

Chapter 1

Simplicial Complex class Design Document

1.1 数学概念

该节所有定义出自 Rotman(1988).

1.1.1 simplex

Definition 1.1. Let $\{p_0, p_1, \dots, p_m\}$ be an affine independent subset of \mathbb{R}^n . The convex set spanned by this set, denoted by $[p_0, p_1, \dots, p_m]$, is called the (affine) m -simplex with vertices p_0, p_1, \dots, p_m .

Definition 1.2. Let X be a topological space. A (singular) n -simplex in X is a continuous map $\sigma : \Delta^n \rightarrow X$, where Δ^n is the standard n -simplex.

1.1.2 simplicial complex

Definition 1.3. If s is a simplex, then a face of s is a simplex s' with $\text{Vert}(s') \subset \text{Vert}(s)$.

Definition 1.4. A finite simplicial complex K is a finite collection of simplexes in some euclidean space such that:

- (i) if $s \in K$, then every face of s also belongs to K ;
- (ii) if $s, t \in K$, then $s \cap t$ is either empty or a common face of s and of t .

Definition 1.5. If K is a simplicial complex, define its dimension, denoted by $\dim K$, to be

$$\dim K = \sup_{s \in K} \{\dim s\}$$

(of course, a q -simplex has dimension q).

Definition 1.6. Let K be a simplicial complex and let $p \in \text{Vert}(K)$. Then the star of p , denoted by $\text{st}(p)$, is defined by

$$\text{st}(p) = \bigcup_{\substack{s \in K \\ p \in \text{Vert}(s)}} s^\circ \subset |K|.$$

1.2 数据结构

1.2.1 Simplex

功能

1. SimplicialComplex 的基础成员.
2. 记录组成 simplex 的所有 vertices.
(vertex 通过 unsigned int 唯一表示).

数据成员

1. vector<unsigned int> vertices;
构成 Simplex 的 vertex 集合, size 等于 simplex 的 dimension + 1.
size 等于 1 时 Simplex 就是 vertex.
(对 vertices 排序, 便于对 Simplex 排序和哈希, 用于容器保存.)

函数成员

1. int getNSim() const;
输出: simplex 的 dimension.

1.2.2 SimplicialComplex

功能

1. 包含组成 SimplicialComplex 的所有 simplexes.
2. 记录并提供在当前 simplicial complex 中所有 vertices 的相邻信息.
3. 添加和删减 simplicial complex 中的 simplex 并保持 simplicial complex 的性质.

数据成员

1. vector<set<Simplex>> simplexes;
simplicial complex 包含的所有 simplexes, simplexes[i] 是 i 阶 simplex 的集合.
(添加和删减 simplicial complex 中的 simplex 时需要保证 simplex 的唯一性, 所以采用 set 保存.)
2. unordered_map<unsigned int, set<typename set<Simplex>::iterator>>
mVertex2Simplex;
冗余数据, 记录包含对应 vertex 的所有 Simplex 在数据成员 simplexes 中的迭代器.
(在计算 vertex 的 star, link 和删除 simplex 时, 添加该冗余数据能提高搜索效率.)
(在 erase 中需要计算和删除一些 simplex 集合的公共部分, 使用 set 存储第二项能优化这一过程.)

函数成员

1. int starClosure(unsigned int p, SimplicialComplex& Closure) const;
输入: vertex p .
输出: $st(p)$ 的闭包.
实现方式: 检索 mVertex2Simplex 得到所有包含 p 的 Simplex 集合, 插入这些 Simplexes.

2. int link(unsigned int p, unordered_set<unsigned int>& res) const;

输入: vertex p .

输出: 所有与 p 相邻的 vertex 集合.

实现方式: 插入 mVertex2Simplex 中包含 p 的 Simplexes 的 vertices.

3. int insert(const Simplex& s);

输入: simplex s .

输出: 将 s 的所有的 face 和自身插入到 simplicial complex 中, 更新 mVertex2Simplex.

实现方式: 生成 s 所有的 face, 根据维数插入到 simplexes.

4. int erase(const Simplex& s);

输入: simplex s .

输出: 将 simplicial complex 中所有以 s 为 face 的 simplex 和 s 移除, 更新 mVertex2Simplex.

实现方式: mVertex2Simplex 检索包含 s 各个 vertex 的 Simplex 集合, 被这些集合同时包含的 Simplex 是以 s 为 face 的 simplex(需要删减).

5. int getNSim() const;

输出: simplicial complex 的 dimension.

1.2.3 YinSet

模板: template<int Dim, int Order>

Dim 表示空间维数.(目前仅考虑 $\text{Dim} = 2$)

Order 表示拟合阶数.

功能

1. 由于 YinSet 组成的 Yin 空间和 YinSet 的表示空间 Jordan 空间的同构性, 记录 YinSet 边界来表示 YinSet.

新增功能

1. 二维情况, 记录 YinSet 边界上一个 0-simplicial complex kinks.

2. 将 simplicial complex 中的 vertex 单射映射到 YinSet 边界上的点.

数据成员

1. vector<Curve<Dim, Order>> rzable;

二维 YinSet 的边界 Curve 集合.

新增数据成员

1. SimplicialComplex kinks;

记录二维 YinSet 边界上的一些特殊点.

2. unordered_map<unsigned int, pair<unsigned int, unsigned int>> mVertex2Point;

将 vertex p 映射到 YinSet 边界里第 v_0 条 Curve 的第 v_1 个 knot 点上.

(v_0, v_1 依次是 mVertex2Point 第二项中的第一和第二个元素.)

3. `map<rVec, unsigned int> mPoint2Vertex;`
将 Curve 上的部分 knots 映射到 vertices 中.

新增函数成员

1. `const SimplicialComplex& getKinks() const;`
输出: 数据成员 kinks 的引用.
(直接修改 kinks 可能会导致 YinSet 内 vertex 和 point 无法一一对应, 输出 const 引用读取同时禁止修改)
2. `int vertex2Point(unsigned int vertex, pair<unsigned int, unsigned int>& index) const;`
`int vertex2Point(unsigned int vertex, rVec& point) const;`
`int point2Vertex(const pair<unsigned int, unsigned int>& index, unsigned int& vertex) const;`
`int point2Vertex(const rVec& point, unsigned int& vertex) const;`
输入: 检索的 vertex 或 point 在 rzable 的 index 或坐标.
输出: 返回值 0, 1 表示是否成功, 引用输出结果.
3. `int kinksInsert(pair<unsigned int, unsigned int>& index)`
输入: rzable 中的下标.
输出: 返回值 0, 1 表示输入是否合法, 插入 rzable 上下标对应的点到 kinks 中.
4. `int kinksErase(unsigned int vertex)`
输入: vertex 身份标识.
输出: 返回值 0, 1 表示输入是否合法, 在 kinks 中删除对应 vertex.