddy Brown, Spencer Kraisler, Zhongyan Wang

Spring 2019

Abstract

Motivated by the fact that the nerve of a simplicial complex is easier and simpler to work with but also shares some of the properties of the original simplicial complex, we explored many things about the nerve in this quarter, including the computations of the nerve, power of nerve, and higher nerves; proof of the existence of higher cycles of nerve; and the difference between nerve and higher nerve.

1 Introduction

We will review some definitions which are vital to our research.

A **simplex** is a generalization of the triangle, but for any arbitrary dimension. You are already familiar with the 2-simplex (triangle) and the 3-simplex (tetrahedron).

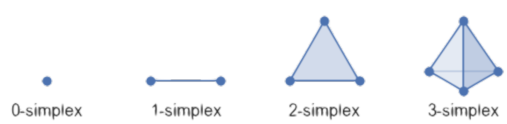
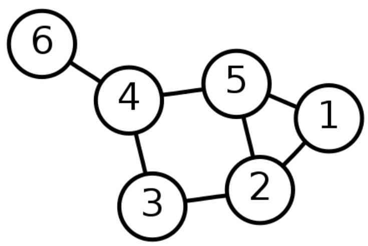


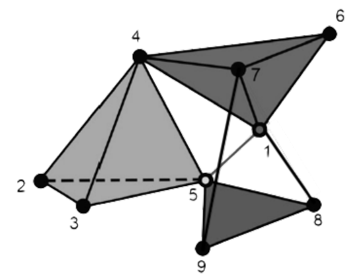
Figure 1: Simplices

A **graph** is a set of vertices and edges. It is helpful to think of graphs as instead sets of 0-simplices and 1-simplices. Simple graphs (i.e., those without loops or multiple edges) may be uniquely characterized as a collection of sets of integers. For instance, Figure 2(a) can be characterized as

$$\Delta = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{1,2\}, \{1,5\}, \{2,5\}, \{2,3\}, \{3,4\}, \{4,5\}, \{4,6\}\}.$$



(a) A graph



(b) A simplicial complex

Figure 2: Examples of simplicial complexes

A **simplicial complex** is a generalization of a graph, as it allows higher dimensional simplicies rather than just 0- and 1-simplices. Simplicial complexes can also be characterized as a collection of sets of integers. For instance, a triangle with vertices $\{1\}, \{2\}, \{3\}$ can be represented as $\Delta = \{\{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$. We call a single set of integers a **face** of the simplicial complex. In Figure 2(b), $\{2,3,4,5\}$ and $\{2,3,5\}$ are examples of faces.

A **facet** is a face that is not contained in any other face. For example, in Figure 2(b), $\{1,2,3,4\}$, $\{3,5\}$, and $\{5,6,7\}$ are facets while $\{2,3,4\}$ and $\{1,7\}$ are not.

A simplicial complex can more conveniently be uniquely characterized as a set of facets. For example, Figure 2(b) can be characterized as

$$\Delta = \langle \{2,3,4,5\}, \{1,4,6,7\}, \{1,5\}, \{1,8\}, \{5,8,9\} \rangle,$$

where we just list the sets representing the facets. Rigorously, a simplicial complex is a collection of sets of integers Δ where for any face $\sigma \in \Delta$ with $\tau \subseteq \sigma$, it follows $\tau \in \Delta$.

An operation we can perform on simplicial complexes that we were particularly interested in this quarter was taking the **nerve**. If we have the set of F_i's representing the facets of Δ, then the nerve of Δ is the following:

$$N(\Delta) = \{\{F_i\} : \left|\bigcap F_i\right| \neq \emptyset\}.$$

That is, the nerve is the collection of sets of facets which intersect at at least one point. In general, the nerve of a simplicial complex is simpler and easier to work with than the original collection of sets. However, it shares many of the same properties.

We can construct the nerve of a simplicial complex more or less by looking at the its graphical representation. When drawing the nerve, we draw a vertex for each of the facets in the original complex. Then, we draw lines (or create filled-in triangles or tetrahedron, etc.) between the vertices representing the facets which are connected. This process is displayed in Figure 3.

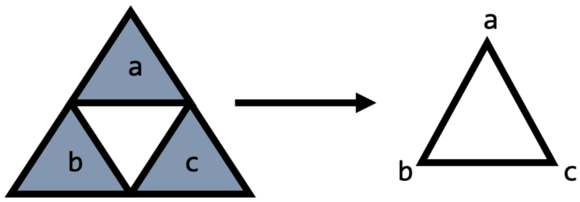


Figure 3: Δ and $N(\Delta)$

2 Past Results

This section details some already known properties of simplicial complexes and nerves which interest us and inspired some of our conjectures.

Betti Numbers: Informally, the k^{th} Betti number counts the number of k-dimensional holes on a topological surface.

- β_0 is the number of connected components.
- β_1 is the number of one-dimensional "circular" holes.
- β_2 is the number of two-dimensional "cavities."

When we take the nerve of a simplicial complex, all of the Betti number values are preserved, due to a well-known theorem of Borsuk.

Theorem 1 (Borsuk's Nerve Theorem, [2]) *The complexes Δ and $N(\Delta)$ are homotopy equivalent. In particular, they have the same Betti numbers.*

A more recent development is the following, due to Barmak and Minian.

Theorem 2 [1] *For a simplicial complex Δ, if there exists some $k \in \mathbb{N}$ where $N^k(\Delta) = \langle 1 \rangle$, then Δ is collapsible. If Δ is collapsible, then its Betti numbers are all zero.*

We will not give precise definitions for homotopy equivalence and collapsibility here; they can found in most algebraic topology textbooks (see, for example, Hatcher's *Algebraic Topology*). We will define $N^k(\Delta)$ in the next section.

3 Our Research

Over the course of this quarter, we spent most of our time exploring two ideas related to nerves: powers of nerves and higher nerves. In the former, we take the nerve of a complex several times. In the latter, we take the nerve of complexes in a slightly modified way that reveals different information about the original complex.

3.1 Powers of Nerves

We call applying the nerve transformation several times as taking the "power" of the nerve. Taking the nerve of a simplicial complex Δ twice can be written as

$$N(N(\Delta)) = N^2(\Delta).$$

Taking the nerve of Δ k times is subsequently written as

$$N^k(\Delta) = N(N^{k-1}(\Delta)).$$

We refer to the above as the "k-th power" of the nerve of Δ.

We used our Python code (detailed in the appendix) to take nerve powers of randomly-generated simplicial complexes. We noticed this resulted in two behaviors. Some complexes, such as Figure 4, will converge to a particular complex. Continuing to take the nerve again will result in the same complex, or $N^k(\Delta) = N^{k-1}(\Delta)$. Other complexes, like Figure 5, will eventually toggle between two complexes, or $N^k(\Delta) = N^{k-2}(\Delta) \neq N^{k-1}(\Delta)$.

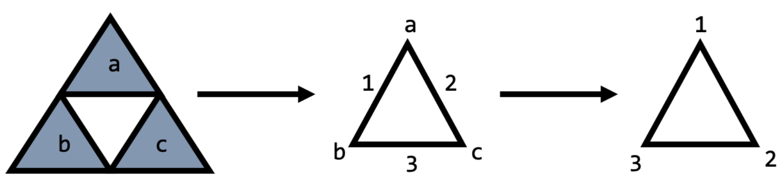


Figure 4: Δ, $N(\Delta)$, and $N^2(\Delta)$ for a simplicial complex Δ which converges to one complex.

We wondered if there were any complexes that converged to a cycle of three or more complexes. After running many randomly-generated complexes, our experimental evidence suggested there were not any complexes that entered a 3-cycle or higher. We have an outline for a possible proof in the works.

3.2 Higher Nerves

Something else we spent a lot of time thinking about was the concept of "higher nerves." When we take higher nerves of complexes, instead of looking for any

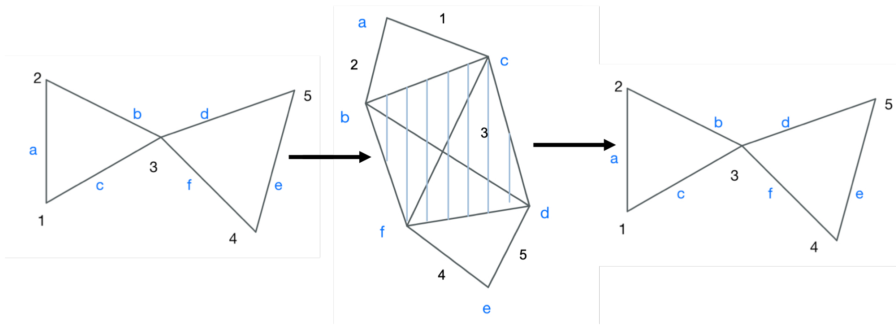


Figure 5: Δ, $N(\Delta)$, and $N^2(\Delta)$ for a simplicial complex Δ which toggles between two complexes after taking the nerve several times.

intersection between facets, we look for at least k intersections between facets. This reveals different information about the original complex than we get by just taking the nerve.

The "k-th nerve" of a simplicial complex is formally defined by the following:

$$N_k(\Delta) = \{\{F_i\} : \left|\bigcap F_i\right| \geq k\}.$$

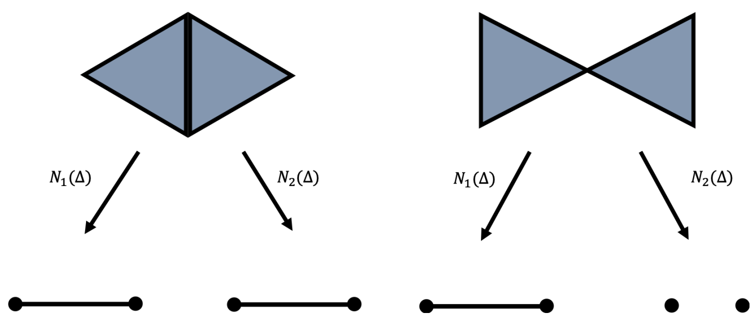


Figure 6: Two simplicial complexes with identical first nerves but differing second nerves.

Unlike the original nerve, higher nerves do not preserve holes or Betti numbers. We conjectured that taking the higher nerve repeatedly of any simplicial complex will eventually result in the empty set. One reason this makes sense is because the Betti numbers (number of holes) are not preserved, so nothing has to "stay" in the complex when we take its higher nerve. We do not have a proof for this conjecture yet, but we have strong experimental evidence that it is true.

4 Conclusion

We explored simplicial complexes and what it means to take their nerve. We have created multiple conjectures regarding the repeated computation of nerves. Future goals of this project would be to revise our proof on the non-existence of n-cycles for powers of nerves and to show that our conjecture about powers of higher nerves is true, or if its not, to observe something about complexes which have higher nerves that do not go to the empty set.

5 Appendix

Below is Python code we wrote to manipulate simplicial complexes, and compute nerves, the power of nerves, and higher nerves in an efficient manner. The code represents simplicial complexes as sets of integers (which are facets), as any simplicial complex can conveniently be characterized uniquely as a set of facets. There is also error checking in case someone enters a set of sets of integers that are not facets. Regarding the nerve transformation, we were able to create a fast method for computing the nerve. Using the original definition, time complexity was $O(2^n)$, but our new algorithm brought it down to $O(n^2)$, where n is the number of vertices of the simplicial complex.

```
class SComplex:

    def __init__(self, facets):
        if not self.facetsValueCheck(facets):
            raise ValueError("Facets have incorrect form")

        self.facets = sorted(facets, key=len)
        self.dim = 0
        self.vertices = set()
        for facet in self.facets:
            for i in facet:
                self.vertices.add(i)
            if len(facet) > self.dim:
                self.dim = len(facet)
        self.vertices = list(self.vertices)

        max_vertex_index = 1
        for facet in self.facets:
            for v in facet:
                if v > max_vertex_index:
                    max_vertex_index = v
        if max_vertex_index > len(self.vertices):
            vert_dict = {}
            n = 1
            for v in self.vertices:
```

```
                vert_dict[v] = n
                n += 1
            reform_facets = []
            for facet in self.facets:
                reform_facet = []
                for v in facet:
                    reform_facet.append(vert_dict[v])
                reform_facets.append(reform_facet)
            self.facets = reform_facets

    def __getitem__(self, idx):
        return self.facets[idx - 1]

    def __len__(self):
        return len(self.facets)

    def __str__(self):
        return str(self.facets)

    def print(self):
        print(self)

    # returns true if alpha is a strict subset of A
    def is_strict_subset(self, alpha, A):
        if alpha is A:
            return False
        for a in alpha:
            if a not in A:
                return False
        return True

    # returns true if alpha is not a strict subset of any set in A
    def is_maximal(self, alpha, A):
        if len(alpha) == 0 or len(A) == 0:
            return False
        for a in A:
            if self.is_strict_subset(alpha, a):
                return False
        return True

    # returns true if every given facet is not a subset of another facet
    def facetsValueCheck(self, facets):
        size = len(facets)
        for facet in facets:
            if not self.is_maximal(facet, facets):
                return False
```

```
        return True

    def get_powerset(self, A, k=None):
        powerset = []
        size = len(A)
        masks = [1 << i for i in range(size)]
        for i in range(1 << size):
            powerset.append([aa for mask, aa in zip(masks, A) if i & mask])
        if k is not None:
            _powerset = [p for p in powerset if len(p) == k]
            powerset = _powerset
        return powerset

    def get_maximal_sets(self, A):
        maximal_sets = [a for a in A if self.is_maximal(a, A)]
        return maximal_sets

    def higher_nerve(self, k):
        V = self.get_powerset(self.vertices, k)
        nerve = [[] for _ in range(len(V))]
        for i in range(1, len(self) + 1):
            facet = self[i]
            F = self.get_powerset(facet, k)
            for f in F:
                index = V.index(f)
                nerve[index].append(i)
        __nerve = []
        for face in nerve:
            if face not in __nerve:
                __nerve.append(face)
        nerve = self.get_maximal_sets(__nerve)
        nerve = SComplex(nerve)
        return nerve

    def nerve(self):
        nerve = [[] for _ in range(len(self.vertices))]
        for i in range(1, len(self) + 1): # indexing facets (1 to # of facets)
            for j in self[i]:
                nerve[j - 1].append(i)

        # removes non-unique elements
        __nerve = []
        for face in nerve:
            if face not in __nerve:
                __nerve.append(face)
```

```
# removes non−maximal faces
nerve = self.get_maximal_sets(___nerve)
return SComplex(nerve)
```

References

[1] Barmak, J.A. and Minian, E.G. *Strong Homotopy Types, Nerves and Collapses.* Discrete Comput Geom (2012), 47: 301.

[2] Borsuk, Karol. *On the imbedding of systems of compacta in simplicial complexes.* Fundamenta Mathematicae **35** (1948), no. 1, 217–234.

[3] McInnes L, Healy J, *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*, 2018 URL: https://umap-learn.readthedocs.io/en/latest/index.html