Introducción al análisis topológico de datos

Autor:
Juan Camilo Gutiérrez Díaz

Asesor:
Jairo Andrés Angel Cardenas

Trabajo de grado
Pregrado en Matemáticas

Universidad de Los Andes
Facultad de Ciencias
Departamento de Matemáticas

Bogotá, Colombia, 2023

Índice general

Índice de figuras

Introducción

La cantidad de datos obtenidos de diversas fuentes ha crecido exponencialmente en los últimos años. El análisis de grandes conjuntos de datos, conocido como "Big Data", se ha convertido en una importante área de investigación en campos como la inteligencia artificial, la biología, la física y la economía.

Sin embargo, el análisis de estos datos presenta desafíos significativos, ya que los métodos tradicionales de análisis estadístico pueden no ser suficientes para extraer información significativa de conjuntos de datos complejos de alta dimensión. En este contexto, el análisis de datos topológicos se ha convertido en una herramienta más prometedora para un análisis más eficiente de conjuntos de datos.

El análisis de datos topológicos utiliza conceptos topológicos para analizar un conjunto de datos y descubrir su estructura subyacente. La topología es una rama de las matemáticas que estudia las propiedades espaciales de los objetos, como su forma y conectividad.

Esta tesis tiene como objetivo examinar el uso del análisis de datos topológicos desde una perspectiva matemática rigurosa. Introduce los fundamentos teóricos de la topología y describe herramientas matemáticas para llevar a cabo este análisis.

En esta se exploran varios enfoques para el análisis de datos topológicos, como la persistencia homológica, que permite la detección de características topológicas persistentes en los datos, y la homología, que permite medir las relaciones entre conjuntos de datos. Esto se logrará mediante el estudio e implementación de los algoritmos de Mapper y Barcodes, los cuales son dos de los más usados dentro del área del análisis topológico de datos.

Capítulo 1

Complejos Simpliciales

Para comenzar a hablar sobre análisis topológico de datos, primero se tiene que hablar sobre los datos que se usaran para llevar a cabo el análisis. Estos datos usualmente se representan como un conjunto de puntos en el espacio, los cuales se pueden representar como vectores en $\mathbb{R}^d$. Este conjunto de datos puede provenir de dos fuentes, la primera es que estos provengan de una medición o la segunda es que estos sean generados de forma aleatoria. Note que por la naturaleza de la primera fuente, se asume que los datos vienen de un muestreo con cierta exactitud, mientras en la segunda fuente se asume un grado de aleatoriedad. Sin importar la fuente de la cual provengan los datos, nuestro objetivo será identificar propiedades topológicas de la variedad a la cual vienen asociados los datos. Para lograrlo, lo primero que se debería hacer es construir un complejo simplicial a partir del conjunto de datos para obtener una forma aproximada de la variedad topológica asociada. Las referencias para este capítulo son [5], [11] y [1].

1.1 Definiciones básicas

Para empezar, suponga que se tiene un conjunto de datos $\{x_0, \ldots, x_k\}$ en $\mathbb{R}^n$. Se asume que estos puntos satisfacen la condición de que el conjunto de vectores en $\mathbb{R}^n$ representados por las diferencias $\{x_1 - x_0, \ldots, x_k - x_0\}$ son linealmente independientes.

Definición 1.1.1. El K-símplice generado por los puntos $\{x_0, \ldots, x_k\}$ es el conjunto de todos los puntos $z = \sum_{i=0}^{k} a_i x_i$ donde $a_i \geq 0$ y $\sum_{i=0}^{k} a_i = 1$. Para un z dado, nos referimos a a_i como la i-ésima coordenada baricéntrica.

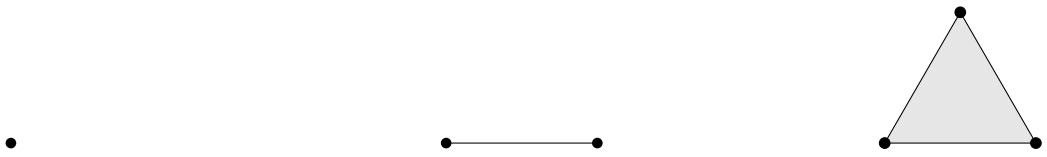

Figura 1.1: Ejemplo símplices

Definición 1.1.2. El interior de un símplice S generado por los puntos $x_0, \ldots, x_k$, denotado por $\text{int}(S)$, es el subconjunto de puntos donde $a_i > 0$ para todas las coordenadas baricéntricas a_i. La frontera $\text{bd}(S)$ se define cómo $S \setminus \text{int}(S)$.

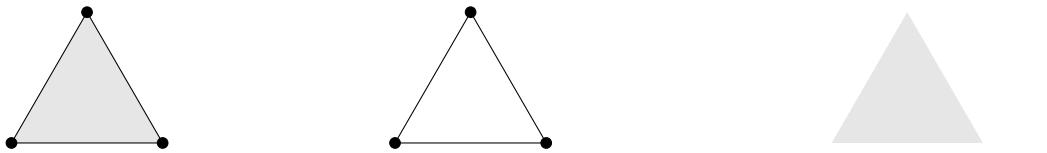

Figura 1.2: Ejemplo partes de símplice

Definición 1.1.3. Para un símplice S generado por los puntos $P = \{x_0, \ldots, x_k\}$, una *cara* de S se refiere a cualquier símplice generado por un subconjunto dé P.

Definición 1.1.4. Un complejo simplicial X en $\mathbb{R}^n$ es un conjunto de símplices en $\mathbb{R}^n$ tal que:

1. Cada cara de un símplice en X también es un símplice en X.
2. Toda intersección no vacía de dos símplices en X es una *cara* común.

Observacion 1.1.1. La dimensión de un símplice viene dada por la cantidad de vértices necesarios para definirlos menos uno, de igual manera la dimensión de un complejo simplicial será el máximo de las dimensiones de los símplices en él.

Definición 1.1.5. Definimos Y como un subcomplejo de un complejo simplicial X si Y es un subconjunto de X que también cumple con la definición de un complejo simplicial.

Ejemplo 1.1.1. El siguiente es un ejemplo de un complejo simplicial en $\mathbb{R}^2$, sea X el complejo simplicial definido de la siguiente manera:

$$X = \{\langle a_0\rangle, \langle a_1\rangle, \langle a_2\rangle, \langle a_3\rangle, \langle a_4\rangle, \langle a_5\rangle, \langle a_6\rangle, \langle a_0, a_1\rangle, \langle a_0, a_2\rangle, \langle a_0, a_4\rangle, \langle a_0, a_5\rangle, \langle a_1, a_2\rangle, \langle a_2, a_5\rangle, \langle a_2, a_3\rangle, \langle a_3, a_4\rangle, \langle a_3, a_5\rangle, \langle a_3, a_6\rangle, \langle a_4, a_5\rangle, \langle a_4, a_6\rangle, \langle a_2, a_3, a_5\rangle \langle a_3, a_4, a_5\rangle\}$$

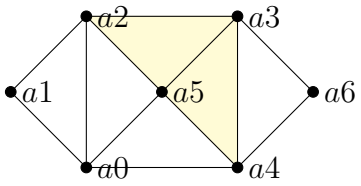

Figura 1.3: Ejemplo complejo Simplicial en $\mathbb{R}^2$

Note que el complejo simplicial se compone de 7 0-símplices,12 1-símplice y 2 3-símplice.

1.2 Recubrimiento

En esta sección definiremos los conceptos de recubrimiento y nervio, los cuales nos serán útiles para la construcción de los complejos simpliciales que definiremos más adelante.

Definición 1.2.1. Dado un conjunto X, definimos su recubrimiento como la colección de conjuntos no vacíos $\mathcal{C} = \{C_\alpha\}$, tales que su unión cubre completamente X, es decir, $X \subset \bigcup_\alpha C_\alpha$.

Definición 1.2.2. El nervio de un recubrimiento $\mathcal{C}$ se define como el complejo simplicial donde:

- Los vértices corresponden a los conjuntos en $\mathcal{C}$, es decir, $\{C_\alpha\}$.
- Los K-símplices $(\alpha_0, \ldots, \alpha_k)$ cuando la intersección de sus respectivos conjuntos en el recubrimiento no es vacía.

En otras palabras, cada vértice vendría dado por los conjuntos del recubrimiento, por cada pareja de recubrimientos que se intercepten se crea una arista (1-símplice), por cada terna se crea un triángulo (2-símplice), y así sucesivamente. Para ilustrar mejor esta definición se realizará un ejemplo de un nervio.

Ejemplo 1.2.1. Considere el conjunto de puntos $X = \{a_0, \ldots, a_{10}\}$ y un recubrimiento $\mathcal{C} = \{C_1, C_2, C_3, C_5, C_6\}$ de dicho conjunto.

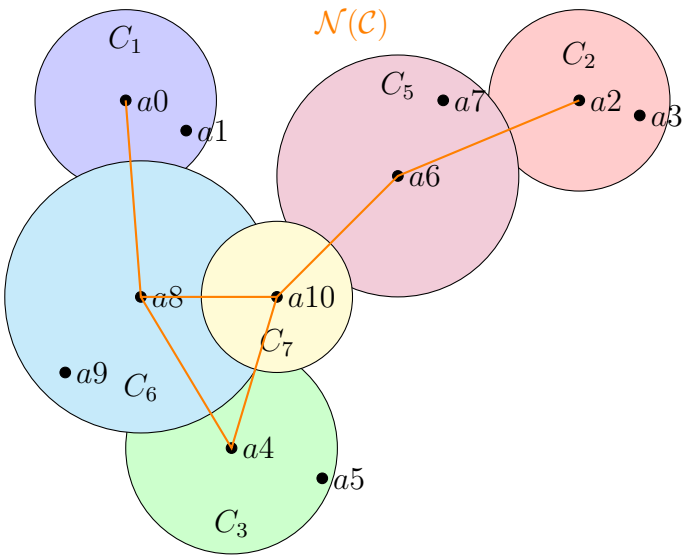

Figura 1.4: Ejemplo Nervio de complejo simplicial.

En este ejemplo se puede ver cómo el nervio es el complejo simplicial

$$\mathcal{N}(\mathcal{C}) = \{\langle a_0\rangle, \langle a_2\rangle, \langle a_4\rangle, \langle a_6\rangle, \langle a_8\rangle, \langle a_{10}\rangle, \langle a_0, a_8\rangle, \langle a_2, a_6\rangle, \\ \langle a_4, a_8\rangle, \langle a_4, a_{10}\rangle, \langle a_6, a_{10}\rangle, \langle a_8, a_{10}\rangle, \langle a_4, a_8, a_{10}\rangle\}$$

Gracias al siguiente teorema, podemos establecer una relación de homotopía entre el recubrimiento y su respectivo nervio.

Teorema 1.2.1. Suponga que X es un espacio topológico paracompacto y U un recubrimiento que consiste en conjuntos abiertos numerables. Además, suponga que para todo conjunto no vacío, $\emptyset \neq S \subseteq A$ donde A es el conjunto de vértices, y se cumple que la intersección de los conjuntos U_s para $s \in S$ es contráctil o vacía. Entonces, $\mathcal{N}(U)$ es homotópicamente equivalente a X.

Con las definiciones dadas hasta el momento, podemos definir dos de los complejos simpliciales más utilizados a la hora de realizar Análisis topológico de datos, los cuales son el Complejo de Čech y el Complejo de Vietoris-Rips

1.3 Complejo de Čech

Este complejo es el más elemental que se puede construir sobre un conjunto de puntos X, que a su vez nos ayuda a describir como este recubre a una variedad topológica M.[1]

Definición 1.3.1. Dado un conjunto de puntos X y un $\epsilon > 0$, considere el recubrimiento abierto $U_\epsilon = \{B(x, \epsilon) | x \in X\}$. El complejo de Čech $C(X, \epsilon)$ se define como el nervio dé U_ϵ. Además, $C(S, 0) = \emptyset$.

Por la definición, podemos ver que los vértices de $C(X, \epsilon)$ vendrían dados por los puntos en X y los símplices se definirían por los subconjuntos de bolas con intersección no vacía. Note que estas bolas son convexas y contráctiles, por lo cual al aplicar el teorema del nervio tendríamos que $C(X, \epsilon)$ sería homotópica equivalente a la variedad M asociada a X.Para ilustrar el complejo se da un ejemplo utilizando el mismo conjunto de puntos X utilizado en el ejemplo anterior. Para el recubrimiento se usará $\epsilon = 0{,}6$

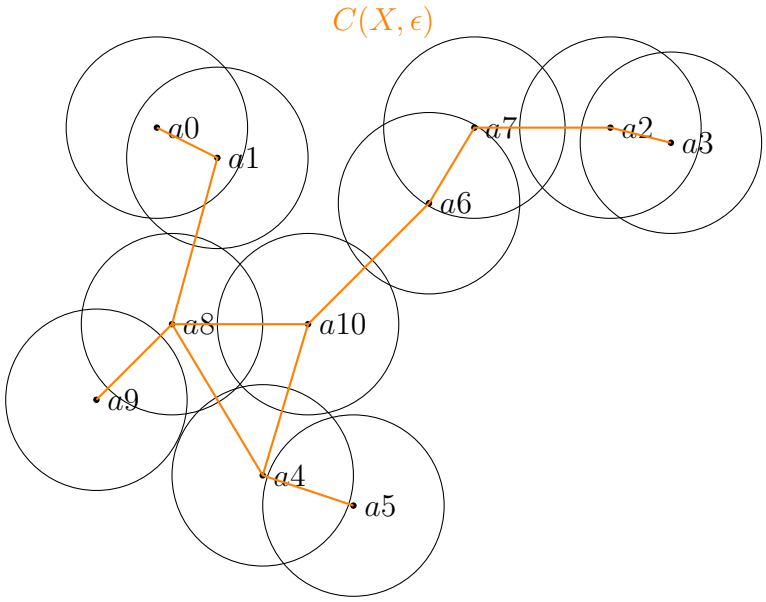

Figura 1.5: Ejemplo Complejo de Čech.

El complejo simplicial resultante se compone de 11 0-símplices, 11 1-símplices y 0 2-símplice. Sin embargo, por su construcción el único número fijo de elementos es el de los símplices de dimensión 0.

Observacion 1.3.1. Note que sí $0 < \delta < \epsilon$, tendríamos que $C(X, \delta) \subset C(X, \epsilon)$, adicionalmente, para un cierto $N > 0$ muy grande, se tendría que el complejo $C(X, N)$ sería un $(N-1)$ - símplice. Ahora miremos un resultado que usa esta construcción. [1]

Teorema 1.3.1. Sea M una variedad riemanniana compacta. Entonces existe un número positivo n tal que $C(M, \epsilon)$ es homotópicamente equivalente a M siempre que $\epsilon \leq n$. Además, para cada $\epsilon \leq n$, existe un subconjunto finito $V \subseteq M$ tal que el subcomplejo de $C(V, \epsilon) \subseteq C(M, \epsilon)$ es también, homotópicamente equivalente a M.

El principal problema de esta construcción es que es computacionalmente costosa, dado que necesita almacenar símplices de varias dimensiones. Para lidiar con este problema sería bueno poder construir un complejo simplicial

que se pueda recuperar únicamente a partir de la información de las aristas, por lo cual se definirá otra forma de realizar complejos simpliciales conocida como complejo de Vietoris-Rips.

1.4 Complejo de Vietoris-Rips

Definición 1.4.1. Definamos X como un espacio métrico con métrica d. El complejo de Vietoris-Rips para X, con parámetro ϵ, denotado por $VR(X, \epsilon)$, es el complejo simplicial cuyo conjunto de vértices es X, y donde $\{x_0, x_1, ..., x_k\}$ forma un k-símplice si y solo si $d(x_i, x_j) \leq \epsilon$ para todo $0 \leq i, j \leq k$.

Para ilustrar mejor el complejo se usará el mismo escenario usado para el complejo de Čech ($\delta = 2\epsilon$):

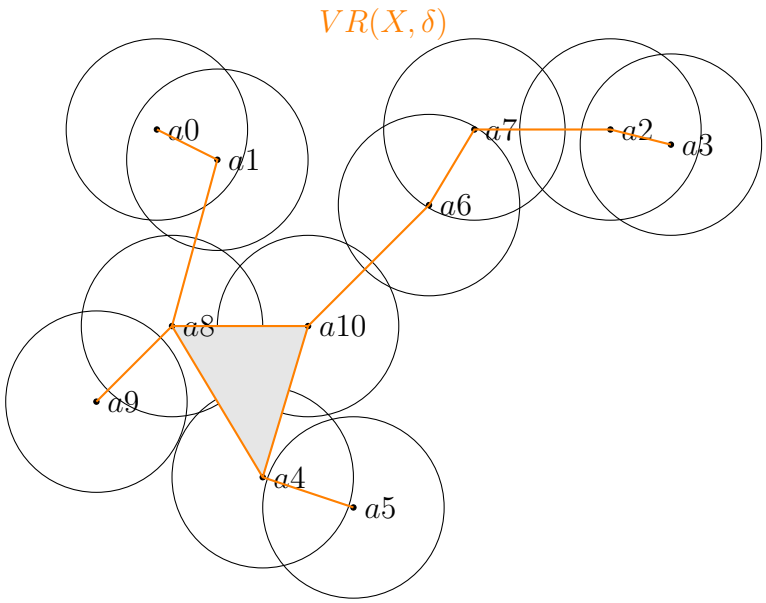

Figura 1.6: Ejemplo Complejo de Vietoris-Rips.

Note que pese a ser las mismas bolas, tenemos un 2-símplice, ya que no es necesario que se intercepten las 3 bolas al tiempo. La principal diferencia este estos dos tipos de complejos se observa a la hora de definir sus símplices de dimensión mayor o igual a 2, dado que los vértices vienen definidos por

la cantidad de conjuntos en el recubrimiento y las aristas vendrían definidas por la matriz de distancias.

Lema 1.4.1. Para $X \subset \mathbb{R}^n$ se cumple $C(X, \epsilon) \subset VR(X, 2\epsilon) \subset C(X, 2\epsilon), \forall \epsilon > 0$.

Demostración: Sea $S = \{x_0, x_1, \ldots, x_k\}$ un k - símplice cualquiera de $C(X, \epsilon)$ por definición, tendríamos que las bolas de radio ϵ centradas en cada uno de los x_i tienen intersección común. Sea p un punto de la intersección, entonces $\forall i, j$, con $0 \leq i, j \leq k$ se tendría:

$$d(x_i, x_j) \leq d(x_i, p) + d(p - x_j) \leq \epsilon + \epsilon = 2\epsilon.$$

Es decir, $S \in VR(X, 2\epsilon)$.
Ahora, Sea $S = \{x_0, x_1, \ldots, x_k\}$ un k - símplice cualquiera de $VR(X, 2\epsilon)$, por definición, tendríamos que $d(x_0, x_i) \leq 2\epsilon \ \forall i$ entre 0 y k, por lo cual x_0 pertenece a todas las bolas de radio 2ϵ centradas en x_i, lo que implicaría que todas las bolas de radio 2ϵ tienen intersección no vacía. Por lo tanto, $S \in C(X, 2\epsilon)$.

Logrando así las inclusiones $C(X, \epsilon) \subset VR(X, 2\epsilon)$ y $VR(X, 2\epsilon) \subset C(X, 2\epsilon)$ □

En resumen, complejo simplicial de Čech tiende a capturar mejor las propiedades locales del conjunto, mientras que el complejo simplicial de Vietoris-Rips es más sensible a la estructura global de los puntos con relación a la distancia.

Capítulo 2

Preliminares
Homología persistente

En el capítulo anterior se vio el concepto de complejo simplicial, el cual nos ayuda en la representación de los datos, ahora miraremos como obtener información sobre su topología. Para esto usaremos el concepto de grupos de homología, este concepto nos ayudará a estudiar la estructura topología en términos de ciclos y fronteras. Específicamente miraremos la homología de un complejo Simplicial. Las referencias utilizadas en este capítulo son [15] , [10] y [6].

2.1 Homología Simplicial

De acuerdo a la definición dada en el capítulo anterior de un complejo simplicial, un símplice σ ,viene dado por un conjunto de vértices $\{x_{i_0}, x_{i_1}, ..., x_{i_p}\}$ en X. Para precisar que es la homología simplicial, suponemos que los símplices están ordenados de acuerdo a sus vértices.

2.1.1. Símplices ordenados

Definición 2.1.1. Un símplice ordenado $\{x_{i_0}, x_{i_1}, ..., x_{i_p}\}$ se considera bien ordenado, sí $i_0 < i_1 < ... < i_p$, es decir, si los vértices siguen el orden especificado inicialmente en S. Por otra parte, un símplice ordenado se considera orientado positivamente si se puede encontrar una permutación par de sus vértices que lo transforma en un símplice bien ordenado. En caso contrario, se dice que el símplice ordenado está orientado negativamente.

De acuerdo a la definición dada, tenemos que el símplice $\{x_0, x_1\}$ sería el mismo que $\{x_1, x_0\}$ pero con diferente orden, en particular tendríamos que el primero estaría ordenado de forma positiva, mientras el otro estaría en forma negativa. Una vez definido este concepto proseguiremos definiendo grupos de cadenas.

2.1.2. Grupos de Cadenas

Definición 2.1.2. Si consideramos un complejo simplicial S, el grupo de cadenas $C_n(S)$ correspondiente a su n-ésima dimensión es un grupo abeliano libre cuyos generadores son los n-símplices bien ordenados.

Un reordenamiento de los vértices de un símplice se considera equivalente a un cambio de signo si la permutación involucrada es impar, mientras que no tiene ningún efecto si la permutación es par.
Dado que un complejo simplicial no contenga n-símplices, entonces no existen generadores y $C_n(S) = \{0\}$. A partir de S, podemos construir el grupo $C_n(S)$ para cualquier $n \in \mathbb{Z}$.

Definición 2.1.3. Un elemento $c \in C_n(S)$, se denomina una n-cadena y puede ser ilustrada como $c = \sum_i c_i \sigma_i$, donde σ_i son n-símplices bien ordenados de S y c_i son coeficientes en $\mathbb{Z}$.

2.2 Homomorfismo de frontera

Con lo anteriormente definido, podemos comenzar a definir los homomorfismos de frontera, los cuales nos ayudaran a describir como se relacionan los grupos de cadena entre ellos.

Definición 2.2.1. El homomorfismo de frontera n-ésimo es un homomorfismo de grupos definido $\partial_n : C_n(S) \to C_{n-1}(S)$, donde su operación se define como:

$$\partial_n(\sigma) = \sum_{i=0}^{n} (-1)^i [x_0, \ldots, \hat{x_i}, \ldots, x_n]$$

Donde σ es un símplice y $\hat{x_i}$ representa el vértice que se ha eliminado de este. Ahora miremos un ejemplo de un homomorfismo de frontera.

Ejemplo 2.2.1. Consideremos un 3-símplice con vértices $[x_0, x_1, x_2, x_4]$.

$$\partial_4(\sigma) = [x_1, x_2, x_3] - [x_0, x_2, x_3] + [x_0, x_1, x_3] - [x_0, x_1, x_2]$$

Definición 2.2.2. Sea c una n-cadena, definiremos frontera de c a $\partial_n(c)$. Si la frontera de c llega a ser nulo, lo denominaremos n- ciclo.

Antes de continuar se mencionará un lema fundamental para los homomorfismos de frontera.

Lema 2.2.1. La composición de dos homomorfismos consecutivos de frontera de un complejo simplicial S, da como resultado 0.

A continuación se mostrará un ejemplo de composición de dos homomorfismos de frontera consecutivos:

Ejemplo 2.2.2. Consideremos un 3-símplice con vértices $[x_0, x_1, x_2, x_4]$.

$$\begin{aligned}\partial_3 \circ \partial_4([x_0,x_1,x_2,x_3]) &= \partial_3([x_1,x_2,x_3]) - \partial_3([x_0,x_2,x_3]) + \partial_3([x_0,x_1,x_3]) - \partial_3([x_0,x_1,x_2]) \\ &= ([x_2,x_3] - [x_1,x_3] + [x_1,x_2]) - ([x_2,x_3] - [x_0,x_3] + [x_0,x_2]) \\ &+ ([x_1,x_3] - [x_0,x_3] + [x_0,x_1]) - ([x_1,x_2] - [x_0,x_2] + [x_0,x_1]) \\ &= [x_2,x_3] - [x_1,x_3] + [x_1,x_2] - [x_2,x_3] + [x_0,x_3] - [x_0,x_2] \\ &+ [x_1,x_3] - [x_0,x_3] + [x_0,x_1] - [x_1,x_2] + [x_0,x_2] - [x_0,x_1] \\ &= 0.\end{aligned}$$

Como se mencionó anteriormente, este homomorfismo nos ayuda a realizar una conexión entre las cadenas, el cual definiremos de la siguiente manera.

Definición 2.2.3. Partiendo de un complejo simplicial S, definimos su complejo de cadenas C_* como el conjunto $\{\{C_n\}_{n\in\mathbb{N}}, \{d_n\}_{n\in\mathbb{N}}\}$ de grupos de cadenas y homomorfismos de frontera construyendo el siguiente diagrama:

$$C_*(S) : \ldots \to C_{n+1}(S) \xrightarrow{\partial_{n+1}} C_n(S) \xrightarrow{\partial_n} C_{n-1}(S) \to \ldots$$

Siguiendo con nuestro objetivo, lo siguiente sería definir los grupos de homología, por lo cual se introducirán las siguientes definiciones.

Definición 2.2.4. Las fronteras de dimensión n de S son

$$B_n(S) = Im(\partial_{n+1})$$

Definición 2.2.5. Los ciclos de dimensión n de S son

$$Z_n(S) = Ker(\partial_n)$$

Observacion 2.2.1. Note que $B_n(S) \subset Z_n(S)$

Demostración: Por definición, si $\sigma \in B_n(S)$ significa que existe un $n+1$ - símplice k de tal manera que $\partial_{n+1}(k) = \sigma$. Ahora, por el lemma 2.2.1, tendríamos que $\partial_n \circ \partial_{n+1}(k) = \partial_n(\sigma) = 0$, es decir, $\sigma \in Z_n(S)$ □

2.3 Grupo de Homología

Dado las definiciones y resultados anteriormente mencionados, podemos dar la definición de grupo de homología.

Definición 2.3.1. Si S es un complejo simplicial, definimos él n-ésimo grupo de homología como

$$H_n(S) = \frac{Z_n(S)}{B_n(S)}$$

La dimensión de $H_n(S)$ se le denomina él n-ésimo número de Betti. De acuerdo a las definiciones anteriormente dadas, tendríamos que $Z_n(S)$ serían todos los n-ciclos de S, por otro lado, $B_n(S)$ sería el grupo de los n-ciclos que son frontera en S, por lo cual tendríamos que $H_n(S)$ solo conservaría los ciclos que representarían agujeros en S.

Hasta el momento todo lo que hemos calculado y definido se encuentra ligado a un complejo simplicial; sin embargo, en el siguiente capítulo introduciremos el concepto de homología persistente, el cual es posible mediante el concepto de filtración.

Definición 2.3.2. Una filtración de un complejo simplicial S es una colección de complejos simpliciales, $\{S_i\}_{i\in\mathbf{N}}$ los cuales se contienen de la siguiente manera

$$\emptyset = S_0 \subset S_1 \subset \cdots \subset S_n = S$$

A continuación se mostrará un ejemplo de filtración

Ejemplo 2.3.1. Considere el complejo simplicial dado en el Ejemplo 1.1.1, definiremos la filtración de este a través de una colección de seis complejos Simpliciales $\{S_i\}_{0\leq i\leq 5}$,los cuales se verían de la siguiente manera:

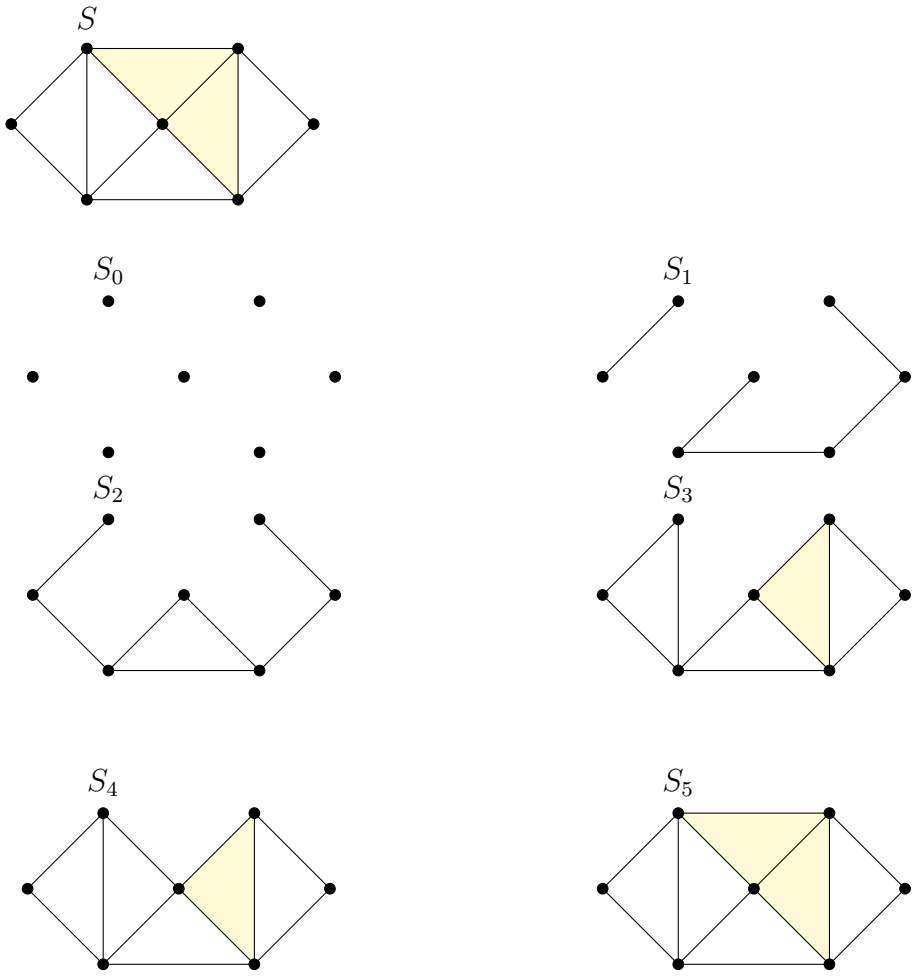

Figura 2.1: Ejemplo de filtración de complejo simplicial

Capítulo 3

Homología persistente

Cada uno de los complejos simpliciales que forman una filtración es una aproximación de la topología subyacente a los datos. En este capítulo, dejaremos de analizar individualmente lo que sucede en cada nivel de la filtración y en su lugar examinaremos la homología del complejo en todos los niveles de la filtración. Esto nos permitirá obtener una visión global de la estructura topológica presente en los datos. Esto será posible gracias a los grupos de homología persistente, la cual es una estructura algebraica que tiene como propósito reunir la información de la filtración. Las referencias para este capítulo son [4] y [11].

3.1 Grupo de homología persistente

Dada la definición de filtración del capítulo anterior, vamos a definir la función de inclusión de forma natural $i_{n,n+1} : S_n \hookrightarrow S_{n+1}$, esta función de inclusión genera un homomorfismo en sus respectivos grupos de homología de la siguiente manera

$$i^*_{j,k} : H_n(S_j) \to H_n(S_k)$$

Definición 3.1.1. Sea S,F un complejo simplicial y su filtración asociada, el (j,k)-ésimo grupo de homología persistente se define como:

$$H_{(j,k)n}(S, F) = Im(i^*_{j,k})$$

Al igual que con los grupos de homología simplicial, la dimensión del espacio vectorial $H_{(j,k)n}(S, F)$ se conoce como el (j, k)-ésimo número de Betti, el cual denotaremos $\beta_{(j,k)n}$. Las siguientes definiciones nos ayudarán para posteriormente definir los tipos de representaciones de la homología persistente.

Definición 3.1.2. Sea $\omega \in H(S_j)$ décimos que ω nace en j sí $\omega \notin H_{(j-1,j)n}(S, F)$, de ser así se denomina j como el valor de nacimiento de ω.

Definición 3.1.3. Sea $\omega \in H_n(S_j)$ décimos que ω es ancestro de $\alpha \in H_n(S_k)$ si $i_{j,k}(\omega) = \alpha$, donde $j \leq k$. Además, definimos como primer ancestro al ancestro de ω con valor mínimo.

De manera análoga definimos el concepto de descendiente.

Definición 3.1.4. Sea $\omega \in H_n(S_j)$ décimos que ω es descendiente de $\alpha \in H_n(S_k)$ si $i_{j,k}(\alpha) = \omega$, donde $k \leq j$.

Definición 3.1.5. Sea $\omega \in H(S_j)$ con valor de nacimiento k, decimos que este muere en m sí:

$$i_{k,j}(\omega) \notin H_{(k-1,j)n}(S, F) \text{ y } i_{k,j+1}(\omega) \in H_{(k-1,j+1)n}(S, F)$$

Por notación vamos a definir $n(\omega)$ como el valor de nacimiento de ω y $m(\omega)$ como su valor de muerte. Además, definiremos $n(0) = \infty$, es decir, que el cero nace en $-\infty$ y si ω no tiene descendientes, denotaremos $m(\omega) = \infty$.

Definición 3.1.6. Sea $\omega \in H(S_j)$ diferente de cero, definimos su tiempo de vida o persistencia como:

$$p(\omega) = m(\omega) - n(\omega)$$

Note que si ω no posee descendientes, $p(\omega) = \infty$.

Con estas definiciones, podemos describir los algoritmos para representar la homología persistente de un complejo simplicial.

Capítulo 4

Algoritmos para representar Homología persistente

En este capítulo se mostrará como se puede representar la homología persistente por medio de representaciones visuales. Específicamente se estudiará el algoritmo de código de barras y los resúmenes de persistencia. La referencias para este capítulo son [3], [2], [9] y [13].

4.1 Código de barras

El algoritmo de código de barras o"Barcode" en la dimensión n se representa en el plano real extendido, $[0, \infty]^2$ donde el eje horizontal representa el valor del parámetro ε, y a lo largo del eje vertical tenemos segmentos de los cuales representan una clase de homología n-dimensional (estos segmentos se asignan de forma aleatoria), comenzando en el valor de nacimiento y terminando en el valor de muerte.

4.2 Implementación

En nuestro caso se implementa el algoritmo de "Barcodes" en el lenguaje de programación Python, con ayuda de la librería `gudhi`, `ripser` y `persim`. Estas librerías nos permiten crear un Complejo simplicial de VR a partir de un conjunto de datos en forma de nube de puntos y posteriormente calcular sus clases de homología n - dimensionales. Una vez tengamos estos datos se realiza el gráfico descrito anteriormente. Por convención, se usarán líneas negras para representar las clases de homología de dimensión cero y rojo para las clases de dimensión 1.

A continuación se muestra un ejemplo del algoritmo de "Barcodes" aplicado a un conjunto de datos de un círculo con ruido centrado en $(0, 0)$.

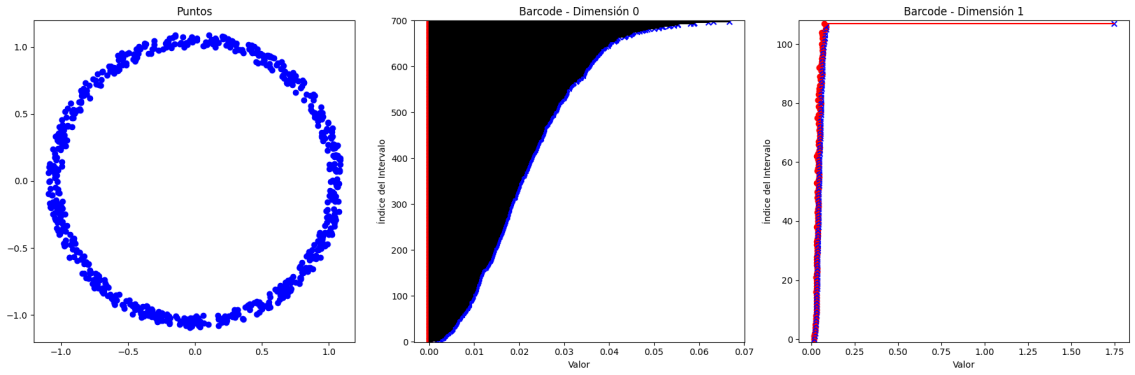

Figura 4.1: Ejemplo aplicación algoritmo "Barcodes" en S^1

.

En este ejemplo podemos observar como se generan varias líneas en el grupo de homología de dimensión 0, el cual hace referencia a las componentes conexas. Esto se debe a que todos los puntos se toman como componentes conexas al principio del algoritmo. Por otro lado, podemos notar como solo una línea destaca en los grupos de homología de dimensión 1, los cuales hacen referencia a los "ciclos".

Este tipo de representaciones resultan más amenas de leer, sin embargo, la información presentada en estos se pueden resumir en diagramas de persistencia.

4.3 Diagramas de persistencia

Al igual que con los "Barcodes", los diagramas de persistencia tienen como finalidad representar la homología persistente de un complejo simplicial. El diagrama de persistencia en la dimensión n se representa en el plano real extendido $[0, \infty]^2$, donde para cada elemento de $\omega \in H_n$, se dibuja el punto $(n(\omega), m(\omega)$, además se le añade la recta y=$\{(x, x)|x \in \mathbb{R}\}$. Note que, por definición, se tiene que $n(\omega) \leq m(\omega)$, por lo cual todos los puntos quedan arriba o sobre la recta y, por otro lado, los puntos que se encuentran alejados de la recta serían las clases de homología que más persisten.

Ahora, miremos el diagrama de persistencia para el conjunto de datos anteriormente usado.

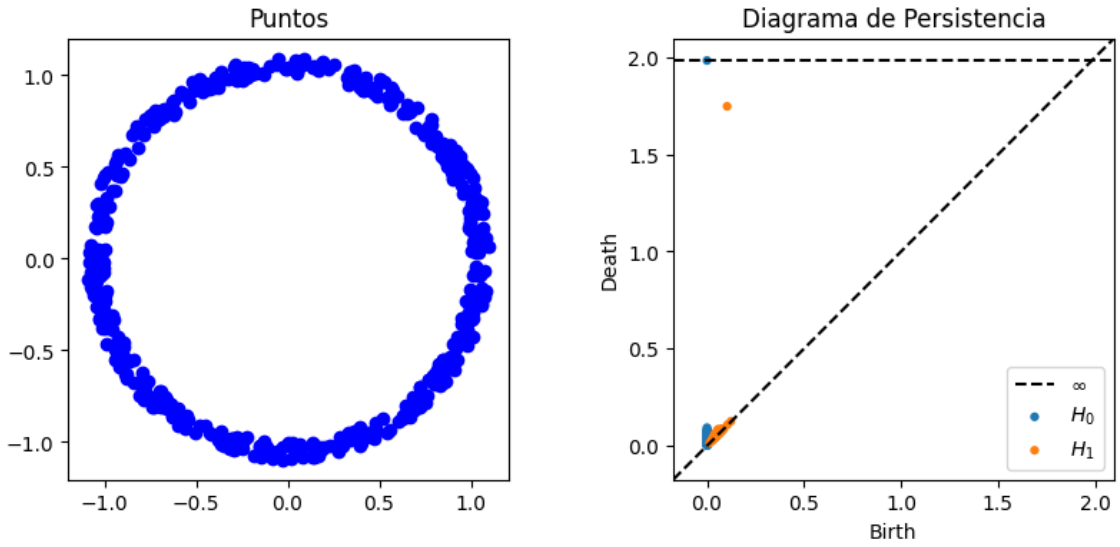

Figura 4.2: Ejemplo aplicación algoritmo diagrama de Persistencia en S^1

.

En este caso podemos identificar que al igual que en el algoritmo de "Barcodes" tenemos dos clases de homología que persisten con respecto al tiempo.
Para este tipo de diagramas se puede definir una métrica para medir la distancia entre dos de estos, esta métrica recibe el nombre de distancia cuello de botella y se define de la siguiente forma:

Definición 4.3.1. Sean P, D dos diagramas de persistencia, en los cuales definimos la distancia cuello de botella (bottleneck) como:

$$d_B(P, D) = \inf_{\gamma} \sup_{p \in P} \|p - \gamma(p)\|$$

en este caso γ hace parte del conjunto de biyecciones entre P y D.

El siguiente teorema establece una relación entre la distancia "bottleneck" y la distancia de Hausdorff

4.3.1. Teorema de estabilidad para diagramas de persistencia

Teorema 4.3.1. Sean P, D los diagramas de persistencia para los espacios topológicos compactos (X, Y) respectivamente, se tendría:

$$d_B(P, D) \leq d_H(X, Y).$$

Donde $d_H(X, Y)$ sería la distancia de Hausdorff entre X y Y.

En otras palabras, los diagramas de persistencia son estables bajo perturbaciones pequeñas. Para ilustrar esta estabilidad se proporciona el siguiente ejemplo

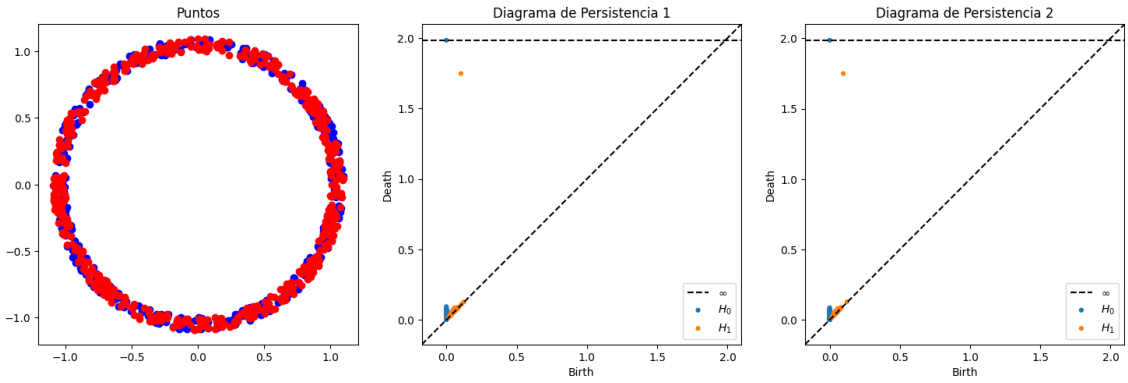

Figura 4.3: Ejemplo comparación diagramas de Persistencia

.

En este ejemplo se utilizaron dos muestreos aleatorios M_1, M_2 sobre S_1 donde D_1, D_2 serían sus respectivos diagramas de persistencia. La distancia $d_H(M_1, M_2) = 0{,}0528$, mientras que $d_B(D_1, D_2) = 0{,}0090$ en H_0 y $d_B(D_1, D_2) = 0{,}0082$ en H_1. Note que, a pesar de provenir de muestreos diferentes, sus diagramas de persistencia son muy parecidos.

En conclusión, estas representaciones visuales son bastante útiles a la hora de trabajar con complejos simpliciales de gran tamaño, dado que ayudan a comprender que características topológicas posee este.

Capítulo 5

Algoritmo de Mapper

En este capítulo, se presenta el algoritmo de Mapper, el cual es una técnica muy utilizada en el análisis topológico de datos para visualizar conjuntos de datos de altas dimensiones. Se dará una definición del algoritmo en pseudocódigo, diferentes casos de uso y por último se brindará un ejemplo gráfico de una implementación por medio de Python. Las referencias para este capítulo son [12],[7] y [14].

5.1 Introducción

El algoritmo de Mapper es una herramienta ampliamente utilizada en la topología aplicada y el análisis de datos. Este permite detectar estructuras topológicas significativas en conjuntos de datos complejos y de alta dimensionalidad. El objetivo principal del algoritmo es resumir la información topológica de los datos en una representación simplificada y visualmente interpretable. Esto lo logra mediante técnicas de agrupación y reducción de dimensionalidad.

5.2 Descripción del algoritmo

El algoritmo de Mapper se define de la siguiente manera

Definición 5.2.1. Considere el espacio X y una función de filtro f, el algoritmo de Mapper sigue los siguientes pasos:

1. Lo primero que se realiza es recubrir con intervalos de igual longitud el rango de los valores que toma f, estos intervalos se deben solapar unos a otros.

2. Una vez se tengan los intervalos se les calcula su respectiva pre imagen.

3. Dadas las preimágenes de los intervalos, se les aplicaría un algoritmo de agrupamiento ("clustering") a cada una de ellas.

4. Con la información de los agrupamientos se crea un complejo simplicial S, en el cual sus vértices vendrían dados por los clústeres.

5. Por último, se define un símplice S con los clústeres $a_0, a_1, \ldots, a_n$ si y solo si estos comparten un punto en X

Este algoritmo se puede implementar de varias formas, dado que no se define la función de filtrado o proyección ni el método de agrupación. Por lo cual antes de pasar a la implementación del algoritmo se proporcionara unos ejemplos de funciones de filtrado y de algoritmos de agrupación.

5.2.1. Funciones de filtrado

En este caso, las funciones de filtrado ayudan a reducir la dimensionalidad de los datos, esto lo logran al proyectar los datos en un espacio de menor dimensión. Ahora, se mirará una forma de llevar a cabo este proceso por medio de la técnica de Análisis de componente Principal.

Análisis de componente principal (PCA)

Sea $\mathbf{X} \in \mathbb{R}^{n \times p}$ la matriz de datos de entrada, donde el número de observaciones es n y el número de variables es p. El objetivo del análisis de componentes principales es encontrar una transformación lineal $\mathbf{T}$ que proyecte los datos originales a un nuevo espacio de menor dimensión $\mathbb{R}^{n \times k}$, donde las nuevas variables, conocidas como componentes principales, son obtenidas mediante combinaciones lineales de las variables originales.
Ejemplo gráfico del algoritmo usando un muestreo aleatorio en 3 dimensiones.

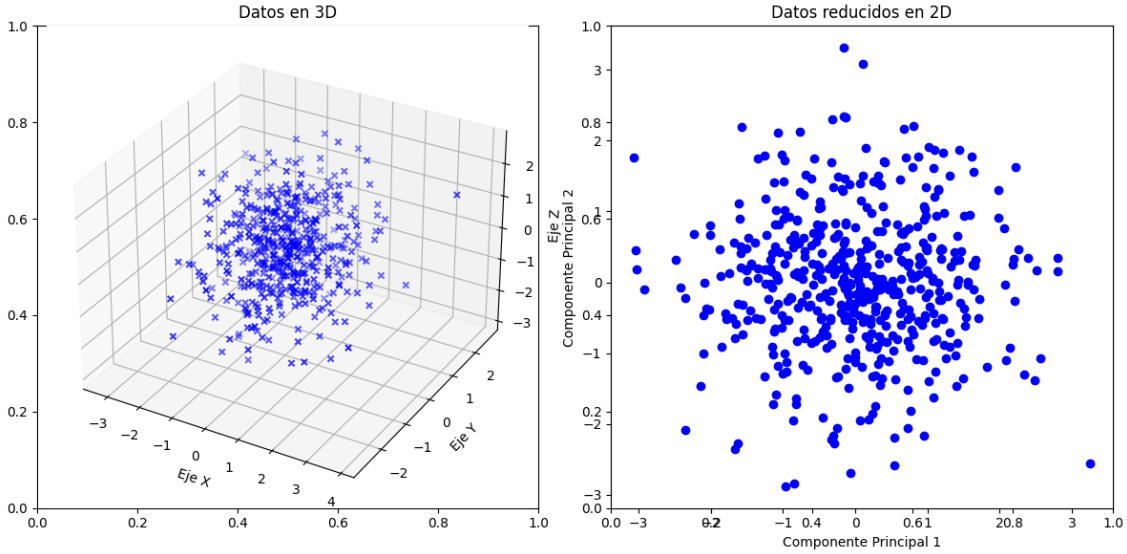

Figura 5.1: Ejemplo PCA

5.2.2. Algoritmos de agrupación

Como su nombre indica, estos tipos de algoritmos buscan agrupar un conjunto de datos en subconjuntos, donde cada uno de estos subconjuntos recibe el nombre de clústeres. A continuación, se presenta una breve descripción del algoritmo K-means.

Algoritmo K-means

El algoritmo K-means es un método de agrupamiento no supervisado que se utiliza para dividir un conjunto de datos en K grupos o clusters. El objetivo del algoritmo es minimizar la varianza intra-cluster, es decir, la suma de las distancias al cuadrado entre cada punto de datos y el centroide del cluster al que pertenece.
A continuación se provee un pseudocódigo que describe el algoritmo:

Algorithm 1 Algoritmo K-means

Entrada: Conjunto de datos $X = \{x_1, x_2, \ldots, x_n\}$, número de clusters K.
Salida: Conjunto de centroides $C = \{c_1, c_2, \ldots, c_K\}$.

1 Inicializar los centroides C de forma aleatoria o utilizando algún otro método.
2 **Repetir hasta convergencia**
3 Asignar cada dato x_i al centroide más cercano:
$c_i = \arg\min_{c_j \in C} \|x_i - c_j\|^2$
4 Actualizar cada centroide como el promedio de los datos asignados a él:
$c_j = \frac{1}{|\{i : c_i = c_j\}|} \sum_{i:c_i=c_j} x_i$
5 **fin**

Ahora miremos un ejemplo gráfico de este algoritmo usando los mismos datos que se usaron en el ejemplo de PCA.

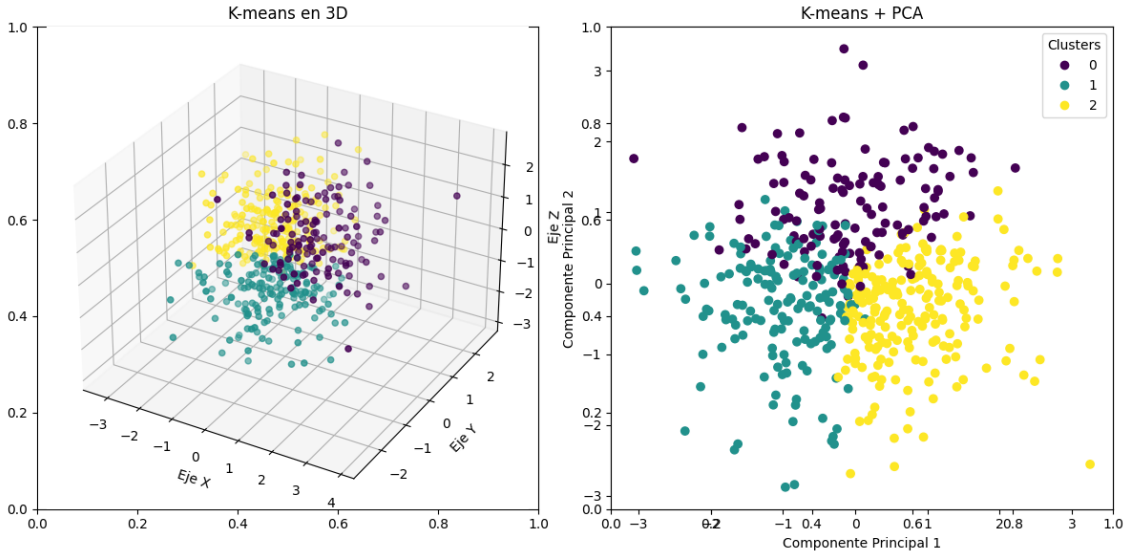

Figura 5.2: Ejemplo PCA

.

Una vez definido un ejemplo de filtrado y agrupamiento, procedemos a definir la implementación del algoritmo Mapper.

5.3 Implementación

El algoritmo de Mapper puede implementarse utilizando diversos lenguajes de programación y bibliotecas especializadas en análisis topológico de datos. Algunas bibliotecas populares incluyen `KeplerMapper` en Python y `TDAmapper` en R. En nuestro caso usaremos la librería `KeplerMapper`, esta nos ayudará construir el algoritmo de Mapper a partir de una configuración dada. Esta librería arroja como resultado un archivo HTML interactivo, en nuestro caso lo implementaremos usando PCA como función de filtrado y K-means como algoritmo de "clustering".

5.4 Ejemplo gráfico

A continuación se muestra un ejemplo gráfico del algoritmo de Mapper aplicado al conjunto previamente utilizado, en este usamos PCA para reducir dimensionalidad con 2 componentes principales y usamos el algoritmo de K-means con 3 clústeres.

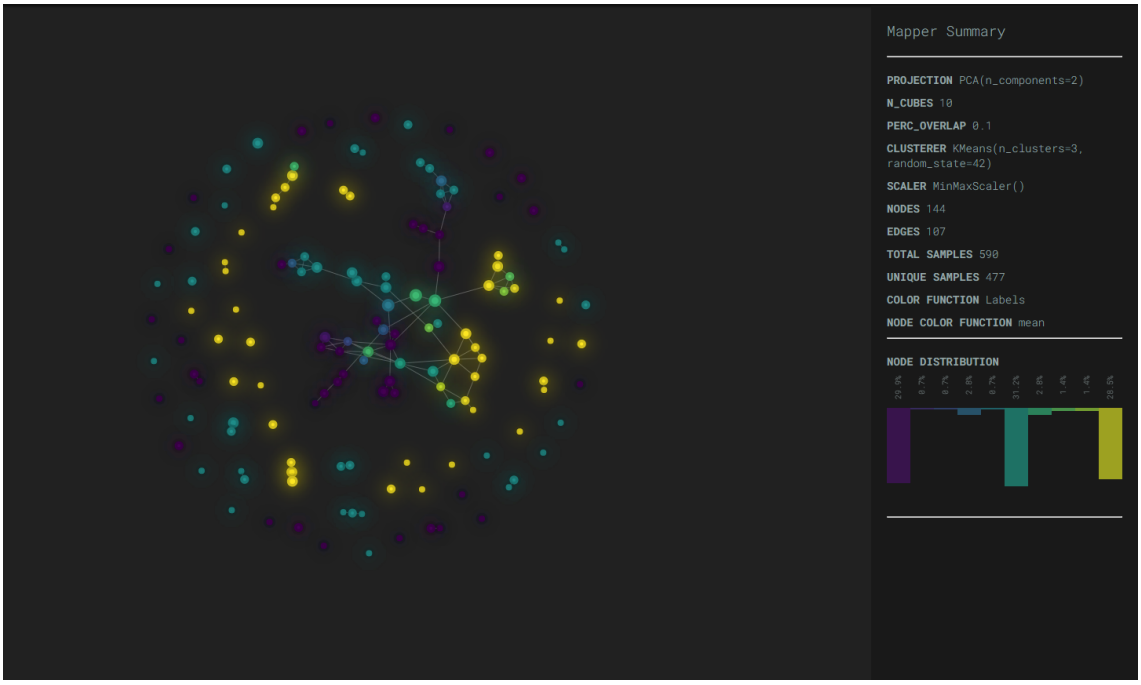

Figura 5.3: Ejemplo Mapper

En este ejemplo, el algoritmo de Mapper ha revelado tres agrupaciones distintas en los datos, representadas por los nodos coloreados en la red de cobertura. Estas agrupaciones pueden corresponder a diferentes clases, características o patrones presentes en los datos. Por defecto, el recubrimiento que utiliza este algoritmo para generar los intervalos son n-cubos que se sobreponen un 10 % unos a otros.

En conclusión, a diferencia de las representaciones de homología persistente, Mapper nos permite visualizar las estructuras topológicas presentes en conjuntos de datos complejos mediante una representación simplificada que facilita la interpretación y comprensión de los datos.

Capítulo 6

Implementación de algoritmos

En este capítulo se dará un ejemplo de la implementación de los algoritmos anteriormente mencionados con un conjunto de datos reales. Esta implementación se realizó en el lenguaje de programación Python.

6.1 Descripción del conjunto de datos

El conjunto de datos que se utilizó en la implementación es uno similar al usado en el artículo [8], este contiene la información de la matriz de distancias de información genética de casos de malaria en la región de Guapi, Colombia y la fecha en la cual se registró el caso. Esta matriz se construye calculando las distancias genéticas entre las secuencias de ADN del parásito de la malaria obtenidas de cada caso. Una de las ventajas que brinda esta representación es que se puede utilizar para crear un complejo simplicial de Vietoris-Rips.

Esta información genética contiene información de 97 casos de malaria, los cuales se encuentran divididos en tres grupos (A, B y C).

6.2 Homología persistente

En esta primera parte se implementarán los algoritmos de "Barcodes" y Diagramas de persistencia para analizar las características topológicas asociadas a los datos.

Cuadro 6.1: Tabla de Grupo y Número de Casos de malararia en Guapi

Grupo	Número de Casos
A	33
B	42
C	22

Para comenzar se examina el diagrama de persistencia y "Barcodes" de todos los datos juntos:

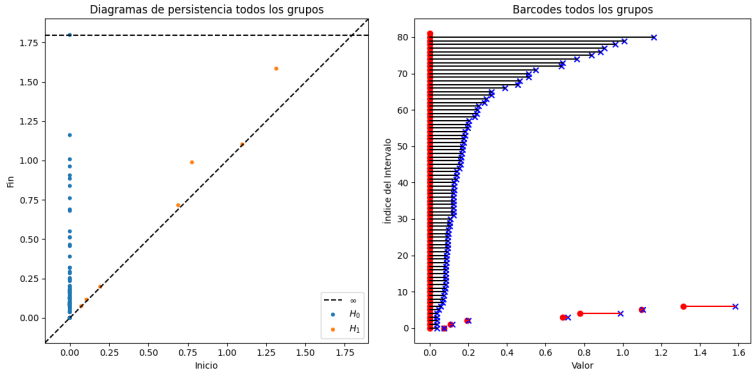

Figura 6.1: Diagrama de persistencia de todos los grupos

Ahora miremos los diagramas de persistencia y "Barcodes" de cada uno de los grupos.

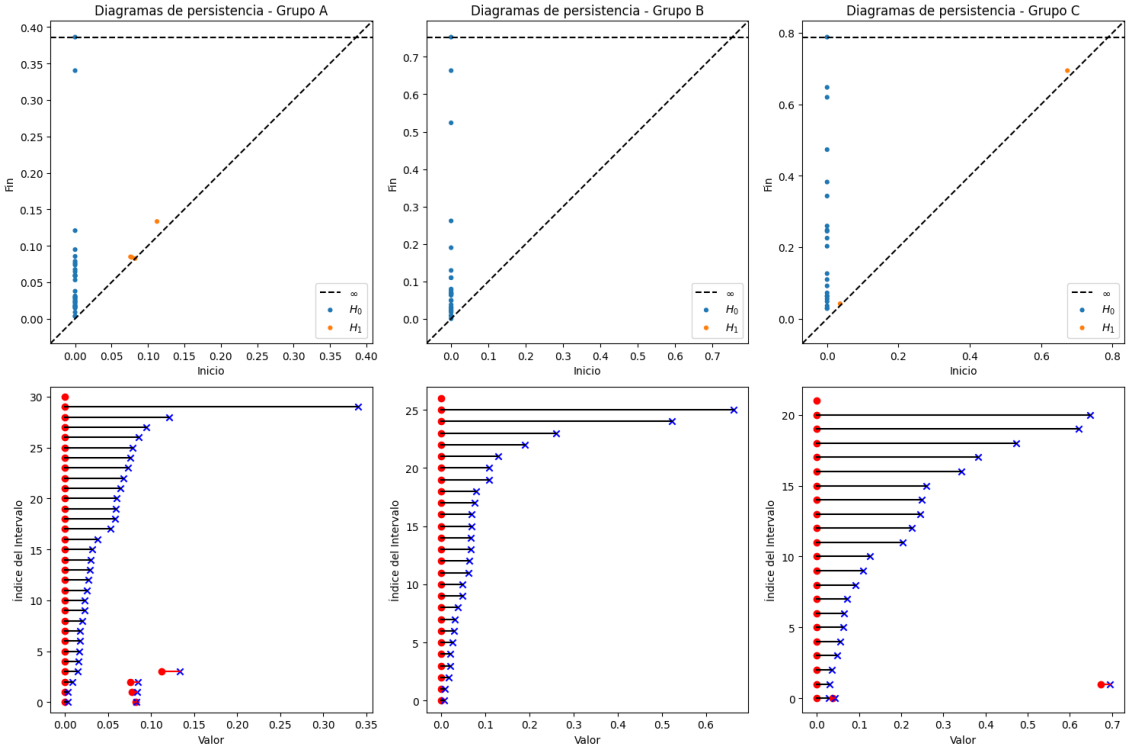

Figura 6.2: Diagrama de persistencia y "Barcodes" de cada grupo

Note que en las gráficas de todos los datos juntos hay presencia de ciclo, mientras que en los grupos individuales estos aparecen de muy corta duración.

Ahora se mirarán los diagramas de persistencia y "Barcodes" dependiendo la fecha de identificación del caso, para esto se realizó una agrupación similar a la que se hizo en el artículo, la cual queda de la siguiente manera:

Intervalo	**Día de inicio**	**Día final**
1	0	200
2	81	281
3	174	374
4	261	461
5	348	548
6	435	635
7	522	722
8	609	809
9	696	896
10	783	983

Cuadro 6.2: Tabla de intervalos

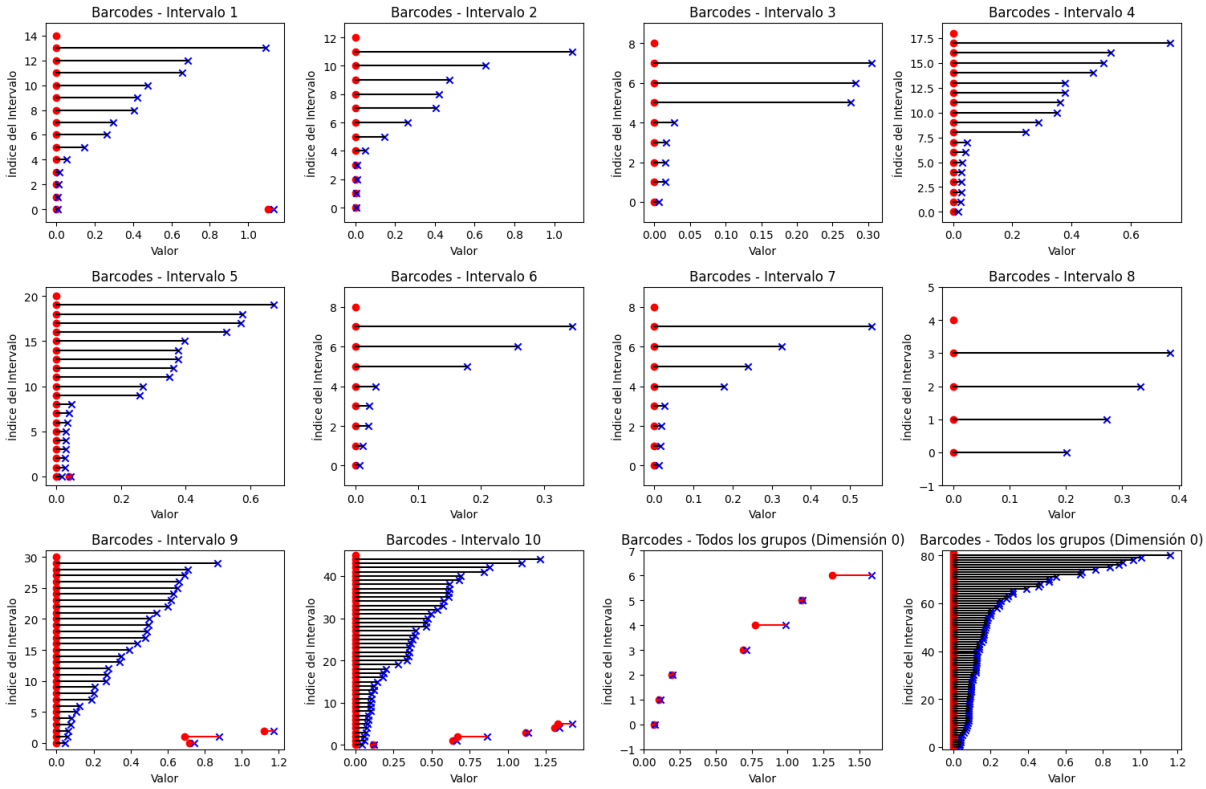

Figura 6.3: "Barcodes" de cada intervalo de tiempo

En estos gráficos, podemos notar como en los últimos intervalos de tiempo se presencian los ciclos, los cuales también coinciden con los intervalos donde más casos hubo, esto se puede inferir dado el gran número de características de dimensión 0, por lo cual podríamos suponer que en los picos de una pandemia hay más probabilidad de encontrar estos ciclos.

6.3 Algoritmo Mapper

En esta segunda parte, se implementará el algoritmo de Mapper al set de datos proporcionado. Para comenzar se realizará la implementación del algoritmo, usando PCA con dos componentes principales, K-means con 3 clústeres, recubrimientos de 20 n-cubos y solapamiento del 50 %

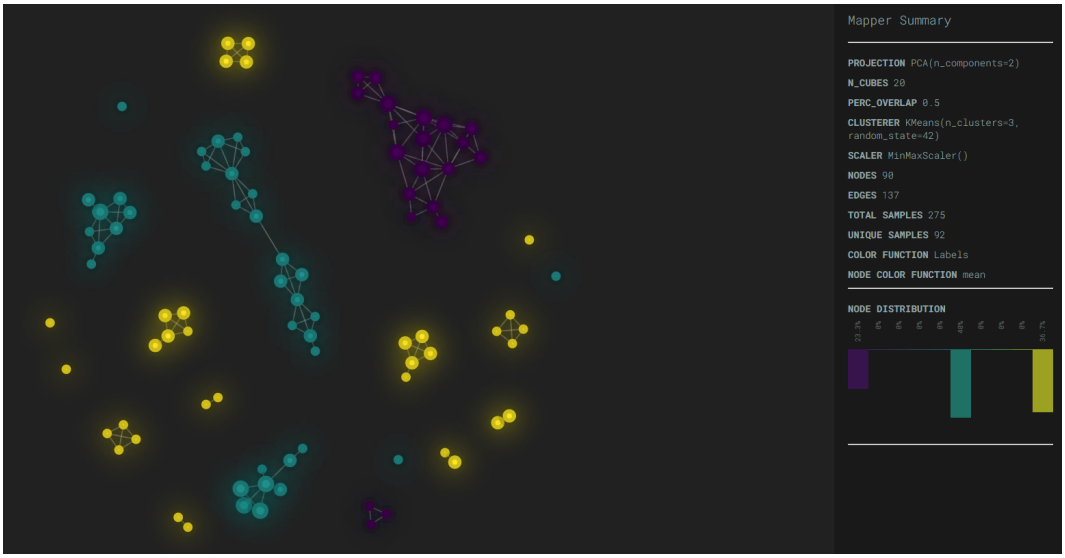

Figura 6.4: Algoritmo Mapper implementado con todos los datos de Guapi

El algoritmo nos da como resultado un grafo el cual se distribuye en 3 grupos, morado con 23,3 %, amarillo con 36,7 % y azul con 40 %. Estos porcentajes se asemejan bastante a los grupos en los cuales se encuentra dividido los datos, a continuación se muestra una tabla con la distribución de los clústeres

Clúster	#A	#B	#C
Morado	2	0	20
Amarillo	33	0	0
Azul	0	42	0

Cuadro 6.3: Tabla de con distribución de Clústeres

6.4 Resultados

Al aplicar el algoritmo de homología persistente en estos datos, se pudo evidenciar que a medida que avanza el tiempo, empieza a haber indicios de ciclos o agujeros 1 - dimensionales. Estos también están presentes al reunir

toda la población. Como se menciona en [8] esto puede ser indicio de recombinación genética, lo cual ocurre cuando fragmentos de ADN de diferentes orígenes se intercambian durante la reproducción de un organismo. Por otra parte, pudimos ver como el algoritmo de Mapper es capaz de presentar cualidades globales del set de datos, como por ejemplo, la cantidad de grupos presente en este.

Capítulo 7

Conclusiones

El objetivo principal de este trabajo de grado era presentar una introducción al análisis topológico de datos mediante la revisión bibliográfica de dos de sus algoritmos más utilizados y culminar con la implementación de estos en un conjunto de datos reales.

Para lograr esto, primero se describió el concepto de símplice, los cuales sirvieron para posteriormente describir complejos simpliciales, al hacer esto, pudimos ver como complejo simplicial de Čech tiende a capturar mejor las propiedades locales del conjunto, mientras que el complejo simplicial de Vietoris-Rips es más sensible a la estructura global de los puntos con relación a la distancia.

Una vez definidos los conceptos anteriores, se introdujo la noción de homología simplicial, con la cual se definió posteriormente los grupos de homología. Esto nos ayudó a definir el concepto de números de Betti, y nos sirvió para caracterizar las propiedades topológicas de un espacio.

Posteriormente, definimos el concepto de homología persistente y se vieron dos algoritmos que ayudan a visualizar la representación de esta. En esta sección se da un resultado interesante en torno a los diagramas de persistencia, el cual nos dice que estos son estables bajo perturbaciones pequeñas.

Además de los algoritmos de homología persistente, se introdujo el algoritmo de Mapper, el cual a diferencia de los anteriores algoritmos, nos permite visualizar las estructuras topológicas presentes en conjuntos de datos complejos mediante una representación simplificada que facilita la interpretación y comprensión de los datos.

Bibliografía

[1] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[2] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 263–271, 2005.

[3] Anne Collins, Afra Zomorodian, Gunnar Carlsson, and Leonidas J Guibas. A barcode shape descriptor for curve point cloud data. *Computers & Graphics*, 28(6):881–894, 2004.

[4] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Proceedings 41st annual symposium on foundations of computer science*, pages 454–463. IEEE, 2000.

[5] Robert W Ghrist. *Elementary applied topology*, volume 1. Createspace Seattle, 2014.

[6] Allen Hatcher. *Algebraic topology.* , 2005.

[7] Sasan Karamizadeh, Shahidan M Abdullah, Azizah A Manaf, Mazdak Zamani, and Alireza Hooman. An overview of principal component analysis. *Journal of Signal and Information Processing*, 4(3B):173, 2013.

[8] Angélica Knudson, Felipe González-Casabianca, Alejandro Feged-Rivadeneira, Maria Fernanda Pedreros, Samanda Aponte, Adriana Olaya, Carlos F Castillo, Elvira Mancilla, Anderson Piamba-Dorado, Ricardo Sanchez-Pedraza, et al. Spatio-temporal dynamics of plasmodium falciparum transmission within a spatial unit on the colombian pacific coast. *Scientific reports*, 10(1):3756, 2020.

[9] Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In *Mathematical Software–ICMS 2014: 4th International Congress, Seoul, South Korea, August 5-9, 2014. Proceedings 4*, pages 167–174. Springer, 2014.

[10] James R Munkres. *Elements of algebraic topology.* CRC press, 2018.

[11] Raúl Rabadán and Andrew J Blumberg. *Topological data analysis for genomics and evolution: topology in biology*. Cambridge University Press, 2019.

[12] Gurjeet Singh, Facundo Mémoli, Gunnar E Carlsson, et al. Topological methods for the analysis of high dimensional data sets and 3d object recognition. *PBG@ Eurographics*, 2:091–100, 2007.

[13] Christopher Tralie, Nathaniel Saul, and Rann Bar-On. Ripser. py: A lean persistent homology library for python. *Journal of Open Source Software*, 3(29):925, 2018.

[14] H van Veen. Kepler mapper. https://github.com/MLWave/kepler-mapper, 2015.

[15] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 347–356, 2004.