Análisis de Redes

Tomás Bossi
tomasbossi97@gmail.com

Francisco Valentini
ft.valentini@gmail.com

Jazmín Vidal
jazmin.vidald@gmail.com

Resumen

Este informe se centra en el análisis de redes mediante el cálculo de autovalores y autovectores. En primer lugar, presentamos los conceptos fundamentales sobre redes y la relevancia de los autovectores y autovalores en el análisis de estas estructuras. Luego, describimos el método de la potencia con deflación para hallar autovectores y autovalores, presentamos una implementación y estudiamos su desemepeño. Por último, exploramos diversas aplicaciones en el análisis de dos redes, el club de Karate [Zachary, 1977] y Facebook [Leskovec and Mcauley, 2012], incluyendo la medición de la centralidad de nodos, la evaluación de la conectividad y la comparación de redes a través de la correlación de autovalores. También abordamos la reducción de la dimensionalidad de matrices de atributos mediante el Análisis de Componentes Principales (PCA), técnica que se vale del cálculo de autovectores.

1 Introducción

Podemos definir una red como el conjunto de los componentes de un sistema, típicamente llamados nodos o vértices, y de las interacciones entre ellos, denominadas aristas [Barabási, 2013]. Las redes pueden representar sistemas de distinta naturaleza, como redes sociales, redes de transporte, redes de comunicación, redes biológicas, etc.

El conjunto de aristas de una red se puede representar mediante una matriz de adyacencia. La matriz de adyacencia $A \in \mathbb{R}^{n\times n}$ de una red con n vértices es una matriz cuyas entradas A_{ij} son cero o uno, dependiendo de si los vértices i y j están conectados o no: $A_{ij} = 1$ si hay un arista desde el nodo i hacia el nodo j, y $A_{ij} = 0$ si los nodos i y j no están conectados.

En este informe nos circunscribimos a redes no dirigidas. Para este tipo de red, la matriz de adyacencia tiene dos entradas iguales para cada arista de la red porque el arista (i, j) se representa como $A_{ij} = 1$ y $A_{ji} = 1$. Por lo tanto, la matriz de adyacencia de una red no dirigida es simétrica, $A = A^T$.

La ecuación de autovalores de la matriz A viene dada por $Av = \lambda v$, donde el número $\lambda \in \mathbb{R}$ es un autovalor y el vector $v \in \mathbb{R}^n$ es el autovector asociado. Si A es simétrica, sabemos que tiene todos sus autovalores $\lambda_1, \ldots, \lambda_n$ reales y que sus autovectores asociados $v_1, \ldots, v_n$ forman una base ortonormal [Strang, 2022]. Podemos ordenar los λ_i de manera tal que $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$. Si se cumple que $|\lambda_1| > |\lambda_2|$, λ_1 es el autovalor dominante y v_1, el autovector dominante de A.

Si A tiene un autovalor dominante, entonces podemos encontrar a éste y al autovector dominante usando el método de la potencia [Watkins, 2004]. La idea es elegir un vector inicial q y formar la secuencia $q, Aq, A^2q, A^3q, \ldots$, de manera que cada vector en la secuencia se obtiene multiplicando el vector anterior por A, es decir: $A^{k+1}q = A(A^kq)$.

Dado que $v_1, \ldots, v_n$ forman una base por tratarse de una matriz simétrica podemos encontrar constantes $c_1, \ldots, c_n$ tales que $q = c_1v_1 + \ldots + c_nv_n$. Multiplicando por A, tenemos que:

$$Aq = c_1Av_1 + \ldots + c_nAv_n = c_1\lambda_1v_1 + \ldots + c_n\lambda_nv_n$$

porque $Av_i = \lambda_iv_i$ para $i = 1, \ldots, n$. Si seguimos multiplicando por A, obtenemos

$$\begin{aligned} A^kq &= c_1\lambda_1^kv_1 + c_2\lambda_2^kv_2 + \ldots + c_n\lambda_n^kv_n \\ &= \lambda_1^k(c_1v_1 + c_2(\lambda_2/\lambda_1)^kv_n + \ldots + c_n(\lambda_n/\lambda_1)^kv_n). \end{aligned} \tag{1}$$

Dado que los vectores son direcciones, podemos escalar por λ_1^{-k}, llegando así a

$$A^k q/\lambda_1^k = c_1 v_1 + c_2(\lambda_2/\lambda_1)^k v_n + \ldots + c_n(\lambda_n/\lambda_1)^k v_n.$$

Como $|\lambda_1| > |\lambda_i|$ para $i = 2, \ldots, n$, los términos $(\lambda_i/\lambda_1)^k$ tiende a cero cuando $k \to \infty$, y el componente en la dirección de v_1 domina en relación a los otros componentes. Es decir, para k suficientemente alto, $A^k q/\lambda_1^k$ es una buena aproximación del autovector dominante.

En la práctica, la secuencia $q_k = A^k q/\lambda_1^k$ es inaccesible porque no conocemos λ_1 de antemano. Además, $||A^k q|| \to \infty$ si $|\lambda_1| > 1$ y $||A^k q|| \to 0$ si $|\lambda_1| < 1$, por lo cual q_k puede volverse demasiado grande o demasiado pequeño, generando underflow o overflow. Por lo tanto, en la práctica usamos algún tipo de normalización para obtener el autovector dominante, por ejemplo:

$$q_{k+1} = Aq_k/||Aq_k||_2 \tag{2}$$

Y al usar la norma-2, la cantidad que converge al autovalor dominante λ_1 es:

$$\lambda_k = q_k^T Aq_k/q_k^T q_k \tag{3}$$

Resumiendo, dado un vector q_0 inicial, el método de la potencia nos devuelve un vector $q_k = Aq_{k-1}/||Aq_{k-1}||_2$ que converge al autovector unitario v_1 de A, mientras que $\lambda_k = q_k^T Aq_k/q_k^T q_k$ converge al autovalor dominante λ_1.

Si además asumimos que $|\lambda_2| > |\lambda_3|$, podemos aplicar el método de deflación para obtener el segundo autovalor dominante λ_2 y su autovector asociado v_2. La idea es construir una matriz "desinflada" A' a partir de A, v_1 y λ_1, y aplicar el método de la potencia a A' para obtener λ_2 y v_2 [Watkins, 2004].

En particular, si consideramos $A' = A - \lambda_1 v_1 v_1^T$ vemos que A' tiene autovalores $0, \lambda_2, \ldots, \lambda_n$ y autovectores asociados $v_1, v_2, \ldots, v_n$ porque

$$(A - \lambda_1 v_1 v_1^T)v_1 = Av_1 - \lambda_1 v_1(v_1^T v_1) = \lambda_1 v_1 - \lambda_1 v_1 = 0v_1$$

y

$$(A - \lambda_1 v_1 v_1^T)v_i = Av_i - \lambda_1 v_1(v_1^T v_i) = \lambda_i v_i$$

El autovalor dominante de A' es, entonces, λ_2 y su autovector asociado es v_2. Por lo tanto, podemos aplicar el método de la potencia descrito más arriba a A' para obtener λ_2 y v_2. Aplicando este método iterativamente, obtenemos todos los autovalores y autovectores de A, siempre que se cumpla que $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$ y que los autovectores conformen una base ortonormal, lo cual es cierto si A es simétrica.

Los autovalores y autovectores se pueden usar para caracterizar redes a partir de la matriz de adyacencia A. Por ejemplo, podemos usar el autovector dominante de A para medir la centralidad de cada nodo de una red. Una medida de centralidad cuantifica la importancia de un vértice o nodo en una red. Intuitivamente, cuantos más vértices estén conectados a un nodo dado, mayor será su centralidad. Una propiedad adicional deseable es que la centralidad no dependa únicamente de las aristas directas hacia un nodo, sino que también dependa, de manera recursiva, de la centralidad de los nodos que apuntan hacia él [Zaki and Meira, 2014].

A partir de esta intuición de que la centralidad de un nodo depende de la centralidad de otros nodos que apuntan a él, podemos definir a c_i, la centralidad del vértice i, como:

$$c_i = \sum_{j=1}^{n} A_{ij} c_j \tag{4}$$

donde $A \in \mathbb{R}^{n \times n}$ es la matriz de adyacencia simétrica. Nos restringimos a que c_i sea un número real positivo [Zaki and Meira, 2014]. Para todos los vértices en simultáneo, podemos expresar matricialmente y de manera recursiva la expresión 4:

$$c' = Ac$$

donde c es un vector de n valores que indican la centralidad de cada vértice. Si comenzamos con un vector de centralidad inicial $c_{(0)}$, podemos usar esta ecuación para obtener un vector actualizado de manera iterativa: si $c_{(k-1)}$ es el vector de centralidad en la iteración $k-1$, entonces el vector en la iteración k se obtiene como:

$$\begin{aligned} c_{(k)} &= Ac_{(k-1)} \\ &= A(Ac_{(k-2)}) = A^2 c_{(k-2)} \\ &= A^2(Ac_{(k-3)}) = A^3 c_{(k-3)} = \ldots \\ c_{(k)} &= A^k c_{(0)} \end{aligned} \tag{5}$$

Como hemos visto en la ecuación 1, el vector $c_{(k)}$ converge al autovector dominante de A a medida que aumenta k. El autovector dominante nos da, entonces, una medida de centralidad de los nodos de la red. Este autovector se puede obtener aplicando el método de la potencia.

Otra manera de caracterizar una red es midiendo su conectividad. Para esto, podemos usar la matriz Laplaciana de la matriz de adyacencia A que se define como $L = D - A$, donde D es una matriz diagonal cuyos valores d_{ii} indican la cantidad de aristas asociadas al vértice i. Se puede demostrar que para matrices de adyacencia simétricas, L es simétrica y semidefinida positiva con todos autovalores iguales o mayores que cero, que la suma de sus filas y de sus columnas es cero y, considerando la ecuación de autovalores $Lv = \lambda v$, podemos ver que el autovalor $\lambda = 0$ es un autovalor de L y que su autovector asociado es un vector de unos [Dutta et al., 2022].

El autovalor más pequeño distinto de cero de L se conoce como "conectividad algebraica" y su autovector asociado es el "vector de Fiedler", v_F. La magnitud de la conectividad algebraica refleja el grado de conectividad de la red. Por otra parte, el vector de Fiedler puede usarse para particionar una red en dos componentes densamente conectados. Esto se puede lograr separando los vértices i en dos grupos: los que toman valores positivos $v_{Fi} > 0$ y los que toman valores negativos $v_{Fi} < 0$ [Dutta et al., 2022].

Por último, introducimos la técnica de Análisis de Componentes Principales (PCA), la cual usamos para reducir la dimensión de una matriz de atributos. Dada una matriz de datos con n observaciones y p atributos, $X \in \mathbb{R}^{n \times p}$, PCA busca una base r-dimensional que maximice la varianza de los datos. La dirección con la mayor varianza proyectada se denomina primera componente principal. La dirección ortogonal que captura la segunda mayor varianza proyectada se denomina segunda componente principal, y así sucesivamente.

Para hallar estas direcciones podemos partir de la matriz de covarianza de X, la cual tiene la siguiente forma:

$$\Sigma = \frac{1}{n-1} X_c^T X_c$$

donde X_c es la matriz de datos centrados, es decir, $X_c = X - \mu$, siendo μ es el vector de medias de los atributos. El valor Σ_{ij} es la covarianza entre los atributos i y j, y por lo tanto, el valor en la diagonal Σ_{ii} es la varianza del atributo i. Es decir, Σ es una matriz simétrica (y semidefinida positiva, ver Bishop and Nasrabadi, 2006). Al tratarse de una matriz con estas características, podemos diagonalizar la matriz de covarianza Σ de los datos X para obtener una base de autovectores V y una matriz diagonal de autovalores D tales que $\Sigma = VDV^T$. Asimismo, los autovectores forman una base ortonormal y los autovalores son todos no negativos y pueden ordenarse de manera decreciente tal que $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$.

Las columnas de V son los autovectores de Σ y las componentes principales de los datos X. Para hallar una proyección de dimensión k que maximice la varianza proyectada de los datos definimos una matriz $W = V_k \in \mathbb{R}^{p \times k}$, donde V_k es la matriz que contiene los k autovectores asociados a los k autovalores más grandes de Σ. La proyección de los datos centrados X_c en la matriz W se obtiene como $Z = X_c W \in \mathbb{R}^{n \times k}$. Por medio de esta operación, obtenemos una matriz de datos Z de dimensión reducida $k \leq p$ que captura la mayor varianza posible de los datos originales X.

1.1 Objetivos del trabajo

Los objetivos del presente trabajo son:

1. Implementar el método de la potencia con deflación para hallar autovectores y autovalores, analizando sus resultados y su convergencia en casos específicos de interés.
2. Ilustrar el uso de los autovectores y autovalores para caracterizar redes en las siguientes aplicaciones:
   (a) Medición de la centralidad de nodos.
   (b) Medición de la conectividad de una red mediante el cálculo de los autovectores de la matriz Laplaciana.
   (c) Comparación de redes midiendo la asociación entre su autovalores.
   (d) Reducción de la dimensionalidad de una matriz de atributos con PCA.

Para responder el segundo objetivo usamos los conjuntos de datos del Club de Karate [Zachary, 1977] y de Facebook [Leskovec and Mcauley, 2012]

La organización del trabajo se estructura de la siguiente manera: en la sección 2 describimos la implementación del método de la potencia con deflación y analizamos su desempeño (sección 2.1), y presentamos los

experimentos de aplicación de autovectores y autovalores en redes (sección 2.2). En la sección 3 presentamos los resultados de los experimentos. Finalmente, concluimos el trabajo en la sección 4.

2 Desarrollo

En la primera subsección presentamos el pseudocódigo del algoritmo del método de la potencia con deflación. Evaluamos la implementación en matrices donde los autovectores y autovalores son conocidos de antemano, y analizamos la convergencia en casos con autovalores repetidos y autovalores cercanos. En la segunda subsección presentamos los experimentos que ilustran la relevancia del cálculo de autovalores y autovectores para analizar redes tomando como caso el Club de Karate y la red de Facebook de un usuario.

2.1 Método de la potencia con deflación

El pseudocódigo para el algoritmo que implementa el método de la potencia se define por las siguientes funciones:

Algoritmo 1 Método de la potencia para hallar el autovector dominante v de A, dada una cantidad máxima de iteraciones ($niter$) y un valor para determinar la convergencia (ϵ).

function METODOPOTENCIA(A, $niter$, ϵ)
  $numRows \leftarrow$ # de filas de A
  $v \leftarrow$ Vector aleatorio de tamaño $numRows$
  for $i \leftarrow 0$ to $niter - 1$ **do**
    $v\_previo \leftarrow v$
    $v \leftarrow A \cdot v$
    $v \leftarrow \frac{v}{\|v\|}$
    if $\left|v^T \cdot v\_previo - 1.0\right| < \epsilon$ **then**
      break
    end if
  end for
  return v
end function

Algoritmo 2 Función auxiliar para obtener el autovalor dominante λ de A asociado al autovector dominante v.

function AUTOVALORDOMINANTE(A, v)
  $scale \leftarrow \|v\|^2$
  $\lambda \leftarrow v^T \cdot A \cdot v$
  $\lambda \leftarrow \frac{\lambda}{scale}$
  return λ
end function

Inicialmente implementamos el método de la potencia para obtener el autovector dominante de una matriz cuadrada A siguiendo la sucesión planteada en la ecuación 2 (algoritmo 1). Para hallar el autovalor dominante dado el autovector dominante, usamos una función auxiliar que implementa la expresión 3 (algoritmo 2).

Luego implementamos el método de potencia con deflación (algoritmo 3), el cual recibe como entradas una matriz A, los parámetros del método de la potencia, $niter$ y ϵ, y la cantidad de autovalores a encontrar, num. La salida de esta función es una matriz $R \in \mathbb{R}^{(n+1)\times n}$ con los n autovectores de A en las primeras n filas, y los autovalores asociados a cada uno, en la última fila, es decir:

$$R = \begin{bmatrix} & v_1^T & \\ & v_2^T & \\ & \vdots & \\ & v_n^T & \\ \lambda_1 & \lambda_2 \quad \cdots \quad \lambda_{n-1} & \lambda_n \end{bmatrix}$$

Estas funciones fueron implementadas en C++. Compilamos un programa con el compilador GCC 4.9 que recibe como argumento un archivo de texto con los datos de la matriz A, la cantidad de iteraciones ($niter$) y

Algoritmo 3 Método de la potencia con deflación para hallar todos los autovalores y sus autovectores asociados.

function POTENCIACONDEFLACION(A, num, $niter$, eps)
  $numRows \leftarrow$ # de filas de A
  $R \leftarrow$ Matriz de tamaño $(numRows + 1) \times numRows$ ▷ Inicializa matriz de rdos.
  $M \leftarrow A$ ▷ Inicializa matriz desinflada
  $a \leftarrow$ Vector de tamaño num ▷ Vector de autovalores
  for $i \leftarrow 0$ to $num - 1$ **do**
    $v \leftarrow$ MetodoPotencia$(M, niter, eps)$
    $\lambda \leftarrow$ AutovalorDominante(M, v)
    $R[i, :] \leftarrow v^T$ ▷ Asigna fila i-ésima
    $a[i] \leftarrow \lambda$ ▷ Asigna elemento i-ésimo
    $M \leftarrow M - \lambda \cdot (v \cdot v^T)$ ▷ Desinfla matriz
  end for
  $R[numRows, :] \leftarrow a^T$ ▷ Asigna última fila
  return R
end function

un valor de tolerancia para la convergencia (ϵ). La salida son dos archivos de texto, uno con los autovalores y otro con los autovectores como columnas de una matriz. Para manipular matrices usamos la librería Eigen 3.4.0 [1].

Verificamos la implementación del método de la potencia en casos donde los autovalores y autovectores son conocidos. Para esto, armamos una matriz M semejante a una matriz diagonal D con los autovalores en la diagonal y los autovectores Q y Q^T multiplicando a izquierda y derecha tal que:

$$M = Q^T \begin{pmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \lambda_n \end{pmatrix} Q$$

con $Q = I - 2vv^T$ y $||v||_2 = 1$, siendo Q la matriz de reflexión (ortogonal) que hace las veces de matriz de autovectores. Además, comparamos estos autovectores y autovalores conocidos con los calculados usando el método $linalg.eig$ de la biblioteca NumPy de Python, especializada en cálculo numérico eficiente.

En una primera exploración verificamos que se cumpla que Mv_i se aproxime a Ml_i para $i = 1, \ldots, n$, donde l_i y v_i son los autovalores y autovectores calculados con el algoritmo 3, respectivamente. En distintas matrices M predeterminadas observamos que el método converge bien. Respecto de NumPy, observamos que nuestro método devuelve los mismos autovalores y autovectores, pero en algunas ocasiones no coincide en signo, es decir, devuelve otros sentidos de la misma dirección.

Más allá de esta primera verificación, analizamos la convergencia de nuestro método para casos de matrices con todos autovalores distintos. En particular, buscamos analizar el error en el cálculo de autovalores para distintos números de iteración y en comparación con NumPy.

Planteamos un experimento que consiste en fijar el valor de convergencia en $\epsilon = 0$ para aislar el impacto del número de iteraciones en el cálculo de los autovalores y autovectores. Luego, para los valores $niter = \{1, 10, 100, 1000, 2000, 5000, 10000\}$ calculamos los autovalores y autovectores de una matriz $M \in \mathbb{R}^{100 \times 100}$. Para cada uno de los 100 autovalores estimados para cada $niter$ y para NumPy, calculamos el valor absoluto del error relativo entre el autovalor real (l_{true}) y el calculado (l_{est}): $|(l_{est} - l_{true})/l_{true}|$.

Observamos que en nuestro método disminuye el error a medida que aumenta la cantidad de iteraciones, y que esto encuentra un límite en el orden de las 10^3 iteraciones (Figura 1). NumPy tiene consistentemente menor error respecto de nuestos mejores casos.

Se analizó el caso particular de matrices con pares de autovalores muy parecidos entre sí, así como también el caso de matrices con autovalores repetidos. Para ello, se generaron al azar 100 matrices de tamaño 10×10 de autovalores y autovectores conocidos, y a partir de cada una se generaron matrices para las que se alteró sólo a λ_2, el segundo autovalor en orden de dominancia. Para matrices con autovalores parecidos se eligió $\lambda_2 = \lambda_1 - 10^{-10}$, y para matrices con autovalores repetidos se tomó $\lambda_2 = \lambda_1$. Para cada matriz se midió el

[1] `https://eigen.tuxfamily.org/`

error de la estimación de autovalores y autovectores como $\frac{\sum_{i=1}^{10}(Av_1-\lambda_1 v_1)_i^2}{10}$, es decir, como el error cuadratico medio (ECM) entre Av_1 y $\lambda_1 v_1$.

En ambos casos se esperaba encontrar que la convergencia del método sea más lenta para estos tipos de matrices, es decir, que para una misma cantidad de iteraciones del método el error en la estimación sea mayor respecto a matrices con autovalores similares pero sin autovalores parecidos ni repetidos. Obtuvimos resultados contrarios a los esperados (figura 2), y al momento de escribir este informe no tenemos claro por qué.

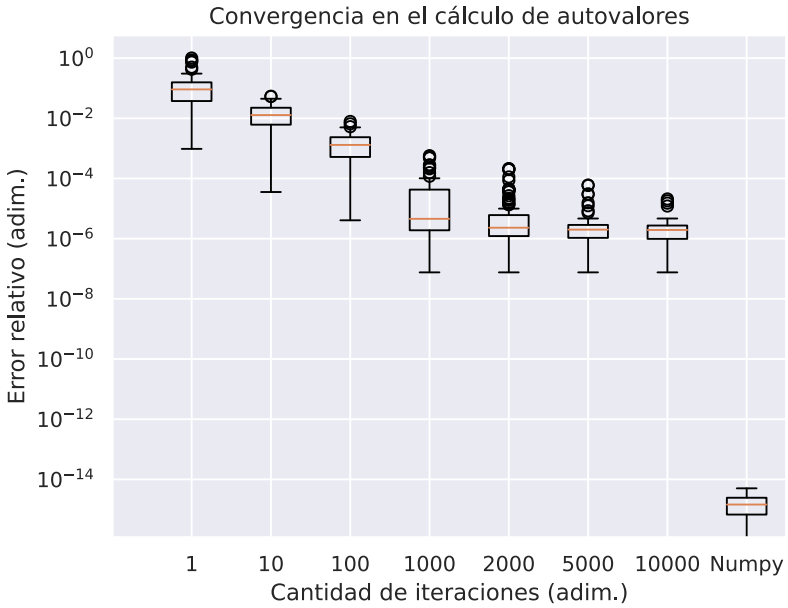

Figura 1: Error relativo de estimación de autovalores en función de la cantidad de iteraciones del algoritmo 3, y de NumPy. Para cada condición se grafica, en forma de boxplot, la distribución de errores relativos (en módulo) entre los autovalores reales y los estimados.

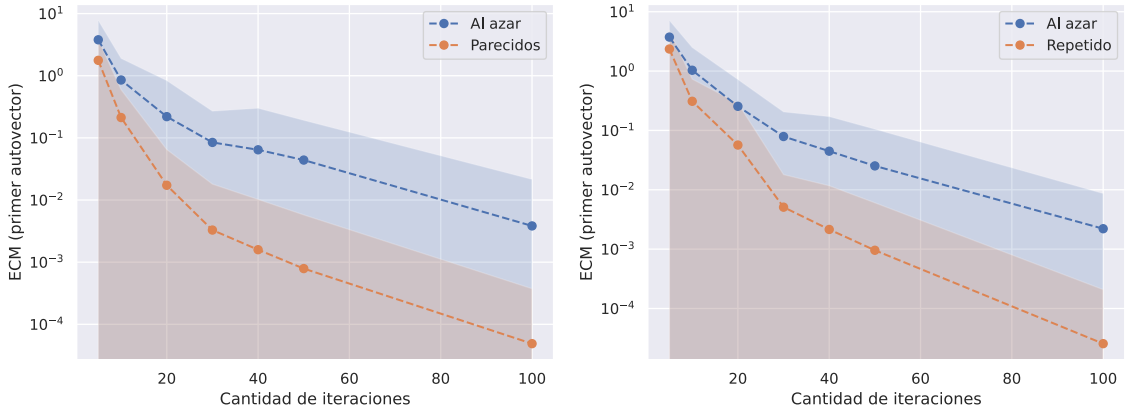

Figura 2: Convergencia de la estimación del primer autovalor y autovector, medida como el error cuadrático medio entre Av_1 y $\lambda_1 v_1$ en función de la cantidad de iteraciones del método. Cada punto representa el error promedio en la estimación para 100 matrices distintas, todas de tamaño 10×10. En azul: metrices con autovalores elegidos al azar. En naranja: matrices modificadas para que sus dos autovalores más dominantes sean muy parecidos entre sí (izquierda) o iguales (derecha), sin modificar al resto de los autovalores.

2.2 Aplicaciones en redes

Nos proponemos ilustrar el uso del cálculo de autovalores y autovectores para caracterizar redes a partir de un conjunto de experimentos en dos conjuntos de datos: el "club de Karate" y "Facebook".

2.2.1 Club de Karate

El "Club de Karate" es una red social documentada por Zachary [1977], que resume los vínculos entre 34 miembros de un club de karate, en el cual se documentaron 78 aristas no dirigidas según las interacciones regulares entre los miembros fuera del club. Estas interacciones se pueden representar con una matriz de adyacencia $A \in \mathbb{R}^{78\times 78}$, como explicamos en la sección 1.

Durante el estudio ocurrió un conflicto que dividió al club en dos: la mitad de los miembros formaron un nuevo club mientras los miembros de la otra parte encontraron un nuevo instructor o abandonaron el karate. Por lo tanto, cada nodo está etiquetado según la pertencencia a cada uno de estos dos grupos post-conflicto.

Sobre este dataset hacemos los siguientes experimentos:

1. Identificamos los nodos más centrales computando la centralidad de autovector con el algoritmo 3. Para esto, computamos el autovector dominante de la matriz de adyacencia A como se describe en la sección 1 (ecuación 5).
2. Buscamos el autovector de la matriz Laplaciana que mejor predice la pertenencia a los grupos post-conflicto. Para esto, medimos el valor absoluto de la correlación de Pearson entre cada autovector y el vector binario que indica la pertenencia a los grupos post-conflicto. Para encontrar los autovectores de la matriz Laplaciana usamos el algoritmo 3 y comparamos con los resultados que se obtienen con la librería NumPy para Python [2].

2.2.2 Facebook

El set de datos de usuarios de Facebook de Leskovec and Mcauley [2012] contiene una matriz de atributos $X \in \mathbb{R}^{n\times p}$ conformada por $n = 792$ usuarios (filas) descritos por $p = 319$ atributos binarios (columnas) que indican si el usuario tiene o no ciertas características. Para estos usuarios se cuenta también con una red no dirigida de amistades que se puede representar mediante una matriz de adyacencia simétrica $A \in \mathbb{R}^{792\times 792}$, donde $A_{ij} = 1$ si el usuario i es amigo del usuario j, y $A_{ij} = 0$ en caso contrario.

A partir de la matriz de atributos computamos un matriz de similitud $S \in \mathbb{R}^{792\times 792}$ donde S_{ij} representa la similitud entre los usuarios i y j definida como el número de atributos en los que ambos usuarios tienen valor 1. Es decir, S_{ij} se puede computar como el producto interno entre las filas i y j de la matriz de atributos X. Por lo tanto, S se puede obtener como $S = XX^T$.

Si fijamos un umbral de similitud u, podemos construir un grafo no dirigido donde los nodos son los usuarios y las aristas son las conexiones entre usuarios que tienen una similitud mayor o igual a u. Para este grafo obtenemos la matriz de adyacencia A_S. Por otra parte, es posible reducir la dimensionalidad de X a k dimensiones mediante PCA (ver sección 1) antes de obtener S y A_S. Llamamos a la matriz de adyacencia que obtenemos mediante este método A_S^k.

Llevamos a cabo los siguientes experimentos:

1. Buscamos el valor del umbral u que optimiza la similitud entre la red de amistades (representada por A) y la red construida a partir de los atributos (representada por A_S).

   Medimos la similitud entre redes para cada umbral usando dos métricas: (1) la correlación de Pearson de las matrices de adyacencia "estiradas" en un vector, y (2) la correlación de Pearson entre los vectores de autovalores ordenados de mayor a menor. Para obtener los autovalores usamos el algoritmo 3 y validamos los resultados con la librería NumPy.
2. Analizamos cómo la proyección de X a sus k primeras componentes principales afecta a la búsqueda del umbral u que optimiza las métricas de correlación. En particular, para distintas cantidades de componentes principales k examinamos cómo varía la correlación en función de u.

[2] `https://NumPy.org/`

Realizamos todos los análisis y visualizaciones usando Python. Para graficar las redes usamos la biblioteca networkx [3].

3 Resultados

En esta sección presentamos los resultados de los análisis propuestos en la sección 2.2.

3.1 Club de Karate

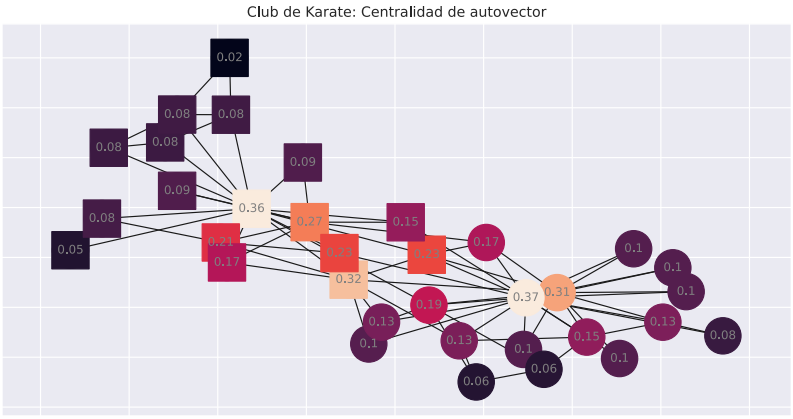

Figura 3: Centralidad de autovector del Club de Karate. Los valores están normalizados de manera que el autovector es unitario.

Visualizamos la centralidad de autovector de cada nodo de la red de Karate, normalizando el vector con la norma-2 (Figura 3). Identificamos que los nodos más centrales son los asociados a las coordenadas del vector de valor 0.36 y 0.37, siendo el primero perteneciente al grupo de etiqueta 0 y el segundo al grupo de etiqueta 1. De acuerdo a esta definición de centralidad, un nodo es importante cuando está conectado a muchos nodos que a su vez son importantes.

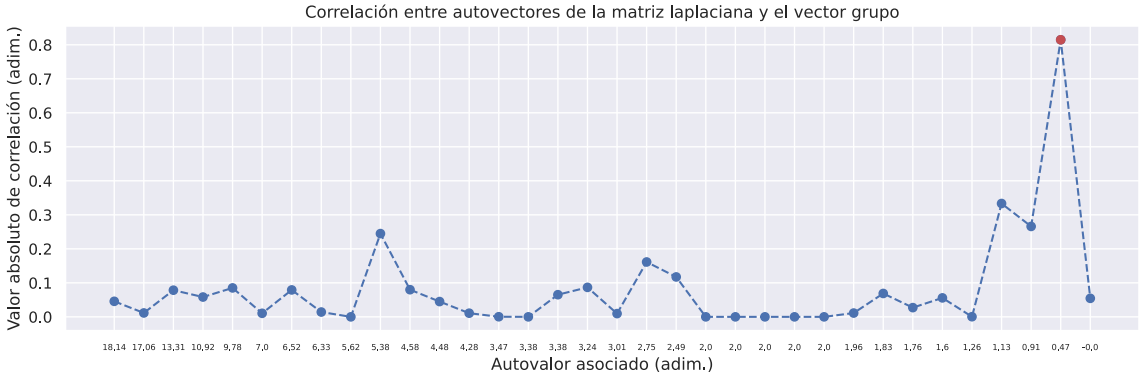

Figura 4: Predicción de grupos después del conflicto. El eje vertical representa el valor absoluto de la correlación entre cada autovector de la matriz Laplaciana y el vector binario de grupos post-conflicto. Cada autovector está identificado en el eje horizontal por su respectivo autovalor. Se encuentra resaltado el punto correspondiente al autovalor asociado al autovector que produce la correlación máxima.

Encontramos que el autovector de la matriz Laplaciana que mejor predice la separación de grupos es el asociado al menor de los autovalores no negativos, que es el segundo con los autovalores ordenados de menor

[3] `https://networkx.org/`

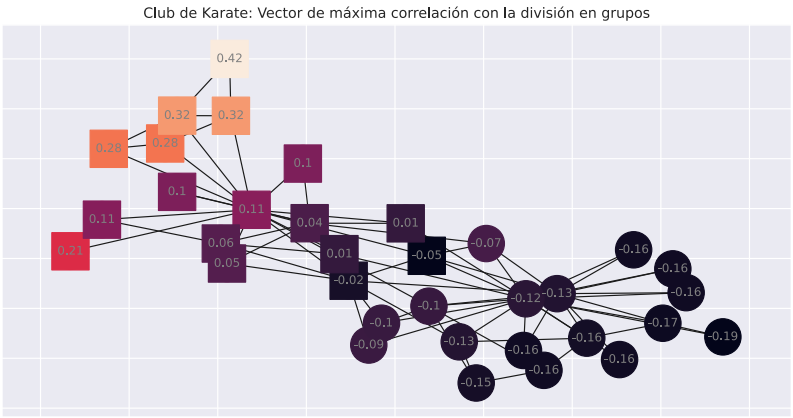

Figura 5: Vector de Fiedler del Club de Karate. Este vector es un buen predictor de la pertenencia a los grupos post-conflicto. Los grupos se indican por la forma de los nodos, siendo los nodos cuadrados pertenecientes al grupo de etiqueta 0 y los circulares al grupo de etiqueta 1.

a mayor (Figura 4). A este autovector se le denomina "vector de Fiedler", y a su autovalor asociado se le conoce como conectividad algebraica (ver sección 1). Al visualizar el valor asociado a cada nodo en el vector de Fiedler, observamos que los nodos pertenecientes al grupo 0 tienden a estar asociados a coordenadas del autovector de valor positivo mientras que los del grupo 1, al contrario, tienden a tener valores negativos. Es decir, este vector es un buen predictor de la pertenencia a los grupos (ver Figura 5, en la que los nodos del grupo 0 se representan como cuadrados y los del grupo 1 como círculos).

Verificamos que los resultados computados con el algoritmo 3 son similares a los que se obtienen con la libraría NumPy tanto para los autovalores como los autovectores. En particular, observamos diferencias absolutas máximas en los autovalores y autovectores en el orden de 3×10^{-6}.

3.2 Facebook

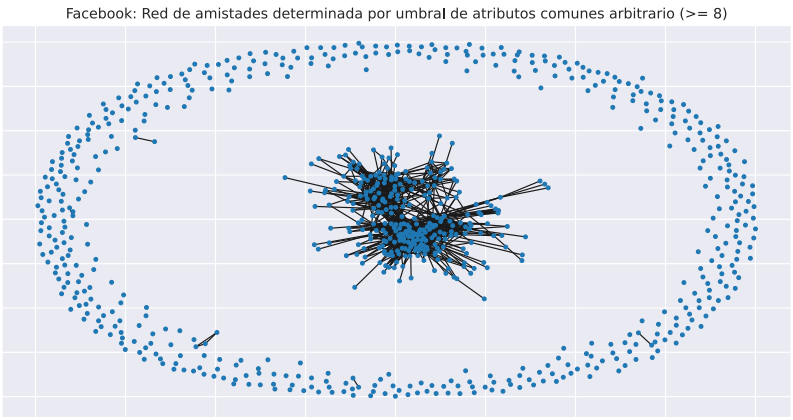

Figura 6: Visualización de la red de Facebook determinada por un umbral de atributos comunes entre los usuarios igual a 8. Los aristas entre vértices representan pares de usuarios que superan ese umbral.

Inicialmente, computamos la matriz de similitud $S = XX^T$ y obtenemos una matriz de adyacencia asociada A_s a partir de un umbral fijado arbitrariamente $u = 8$ (ver Figura 6).

Para este umbral obtenemos una correlación de las matrices de adyacencia aplanadas ≈ 0.069 y una correlación de listas de de autovalores ≈ 0.935. Al calcular este coeficiente de correlación con los autovalores que se obtienen con la librería NumPy obtenemos un valor muy cercano ≈ 0.926.

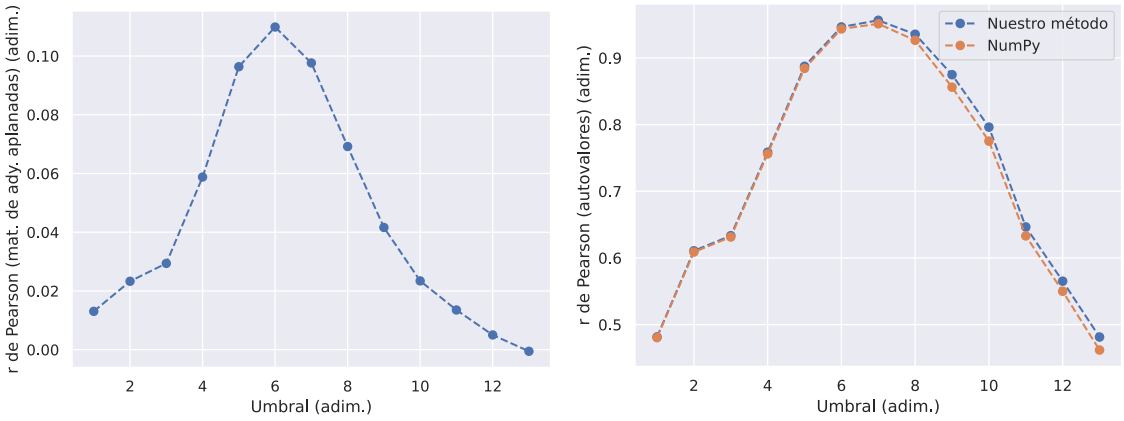

Figura 7: Similitud de redes para distintos valores del umbral de atributos en común u. La similitud se mide como la correlación de Pearson de las matrices de adyacencia "estiradas" en un vector (izquierda) y la correlación de Pearson entre los vectores de autovalores ordenados de mayor a menor (derecha).

El umbral u que maximiza la similitud de la red de amistades con la red generada a partir de los atributos es similar para las dos métricas propuestas (ver Figura 7). Observamos también que, si bien las magnitudes de la correlación difieren entre una métrica y la otra, el patrón es el mismo: para valores bajos y altos del umbral la similitud es relativamente baja, mientras que tiende a crecer y maximizarse en valores intermedios, $u = 5$ para la correlación entre matrices aplanadas ($r \approx 0.110$) y $u = 7$ para la correlación entre listas de autovalores ($r \approx 0.956$). No se pusieron a prueba valores de u mayores a 13 pues se encontró que ningún par de usuarios compartía más de 13 atributos. Estos resultados sugieren que la cantidad de atributos compartidos entre usuarios podría explicar una parte no despreciable de las amistades entabladas en la red.

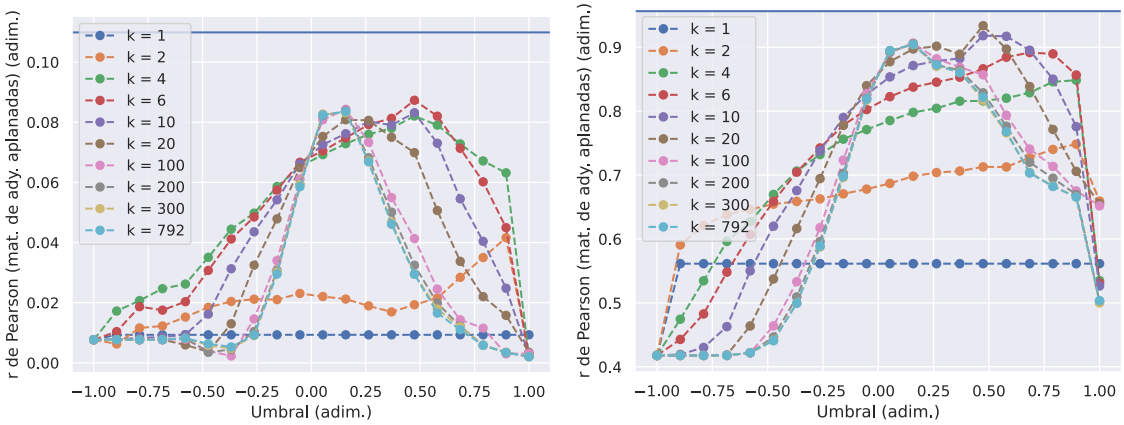

Figura 8: Variación de la similitud entre redes en función de u para distintos valores de k, la cantidad de primeras componentes principales usadas para la proyección de X. Similitud medida como correlación entre matrices de adyacencia aplanadas (izquierda) y como correlación entre listas de autovalores (derecha). En ambos casos, las rectas azules horizontales representan la correlación máxima obtenida en el experimento anterior (sin PCA).

Finalmente, realizamos el mismo análisis pero partiendo de una matriz de atributos X proyectada a las primeras k componentes principales obtenidas por PCA, para distintos valores de k. Como explicamos en la sección 1, para obtener las componentes principales de X calculamos la matriz de atributos centrada X_c , luego la matriz de covarianzas como $\Sigma = \frac{X_c^T X_c}{791}$, calculamos los autovectores de Σ y los ordenamos según el valor en módulo de sus autovalores asociados, llegando así a la matriz de autovectores V. Luego, para cada valor de k se computó la proyección a las primeras k componentes principales como $Z = X_c V_k$, con V_k la matriz compuesta por las primeras k columnas de V en orden. Para poder usar un mismo rango de umbrales para todo k, normalizamos Z al rango $[-1; 1]$ diviendo coordenada a coordenada por el valor correspondiente de

la matriz vv^T, con v el vector de normas de las columnas de Z. Es decir, usamos la similitud coseno como medida de similitud entre usuarios.

Barrimos valores de umbral u uniformemente distribuidos entre -1 y 1 y para cada uno generamos una matriz de adyacencia a partir de la matriz Z normalizada. Para cada combinación de k y u, comparamos a la matriz de adyacencia generada con la real por los dos métodos descriptos anteriormente.

Los resultados se resumen en la Figura 8. Independientemente de la métrica y de la combinación entre k y u, fue imposible alcanzar la correlación máxima obtenida en el experimento anterior (sin PCA), lo que para el caso de $k = 792$ (en el que esperabamos la misma correlación máxima que en el caso sin PCA) podría deberse al error numérico introducido por el método de la potencia. Puede verse que para valores de k pequeños (en los ordenes de 10^0 y 10^1) se obtienen curvas con máximos que representan una parte significativa de la correlación máxima sin PCA. Esto implica que, como se esperaba, las primeras componentes principales de la matriz de atributos contienen la mayor parte de la información relevante para la predicción de amistades en la red.

Este mismo experimento fue realizado de manera preliminar para más valores de k que los mostrados en la figura 8, calculando los autovalores de Z usando NumPy por restricciones de tiempo. No encontramos un valor de k que resulte en correlaciones notoriamente mejores a las alcanzadas para los k mostrados.

4 Conclusiones

En este informe implementamos el método de la potencia con deflación para hallar autovectores y autovalores. Analizamos la convergencia de este método en los casos particulares donde la matriz puede tener autovalores repetidos o muy cercanos. A partir de esta implementación, realizamos una serie de experimentos en dos conjuntos de datos: el "club de Karate" y una red de usuarios de Facebook.

En el caso del club de Karate, usamos el autovector dominante para medir la centralidad de los nodos en la red y analizamos la correlación entre los autovectores de la matriz Laplaciana y la pertenencia a los grupos post-conflicto. Observamos que el autovector de Fiedler que surge de la matriz Laplaciana tiene una correlación alta con la estructura comunitaria de esta red, caracterizada por la presencia de dos bandos.

En el caso de Facebook, usamos la correlación de autovalores como una manera de medir la similitud entre dos matrices de adyacencia. En particular, construimos una matriz de adyacencia no pesada a partir de una cantidad mínima de atributos que tienen en común los usuarios, y la comparamos con la matriz que surge de la red de amistades en la red social. Usamos la correlación de Pearson entre los vectores de autovalores para medir la similitud y comparamos esta métrica con la correlación entre las matrices de adyacencia "aplanadas". Exploramos el impacto del umbral de atributos en común en la similitud de las redes. Por último, estudiamos el impacto de aplicar PCA en la matriz de atributos en la relación entre el umbral y la similitud.

En conjunto, este trabajo muestra que el cálculo de autovectores y autovalores puede proporcionar información valiosa para entender la estructura de redes, en particular, en el caso de redes sociales.

Referencias

Albert-László Barabási. Network science. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1987):20120375, 2013.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Sagnik Dutta, Nilava Metya, and Sagnik Mukherjee. Fiedler vector method. `https://nilavam.github.io/FV/report.pdf`, 2022.

Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego networks. *Advances in neural information processing systems*, 25, 2012.

Gilbert Strang. *Introduction to linear algebra*. SIAM, 2022.

David S Watkins. *Fundamentals of matrix computations*. John Wiley & Sons, 2004.

Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.

Mohammed J Zaki and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.