Cálculo de grupos de homología usando forma normal de Smith

Luis Alejandro Aguiar Reina
Departamento de Matemáticas
Universidad Nacional de Colombia
luaguiarr@unal.edu.co

Nicolás Duque Molina
Departamento de Matemáticas
Universidad Nacional de Colombia
niduque@unal.edu.co

Marcos Pinzón Pardo
Departamento de Matemáticas
Universidad Nacional de Colombia
mpinzonpa@unal.edu.co

Abstract. Este proyecto tiene como objetivo calcular grupos de homología simplicial con coeficientes en $\mathbb{Z}$ a partir de complejos finitos, utilizando la forma normal de Smith como herramienta principal de reducción de matrices. A partir de la estructura algebraica de los complejos de cadenas asociados, se construyen las matrices de frontera y se obtiene una descripción explícita de los grupos de homología como grupos abelianos finitamente generados. La torsión y el rango libre de cada grupo se extraen directamente de la diagonal de la matriz reducida. Además, se desarrolla una implementación computacional en Python que automatiza el proceso y permite comparar resultados con el software SageMath. Este enfoque permite no solo ilustrar el vínculo entre la topología algebraica y el álgebra lineal entera, sino también explorar la torsión homológica como un invariante topológico relevante en distintos contextos.

Palabras Clave: Homología simplicial, topología algebraica, forma normal de Smith, Álgebra computacional, reducción de matrices enteras, homología computacional

Introducción

En topología algebraica, los grupos de homología permiten estudiar y clasificar espacios topológicos mediante invariantes algebraicos que capturan la presencia de componentes conexas, túneles, cavidades y estructuras de mayor dimensión. Aunque tradicionalmente se han utilizado coeficientes en un cuerpo como $\mathbb{Z}_2$ o $\mathbb{Q}$, el uso de coeficientes enteros, específicamente $\mathbb{Z}$, proporciona una descripción más completa al revelar tanto la parte libre como la torsión de los grupos de homología.

Históricamente, el desarrollo de la homología surgió en el siglo XIX como una respuesta a la necesidad de formalizar nociones intuitivas sobre la forma y la conectividad de los objetos geométricos. Figuras como Henri Poincaré, Emmy Noether y Leopold Kronecker contribuyeron significativamente a la consolidación del enfoque algebraico. Con el tiempo, la topología algebraica evolucionó hasta convertirse en una disciplina rigurosa con aplicaciones que van desde la geometría diferencial hasta la teoría de números.

En las últimas décadas, la emergencia del campo conocido como topología computacional ha revitalizado el interés por métodos efectivos de cálculo de invariantes topológicos. Entre ellos, la homología juega un papel central en el análisis de datos complejos. En particular, técnicas como el Análisis Topológico de Datos (TDA, por sus siglas en inglés) utilizan conceptos de la homología para extraer patrones robustos en conjuntos de datos de alta dimensión, sin requerir una estructura métrica explícita. Aplicaciones concretas se encuentran en áreas como biología computacional, redes neuronales, visión por computador, análisis de texto y finanzas.

En este contexto, el presente proyecto se propone calcular grupos de homología simplicial con coeficientes en $\mathbb{Z}$, utilizando la forma normal de Smith como herramienta clave. Esta técnica permite obtener, a partir de las matrices de frontera asociadas a un complejo simplicial, una descripción precisa de los grupos homológicos como grupos abelianos finitamente generados. Para ello, se desarrolla una implementación computacional en Python y se compara su desempeño con herramientas como SageMath. Además de fortalecer la comprensión algebraica de los invariantes topológicos, este enfoque brinda una base sólida para futuras aplicaciones en análisis de datos estructurados y topología aplicada.

I. Topología Algebraica y Homología

La topología algebraica surge como una disciplina que busca traducir problemas geométricos en términos algebraicos, permitiendo estudiar propiedades cualitativas de los espacios topológicos mediante estructuras como grupos, anillos y módulos. Uno de sus principales objetivos es construir invariantes topológicos que permitan distinguir espacios, detectar agujeros o caracterizar su comportamiento global de manera robusta frente a deformaciones continuas.

Para alcanzar este propósito, es necesario establecer una base conceptual que parte de la noción de espacio topológico y su estudio mediante herramientas como la homotopía, que captura equivalencias entre espacios bajo deformaciones suaves. Sin embargo, para obtener invariantes computables y aplicables, resulta más práctico restringirse a representaciones discretas, como los complejos simpliciales, que permiten modelar espacios mediante combinaciones finitas de vértices, aristas, triángulos y sus generalizaciones de mayor dimensión.

A partir de estos complejos, se construyen los grupos de cadenas, cuyos elementos combinan formalmente símplices orientados. La introducción de los operadores de frontera permite definir el complejo de cadenas, una sucesión de espacios vectoriales (o módulos) conectados por aplicaciones lineales. Los conceptos de ciclos (elementos sin frontera) y bordes (fronteras de elementos de dimensión superior) conducen naturalmente a la definición de los grupos de homología, que miden la "obstrucción" a que un ciclo sea un borde.

Esta sección tiene como objetivo presentar de forma clara y rigurosa los elementos fundamentales de la topología algebraica necesarios para el desarrollo del proyecto. Se abordarán desde la intuición geométrica hasta las formulaciones algebraicas, incluyendo ejemplos clásicos como la esfera, el toro y el plano proyectivo, para ilustrar la potencia de los grupos de homología como herramientas para estudiar la forma de los espacios.

I-A. Conceptos fundamentales de topología algebraica

Definición I.1. Una ***topología*** sobre un conjunto X es una colección τ de subconjuntos de X con las siguientes propiedades:

1. $\emptyset, X \in \tau$.
2. La unión arbitraria de elementos de τ está en τ.
3. La intersección finita de elementos de τ está en τ.

Un conjunto X para el que se ha definido una topología τ se llama ***espacio topológico***.

Nota I.1. Un elemento $U \in \tau$ se llama ***abierto***.

Ejemplo I.1. Dado $X = \{a, b, c\}$ podemos definir una topología sobre X como $\tau = \{\emptyset, \{b\}, \{a, b\}, \{b, c\}, X\}$.

Definición I.2. Sean X, Y espacios topológicos. Una función $f : X \to Y$ se dice continua si para todo abierto V de Y, se cumple que $f^{-1}[V]$ es un abierto de X.

Ejemplo I.2. Sea X un espacio topológico, consideramos el mapa identidad $id_X : X \to X$ tal que $id_X(x) = x$ para todo $x \in X$. Claramente esta función es continua.

Definición I.3. Sean X, Y espacios topológicos y sean $f, g : X \to Y$ funciones continuas. Se dice que f es ***homotópica*** a g ($f \simeq g$) si existe una aplicación continua

$$H : X \times [0, 1] \to Y$$

tal que para todo $x \in X$:

1. $H(x, 0) = f(x)$,
2. $H(x, 1) = g(x)$.

Esta aplicación H se llama una ***homotopía*** entre f y g. Se interpreta como una deformación continua de f en g a lo largo del parámetro $t \in [0, 1]$.

Ejemplo I.3. Considere las funciones $f, g : \mathbb{R} \to \mathbb{R}$ dadas por $f(x) = x$ y $g(x) = x^2$. Estas son homotópicas mediante la homotopía lineal:

$$\begin{aligned} H(x, t) &= (1 - t)f(x) + t(g(x)) \\ &= (1 - t)x + tx^2 \end{aligned}$$

Definición I.4. Sean X, Y dos espacios topológicos. Se dicen ***homotópicamente equivalentes*** si existen $f : X \to Y$ y $g : Y \to X$ funciones continuas tales que:

1. $f \circ g \simeq id_X$,
2. $g \circ f \simeq id_Y$.

Las funciones f, g son llamadas ***equivalencias homotópicas***.

La idea detrás de dos espacios homotópicamente equivalentes es deformar uno en el otro sin romper ni pegar alguno de ellos (como estirar o comprimir, pero no rasgar).
Una forma de pensarlo, es que los espacios son "lo mismo" siempre que sean homotópicamente equivalentes. Acá encontramos un primer problema de la topología algebraica, pues, el estudio de homotopía no siempre resulta sencillo; por ejemplo, para mostrar que dos espacios **no** son homotópicamente equivalentes, deberíamos mostrar que no existe equivalencia homotópica alguna. Del mismo modo, demostrar que si lo son parte de conjeturar un par de equivalencias homotópicas, lo cual no siempre resulta intuitivo o efectivo.
Por lo anterior, resulta mejor usar las invariantes homotópicas de un espacio, más adelante mostraremos que si dos espacios X, Y son tales que $X \simeq Y$, entonces sus grupos de homología son exactamente el mismo.

Ejemplo I.4. A continuación, daremos dos ejemplos intuitivos, sin embargo no profundizaremos en sus pruebas.

- Un disco $D^2 \subseteq \mathbb{R}^2$ es homotópicamente equivalente a un punto $\{x\}$, pues se puede contraer de forma continua el disco a su centro.
- Un cilindro $S^1 \times [0, 1]$ es homotópicamente equivalente a un círculo S^1, ya que la altura del cilindro se puede comprimir.

Una vez dadas estas herramientas básicas, podemos pasar a desarrollar la teoría en mayor profundidad. Sin embargo, la topología algebraica nuevamente "tropieza" cuando se trata del estudio general de homología. Por ello nos vamos a centrar en los complejos simpliciales, que resultan más amigables en su desarrollo teórico, además de ser aquellos que resultan útiles en el apartado computacional que realizamos más adelante.

I-B. Complejos Símpliciales

Definición I.5. Un ***símplice*** (algunas veces llamado simplex) es la generalización de un triángulo a dimensiones no negativas.
El k***-símplice*** se define como la colección

$$\{x \in \mathbb{R}^{k+1} | x_0 + \cdots + x_k = 1, \text{ y } x_i \geq 0 \text{ para } i = 0, \ldots, 1\}.$$

Es fácil verificar que para $k = 0$, el 0-símplice es el punto $1 \in \mathbb{R}$, para $k = 1$, el 1-símplice es la recta en $\mathbb{R}^2$ donde una vez fijo x_0 se determina únicamente $x_1 = 1 - x_0$ y para $k = 2$, el 2-símplice es un triángulo en el plano $0 = x_0 - x_1 - x_2 + 1$.

Definición I.6. La ***frontera*** del k-símplice (para $k > 0$) es el subconjunto del k-símplice donde al menos un $x_i = 0$.

Definición I.7. Las ***caras*** de un k-símplice son los $k-1$-símplices que forman su frontera. Otra forma de verlo es como un símplice generado por un subconjunto no vacío de sus vértices.
Una ***cara propia*** es una cara que no es todo el k-símplice.

Definición I.8. Sea K un k-símplice. Una orientación en de K es un orden en los 0-símplices en K, se escribe como $[v_0, \ldots, v_k]$. Otra forma de verlo, es como un orden en sus vértices.

Nota I.2.

- Diremos que dos orientaciones son la misma si difieren por una permutación par.
- Diremos que una orientación es opuesta a la otra si difieren por una permutación impar.

Definición I.9. Un ***complejo símplicial*** K es una colección finita de símplices en $\mathbb{R}^n$ que satisface dos propiedades:

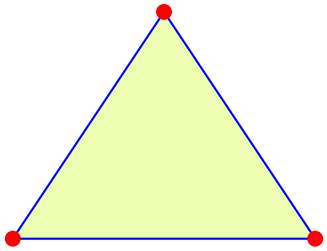
- **Cerrado bajo caras**. Todas las caras de K están en K
- **Intersecciones no degeneradas**. Si $\sigma_1, \sigma_2 \in K$ entonces $\sigma_1 \cap \sigma_2$ es vacío o una cara común de ambos.

De manera intuitiva, podemos pensar en construir un complejo símplicial K "pegando" una cara de un k-símplice a un $k-1$-símplice.

Definición I.10. De manera similar, definimos una orientación para un complejo símplicial K como un ordenamiento de los 0-símplices contenidos en K. Además, lo planteado en Nota I.2 se mantiene para este caso. Por tanto, para todo símplice en K hereda una orientación de K.

Ejemplo I.5. Un ejemplo de un complejo símplicial es un tríangulo relleno, pues este incluye:

- Tres 0-símplices (vértices),
- Tres 1-símplices (aristas),
- Un 2-símplice (tríangulo),
- Todas sus caras.



I-C. Grupos de Cadenas

Nota I.3. Sea S un complejo símplicial orientado, denotamos por S_k el conjunto de todos los k-símplices orientados en S, donde cada S_k hereda la orientación de S.

Definición I.11. Sea S un complejo simplicial, el ***grupo de k-cadenas*** con coeficientes enteros es el grupo libre abeliano generado por los k-símplices orientados de S, se denota por

$$C_k(S, \mathbb{Z}).$$

Tiene como base el conjunto de k-símplices orientados de S, $B = \{\sigma_1, \sigma_2, \ldots, \sigma_l\}$. De este modo, un elemento del grupo se llama k-cadena y es una combinación lineal formal de k-símplices con coeficientes enteros, es decir, para todo $\sigma \in C_k(S, \mathbb{Z})$ se tiene

$$\sigma = c_1\sigma_1 + c_2\sigma_2 + \cdots + c_l\sigma_l; c_i \in \mathbb{Z}$$

Nota I.4. Vamps a dar la condición que para todo símplice orientado, su "opuesto" es el símplice formado con la orientación opuesta. Es decir, si $\sigma_i = [v_1, v_2]$ es un 1-símplice en S, donde v_j es un 0-símplice en S para $j = 1, 2$, entonces $-\sigma = [v_2, v_1] \in C_k$

Una vez dicho eso, vamos a mostrar la estructura algebraica en este conjunto.

Estructura algebraica

- Dadas dos cadenas $\sigma = \sum c_i\sigma_i, \sigma' = \sum b_i\sigma_i$, definimos su **suma** como sigue

$$\sigma + \sigma' = \sum (c_i + b_i)\sigma_i.$$

- Vamos a tener como **elemento neutro** la cadena 0_{C_k}, donde todoso los coeficientes enteros $c_i = 0$.
- Para una cadena σ su **inverso** es la cadena $-\sigma$ definida previamente.

Ejemplo I.6. Sea X un complejo simplicial con 3 1-símplices, siendo estos:

- $[v_0, v_1]$,
- $[v_1, v_2]$,
- $[v_2, v_0]$

En realidad, este es un triángulo sin rellenar.
De este modo las 1-cadenas son combinaciones como

$$\sigma = 2[v_0, v_1] - 7[v_1, v_2] + 4[v_2, v_0]$$

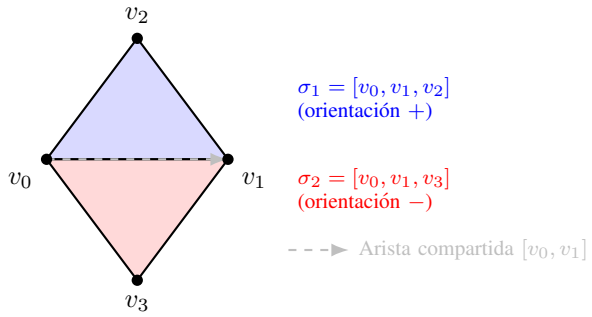
En este caso es sencillo darle una interpretación geométrica, pues los coeficientes indican las veces que se "recorre" una arista y la orientación de este recorrido.

Otro ejemplo sencillo de ver, sería el caso $C_0(X, \mathbb{Z})$ que consiste en combinaciones lineales formales de vértices. Uno un poco más díficil de visualizar, sin embargo, intuitivo es el grupo de las 2-cadenas $C_2(X, \mathbb{Z})$, consistiendo

de combinaciones lineales formales de triángulos rellenos. Pensemos un poco en su interpretación geométrica, para ello tomemos un caso particular

$$\sigma = [v_0, v_1, v_2] - [v_0, v_1, v_3]$$

En este caso, imaginamos que ambos triángulos comparten una arista (v_0, v_1), luego, el primer triángulo se agrega, mientras el otro se resta



I-D. Operadores de frontera

La intuición detrás del operador de frontera es tener una aplicación lineal con entrada un k-símplice orientado y poder transformarlo en una combinación lineal de sus caras de dimensión inferior (Elementos de C_{k-1}).

Definición I.12. Para un k-símplice orientado σ, definimos

$$\partial_k([v_0, \ldots, v_k]) = \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_k].$$

donde $[v_0, \ldots, \hat{v}_i, \ldots, v_k]$ es la cara de σ que no contiene al 0-símplice v_i. En la expresión, la utilidad del $(-1)^i$ sirve para guardar el signo que viene dado por la orientación inducida, verifica que se forme un perímetro cerrado.

Teorema I.1. Para cada entero no negativo k, el operador de frontera $\partial_k : C_k(X, \mathbb{Z}) \to C_{k-1}(X, \mathbb{Z})$ es un homomorfismo de grupos abelianos.

Demostración. Veamos que $\partial_k(0_{C_k}) = 0_{C_{k-1}}$.
Dado $0_{C_k} \in C_k(X, \mathbb{Z})$, aplicamos el operador frontera como sigue

$$\begin{aligned} \partial_k(0_{C_k}) &= \partial_k\Big(\sum_{j=1}^{l} 0 \cdot \sigma_j\Big) \\ &= \sum_{j=1}^{l} 0 \cdot \partial_k(\sigma_j) \end{aligned}$$

Ahora, sabemos que $\partial_k(\sigma_j)$ son $(k-1)$-símplices, luego, se sigue que la expresión es en efecto $0_{C_{k-1}}$.
Ahora, dadas $c_1, c_2 \in C_k(X, \mathbb{Z})$ dos k-cadenas, podemos escribirlas como $c_1 = \sum_{j=1}^{l} n_j \sigma_j$ y $c_2 = \sum_{j=1}^{l} p_i \sigma_i$, de este modo su suma es

$$\begin{aligned} c_1 + c_2 &= \sum_{j=1}^{l} n_j \sigma_j + \sum_{j=1}^{l} p_j \sigma_j \\ &= \sum_{j=1}^{l} (n_j + p_j) \sigma_j. \end{aligned}$$

Ahora, aplicando el operador de frontera tenemos

$$\begin{aligned}\partial_k(c_1+c_2) &= \partial_k\Big(\sum_{j=1}^{l}(n_j+p_j)\sigma_j\Big)\\ &= \sum_{j=1}^{l}(n_j+p_j)\partial_k(\sigma_j)\\ &= \sum_{j=1}^{l} n_j\partial_k(\sigma_j)+p_j\partial_k(\sigma_j)\\ &= \sum_{j=1}^{l} n_j\partial_k(\sigma_j)+\sum_{j=1}^{l} p_j\partial_k(\sigma_j)\\ &= \partial_k\Big(\sum_{j=1}^{l} n_j\sigma_j\Big)+\partial_k\Big(\sum_{j=1}^{l} p_j\sigma_j\Big)\\ &= \partial_k(c_1)+\partial_k(c_2)\end{aligned}$$

Así concluímos lo que queríamos demostrar. □

Nota I.5. La prueba anterior se pudo hacer si hubiéramos asumido un teorema universal de los grupos libres abelianos, el cual establece que si tenemos una función definida sobre los generadores del grupo la cual mapea a otro grupo libre abeliano, entonces esta función se extiende de manera única a un homomorfismo de grupos sobre todo el grupo libre.

Ejemplo I.7. A continuación, daremos una serie de ejemplos, sin embargo no será a profundidad.

- Caso para un 1-símplice.
  Un 1-símplice es un segmento de recta, digamos $[v_0, v_1]$, veamos el operador de frontera en este caso.

  $$\begin{aligned}\partial_1([v_0,v_1]) &= (-1)^0[v_1]+(-1)^1[v_0]\\ &= [v_1]-[v_0]\end{aligned}$$

  Esto significa que la frontera de un segmento de recta son dos puntos, el punto final $[v_1]$ con signo positivo y el punto inicial $[v_0]$ con signo negativo. Esto es natural, pues v_1 es el "destino" mientras que v_0 es el "punto de partida" de la orientación del segmento.
- Caso para un 2-símplice.
  Un 2-símplice es un triángulo, digamos $[v_0, v_1, v_2]$, veamos el operador de frontera.

  $$\begin{aligned}\partial_2([v_0,v_1,v_2]) &= (-1)^0[v_1,v_2]+(-1)^1[v_0,v_2]+(-1)^2[v_0,v_1]\\ &= [v_1,v_2]-[v_0,v_2]+[v_0,v_1]\end{aligned}$$

  Si entendemos geométricamente, esto se trata de los tres lados del triángulo, donde "recorremos" en sentido opuesto el lado $[v_0, v_2]$. Si imaginamos que vamos a "caminar" a lo largo del mismo, recorriendo lados en el orden indicado por los signos y el orden de los vértices, partiríamos de v_0 y llegaríamos a v_1, posterior a ello iríamos de v_1 y llegaríamos a v_2 y finalmente de v_2 llegaríamos a v_0, es decir, se forma un ciclo cerrado.

A continuación, daremos uno de los resultados teóricos fundamentales del operador de frontera, el cual nos será de ayuda más adelante.

Teorema I.2. Para todo entero $k \geq 1$, la composición de dos operadores de frontera consecutivos es el homomorfismo nulo.

$$\partial_{k-1} \circ \partial_k \equiv 0$$

Es decir, para toda k-cadena $c \in C_k(X, \mathbb{Z})$, se cumple $\partial_{k-1}(\partial_k(c)) = 0$.

Demostración. Podemos realizar la prueba solamente sobre los generadores de $C_k(X, \mathbb{Z})$, pues, por la Nota I.5, si tenemos que la función es 0 en ellos, se nos garantiza el mapeo de manera única a un homomorfismo de grupos, en este caso, el homomorfismo nulo.

De este modo, sea $\sigma = [v_0, v_1, \dots, v_k]$ un k-símplice orientado.
Aplicamos así el operador de frontera y tenemos

$$\partial_k(\sigma) = \sum_{i=0}^{k} (-1)^i [v_0, \dots, \hat{v}_i, \dots v_k]$$

Cada término de la suma resulta ser un $(k-1)$-símplice. Llamemos a cada uno de estos $\sigma_i = [v_0, \dots, \hat{v}_i, \dots, v_k]$, de este modo tenemos $\partial_k(\sigma) = \sum_{i=1}^{k} (-1)^i \sigma_i$.
Ahora, apliquemos ∂_{k-1} a cada σ_i, primero, escribimos $\sigma_i = [u_0, \dots, u_{k-1}]$ y ahora

$$\partial_{k-1}(\sigma_i) = \sum_{j=0}^{k-1} (-1)^j [u_0, \dots, \hat{u}_j, \dots, u_{k-1}]$$

Entendamos que significa eliminar u_j de la secuencia de σ_i, para ello vemos dos casos:

- Si $j < i$, el j-ésimo vértice de σ_i es v_j,
- Si $j \geq i$, el j-esimo vértice de σ_i es v_{j+1} (pues v_i fue eliminado).

Entonces, eliminar el j-ésimo vértice de σ_i, eliminamos dos vértices del σ original, v_i y v_j o v_{j+1}.
Ahora sustituimos $\partial_{k-1}(\sigma_i)$ en la expresión de $\partial_k(\sigma)$ y tenemos

$$\begin{aligned} \partial_{k-1}(\partial_k(\sigma)) &= \partial_{k-1}\Big(\sum_{i=0}^{k} (-1)^i [v_0, \dots, \hat{v}_i, \dots, v_k]\Big) \\ &= \sum_{i=0}^{k} (-1)^i \partial_{k-1}([v_0, \dots, \hat{v}_i, \dots, v_k]). \end{aligned}$$

Cada término en la suma es un $(k-2)$-símplice de la forma $[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k]$. Si elegimos un término génerico en el que $x \neq y$, este $(k-2)$-símplice aparece en la suma de dos maneras:

- Si v_x fue eliminado primero para formar σ_x y luego se eliminó v_y,
- Si v_y fue eliminado primero para formar σ_y y luego se eliminó v_x.

Podemos ahora analizar ambos casos por separado. Asumiendo $x < y$ sin perdida de generalidad.

- **Caso 1**.
  Primero, se elimina v_x y se forma $\sigma_x = [v_0, \dots, \hat{v}_x, \dots, v_k]$ con signo $(-1)^x$.
  Posterior a eso, se elimina v_y de σ_x, por como asumimos que $x < y$, el vértice v_y se encuentra en la posición $(y-1)$ de σ_x. Por lo tanto, el signo de esta eliminación es $(-1)^{y-1}$.
  El término que resulta en este caso es $[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k]$ con signo $(-1)^x(-1)^{y-1} = (-1)^{x+y-1}$.
- **Caso 2**.
  Primero se elimina v_y y se forma $\sigma_y = [v_0, \dots, \hat{v}_y, \dots, v_k]$ con signo $(-1)^y$.
  Luego se elimina v_x de σ_y, en la secuencia de σ_y el vértice v_x aparece en la posición x por como asumimos $x < y$, por lo tanto el signo de esta eliminación es $(-1)^x$.
  El término que resulta en este caso es $[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k]$ con signo $(-1)^y(-1)^x = (-1)^{x+y}$.

Así, en la sumatoria cada $(k-2)$-símplice con vértices eliminados v_x, v_y aparece dos veces en la sumatoria, si sumamos su primera aparición con la segunda tenemos

$$\begin{aligned} &(-1)^{x+y-1}[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k] + (-1)^{x+y}[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k] \\ &= (-1)^{x+y-1}[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k] - (-1)^{x+y-1}[v_0, \dots, \hat{v}_x, \dots, \hat{v}_y, \dots, v_k] \\ &= 0 \end{aligned}$$

Es decir, todos los $(k-2)$-símplices se cancelan en pares, por tanto $\partial_{k-1}(\partial_k(\sigma)) = 0$.
De este modo $\partial_{k-1} \circ \partial_k \equiv 0$. □

I-E. Matrices de frontera

Ahora, daremos la noción de matrices de frontera. Ellas nos permiten representar y calcular el operador ∂_k de manera concreta y computacional, herramienta que luego será de utilidad en el cálculo de los grupos de homología. Como el operador de frontera es un homomorfismo entre grupos abelianos libres. Sabemos del álgebra lineal que

cualquier homomorfismo entre módulos libres sobre $\mathbb{Z}$ se puede representar como una matriz una vez fijas las bases para los espacios dominio y codominio.
En nuestro caso, es claro que el dominio es $C_k(X,\mathbb{Z})$ y el codominio $C_{k-1}(X,\mathbb{Z})$. Por lo realizado en la sección anterior, sabemos que sus bases son los símplices orientados. Así, procedemos a explicar su construcción.
Supongamos que tenemos N_k k-símplices orientados en X, siendo estos: $\sigma_1^k,\ldots,\sigma_{N_k}^k$.
Del mismo modo, tenemos N_{k-1} $(k-1)$-símplices orientados en X, siendo estos: $\sigma_1^{k-1},\ldots,\sigma_{N_{k-1}}^{k-1}$.
Así, la matriz M_k tendrá dimensiones $N_{k-1}\times N_k$.
La entrada $[M_k]_{i,j}$ es el coeficiente del $(k-1)$-símplice σ_i^{k-1} cuando se expresa $\partial_k(\sigma_j^k)$ como una combinación lineal de los $(k-1)$-símplices.
Formalmente

$$\partial_k(\sigma_j^k)=\sum_{i=1}^{N_{k-1}}[M_k]_{i,j}\cdot\sigma_i^{k-1}$$

La pregunta natural ahora es, ¿Cómo calcular $[M_k]_{i,j}$? La respuesta la damos a continuación.
La entrada $[M_k]_{i,j}$ es $-1, 1$ o 0.

- $[M_k]_{i,j}=1$ si σ_i^{k-1} es una cara de σ_j^k y tiene la misma orientación inducida,
- $[M_k]_{i,j}=-1$ si σ_i^{k-1} es una cara de σ_j^k y tiene la orientación opuesta a la inducida,
- $[M_k]_{i,j}=0$ si σ_i^{k-1} no es una cara de σ_j^k.

Ejemplo I.8. Vamos a considerar el ejemplo que ya hemos venido trabajando previamente, el triángulo relleno. Sabemos que este cuenta de 3 0-símplices (sus vértices).
De este modo, tomemos como base para $C_0(X,\mathbb{Z})$ el conjunto $\{[v_0],[v_1],[v_2]\}$.
De igual manera, cuenta con 3 1-símplices (sus aristas). Vamos a elegir orientaciones para estos como sigue:

- $e_0=[v_0,v_1]$,
- $e_1=[v_1,v_2]$,
- $e_2=[v_0,v_2]$

Así, elegimos como base para $C_1(X,\mathbb{Z})$ el conjunto $\{e_0,e_1,e_2\}$.
Finalmente, tenemos el 2-símplice $\tau_0=[v_0,v_1,v_2]$ y por tanto nuestra base para $C_2(X,\mathbb{Z})$ es el conjunto $\{\tau_0\}$.
Ahora, construyamos las matrices de frontera.

- $M_1: C_1(X,\mathbb{Z})\to C_0(X,\mathbb{Z})$.
  El tamaño de esta matriz es $N_0\times N_1=3\times 3=9$. Computamos y tenemos:
  - $\partial_1(e_0)=\partial([v_0,v_1])=[v_1]-[v_0]$,
  - $\partial_1(e_1)=\partial([v_1,v_2])=[v_2]-[v_1]$,
  - $\partial_1(e_2)=\partial([v_0,v_2])=[v_2]-[v_0]$

  Por lo tanto tenemos que la matriz M_1 es

$$M_1=\begin{pmatrix}-1&0&-1\\1&-1&0\\0&1&1\end{pmatrix}$$

  Donde las filas se indexan por 0-símplices y las columnas por 1-símplices.

- $M_2: C_2(X,\mathbb{Z})\to C_1(X,\mathbb{Z})$.
  Ahora su dimensión es $N_1\times N_2=3\times 1$. Aplicamos el operador de frontera a τ_0 y tenemos $\partial_2(\tau_0)=\partial_2([v_0,v_1,v_2])=[v_1,v_2]-[v_0,v_2]+[v_0,v_1]=e_0+e_1-e_2$. De este modo:

$$M_2=\begin{pmatrix}1\\1\\-1\end{pmatrix}$$

Con este ejemplo podemos verificar el teorema Teorema I.2 en términos de matrices como sigue:

$$M_1 M_2 = \begin{pmatrix} -1 & 0 & -1 \\ 1 & -1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Con lo que concluimos nuestro ejemplo.

I-F. Complejos de cadenas

Un complejo de cadenas es una estructura algebraica que encapsula la información sobre las diferentes dimensiones de un espacio simplicial y cómo sus componentes de una dimensión se relacionan con las de la dimensión adyacente a través de sus fronteras.
A partir de ahora escribiremos $C_k(X, \mathbb{Z})$ como $C_k(X)$ o C_k.

Definición I.13. Un **complejo de cadenas** es una sucesión de grupos abelianos (o módulos) C_k y de homomorfismos de grupos $\partial_k : C_k \to C_{k-1}$ tales que la composición de dos homomorfismos consecutivos es el homomorfismo nulo.
Se representa usualmente de la siguiente manera:

$$\ldots \xrightarrow{\partial_{k+2}} C_{k+1} \xrightarrow{\partial_{k+1}} C_k \xrightarrow{\partial_k} C_{k-1} \xrightarrow{\partial_{k-1}} \ldots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0$$

Donde:

- C_k corresponde al grupo de k-cadenas del complejo simplicial X,
- ∂_k es el operador de frontera que definimos previamente,
- $\partial_{k-1} \circ \partial_k = 0$. Esta es la condición fundamental del complejo de cadenas. En términos de teoría de grupos, significa $\mathrm{Im}(\partial_k) \subseteq \ker(\partial_{k-1})$

I-G. Ciclos y Bordes

La condición mencionada anteriormente ($\mathrm{Im}(\partial_k) \subseteq \ker(\partial_{k-1})$) es la que nos permite definir dos subgrupos de alta importancia dentro de cada grupo de cadenas $C_k(X)$, siendo estos: los ciclos y los bordes (o fronteras).

- **Grupo de ciclos.**
  El grupo de k-ciclos, denotado por $Z_k(X)$ (o simplemente Z_k), es el kernel del operador de frontera ∂_k. Es decir

  $$Z_k(X) = \ker(\partial_k) = \{x \in C_k(X) | \partial_k(x) = 0\}.$$

  Geométricamente, podemos interpretar un k-ciclo como una k-cadena cuya frontera es nula. Esto se traduce en combinaciones de símplices que forman "caminos cerrados" o "formas sin borde".
  - 0-ciclos (Z_0): Son combinaciones de puntos. Un punto $[v_0]$ tiene $\partial_0([v_0]) = 0$ por convención al ser visto como un grupo trivial. Sin embargo, en contextos más amplios, los 0-ciclos no necesariamente son 0. Un 0-ciclo es una combinación de puntos donde la suma de los coeficientes es cero, por ejemplo, $[v_1] - [v_0]$.
  - 1-ciclos (Z_1): Son combinaciones de segmentos que forman caminos cerrados. Por ejemplo, en un triángulo con lados $e_0 = [v_0, v_1]$, $e_1 = [v_1, v_2]$ y $e_2 = [v_0, v_2]$, la cadena $e_0 + e_1 - e_2$ es un 1-ciclo pues $\partial_1(e_0 + e_1 - e_2) = 0$.
  - 2-ciclos (Z_2): Son combinaciones de triángulos (o superficies) que forman "superficies sin borde". Por ejemplo, la esfera hueca puede ser vista como un 2-ciclo.

  Ahora, pasamos a definir el grupo de bordes.

- **Grupo de bordes.**
  El grupo de k-bordes (o k-fronteras) denotado por $B_k(X)$ (o simplemente B_k), es la imagen del operador de frontera ∂_{k+1}. Es decir

  $$B_k(X) = \mathrm{Im}(\partial_{k+1}) = \{x \in C_k(X) | x = \partial_{k+1}(y) \text{ para } y \in C_{k+1}(X)\}.$$

Geométricamente, podemos interpretar un k-borde como una k-cadena que es la frontera de una $(k+1)$-cadena. Los bordes son los contornos de objetos de una dimensión superior.

- 0-bordes (B_0): La frontera de un segmento $[v_0, v_1]$ es $[v_1] - [v_0]$, que es un 0-borde.
- 1-bordes (B_1): La frontera de un triángulo $[v_0, v_1, v_2]$ es $[v_1, v_2] - [v_0, v_2] + [v_0, v_1]$. Este es el borde de un triángulo en dos dimensiones.
- 2-bordes (B_2): La frontera de un tetraedro (3-símplice) es una combinación de sus caras triangulares, formando un 2-borde.

Como ya probamos la propiedad fundamental de los complejos de cadenas, tenemos que todo borde es un ciclo. Si algo es la frontera de un objeto de dimensión mayor, entonces su propia frontera es 0 (forma un camino cerrado). Sin embargo, no todo ciclo es un borde. Aquí es donde aparece el concepto de "agujero".

I-H. Grupos de Homología

Los grupos de homología se definen precisamente para capturar estas diferencias. Miden la "cantidad" y "tipo" de agujeros en un espacio. Los grupos de homología son invariantes topológicos que nos permiten distinguir espacios topológicos que no se pueden deformar continuamente uno en el otro. Su construcción se basa en la idea de cuantificar los "agujeros" de un espacio en diferentes dimensiones.

Definición I.14. El k-ésimo grupo de homología simplicial de un complejo simplicial X con coeficientes enteros denotado por $H_k(X;\mathbb{Z})$ (o simplemente H_k) se define como el grupo cociente de los k-ciclos por los k-bordes. Es decir

$$H_k(X;\mathbb{Z}) = \frac{Z_k(X)}{B_k(X)} = \frac{\ker(\partial_k)}{\mathrm{Im}(\partial_{k+1})}.$$

Un elemento no trivial en $H_k(X;\mathbb{Z})$ representa un agujero de dimensión k en el espacio X.
Al tomar el cociente, estamos identificando (o considerando triviales) los ciclos que pueden ser "contraidos" (o "rellenados"). Lo que queda en el grupo de homología son precisamente las clases de ciclos que representan agujeros "genuinos".

I-H1. Ejemplos clásicos de grupos de homología

Calcular los grupos de homología directamente de la definición puede ser laborioso, pero los resultados para espacios comunes son bien conocidos y revelan su "forma" de una manera algebraica. Cuando escribamos 0 como grupo, hacemos referencia al trivial.

- **Considerando la esfera S^n**:
  - S^0 (dos puntos): $H_0(S^0;\mathbb{Z}) \cong \mathbb{Z} \oplus \mathbb{Z}$ (esto describe las dos componentes conexas). Para $k \geq 1$ tenemos $H_k(S^0;\mathbb{Z}) \cong 0$.
  - S^1 (círculo): $H_0(S^1;\mathbb{Z}) \cong \mathbb{Z}$ (esto describe su componente conexa). $H_1(S^1;\mathbb{Z}) \cong \mathbb{Z}$ (describe un "agujero" 1 dimensional, para $k \geq 2$, $H_k(S^1;\mathbb{Z}) \cong 0$.
  - S^2 (superficie de la esfera): $H_0(S^2;\mathbb{Z}) \cong \mathbb{Z}$, $H_1(S^2;\mathbb{Z}) \cong 0$, $H_2(S^2;\mathbb{Z}) \cong \mathbb{Z}$ (un hueco de dos dimensiones, el volumen que encierra). Para $k \geq 3$ $H_k(S^2;\mathbb{Z}) \cong 0$.
  - Generalizando para S^n, tenemos: $H_0(S^n;\mathbb{Z}) \cong \mathbb{Z}$ y $H_n(S^n;\mathbb{Z}) \cong \mathbb{Z}$ (el agujero n dimensional que encierra). Para $k \neq 0, n$ $H_k(S^n;\mathbb{Z}) \cong 0$.
- **El toro T^2**:
  - $H_0(T^2;\mathbb{Z}) \cong \mathbb{Z}$ pues tiene una componente conexa.
  - $H_1(T^2;\mathbb{Z}) \cong \mathbb{Z} \oplus \mathbb{Z}$ pues tiene dos agujeros de una dimensión, uno al rededor del cuerpo del toro y otro a través del agujero central.
  - $H_2(T^2;\mathbb{Z}) \cong \mathbb{Z}$ pues tiene un agujero de dos dimensiones encerrado por la superficie del toro.
  - Para $k \geq 3$, tenemos $H_k(T^2;\mathbb{Z}) \cong 0$.
- **El plano proyectivo real $\mathbb{PR}^2$**:
  Este ejemplo resulta crucial pues es un ejemplo donde aparece la torsión. Se puede visualizar como un disco con su frontera identificada con la frontera de otro disco después de unn giro.
  - $H_0(\mathbb{PR}^2;\mathbb{Z}) \cong \mathbb{Z}$,
  - $H_1(\mathbb{PR}^2;\mathbb{Z}) \cong \mathbb{Z}_2$. Esto se puede interpretar como un agujero que no se puede rellenar a menos que se recorra dos veces. Es un ejemplo de orientación no trivial que se cancela al doblarla.

- Para $k \geq 2$, tenemos $H_k(\mathbb{PR}^{\not\vDash}; \mathbb{Z}) \cong 0$

I-H2. Descomposición en suma directa para grupos de homología

Una de las propiedades más importantes de los grupos de homología con coeficientes enteros es que son grupos abelianos finitamente generados. Esto significa que el Teorema Fundamental de los Grupos Abelianos Finitamente Generados se aplica a ellos.

Este teorema establece que cada grupo abeliano finitamente generado G es isomorfo a una suma directa de grupos cíclicos, es decir

$$G \cong \mathbb{Z}^r \oplus \mathbb{Z}_{p_1^{a_1}} \oplus \mathbb{Z}_{p_2^{a_2}} \oplus \cdots \oplus \oplus \mathbb{Z}_{p_k^{a_k}}$$

Aplicado a los grupos de homología, significa que podemos descomponerlos en dos partes

1. **Parte libre de torsión.**
   Es la componente $\mathbb{Z}^r$ de la descomposición del grupo de homología. Su rango r se conoce como el número de Betti b_k.
   El número de Betti cuenta el número de agujeros de dimensión k que no peuden se rellenados y que son de tipo sin torsión. Estos agujeros corresponden a ciclos que no son bordes y que no se cancelan a sí mismos al repetirlos. Representan la conectividad del espacio de una manera sin auto-intersecciones.
2. **Parte de torsión.**
   Los factores $\mathbb{Z}_{p_j^{a_j}}$ son grupos cíclicos finitos donde p_j es un número primo. La parte de torsión detecta agujeros en el espacio que solo se vuelven triviales (o que pueden ser rellenados) después de ser recorridos un cierto número de veces. Estos fenómenos a menudo se dan por la orientación del espacio o la presencia de auto-intersecciones.

II. Cálculo de grupos de homología con la forma normal de Smith

Tras haber establecido los fundamentos conceptuales de la homología simplicial, esta sección se centra en el cálculo explícito de los grupos de homología con coeficientes en $\mathbb{Z}$. A diferencia del caso sobre un cuerpo, donde los espacios de cadenas forman espacios vectoriales y los grupos de homología pueden calcularse mediante técnicas estándar de álgebra lineal, trabajar sobre $\mathbb{Z}$ introduce una estructura mucho más rica: los grupos resultantes pueden tener tanto una parte libre como una parte de torsión.

Para acceder a esta estructura completa, se utiliza la forma normal de Smith, una herramienta central del álgebra lineal entera que permite reducir matrices de manera que se explicite la descomposición del módulo cociente $\ker \partial_n / \operatorname{im} \partial_{n+1}$ como grupo abeliano finitamente generado. El cálculo de esta forma se basa en el llamado algoritmo de Smith-MacMillan, que consiste en una sucesión de transformaciones elementales sobre filas y columnas, manteniendo la equivalencia del módulo representado.

El fundamento teórico de este procedimiento se apoya en resultados clásicos de la teoría de módulos libres finitamente generados sobre dominios de ideales principales (DIP). En particular, el teorema de estructura de estos módulos garantiza que cualquier módulo sobre un DIP —como $\mathbb{Z}$— puede descomponerse como suma directa de un módulo libre y un módulo de torsión. Este teorema es precisamente lo que permite interpretar la forma normal de Smith como una descripción precisa de la homología.

En esta sección se presentará el algoritmo, se explicará su relación con los grupos de homología y se aplicará a ejemplos concretos de complejos simpliciales. Se mostrará cómo a partir de las matrices de frontera es posible obtener la estructura completa de los grupos de homología, incluyendo los invariantes de torsión, que no pueden detectarse cuando se trabaja con coeficientes en un cuerpo.

II-A. Forma normal de Smith

La forma normal de Smith es una forma diagonal especial a la que se puede reducir cualquier matriz de enteros mediante operaciones de fila y columna invertibles. Es particularmente útil en la teoría de módulos sobre dominios de ideales principales (como los enteros), por lo que resulta ideal para nuestros grupos de cadenas.

Definición II.1. Sea $A \in M^{n\times m}(\mathbb{Z})$. La forma normal de Smith de la matriz A es una matriz diagonal $D \in M^{n\times m}(\mathbb{Z})$ que se obtiene a partir de A mediante una secuencia finita de operaciones elementales de fila y columna sobre $\mathbb{Z}$. La matriz D es de la forma:

$$D = \begin{pmatrix} d_1 & 0 & \dots & \dots & 0 \\ 0 & d_2 & \dots & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & d_r & 0 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

Donde:

- Los d_i son enteros no negativos, únicos salvo por multiplicación por -1,
- Los d_i satisfacen la condición de divisibilidad $d_1|d_2|\dots|d_r$,
- r es el rango de la matriz A,
- Las entradas fuera de la diagonal principal son 0.

Sea A una matriz $n \times m$ con entradas en un dominio de ideales principales R.
Definimos $\Delta_k(A)$ como el máximo común divisior de todos los menores de orden $k \times k$ de la matriz A. Donde un menor $k \times k$ es el determinante de una submatriz cuadrada $k \times k$ obtenida al seleccionar k filas y k columnas de la matriz A.

Nota II.1. Si D es la forma normal de Smith de A con elementos diagonales $d_1, \dots, d_r$, entonces:

1. $\Delta_k(A) = d_1 \cdot d_2 \cdot \dots \cdot d_k$ para $1 \leq k \leq r$.
2. $\Delta_k(A) = 0$ para $k > r$.

Podemos entender esto como sigue: $\Delta_k(A)$ es el ideal principal generado por g_k, donde g_k es el máximo común divisor de todos los menores $k \times k$ de A.
Dada esta interpretación, tenemos que $d_1 \cdot d_2 \dots d_k$ es un generador de $\Delta_k(A)$ cuando $1 \leq k$ y $\Delta_k(A) = \{0\}$ si $k > r$.

Lema II.1. La forma normal de Smith de una matriz A satisface lo siguiente:

1. La forma normal de Smith D es única para una matriz dada, salvo por el signo de cada uno de los elementos d_i. Los elementos d_i se llaman *divisores elementales* o *factores invariantes*.
2. **Operaciones elementales.** Las operaciones elementales admitidas sobre $\mathbb{Z}$ son las siguientes:
   - Intercambio de filas y columnas,
   - Multiplicar una fila o columna por ± 1,
   - Sumar un múltiplo entero de una fila (o columna) a otra fila (o columna). Estas operaciones corresponden a multiplicar la matriz A por matrices P y Q invertibles (con entradas enteras) tales que $PAQ = D$

Demostración.

- **Unicidad de la forma normal de Smith**.
  Vamos a demostrar que los $\Delta_k(A)$ son invariantes bajo operaciones elementales de filas y columnas. En este caso, asumiremos siempre operaciones elementales de filas, pues no se pierde la generalidad al hablar de columnas.
  1. **Permutaciones de filas/columnas.**
     Si tenemos un menor $k \times k$ y se permuta una fila del mismo, este resulta ser también un menor $k \times k$, luego, la acción de permutación no afecta el conjunto, solo el orden o signo de algunos, pero esto no afecta el ideal que generan.
  2. **Multiplicación de fila/columna por unidad.**
     Si una fila se multiplica por una unidad u, todos los menores $k \times k$ que incluyen dicha fila se multiplican por u. Como $\mathrm{mcd}(u \cdot a, u \cdot b, \dots, u \cdot \alpha) = u \cdot \mathrm{mcd}(a, b, \dots, \alpha)$. Así el ideal $\langle g_k \rangle$ se convierte en el ideal $\langle u \cdot g_k \rangle$, sin embargo, como $\mathbb{Z}$ es un dominio de ideales principales, sabemos que $\langle g_k \rangle = \langle u \cdot g_k \rangle$.

3. **Adición de un múltiplo de una fila a otra.**
   Sea M' un menor de orden k que involucra la fila R_i y no la fila R_j, este se transforma en el menor $M' + cM''$, donde M'' es el menor obtenido de reemplazar R_i por R_j. Como en un dominio de ideales principales tenemos $\mathrm{mcd}(a, b) = \mathrm{mcd}(\mathrm{a}, \mathrm{b} + \mathrm{c} \cdot \mathrm{a})$. El ideal generado por los menores de la nueva matriz resulta ser el mismo que el ideal original.

Ahora, si D es la forma normal de Smith de A, podemos ver los menores de D de una forma muy simple:

- Si $k > r$, tenemos que un menor $k \times k$ de D contiene un 0 en su diagonal, luego su determinante es 0 y por tanto $\Delta_k(D) = 0$.
- Si $k \leq r$, un menor $k \times k$ de D no contiene ningún 0 en su diagonal, solo los menores que consisten de los primeros k elementos diagonales $d_1, \ldots, d_k$ son no nulos. El determinante de tal menor es $d_1 d_2 \cdots d_k$. Todos los demás menores que incluyan elementos no diagonales tienen determinante 0, luego $\Delta_k(D) = d_1 d_2 \cdots d_k$ cuando $1 \leq k \leq r$.

Dado que A y D son matrices equivalentes por fila y columna, y las operaciones elementales no cambian los ideales, entonces $\Delta_k(D) = \Delta_k(A)$ para todo k. Más aún, como $d_1 | d_2 | \ldots | d_r$, podemos recuperar los d_i de manera recursiva y única (salvo producto por unidad) como sigue:

- $d_1 = \Delta_1(A)$,
- $d_k = \frac{\Delta_k(A)}{\Delta_{k-1}(A)}$ para $k > 1$.

Dado que los ideales $\Delta_k(A)$ son determinados de manera única a partir de A y los d_i se obtienen a partir de los mismos, los d_i deben ser únicos salvo producto por unidad en $\mathbb{Z}$.

La prueba del segundo hecho radica en un resultado fundamental de la teoría de matrices, sin embargo, debemos mostrar que las operaciones elementales son en efecto suficientes para reducir cualquier matriz a su forma normal de Smith. Para ello, mostraremos el algoritmo que usando solo dichas operaciones reduce toda matriz a su forma normal en la siguiente subsección. □

II-B. El Algoritmo de Smith-MacMillan

A continuación daremos un bosquejo del algortimo, pseudocódigo del mismo y una implementación en python para el mismo.

II-B1. Bosquejo del algoritmo.

Este algoritmo procede inductivamente. Consideramos una matriz A de tamaño $m \times n$:

1. Encontrar el elemento no nulo de menor valor absoluto en toda la matriz. Usar operaciones de fila/columna para mover este elemento a la posición $(1, 1)$. Si la matriz es nula, ya terminamos.
2. Hacer que este elemento divida a todos los elementos de la primera fila y primera columna. Si no lo hace, usar el algoritmo euclidiano. Por ejemplo, si $a_{1,1}$ no divide a $a_{i,j}$, podemos hacer $C_j \leftarrow C_j - qC_1$, donde $a_{1,j} = q \cdot a_{1,1} + r$. Si $r = 0$, ya lo divide. Si $r \neq 0$, el nuevo elemento $a_{1,j} = r < |a_{1,1}|$, por lo tanto repetimos el primer paso para llevar a la posición $(1, 1)$ a r. Iteramos hasta obtener lo deseado. Sabemos que este paso llega a un final porque los valores absolutos son enteros y decrecen. Una vez que el elemento en la posición $(1, 1)$ divide a toda su fila, podemos hacer todos los elementos $a_{1,i}$ 0 por medio de $C_i \leftarrow C_i - (a_{1,i}/a_{1,1})C_1$. De manera análoga lo hacemos para la primera columna.
3. Nos aseguramos que $a_{1,1}$ divida a todos los elementos restantes de la submatriz. Si $a_{1,1}$ no divide a algún $a_{i,j}$ con $i, j > 1$, se suma la fila i a la fila 1 ($R_1 \leftarrow R_1 + R_i$) o la columna j a la columna 1 ($C_1 \leftarrow C_1 + C_j$). Esto introduce $a_{i,j}$ en la primera fila/columna, y volvemos al paso 2 (reduciendo el valor absoluto del elemento en la posición $(1, 1)$ si $a_{i,j}$ no es múltiplo de $a_{1,1}$). Este proceso termina.
4. Una vez que $a_{1,1} = d_1$ divide a todos los elementos de la matriz, hacemos ceros los elementos de la primera fila y primera columna usando operaciones elementales.
5. Ahora tenemos una matriz en la forma $d_1 \oplus A' = \begin{pmatrix} d_1 & 0 \\ 0 & A' \end{pmatrix}$ donde A' es una matriz de tamaño $(m-1) \times (n-1)$ con entradas enteras.
6. Aplicamos de manera recursiva el algoritmo a la matriz A', el hecho de que $d_1 | d_2 \ldots | d_r$ es una propiedad del algoritmo por como se obtiene cada d_i respecto a los demás elementos de la matriz.

Vamos a mostrar como funciona por medio de un ejemplo.

Ejemplo II.1. Consideremos la matriz A:

$$A = \begin{pmatrix} 6 & 2 & 3 \\ 2 & 4 & 0 \\ 3 & 0 & 1 \end{pmatrix}$$

Y vamos a realizar el algoritmo, de este modo:

1. Encontramos el pivote de menor valor absoluto y moverlo a la posición $(1,1)$. Identificamos que el elemento de la matriz con menor valor absoluto es 1 que se encuentra en la posición $(3,3)$, por lo que lo vamos a mover a la $(1,1)$.
   a) Intercambiamos la fila 1 con la fila 3, así
   $$A \sim \begin{pmatrix} 3 & 0 & 1 \\ 2 & 4 & 0 \\ 6 & 2 & 3 \end{pmatrix}$$
   b) Intercambiamos la columna 1 con la columna 3:
   $$A \sim \begin{pmatrix} 1 & 0 & 3 \\ 0 & 4 & 2 \\ 3 & 2 & 6 \end{pmatrix}$$
   Así, tenemos nuestro pivote en la posición deseada.
2. Hacer ceros la primera fila y columna.
   a) Como $A_{1,3} = 3$ y trivialmente $1|3$, restamos 3 veces la columna 1 de la columna 3 y tenemos:
   $$A \sim \begin{pmatrix} 1 & 0 & 3-3(1) \\ 0 & 4 & 2-3(0) \\ 3 & 2 & 6-3(3) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 2 \\ 3 & 2 & -3 \end{pmatrix}$$
   b) Como $A_{3,1} = 3$ y trivialmente $1|3$, restamos 3 veces la fila 1 de la fila 3, tenemos:
   $$A \sim \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 2 \\ 3-3(1) & 2-3(0) & -3-3(0) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 2 \\ 0 & 2 & -3 \end{pmatrix}$$
   Ahora pasamos al siguiente paso del algoritmo.
3. Pasar a la submatriz A'.
   Ahora nos centramos en la matriz A' donde
   $$A' = \begin{pmatrix} 4 & 2 \\ 2 & -3 \end{pmatrix}$$
   En este caso, el elemento de menor valor absoluto es 2, podemos elegirlo en dos posiciones, en este caso elegimos $A'_{1,2} = A_{2,3}$ y lo movemos a la posición $A'_{1,1} = A_{2,2}$.
   a) Intercambiamos la columna 2 y la columna 3, de ese modo
   $$A' \sim \begin{pmatrix} 2 & 4 \\ -3 & 2 \end{pmatrix}$$
   b) Como $A'_{1,2}$ es 4 y $2|4$, restamos 2 veces la columna 1 de la columna 2 y tenemos
   $$A' \sim \begin{pmatrix} 2 & 0 \\ -3 & 8 \end{pmatrix}$$
   c) Ahora, $A'_{2,1} = -3$, sin embargo $2 \nmid 3$, por lo tanto debemos hacer uso de la lógica del mcd. Sabemos que $\mathrm{mcd}(2,-3) = 1$, por lo que transformamos $A'_{1,1}$ en 1. Para ello primero sumamos la fila 1 a la fila 2 y tenemos
   $$A' \sim \begin{pmatrix} 2 & 0 \\ -1 & 8 \end{pmatrix}$$

Luego intercambiamos fila 2 y fila 2, tenemos

$$A' \sim \begin{pmatrix} -1 & 8 \\ 2 & 0 \end{pmatrix}$$

Multiplicamos la fila 1 por -1 y tenemos

$$A' \sim \begin{pmatrix} 1 & -8 \\ 2 & 0 \end{pmatrix}$$

d) Ahora hacemos 0 la posición $A'_{1,2}$, para ello sumamos 8 veces la columna 1 a la columna 2, tenemos

$$A' \sim \begin{pmatrix} 1 & 0 \\ 2 & 16 \end{pmatrix}$$

Ahora hacemos 0 la posición $A'_{2,1}$, para ello restamos a la fila 2, 2 veces la fila 1 y tenemos

$$A' \sim \begin{pmatrix} 1 & 0 \\ 0 & 16 \end{pmatrix}$$

4. Ahora nuestra matriz A'' sería una matriz 1×1 cuya única entrada es 16, por lo que detenemos el proceso. Finalmente, concluimos que la forma normal de nuestra matriz A es la matriz

$$D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 16 \end{pmatrix}$$

II-C. Uso de la forma normal de Smith para grupos de homología

Vamos a dar un teorema que nos ayudará a calcular los grupos de homología usando la forma normal de Smith.

Teorema II.1. Sean $A \in M^{n\times m}(\mathbb{Z}), B \in M^{l\times m}(\mathbb{Z})$ tales que $BA = 0$. Entonces

$$\frac{\ker(B)}{\mathrm{Im}(A)} = \bigoplus_{i=1}^{r} \mathbb{Z}/\alpha_i\mathbb{Z} \oplus \mathbb{Z}^{m-r-s}$$

Donde $r = \mathrm{rank}(A), s = \mathrm{rank}(B)$ y los α_i son los elementos no nulos de la diagonal de la forma normal de Smith de A.

Demostración. Inicialmente, notemos que $BA = 0$ implica $\mathrm{Im}(A) \subseteq \ker(B)$.
Ahora, como A es una matriz de entradas enteras, sabemos que tiene forma normal de Smith, es decir, existen matrices P_A y Q_A de tamaños $m \times m$ y $n \times n$ respectivamente tales que

$$P_A A Q_A = D_A$$

Donde D_A es la forma normal de Smith de A

$$D_A = \begin{pmatrix} \alpha_1 & & & & \\ & \ddots & & & \\ & & \alpha_r & & \\ & & & 0 & \\ & & & & \ddots \end{pmatrix}$$

Donde $\alpha_1 | \alpha_2 | \ldots | \alpha_r$ y $\mathrm{rank}(A) = r$.
La imagen de A es isomorfa a la imagen de D_A con respecto al cambio de base dado por P_A y Q_A.
La imagen de $D_A : \mathbb{Z}^n \to \mathbb{Z}^m$ es el subgrupo generado por

$$\mathcal{B} = \left\{ \begin{pmatrix} \alpha_1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \alpha_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ \vdots \\ \alpha_r \\ \vdots \\ 0 \end{pmatrix} \right\}$$

De manera similar, para B existen matrices P_B, Q_B de tamaños $l \times l$ y $m \times m$ respectivamente tales que $P_B B Q_B = D_B$ donde D_B es la forma normal de Smith de B

$$D_B = \begin{pmatrix} \beta_1 & & & & \\ & \ddots & & & \\ & & \beta_s & & \\ & & & 0 & \\ & & & & \ddots \end{pmatrix}$$

El kernel de B es el conjunto de $x \in \mathbb{Z}^m$ tales que $Bx = 0$.
Esto es equivalente a $P_B B x = 0$, o $D_B Q_B^{-1} x = 0$. Fijamos $y = Q_B^{-1} x$ y tenemos $Dy = 0$. Una solución para y es de la forma

$$y = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ y_{s+1} \\ \vdots \\ y_m \end{pmatrix}$$

Por lo que $\ker(D_B) \cong \ker(B) \cong \mathbb{Z}^{m-s}$. Ahora debemos elegir bases apropiadas para $\mathbb{Z}^n$ y $\mathbb{Z}^m$ previo a ver el cociente entre ambos espacios.
Sea $e_1, e_2, \ldots, e_n$ la base estándar de $\mathbb{Z}^n$ y $f_1, f_2, \ldots, f_m$ la base estándar de $\mathbb{Z}^m$. Podemos elegir una nueva base $f_1', f_2', \ldots, f_m'$ para $\mathbb{Z}^m$ tal que $\mathrm{Im}(A)$ es generado por $\alpha_1 f_1', \ldots, \alpha_r f_r'$. Esta nueva base viene dada por la forma normal de Smith de A, como $P_A A Q_A = D_A$, los vectores $P_A^{-1}(\alpha_i f_i) = f_i'$ para $i = 1, \ldots, r$ forman una base para $\mathrm{Im}(A)$. De este modo $\mathrm{Im}(A) = \mathrm{gen}_{\mathbb{Z}}\{\alpha_1 f_1', \ldots, \alpha_r f_r'\}$.
Ahora, como $\mathrm{Im}(A) \subseteq \ker(B)$, los vectores $\alpha_i f_i' \in \ker(B)$ para $i = 1, \ldots, r$. Ahora extendemos esta base para una base de $\ker(B)$. El kernel de B es un grupo libre abeliano de rango $m - s$. Por lo tanto, $\ker(B)$ tiene una base, digamos $g_1, \ldots, g_{m-s}$.
El cociente $\ker(B)/\mathrm{Im}(A)$ es un cociente de grupos libres abelianos. Por teoría de módulos, sabemos que si F es un módulo libre de rango finito sobre $\mathbb{Z}$ y S es un submódulo, entonces $F/S \cong \mathbb{Z}^k \oplus \bigoplus \mathbb{Z}/d_i\mathbb{Z}$. Necesitamos relacionar los d_i con los α_i. Nuevamente, por un resultado de módulos libres (a veces se puede encontrar como teorema de estructura para módulos finitamente generados sobre un dominio de ideales principales) establece que si F es un $\mathbb{Z}$-módulo libre de rango N y S es un submódulo, existe una base $x_1, \ldots, x_N$ para F y enteros no negativos $d_1, \ldots, d_k$ donde $k = \mathrm{rank}(S)$ y $d_1 | d_2, \ldots, | d_k$ tales que $d_1 x_1, \ldots, d_k x_k$ es una base para S. De ese modo $F/S \cong \mathbb{Z}/d_1\mathbb{Z} \oplus \mathbb{Z}/d_2\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_k\mathbb{Z} \oplus \mathbb{Z}^{N-k}$.
En nuestro caso $F = \ker(B)$, luego $N = m - s$ y $S = \mathrm{Im}(A)$ y $k = r$, por lo tanto

$$\ker(B)/\mathrm{Im}(A) \cong \mathbb{Z}/d_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_r\mathbb{Z} \oplus \mathbb{Z}^{m-s-r}$$

Resta mostrar que los d_i coinciden con los α_i.
Para ello consideramos otro teorema que establece que dado un submódulo $S \subseteq F$ de un módulo libre finitamente generado sobre un dominio de ideales principales y $x_1, \ldots, x_r$ es una base de S donde $\mathrm{rank}(S) = r$ y $y_1, \ldots, y_N$ es una base de F con $N = \mathrm{rank}(F)$ y M es la matriz de los x_i en términos de los y_j, entonces

$$F/S \cong \bigoplus \mathbb{Z}/d_i\mathbb{Z} \oplus \mathbb{Z}^{N-r'}$$

Donde los d_i son los factores no unidades invariantes de M y r' son los d_i no nulos.
En nuestro caso, $\mathrm{Im}(A)$ tiene rango r. Por lo tanto, hay r de los d_i que nos interesan. Estos d_i son precisamente los α_i de la forma normal de Smith de A, por la unicidad de la misma concluimos lo que se buscaba demostrar. □

El teorema anterior nos dice entonces como calcular los grupos de homología usando la forma normal de Smith. Veamos un ejemplo sencillo de este cálculo.

Ejemplo II.2 (Plano proyectivo real)**.** Podemos pensar en el plano proyectivo real $\mathbb{PR}^2$ como el espacio de todas las rectas que pasan por el origen de $\mathbb{R}^3$ y un punto al infinito. Sin embargo, hay una forma de dar su noción usando un círculo y un disco como sigue:

1. Empezamos con un círculo S^1. En el contexto de rectas a través del origen, podemos pensar en el círculo como el conjunto de rectas en un plano específico.
2. Tomamos un disco D^2 y se pega mediante un homeomorfismo donde el borde del disco coincide con el círculo.
3. El borde del disco se pega al círculo de tal forma que este se enrolla dos veces al rededor del círculo inicial.

Así, vamos a construir un complejo de cadenas para poder hacer uso de la teoría desarrollada previamente.

- Cuenta con un vértice: v con el que vamos a identificar todos los puntos del círculo de partida S^1.
- Cuenta con una arista: a que representa el ecuador de nuestra esfera, este es el ciclo S^1.
- Cuenta con una cara: f que representa el disco que se pega.

Ahora calculamos las fronteras:

- $\partial(v) = 0$,
- $\partial(a) = v - v = 0$ pues a inicia y término en el mismo punto v.
- $\partial(f) = 2a$ pues el disco se pega a a de tal manera que su borde se "enrolla" dos veces alrededor de a. Al recorrer el borde del disco, se recorre dos veces la misma arista.

Ahora para los grupos de cadenas y las matrices de fontera vemos lo siguiente:

- $C_0(\mathbb{PR}^2) \cong \mathbb{Z}$ al ser generado únicamente por v.
- $C_1(\mathbb{PR}^2) \cong \mathbb{Z}$ al ser generado únicamente por a.
- $C_2(\mathbb{PR}^2) \cong \mathbb{Z}$ al ser generado únicamente por f.
- Para otros valores de k se tiene $C_k(\mathbb{PR}^2) \cong 0$.

Ahora definimos los operadores de frontera ∂_k junto a sus matrices:

- $\partial_0 : C_0 \to 0$. Resulta en una matriz trivial (anula a todos),
- $\partial_1 : C_1 \to C_0$. Satisface $\partial_1(a) = 0 \cdot v$, luego la matriz M_1 es

$$M_1 = \begin{pmatrix} 0 \end{pmatrix}$$

- $\partial_2 : C_2 \to C_1$. Satisface $\partial_2(f) = 2a$, luego la matriz M_2 es

$$M_2 = \begin{pmatrix} 2 \end{pmatrix}$$

Y así, usaremos esto para cálcular los primeros dos grupos de Homología teniendo: $H_k = \ker(\partial_k)/\operatorname{Im}(\partial_{k+1})$.

1. $H_0(\mathbb{PR}^2; \mathbb{Z})$.
   En este caso tenemos que $\ker(\partial_0) = C_0 \cong \mathbb{Z}$. Además $\operatorname{Im}(\partial_1) = \{0\}$. Así, el primer grupo de homología es
   $$H_0(\mathbb{PR}^2; \mathbb{Z}) = \ker(\partial_0)/\operatorname{Im}(\partial_1) \cong \mathbb{Z}/\{0\} \cong \mathbb{Z}$$
   Esto coincide con lo esperado, además nos dice que $\mathbb{PR}^2$ tiene una única componente conexa.
2. $H_1(\mathbb{PR}^2; \mathbb{Z})$.
   En este caso tenemos $\ker(\partial_1) = C_1 \cong \mathbb{Z}$.
   Además, $\operatorname{Im}(\partial_2) \cong 2\mathbb{Z}$ Así, el segundo grupo de homología es
   $$H_1(\mathbb{PR}^2; \mathbb{Z}) = \ker(\partial_1)/\operatorname{Im}(\partial_2) \cong \mathbb{Z}/2\mathbb{Z}$$
   Acá la forma normal de Smith se hace más evidente. La matriz $M_2 = \begin{pmatrix} 2 \end{pmatrix}$ ya está en su forma normal de Smith, con un único elemento diagonal no nulo $d_1 = 2$ que representa el factor de torsión.

III. Apartado Computacional

III-A. Verificación computacional, análisis de resultados y limitaciones

En esta sección documentamos el proceso de verificación empírica de la implementación computacional del algoritmo de Smith-MacMillan, así como las decisiones técnicas que guiaron su desarrollo y los ajustes realizados para asegurar su correcta funcionalidad. Además, analizamos en profundidad las diferencias estructurales entre nuestra implementación y otras herramientas computacionales como *SageMath*, con énfasis en la interpretación de los invariantes elementales y la noción de trivialidad en el contexto homológico.

III-B. Naturaleza y propósito de los tests

Los tests diseñados tienen como propósito fundamental validar el comportamiento del algoritmo frente a una diversidad de matrices con estructuras particulares. Cada prueba consiste en aplicar el algoritmo de Smith-MacMillan a una matriz A de coeficientes enteros, extraer su forma normal reducida, y comparar los invariantes elementales obtenidos con aquellos producidos por una función de referencia como `smith_normal_form()` de *SymPy* o el resultado directo de una reducción teórica esperada.

En términos precisos, se espera obtener una matriz diagonal D tal que exista un par de matrices unimodulares S y T que cumplan

$$S \cdot A \cdot T = D,$$

donde los elementos de la diagonal de D satisfacen la condición de divisibilidad

$$d_1 \mid d_2 \mid \cdots \mid d_r,$$

y representan los invariantes elementales de la matriz original. Estos invariantes corresponden, desde el punto de vista homológico, a la estructura del módulo cociente que resulta de la combinación de los espacios de ciclos y bordes en un complejo simplicial.

Las pruebas computacionales, por tanto, no tienen como objetivo evaluar únicamente la corrección sintáctica del código, sino verificar que la reducción conserva la información estructural del objeto algebraico representado. La forma normal de Smith permite identificar de manera directa la parte libre (correspondiente a los unos en la diagonal) y la parte de torsión (correspondiente a los enteros mayores que uno), razón por la cual su correcta determinación es esencial.

III-B1. Expectativas de salida y criterios de validación

En cada test, esperamos que la implementación retorne una matriz diagonal D que:

- Presente todos los invariantes que definen la estructura del grupo o módulo considerado;
- Satisfaga la condición de divisibilidad entre elementos consecutivos;
- Sea congruente con el resultado esperado según teoría o herramientas externas.

Para validar esto, se realiza una comparación explícita de las listas de invariantes obtenidas por ambos métodos (el propio y el de referencia). La verificación incluye, además, un análisis adicional: comprobar si la matriz resultante D obtenida cumple efectivamente la igualdad con $S \cdot A \cdot T$, es decir, si proviene de una secuencia de transformaciones unimodulares válidas. Esta verificación garantiza no solo que la estructura algebraica del resultado es adecuada, sino que el proceso computacional mismo está correctamente construido.

III-B2. Dificultades técnicas y concesiones realizadas

Durante el desarrollo de la implementación, surgieron diversas dificultades tanto conceptuales como prácticas. Para abordarlas, fue necesario realizar ciertas concesiones que modificaron la forma en que se abordó el algoritmo. Estas decisiones fueron tomadas con base en criterios de estabilidad computacional, claridad teórica y compatibilidad con las herramientas de verificación.

III-B2a. Reescritura iterativa del algoritmo.

Aunque el algoritmo de Smith-MacMillan puede ser enunciado de manera natural en términos recursivos, implementarlo así produjo varios problemas técnicos: ciclos sin fin, dificultad para acceder y actualizar bloques internos de la matriz y errores de segmentación al trabajar con submatrices. Por esta razón, se optó por reescribir el procedimiento como un algoritmo estrictamente iterativo, manteniendo explícitamente la matriz completa en cada paso. Esta reescritura conserva la lógica del procedimiento original, pero mejora la trazabilidad de los cambios, permite aplicar reglas de parada más claras y facilita el manejo de pivotes en matrices densas.

III-B2b. Conservación de invariantes triviales.

Uno de los hallazgos más importantes durante la validación fue que existen diferencias de interpretación entre lo que una implementación computacional considera ïnvariante significativoz lo que la teoría identifica como parte de la forma normal. En particular, algunas funciones de referencia como *SymPy* tienden a omitir automáticamente factores 1 en la forma normal de Smith. Esta omisión se justifica algebraicamente porque, desde el punto de vista de módulos, los factores 1 no generan torsión y, por tanto, no alteran la estructura esencial del grupo.

Sin embargo, en nuestra implementación optamos por conservar estos factores de manera explícita. Esto permite observar con claridad cuántas veces aparece el factor 1, y facilita la reconstrucción del rango libre del grupo sin ambigüedad. El hecho de que aparezcan cuatro unos en la diagonal, por ejemplo, indica de manera directa que

el grupo contiene un subgrupo isomorfo a $\mathbb{Z}^4$. Omitir estos elementos, aunque válido desde el punto de vista del isomorfismo, oculta información relevante para el análisis pedagógico y estructural.

III-B2c. Normalización posterior.

Para compatibilizar nuestra salida con la de herramientas como *SymPy*, fue necesario introducir una etapa adicional de normalización. En esta etapa, se realiza un post-procesamiento de los invariantes obtenidos que elimina los unos cuando se detecta que la matriz es completamente nula, o cuando todos los factores son unos y su aparición no aporta información estructural. Esta decisión fue tomada con el fin de evitar falsos negativos en la comparación automática de resultados, pero se aplica de manera cuidadosa para no interferir con los casos donde los unos son parte esencial de la estructura.

III-B3. Implementaciones

La primera implementación utiliza `sympy` y operaciones exactas en matrices simbólicas. Esta versión produce también las matrices de transformación S y T, por lo que permite verificar directamente si $S \cdot A \cdot T = D$. A continuación se muestra el código correspondiente:

Listing 1: Implementación con SymPy: forma_normal_smith

```
from sympy import Matrix, eye, ZZ

def swap_filas(matriz, i, j):
    matriz[i, :], matriz[j, :] = matriz[j, :], matriz[i, :]

def swap_columnas(matriz, i, j):
    matriz[:, i], matriz[:, j] = matriz[:, j], matriz[:, i]

def multiplicar_izquierda_2(matriz, fila0_idx, fila1_idx, a, b, c, d):
    temp0 = []
    temp1 = []
    for j in range(matriz.cols):
        val0 = matriz[fila0_idx, j]
        val1 = matriz[fila1_idx, j]
        temp0.append(a * val0 + b * val1)
        temp1.append(c * val0 + d * val1)
    for j in range(matriz.cols):
        matriz[fila0_idx, j] = temp0[j]
        matriz[fila1_idx, j] = temp1[j]

def multiplicar_derecha_2(matriz, col0_idx, col1_idx, a, b, c, d):
    temp0 = []
    temp1 = []
    for i in range(matriz.rows):
        val0 = matriz[i, col0_idx]
        val1 = matriz[i, col1_idx]
        temp0.append(a * val0 + b * val1)
        temp1.append(c * val0 + d * val1)
    for i in range(matriz.rows):
        matriz[i, col0_idx] = temp0[i]
        matriz[i, col1_idx] = temp1[i]

def division_segura(a, b, dominio):
    return dominio.quo(a, b) if b != 0 else 0

def forma_normal_smith(matriz_entrada, dominio=ZZ, verbose=False):
    if verbose:
        print("Iniciando c lculo de Forma Normal de Smith...")
    matriz = Matrix(matriz_entrada).copy()
    s_transformacion = eye(matriz.rows)
    t_transformacion = eye(matriz.cols)
    ultima_col_procesada = -1

    for i in range(matriz.rows):
        if verbose:
            print(f"\nProcesando fila {i}...")
        j_encontrada = -1
        for j in range(ultima_col_procesada + 1, matriz.cols):
            if not matriz.col(j).is_zero:
                j_encontrada = j
                break
```

```
        else:
            if verbose:
                print("  No se encontr  columna no nula. Terminando.")
            break

        if verbose:
            print(f"  Columna pivote encontrada: {j_encontrada}")

        if matriz[i, j_encontrada] == 0:
            fila_no_cero = -1
            for ii in range(matriz.rows):
                if matriz[ii, j_encontrada] != 0:
                    fila_no_cero = ii
                    break
            if verbose:
                print(f"  Intercambiando fila {i} con fila {fila_no_cero} para obtener pivote no
                    nulo")
            swap_filas(matriz, i, fila_no_cero)
            swap_filas(s_transformacion, i, fila_no_cero)

        swap_columnas(matriz, j_encontrada, i)
        swap_columnas(t_transformacion, j_encontrada, i)
        j_actual = i
        if verbose:
            print(f"  Pivote actual en ({i}, {j_actual}) = {matriz[i, j_actual]}")

        filas_reducidas = set()
        columnas_reducidas = set()

        hubo_actualizacion = True
        while hubo_actualizacion:
            if verbose:
                print("  Iteraci n de reducci n en curso...")
            hubo_actualizacion = False

            for fila_abajo in range(i + 1, matriz.rows):
                if fila_abajo in filas_reducidas:
                    continue
                valor_antes = matriz[fila_abajo, j_actual]
                if valor_antes == 0:
                    filas_reducidas.add(fila_abajo)
                    continue
                if verbose:
                    print(f"    Reduciendo fila {fila_abajo} usando fila {i}")
                hubo_actualizacion = True

                if matriz[i, j_actual] != 0 and dominio.rem(valor_antes, matriz[i, j_actual]) !=
                    0:
                    if verbose:
                        print("      Divisi n no exacta: aplicando gcdex por filas")
                    coef1, coef2, g = dominio.gcdex(matriz[i, j_actual], valor_antes)
                    coef3 = division_segura(valor_antes, g, dominio)
                    coef4 = division_segura(matriz[i, j_actual], g, dominio)
                    multiplicar_izquierda_2(matriz, i, fila_abajo, coef1, coef2, -coef3, coef4)
                    multiplicar_izquierda_2(s_transformacion, i, fila_abajo, coef1, coef2,
                        -coef3, coef4)

                if matriz[i, j_actual] != 0:
                    coef_div = division_segura(matriz[fila_abajo, j_actual], matriz[i,
                        j_actual], dominio)
                    multiplicar_izquierda_2(matriz, i, fila_abajo, 1, 0, -coef_div, 1)
                    multiplicar_izquierda_2(s_transformacion, i, fila_abajo, 1, 0, -coef_div, 1)

                if matriz[fila_abajo, j_actual] == 0:
                    filas_reducidas.add(fila_abajo)

            for col_derecha in range(j_actual + 1, matriz.cols):
                if col_derecha in columnas_reducidas:
                    continue
                valor_antes = matriz[i, col_derecha]
                if valor_antes == 0 or matriz[i, j_actual] == 0:
```

```
                    columnas_reducidas.add(col_derecha)
                    continue
                if verbose:
                    print(f"    Reduciendo columna {col_derecha} usando columna {j_actual}")
                hubo_actualizacion = True

                if dominio.rem(valor_antes, matriz[i, j_actual]) != 0:
                    if verbose:
                        print("      Divisi n no exacta: aplicando gcdex por columnas")
                    coef1, coef2, g = dominio.gcdex(matriz[i, j_actual], valor_antes)
                    coef3 = division_segura(valor_antes, g, dominio)
                    coef4 = division_segura(matriz[i, j_actual], g, dominio)
                    multiplicar_derecha_2(matriz, j_actual, col_derecha, coef1, coef2, -coef3,
                        coef4)
                    multiplicar_derecha_2(t_transformacion, j_actual, col_derecha, coef1, coef2,
                        -coef3, coef4)

                coef_div = division_segura(matriz[i, col_derecha], matriz[i, j_actual], dominio)
                multiplicar_derecha_2(matriz, j_actual, col_derecha, 1, -coef_div, 0, 1)
                multiplicar_derecha_2(t_transformacion, j_actual, col_derecha, 1, -coef_div, 0,
                    1)

                if matriz[i, col_derecha] == 0:
                    columnas_reducidas.add(col_derecha)

        ultima_col_procesada = j_actual

    if verbose:
        print("C lculo completado.")
    return (s_transformacion, matriz, t_transformacion)

def normalizar_forma_smith(D):
    from math import gcd
    from functools import reduce

    # Extraer diagonal significativa
    diag = [abs(D[i, i]) for i in range(min(D.rows, D.cols)) if D[i, i] != 0]
    diag.sort()

    # Ajustar para que d_i | d_{i+1}
    for i in range(len(diag) - 1):
        g = gcd(diag[i], diag[i+1])
        if g != diag[i]:
            diag[i+1] = (diag[i+1] * diag[i]) // g
            diag[i] = g

    # Reconstruir matriz diagonal
    D_final = Matrix.zeros(D.rows, D.cols)
    for i, val in enumerate(diag):
        D_final[i, i] = val

    return D_final
```

La segunda implementación sigue una lógica iterativa usando `numpy`, realizando transformaciones manuales de filas y columnas. Aunque no calcula las matrices S y T, permite reducir efectivamente la matriz a su forma normal. Este método fue necesario debido a problemas de estabilidad en versiones recursivas. El código es el siguiente:

Listing 2: Versión iterativa con NumPy: smith_macmillan

```
# smith_macmillan.py
# Implementaci n del algoritmo iterativo para calcular la forma normal de Smith
# Basado en transformaciones de filas y columnas con operaciones enteras
# Esta versi n busca coincidir con el comportamiento de forma_normal_smith de SymPy

import numpy as np
from math import gcd

def smith_macmillan(A_input):
    """
    Calcula la forma normal de Smith (Smith-MacMillan) para una matriz entera A.

```

```
    Par metro:
        A_input (array-like): matriz de enteros (lista de listas, ndarray o similar)

    Retorna:
        ndarray: matriz diagonal D tal que D es equivalente a A bajo transformaciones
        elementales de filas y columnas, y satisface d_i | d_{i+1} en la diagonal.
    """
    A = np.array(A_input, dtype=int)
    A = A.copy()
    m, n = A.shape
    k = 0  #  ndice  del pivote

    # Recorremos cada pivote desde (0,0) hasta min(m,n)
    while k < min(m, n):
        # Buscamos el menor valor absoluto distinto de cero en la submatriz A[k:, k:]
        sub = A[k:, k:]
        abs_sub = np.abs(sub)
        if not np.any(abs_sub):
            break  # Si el resto es nulo, terminamos

        # Encontramos el menor valor no nulo y lo llevamos a la posici n (k,k)
        min_val = np.min(abs_sub[abs_sub > 0])
        i_min, j_min = np.argwhere(abs_sub == min_val)[0]
        i_min += k
        j_min += k

        A[[k, i_min]] = A[[i_min, k]]         # intercambiamos filas
        A[:, [k, j_min]] = A[:, [j_min, k]]   # intercambiamos columnas

        # Intentamos que A[k,k] divida a los dem s elementos de su fila y columna
        done = False
        max_iter = 50
        iter_count = 0
        while not done and iter_count < max_iter:
            done = True
            for i in range(k + 1, m):
                if A[i, k] != 0:
                    g = gcd(abs(A[k, k]), abs(A[i, k]))
                    if g < abs(A[k, k]):
                        A[k] += A[i]
                        done = False
            for j in range(k + 1, n):
                if A[k, j] != 0:
                    g = gcd(abs(A[k, k]), abs(A[k, j]))
                    if g < abs(A[k, k]):
                        A[:, k] += A[:, j]
                        done = False
            iter_count += 1

        # Anulamos el resto de la columna y fila
        for i in range(m):
            if i != k and A[i, k] != 0:
                q = A[i, k] // A[k, k]
                A[i] -= q * A[k]
        for j in range(n):
            if j != k and A[k, j] != 0:
                q = A[k, j] // A[k, k]
                A[:, j] -= q * A[:, k]

        # Nos aseguramos de que el pivote sea positivo
        if A[k, k] < 0:
            A[k] = -A[k]

        k += 1  # pasamos al siguiente pivote

    # Extraemos la diagonal no nula y la ordenamos
    diag = [abs(A[i, i]) for i in range(min(m, n)) if A[i, i] != 0]
    diag.sort()

    # Forzar divisibilidad: d_i | d_{i+1}
    for i in range(len(diag) - 1):
```

```
            g = gcd(diag[i], diag[i + 1])
            diag[i + 1] = (diag[i + 1] * diag[i]) // g
            diag[i] = g

    # Post-procesamiento adicional para igualar a forma_normal_smith de SymPy
    # Si todos los elementos son 1 y no hay ninguna torsi n significativa, descartamos
    if np.count_nonzero(A) == 0:
        diag = []


    # Construimos la matriz diagonal final
    D = np.zeros_like(A)
    for i in range(len(diag)):
        D[i, i] = diag[i]

    return D
```

III-B4. Por qué un método toma ciertos invariantes como triviales y el otro no

La clave de esta diferencia está en la interpretación de los invariantes en el contexto de la teoría de módulos. Cuando reducimos una matriz a su forma normal de Smith, los valores en la diagonal representan los factores directos del grupo abeliano que se obtiene como cociente. Si en esa diagonal aparecen valores iguales a 1, esto indica que el módulo contiene una parte libre (una copia de $\mathbb{Z}$), y que ese componente no tiene torsión.

Desde el punto de vista algebraico, la torsión trivial (es decir, los factores 1) puede ser omitida sin alterar la estructura del grupo, ya que $\mathbb{Z}/1\mathbb{Z} \cong 0$, y por lo tanto no contribuye a la descomposición primaria del grupo. Sin embargo, desde el punto de vista computacional o pedagógico, conservar estos unos permite entender cuántas reducciones se realizaron, cómo evolucionó el rango y en qué medida el algoritmo logró separar la parte libre de la torsión.

Por ello, algunas bibliotecas como *SymPy* optan por omitir esos valores para devolver una representación más concisa y algebraicamente esencial, mientras que otras (como nuestra implementación) los preservan para reflejar con mayor fidelidad el proceso completo. Esta diferencia puede generar aparentes discrepancias en los resultados, que en realidad no representan errores, sino convenciones de salida distintas. Ambas formas describen correctamente al mismo grupo, pero desde enfoques diferentes.

III-B5. Discusión y observaciones finales

Los resultados obtenidos en la verificación muestran una alta coherencia entre nuestra implementación iterativa del algoritmo de Smith-MacMillan y los métodos de referencia. En todos los casos donde se esperaba una forma normal con estructura no trivial, los invariantes producidos coinciden. Las diferencias observadas pueden explicarse a partir de la política de normalización adoptada por cada herramienta, y no indican fallos conceptuales en la implementación.

Cabe mencionar, sin embargo, que la comparación basada en la igualdad exacta $S \cdot A \cdot T = D$ solo es válida antes del post-procesamiento. Una vez que se aplica la etapa de normalización, esa igualdad puede dejar de cumplirse, ya que la nueva matriz D no es necesariamente resultado de aplicar transformaciones elementales adicionales, sino una versión canónica construida desde los invariantes. Por este motivo, se recomienda separar conceptualmente la verificación algebraica de la reconstrucción estética.

III-B6. Limitaciones y posibilidades de extensión

Si bien los resultados son satisfactorios, existen algunas limitaciones importantes:

- La implementación está diseñada para matrices con coeficientes enteros. Extensiones a otros anillos (por ejemplo, polinomios sobre $\mathbb{Z}$ o cuerpos finitos) requerirían una reimplementación sustancial del algoritmo.
- No se conservan las matrices de transformación S y T en la versión iterativa, lo que impide reconstruir la transformación completa o verificar explícitamente la equivalencia por conjugación.
- Los tests actuales se concentran en la extracción de invariantes, pero no contemplan una validación cruzada con la estructura homológica completa (por ejemplo, comparando directamente los grupos de homología).

A pesar de estas restricciones, consideramos que la implementación es adecuada para propósitos educativos y exploratorios, y permite aplicar de manera efectiva el algoritmo de Smith-MacMillan en el contexto del cálculo de homología simplicial.

III-C. Resultados

Para hacer las pruebas con los códigos que fueron descritos anteriormente, usamos las siguientes matrices

Listing 3: Matrices de prueba

```
m_1 = Matrix([
        [6, 2, 3],
        [2, 4, 0],
        [3, 0, 1]
    ])

    m_2 = Matrix([
        [0, -1, -1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, -1],
        [-1, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0],
        [0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, -1, 0, 0, 0],
        [1, 1, 0, 1, 0, -1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, -1, 0, -1, 1, 1, 0, 0, 1, 0],
        [0, 0, 0, 0, 0, 1, 1, 0, -1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 1, 1, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0]
    ])

    m_3 = Matrix([
        [1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1],
        [-1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0],
        [0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [0, -1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, -1],
        [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 1],
        [0, 0, 0, -1, 0, 0, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0],
        [0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
        [0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 1, 0, -1, 0, 0, 0, 0, 0, 0, -1, 0],
        [0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 1, 0, 0, 0, -1, 0],
        [0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0]
    ])

    m_4 = Matrix([
        [0, -1, -1, 0, -1, 0, -1, 0, -1, 0, 0, 0],
        [-1, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, -1],
        [1, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, -1, 0, 0, 0, 0, -1, 1],
        [0, 0, 0, 0, 0, 1, 1, -1, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0]
    ])

    m_5 = Matrix([
        [1, 0, 0, 0, 0, 0],
        [-1, 1, 0, 0, 0, 0],
        [1, 0, 0, 0, -1, 0],
        [0, 1, 0, 0, 0, 0],
        [0, -1, 1, 0, 0, 0],
        [0, 0, 1, 0, 0, 0],
        [0, 0, -1, 1, 0, 0],
        [0, 0, 0, 1, 0, 0],
        [0, 0, 0, 1, 0, 0],
        [0, 0, 0, -1, 1, 0],
        [0, 0, 0, 0, -1, -1],
        [0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 1]
    ])
```

y los resultados obtenidos fueron los siguientes:

Para el algoritmo de Smith

Listing 4: Resultados Smith

```
Probando matriz:
m_1

Forma Normal de Smith D:
Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 16]])
Verificaci n exitosa: S * A * T = D

Probando matriz:
m_2

Forma Normal de Smith D:
Matrix([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0]])
Verificaci n exitosa: S * A * T = D

Probando matriz:
m_3

Forma Normal de Smith D:
Matrix([[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
Verificaci n exitosa: S * A * T = D

Probando matriz:
m_4

Forma Normal de Smith D:
Matrix([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
Verificaci n exitosa: S * A * T = D

Probando matriz:
m_5

Forma Normal de Smith D:
Matrix([[1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0], [0, 0,
    0, 0, 1, 0], [0, 0, 0, 0, 0, 1], [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0]])
Verificaci n exitosa: S * A * T = D
```

Para el algoritmo de Smith - MacMillan

Listing 5: Resultados SmithMacMillan

```
=== m_1 ===
Resultado Smith-MacMillan:
[[ 1  0  0]
 [ 0  1  0]
 [ 0  0 16]]
Diagonal no nula: [1, 1, 16]
Es una forma normal de Smith v lida.

=== m_2 ===
Resultado Smith-MacMillan:
[[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]]
Diagonal no nula: [1, 1, 1, 1, 1, 1, 1]
Es una forma normal de Smith v lida.

=== m_3 ===
Resultado Smith-MacMillan:
[[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
 ... 7 filas de ceros ...       ]]
Diagonal no nula: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
Es una forma normal de Smith v lida.

=== m_4 ===
Resultado Smith-MacMillan:
[[1 0 0 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]]
Diagonal no nula: [1, 1, 1, 1, 1]
Es una forma normal de Smith v lida.

=== m_5 ===
Resultado Smith-MacMillan:
[[1 0 0 0 0 0]
 [0 1 0 0 0 0]
 [0 0 1 0 0 0]
 [0 0 0 1 0 0]
 [0 0 0 0 1 0]
 [0 0 0 0 0 1]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]]
Diagonal no nula: [1, 1, 1, 1, 1, 1]
Es una forma normal de Smith v lida.
```

Comparando resultados

Listing 6: Comparación de resultados

```
=== m_1 ===
smith_macmillan: [1, 1, 16]
forma_normal_smith: [1, 1, 16]
Coinciden los invariantes elementales.

=== m_2 ===
smith_macmillan: [1, 1, 1, 1, 1, 1, 1]
forma_normal_smith: []
Diferencias encontradas en los invariantes.

=== m_3 ===
smith_macmillan: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
forma_normal_smith: [1, 1, 1, 1]
Diferencias encontradas en los invariantes.

=== m_4 ===
smith_macmillan: [1, 1, 1, 1, 1]
forma_normal_smith: []
Diferencias encontradas en los invariantes.

=== m_5 ===
smith_macmillan: [1, 1, 1, 1, 1, 1]
forma_normal_smith: [1, 1, 1, 1, 1, 1]
Coinciden los invariantes elementales.
```

Repositorio con el código, resultados y recursos utilizados
Proyecto AAC (GitHub) - Aguiar, Duque, Pinzón

Referencias

[AS69] Michael F Atiyah and Graeme B Segal. Equivariant k-theory and completion. *Journal of Differential Geometry*, 3(1-2):1–18, 1969.

[Ati66] Michael Francis Atiyah. K-theory and reality. *The Quarterly Journal of Mathematics*, 17(1):367–386, 1966.

[BC09] Christopher Bradley and Arthur Cracknell. *The mathematical theory of symmetry in solids: representation theory for point groups and space groups*. Oxford University Press, 2009.

[EH10] Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

[FH13] William Fulton and Joe Harris. *Representation theory: a first course*, volume 129. Springer Science & Business Media, 2013.

[Hat01] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2001.

[KMM04] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. *Computational Homology*, volume 157 of *Applied Mathematical Sciences*. Springer, New York, 2004.

[NG82] JD Newmarch and RM Golding. The character table for the corepresentations of magnetic groups. *Journal of Mathematical Physics*, 23(5):695–704, 1982.

[SUX25] Higinio Serrano, Bernardo Uribe, and Miguel A XicotÃŠncatl. Magnetic equivariant k-theory. *arXiv preprint arXiv:2503.06267*, 2025.

[Wig12] Eugene Wigner. *Group theory: and its application to the quantum mechanics of atomic spectra*, volume 5. Elsevier, 2012.