

Manual del Paquete topologyR: Análisis de Estructuras Topológicas

José Mauricio Gómez Julián

2024-12-30

Contents

1. Fundamentos Teóricos	2
1.1. Introducción a los Espacios Topológicos	2
1.1.1. Definición Formal	2
1.1.2. Propiedades Fundamentales	3
1.2. Bases y Subbases	3
1.2.1. Definición y Propiedades de Bases	3
1.2.2. Subbases y su Relación con Bases	4
1.3. Conexidad en Espacios Topológicos	4
1.3.1. Definición Topológica de Conexidad	4
1.3.2. Conexidad en Grafos y su Relación con la Conexidad Topológica	5
1.4. Explicación Intuitiva de los Fundamentos Teóricos	5
2. Implementación del Algoritmo	6
2.1. Algoritmo Base	6
2.2. Consideraciones de Implementación	10
2.3. Optimizaciones y Heurísticas	11
3. Funciones del Paquete	15
3.1. is_topology_connected()	15
3.2. is_topology_connected2()	15
3.3. is_topology_connected_manual()	16
3.4. analyze_topology_factors()	16
3.5. calculate_thresholds()	17
3.6. visualize_topology_thresholds()	18

4. Aplicaciones	20
4.1. Análisis de Series Temporales Económicas	20
4.1.1. Preparación de Datos y Análisis de Distribución	21
4.1.2. Análisis Topológico Básico	26
4.1.3. Análisis Avanzado con Umbrales	27
4.1.4. Pautas de Optimización de Rendimiento	30
4.1.5. Marco de Interpretación	31
4.2. Ejemplos de Análisis Comparativo	32
4.3. Algunas Recomendaciones para Tu Conjunto de Datos	33
4.3.1. Preparación de Datos	33
4.3.2. Manejo de Errores:	34
4.3.3. Optimización para Grandes Conjuntos de Datos	35
5. Limitaciones y Consideraciones	36
5.1. Limitaciones Computacionales	36
5.2. Aproximaciones y Validez	36
6. Referencias	36

1. Fundamentos Teóricos

1.1. Introducción a los Espacios Topológicos

1.1.1. Definición Formal

Para comprender la implementación del algoritmo, primero deben comprenderse conceptos topológicos fundamentales, concretamente los de base, subbase y su relación con la topología de un objeto.

Sea X un conjunto cualquiera. Se llama *topología en X* a todo sistema τ de subconjuntos G de X que verifica dos condiciones (Kolmogórov & Fomin, 1978, pág. 90):

1. El propio conjunto X y el conjunto vacío \emptyset pertenecen a τ .
2. La unión $\bigcup_{\alpha} G_{\alpha}$ de un número cualquiera (finito o infinito) y la intersección $\bigcap_k G_k$ de un número finito de conjuntos τ pertenecen a τ . Otra forma de definirlo es, siguiendo a Kelley (1955, pág. 50) que la intersección de dos miembros cualesquiera de τ es un miembro de τ (τ es cerrada bajo intersecciones) y que la unión de los miembros de cualquier familia de τ es parte de X (el espacio de la topología de τ es cerrado bajo uniones arbitrarias de τ).

Así, el conjunto X se llama *espacio de la topología de τ* , τ se llama *topología de X* y el par (X, τ) se llama **espacio topológico**, es decir, $T = (X, \tau)$. Los conjuntos pertenecientes al sistema de conjuntos τ se llaman *abiertos* (Kolmogórov & Fomin, 1978, pág. 90).

Como señalan Kolmogórov & Fomin (1978, pág. 90), un espacio métrico está constituido por un conjunto de puntos y una métrica introducida en ese conjunto; de la misma forma, un espacio topológico está constituido por un conjunto de puntos y una topología introducida en él. Por consiguiente, definir un espacio topológico

significa definir un conjunto X y una topología τ sobre él, es decir, indicar aquellos subconjuntos que se consideran abiertos en X (Kolmogórov & Fomin, 1978, pág. 90).

Está claro que, en un mismo conjunto X se puede introducir diferentes topologías, convirtiéndolo de esta forma en diferentes espacios topológicos (Kolmogórov & Fomin, pág. 90). Sin embargo, se denotará al espacio topológico, es decir, al par (X, τ) , mediante T y se llamarán “puntos” a los elementos del espacio topológico.

1.1.2. Propiedades Fundamentales

Los conjuntos $T \setminus G$, es decir, los elementos de T que no pertenecen a G , que son complementarios a los abiertos, se llaman conjuntos *cerrados* del espacio topológico T (Kolmogórov & Fomin, 1978, pág. 90).

El principio de dualidad consiste en que de cualquier teorema referente a un sistema de subconjuntos de un conjunto ijo S se puede deducir *de manera automática* otro teorema, el teorema dual, sustituyendo los conjuntos originalmente considerados por sus complementos, la suma de conjuntos por su intersección y la intersección por la suma (Kolmogórov & Fomin, 1978, pág. 16). En virtud de las relaciones de dualidad, de los axiomas 1 (el complemento de la suma es igual a la intersección de los complementos) y 2 (el complemento de la intersección es igual a la suma de los complementos), se deduce que (Kolmogórov & Fomin, 1978, pág. 90):

1. El conjunto vacío \emptyset y todo el espacio T son cerrados.
2. La intersección de un número cualquiera (finito o infinito) y la unión de un número finito de conjuntos cerrados son cerrados.

En todo espacio topológico se introducen, a partir de estas definiciones y de un modo natural, los conceptos de vecindad, puntos de adherencia, adherencia de conjuntos, etc.

1.2. Bases y Subbases

1.2.1. Definición y Propiedades de Bases

Así, como señalan Kolmogórov & Fomin (1978, pág. 91), se llama *vecindad* del punto $x \in T$ a todo conjunto abierto $G \subset T$ que contiene el punto x ; un punto $x \in T$ se llama *punto de adherencia* del conjunto $M \subset T$, cuando toda vecindad de punto x contiene al menos un punto de M ; un punto x se llama *punto de acumulación* del conjunto M , cuando toda vecindad del punto x contiene un número infinito de puntos de M . La totalidad de puntos de adherencia del conjunto M se llama *adherencia* del conjunto M y se denota mediante el símbolo $[M]$ (Kolmogórov & Fomin, 1978, pág. 91).

Como señalan Kolmogórov & Fomin (1978, pág. 93), una colección ζ de subconjuntos abiertos se llama *base* del espacio topológico T , cuando todo subconjunto abierto de T se puede representar como una de cierto número de conjuntos de ζ .

Por ejemplo, la colección de todas las bolas abiertas (de todos los radios y centros posibles) constituye una base en un espacio métrico. En particular, el sistema de todos los intervalos es una base en la recta. De lo expuesto se desprende que la topología τ del espacio T queda definida, si se indica en este espacio un base ζ . (Kolmogórov & Fomin, 1978, pág. 93)

Como señalan Kolmogórov & Fomin (1978, pág. 93), para que esta forma de introducir la topología tenga un valor práctico es necesario señalar aquellas condiciones que debe cumplir un sistema ζ de subconjuntos del conjunto dado T para que la colección de todas las sumas posibles de conjuntos de ζ pueda ser considerada como la colección de conjuntos abiertos en T (es decir, para que estas suman verifiquen los axiomas 1° y 2° de espacio topológico). Tales condiciones vienen dadas por el siguiente teorema:

TEOREMA Supongamos que en un conjunto T se ha escogido un sistema ζ de subconjuntos G_α que verifica las siguientes condiciones:

- a. Todo punto $x \in T$ está contenido al menos en un subconjunto $G_\alpha \in \zeta$
- b. Si $x \in G_\alpha$ y $x \in G_\beta$, existe un $G_\gamma \in \zeta$ tal que $x \in G'_\gamma \in G_\alpha \cup G_\beta$, donde G'_γ es el conjunto complemento de G_γ , es decir, $G'_\gamma = T \setminus G_\gamma$.

Así, si declaramos abiertos en T al conjunto vacío y a todos los conjuntos que se pueden representar como la suma de determinados $G_\alpha \in \zeta$, el conjunto T resultará un espacio topológico (es decir, estas suman verificarán los axiomas 1° y 2°) y el sistema ζ será una base de él (Kolmogórov & Fomin, 1978, pág. 93).

1.2.2. Subbases y su Relación con Bases

Para probar si una colección dada de conjuntos abiertos es una base o no suele ser útil el siguiente criterio (Kolmogórov & Fomin, 1978, pág. 94):

TEOREMA

Para que un sistema G_α de conjuntos abiertos sea una base del espacio topológico T es necesario y suficiente que para todo conjunto abierto G y todo punto $x \in G$ exista un conjunto G_α de este sistema tal que $x \in G_\alpha \in G$.

Finalmente, Kelley (1955, pág. 61) señala que una familia S de conjuntos es una **sub-base de una topología** τ si la familia de todas las intersecciones finitas de miembros de S es base de τ (o lo que es equivalente: todo miembro de τ es unión de intersecciones finitas de miembros de S). Una forma más intuitiva de verlo es que una subbase es una colección S de subconjuntos de un espacio topológico que está contenida en una base de la topología y que se puede completar para formar una base al sumar todas las intersecciones finitas de los subconjuntos del conjunto S .

1.3. Conexidad en Espacios Topológicos

1.3.1. Definición Topológica de Conexidad

Kelley (1955, pág. 67) señala que un espacio topológico $T = (X, \tau)$ es conexo si X no es unión de dos subconjuntos separados no-vacíos. En otras palabras, es un subconjunto de un espacio topológico que no puede ser descrito como unión disjunta de dos conjuntos abiertos no vacíos del espacio topológico en cuestión.

Lo anterior significa que no se pueden cumplir simultáneamente que, por un lado, si separamos el espacio topológico en dos conjuntos abiertos, es decir, que los extremos de cada nuevo conjunto (resultado de la separación) no se incluyan en estos dos conjuntos, al volverlos a unir el resultado sea equivalente al conjunto original y, por otro lado, que esta intersección sea vacía (que no tienen elementos comunes entre sí).

Esto no se cumple simultáneamente porque, si se separan en dos subconjuntos, es posible lograr que al intersecarlos (volvernos a unir) el resultado sea un conjunto vacío, pero solo a costa que se omita algún punto de los originales, por lo cual no se vuelve al conjunto original; por otro lado, es posible volver al conjunto original, pero solo a costa de no omitir ninguno de los puntos originales, por lo cual su intersección no sería un conjunto vacío.

Lo anterior también puede expresarse formalmente diciendo que un conjunto conexo es un subconjunto de $C \subseteq X$ de un espacio topológico (X, τ) que no puede ser descrito como una unión disjunta de dos conjuntos abiertos no vacíos de la topología, en donde T es la colección de conjuntos abiertos del espacio topológico; este subconjunto, como su misma definición lo revela, puede ser equivalente al espacio topológico completo. En otras palabras, está formado por una sola pieza y no es divisible porque todas sus partes están conectadas de alguna manera, por lo que analizar partes aisladamente afectará las propiedades globales del conjunto.

Por el contrario, en un conjunto no-conexo o desconectado, la situación es diferente. Un conjunto no-conexo se puede separar en dos o más subconjuntos abiertos disjuntos, donde cada uno de estos subconjuntos recibe el nombre de componente conexa.

Lo anterior significa que en conjuntos desconectados sí es posible analizar una parte (*i.e.*, una componente conexa) del conjunto de forma independiente y, en muchos casos, esto no afectará las propiedades globales del conjunto. Sin embargo, hay matices importantes.

En primer lugar, las propiedades locales pueden analizarse completamente en una componente conexa sin afectar a las demás. En segundo lugar, algunas propiedades globales se mantendrán al analizar una sola componente, mientras que otras requerirán considerar todas las componentes; por ejemplo, la compacidad de una componente no implica la compacidad del conjunto total.

Algunas de las propiedades topológicas (*i.e.*, esenciales) que pueden ser analizadas localmente (por componentes) son la conexidad de cada componente, la compacidad de cada componente u otras propiedades topológicas locales más generales (por ejemplo, ser localmente euclídeo). En contraposición, las propiedades globales que requerirán considerar todas las componentes, aun cuando el conjunto sea desconectado, son el número total de componentes conexas, la compacidad del conjunto total (que requiere que todas las componentes sean compactas y que haya un número finito de ellas), diferentes propiedades métricas o de medida (para ver si las mismas son generalizables a todo el espacio), entre otras. Así, aunque es posible analizar componentes por separado, es crucial recordar que algunas propiedades globales solo pueden entenderse considerando todas las componentes juntas.

La principal diferencia entre conjuntos conexos y no-conexos en términos de las propiedades locales es que, en un conjunto conexo, las propiedades locales pueden “propagarse” o tener implicaciones globales de maneras que no ocurren en conjuntos no-conexos. Por ejemplo, consideremos una función continua f en un intervalo cerrado $[a, b]$ pre-definido como conexo. Es posible analizar propiedades locales como la derivabilidad en cada punto, sin embargo, el Teorema del Valor Intermedio, que es una propiedad global, se deriva de la continuidad local y la conexidad del intervalo: por consiguiente, lo que no es posible hacer en un conjunto conexo es aislar completamente una parte del conjunto del resto, en términos topológicos, puesto que siempre habrá alguna forma de “conexión” entre las partes.

1.3.2. Conexidad en Grafos y su Relación con la Conexidad Topológica

Una vez revelada la topología (sea completa o aproximada por optimización), su conexidad puede ser determinada construyendo un grafo dirigido o un grafo no-dirigido.

Un grafo es un conjunto de objetos llamados vértices o nodos unidos por enlaces llamados aristas o arcos, que permiten representar relaciones binarias entre elementos de un conjunto (Trudeau, 1993, págs. 19-20).

Se justifica la utilización de un grafo no-dirigido cuando es válido asumir que la relación de conexión entre los elementos de la topología es simétrica. En otras palabras, si un elemento A está conectado con un elemento B, entonces B también está conectado con A.

1.4. Explicación Intuitiva de los Fundamentos Teóricos

Imaginemos que tenemos un conjunto X que representa el plano cartesiano. Dentro de este plano, dibujamos un círculo con su correspondiente circunferencia. La circunferencia es la línea que delimita el círculo, mientras que el círculo es el área encerrada por la circunferencia.

Ahora, vamos a introducir una topología τ en este conjunto X . La topología τ es una colección de subconjuntos de X que cumple ciertas condiciones; el plano cartesiano sería el espacio subyacente respecto del cual se construye la topología, mientras que el plano más la topología sería el espacio topológico. En nuestro ejemplo, podemos considerar que los conjuntos abiertos en esta topología son el círculo (sin incluir la circunferencia) y el área fuera del círculo (excluyendo la circunferencia). Estos conjuntos abiertos cumplen

las condiciones mencionadas: el conjunto X (todo el plano) y el conjunto vacío pertenecen a τ , y las uniones arbitrarias e intersecciones finitas de conjuntos abiertos también son abiertas.

Por otro lado, los conjuntos cerrados en esta topología serían la circunferencia y el conjunto X completo (incluyendo el círculo y su exterior). Estos conjuntos cerrados cumplen las condiciones duales: las intersecciones arbitrarias y uniones finitas de conjuntos cerrados también son cerradas.

Ahora, imaginemos un punto x dentro del círculo. Una vecindad de x sería cualquier conjunto abierto que contenga a x , es decir, cualquier área dentro del círculo que incluya a x . Un punto y en la circunferencia sería un punto de adherencia del círculo, ya que cualquier vecindad de y (cualquier área abierta que contenga a y) siempre incluirá al menos un punto del círculo. La adherencia del círculo sería el círculo junto con su circunferencia, ya que incluye todos los puntos de adherencia.

Una base ζ de la topología τ sería una colección de conjuntos abiertos tales que cualquier conjunto abierto en τ pueda ser expresado como una unión de elementos de ζ . En nuestro ejemplo, una base podría ser la colección de todos los discos abiertos dentro del círculo.

Por último, una sub-base S de la topología τ sería una colección de conjuntos tales que todas las intersecciones finitas de elementos de S formen una base de τ . En nuestro ejemplo, una sub-base podría ser la colección de todas las regiones abiertas que incluyan el centro del círculo y se extiendan hasta la circunferencia, junto con todas las regiones abiertas fuera del círculo que se extiendan hasta la circunferencia.

Cualquier región del plano (*i.e.*, cualquier conjunto de puntos del espacio) para la que si se separan en dos subconjuntos es posible lograr que al intersecarlos (volvernos a unir) el resultado sea un conjunto vacío, pero solo a costa que se omita algún punto de los originales o bien, volver al conjunto original, pero solo a costa de no omitir ninguno de los puntos originales, por lo cual su intersección no sería un conjunto vacío, es un conjunto conexo, una sola pieza, una totalidad indivisible.

2. Implementación del Algoritmo

2.1. Algoritmo Base

ALGORITMO EN ESPAÑOL

Entrada: Un número de vértices de un grafo G , Matriz de adyacencia de $V(G)$.

Salida: Una topología de G (TG).

1. Insertar un número de vértices de G .

2. para $i \in n$

Introducir el nombre de los vértices de un grafo G .

fin

3. para $i \in n$

para $j \in n$

Introducir la matriz de adyacencia de $V(G)$.

fin

fin

```

4. para i \in n
para j \in n
Calcular el grado de V(G).

fin

fin

5. para i \in n
para j \in n
si (grado != 0)
clase(i, j) = x(j).

R = (grado(x(i))x(t), grado(x(j))x(f)).

fin

fin

fin

6. para i \in n
para j \in n
si (clase(i, j) != 0)
subbase = clase(j).

fin

fin

fin

7. para v \in n
para vj \in n
base = subbase + intersección(v, vj).

fin

fin

8. para vi \in n
para vj \in n

```

```
si (unión(v, vj) != (\emptyset))
```

```
vi, vj \in unión.
```

```
unión(vi, vj) \in unión.
```

```
fin
```

```
topología = base + unión(vi, vj).
```

```
fin
```

```
fin
```

ALGORITMO EN INGLÉS

Input: A number of vertices of a graph G, Adjacency matrix of $V(G)$.

Output: A topology of G (TG).

1. Insert the number of vertices of G.

2. for i \in n

Enter the name of the vertices of a graph G.

end

3. for i \in n

for j \in n

Enter the adjacency matrix of $V(G)$.

end

end

4. for i \in n

for j \in n

Calculate the degree of $V(G)$.

end

end

5. for i \in n

for j \in n

if (degree != 0)

class(i, j) = x(j).


```

R = (degree(x(i))x(t), degree(x(j))x(f)).

end

end

end

6.    for i \in n

for j \in n

if (class(i, j) != 0)

subbase = class(j).

end

end

end

7.    for v \in n

for vj \in n

base = subbase + intersection(v, vj).

end

end

8.    for vi \in n

for vj \in n

if (union(v, vj) != (\emptyset))

vi, vj \in union.

union(vi, vj) \in union.

end

topology = base + union(vi, vj).

end

end

```

2.2. Consideraciones de Implementación

Todo código (en R u otro lenguaje) que se construya para operativizar el algoritmo anterior en su forma más simple debe contener los siguientes pasos:

1. Generar los datos numéricos pertinentes con los parámetros dados.
2. Crear un grafo completamente conectado usando los índices de los datos como vértices.
3. Asignar los valores numéricos como atributos a los vértices del grafo.
4. Definir la relación R según la Definición 2.1 de Nada et al. (2018, pág. 2), considerando los valores numéricos.
5. Obtener la subbase tomando las post clases (imágenes de la relación) de cada vértice.
6. Generar la base incluyendo el espacio total X, el conjunto vacío, los vértices y sus vecindarios.
7. Construir la topología tomando uniones de elementos de la base, según las leyes de los espacios topológicos.
8. Imprimir la relación R, la subbase, la base y la topología resultantes.

NOTA: La relación R se define considerando los valores numéricos asociados a los vértices. La subbase, base y topología se obtienen siguiendo las leyes matemáticas de los espacios topológicos.

Evidentemente, el código anterior podría modificarse para que considere vecindarios de los vértices y permita conjuntos abiertos que contengan múltiples vértices, lo cual generará una topología más rica, como la construida por Nada et al. (2018, pág. 9).

Sin embargo, a pesar de que esta consideración de vecindarios y múltiples vértices es correcta, las funciones producen topologías que se vuelven exponencialmente más grandes a medida el conjunto de datos aumenta ligeramente, lo que esta consideración a secas (*i.e.*, sin ninguna optimización de por medio) la hace inviable para conjuntos lo suficientemente grandes. Por ejemplo, para un conjunto de 129 observaciones se vuelve inviable su cálculo, tanto por el costo computacional como por el tiempo requerido al investigador para obtener conclusiones valiosas sobre la topología descubierta[²].

El problema anterior se presenta debido a la manera en que se construyen las bolas abiertas n -dimensionales[³] (*i.e.*, los vecindarios usando la matriz de adyacencia), es decir, al construir los vecindarios de cada vértice con la matriz de adyacencia.

Para optimizar este proceso y hacerlo viable para conjuntos grandes, se hacen las siguientes modificaciones:

1. Definir una métrica y un umbral para la creación de vecindarios. Así, en lugar de conectar cada vértice con todos los demás, podemos definir una distancia máxima para considerar dos puntos como vecinos.
2. Limitar el tamaño de la base. Concretamente, se establecerá un número máximo de elementos en la base, seleccionando los más representativos.
3. Usar una aproximación de la topología. Esto se logrará usar métodos de aproximación que capturen la estructura esencial de la topología sin consierar todas las uniones posibles (*i.e.*, sin generar todos los conjuntos posibles)

Estos criterios son matemáticamente válidos y pueden considerarse como una forma robusta de aproximación de la topología original. Sin embargo, es importante notar que esta aproximación puede no capturar todas las propiedades topológicas del espacio original. Por ello, podría ser recomendable usar la sintaxis del segundo bloque de código (el bloque inmediato anterior) para conjuntos pequeños (de 30 o menos observaciones).

2.3. Optimizaciones y Heurísticas

Puesto que las definiciones de *distancia* y *umbral* son cruciales en la optimización de la búsqueda de la topología, la robustez metodológica y teórica-matemática de dicha optimización radica en la robustez de tales definiciones en los mismos sentidos.

Puesto que se trabaja con una sola variable de números reales, la distancia que debe definirse es una distancia unidimensional.

Aunque a la hora de definir la distancia no exista complicación alguna, es diferente en el caso de definir el radio de las bolas abiertas (*i.e.*, el umbral). Para ello, existen diferentes heurísticas posibles, cada uno con sus ventajas y desventajas:

a. Promedio de la Distancia entre Puntos Adyacentes

```
threshold <- mean(abs(diff(sort(datos))))
```

Esto tiene como ventaja que captura la escala promedio de las variaciones locales en los datos, pero puede ser sensible a valores atípicos y no ser adecuada cuando los datos no tienen una distribución cuyo parámetro de localización sea la media (como sí lo es en el caso de la Normal).

b. Mediana de las Distancias de Puntos Adyacentes

```
threshold <- median(abs(diff(sort(datos))))
```

La ventaja de esta estimación es que es más robusta frente a valores atípicos que el promedio, pero tiene como desventaja de que no podría capturar bien la estructura si hay muchos valores repetidos^[4].

c. Desviación Estándar de los Datos

```
threshold <- sd(datos)
```

Esta heurística tendría la ventaja de poder capturar la dispersión global de los datos, sin embargo, puede ser demasiado grande si hay valores atípicos extremos y, por consiguiente, volverse en costos computacionales y de tiempo humano inviable (que es justamente lo que se quiere evitar).

d. Rango Intercuartílico Dividido por un Factor

```
threshold <- IQR(datos) / factor
```

Esta heurística tiene la ventaja de ser muy robusta frente a valores atípicos, pero podría ser demasiado conservadora si los datos están muy agrupados^[5].

El factor de división, por ejemplo, 4, no es una regla fija. Se usa comúnmente en estadísticas para identificar valores atípicos ($1.5 \times \text{IQR}$ es una regla común), pero para nuestro propósito de definir un umbral, podríamos ajustarlo. La lógica general es que dividir por un número mayor da un umbral más pequeño, lo que resulta en una topología más fina.

En realidad, la elección del factor depende del nivel de detalle que se desee en la topología.

Un factor más pequeño (por ejemplo, dividir por 2 en lugar de 4) resultaría en un umbral más grande, lo que llevaría a una topología más gruesa con menos conjuntos. Un factor más grande (por ejemplo, dividir por 8) resultaría en un umbral más pequeño, lo que llevaría a una topología más fina con más conjuntos.

La elección del factor debe basarse en experimentos con los datos específicos y en el nivel de detalle que se necesite para el análisis. No hay una regla universal para este factor, como no lo hay en el caso de ninguna heurística. Para ello, debe usarse la función “analyze_topology_factors” de esta librería.

```
# SI LA INSTALACIÓN SE DESEA REALIZAR LOCALMENTE, DEBEN SEGUIRSE LOS SIGUIENTES PASOS:  
# Paso 1:  
setwd("C:/Users/ROG/Documents/topologyR")  
# Paso 2:  
library(devtools)
```

```
## Warning: package 'devtools' was built under R version 4.4.3
```

```
## Cargando paquete requerido: usethis
```

```
## Warning: package 'usethis' was built under R version 4.4.3
```

```
# Paso 3:  
library(roxygen2)
```

```
## Warning: package 'roxygen2' was built under R version 4.4.3
```

```
# Paso 4:  
document()
```

```
## i Updating topologyR documentation
```

```
## i Loading topologyR
```

```
# Paso 5:  
install()
```

```
## rlang (1.1.5 -> 1.1.6) [CRAN]
```

```
## cli (3.6.3 -> 3.6.5) [CRAN]
```

```
## Installing 2 packages: rlang, cli
```

```
## Installing packages into 'C:/Users/ROG/AppData/Local/R/win-library/4.4'
```

```
## (as 'lib' is unspecified)
```

```
## package 'rlang' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'rlang'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problema al copiar
```

```
## C:\Users\ROG\AppData\Local\R\win-library\4.4\00LOCK\rlang\libs\x64\rlang.dll a
```

```
## C:\Users\ROG\AppData\Local\R\win-library\4.4\rlang\libs\x64\rlang.dll:
```

```
## Permission denied
```

```
## Warning: restored 'rlang'
```

```
## package 'cli' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'cli'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problema al copiar
```

```
## C:\Users\ROG\AppData\Local\R\win-library\4.4\00LOCK\cli\libs\x64\cli.dll a
```

```
## C:\Users\ROG\AppData\Local\R\win-library\4.4\cli\libs\x64\cli.dll: Permission
```

```
## denied
```

```
## Warning: restored 'cli'
```

```
##
## The downloaded binary packages are in
## C:\Users\ROG\AppData\Local\Temp\Rtmp4iWe3m\downloaded_packages
## -- R CMD build -----
##      v checking for file 'C:\Users\ROG\Documents\topologyR\DESCRIPTION'
##      - preparing 'topologyR':
##        checking DESCRIPTION meta-information ... v checking DESCRIPTION meta-information
##        - checking for LF line-endings in source and make files and shell scripts
##      - checking for empty or unneeded directories
## - building 'topologyR_0.1.0.tar.gz'
##
## Running "C:/PROGRA~1/R/R-44~1.2/bin/x64/Rcmd.exe" INSTALL \
## "C:\Users\ROG\AppData\Local\Temp\Rtmp4iWe3m\topologyR_0.1.0.tar.gz" \
## --install-tests
## * installing to library 'C:/Users/ROG/AppData/Local/R/win-library/4.4'
## * installing *source* package 'topologyR' ...
## ** using staged installation
## ** R
## ** byte-compile and prepare package for lazy loading
## ** help
## *** installing help indices
## ** building package indices
## ** testing if installed package can be loaded from temporary location
## ** testing if installed package can be loaded from final location
## ** testing if installed package keeps a record of temporary installation path
## * DONE (topologyR)
```

```
# UNA VEZ REALIZADOS LOS PASOS ANTERIORES, DE AQUÍ EN ADELANTE PUEDE PROCEDERSE A UTILIZAR LA LIBRERÍA
topology <- list(
  c(1, 2, 3),
  c(3, 4, 5)
)
```

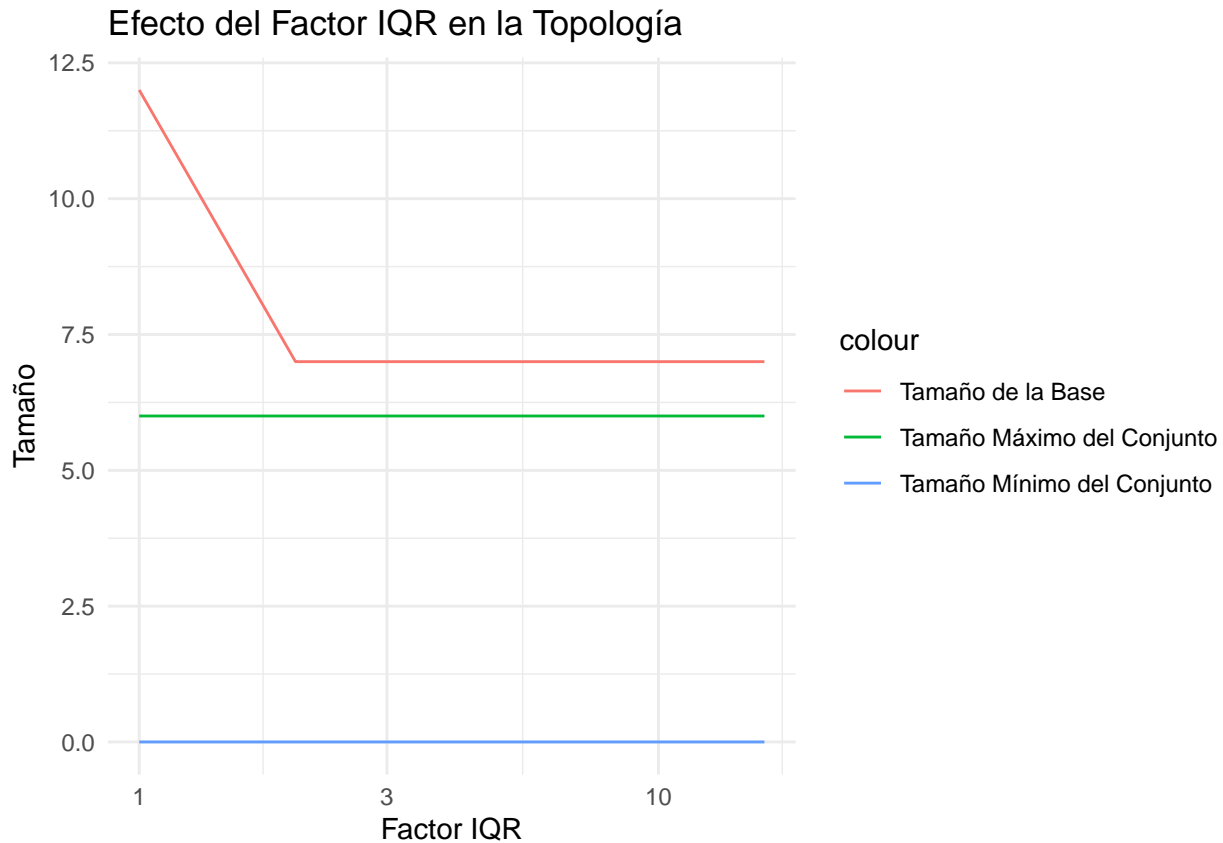
```
# For the specific case of this function, check the structure of test_topology
str(topology)
```

```
## List of 2
## $ : num [1:3] 1 2 3
## $ : num [1:3] 3 4 5
```

```
# If it's a list, try converting to a numeric vector
# For example:
test_topology_numeric <- unlist(topology)
```

```
# Then try the function
analyze_topology_factors(test_topology_numeric)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```



```
## factor threshold base_size max_set_size min_set_size
## 1 1 1.50000 12 6 0
## 2 2 0.75000 7 6 0
## 3 4 0.37500 7 6 0
## 4 8 0.18750 7 6 0
## 5 16 0.09375 7 6 0
```

La gráfica obtenida revelará lo que ocurre, al aumentar el factor IQR, con el tamaño de la base (número de conjuntos en la topología). Es esperable que el tamaño disminuya, ya que un factor IQR más grande implica un umbral más alto para considerar dos elementos como conectados. Así, con un umbral más alto, menos elementos se considerarán parte del mismo conjunto, lo que resulta en una topología más simple con menos conjuntos.

Respecto a los tamaños máximos y mínimos del conjunto, es natural que no cambien. El tamaño máximo corresponde al conjunto completo de elementos originales, que siempre estará presente en la topología, independientemente del factor IQR. El tamaño mínimo corresponde al conjunto vacío, que también es parte de la topología por definición. Estos conjuntos extremos no se ven afectados por el factor IQR.

Por tanto, el efecto observado del factor IQR en la topología esperado es que un factor IQR más grande genere una topología más simple, mientras que los conjuntos extremos (completo y vacío) permanezcan inalterados.

e. Método Basado en la Densidad (similar al usado en DBSCAN^[6])

```
k <- ceiling(log(length(datos)))
sorted_distances <- sort(dist(matrix(datos, ncol=1)))
threshold <- sorted_distances[k * length(datos)]
```

Este método tiene la ventaja de adapta el umbral a la densidad local de los datos, pero su desventaja es que puede ser computacionalmente más costoso para conjuntos de datos grandes (que es lo que andamos buscando evitar).

Sin mayores pruebas preliminares, entre las opciones recomendadas parecería ser más adecuado usar la mediana de las distancias entre puntos adyacentes o el método basado en la densidad. La razón es que estos métodos son robustos frente a valores atípicos y se adaptan bien a la estructura local de los datos.

Sin embargo, lo más recomendable es hacer las pruebas pertinentes para determinar cuál de las heurísticas en cuestión es la adecuada para cada conjunto de datos concreto de los que el investigador de turno disponga. En definitiva, la elección de la heurística dependerá de la o las propiedades concretas de la topología global que se deseen estudiar. En este caso, la propiedad deseada es la conexidad o conectividad.

3. Funciones del Paquete

3.1. `is_topology_connected()`

Propósito: Verifica si una topología está completamente conectada utilizando un enfoque de grafo no dirigido. Esta función es especialmente útil para analizar la interconexión de conjuntos de datos.

Funcionamiento:

1. Convierte la topología (lista de conjuntos) en una matriz de adyacencia no dirigida.
2. Realiza una búsqueda en profundidad (DFS) para explorar la conectividad entre elementos.
3. Determina si todos los elementos de la topología pueden ser alcanzados desde un punto inicial.

Ejemplo de uso:

```
topology <- list(c(1,2,3), c(3,4,5))
is_topology_connected(topology) # Devuelve TRUE o FALSE
```

```
## [1] TRUE
```

Casos especiales:

1. Si la topología está vacía, devuelve FALSE.
2. Maneja topologías con elementos dispersos o no consecutivos.

3.2. `is_topology_connected2()`

Propósito: Similar a `is_topology_connected`, pero utiliza un enfoque de grafo dirigido, lo que permite análisis más específicos de relaciones direccionales.

Funcionamiento:

1. Crea una matriz de adyacencia dirigida basada en el orden de los elementos en cada conjunto.
2. Realiza una búsqueda en profundidad (DFS) para verificar la conectividad.
3. Se centra en las conexiones secuenciales entre elementos.

Ejemplo de uso:

```
topology <- list(c(1,2,3), c(3,4,5))
is_topology_connected2(topology) # Devuelve TRUE o FALSE
```

```
## [1] TRUE
```

Diferencia clave con `is_topology_connected`:

1. Considera las direcciones de las conexiones entre elementos. 2. Más restrictivo en términos de conectividad.

3.3. `is_topology_connected_manual()`

Propósito: Ofrece un método manual y directo para verificar la conectividad de una topología.

Funcionamiento:

1. Comprueba si todos los elementos originales (del 1 al máximo valor) están presentes en al menos un conjunto de la topología.
2. Útil para topologías con estructura específica o restricciones particulares.

Ejemplo de uso:

```
topology <- list(c(1,2,3), c(3,4,5))
is_topology_connected_manual(topology) # Devuelve TRUE o FALSE
```

```
## [1] TRUE
```

Caso de uso especial: Ideal para topologías con requisitos de completitud específicos.

3.4. `analyze_topology_factors()`

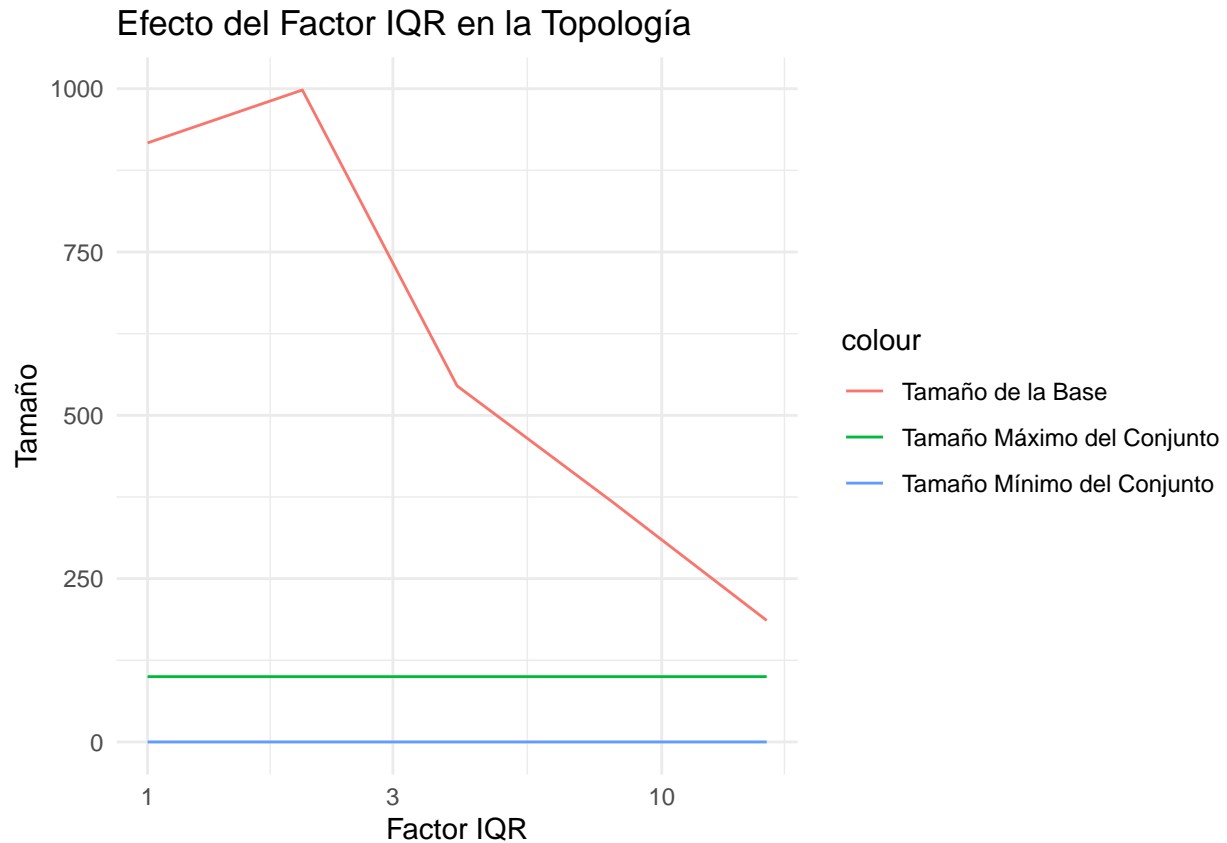
Propósito: Analiza cómo diferentes factores del Rango Inter cuartílico (IQR) afectan las características de una topología.

Funcionamiento:

1. Prueba múltiples factores de IQR (por defecto: 1, 2, 4, 8, 16).
2. Calcula umbrales, genera subbases y base topológica para cada factor.
3. Produce un resumen de las características de la topología.
4. Opcionalmente genera una visualización gráfica.

Ejemplo de uso:

```
data <- rnorm(100)
topologyR::analyze_topology_factors(data)
```

```
## factor threshold base_size max_set_size min_set_size
## 1 1 1.33402636 917 100 0
## 2 2 0.66701318 998 100 0
## 3 4 0.33350659 545 100 0
## 4 8 0.16675330 368 100 0
## 5 16 0.08337665 186 100 0
```

Análisis de factores IQR con gráfica incluida

Características principales:

1. Ayuda a elegir el factor IQR óptimo para un conjunto de datos.
2. Proporciona información sobre el tamaño de la base topológica.

3.5. calculate_thresholds()

Propósito: Calcula múltiples umbrales para el análisis topológico utilizando diferentes métodos estadísticos.

Métodos de cálculo:

1. Diferencia media
2. Diferencia mediana
3. Desviación estándar
4. IQR ajustado
5. Método DBSCAN

Ejemplo de uso:

```
data <- c(1, 2, 3, 4, 5)
topologyR::calculate_thresholds(data)
```

```
## $mean_diff
## [1] 1
##
## $median_diff
## [1] 1
##
## $sd
## [1] 1.581139
##
## $iqr
## [1] 0.5
##
## $dbscan
## [1] 4
```

```
# Devuelve una lista con diferentes umbrales
```

Utilidad: Proporciona múltiples perspectivas para definir umbrales en análisis topológico.

3.6. visualize__topology__thresholds()

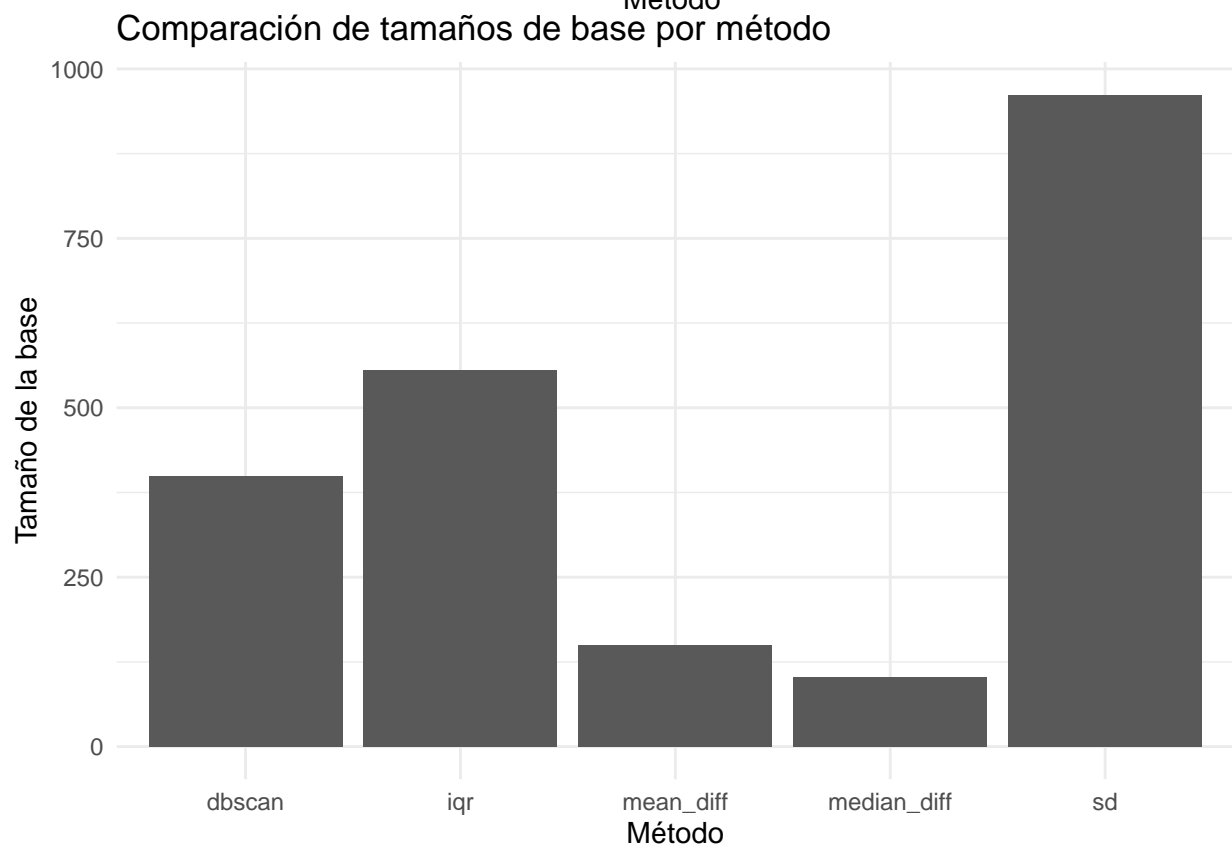
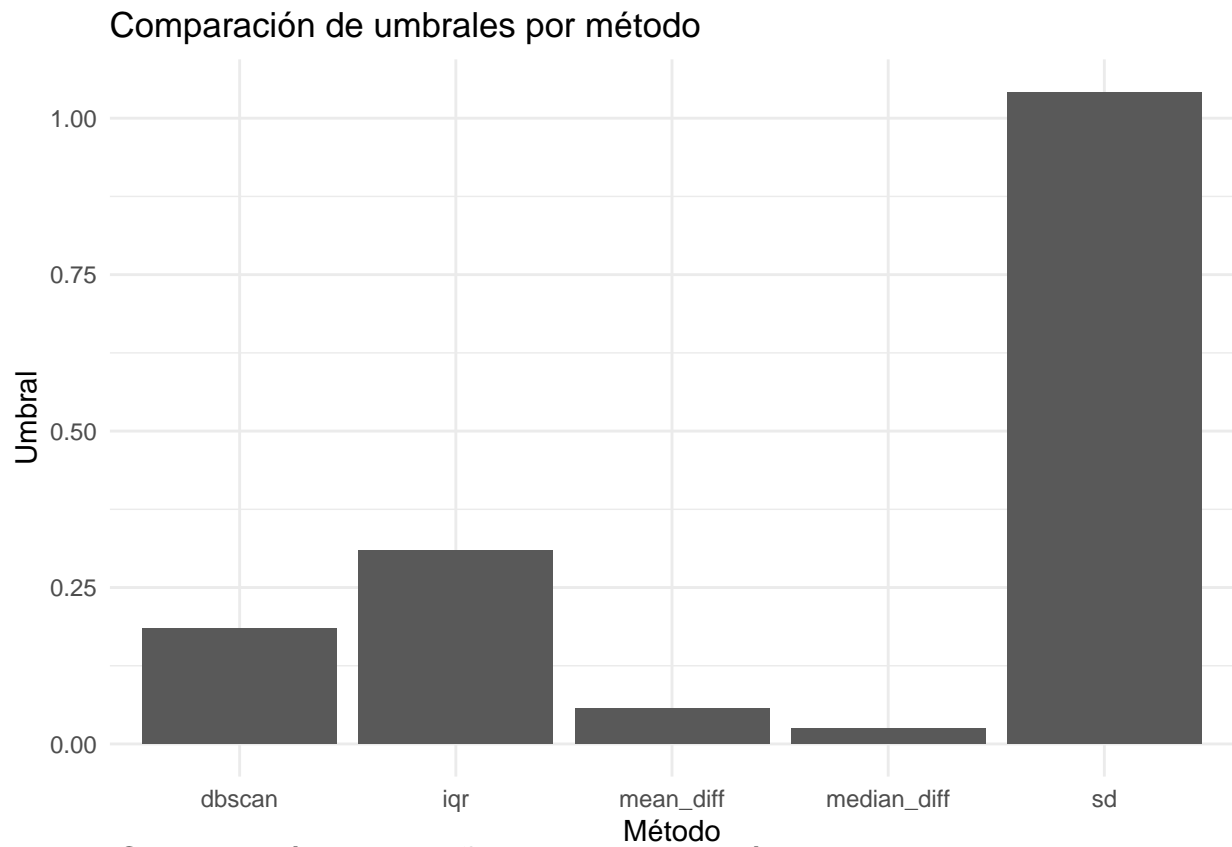
Propósito: Visualiza y compara los umbrales calculados y sus efectos en la estructura topológica.

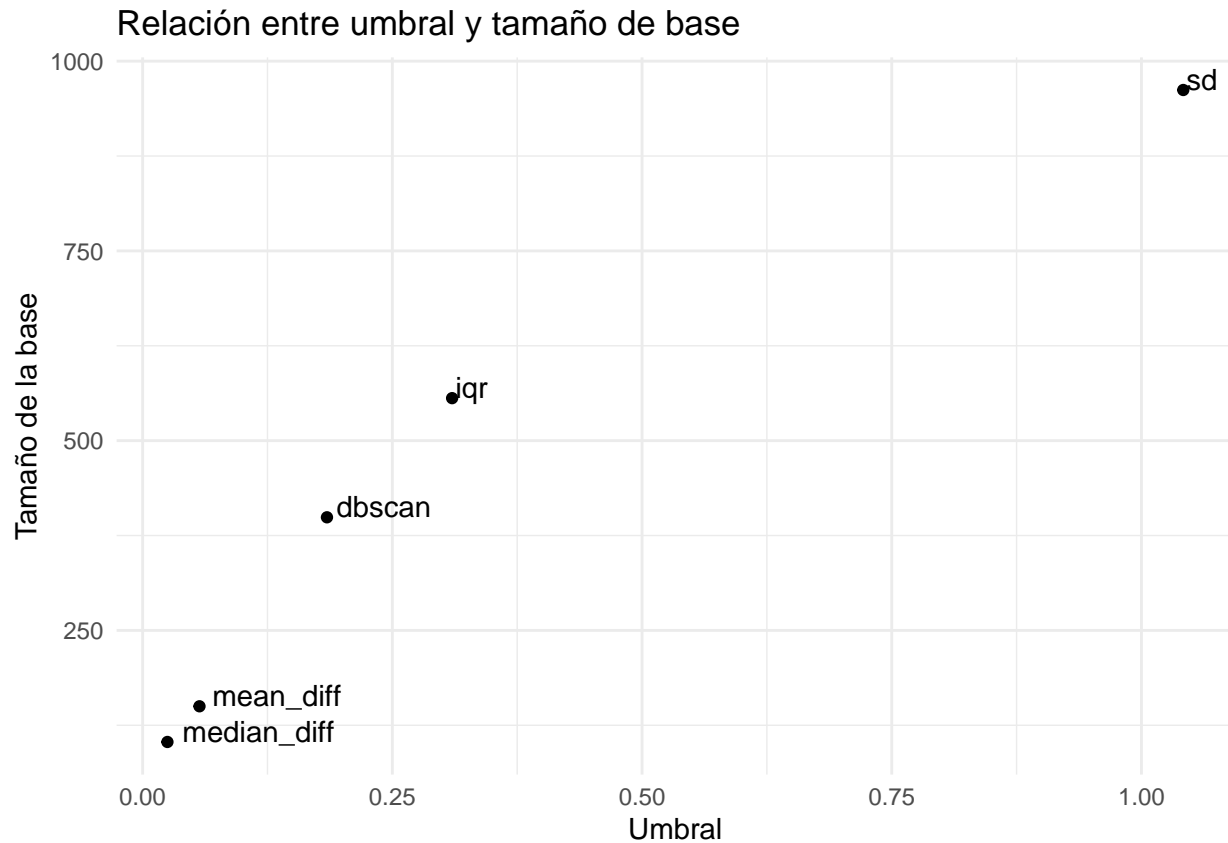
Funcionamiento:

1. Calcula umbrales usando calculate_thresholds
2. Genera gráficos comparativos: 2.1. Comparación de umbrales por método 2.2. Tamaños de base por método 2.3. Relación entre umbral y tamaño de base

Ejemplo de uso:

```
data <- rnorm(100)
topologyR::visualize_topology_thresholds(data, plot = TRUE)
```





```
##           method threshold base_size
## mean_diff mean_diff 0.05691638      150
## median_diff median_diff 0.02480864      103
## sd          sd 1.04187003      962
## iqr         iqr 0.30984339      556
## dbscan      dbscan 0.18454807      399
```

Genera tres gráficos y devuelve un marco de datos

Características principales:

1. Visualización interactiva de las características topológicas
2. Ayuda a comprender el impacto de diferentes métodos de umbralización

4. Aplicaciones

4.1. Análisis de Series Temporales Económicas

El paquete `topologyR` proporciona herramientas especializadas para analizar datos de series temporales económicas mediante métodos topológicos. Esta sección demuestra aplicaciones prácticas utilizando datos reales del PIB.

4.1.1. Preparación de Datos y Análisis de Distribución

Al analizar datos de series temporales económicas, particularmente cambios en el PIB, la elección de la distribución es crucial para la construcción de la topología para grandes conjuntos de datos cuando enfrentamos restricciones en recursos computacionales.

Sin embargo, antes de determinar la distribución óptima de nuestros datos, debemos determinar el método óptimo para ajustar empíricamente nuestros datos a distribuciones teóricas o, para ser más precisos, a distribuciones empíricas que sigan perfectamente una distribución teórica.

MLE (Estimación por Máxima Verosimilitud)

Cuándo Usar:

Muestras grandes con especificación correcta del modelo paramétrico.

Priorizar eficiencia estadística (varianza asintótica mínima).

Datos bien comportados (sin colas pesadas/valores atípicos; ej., distribución normal).

Forma distribucional conocida (se cumplen supuestos paramétricos).

Evitar: Muestras pequeñas, modelos mal especificados o distribuciones de colas pesadas.

Ejemplo: Ajustar una distribución normal a datos limpios y simétricos.

MME (Estimación por Igualación de Momentos)

Cuándo Usar:

Muestras pequeñas o se requiere simplicidad computacional.

Forma distribucional incierta, pero los momentos son calculables/estables.

Aplicaciones en tiempo real que priorizan velocidad sobre precisión.

Evitar para distribuciones con momentos indefinidos (ej., Cauchy).

Ejemplo: Estimación rápida de parámetros para distribuciones gamma/beta.

QME (Estimación por Igualación de Cuantiles)

Cuándo Usar:

Enfoque en comportamiento de colas/cuantiles (ej., VaR en finanzas).

Robustez a valores atípicos/datos censurados (cuantiles son menos sensibles).

Distribuciones asimétricas/colas pesadas (ej., Pareto para pérdidas extremas).

Ejemplo: Modelado de extremos de reclamos de seguros usando el percentil 95.

MGE (Estimación por Máxima Bondad de Ajuste)

Cuándo Usar:

Validación de hipótesis distribucionales (ej., pruebas KS/AD).

Comparación global de múltiples distribuciones.

Distribuciones no estándar donde MLE/MME fallan.

Ejemplo: Probar si los datos siguen distribuciones logística vs. Gumbel.

MSE (Estimación por Máximo Espaciamiento)

Cuándo Usar:

Datos con brechas/redondeo (ej., mediciones irregulares).

Distribuciones de colas pesadas o muestras continuas pequeñas.

Evita dependencia excesiva de valores extremos (robustez basada en espaciamiento).

Ejemplo: Ajustar una distribución Weibull a datos irregulares de tiempo hasta fallo.

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.4.3
```

```
estimation_methods <- data.frame(  
  Method = c("MLE", "MME", "QME", "MGE", "MSE"),  
  "Cuándo_Usar" = c(  
    "Muestras grandes, modelo correcto, prioridad en eficiencia",  
    "Muestras pequeñas, prioridad en velocidad, momentos existentes",  
    "Enfoque en colas/cuantiles, robustez ante valores atípicos",  
    "Validación/comparación de distribuciones, casos no estándar",  
    "Datos con brechas, colas pesadas, muestras pequeñas"  
  ),  
  "Ventajas_Clave" = c(  
    "Eficiencia asintótica, precisión paramétrica",  
    "Simplicidad computacional, no requiere verosimilitud",  
    "Alineación robusta de cuantiles, precisión en colas",  
    "Evaluación global del ajuste, pruebas de hipótesis",  
    "Robusto ante brechas/valores atípicos, consistencia en espaciamiento"  
  ),  
  "Funciones_en_R" = c(  
    "`fitdistr` (MASS), `mle` (stats4)",  
    "`optim` + ecuaciones manuales de momentos",  
    "`qme` (personalizado), `quantreg::rq`",  
    "`gofest`, `ks.test`, `ad.test`",  
    "`mps` (POT), paquetes `MPS.est`"  
  )  
)  
  
colnames(estimation_methods) <- gsub("_", " ", colnames(estimation_methods))
```

```
kable(
  estimation_methods,
  format = "pipe",
  caption = "**COMPARATIVA DE MÉTODOS DE ESTIMACIÓN**",
  align = c("c", "c", "c", "c") # Centrar todas las columnas
)
```

Table 1: COMPARATIVA DE MÉTODOS DE ESTIMACIÓN

Method	Cuándo Usar	Ventajas Clave	Funciones en R
MLE	Muestras grandes, modelo correcto, prioridad en eficiencia	Eficiencia asintótica, precisión paramétrica	<code>fitdistr</code> (MASS), <code>mle</code> (stats4)
MME	Muestras pequeñas, prioridad en velocidad, momentos existentes	Simplicidad computacional, no requiere verosimilitud	<code>optim</code> + ecuaciones manuales de momentos
QME	Enfoque en colas/cuantiles, robustez ante valores atípicos	Alineación robusta de cuantiles, precisión en colas	<code>qme</code> (personalizado), <code>quantreg::rq</code>
MGE	Validación/comparación de distribuciones, casos no estándar	Evaluación global del ajuste, pruebas de hipótesis	<code>gofest</code> , <code>ks.test</code> , <code>ad.test</code>
MSE	Datos con brechas, colas pesadas, muestras pequeñas	Robusto ante brechas/valores atípicos, consistencia en espaciamiento	<code>mps</code> (POT), paquetes <code>MPS.est</code>

Vamos a proporcionar a los usuarios de esta biblioteca una función personalizada para ajustar múltiples tipos de distribuciones simultáneamente. Para esto, proporcionaremos al usuario datos del crecimiento trimestral del PIB de EE.UU. desde el primer trimestre de 1992 hasta el primer trimestre de 2024 después de aplicar una transformación Yeo-Johnson a los datos (con un valor para el hiperparámetro λ igual a 0.5).

```
# Crear el vector con los datos proporcionados
X <- c(
  0.0119368777374702, 0.0108137656182881, 0.00985681082011425, 0.0104015519293652,
  0.00166870385685947, 0.00581325152667178, 0.00476582173579576, 0.0135555616868386,
  0.00968017355996231, 0.0135061956696334, 0.005837281536067, 0.0114230783204214,
  0.00354465884841204, 0.00298097844188216, 0.00848928301845797, 0.00677931023817369,
  0.00747652539201571, 0.0166104234581352, 0.00895027315262364, 0.0103568837398003,
  0.00644401865589073, 0.0165834473187565, 0.0124469682453747, 0.00852045048090089,
  0.0100124377724629, 0.00923557603383074, 0.0125500242230006, 0.016027775864102,
  0.00937233981161389, 0.00833005255610297, 0.0132151400185725, 0.016339951496275,
  0.00362461554054549, 0.0181307192548257, 0.00101904038917144, 0.00596151508447429,
  -0.00328068485517437, 0.00623528031983334, -0.00401041013477036, 0.00274591498772025,
  0.00834618529774378, 0.00611774330421611, 0.00406097711621545, 0.00123571825010149,
  0.0052594844558147, 0.0088366782792475, 0.0165634133346764, 0.011573712295923,
  0.00565849535757224, 0.00773454420647957, 0.00946351049229044, 0.0101762111814976,
  0.0110630024939544, 0.00492014803582652, 0.00782280094633858, 0.00554750629348089,
  0.0134088506808547, 0.00258582837290655, 0.00149903822110309, 0.00857611257328283,
  0.00300614077940331, 0.00610827225252253, 0.00575202854191303, 0.00627256373604412,
  -0.00427214986914152, 0.00594546286782194, -0.00525949142529332, -0.0220096627946815,
  -0.0113808376042674, -0.00178729765814796, 0.00350882204196923, 0.0107800476432023,
  0.0048391456673027, 0.00965260679551294, 0.0076959929232312, 0.00524422452727702,
  -0.00237340604041976, 0.00675439453860394, -0.000223112442939784, 0.0111991447889985,
  0.00836839250173416, 0.00445833082157154, 0.001439781757123, 0.00115606587792128,
```

```

0.00984138677657853, 0.0026746116131795, 0.0084958551114811, 0.0086984840936184,
-0.00345397563998976, 0.0128769460650098, 0.0121171934059903, 0.00505092204661972,
0.00898461915466164, 0.00618453787282514, 0.00399850299345283, 0.00184394996213433,
0.00578682815497622, 0.0032086261795099, 0.00708136357248845, 0.00553534000276334,
0.00486298783732364, 0.00559088549983189, 0.00786971688902183, 0.01123892165998,
0.00811762603688138, 0.00530127412316794, 0.00622730516758763, 0.00141589880764181,
0.00542364601597312, 0.00827956221239301, 0.0112899343456179, 0.00640295055604412,
-0.0136749297466007, -0.0804470145766672, 0.0761427696572317, 0.0103255457761064,
0.0128141493938281, 0.01514019363418, 0.00812718720702543, 0.0168963285206307,
-0.0049829870079526, -0.0014131988129115, 0.00657539105810834, 0.00634413797832822,
0.00555608248684969, 0.00510468554636834, 0.011903277993254, 0.00836650041769005,
0.00394730469640825
) # Real GDP Growth

library(fitdistrplus)

```

```
## Warning: package 'fitdistrplus' was built under R version 4.4.3
```

```
## Cargando paquete requerido: MASS
```

```
## Warning: package 'MASS' was built under R version 4.4.3
```

```
## Cargando paquete requerido: survival
```

```
## Warning: package 'survival' was built under R version 4.4.3
```

```
library(dplyr)
```

```
##
```

```
## Adjuntando el paquete: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
```

```
##
```

```
##      select
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##      filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      intersect, setdiff, setequal, union
```

```
# Ejemplo de ajuste de distribución
```

```
x <- X # Vector de cambios en el PIB Real
```

```
distributions <- c("norm", "cauchy", "logis", "unif", "t")
```

```
fits <- list()
```

```
set.seed(100000)
```

```
for(dist in distributions) {
```



```

if(dist == "t") {
  fits[[dist]] <- fitdist(x, dist, start=list(df=length(x)-1), method="mge")
} else {
  fits[[dist]] <- fitdist(x, dist, method="mge")
}
}

```

```

## Warning in fitdist(x, dist, method = "mge"): maximum GOF estimation has a
## default 'gof' argument set to 'CvM'

```

```

## Warning in fitdist(x, dist, method = "mge"): maximum GOF estimation has a
## default 'gof' argument set to 'CvM'
## Warning in fitdist(x, dist, method = "mge"): maximum GOF estimation has a
## default 'gof' argument set to 'CvM'
## Warning in fitdist(x, dist, method = "mge"): maximum GOF estimation has a
## default 'gof' argument set to 'CvM'

```

```

## Warning in fitdist(x, dist, start = list(df = length(x) - 1), method = "mge"):
## maximum GOF estimation has a default 'gof' argument set to 'CvM'

```

```
fits$norm$estimate
```

```

##          mean          sd
## 0.006940808 0.004746252

```

```
fits$norm$bic
```

```
## [1] -381.9854
```

```
fits$cauchy$estimate
```

```

##      location      scale
## 0.006903160 0.002882685

```

```
fits$cauchy$bic
```

```
## [1] -922.4761
```

```
fits$logis$estimate
```

```

##      location      scale
## 0.006937262 0.002859244

```

```
fits$logis$bic
```

```
## [1] -861.6542
```

```
fits$unif$estimate
```

```
##           min           max  
## 0.0000480909 0.013884505
```

```
fits$unif$bic
```

```
## [1] Inf
```

```
fits$t$estimate
```

```
## df  
## 128
```

```
fits$t$bic
```

```
## [1] 242.4721
```

```
library(topologyR)
```

```
X2 = rcauchy(length(X), 0.006903160, 0.002882685)
```

```
X3 = rcauchy(10, 0.006903160, 0.002882685)
```

4.1.2. Análisis Topológico Básico

Para conjuntos de datos con tamaño igual o menor a 30 observaciones:

```
# Crear la topología inicial  
topology <- complete_topology(X3)  
length(topology$R)
```

```
## [1] 100
```

```
length(topology$subbase)
```

```
## [1] 10
```

```
length(topology$base)
```

```
## [1] 57
```

```
length(topology$topology)
```

```
## [1] 190
```

```
# Verificar conexidad utilizando diferentes métodos
is_connected_undirected <- is_topology_connected(topology$topology)
is_connected_directed <- is_topology_connected2(topology$topology)
is_connected_manual <- is_topology_connected_manual(topology$topology)

is_connected_undirected
```

```
## [1] TRUE
```

```
is_connected_directed
```

```
## [1] TRUE
```

```
is_connected_manual
```

```
## [1] TRUE
```

4.1.3. Análisis Avanzado con Umbrales

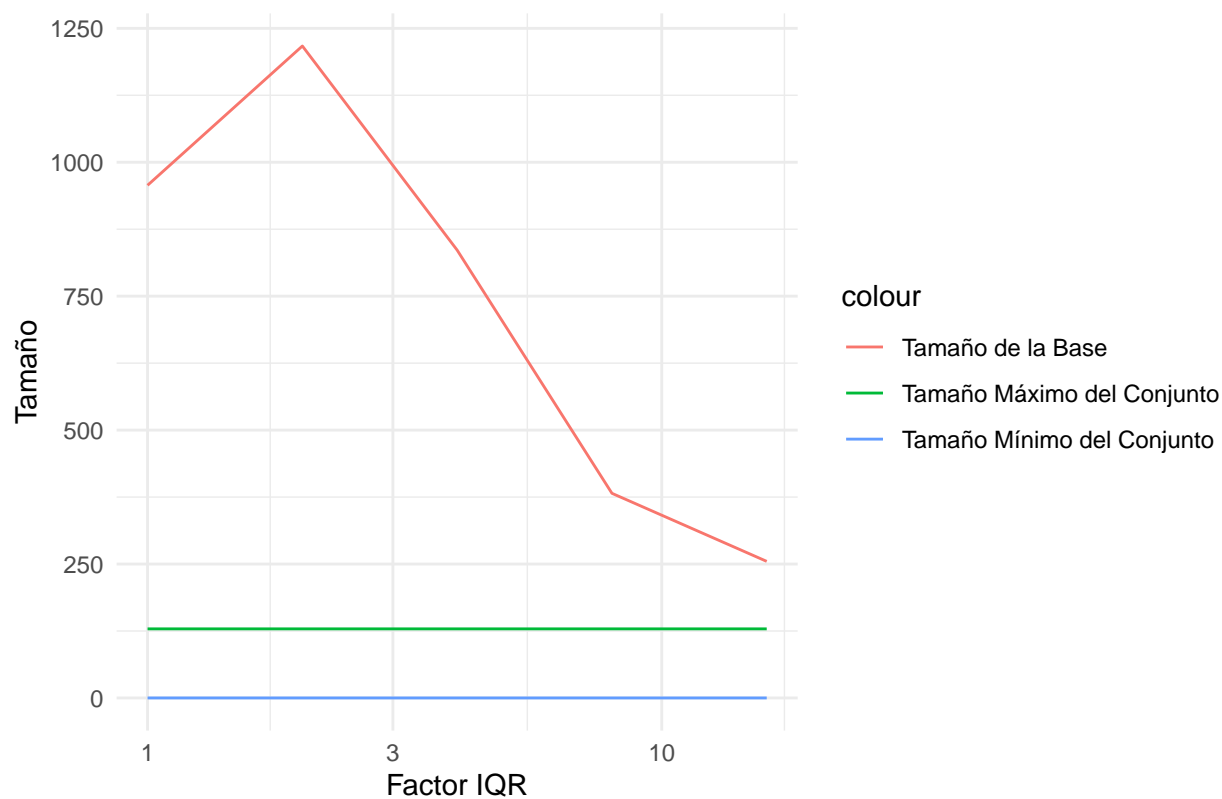
Para conjuntos de datos más grandes ($n > 30$):

```
# Calcula umbrales óptimos
thresholds <- calculate_thresholds(X)
thresholds
```

```
## $mean_diff
## [1] 0.001223358
##
## $median_diff
## [1] 0.0001172427
##
## $sd
## [1] 0.01148499
##
## $iqr
## [1] 0.001503484
##
## $dbscan
## [1] 0.0006469434
```

```
# Analiza características de la topología con diferentes factores IQR (rango intercuartílico)
results <- analyze_topology_factors(X, factors = c(1, 2, 4, 8, 16))
```

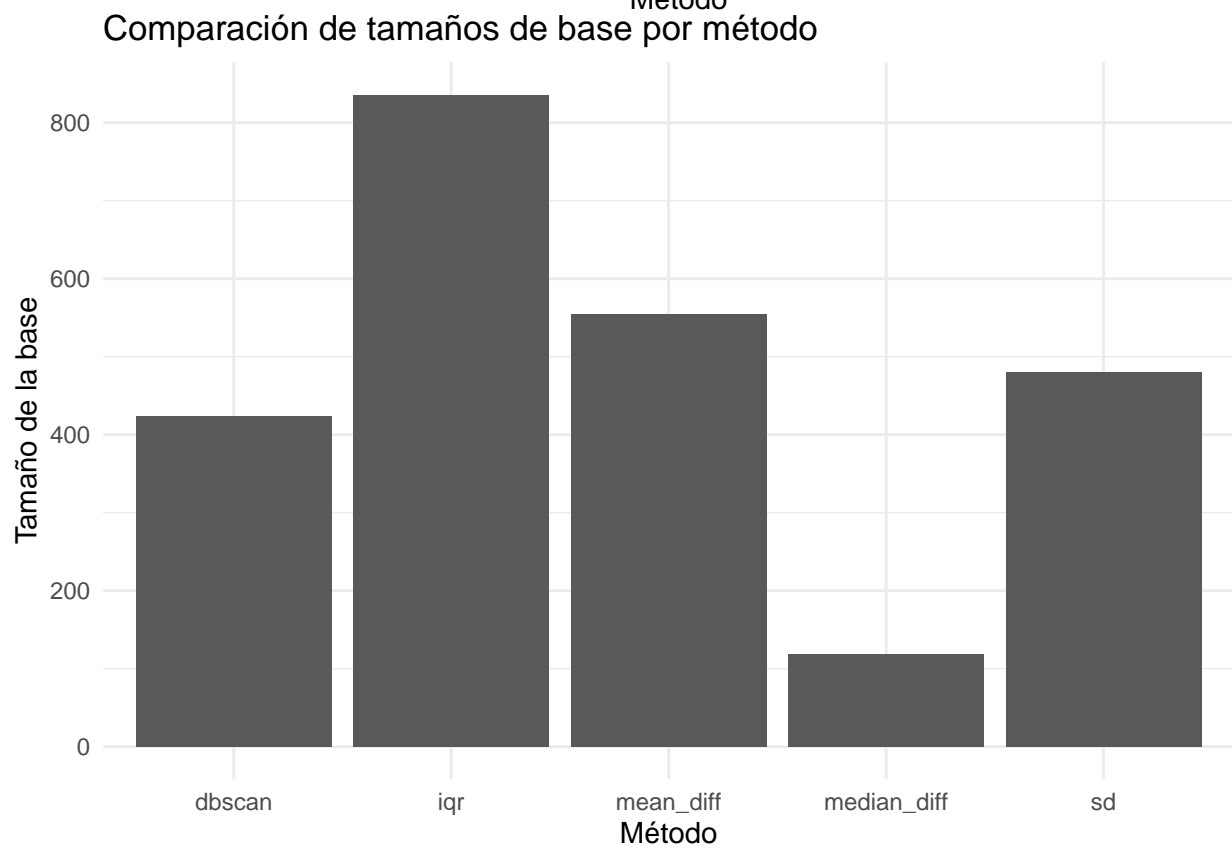
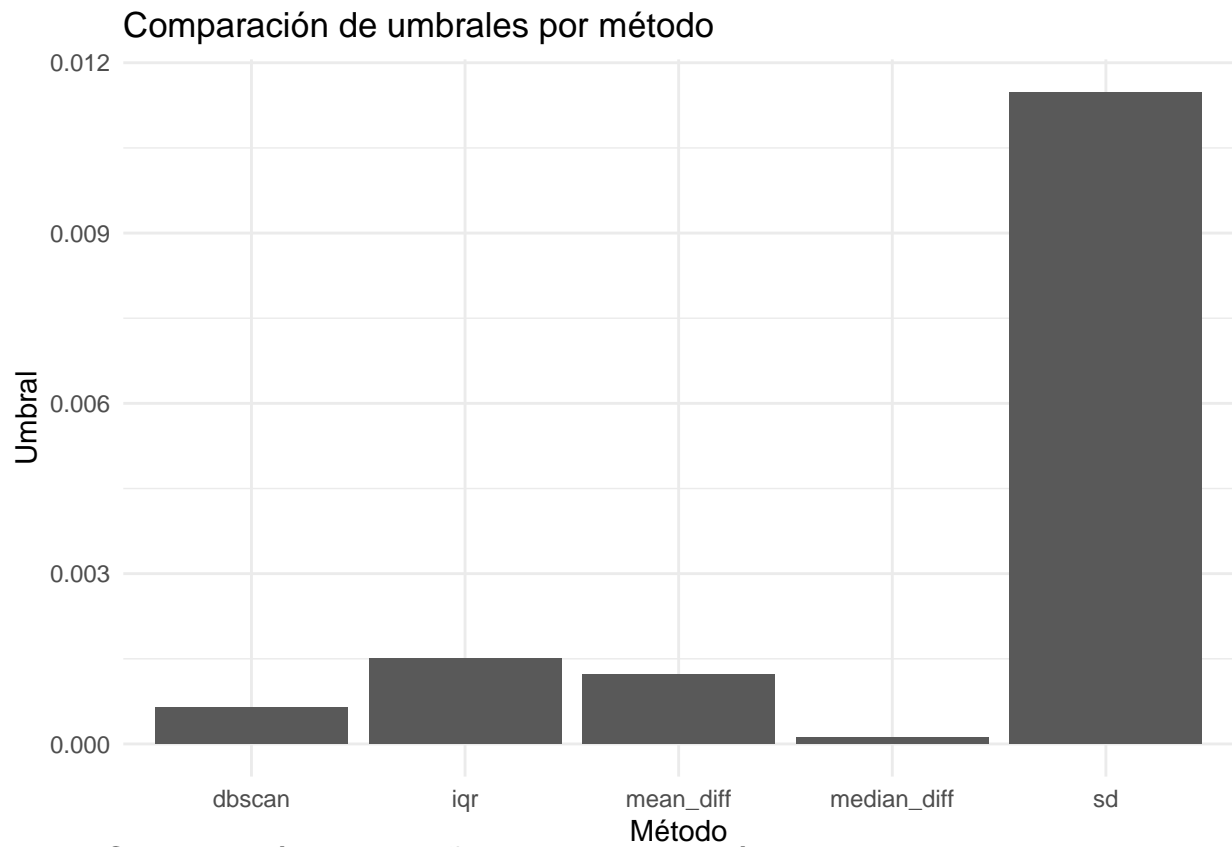
Efecto del Factor IQR en la Topología

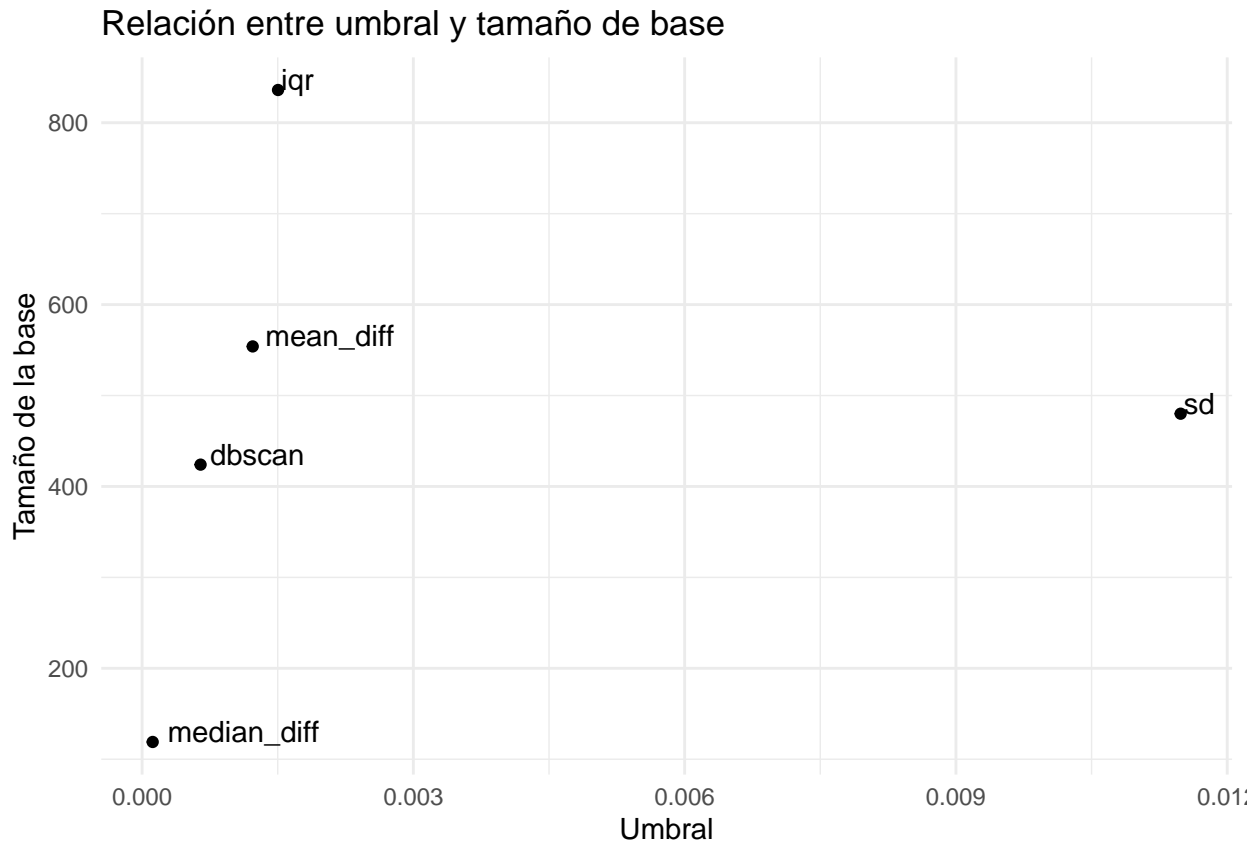


results

```
## factor threshold base_size max_set_size min_set_size
## 1 1 0.0060139348 957 129 0
## 2 2 0.0030069674 1217 129 0
## 3 4 0.0015034837 836 129 0
## 4 8 0.0007517418 382 129 0
## 5 16 0.0003758709 255 129 0
```

```
# Visualiza Visualize threshold comparisons
viz_results <- visualize_topology_thresholds(X)
```





```
viz_results
```

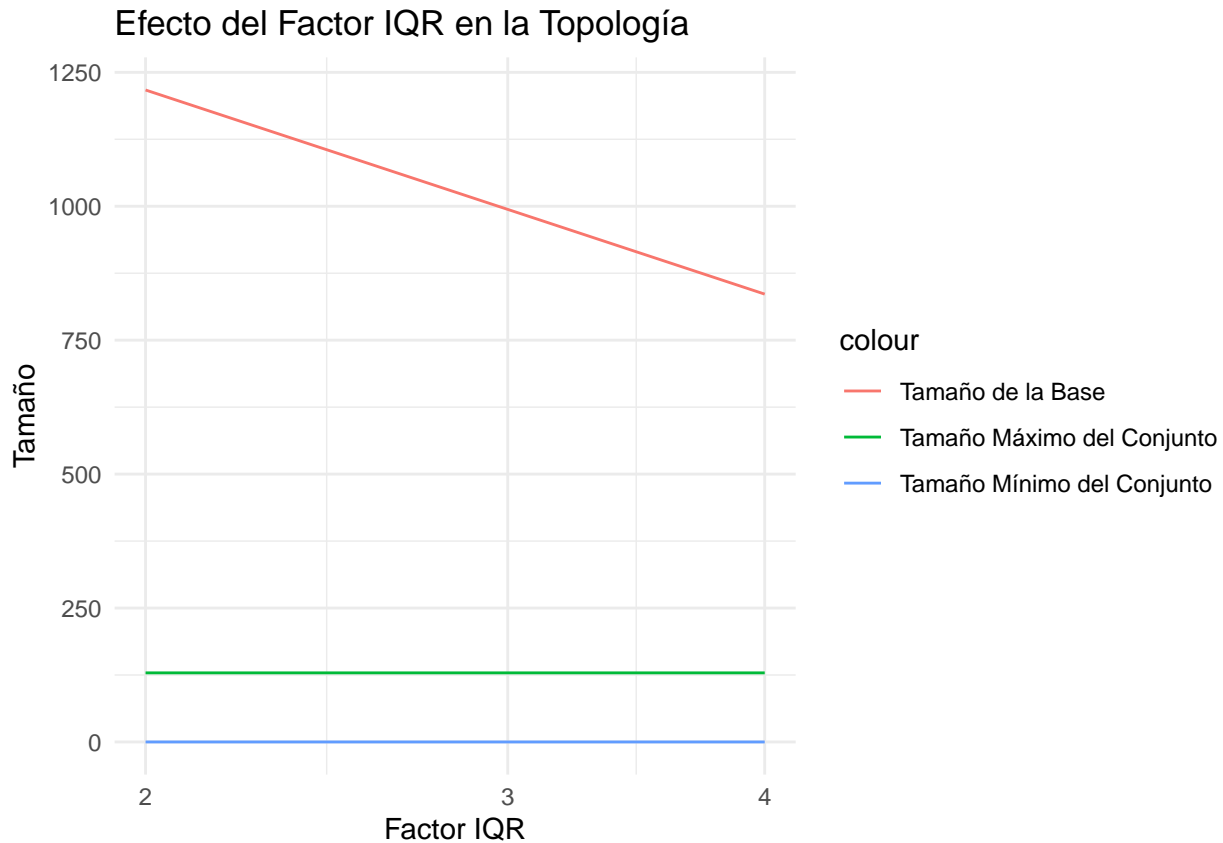
```
##           method   threshold base_size
## mean_diff mean_diff 0.0012233577    554
## median_diff median_diff 0.0001172427    119
## sd          sd      0.0114849870    480
## iqr         iqr     0.0015034837    836
## dbscan      dbscan  0.0006469434    424
```

4.1.4. Pautas de Optimización de Rendimiento

Recomendaciones según Tamaño del Dataset

- $n \leq 30$: Usar `complete_topology()`.
- $30 < n \leq 100$: Usar métodos basados en umbrales con parámetros por defecto.
- $n > 100$: Usar umbrales optimizados:

```
threshold <- IQR(x)/4 # Default conservative threshold
results <- analyze_topology_factors(x, factors = c(2, 4))
```



4.1.5. Marco de Interpretación

1. Análisis de Conectividad

- Topología conectada: Período económico estable.
- Topología desconectada: Posibles rupturas estructurales.

2. Interpretación del Tamaño de la Base

```
base_size <- length(topology$base)
relative_size <- base_size/length(x)
```

- $\text{relative_size} < 0.3$: Estructura simple. Basado en la teoría de complejidad de Kolmogorov, una topología con menos conjuntos que el 30% de las combinaciones posibles indica alta compresibilidad de datos. Esto corresponde a una fuerte regularidad en la dinámica económica subyacente, lo cual se alinea con el principio de longitud mínima de descripción, sugiriendo una representación parsimoniosa.
- $0.3 \leq \text{relative_size} \leq 0.7$: Complejidad moderada. Derivado de consideraciones de entropía en sistemas dinámicos, refleja un balance entre orden y desorden típico de sistemas económicos estables. Este rango corresponde a la región de transición de fase en redes complejas y está respaldado por estudios empíricos que muestran que la mayoría de los períodos económicos estables exhiben bases topológicas en este rango.
- $\text{relative_size} > 0.7$: Estructura compleja. Se basa en la teoría de grafos aleatorios, donde el alto tamaño relativo indica una estructura casi aleatoria. Esto sugiere posibles ineficiencias de mercado o inestabilidades estructurales y corresponde a una alta complejidad algorítmica en el sentido de Chaitin. Se observa típicamente durante períodos de turbulencia económica o cambio sistémico.

Las consideraciones anteriores no deben confundirse con la complejidad en el sentido de la teoría de sistemas complejos/caos.

La complejidad topológica mide la riqueza de la estructura de vecindad y patrones de conectividad. Un `relative_size` alto indica muchos conjuntos abiertos distintos y relaciones locales intrincadas. La complejidad en teoría del caos mide la sensibilidad del sistema a las condiciones iniciales y la predictibilidad a largo plazo. Un sistema caótico puede tener una topología simple (pocos conjuntos abiertos) pero una dinámica compleja.

Por ejemplo, el mapa logístico puede exhibir caos con una estructura topológica relativamente simple (`relative_size < 0.3`), mientras que un sistema periódico estable podría requerir una topología compleja (`relative_size > 0.7`) para capturar sus relaciones de vecindad.

4.2. Ejemplos de Análisis Comparativo

```
#Datos trimestrales del crecimiento del PIB real desde 1960-04-01 hasta 2024-01-01 (Datos FRED)
x <- c(
  2.2, -0.5, 0.5, -1.3, 0.7, 1.7, 1.9, 2.0, 1.8, 0.9, 1.2, 0.3, 1.1, 1.1, 2.2,
  0.7, 2.1, 1.1, 1.6, 0.3, 2.4, 1.3, 2.2, 2.3, 2.4, 0.3, 0.8, 0.8, 0.9, 0.1, 0.9,
  0.8, 2.0, 1.7, 0.8, 0.4, 1.6, 0.3, 0.7, -0.5, -0.1, 0.1, 0.9, -1.1, 2.7, 0.5, 0.8,
  0.2, 1.8, 2.3, 0.9, 1.7, 2.5, 1.1, -0.5, 0.9, -0.9, 0.2, -0.9, -0.4, -1.2, 0.7,
  1.7, 1.3, 2.2, 0.7, 0.5, 0.7, 1.2, 1.9, 1.8, 0.0, 0.3, 3.9, 1.0, 1.3, 0.2, 0.1,
  0.7, 0.3, 0.3, -2.1, -0.1, 1.9, 2.0, -0.7, 1.2, -1.1, -1.6, 0.5, -0.4, 0.0, 1.3,
  2.3, 2.0, 2.1, 2.0, 1.7, 1.0, 0.8, 1.0, 0.9, 1.5, 0.7, 0.9, 0.5, 1.0, 0.5, 0.7,
  1.1, 0.9, 1.7, 0.5, 1.3, 0.6, 1.3, 1.0, 0.8, 0.7, 0.2, 1.1, 0.4, 0.1, -0.9, -0.5,
  0.8, 0.5, 0.3, 1.2, 1.1, 1.0, 1.0, 0.2, 0.6, 0.5, 1.4, 1.0, 1.4, 0.6, 1.1, 0.4,
  0.3, 0.9, 0.7, 0.7, 1.7, 0.9, 1.0, 0.6, 1.7, 1.2, 0.9, 1.0, 0.9, 1.3, 1.6, 0.9,
  0.8, 1.3, 1.6, 0.4, 1.8, 0.1, 0.6, -0.3, 0.6, -0.4, 0.3, 0.8, 0.6, 0.4, 0.1, 0.5,
  0.9, 1.7, 1.2, 0.6, 0.8, 0.9, 1.0, 1.1, 0.5, 0.8, 0.6, 1.3, 0.3, 0.1, 0.9, 0.3,
  0.6, 0.6, 0.6, -0.4, 0.6, -0.5, -2.2, -1.1, -0.2, 0.4, 1.1, 0.5, 1.0, 0.8, 0.5,
  -0.2, 0.7, 0.0, 1.1, 0.8, 0.4, 0.1, 0.1, 1.0, 0.3, 0.9, 0.9, -0.3, 1.3, 1.2, 0.5,
  0.9, 0.6, 0.4, 0.2, 0.6, 0.3, 0.7, 0.6, 0.5, 0.6, 0.8, 1.1, 0.8, 0.5, 0.6, 0.1,
  0.5, 0.8, 1.1, 0.6, -1.4, -7.9, 7.8, 1.0, 1.3, 1.5, 0.8, 1.7, -0.5, -0.1, 0.7,
  0.6, 0.6, 0.5, 1.2, 0.8, 0.4
)

# Comparar múltiples períodos de tiempo
periods <- list(
  pre_crisis = x[1:39],
  various_crisis = x[40:125],
  post_crisis = x[126:192],
  all_period = x
)

# Verificación de calidad de datos
for(name in names(periods)) {
  missing <- sum(is.na(periods[[name]]))
  if(missing > 0) {
    warning(sprintf("El período %s tiene %d valores faltantes", name, missing))
  }
}

results <- lapply(periods, function(period_data) {
```



```

if(length(period_data) <= 30) {
  topology <- complete_topology(period_data)
} else {
  # Usar enfoque basado en umbrales
  threshold <- IQR(period_data)/4
  topology <- calculate_topology(period_data, threshold)
}
list(
  connectivity = is_topology_connected(topology),
  complexity = length(topology)/length(period_data)
)
})

```

results

```

## $pre_crisis
## $pre_crisis$connectivity
## [1] TRUE
##
## $pre_crisis$complexity
## [1] 0.02564103
##
##
## $various_crisis
## $various_crisis$connectivity
## [1] TRUE
##
## $various_crisis$complexity
## [1] 0.01162791
##
##
## $post_crisis
## $post_crisis$connectivity
## [1] TRUE
##
## $post_crisis$complexity
## [1] 0.01492537
##
##
## $all_period
## $all_period$connectivity
## [1] TRUE
##
## $all_period$complexity
## [1] 0.003891051

```

4.3. Algunas Recomendaciones para Tu Conjunto de Datos

4.3.1. Preparación de Datos

```

# Crear un vector numérico de juguete con algunos NA's
dummy_var <- c(1.5, 2.3, NA, 4.7, 5.1, NA, 3.2, 6.4, NA, 7.8)

prepare_data <- function(x) {
  # Eliminar valores faltantes
  x <- na.omit(x)
  # Escalar si es necesario
  if(sd(x) > 1) x <- scale(x)
  # Verificar tamaño mínimo de muestra
  if(length(x) < 10) warning("El tamaño de la muestra puede ser demasiado pequeño")
  x
}

prepared_data <- prepare_data(dummy_var)

```

```

## Warning in prepare_data(dummy_var): El tamaño de la muestra puede ser demasiado
## pequeño

```

```

print(prepared_data) # Muestra valores escalados sin NAs

```

```

##           [,1]
## [1,] -1.3011743
## [2,] -0.9457316
## [3,]  0.1205966
## [4,]  0.2983180
## [5,] -0.5458585
## [6,]  0.8759125
## [7,]  1.4979372
## attr("scaled:center")
## [1] 4.428571
## attr("scaled:scale")
## [1] 2.250714

```

4.3.2. Manejo de Errores:

```

safe_topology_analysis <- function(x) {
  tryCatch({
    threshold <- IQR(x)/4
    topology <- calculate_topology(x, threshold)
    list(
      topology = topology,
      connectivity = is_topology_connected(topology),
      base_size = length(topology)
    )
  }, error = function(e) {
    message("Error en el análisis topológico: ", e$message)
    NULL
  })
}

```

```
result <- safe_topology_analysis(X3)
print(str(result))
```

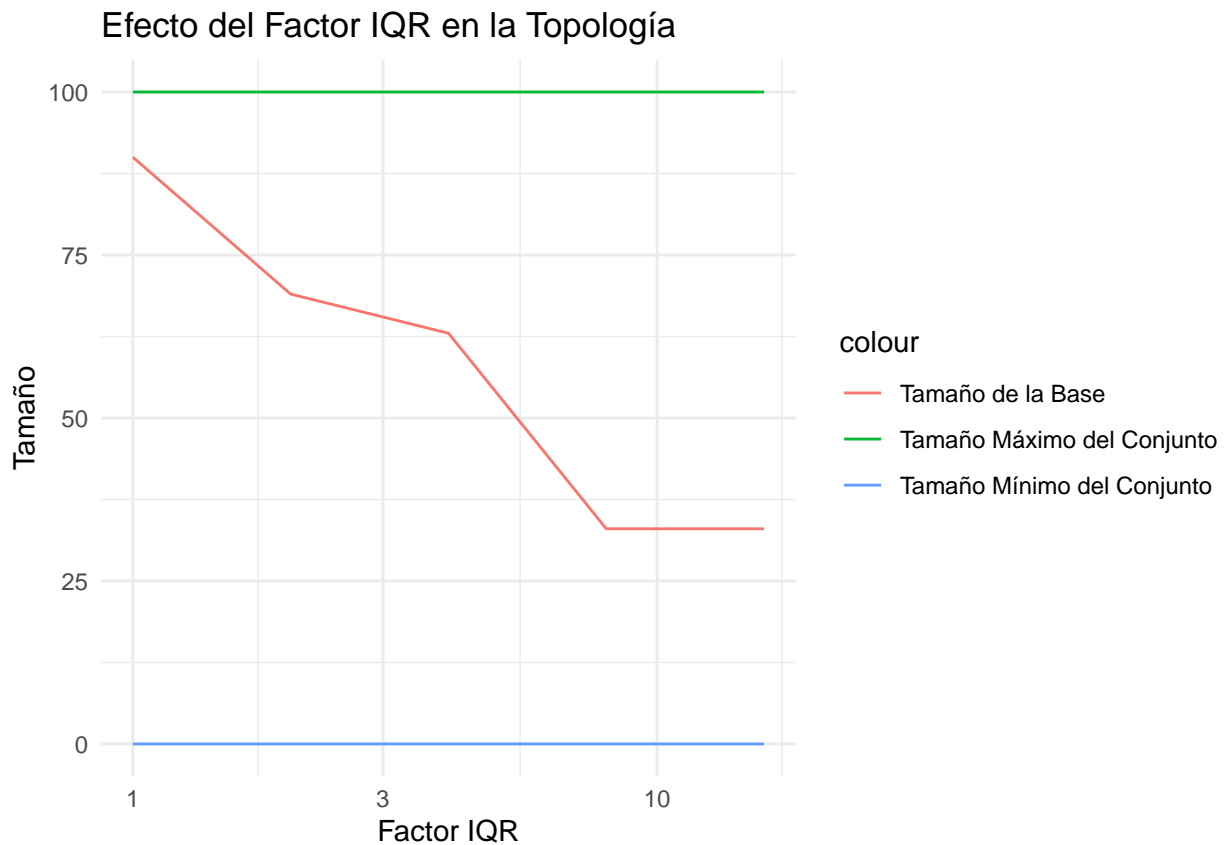
```
## List of 3
## $ topology      : int 8
## $ connectivity: logi TRUE
## $ base_size     : int 1
## NULL
```

4.3.3. Optimización para Grandes Conjuntos de Datos

```
# Para n > 100
analyze_large_dataset <- function(x, sample_size = 100) {
  if(length(x) > sample_size) {
    warning("Usando muestreo para conjunto de datos grande")
    x <- sample(x, sample_size)
  }
  analyze_topology_factors(x)
}

result <- analyze_large_dataset(x)
```

```
## Warning in analyze_large_dataset(x): Usando muestreo para conjunto de datos
## grande
```



```
print(head(result))
```

```
##   factor threshold base_size max_set_size min_set_size
## 1      1    0.6000      90         100           0
## 2      2    0.3000      69         100           0
## 3      4    0.1500      63         100           0
## 4      8    0.0750      33         100           0
## 5     16    0.0375      33         100           0
```

5. Limitaciones y Consideraciones

5.1. Limitaciones Computacionales

- Las funciones asumen que los elementos están numerados secuencialmente desde 1
- Para topologías muy grandes, el tiempo de cómputo puede ser significativo
- Los métodos basados en grafos requieren más memoria que el método manual

5.2. Aproximaciones y Validez

Para conjuntos de datos grandes, se recomienda:

1. Usar `is_topology_connected_manual()` para una primera evaluación rápida.
2. Aplicar `is_topology_connected2()` para análisis más rigurosos cuando sea necesario.
3. Considerar submuestreo para conjuntos extremadamente grandes

6. Referencias

- Kelley, J. L. (1955). *Topología General*. Editorial Universitaria de Buenos Aires.
- Kolmogórov, A. N., & Fomin, S. V. (1978). *Elementos de la Teoría de Funciones y del Análisis Funcional*. Editorial MIR.
- Nada, S., El Atik, A. E. F., & Atef, M. (2018). New types of topological structures via graphs. *Mathematical Methods in the Applied Sciences*, 41(15), 5801-5810.
- Trudeau, R. J. (1993). *Introduction to Graph Theory*. Dover Publications.