Extended Abstract: A Description of the JavaPlex Software Package

Author[1,*] and Author[2]

[1]Affiliation
[2]Affiliation

January 20, 2011

Abstract

In this document, we describe the design and goals of the JavaPlex software package. This program can be summarized in two main capabilities: the construction of filtered chain complexes of vector spaces, and the computation of their persistent homology.

1 Algebraic Background on Persistence Modules

// Maybe the best person to write this would be Mikael or Gunnar??
// need to mention grading by filtration index vs grading by dimension

2 Previous Work

The JavaPlex package is the fourth version in the Plex family. These programs have been developed over the past decade by members of the computational topology research group at Stanford University. Each successive version incorporated the results of new advances in the relatively quickly developing fields of computational topology and topological data analysis.

Like JavaPlex, its predecessor jPlex was also written in the Java language. However, it differed in that the main goal of jPlex was the computation of *simplicial* homology. Recent research topics in topological data analysis have required practitioners to move beyond conventional simplicial homology to more general scenarios.

3 Design Goals

- **Support for new directions for research** The main goal of the JavaPlex package is to provide an extensible base to support new avenues for research in computational homology. While its predecesor jPlex was very well suited towards computing simplicial homology, its design made extention difficult.
- **Interoperability** JavaPlex can be run either as a Java application, or it can be called from Matlab in jar form. Future possibilities include providing scripting interfaces in bsh or jython.
- **Adherence to generally accepted software engineering practices**

*Contents of footnote

4 Capabilities

4.1 Filtered Complex Generation

// talk about vietoris-rips, witness complexes // make sure to reference recent paper "fast construction of vietoris rips complex"

JavaPlex supports the consruction of two main types of filtered simplicial complexes. To begin, we are given a finite metric space $(\mathcal{X}, d)$. In practice, it is possible that $\mathcal{X}$ is a set of points in Euclidean space, although this is not necessary.

4.1.1 The Vietoris-Rips Construction

We define the filtered complex $VR(\mathcal{X}, r)$ as follows. Suppose that the points of $\mathcal{X}$ are $\{v_1, ...v_n\}$

- **Add points:** For all points $x \in \mathcal{X}$, $x \in VR(\mathcal{X}, 0)$
- **Add 1-skeleton:** The 1-simplex $[v_i, v_j]$ is in $VR_1(\mathcal{X}, r)$ iff $d(v_i, v_j) \leq r$
- **Inductive expansion:** We define $VR(\mathcal{X}, r)$ to be the maximal simplicial complex containing $VR_1(\mathcal{X}, r)$. That is, a simplex $[v_0, ..v_k]$ is in $VR(\mathcal{X}, r)$ if and only if all of its edges are in $VR_1(\mathcal{X}, r)$.

An extensive discussion on algorithms for computing the Vietoris-Rips complex can be found in CITE. The JavaPlex implementation is based on the results of this paper.

4.2 Witness Complexes and Landmark Sets

5 Homology computation

As mentioned in the abstract, the main goal of JavaPlex is to facilitate the computation of the homology of arbitrary filtered chain complexes of vector spaces. Almost always, such complexes arise from some sort of topological construction. Below we outline these different situations.

// maybe talk about "four-corners"

5.1 Simplicial Homology

This is the "standard" situation, which was also handled by previous versions in the Plex family. Here, we have a filtered sequence of simplicial complexes $X_1 \subset X_2 \subset ... \subset X_n$, from which we define the vector space of chains, $C(X_i)$ consisting of formal sums of elements of X_i with coefficients in the field $\mathbb{F}$.

In this case the boundary operator $\partial : C(X_i) \to C(X_{i-1})$ is the actual geometric boundary defined by

$$\partial([v_0, ..., v_n]) = \sum_i (-1)^i [v_0, ..., \hat{v_i}, ..., v_n]$$

5.2 Cellular Homology

This is where we depart from previous Plex implementations. In this case, $X = X_*$ is a filtered cell complex. This complex is formed by inductively adding n-cells to the $n-1$-skeleton, by the gluing maps

$$\varphi_\alpha^n : S^{n-1} \to X_{d-1}$$

which map the boundaries of the n-cells e_α^n to the $n-1$ skeleton. Note that in the above, we use n to denote the grading by dimension, and d to denote the grading by filtration index.

The boundary operator then becomes

$$\partial(e_\alpha^n) = \sum_\beta \deg(\varphi_{\alpha\beta}^n) e_\beta^{n-1}$$

where deg refers to the topological degree of a map.

Note that while JavaPlex is fully capable of computing the persistent homology of arbitrary cell complex, the specification of such complexes are more tedious than simplicial complexes. Nevertheless they offer the user a parsimonious way of defining a wide class of topological spaces.

5.3 Operations on Chain Complexes

The abstraction away from geometric primitives allows JavaPlex to handle more general algebraic constructions using complexes.

- **Tensor Products** Given two chain complexes (graded by dimension), (A_*, d_*) and (B_*, d'_*), the tensor product complex $(A \otimes B)_*$ is defined by

$$(A \otimes B)_n = \bigoplus_{p+q=n} A_p \otimes B_q$$

  Given two such complexes, the construction of the tensor product is very straightforwardly implemented in JavaPlex.

- **Hom Complex** Given two chain complexes (graded by dimension), (A_*, d_*) and $(B_*, d\prime_*)$, the hom-complex $(\mathrm{Hom}(A, B)_*$ is defined by

$$\mathrm{Hom}(A, B)_n = \bigoplus_{p \in \mathbb{Z}} \mathrm{Hom}(A_p, B_{p+n})$$

  which in the case of field coefficients reduces to

$$\mathrm{Hom}(A, B)_n = \bigoplus_{p \in \mathbb{Z}} A^p \otimes B_{p+n}$$

  Recently, in an upcoming paper (CITE), the hom-complex was used to compute a parameterization for the the space of homotopy classes of chain maps between simplicial complexes.

- **Algebraic Mapping Cylinder**

6 Examples

// Item to include

- examples of filtered simplicial complex construction - witness, vietoris-rips, etc.
- basic simplicial homology
- cellular homology examples - e.g. klein bottle in different coefficient fields
- hom-complex example - maybe a mapping
- benchmarks??

7 Implementation notes

// maybe talk about performance of different coefficient fields - specialized Z/2Z