Vrije Universiteit Brussel

Faculty of Science
Master of Science in Mathematics - profile: Education

Similarity on Graphs & Hypergraphs

Graduation thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Mathematics - profile: Education

Filip Moons

Promotor: Prof. Dr. P. Cara

August 2015

Acknowledgements

First of all, I would like to thank my promotor, Philippe Cara for his incredible attention to detail, for the patience he had with me and the many reminders to continue and to work hard. He also showed me the way to this beautiful topic for a Master thesis. Besides his role as promotor, I will never forget him as professor of the Linear Algebra course in the very first term of the BSc in Mathematics, that without any doubt was the most memorable course during my college years: the hard study and the stress I had before entering the exam room are memories that I still will be telling in my old days.

Secondly, I would like to thank Roger Van Nieuwenhuyze, my math lecturer during the teacher training I followed for one year at the Hogeschool-Universiteit Brussel. Without his encouragements, I would never have dared to leave and study Mathematics at university level.

I also thank all professors and teaching assistant who have learned me a lot during the past five years. A special thanks goes to Reen Tallon, who always managed to overcome any administrative burden caused by the combination of study programs.

Last but not least, I want to thank my parents for their angry looks when they felt I did not study enough, it certainly kept me on track. Also my partner Sacha and my friends from university and the Classical Liberal Student Union made my college years unforgettable for the rest of my life.

Abstract

This thesis is about similarity on graphs and hypergraphs. In the first chapter, we give an overview of all the preliminary knowledge needed to understand everything in the following chapters. All the methods of similarity we will discuss, are eventually solving an eigenvalue problem. So it's no surprise we prove the Perron-Frobenius theorem and introduce the power method in the first chapter, the power method is a numerical algorithm to calculate the largest eigenvalue and a corresponding eigenvector.

In chapter 2, we give a complete overview of the research done on similarity of graphs. The notion of similarity originated from the HITS-algorithm: this algorithm was developed in the late nineties to sort results of a search engine in a meaningful way. The algorithm uses the hyperlink structure of a set of webpages to assign to each webpage in the set a score in how well they are in referring to other pages in the set (the *hub*-score) and it assigns to each webpage a score in how authoritative they are compared to the other sites in the set (the *authority*-score). So you get two groups in the set of webpages: some pages are good hubs, others are good authoirities. Obviously, this dichotomy is strongly related: pages that are good hubs will contain a lot of links to authoritative pages, and, conversely, pages are good authoirites when they have a lot of incoming links of good hubs. This algorithm was very innovative at the time, because search engine results where then sorted based only on the number of incoming links or on the number occurencies of the search query.

The HITS-algorithm is generalized in 2005 to a notion of similarity between graph. A set containing hyperlinked web pages, can indeed be seen as a graph. This generalization wil lead to a method where we can compare two graphs by putting a similarity score between every node of the first graph and every node of the second graph. This similarity score indicates how similar these nodes are: two nodes will have a high similarity score when their adjacent nodes will also have a high similarity score.

Once this node similarity is introduced on graphs, we extend this notion to a similarity method that also returns a similarity score between the edges of two graphs. We will also discuss a similarity method that deals with colored graphs. With all these methods, we have a complete overview of the most recent studies about similiarity on graphs. It's is remarkable that the convergence of the resulting algorithm of all these methods can be proved with the same convergence theorem. Every solution is in fact also just an eigenvalue problem: every extension is a variant of the power method to calculate eigenvectors.

An important remark is that all the methods return slightly different results. At the moment, 'the' similarity score between two vertices of a graph doesn't exists, simply because it depends on the method you are using: in some cases the results are equal (up to a constant), in other cases the results are different because sligtlhy different criteria are used in the calculation. In that case, we can explicitely derive the difference. Still, it is not that worse: it's important to remember that the similarity scores are mainly interesting when compared to

other similarity scores within a graph and with one method. From that point of view, every method give the same ranking. This is also justified by the applications: all of them are using similarity scores in comparison to other similarity scores.

In the last chapter, we discover a new field: we try to transfer the concept of similarity to the more general structure of hyprgraphs. We end this chapter with a reflection on the results of the previous chapter: when we try to develop a method for similarity on hypergraphs, which conditions must be fulfilled? Are these conditions also met by the graph similarity methods of the first chapter? At first, we take a look at the graph representations of hypergraphs and we try to get a good method of similarity by using them. It will be clear that the incidence graph of a hypergraph returns the best results when used for similarity. After that, we also try to develop a method based on the incidence matrix. We conclude the thesis by proving that both methods are equal up to a constant.

Samenvatting

Deze thesis bespreekt similariteit op graffen en hypergraffen. In het eerste hoofdstuk geven we een overzicht van alle voorkennis die nodig is om deze thesis te doorgronden. Bijna alle methodes die in deze thesis besproken worden, komen uiteindelijk neer op het oplossen van een eigenwaardeprobleem: het is dan ook niet verwonderlijk dat we de Perron-Frobeniusstelling bewijzen en de methode van de machten invoeren om eigenvectoren numeriek te berekenen.

De notie van similariteit op graffen, ingevoerd in hoofdstuk 2, vindt zijn oorsprong in het HITS-algoritme: dit algoritme werd eind jaren negentig ontdekt om zoekresultaten van een internetzoekmachine op een betekenisvolle manier te sorteren. Het algoritme onderzoekt de relaties tussen de hyperlinks van een verzameling webpagina's: sommige webpagina's worden bestempeld als goede doorverwijzers, andere webpagina's worden dan weer als een gezaghebbende bron beschouwd voor de zoekterm. Uiteraard is deze tweedeling sterk verwant: pagina's die goede doorverwijzers zijn zullen veel links naar pagina's bevatten die als een gezaghebbende bron gezien worden en, omgekeerd, zullen pagina's als gezaghebbende bron worden beschouwd als ze veel inkomende links krijgen van goede doorverwijzers. Dit algoritme was erg vernieuwend in die tijd, vermits tot dan toe zoekresultaten enkel gesorteerd werden op het aantal inkomende links (zonder daarbij de bronpagina na te gaan) of het aantal keren dat de zoekterm op de pagina voorkwam.

Deze methode werd rond 2005 veralgemeend naar een notie van similariteit tussen graffen. Een verzameling webpagina's die onderling gelinkt zijn zoals die in het HITS-algoritme gebruikt worden, kan immers beschouwd worden als een graf. Deze veralgemening leidt tot een methode waarbij we een soort score kunnen kleven op hoe twee toppen van twee verschillende graffen op elkaar lijken. Deze 'similariteisscore' zal hoger zijn als de adjacente toppen van deze twee toppen ook gelijkaardig zijn.

Eens we similariteit op de toppen van graffen hebben ingevoerd, gaan we deze methode verder uitbreiden tot similariteit op toppen en bogen van graffen en tot similariteit tussen gekleurde graffen. We hebben zo een compleet overzicht van de meest recente studies over het onderwerp. Opvallend is dat de convergentie van de resulterende algoritmes van al deze uitbreidingen via dezelfde convergentiestelling kunnen bewezen worden. Elke uitbreiding komt ook neer op het oplossen van een eigenwaardeprobleem: uiteindelijk is elke uitbreiding een variant op de methode van de machten om eigenvectoren te berekenen.

Een belangrijke opmerking is hier wel dat deze verschillende uitbreidingen lichtjes andere resultaten opleveren. Er bestaat niet zoiets als 'de' similariteitsscore tussen twee toppen van twee graffen, eenvoudigweg omdat elke uitbreiding een licht verschillend resultaat zal opleveren: in sommige gevallen zijn de resultaten op een constante na gelijk aan elkaar, in andere gevallen gebruiken de variaties iets andere criteria om similariteit te bepalen (bv. de methode van similariteit op de toppen van graffen en similariteit op de toppen en de bogen), waardoor de resultaten sowieso verschillend zijn. We kunnen in dat geval wel expliciet het verschil

tussen de methodes uitschrijven. Belangrijk is te onthouden dat similariteitscores vooral in verhouding tot de andere scores interessant zijn: we kunnen bv. gemakkelijk bepalen welke top van de ene graf, het meest lijkt op de top van een andere graf simpelweg door de grootste similariteitsscore te bekijken. Dit zal ook blijken uit het korte overzicht van toepassingen dat op het einde van hoofdstuk 2 toegevoegd, ook daar wordt een similariteitsscore enkel in verhouding tot de andere similareitsscores bekeken.

In het laatste hoofdstuk ontdekken we tot slot nieuw terrein: we proberen het concept van similariteit over te brengen op de meer algemene structuur van hypergraffen. Dat start met een uitgebreide reflectie op de resultaten van het vorige hoofdstuk: aan welke criteria moet zo'n similariteitsmethode voor hypergraffen allemaal voldoen? Voldoen de grafmethodes van het vorige hoofdstuk hier ook aan? Hypergraffen kunnen door verschillende grafrepresentaties worden voorgesteld en we proberen eerst om via deze weg tot een goede methode te komen. Het zal blijken dat de incidentiegrafrepresentatie van een hypergraf de beste resultaten oplevert. Nadien proberen we ook een methode te ontwikkelen aan de hand van de incidentiematrix, ook deze methode voldoet aan alle voorwaarden. Welnu: we kunnen bewijzen dat de methode met de incidentiegraf en de methode met de incidentiematrix op een constante na dezelfde resultaten oplevert.

Contents

Chapter 1

Preliminaries and notations

The Perron-Frobenius theorem states that a real square matrix with nonnegative entries has a unique largest real eigenvalue (i.e. with algebraic multiplicity 1) with an eigenvector that has only positive entries. The theorem was proved by Oskar Perron (1880-1975) in 1907 for strictly positive entries and extended by Ferdinand Georg Frobenius (1849-1917) to irreducible matrices with nonnegative entries.

1.1 Some families of matrices

In this section, we first introduce different kinds of matrices. Note that all matrices in this master thesis have real entries, unless otherwise stated. We start with permutation matrices and their uses. With permutation matrices, we can introduce irreducible matrices. Also nonnegative and primitive square matrices are presented. After defining those, we look at the Perron-Frobenius theorem.

1.1.1 Permutation matrices

Definition 1.1.1. *Given a permutation π of n elements:*

$$\pi : \{1, \dots, n\} \to \{1, \dots, n\},$$

with:

$$\pi = \begin{pmatrix} 1 & 2 & \cdots & n \\ \pi(1) & \pi(2) & \cdots & \pi(n) \end{pmatrix}$$

*the associated **permutation matrix** P_π is the $n \times n$-matrix obtained by permuting the rows of the identity matrix I_n according to π. So:*

$$P_\pi = \begin{pmatrix} \mathbf{e}_{\pi(1)}^T \\ \mathbf{e}_{\pi(2)}^T \\ \vdots \\ \mathbf{e}_{\pi(n)}^T \end{pmatrix}.$$

where $\mathbf{e}_j$ is the j-th column of I_n, the identy matrix.

Example 1.1.2. The permutation matrix P_π corresponding to the permutation

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 3 \end{pmatrix}$$

is:

$$P_\pi = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Note that $p_{ij} = 1$ if and only if $\pi(i) = j$.

Property 1.1.3. *A permutation matrix P satisfies:*

$$PP^T = I_n.$$

Proof. By direct computation, we get with $P = (p_{ij})$:

$$(PP^T)_{ij} = \sum_{k=1}^{n} p_{ik} p_{kj}^T = \sum_{k=1}^{n} p_{ik} p_{jk}$$

Assume $i \neq j$. Then for each k, $p_{ik}p_{jk} = 0$ since there is only one nonzero entry in the k-th row and $i \neq j$, p_{ik} and p_{jk} can't be both the nonzero entry. So, $(PP^T)_{ij} = 0$ when $i \neq j$.

When $i = j$, then there exists a $k' \in \{1, \ldots, n\}$ with $p_{ik'}p_{jk'} = 1$, since there is only one nonzero entry in the k-th row, this k' is unique, which results in $\sum_{k=1}^{n} p_{ik}p_{jk} = (PP^T)_{ij} = 1$.

In other words,

$$(PP^T)_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases},$$

this is exactly the formula for the entries of the identity matrix. □

Corollary 1.1.4. *The transpose of a permutation matrix P is its inverse:*

$$P^T = P^{-1}.$$

This can also more easily be concluded from the fact that a permutation matrix is clearly an orthogonal matrix (a real $n \times n$-matrix with orthonormal entries).

1.1.2 Nonnegative and primitive matrices

Definition 1.1.5. *Let A and B be two real $n \times r$-matrices. Then, $A \geq B$ (respectively $A > B$) if $a_{ij} \geq b_{ij}$ (respectively $a_{ij} > b_{ij}$) for all $1 \leq i \leq n, 1 \leq j \leq r$.*

Definition 1.1.6. *A real $n \times r$-matrix A is **nonnegative** if $A \geq 0$, with 0 the $n \times r$-zero matrix.*

Definition 1.1.7. *A real $n \times r$-matrix A is **positive** if $A > 0$, with 0 the $n \times r$-zero matrix.*

Since column vectors are $n \times 1$-matrices, we shall use the terms nonnegative and positive vector throughout.

Notation 1.1.8. *Let B be an arbitrary complex $n \times r$-matrix, then $|B|$ denotes the matrix with entries $|b_{ij}|$. This is not to be confused with the determinant of a square matrix B, which we denote by* $\det(B)$.

Definition 1.1.9. *A nonnegative square matrix A is called* ***primitive*** *if there is a $k \in \mathbb{N}_0$ such that all entries of A^k are positive.*

1.1.3 Irreducible nonnegative matrices

In developing the Perron-Frobenius theory, we shall first establish a series of theorems and lemmas on nonnegative irredicuble square matrices.

Definition 1.1.10. *A square matrix A is called* ***reducible*** *if there is a permutation matrix P such that*

$$PAP^T = \begin{pmatrix} B & 0 \\ C & D \end{pmatrix}$$

where B, and D are square matrices, each of size at least one and 0 is a zero matrix. A square matrix A is called ***irreducible*** *if it is not reducible.*

It follows immediately that a 1×1-matrix is always irreducible by definition. We now show a useful property to identify a reducible matrix.

Property 1.1.11. *Let A be an $n \times n$-matrix with $n \geq 2$. Consider a nonempty, proper subset S of $\{1, \ldots, n\}$ with $a_{ij} = 0$ for $i \in S$, $j \notin S$. Then A is reducible.*

Proof. Let $S = \{i_1, i_2, \ldots, i_k\}$, where we assume, without loss of generality, that $i_1 < i_2 < \cdots < i_{k-1} < i_k$. Let S^c be the complement of S, consisting of the ordered set of elements $j_1 < j_3 < \cdots < j_{n-k}$. Consider the permutation σ of $\{1, 2, \ldots, n\}$ given by

$$\sigma = \begin{pmatrix} 1 & 2 & \ldots & k & k+1 & k+2 & \ldots & n \\ i_1 & i_2 & \ldots & i_k & j_1 & j_2 & \ldots & j_{n-k} \end{pmatrix}$$

σ can be represented by the permutation matrix $P_\sigma = (p_{ij})$, where $p_{rs} = 1$ if $\sigma(r) = s$. We prove that

$$PAP^T = \begin{pmatrix} B & 0 \\ C & D \end{pmatrix}$$

where B and D are square matrices and 0 is a $k \times (n-k)$ zero matrix. Consider row c and column d, where $1 \leq c \leq k$ and $k+1 \leq d \leq n$:

$$(PAP^T)_{cd} = \sum_i \sum_j p_{ci} a_{ij} p_{dj}. \tag{1.1}$$

It is enough to show that each term in the summation is zero. Suppose $p_{ci} = p_{dj} = 1$. Thus $\sigma(c) = i$ and $\sigma(d) = j$. Since $1 \leq c \leq k$, then $i \in \{i_1, i_2, \ldots, i_k\}$; similarly, since $k+1 \leq d \leq n$, we have $j \in \{j_1, j_2, \ldots, j_{n-k}\}$. By assumption, for such a pair i, j, we have $a_{ij} = 0$. That completes the proof. □

We know prove some equivalent definitions for a nonnegative, irreducible square matrix.

Theorem 1.1.12. *Let $A \geq 0$ be a nonnegative $n \times n$-matrix. Then the following conditions are equivalent:*

(1) A is irreducible.

(2) $(I + A)^{n-1} > 0$

(3) For any pair (i, j), with $1 \leq i, j \leq n$, there is a positive integer $k = k(i, j) \leq n$ such that $(A^k)_{ij} > 0$.

Proof. (1) $\Rightarrow$ (2): Let $\mathbf{x} \geq 0, \mathbf{x} \neq \mathbf{0}$ be an arbitrary vector in $\mathbb{R}^n$. If a coordinate of $\mathbf{x}$ is positive, the same coordinate is positive in $\mathbf{x} + A\mathbf{x} = (I + A)\mathbf{x}$ as well. We claim that $(I + A)\mathbf{x}$ has fewer zero coordinates than $\mathbf{x}$ as long as $\mathbf{x}$ has a zero coordinate. If this claim is not true, then the number of zero coordinates must be at least equal, this means that for each coordinate j with $x_j = 0$ we would have that $x_j + (A\mathbf{x})_j = 0$. Let $J = \{j : x_j > 0\}$. For any $j \notin J, r \in J$, we have $(A\mathbf{x})_j = \sum_k a_{jk}x_k = 0$ and $x_r > 0$. It must be that $a_{jr} = 0$. It follows from Property 1.1.11 that A is reducible, which is a contradiction and the claim is proved. Thus $(I + A)\mathbf{x}$ has at most $n - 2$ zero coordinates. Continuing in this manner we conclude that $(I + A)^{n-1}\mathbf{x} > 0$. Let $\mathbf{x} = \mathbf{e}_i$, then the corresponding column of $(I + A)^{n-1}$ must be positive. Thus (2) holds.

(2) $\Rightarrow$ (3): We have $(I + A)^{n-1} > 0$, $A \geq 0$, so $A \neq 0$ and

$$A(I + A)^{n-1} = \sum_{k=1}^{n} \binom{n-1}{k-1} A^k > 0.$$

Thus for any i, j at least one of the matrices $A, A^2, \ldots, A^n$ has its (i, j)-th element entry positive.

(3) $\Rightarrow$ (1): Suppose A is reducible. Then for some permutation matrix P,

$$PAP^T = \begin{pmatrix} B_1 & 0 \\ C_1 & D_1 \end{pmatrix}$$

where B_1 and D_1 are square matrices. Furthermore, we know from Property 1.1.3 that $PAP^TPAP^T = PA^2P^T$, hence for some square matrices B_2, C_2 we have:

$$PA^2P^T = \begin{pmatrix} B_2 & 0 \\ C_2 & D_2 \end{pmatrix}$$

More generally, for some matrix C_t and square matrices B_t and D_t,

$$PA^tP^T = \begin{pmatrix} B_t & 0 \\ C_t & D_t \end{pmatrix}$$

Thus $(PA^tP^T)_{rs} = 0$ for $t = 1, 2, \ldots$ and for any r, s corresponding to an entry of the zero submatrix in PAP^T. Now, for $t = 1, \ldots, n$:

$$0 = (PA^tP^T)_{rs} = \sum_k \sum_l p_{rk} a_{kl}^{(t)} p_{ls}$$

By using the same reasoning as in 1.1, choose r, s so that $p_{rk} = p_{ls} = 1$. Then $a_{kl}^{(t)} = 0$ for all t, contradicting the hypothesis. This completes the proof. □

Corollary 1.1.13. *If A is irreducible then $I + A$ is primitive.*

Corollary 1.1.14. *A^T is irreducible whenever A is irreducible.*

Property 1.1.15. *No row or column of an irreducible matrix A can vanish. This means that A cannot have a row or a column of zeros.*

Proof. Suppose that A has a zero row, then it could be permuted to

$$PAP^T = \left(\begin{array}{c|c} 0 & 0\dots0 \\ \hline c_1 & \\ \vdots & D \\ c_n & \end{array}\right)$$

by some permutation matrix P. It follows from Definition 1.1.10 that A is reducible. Similarly, if A has zero column, it can be permuted to

$$PAP^T = \left(\begin{array}{c|c} & 0 \\ B & \vdots \\ & 0 \\ \hline c_1 \dots c_n & 0 \end{array}\right),$$

again from Definition 1.1.10 we conclude that A is reducible.

□

1.2 Perron-Frobenius Theorem

1.2.1 Spectral radii of nonnegative matrices

Definition 1.2.1. *Let A be an $n \times n$-matrix with real entries and eigenvalues λ_i, $1 \leq i \leq n$. Then:*

$$\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|$$

*is called the **spectral radius** of the matrix A.*

Geometrically, if all the eigenvalues λ_i of A are plotted in the complex plane, then $\rho(A)$ is the radius of the smallest disk $|z| \leq R$, with center at the origin, which includes all the eigenvalues of the matrix A.

We now establish a series of lemmas on nonnegative irreducible square matrices. These lemmas will allow us to prove the Perron-Frobenius at the end of this section.

If $A \geq 0$ is an irreducible $n \times n$-matrix and $\mathbf{x}$, a vector of size n with $\mathbf{0} \neq \mathbf{x} \geq 0$, let

$$r_{\mathbf{x}} = \min \left\{ \frac{\sum_{j=1}^{n} a_{ij} x_j}{x_i} \right\} \tag{1.2}$$

where the minimum is taken over all i for which $x_i > 0$. Clearly, $r_{\mathbf{x}}$ is a nonnegative real number and is the supremum of all $p \geq 0$ for which

$$A\mathbf{x} \geq p\mathbf{x} \tag{1.3}$$

We now consider the nonnegative quantity r defined by

$$r = \sup_{\substack{\mathbf{x} \geq 0 \\ \mathbf{x} \neq \mathbf{0}}} \{r_{\mathbf{x}}\} \tag{1.4}$$

As $r_{\mathbf{x}}$ and $r_{\alpha\mathbf{x}}$ have the same value for any scalar $\alpha > 0$, we need consider only the set B of vectors $\mathbf{x} \geq 0$ with $||\mathbf{x}|| = 1$, and we correspondingly let Q be the set of all vectors $\mathbf{y} = (I + A)^{n-1}\mathbf{x}$ where $\mathbf{x} \in B$. From Theorem 1.1.12 , Q consists only of positive vectors. Multiplying both sides of the inequality $A\mathbf{x} \geq r_{\mathbf{x}}\mathbf{x}$ by $(I + A)^{n-1}$, we obtain:

$$\forall \mathbf{y} \in Q : A\mathbf{y} \geq r_{\mathbf{x}}\mathbf{y},$$

and we conclude from (1.3) that $r_{\mathbf{y}} \geq r_{\mathbf{x}}$. Therefore, the quantity r of (1.4) can be defined equivalently as:

$$r = \sup_{\mathbf{y} \in Q} \{r_{\mathbf{y}}\} \tag{1.5}$$

As B is a compact set (in the usual topology) of vectors, so is Q, and as $r_{\mathbf{y}}$ is a continuous function on Q, we know from the extreme value theorem that there necessarily exists a positive vector $\mathbf{z}$ for which:

$$A\mathbf{z} \geq r\mathbf{z}, \tag{1.6}$$

and no vector $\mathbf{w} \geq 0$ exists for which $A\mathbf{w} > r\mathbf{w}$.

Definition 1.2.2. *We call all nonnegative, nonzero vectors* $\mathbf{z}$ *satisfying (1.6)* ***extremal vectors*** *of the matrix* A*.*

Lemma 1.2.3. *If* $A \geq 0$ *is an irreducible* $n \times n$*-matrix, the quantity* r *of (1.4) is positive.*

Proof. If $\mathbf{x}$ is the positive vector whose coordinates are all unity, then since the matrix A is irreducible, we know from Property 1.1.15 that no row of A can vanish, and consequently no component of $A\mathbf{x}$ can vanish. Thus, $r_{\mathbf{x}} > 0$, proving that $r > 0$. □

Lemma 1.2.4. *If* $A \geq 0$ *is an irreducible* $n \times n$*-matrix, each extremal vector* $\mathbf{z}$ *is a positive eigenvector of* A *with corresponding eigenvalue* r *of (1.4), i.e.,* $A\mathbf{z} = r\mathbf{z}$ *and* $\mathbf{z} > 0$*.*

Proof. Let $\mathbf{z}$ be an extremal vector with $A\mathbf{z} - r\mathbf{z} = \mathbf{t}$. If $\mathbf{t} \neq \mathbf{0}$, then some coordinate of $\mathbf{t}$ is positive; multiplying through by the matrix $(I + A)^{n-1}$, we have:

$$A\mathbf{w} - r\mathbf{w} > 0, \text{ with } \mathbf{w} = (I + A)^{n-1}\mathbf{z}$$

from Theorem 1.1.12 we know that $\mathbf{w} > 0$. It would then follow that $r_{\mathbf{w}} > r$, contradicting the definition of r in (1.5). Thus $A\mathbf{z} = r\mathbf{z}$, and since $\mathbf{w} > 0$ and $\mathbf{w} = (1 + r)^{n-1}\mathbf{z}$, then we have $\mathbf{z} > 0$, completing the proof. □

Lemma 1.2.5. *Let* $A \geq 0$ *be an irreducible* $n \times n$*-matrix, and let* B *be an* $n \times n$*- complex matrix with* $|B| \leq A$*. If* β *is any eigenvalue of* B*, then*

$$|\beta| \leq r, \tag{1.7}$$

where r *is the positive quantity of (1.4). Moreover, equality is valid in (1.7), i.e.,* $\beta = re^{i\phi}$*, if and only if* $|B| = A$*, and where* B *has the form:*

$$B = e^{i\phi}DAD^{-1}, \tag{1.8}$$

and D *is a diagonal matrix whose diagonal entries have modulus unity.*

Proof. If $\beta\mathbf{y} = B\mathbf{y}$ where $\mathbf{y} \neq \mathbf{0}$, then

$$\beta y_i = \sum_{j=1}^{n} b_{ij}y_i, \text{ with } 1 \leq i \leq n.$$

Using the hypotheses of the lemma and the notation of Definition 1.1.8, it follows that:

$$|\beta||\mathbf{y}| \leq |B||\mathbf{y}| \leq A|\mathbf{y}|, \tag{1.9}$$

which implies that $|\beta| \leq r_{|\mathbf{y}|} \leq r$, proving (1.7). If $|\beta| = r$, then $|\mathbf{y}|$ is an extremal vector of A. Therefore, from Lemma 1.2.4, $|\mathbf{y}|$ is a positive eigenvector of A corresponding to the positive eigenvalue r. Thus,

$$r|\mathbf{y}| = |B||\mathbf{y}| = A|\mathbf{y}|, \tag{1.10}$$

and since $|\mathbf{y}| > 0$, we conclude from (1.10) and the hypothesis $|B| \leq A$ that

$$|B| = A \tag{1.11}$$

For the vector $\mathbf{y}$, where $|\mathbf{y}| > 0$, let

$$D = \text{diag}\left\{\frac{y_1}{|y_1|}, \ldots, \frac{y_n}{|y_n|}\right\}.$$

It is clear that the diagonal entries of D have modulus unity, and

$$\mathbf{y} = D|\mathbf{y}|. \tag{1.12}$$

Setting $\beta = re^{i\phi}$, then $B\mathbf{y} = \beta\mathbf{y}$ can be written as:

$$C|\mathbf{y}| = r|\mathbf{y}|, \tag{1.13}$$

where

$$C = e^{-i\phi}D^{-1}BD. \tag{1.14}$$

From (1.10) and (1.13), equiting terms equal to $r|\mathbf{y}|$ we have

$$C|\mathbf{y}| = |B||\mathbf{y}| = A|\mathbf{y}|. \tag{1.15}$$

From the definition of the matrix C in (1.14), $|C| = |B|$. Combining with (1.11), we have:

$$|C| = |B| = A. \tag{1.16}$$

Thus, from (1.15) we conclude that $C|\mathbf{y}| = |C||\mathbf{y}|$, and as $|\mathbf{y}| > 0$, it follows that $C = |C|$ and thus $C = A$ from (1.16). Combining this result with (1.14), gives the desired result that $B = e^{i\phi}DAD^{-1}$. Conversely, it is obvious that if B has the form in (1.8), then $|B| = A$, and B has an eigenvalue β with $|\beta| = r$, which completes the proof. □

Corollary 1.2.6. *If $A \geq 0$ is an irreducible $n \times n$-matrix, then the positive eigenvalue r of Lemma 1.2.4 equals the spectral radius $\rho(A)$ of A*

Proof. Setting $B = A$ in Lemma 1.2.5 immediately gives us this result. □

In other words, if $A \geq 0$ is an irreducible $n \times n$-matrix, its spectral radius $\rho(A)$ is positive, and the intersection in the complex plane of the circle $|z| = \rho(A)$ with the positive real axis is an eigenvalue of A.

Definition 1.2.7. *A **principal square submatrix** of an $n \times n$-matrix A is any matrix obtained by crossing out any j rows and the corresponding j columns of A, with $1 \leq j \leq n$.*

Lemma 1.2.8. *If $A \geq 0$ is an irreducible $n \times n$-matrix, and B is any principal square submatrix of A, then $\rho(B) < \rho(A)$.*

Proof. If B is any principal submatrix of A, then there is an $n \times n$-permutation matrix P such that $B = A_{11}$ where

$$C = \begin{pmatrix} A_{11} & 0 \\ 0 & 0 \end{pmatrix}; PAP^T = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \tag{1.17}$$

Here, A_{11} and A_{22} are, respectively, $m \times m$ and $(n-m) \times (n-m)$ principal square submatrices of PAP^T, $1 \leq m \leq n$. Clearly, $0 \leq C \leq PAP^T$, and $\rho(C) = \rho(B) = \rho(A_{11})$, but as $C = |C| \neq PAP^T$, the conclusion follows immediately from Lemma 1.2.5 and Corollary 1.2.6. □

The following lemma is used to prove that $\rho(A)$ is a simple eigenvalue of A in the Perron-Frobenius theorem. The proof uses the extension of the product rule of derivation for multilinear functions $M(a_1, \ldots, a_k)$. Suppose $x_1, .., x_k$ are differentiable vector functions, then $M(x_1, \ldots, x_k)$ is differentiable and:

$$\frac{\mathrm{d}}{\mathrm{d}t}M(x_1, \ldots, x_k) = M(\frac{\mathrm{d}}{\mathrm{d}t}x_1, x_2, \ldots, x_k) + M(x_1, \frac{\mathrm{d}}{\mathrm{d}t}x_2, \ldots, x_k) + \ldots + M(x_1, x_2, \ldots, \frac{\mathrm{d}}{\mathrm{d}t}x_k)$$

The most important application of this rule is for the derivative of the determinant:

$$\frac{\mathrm{d}}{\mathrm{d}t}\det(x_1, \ldots, x_k) = \det(\frac{\mathrm{d}}{\mathrm{d}t}x_1, x_2, \ldots, x_k) + \det(x_1, \frac{\mathrm{d}}{\mathrm{d}t}x_2, \ldots, x_k) + \ldots + \det(x_1, x_2, \ldots, \frac{\mathrm{d}}{\mathrm{d}t}x_k)$$

Lemma 1.2.9. *Let A be an $n \times n$-matrix over the complex numbers and let $\phi(A, \lambda) = \det(\lambda I_n - A)$ be the characteristic polynomial of A. Let B_i be the principal submatrix of A formed by deleting the i-th row and column of A and let $\phi(B_i, \lambda)$ be the characteristic polynomial of B_i. Then:*

$$\phi'(A, \lambda) = \frac{\mathrm{d}\phi(A, \lambda)}{\mathrm{d}\lambda} = \sum_i \phi(B_i, \lambda)$$

Proof. The proof is immediately done by direct computation:

$$\phi(A, \lambda) = \det \begin{pmatrix} \lambda - a_{1,1} & -a_{1,2} & \ldots & -a_{1,n} \\ -a_{2,1} & \lambda - a_{2,2} & \ldots & -a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n,1} & -a_{n,2} & \ldots & \lambda - a_{n,n} \end{pmatrix}.$$

Using the extension of the product rule of derivation for multilinear functions

$$\phi'(A, \lambda) = \det \begin{pmatrix} 1 & 0 & \ldots & 0 \\ -a_{2,1} & \lambda - a_{2,2} & \ldots & -a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n,1} & -a_{n,2} & \ldots & \lambda - a_{n,n} \end{pmatrix} + \det \begin{pmatrix} \lambda - a_{1,1} & -a_{1,2} & \ldots & -a_{1,n} \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n,1} & -a_{n,2} & \ldots & \lambda - a_{n,n} \end{pmatrix} + \ldots$$
$$+ \det \begin{pmatrix} \lambda - a_{1,1} & -a_{1,2} & \ldots & -a_{1,n} \\ -a_{2,1} & \lambda - a_{2,2} & \ldots & -a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{pmatrix} = \sum_i \phi(B_i, \lambda).$$

□

1.2.2 Proof

We now collect the above results into the following main theorem: we finally arrived at the Perron-Frobenius Theorem:

Theorem 1.2.10. ***(Perron-Frobenius theorem)***
Let $A \geq 0$ be an irreducible $n \times n$-matrix. Then,

1. *A has a positive real eigenvalue equal to its spectral radius.*
2. *To $\rho(A)$ there corresponds an eigenvector $\mathbf{x} > 0$.*

3. $\rho(A)$ *increases when any entry of* A *increases.*

4. $\rho(A)$ *is a simple eigenvalue of* A*.*

5. If $A\mathbf{x} = \rho(A)\mathbf{x}$ *where* $\mathbf{x} > 0$ *and* $\mathbf{x}$ *is a normalized vector, then* $\mathbf{x}$ *is unique.*

Proof. (1) and (2) follow immediately from Lemma 1.2.4 and Corollary 1.2.6.

(3) Suppose we increase some entry of the matrix A, giving us a new irreducible matrix $\tilde{A}$ where $\tilde{A} \geq A$ and $\tilde{A} \neq A$. Applying Lemma 1.2.5, we conclude that $\rho(\tilde{A}) > \rho(A)$.

(4) $\rho(A)$ is a simple eigenvalue of A, i.e., $\rho(A)$ is a zero of multiplicity one of the characteristic polynomial $\phi(\lambda) = \det(\lambda I_n - A)$, we make use of Lemma 1.2.9 by using the fact that $\phi'(\lambda)$ is the sum of the determinants of the principal $(n-1) \times (n-1)$ submatrices of $\lambda I - A$. If A_i is any principal submatrix of A, then from Lemma 1.2.8, $\det(\lambda I - A_i)$ (with I the identity matrix with the same size as the principal submatrix A_i) cannot vanish for any $\lambda \geq \rho(A)$. From this it follows that:

$$\det(\rho(A)I - A_i) > 0,$$

and thus

$$\phi'(\rho(A)) > 0.$$

Consequently, $\rho(A)$ cannot be z zero of $\phi(\lambda)$ of multiplicity greater than one and thus $\rho(A)$ is a simple eigenvalue of A.

(5) If $A\mathbf{x} = \rho(A)\mathbf{x}$ where $\mathbf{x} > 0$ and $||\mathbf{x}|| = 1$ ($\|\cdot\|$ denotes the standard Euclidean norm), we cannot find another eigenvector $\mathbf{y} \neq s\mathbf{x}$, with s a scalar, of A with $A\mathbf{y} = \rho(A)\mathbf{y}$, so that the eigenvector $\mathbf{x}$, meaning that the normalized eigenvector $\mathbf{x}$ is uniquely determined. □

With the previous proof in mind, the following definition comes not unexpected:

Definition 1.2.11. *If a matrix* A *has an eigenvalue equal to the spectral radius* $\rho(A)$*, this eigenvalue is called the* ***Perron root****, the corresponding eigenvector* $\mathbf{x}$ *such that:*

$$A\mathbf{x} = \rho(A)\mathbf{x} \quad \textit{and} \quad \|\mathbf{x}\|_1 = 1$$

is called the ***Perron vector****.*

Corollary 1.2.12. *Being a strictly positve square matrix (*$A > 0$*) is a sufficient condition to apply the Perron-Frobenius Theorem.*

Proof. This follows immediately from Theorem 1.1.12: a positive square matrix is always irreducible. □

1.2.3 Example

To check wether a matrix with nonnegative entries is primitive, irreducible or neither, we just have to replace all nonzero entries by 1 since this does not affect the classification. The matrix

$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

is strictly positive and thus primitive. The matrices

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

both have 1 as a double eigenvalue hence can not be irreducible. The matrix $\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$ satisfies:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^2 = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

and hence is primitive. The same goes for

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix},$$

this matrix is irreducible but not primitive. Its eigenvalues are 1 and -1.

1.3 Norms

If one has several vectors in $\mathbb{R}^n$ or several matrices in $\mathbb{R}^{n\times m}$, how do we measure that some of them are 'large' and some of them are 'small'? One way to answer this question is to study norms, which are basically functions that a assign a positive 'size' to a vector in a vector space. The norms that we define in this master thesis are limited to $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ and we call them respectively *vector norms* (norms on $\mathbb{R}^n$) and *matrix norms* (norms on $\mathbb{R}^{n\times m}$). This section is mainly based on chapter 2 from [GVL12] and chapter 5 form [HJ86].

1.3.1 Vector norms

Definition 1.3.1. *A* ***vector norm*** *on $\mathbb{R}^n$ is a function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ with the following properties:*

1. $\|\mathbf{x}\| \geq 0$, *for all* $\mathbf{x} \in \mathbb{R}^n$ *with equality if and only if* $\mathbf{x} = \mathbf{0}$.
2. $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ *for all* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.
3. $\|\alpha\mathbf{x}\| = |\alpha|\|\mathbf{x}\|$ *for all* $\alpha \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^n$.

We now give some well known vector norms:

p-norms

Definition 1.3.2. *The Hölder or* p-norms *are defined by:*

$$\|\mathbf{x}\|_p = (|x_1|^p + \cdots + |x_n|^p)^{\frac{1}{p}} = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}},$$

for $\mathbf{x} \in \mathbb{R}^n$.

The 1-norm is also known as the *Manhattan* norm:

$$\|\mathbf{x}\|_1 = |x_1| + \cdots + |x_n|$$

The 2-norm is also known as the standard *Euclidean norm*:

$$\|\mathbf{x}\|_2 = (|x_1|^2 + \cdots + |x_n|^2)^{\frac{1}{2}} = (\mathbf{x}^T\mathbf{x})^{\frac{1}{2}}$$

Notice that the 2-norm is invariant under orthogonal transformation, for if $QQ^T = I$ with $Q \in \mathbb{R}^{n\times n}$ and $\mathbf{x} \in \mathbb{R}^n$:

$$\|Q\mathbf{x}\|_2^2 = \mathbf{x}QQ^T\mathbf{x}^T = \mathbf{x}^T\mathbf{x} = \|\mathbf{x}\|_2^2$$

Maximum norm

Finally, when $p \to \infty$ we get the *maximum norm*:

$$\|\mathbf{x}\|_\infty = \max(|x_1|, \ldots, |x_n|)$$

We will prove this:

Theorem 1.3.3. *Let* $\mathbf{x} \in \mathbb{R}^n$ *then:*

$$\lim_{p\to\infty} \|\mathbf{x}\|_p = \|\mathbf{x}\|_\infty = \max(|x_1|, \ldots, |x_n|)$$

Proof. Rewrite $\|\mathbf{x}\|_p$ as:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}} = \|\mathbf{x}\|_\infty \left(\sum_{i=1}^{n} \left(\frac{|x_i|}{\|\mathbf{x}\|_\infty}\right)^p\right)^{\frac{1}{p}}$$

Note that $\left(\frac{|x_i|}{\|\mathbf{x}\|_\infty}\right) \leq 1$ for every i, with equality at least once and at most n times, then:

$$\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_p \leq \|\mathbf{x}\|_\infty n^{\frac{1}{p}}$$

because $n > 0$, we get $\lim_{p\to\infty} n^{\frac{1}{p}} = 1$, then:

$$\lim_{p\to\infty} \|\mathbf{x}\|_p = \|\mathbf{x}\|_\infty.$$

□

Norm equivalence

One very important property of all the norms of $\mathbb{R}^n$ is that they are all *equivalent*, meaning that when two vectors have about the same size according to one vector norm, they also will have more or less the same size according to another vector norm.

Theorem 1.3.4. *All norms on* $\mathbb{R}^n$ *are* equivalent*, i.e., if* $\|\cdot\|_\alpha$ *and* $\|\cdot\|_\beta$ *are norms on* $\mathbb{R}^n$*, then there exist positive constants* $c_1, c_2 \in \mathbb{R}^+$ *such that for all* $\mathbf{x} \in \mathbb{R}$ËĘ*n:*

$$c_1\|\mathbf{x}\|_\alpha \leq \|\mathbf{x}\|_\beta \leq c_2\|\mathbf{x}\|_\alpha$$

Proof. We will prove for the norm $\|\cdot\|_2$ that there are $c_1 > 0, c_2 > 0$ such that for all $\mathbf{x} > 0$ it holds:

$$c_1\|\mathbf{x}\|_\alpha \leq \|\mathbf{x}\|_2 \leq c_2\|\mathbf{x}\|_\alpha.$$

From this, the theorem easily follows (notice that for $\mathbf{x} = \mathbf{0}$ this inequality evidently holds by definition of a norm). First let

$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \cdots + x_n\mathbf{e}_n,$$

where $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$ is a basis for $\mathbb{R}^n$. By the triangle inequality:

$$\|\mathbf{x}\|_\alpha \leq \sum_{j=1}^{n} |x_j| \|\mathbf{e}_j\|_\alpha.$$

By Cauchy's inequality, for the standard inner product $\sum_{j=1}^{n} |x_j| \|\mathbf{e}_j\|_\alpha$,

$$\sum_{j=1}^{n} |x_j| \|\mathbf{e}_j\|_\alpha \leq \left(\sum_{j=1}^{n} x_j^2\right)^{1/2} \left(\sum_{j=1}^{n} \|\mathbf{e}_j\|_\alpha^2\right)^{1/2},$$

so:

$$c_1\|\mathbf{x}\|_\alpha \leq \|\mathbf{x}\|_2, \text{where } c_1 = \frac{1}{\left(\sum_{j=1}^n \|\mathbf{e}_j\|_\alpha^2\right)^{1/2}}.$$

Now consider the function $\mathbf{x} \to \|\mathbf{x}\|_\alpha$ on the set $\|\mathbf{x}\|_2 = 1$. The set $K = \{\mathbf{x} \mid \|\mathbf{x}\|_2 = 1\}$ is closed and bounded, so the theorem of Heine-Borel (see Theorem 3.5.2 in [Cae11]) shows that it is compact. Now, we have actually proved that the function $\mathbf{x} \to \|\mathbf{x}\|$ is continuous on $\mathbb{R}^n$, meaning that if $\|x_j\|_2 \to 0$ then $\|x_j\|_\alpha \to 0$ $(j \in \{1, \ldots, n\})$, so this function attains it's minimum m on K, m is strictly greater than 0 because a norm is always greater or equal to zero and $\mathbf{0} \notin K$. Now let $\mathbf{x} \neq \mathbf{0}$ be any vector on $\mathbb{R}^n$ and let $\mathbf{u} = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \in \mathbb{R}^n$. Then $\|\mathbf{u}\|_2 = 1$ so $\|\mathbf{u}\|_\alpha \geq m$. Hence:

$$\mathbf{x} \geq m\|\mathbf{x}\|_2, \text{ or } \|\mathbf{x}\|_2 \leq c_2\|\mathbf{x}\|, \text{ where } c_2 = \frac{1}{m}.$$

□

For example, for any $\mathbf{x} \in \mathbb{R}^n$ we have:

$$\begin{aligned} \|\mathbf{x}\|_2 &\leq \|\mathbf{x}\|_1 \leq \sqrt{n}\|\mathbf{x}\|_2 \\ \|\mathbf{x}\|_\infty &\leq \|\mathbf{x}\|_2 \leq \sqrt{n}\|\mathbf{x}\|_\infty \\ \|\mathbf{x}\|_\infty &\leq \|\mathbf{x}\|_1 \leq n\|\mathbf{x}\|_\infty \end{aligned}$$

1.3.2 Matrix norms

The analysis of algorithms where matrices are involved requires that we are able to assess the size of matrices. We introduce therefore *matrix norms*, off course, this definition is completely analogous but with an extra condition: for square matrices, the matrix norm must be submultiplicative :

Definition 1.3.5. *A **matrix norm** on $\mathbb{R}^{n\times m}$ is a function $\|\cdot\| : \mathbb{R}^{n\times m} \to \mathbb{R}$ with the following properties:*

1. *$\|A\| \geq 0$, for all $A \in \mathbb{R}^{n\times m}$ with equality if and only if A is a matrix with only zero entries.*
2. *$\|A + B\| \leq \|A| + \|B\|$ for all $A, B \in \mathbb{R}^{n\times m}$.*
3. *$\|\alpha A\| = |\alpha|\|A\|$ for all $\alpha \in \mathbb{R}$, $A \in \mathbb{R}^{n\times m}$.*
4. *For all $A, B \in \mathbb{R}^{n\times n}$: $\|AB\| \leq \|A\|\|B\|$.*

Frobenius norm

One of the most frequently used matrix norms for a matrix $A \in \mathbb{R}^{n\times m}$ is the so called *Frobenius norm*:

$$\|A\|_F = \left(\sum_{i=1}^n \sum_{j=1}^m a_{ij}^2\right)^{\frac{1}{2}} = \text{trace}(A^T A)^{\frac{1}{2}},$$

If $A \in \mathbb{R}^{1\times m}$, then the Frobenius norm equals the 2-norm.

p-norms

We can also define p-norms on matrices:

$$\|A\|_p = \sup_{\mathbf{x}\neq 0} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$$

Maximum norm

First notice that the norm defined for $A \in \mathbb{R}^{n\times n}$ as:

$$\|A\|_{\max} = \max_{1\leq i,j\leq n} |a_{ij}|$$

is a norm on the vector space $\mathbb{R}^{n\times n}$, but is not a matrix norm. Consider the matrix $J = \left(\begin{smallmatrix} 1 & 1 \\ 1 & 1 \end{smallmatrix}\right)$ and compute $J^2 = 2J$. So: $\|J\|_{\max} = 1$, $\|J^2\|_{\max} = \|2J\|_{\max} = 2\|J\|_{\max} = 2$. So it is not the case that $\|J^2\|_{\max} \leq \|J\|^2_{\max}$, and hence $\|\cdot\|_{\max}$ is not a submultiplicative norm. Therefore we define the maximum norm as follows:

Definition 1.3.6. *Let $\|\cdot\|$ be a vector norm on $\mathbb{R}^n$. Define the* ***maximum norm*** *$|||\cdot|||$ on $\mathbb{R}^{n\times n}$ by:*

$$|||A||| = \max_{\|\mathbf{x}\|=1} \|A\mathbf{x}\|$$

It's easy to see these equivalences:

$$\begin{aligned} |||A||| &= \max_{\|\mathbf{x}\|=1} \|A\mathbf{x}\| \\ &= \max_{\|\mathbf{x}\|\leq 1} \|A\mathbf{x}\| \\ &= \max_{\mathbf{x}\neq 0} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|} \end{aligned}$$

Because we need in the next paragraph the fact that the maximum norm is indeed a matrix norm, we prove this:

Theorem 1.3.7. *The function $|||\cdot|||$, also called the maximum norm, is a matrix norm on $\mathbb{R}^{n\times n}$.*

Proof. Let A be a $n \times n$-matrix. Property (1) of matrix norms follows from the fact that $|||A|||$ is the maximum of a nonnegative valued function. That this maximum exists follows from the extreme value theorem of Bolzano (see Theorem 5.5.1 in [Cae11]) because $\|A\mathbf{x}\|$ is a continuous function of $\mathbf{x}$ on the unit ball with $\|\mathbf{x}\| = 1$, which is a compact set. That $|||A||| = 0$ occurs only when A is a matrix with only zero entries follows from the fact that $A\mathbf{x} = \mathbf{0}_{n\times 1}$ for all $\mathbf{x}$ is only possible when $A = \mathbf{0}_{n\times n}$.

To see (2), we notice that the triangle inequality is inherited from the vector norm $\|\cdot\|$,

since:

$$\begin{aligned} |||A+B||| &= \max_{\|\mathbf{x}\|=1} \|(A+B)\mathbf{x}\| \\ &= \max_{\|\mathbf{x}\|=1} \|A\mathbf{x}+B\mathbf{x}\| \\ &\leq \max_{\|\mathbf{x}\|=1} (\|A\mathbf{x}\|+\|B\mathbf{x}\|) \\ &\leq \max_{\|\mathbf{x}\|=1} \|A\mathbf{x}\| + \max_{\|\mathbf{x}\|=1} \|B\mathbf{x}\| \\ &= |||A||| + |||B||| \end{aligned}$$

Property (3) of matrix norms follows from the calculation :

$$|||\alpha A||| = \max_{\|\mathbf{x}\|=1} \|\alpha A\mathbf{x}\| = \max_{\|\mathbf{x}\|=1} |\alpha| \|A\mathbf{x}\| = |\alpha| \max_{\|\mathbf{x}\|=1} \|A\mathbf{x}\| = |\alpha| |||A|||$$

The submultiplicative property (4) follows from the fact that:

$$\begin{aligned} |||AB||| &= \max_{\mathbf{x}\neq 0} \frac{\|AB\mathbf{x}\|}{\|\mathbf{x}\|} \\ &= \max_{\mathbf{x}\neq 0} \frac{\|AB\mathbf{x}\|}{\|B\mathbf{x}\|} \frac{\|B\mathbf{x}\|}{\|\mathbf{x}\|} \\ &\leq \max_{\mathbf{x}\neq 0} \frac{\|A\mathbf{y}\|}{\|\mathbf{y}\|} \max_{\mathbf{x}\neq 0} \frac{\|B\mathbf{x}\|}{\|\mathbf{x}\|} \\ &= |||A|||.|||B||| \end{aligned}$$

□

Again, we can show that all matrix norms are equivalent with the same reasoning as in Theorem 1.3.4.

1.3.3 Spectral radius formula

In this section, we prove the spectral radius formula also known as the Gelfand's formula. This formula is a way to calculate the spectral radius of a square matrix as a limit of matrix norms. The formula will play an important role in Chapter 2, where it will be used to get some more advanced results of the Perron-Frobenius when considering nonnegative, symmetric matrices.

Lemma 1.3.8. *Let A be a square matrix, then*

$$\rho(A)^k = \rho(A^k)$$

Proof. This follows immediately from the fact that if A has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, then $\lambda_1^k, \lambda_2^k, \ldots, \lambda_n^k$ are all the eigenvalues of A^k (see 1.2 in [**?**]), so:

$$\rho(A)^k = \left(\max_{1\leq i\leq n} |\lambda_i|\right)^k = |\max \lambda_i|^k = \rho(A^k).$$

□

Lemma 1.3.9. *Let A be a square matrix and let $\|\cdot\|$ be a matrix norm then:*

$$\rho(A) \leq \|A\|$$

Proof. Let λ be an eigenvalue of A and let $\mathbf{x} \neq \mathbf{0}$ be a corresponding eigenvector ($\mathbf{x} \in \mathbb{R}^n$). From $A\mathbf{x} = \lambda\mathbf{x}$, we get:

$$AX = \lambda X, \quad \text{where } X = (\mathbf{x} \dots \mathbf{x}) \in \mathbb{R}^{n\times n} \setminus \mathbf{0_{n\times n}}$$

It follows from property (3) and (4) of matrix norms that:

$$|\lambda|\|X\| = \|\lambda X\| = \|AX\| \leq \|A\|\|X\|,$$

and simplifying by $\|X\|$ ($\|X\| > 0$ by matrix norm property 1) gives:

$$|\lambda| \leq \|A\|.$$

This holds for all eigenvalues, so also for the maximum of all the eigenvalues of A. Hence the result follows. □

Before we prove the following lemma, remember the well known theorem about the Jordan canonical form of a square matrix A (see Theorem 2.3.1 in [**?**]).

Theorem 1.3.10. *For each square matrix A there exists an invertible matrix P such that*

$$PAP^{-1} = J'$$

*J' is called the **Jordan normal form** of A and:*

$$J' = \begin{pmatrix} J_1 & & \\ & \ddots & \\ & & J_p \end{pmatrix}$$

where each block J_i is a square matrix of the form:

$$J_i = \begin{pmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \lambda_i & 1 \\ 0 & 0 & 0 & 0 & \lambda_i \end{pmatrix},$$

with λ_i's eigenvalues.

Lemma 1.3.11. *Let A be an $n \times n$ matrix and $\epsilon > 0$, there exist a matrix norm $\|.\|_\rho$ such that:*

$$\|A\|_\rho \leq \rho(A) + \epsilon$$

Proof. The Jordan canonical form of A is:

$$A = S \begin{pmatrix} J_{n_1}(\lambda_1) & 0 & \dots & 0 \\ 0 & J_{n_2}(\lambda_2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & J_{n_k}(\lambda_k) \end{pmatrix} S^{-1},$$

where $S \in \mathbb{R}^{n\times n}$ is an invertible matrix, $\lambda_1, \ldots, \lambda_k$ are the eigenvalues of A and $n_1 + \cdots + n_k = n$. Let:

$$D(\eta) = \begin{pmatrix} D_{n_1}(\eta) & 0 & \ldots & 0 \\ 0 & D_{n_2}(\eta) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & D_{n_k}(\eta) \end{pmatrix} \quad \text{with } D_m(\eta) = \begin{pmatrix} \eta & 0 & \ldots & 0 \\ 0 & \eta^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & \eta^m \end{pmatrix}$$

Since the left multiplication by $D_m(1/\epsilon)$ multiplies the ith row by $1/\epsilon^i$ and the right multiplication on the right by $D_m(\epsilon)$ multiplies the jth colum by ϵ^j, we calculate:

$$D(1/\epsilon)S^{-1}ASD(\epsilon) = \begin{pmatrix} B_{n_1}(\lambda_1, \epsilon) & 0 & \ldots & 0 \\ 0 & B_{n_2}(\lambda_2, \epsilon) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & B_{n_k}(\lambda_k, \epsilon) \end{pmatrix}$$

with

$$B_m(\lambda, \epsilon) = D_m(1/\epsilon)J_m(\lambda)D_m(\epsilon) = \begin{pmatrix} \lambda & \epsilon & 0 & \ldots & 0 \\ 0 & \lambda & \epsilon & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \lambda & \epsilon \\ 0 & \ldots & 0 & 0 & \lambda \end{pmatrix}$$

We now define the matrix norm for $M \in \mathbb{R}^{n\times n}$ by:

$$\|M\|_\rho = \max_{\|\mathbf{x}\|_1=1} \|D(1/\epsilon)S^{-1}MSD(\epsilon)\mathbf{x}\|_1 \tag{1.18}$$

$$= \max_{l\in[1:n]} \sum_{k=1}^{n} |(D(1/\epsilon)S^{-1}MSD(\epsilon))_{k,l}|. \tag{1.19}$$

The conditions for being a matrix norm are trivially met because $\max_{\|\mathbf{x}\|_1=1} \|A\mathbf{x}\|$ is a matrix norm by Theorem 1.3.7. □

Theorem 1.3.12. ***Spectral radius formula*** *Let a be an $n \times n$ matrix and let $\|\cdot\|$ be a matrix norm then:*

$$\rho(A) = \lim_{k\to\infty} \|A^k\|^{1/k}$$

Proof. Given $k \geq 0$, we use Lemma 1.3.9 to write:

$$\rho(A)^k = \rho(A^k) \leq \|A^k\|,$$

so:

$$\rho(A) \leq \|A^k\|^{1/k}.$$

Taking the limit as $k \to \infty$ gives $\rho(A) \leq \lim_{k\to\infty} \|A^k\|^{1/k}$. To establish the reverse inequality, we need to prove that, for any $\epsilon > 0$, there exists a $K \geq 0$ such that $\|A^k\|^{1/k} \leq \rho(A) + \epsilon$ for all $k \geq K$. From Lemma 1.3.11, we know that there exists a matrix norm $\|.\|_\rho$ so $\|A\|_\rho \leq \rho(A) + \epsilon$.

Moreover, by the equivalence of the norms (see Theorem 1.3.4) on $\mathbb{R}^{n\times n}$, we know that there exists some constant $C > 0$ such that $\|M\| \leq C\|M\|_\rho$ for all $M \in \mathbb{R}^{n\times n}$. Then, for any $k \geq 0$,

$$\|A^k\| \leq C\|A^k\|_\rho \leq C\|A\|_\rho^k \leq C(\rho(A) + \epsilon)^k$$

So:

$$\|A^k\|^{1/k} \leq C^{1/k}(\rho(A) + \epsilon),$$

and thus:

$$\lim_{k\to\infty} \|A^k\|^{1/k} \leq \rho(A) + \epsilon$$

This implies the existence of $K \geq 0$ such that $\|A^k\|^{1/k} \leq \rho(A) + \epsilon$ for $k \geq K$, as desired. □

1.4 Numerical analysis

1.4.1 Bachmann-Landau notations

For comparing the computational cost of algorithms, it's important to know the *Bachmann-Landau notations*. These notations are used to describe the limiting behavior of a function in terms of simpler functions. These notations are used a lot in computer science to classify algorithms by how their number of steps depends on changes in input size. We are only interested in the effects on the number of steps for really large input sizes, so constants don't play any role in the classification.

Big O

Definition 1.4.1. ***(Big O)***
The Big O of a function g is the set of all functions f that are bounded above by g asymptotically (up to constant factor).

$$O(g(n)) = \{f \mid \exists c, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

We now prove a very simple lemma to show that indeed constant factors don't matter for the Big O:

Lemma 1.4.2. $\forall k > 0 : O(k.g(n)) = O(g(n))$

Proof.

$$\begin{aligned} O(k.g(n)) &= \{f \mid \exists c, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq f(n) \leq k.c.g(n)\} \\ &= \{f \mid \exists c, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq f(n) \leq (k.c).g(n)\} \\ \text{let c' = k.c} & \\ &= \{f \mid \exists c', n_0 \geq 0 : \forall n \geq n_0 : 0 \leq f(n) \leq c'.g(n)\} \\ &= O(g(n)) \end{aligned}$$

□

Small O

Definition 1.4.3. ***(Small O)***
Small O of a function g is the set of all functions f that are dominated by g asymptotically.

$$o(g(n)) = \{f | \forall \varepsilon > 0, \exists n_0 \forall n \geq n_0 : f(n) \leq \varepsilon.g(n)\}$$

Note that the small-notation is a much stronger statement than the corresponding big o-notation: every function that is in the small o of g is also in big o, but the inverse isn't necessarily true. Intuitively, $f(x) \in o(g(x))$ means that $g(x)$ grows much faster than $f(x)$.

Asymptotical Equality

Definition 1.4.4. ***(Asymptotically Equal)***

Let f and g be real functions, then f is asymptotically equal to g iff $\lim_{x \to +\infty} \frac{f(x)}{g(x)} = 1$. *Notation:* $f \sim g$.

In fact asymptotical equality, can also be defined equivalently as an equivalency relation: $f \sim g \Leftrightarrow (f - g) \in o(g)$. It's trivially clear that as $f \sim g \Rightarrow f \in O(g)$.

1.4.2 The Power Method

We now introduce the *classical power method*, also called the *Von Mises iteration* ([GVL12]). This numerical algorithm is the core of the whole thesis, because we will see in the next chapters a lot of different algorithms that are in fact just adaptations of this iterative method.

The power method is an eigenvalue algorithm that, given a diagonalizable matrix A, finds the eigenvalue λ with the greatest magnitude and a corresponding eigenvector $\mathbf{v}$ such that:

$$A\mathbf{v} = \lambda\mathbf{v}.$$

The power method is special because it doesn't use any matrix decomposition technique for obtaining results, making it suitable for very large matrices. On the other hand, it only finds one eigenvalue with a corresponding eigenvector and the the iterative process might converge very slowly.

There are plenty of variations of the power method available that overcome all these limitations, but we limit our discussion here to the very basic method, therefore we call it the *classical* power method.

We first introduce some needed definitions, theorems and notations.

Definition 1.4.5. *Consider a real $n \times n$-matrix A with (not necessarily different) eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. When*

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|,$$

*λ_1 is called the **dominant eigenvalue**.*

Corollary 1.4.6. *If a real, $n \times n$-matrix A has a dominant eigenvalue λ_1, then λ_1 is real.*

Proof. Recall that the eigenvalues of a real matrix A are in general complex, and occur in conjugate pairs. So if λ_1 would be complex, the complex conjugate of λ_1 would also be an eigenvalue with the same modulus. Because λ_1 must be the only eigenvalue that is strictly greater than all the other eigenvalues, this is impossible. □

Notation 1.4.7. *Let $\mathbf{x}^{(i)}$ denote vector $\mathbf{x}$ at iteration step i.*

The algorithm

Let $A \in \mathbb{R}^{n \times n}$ be a diagonalizable matrix with dominant eigenvalue λ_1. We know that A has an eigenbasis $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$. Every vector $\mathbf{x}^{(0)} \in \mathbb{R}^n$ can be written as a linear combination of elements in V, because V spans the space $\mathbb{R}^n$. So:

$$\mathbf{x}^{(0)} = \sum_{i=1}^{n} \xi_i \mathbf{v}_i.$$

Now construct the sequence of vectors $\mathbf{x}^{(k)}$:

$$\mathbf{x}^{(k)} = A\mathbf{x}^{(k-1)} = A^k\mathbf{x}^{(0)}$$

Now:

$$\begin{aligned} A^k\mathbf{x}^{(0)} &= \sum_{i=1}^{n} \xi_i A^k \mathbf{v}_i \\ &= \sum_{i=1}^{n} \xi_i \lambda_i^k \mathbf{v}_i \\ &= \lambda_1^k \left\{ \xi_1 \mathbf{v}_1 + \xi_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \mathbf{v}_2 + \cdots + \xi_n \left(\frac{\lambda_n}{\lambda_1}\right)^k \mathbf{v}_n \right\} \end{aligned}$$

Because $|\lambda_i| < |\lambda_1|$ for $i > 1$, we have that

$$\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0 \text{ for } k \to \infty$$

so:

$$\mathbf{x}^{(k)} = \lambda_1^k \xi_1 \mathbf{v}_1 + o(1) \text{ for } k \to \infty \tag{1.20}$$

This means that for k large enough $\mathbf{x}^{(k+1)}$ is almost equal to λ_1 times $\mathbf{x}^{(k)}$. So when the ratio between the corresponding vector entries in $\mathbf{x}^{(k+1)}$ and $\mathbf{x}^{(k)}$ becomes constant after k iteration steps, then this ratio will be equal to the dominant eigenvalue λ_1. The vector $\mathbf{x}^{(k)}$ will be a corresponding eigenvector because it is is proportional to $\mathbf{v_1}$.

The start value $\mathbf{x}^{(0)}$ must only satisfy the condition that $\xi_1 \neq 0$, in other words: $\mathbf{x}^{(0)}$ must have a non-zero component belonging to the dominant eigenvector. Each randomly chosen option for $\mathbf{x^{(0)}}$ will fulfill this requirement. Even if we are so unlucky to pick a starting vector which doesn't, subsequent $\mathbf{x}^{(k)}$ will again fulfill the requirement because rounding errors sustained during the iteration will have a component in this direction.

A practical problem arises now when one of the components of $\mathbf{x}^{(k)}$ is equal to zero. If we want to take the ratio between the corresponding components of $\mathbf{x}^{(k+1)}$ and $\mathbf{x}^{(k)}$ we get a division by zero. We can solve this by property (3) of vector norms (see Defintion 1.3.1):

$$\|\lambda_1 \mathbf{x}^{(k)}\| = |\lambda_1| \|\mathbf{x}^{(k)}\|.$$

Because $\mathbf{x}^{(k)} \neq \mathbf{0}$ we have that $\|\mathbf{x}^{(k+1)}\| \approx \|\lambda_1 \mathbf{x}^{(k)}\|$, so we calculate λ_1 in the power method by:

$$|\lambda_1| = \lim_{k\to\infty} \frac{\|\mathbf{x}^{(k+1)}\|}{\|\mathbf{x}^{(k)}\|}$$

To decide on the sign of λ_1 (λ_1 is always real, see Corollary 1.4.6), just divide two non-zero components of $\mathbf{x}^{(k+1)}$ and $\mathbf{x}^{(k)}$.

Another issue to address is that the components of $\mathbf{x^{(k)}} = A^k\mathbf{x^{(0)}}$ can become too large or too low, which can cause an overflow or underflow in the real number representation of computers. To avoid this, we use normed versions of the $\mathbf{x^{(k)}}$-vectors: we start with a vector $\mathbf{y}^{(0)}$ with $\|\mathbf{y}^{(0)}\| = 1$. Subsequently, we calculate for $k = 0, 1, \ldots$:

$$\mathbf{z}^{(k+1)} = A\mathbf{y}^{(k)}, \quad \mu_{k+1} = \|\mathbf{z}^{(k+1)}\|, \quad \mathbf{y}^{(k+1)} = \frac{\mathbf{z}^{(k+1)}}{\mu_{k+1}}.$$

The vectors $\mathbf{y}^{(k)}$ all have magnitude 1 and the components of $\mathbf{z}^{(k)}$ are restricted because:

$$\|\mathbf{z}^{(k)}\| = \|A\mathbf{y}^{(k-1)}\| \leq \|A\| \|\mathbf{y}^{(k-1)}\| = \|A\|$$

and when A is invertible we have $\|\mathbf{y}^{(k-1)}\| = 1 \leq \|A^{-1}\|\|\mathbf{z}^{(k)}\|$. Thus $\|\mathbf{z}^{(k)}\| \geq \frac{1}{\|A^{-1}\|}$. If we want to calculate the eigenvalue we can use:

$$\begin{aligned} A^k\mathbf{y}^{(0)} &= A^{k-1}A\mathbf{y}^{(0)} = A^{k-1}\mathbf{z}^{(1)} = A^{k-1}\mu_1\mathbf{y}^{(1)} \\ &= A^{k-2}\mu_1 A\mathbf{y}^{(1)} = A^{k-2}\mu_1\mathbf{z}^{(2)} = A^{k-2}\mu_1\mu_2\mathbf{y}^{(2)} \\ &= \cdots \\ &= \mu_1\mu_2\ldots\mu_k\mathbf{y}^{(k)} \end{aligned}$$

So:

$$\begin{aligned} |\lambda_1| &= \lim_{k\to\infty}\frac{\|A^{k+1}\mathbf{y}^{(0)}\|}{\|A^k\mathbf{y}^{(0)}\|_2} \\ &= \lim_{k\to\infty}\frac{\mu_1\mu_2\ldots\mu_{k+1}\|\mathbf{y}^{(k+1)}\|}{\mu_1\mu_2\ldots\mu_k\|\mathbf{y}^{(k)}\|_2} \\ &= \lim_{k\to\infty}\mu_{k+1} \end{aligned} \tag{1.21}$$

Because μ_{k+1} converges, a good choice for a stop condition for our numerical algorithm could be

$$|\mu_k - \mu_{k-1}| \leq \text{TOL},$$

which guarantees an estimation error of at most TOL (usually TOL=10^{-5}) for the approximation of the dominant eigenvalue. With all this information, we construct algorithm 1. Just to give an understandable algorithm, we used the Euclidean norm in this algorithm.

Data:
A: a matrix
$\mathbf{y^{(0)}}$: a start vector with $\|\mathbf{y^{(0)}}\|_2 = 1$,
Tol: Tolerance for the estimation error.
Result:
$\mathbf{y^{(k)}}$: an estimation of a dominant eigenvector,
μ_k: an estimation of the dominant eigenvalue.
begin `power_method(`$\mathbf{y^{(0)}}$, k`)`
  $k = 1$;
  repeat
    $\mathbf{z^{(k)}} = A\mathbf{y^{(k-1)}}$;
    $\mu_k = \|\mathbf{z^{(k)}}\|_2$;
    $\mathbf{y^{(k)}} = \dfrac{\mathbf{z^{(k)}}}{\mu_k}$;
    $k = k + 1$
  until $k > 2$ *and* $|\mu_k - \mu_{k-1}| <$ *TOL*;
  if *the components* $\mathbf{y^{(k)}}$ *and* $\mathbf{y^{(k-1)}}$ *have a different sign* **then**
    $\mu_k = -\mu_k$;
  end
  return $\mathbf{y^{(k)}}, \mu_k$;
end

Algorithm 1: The Power method

Computational cost and usage

The computational cost of the algorithm is determined by the speed at which the $o(1)$ terms in (1.20) go to zero. This is indicated by the slowest converging term $(\lambda_2/\lambda_1)^k$. This means that the algorithm converges slowly when there is an eigenvalue close in magnitude to the dominant eigenvalue. We get the following expression for approximation μ_k of λ_1:

$$|\mu^k - \lambda_1| = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

In our algorithm we accepted an estimation error of 10^{-5}, so the number of steps n can be computed as:

$$\left|\frac{\lambda_2}{\lambda_1}\right|^n \leq \text{TOL}$$

So, for example, for $\text{TOL} = 10^{-5}$ we get $n = \frac{-5}{\log \frac{\lambda_1}{\lambda_2}}$, so:

$$\texttt{power\_method} \in O\left(\frac{-1}{\log(\lambda_1) - \log(\lambda_2)}\right)$$

Note that this O holds for any estimation error TOL of the form 10^{-e} with $e \in \mathbb{N}$.

When $\lambda_2 \approx \lambda_1$ we see that the power method (almost) needs infinitely many steps. Since we do not know the eigenvalues of A, this means we cannot know in advance whether the power method will work or not. Recall that the eigenvalues of a real matrix A are in general complex, and occur in conjugate pairs. This means, when λ_1 is not real, the power method will certainly fail. Therefore, it is a good idea to apply the power method only to matrices whose eigenvalues are known to be real. The only thing that can go wrong with those matrices is that the dominant eigenvalue has an algebraic multiplicity larger than 1.

From the Perron-Frobenius theorem in 1.2.10, we also get another good choice: namely the irreducible matrices or any matrix with strictly positive entries. Indeed, The Perron-Frobenius theorem tells us that they have a unique dominant eigenvalue.

Example

Example 1.4.8. Consider the matrix:

$$A = \begin{pmatrix} 1 & -3 & 5 \\ -1 & -7 & 11 \\ -1 & -9 & 13 \end{pmatrix}$$

A has a dominant eigenvalue 3 and a double eigenvalue 2. A corresponding dominant eigenvector is $(1, 1, 1)$. Now we use the classical power method with start vector $\mathbf{y^{(0)}} = (1, 0, 0)$, TOL$= 10^{-5}$ and we become the values in Table 1.4.8. Because we know the eigenvalues of A, we can predict the number of steps $n = \frac{-5}{\log(\frac{\lambda_2}{\lambda_2})} = \frac{-5}{\log(2/3)} \approx 28$. Because $\lambda_2/\lambda_1 = 2/3$ is not that small, the convergence here is also not that fast. We have for the approximation of a corresponding eigenvector

$$\mathbf{y^{(29)}} = (-0.577348, -0.577352, -0.577351)$$

which is (more or less) in proportion with $(1, 1, 1)$.

k	μ_k		k	μ_k
0	1.00000		15	3.00459
1	1.73205		16	3.00305
2	4.12311		17	3.00204
3	4.06564		18	3.00136
4	3.58774		19	3.00090
5	3.34047		20	3.00060
6	3.20743		21	3.00040
7	3.13055		22	3.00026
8	3.08385		23	3.00017
9	3.05457		24	3.00011
10	3.03582		25	3.00008
11	3.02363		26	3.00005
12	3.01564		27	3.00003
13	3.01037		28	3.00002
14	3.00689		29	3.00001

Table 1.1: The iteration values μ_k of Example 1.4.8.

1.5 Graphs

After introducing different kinds of matrices and proving the Perron-Frobenius theorem, we now take a closer look at graphs. Here too, we'll look at different families of graphs and prove some relevant properties about them. We also link the concept of graphs with different kinds of matrices, deepening our insight of some theorems of the previous section.

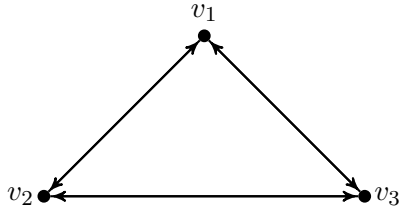
The definitions and results in this section are mainly based on the course 'Discrete Mathematics' by P. Cara [car11].

1.5.1 General definitions

Definition 1.5.1. *A **graph** is a an ordered pair $(V, \to)$ where V is a set and $\to$ is a relation. The elements of V are called **vertices** or **nodes** and $\to$ is called the **adjacency relation**. Let $u, v \in V$, then the ordered pair (u, v) belonging to $\to$ is called an **arc** or **edge** and we write $u \to v$. We also say that u is **adjacent** to v. When $v \to v$ (with $v \in V$) we say that the graph has a **loop** at v. A graph $(V, \to)$ is most of the time denoted by calligraphic letters $\mathscr{G}, \mathscr{H}, \ldots$*

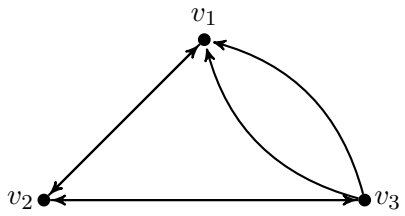
With this defintion, we defined directed graphs, when the relation $\to$ is symmetric, we call the graph **undirected**, in this case we often write $\leftrightarrow$ instead of $\to$.

Example 1.5.2. The graph



is an undirected graph with vertices v_1, v_2, v_3. The adjacency relation $\leftrightarrow$ equals $\{(v_1, v_2), (v_2, v_1), (v_2, v_3), (v_3, v_2), (v_3, v_1), (v_1, v_3)\}$.

There is a small problem with our definition, because not all graphs are taken into account, for example the graph below is not a graph following our definition because you can not define multiple edges between vertices in a relation.



Therefore we define multisets and make a remark that introduces the concept of multiplicity of an edge using these multisets. Multisets are a generalization of the notion of a set in which elements are allowed to appear more than once.

Definition 1.5.3. *A **multiset** $A = (S, \mu)$ is an ordered pair with S a set and $\mu : S \to \mathbb{N}_0$ a function that gives the **multiplicity** of an element of S. A multiset can be written as a set in the following way:*

$$A = (S, \mu) = \{(s, \mu(s)) : s \in S\}$$

*The **cardinality of a multiset** is defined as*

$$|A| = \sum_{s \in S} \mu(s).$$

Consider as an example the multiset $\{a, a, b, b, bc\}$ (with a, b, c different elements), which can be denoted as a set as $\{(a, 2), (b, 3), (c, 1)\}$

We now introduce the following important remark using multisets:

Remark 1.5.4. *Although we will stay writing a graph as $\mathscr{G} = (V, \to)$, this defintion doesn't allow for repeated edges. A graph that can have multiple edges between two vertices is often called a **multigraph**, but in this master thesis we call a multigraph just a graph. To define this in a mathematical correct way, you define $\mathscr{G}$ as an ordered pair $(V, \to)$ where V is a finite set and $\to$ is a multiset consisting of elements of the cartesian product $V \times V$.*

Definition 1.5.5. *The **neighbourhood** of a vertex v of a graph $\mathscr{G} = (V, \to)$ is the induced subgraph $\mathscr{G}_v$ with vertex set V' consisting of all vertices adjacent to v without v itself and with the multiplicity function μ', which is the restriction of μ to the vertices in V'. A vertex with a neighbourhood equal to the empty graph (a graph with an empty set of vertices) is called* isolated.

Definition 1.5.6. *Let $\mathscr{G} = (V, \to)$ be a graph with vertices $v_1, v_2, \ldots, v_n \in V$ and edges (defined as ordered pairs) $e_1, \ldots e_m \in \to$. Let $u \to v$ be an edge between $u, v \in V$. We call u the **source node** and v the **terminal node** of the edge. $s_{\mathscr{G}}(i)$ denotes the source node u_i of edge i, $t_{\mathscr{G}}(i)$ denotes the terminal node w_i of edge i.*

Definition 1.5.7. *The **order of a finite graph** $\mathscr{G}$ is the number of vertices of $\mathscr{G}$ and is denoted by $|\mathscr{G}|$.*

Definition 1.5.8. *The **indegree of a vertex** v in a graph $\mathscr{G}$ is the number of times v is a terminal node of an edge.*

Definition 1.5.9. *The **outdegree of a vertex** v in a graph $\mathscr{G}$ is the number of times v is a source node of an edge.*

Definition 1.5.10. *The **degree of a vertex** v in a graph $\mathscr{G}$ is the sum of the indegree and outdegree of vertex v.*

Definition 1.5.11. *A **walk** in a graph $\mathscr{G}$ is a sequence of vertices*

$$a_0, a_1, \ldots, a_k$$

*such that $a_{i-1} \to a_i$ for each $i \in \{1, \ldots, k\}$. The **length** of the walk is k, one less than the number of vertices.*

Definition 1.5.12. *If all edges are distinct in a walk in a graph $\mathscr{G}$, we call the walk a **path**.*

Definition 1.5.13. *A **cycle** is a walk from v_0 to v_0 in which all vertices except v_0 are distinct.*

Definition 1.5.14. *A **simple graph** is an undirected graph $\mathscr{G} = (V, \to)$ containing no loops and for all vertices $v_i, v_j \in V$, their is at most one edge.*

Definition 1.5.15. *A **clique** in a graph $\mathscr{G} = (V, \to)$ is a subset C of V, such that every two distinct vertices in C are adjacent.*

Definition 1.5.16. *A **bipartite graph** is a graph $\mathscr{G} = (V, \to)$ whose vertices can be divided into two disijount sets U and T such that every edge connects a vertex in U and a vertex V. There are no edges between vertices in U or between vertices in V.*

Product graphs

Definition 1.5.17. *Take two graphs $\mathscr{G} = (U, \to), \mathscr{H} = (V, \to')$, the **product graph** $\mathscr{G} \times \mathscr{H}$ is the graph with $|\mathscr{G}|.|\mathscr{H}|$ vertices and that has an edge between vertices (u_i, v_j) and (u_k, v_l) if there is an edge between u_i and u_k in $\mathscr{G}$ and there is an edge between v_j and v_l in $\mathscr{H}$.*

Colored graphs

Definition 1.5.18. *A **node colored graph** $\mathscr{G}$ is quadruple $(V, \to, C, a)$ with V a set of vertices, $\to$ an adjacency relation, C a set of colors and a a surjective function $a : V \to C$ that assigns to each vertex one color.*

Definition 1.5.19. *In a node colored graph $\mathscr{G} = (V, \to, C, a)$, $c_{\mathscr{G}}(V, i)$ denotes the number of vertices of color i. So:*

$$c_{\mathscr{G}}(V, i) = |\{(i, v_j) \in C \times V | a(v_j) = i\}|$$

Definition 1.5.20. *A **edge colored graph** $\mathscr{G}$ is quadruple $(V, \to, C, b)$ with V a set of vertices, $\to$ an adjacency relation, C a set of colors and b a surjective function $b : (\to) \to C$ that assigns to each edge one color.*

Definition 1.5.21. *In an edge colored graph $\mathscr{G} = (V, \to, C, b)$, $c_{\mathscr{G}}(\to, i)$ denotes the number of edges of color i. So:*

$$c_{\mathscr{G}}(\to, i) = |\{(i, e_j) \in C \times E | b(e_j) = i\}|$$

Definition 1.5.22. *A **node-edge colored graph** or a **full colored graph** $\mathscr{G}$ is 5-tuple $(V, \to, C, a, b)$ with V a set of vertices, $\to$ an adjacency relation, C a set of colors, a a function $a : V \to C$ that assigns to each vertex one color and b a function $b : (\to) \to C$ that assigns to each edge one color with as condition that $a(V) \cup b(\to) = C$.*

Adjacency matrix

We now represent a finite graph in the form of an adjacency matrix. This matrix gives a lot of useful information about the graph and vice versa.

Definition 1.5.23. *Let $\mathscr{G} = (V, \to)$ be a graph of order n and define a numbering on the vertices $v_1, \ldots, v_n$. Then the **adjacency matrix** $A_{\mathscr{G}}$ of $\mathscr{G}$ is the real $n \times n$-matrix with a_{ij} equal to $\mu(v_i, v_j)$.*

Corollary 1.5.24. *The adjacency matrix of an undirected graph $\mathscr{G} = (V, \sim)$ is a symmetric matrix.*

Proof. This is trivial by the definition of an undirected graph. □

Theorem 1.5.25. *Let $k > 0$. The element on place (i, j) in $A^k_{\mathscr{G}}$ contains the number of walks of length k from i to j in the graph $\mathscr{G} = (V, \mu)$.*

Proof. By induction on k.

For $k = 1$ we count the walks of length 1. These are edges and the result follows immediately from the definition of $A_{\mathscr{G}}$.

Let v_l be a vertex of $\mathscr{G}$. If there are b_{ij} walks of length k from i to l and a_{lj} walks of length 1 (edges) from v_l to v_j, then there are $b_{il}a_{lj}$ walks of length $k+1$ from v_i to v_j passing vertex v_l. Therefore, the number of walks of length $k + 1$ between v_i and v_j is equal to:

$$\sum_{l \in V} b_{il}a_{lj} =: c_{ij}.$$

By the induction hypothesis we now that b_{il} equals the element on place (i, l) in $A^k_{\mathscr{G}}$ so c_{ij} is exactly the element on place (i, j) in the matrix product

$$A^k_{\mathscr{G}} A_{\mathscr{G}} = A^{k+1}_{\mathscr{G}}.$$

□

Example 1.5.26. The adjacency matrix of the graph in Example 1.5.2 is:

$$A = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Incidence matrix

A graph can also be represented as an incidence matrix by numbering the vertices and the edges. The resulting incidence matrix will be a with only 1, -1 and 0 entries.

Definition 1.5.27. *The **incidence matrix** of directed graph $\mathscr{G} = (V, \rightarrow)$ with vertices $v_1, \ldots, v_n \in V$ and edges $e_1, \ldots, e_m \in E$ is a $n \times m$-matrix A where rows represent the vertices and the columns represent the edges, such that:*

$$(A)_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is the source node of } e_j \\ -1 & \text{if } v_i \text{ is the terminal of } e_j \\ 0 & \text{otherwise} \end{cases}$$

In the case of an undirected graph $\mathscr{G} = (V, \leftrightarrow)$ with vertices $v_1, \ldots, v_n \in V$ and edges $e_1, \ldots, e_m \in E$ we define:

$$(A)_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is a node of } e_j \\ 0 & \text{otherwise} \end{cases}$$

1.5.2 Strong connectivity

In this section, we take a closer look at *directed graphs* and introduce the concept of connectivity.

Definition 1.5.28. *An undirected graph $\mathscr{G} = (V, \leftrightarrow)$ is **connected** if it possible to establish a path from any vertex to any other vertex.*

Definition 1.5.29. *A directed graph $\mathscr{G}(V, \rightarrow)$ is **connected** if the underlying undirected graph (remove all arrows on the edges) is connected, the directed graph $\mathscr{G}$ is **strongly connected** if there is a path in each direction between each pair of vertices of the graph.*

In the next proof, we study the equivalence of the matrix property of irreducibility of Definition 1.1.10 with the concept of the strongly connected directed graphs of a matrix:

Theorem 1.5.30. *Let $\mathscr{G}$ be a (directed) graph with adjacency matrix A. Then $\mathscr{G}$ is strongly connected if and only if A is irreducible.*

Proof. From Theorem 1.5.25 we know that a graph is strongly connected if and only if for every pair of indices i and j there is an integer k such that $(A^k)_{ij} > 0$, from Theorem 1.1.12 we know this means that A is irreducible and vice versa.

□

1.6 Hypergraphs

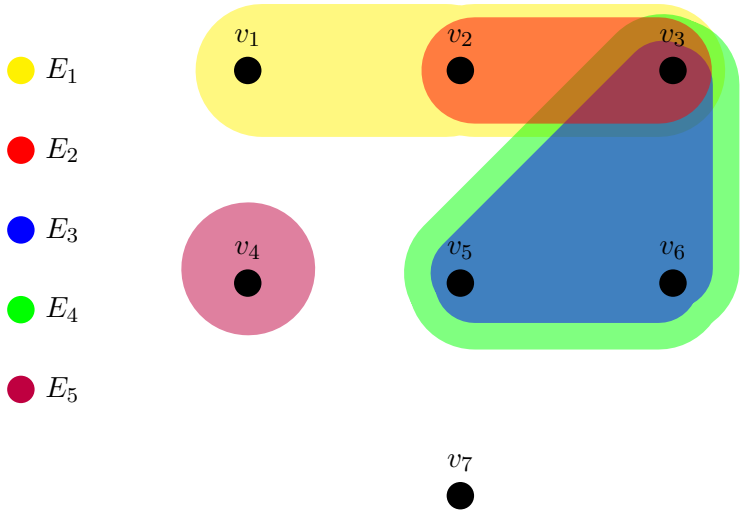
After introducing graphs, we now look at hypergraphs. Intuitively, a hypergraph is a generalization of a graph in which an edge can connect any number of vertices. The definitions presented are mainly from [Ber85] and [PZ14].

1.6.1 General definitions

Definition 1.6.1. *A **hypergraph** is an ordered pair (V, E) with V a finite set of and $E \subseteq 2^V \setminus \emptyset$, the power set of V with $E \neq \emptyset$. The elements of V are called the **vertices** and the elements of E are called the **edges**. An hypergraph (V, E) will be denoted by the calligraphic letters $\mathcal{G}, \mathcal{H}, \ldots$.*

Remark 1.6.2. *As in graphs, the classic definitions of a hypergraph doesn't allow to have multiple edges that cover the same vertices. Also repeated vertices within an edge, often called **hyperloops** are not allowed by the above definition. A hypergraph that can have multiple edges connecting the same vertices, and hyperloops is called a **multi-hypergraph**, but in this master thesis we call a multi-hypergraph just a hypergraph. So a hypergraph $\mathcal{G}$ is an ordered pair (V, E) where V a finite set and E is a multi-set consisting of multi-subsets of V (a multi-subset is just a subset of a set where the elements are allowed to appear more than once).*

Example 1.6.3. The hypergraph $\mathcal{G} = (V, E)$ consists of 7 vertices and 4 edges. The edges are equal to the multiset $\{\{v_1, v_2, v_3\}, \{v_2, v_3\}, \{v_3, v_5, v_6\}, \{v_3, v_5, v_6\}, \{v_4\}\}$. Note that the colors don't have any meaning to the hypergraph (the hypergraph is not an edge colored hypergraph), but only serve to clarify the drawing.



Definition 1.6.4. *In a hypergraph $\mathcal{G} = (V, E)$, two vertices $v_i, v_j \in V$ are called **adjacent** if there is an edge $E_i \in E$ that contains both vertices. Two edges E_k, E_l are called adjacent if their intersection is not empty.*

Definition 1.6.5. *The **order of a finite hypergraph** $\mathcal{G}$ is the number of vertices of $\mathcal{G}$ and is denoted by $|\mathcal{G}|$.*

We now define the degree of a vertex in a hypergraph. Note that we don't define the indegree and outdegree of a vertex as the hypergraphs we consider are undirected.

Definition 1.6.6. *The **degree of a vertex** v in a hypergraph $\mathcal{G}$ is the number of times v is contained in an edge.*

Definition 1.6.7. *A **path** in an hypergraph $\mathcal{G} = (V, E)$ is a sequence $p = (a_0, A_1, a_1, \ldots, A_k, a_k)$, $k \geq 1$ where the a_i's are pairwise distinct vertices, the A_i's are pairwise distinct edges and $a_{i-1}, a_i \in A_i$ for $1 \leq i \leq k$. The path p is said to **join** a_0 and a_k. The **length** of the path is k.*

Definition 1.6.8. *A hypergraph is **connected** if for each vertex there is a path to any other vertex.*

k-hypergraphs

In most applications, the edges of an hypergraph connect a fixed number of vertices.

Definition 1.6.9. *An hypergraph is called a k-**uniform hypergraph** for $k \geq 2 \in \mathbb{N}$ if for all $E_i \in E$, the cardinality $|E_i|$ is equal to k. The cardinality of multisets is defined in Definition 1.5.3. The term k-**graph** is often used instead of a k-uniform hypergraph. The edges in a k-graph are sometimes called k-edges.*

Notice that the 2-hypergraphs are just the undirected graphs we defined in the previous section.

Directed hypergraphs

An important difference with the previous section is that all hypergraphs we have defined are **undirected**: there is no specific order in which an edge connect different vertices. In a graph, directed edges arise naturally as some vertex can be the source node and some vertex can be the terminal node, no other nodes are connected by an edge of a graph. For edges of hypergraphs this concept is not straightforward to generalize: one option is to see edges as paths connecting vertices in a specific order, another option is to partition the vertices connected by an edge in a set of source nodes and a set of terminal nodes. This last notion is studied in [GLPN93]. We will not discuss this topic in detail, all the hypergraphs in this master thesis are undirected.

1.6.2 Incidence matrix

A hypergraph can be represented as an incidence matrix by numbering the vertices and the edges. The resulting incidence matrix will be a boolean matrix with only 1 and 0 entries. An alternative way to represent a hypergraph is by using adjacency tensors.

Definition 1.6.10. *The **incidence matrix** of a hypergraph $\mathcal{G} = (V, E)$ with vertices $v_1, \ldots, v_n \in V$ and edges $e_1, \ldots, e_m \in E$ is a $n \times m$-matrix A where rows represent the vertices and the columns represent the edges, such that:*

$$(A)_{ij} = \begin{cases} 1 & \text{if } v_i \in e_j \\ 0 & \text{if } v_i \notin e_j \end{cases}$$

Example 1.6.11. The corresponding incidence matrix of the hypergraph of Example 1.6.3 is:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Chapter 2

Similarity on graphs

In the previous chapter all the basic terminology and results were introduced, now we take an extensive look at the concept of similarity on graphs. Similarity on graphs is a fairly new concept to compare the nodes of two graphs. The concept arose from the research on algorithms for web searching engines (like *Google*, *Yahoo*,...) in the late nineties. More specifically, Jon M. Kleinberg introduced in his paper 'Authoritative Sources in a Hyperlinked Environment' [Kle99] the famous 'HITS algorithm' for extracting information from the link structure of websites. The method leads to an iterative algorithm where graphs represent the link structure of a collection of websites on a specific topic. Because this paper formed the basis of later research on similarity on graphs, the whole idea and algorithm of Kleinberg is introduced in the first section of this chapter. In 2004, V.D. Blondel et al. [BGH$^+$04] generalized the algorithm of Kleinberg, introducing the notion of similarity on directed graphs. This similarity is covered in the second section. With this similarity on directed graphs, there is a much wider scope of applications than just search algorithms. Next, we extend the notion of similarity on directed graphs: the method of Blondel only returns the concept of node similarity which is in fact a measurement of how similar two nodes of two graphs are to each other.

2.1 The HITS algorithm of Kleinberg

2.1.1 History

Back in the nineties, internet became more and more popular by the public. The popular search engines back then where Altavista and Yahoo, but they weren't as advanced as search engines today. The main pitfall of the first search engines was that the search results were purely based on the number of occurrences of a word in a webpage. This was a pitfall for many reasons. The first reason was the growing popularity of the internet: as more and more webpages were put online, simply getting the relevant pages to a search query in this text-based manner, was a process that could possibly return millions of relevant pages. Also *content similarity* was an issue: a website owner can easily cheat in a text-based search system by just adding and repeating some very popular search words, making his website appear in the results of a large number of search queries. Two possible solutions were simultaneously invented in 1997 and 1998. The first one was the *PageRank*-system developed by Larry Page and Sergey Brin ([BP98]). The PageRank system led to the foundation of the immensely popular Google search engine. Meanwhile, also Jon Kleinberg came up with his own solution,

the *HITS algorithm* (hyperlink-induced topic search). At that time, he was both a professor in the Computer Science Department at the Cornell University and researcher for IBM. The algorithm is used inter alia today by the Ask search engine (www.ask.com). Both these algorithms use the hyperlinks between webpages to rank search results. Because this master thesis is about similarity and this concept is introduced on graphs as a generalization of the HITS algorithm, we don't go to into further detail about the PageRank-algorithm. In the following paragraphs, the HITS algorithm is extensively explained.

2.1.2 Motivation

Kleinberg's work originates in the problems that arise with text-based searching the WWW. Text-based searching just counts all the occurrences of a given search query on webpages and returns a set of webpages ordered by decreasing occurency. When a user supplies a search query, we probably face an *abundance problem* with this method: the number of pages that could reasonably be returned as relevant is far too large for a human user to digest. To provide effective search results under these conditions, we need to filter the 'authoritative' ones. We face some complications when we want to filter the 'authoritative' webpages in a text-based system. For example, if we search for '`job offers in Flanders` ' the most authoritative page and expected first result in a search engine would be `www.vdab.be`. Unfortunately, the query '`job offers`' is used in over a million pages on the internet and `www.vdab.be` is not the one using the term most often. Therefore, there is no way to favor `www.vdab.be` in a text-based ranking function. This a recurring phenomenon, as another example if you search for the query '`computer brands`', there is no reason at all to be sure that the website of Apple or Toshiba even contain this search term.

The HITS algorithm solves these difficulties by analyzing the hyperlink structure among webpages. The idea is that hyperlinks encode a sort of human judgment and that this judgement is crucial to formulate a notion of authority. Specifically, when a page p includes a link to page q, it means that p gives a *conferred authority* on q. Again we face difficulties, because this conferred authority doesn't hold for every link. Links are created for a wide variety of reasons, for example, a large number of links are created for navigation within a website (e.g. "Return to homepage") and these have of course nothing to do with a notion of authority.

The HITS method is based on the relationship between the *authorities* for a topic and those pages that link to many related authorities, called *hubs*. Page p is called an *authority* for the query "`smartphone brand`" if it contains valuable information on the subject. In our example websites of smartphone manufacturers such as "`www.apple.com`", "`www.samsung.com`",... would be good authorities for this search query and these are the results a user expect from a search engine.

A hub is a second category of pages needed to find good authorities. Their role is to advertise authoritative pages. Hubs contain useful links toward these authorities. In our example, consumer websites with reviews on smartphones, websites of smartphone shops,... would be good hubs. In fact, hubs point the search process in the 'right direction'.

To really grasp the idea, we make an analogy with everyday life. If you tell a friend that you think of buying a new smartphone, he might tell you his experiences with smartphones and he will probably share some opinions he got from other friends. He might suggest you some good models and good brands. Now, you are more inclined to buy a smartphone that your friend suggested. Well, this idea is used in the HITS-method: your friend served as hub,

Data:
σ: a query string.
$\mathcal{E}$: a text-based search engine.
t: natural number (usually initiated to 200)
d: natural number (usually initiated to 50).
Result: A page set S_σ satisfying all the properties of our wish list.
begin `create_graph(`σ, $\mathcal{E}$, t, d`)`
  Let R_σ denote the top t results of $\mathcal{E}$ on σ;
  Set $S_\sigma := R_\sigma$;
  for *each page* $p \in R_\sigma$ **do**
    Let $\Gamma^+(p)$ denote the set of all pages p points to;
    Let $\Gamma^-(p)$ denote the set of all pages pointing to p;
    Add all pages in $\Gamma^+(p)$ to S_σ;
    if $|\Gamma^-(p)| \leq d$ **then**
      Add all pages in $\Gamma^-(p)$ to S_σ;
    else
      Add an arbitrary set of d pages from $\Gamma^-(p)$ to S_σ;
    end
  end
  return S_σ;
end

Algorithm 2: Algorithm to construct S_σ.

the brands and models he suggested are good authorities.

2.1.3 Constructing relevant graphs of webpages

Any collection of hyperlinked pages can be transformed to a directed graph $\mathscr{G} = (V, \to)$: the nodes correspond to the pages, and if there is a link from page p to page q, there is an arc $p \to q$. Suppose a search query is performed, specified by a query σ. We wish to determine the authoritative pages by an analysis of the link structure. But first we have to construct a subgraph of the internet on which our algorithm will operate. We want to make the computational effort as efficient as possible, so we restrict the subgraph to the set Q_σ of all pages where the query σ occurs. For this, we could use any already existing text-based search engine. But, for our algorithm Q_σ is possibly much too big: it may contain millions of pages making it impossible for any computer to preform the algorithm. Moreover it is, as explained in the motivation in 2.1.2, possible that Q_σ does not contain some of the most important authorities because they never use the query string σ on their website.

Therefore, we wish to transform the set Q_σ to a set S_σ of pages following this 'wish list' of properties:

1. S_σ is relatively small,
2. S_σ is rich in relevant pages,
3. S_σ contains most of the strongest authorities.

By keeping S_σ small, the computational cost of preforming non-trivial algorithms can be kept under control. By the property of being rich in relevant pages, it will be easier to find good authorities.

To construct S_σ, we first construct a *root set* R_σ with the t highest-ranked pages for σ using a text-based search engine (they sort results based on the occurence of σ). Typically, t is set about 200. R_σ complies with properties 1 and 2 of our wish list, but because $R_\sigma \subset Q_\sigma$, it may fail from satisfying property 3. Now we use the root set R_σ to create the set S_σ satisfying our complete wish list. When a strong authority is not in R_σ, it is very likely that at least one of the pages in R_σ points to this authority. Hence, by using the pages in R_σ, we can expand it to S_σ by looking at the links that enter and leave R_σ. We get algorithm 2.

Thus, we obtain S_σ by expanding R_σ to include any page pointed to by a page in R_σ. We also add d pages that point to a page in R_σ. d is usually initiated to 50. The parameter d is crucial to stay in accordance with property 1 of our wish list. Indeed, a webpage can be pointed to by several thousands and thousands of other webpages, and we don't want to include them all if we want to keep S_σ relatively small. Some experiments in [Kle99] showed that this algorithm resulted in a S_σ with a size in the range of 1000 to 5000 web pages. Property 3 of our wish list is usually met because a strong authority need only be reference once in the t pages of the root set R_σ to be added to S_σ.

Denote the resulting graph of the page set S_σ by $\mathscr{G}[S_\sigma]$. Note that $\mathscr{G}[S_\sigma]$ will contain a lot of links serving only navigational purposes within a website. As mentioned before, these links have nothing to do with the the notion of authority and they must be removed from our final graph if we want a good determination of the authoritative pages by an analysis of the link structure. A very simple heuristic can be used to derive a subgraph of $\mathscr{G}[S_\sigma]$ leaving out all the navigational links: we make a distinction between *transverse* links and *intrinsic* links. Transverse links are links between different domain names (e.g. a link between `www.vub.ac.be` and `www.ua.ac.be`) and intrinsic links are links between the same domain name (e.g. a link between `www.vub.ac.be` and `dwis.vub.ac.be`). Intrinsic links exist to allow navigation within a website and they tell us very little about the authority of the pages they point to. Therefore, we delete all intrinsic links from $\mathscr{G}[S_\sigma]$, keeping only the arcs corresponding to transverse links.

Our graph still contains some meaningless links in the context of page authority. Suppose a large number of pages from the same domain name have a transverse link to the same page p. Most of the time, this means a form of advertisement (by example 'Website created by. . . .' at the bottom of each page). It is useful to only allow m pages (m is usually initiated to 6) from the same domain name to have a transverse link to the same page. If m is exceeded, all the transverse links must be deleted from the graph. Note, however, that not all links to advertisements will be erased because on most web pages, advertisements change on every page which avoids the exceedance of m.

Applying the two described heuristics above on $\mathscr{G}[S_\sigma]$, we get a new graph $\mathscr{G}'_\sigma$ which is exactly what we need to preform our link analysis.

2.1.4 Hubs and Authorities

A very simple approach would now be to order the pages in $\mathscr{G}'_\sigma$ by their indegree. Although this approach can sometimes return good search results, this heuristic is often too simple because S_σ will probably contain some web pages with a lot of incoming links without being very relevant to the search query σ (e.g. advertisements). With these incoming links, those

web pages are ranked high in the final search result, which we want to avoid.

Do we have to return to a text-based approach to avoid irrelevant web pages being on top of the search results? No, the link structure of $\mathscr{G}'_\sigma$ can tell us a lot more than it may seem at first glance. Authoritative pages relevant to query σ should indeed have a large in-degree, but there should also be a considerable overlap in the sets of pages that point to authoritative pages. This set of pages that point to authoritative pages are called *hubs*. Hubs have links to several authoritative pages and they sort of "concentrate" all the authorities on query σ. Figure 2.1.1 shows what this means conceptually.

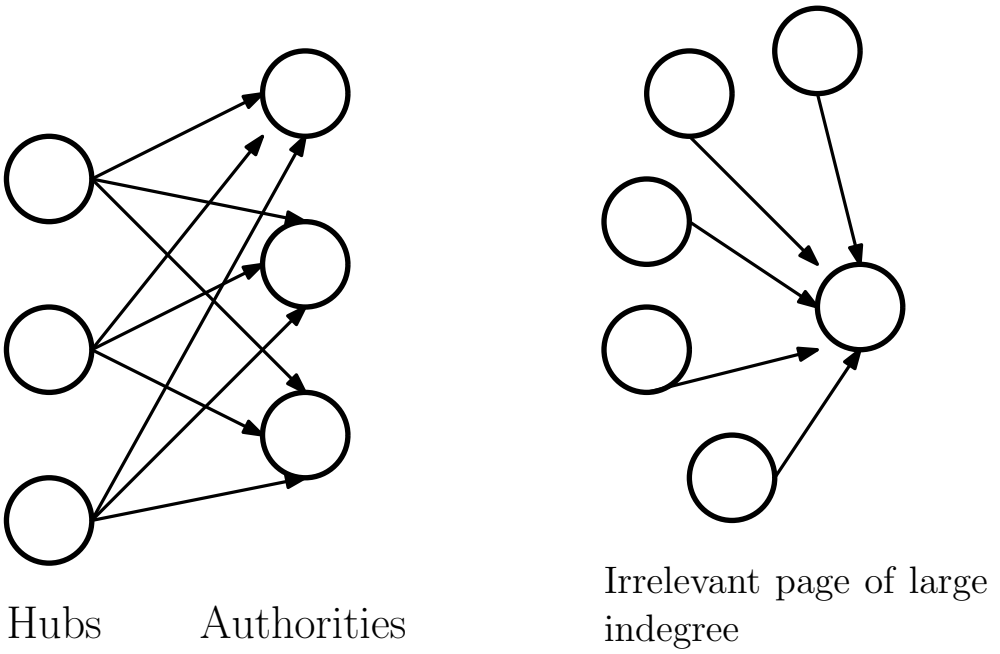


Figure 2.1.1: The concept of hubs and authorities

So, for each page j we assign two scores, an *authority score* which estimates the value of the content of the page and a *hub score* which estimates the value of the outgoing links to other pages. We get now a *mutually reinforcing relation*: a good hub is a page pointing to many good authorities, a good authority is a page that is pointed to by many good hubs. This leads us to a *mutually reinforcing relation* resulting in an iterative method to break this circularity.

So let $\mathscr{G}'_\sigma = (V, \to)$ and let h_j and a_j be the hub and authority scores of vertex v_j (corresponding with page j). These scores must be initialized by some positive start values and then updated simultaneously for all vertices. This leads to a *mutually reinforcing relation* in which the hub score of v_j is set equal to the sum of the authority scores of all vertices pointed to by v_j and in an equal manner the authority score of v_j is set equal to the sum of the hub scores of all vertices pointing to v_j.

$$\begin{cases} h_j := \sum_{i:(v_j,v_i)\in\to} a_i, \\ a_j := \sum_{i:(v_i,v_j)\in\to} h_i. \end{cases}$$

The basic operations in which hubs and authorities reinforce one another are depicted in Figure 2.1.2.

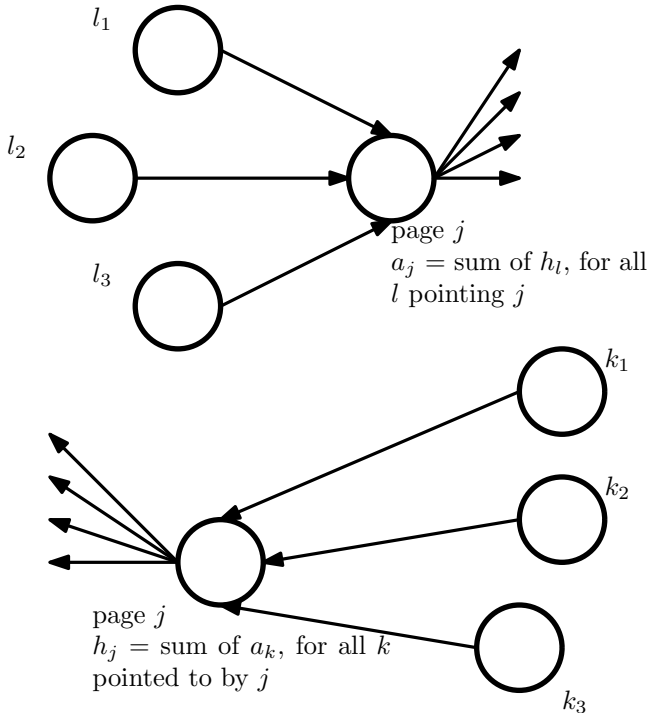


Figure 2.1.2: The basic operations in the reinforcing relation between hubs and authorities

Let B be the adjacency matrix of $\mathscr{G}'_\sigma$ and denote $\mathbf{a}$ as the authority vector with coordinates $(a_1, a_2, \ldots, a_n)$ (with $n = |\mathscr{G}'_\sigma|$, the number of pages) and $\mathbf{h}$ as the hub vector. The mutually reinforcing relation can now be rewritten as:

$$\begin{pmatrix} \mathbf{h} \\ \mathbf{a} \end{pmatrix}^{(k+1)} = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{h} \\ \mathbf{a} \end{pmatrix}^{(k)}, \quad k = 0, 1, \ldots,$$

In compact form, we denote

$$\mathbf{x}^{(k+1)} = M\mathbf{x}^{(k)}, \quad k = 0, 1, \ldots, \tag{2.1}$$

where

$$\mathbf{x}^{(k)} = \begin{pmatrix} \mathbf{h} \\ \mathbf{a} \end{pmatrix}^{(k)}, M = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix}$$

After each iteration, we have to normalize h_j and a_j. Indeed, we want to get the authority and hub weights for each page and in order to compare these after each iteration step, they must be normalized because only the relative differences do matter, otherwise the whole procedure would be meaningless. Pages with larger a_j-scores are viewed as being better authorities, pages with larger h_j-scores are better hubs.

We get the following sequence (with $z^{(0)}$ some positive start value) of normalized vectors:

$$\mathbf{z}^{(0)} = \mathbf{x}^{(0)} > 0, \mathbf{z}^{(k+1)} = \frac{M\mathbf{z}^{(k)}}{||M\mathbf{z}^{(k)}||_2}, \quad k = 0, 1, \ldots, \tag{2.2}$$

Data:
$\mathscr{G}$: a graph of n linked pages.
k: natural number.
Result: A vector $(\mathbf{h}, \mathbf{a})$ containing the hub and authority scores after k steps.
begin `hits(`$\mathscr{G}$`, k)`
  Set $\mathbf{a}^{(0)} = (1, 1, \ldots, 1) \in \mathbb{R}^n$;
  Set $\mathbf{h}^{(0)} = (1, 1, \ldots, 1) \in \mathbb{R}^n$;
  for $i = 1, 2, \ldots, k$ **do**
    Calculate
    $\mathbf{h}'^{(i)} = \left(\sum_{m:(v_1,v_m)\in\to} \mathbf{a}_m^{(i-1)}, \sum_{m:(v_2,v_m)\in\to} \mathbf{a}_m^{(i-1)}, \ldots, \sum_{m:(v_n,v_m)\in\to} \mathbf{a}_m^{(i-1)}\right)$;
    Normalize $\mathbf{h}'^{(i)}$ obtaining $\mathbf{h}^{(i)}$;
    Calculate $\mathbf{a}'^{(i)} = \left(\sum_{m:(v_m,v_1)\in\to} \mathbf{h}_m^{(i)}, \sum_{m:(v_m,v_2)\in\to} \mathbf{h}_m^{(i)}, \ldots, \sum_{m:(v_m,v_n)\in\to} \mathbf{h}_m^{(i)}\right)$;
    Normalize $\mathbf{a}'^{(i)}$ obtaining $\mathbf{a}^{(i)}$;
  end
  return $(\mathbf{h}^{(k)}, \mathbf{a}^{(k)})$;
end

Algorithm 3: The iterative HITS-algorithm.

How do we decide on $\mathbf{x}^{(0)}$? We will see that any positive vector in $\mathbb{R}^{2n}$ is a good choice, but for the sake of simplicity, we make the natural choice[1] $\mathbf{1} \in \mathbb{R}^{2n}$. The limit to which the sequence converges results in 'definitive' hub and authority scores for each page in the graph $\mathscr{G}'_\sigma$.

To compute the iterative algorithm, we update the hub and authority scores in an alternating form (by each step we have to normalize the scores). Because we will prove that the sequence converges, theoretically we can keep on iterating until a fixed point is approximated. But in most practical settings, we choose a fixed number of steps k to reduce the computational cost because we can not know beforehand how large k has to be to reach the limit. But of course, it is extremely important to know that method converges anyway. Let $\mathbf{x}^{(i)}$ denote vector $\mathbf{x}$ at iteration step i as in Notation 1.4.7, and we get Algorithm 3.

To filter the top c hubs and the top c authorities, you can use the trivial Algorithm 4.

How do we decide on the values of k and c? It's immediately clear that c and k must be proportional: for low c values, a lower value for the number of iteration steps k is appropriate and vice versa. Experiments in [Kle99] showed that k set to 20 is sufficient to become stable for finding the 5 best hubs and authorities, thus for $c = 5$.

2.1.5 Convergence of the algorithm

We now want to prove that for arbitrarily large values of k, the sequence $Z^{(k)}$ converge to a limit $(\mathbf{h}', \mathbf{a}')$. Before prove the convergence, note that adjacency matrices are nonnegative by definition, and thus the matrix M is nonnegative too. M is also clearly a symmetric $n' \times n'$-matrix with nonnegative, real entries. We prove that such matrices have n' (not necessarily different) real eigenvalues and that we can diagonalize M. This is the first condition of the power method iwe introduced in section 1.4.2. If we can also prove the second condition

[1] $\mathbf{1}$ is a matrix, or vector, whose entries are all equal to 1.

Data:
$\mathscr{G}$: a graph of n linked pages.
k: natural number.
c: natural number.
Result: A vector $((\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_c), (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_c))$ containing exactly the nodes of the c top hubs and c top authorities.
begin `filter(`$\mathscr{G}$`, `k`, `c`)`
  $(\mathbf{h}, \mathbf{a}) =$ `hits(`$\mathscr{G}$`, `k`)`;
  Sort the pages with the c largest values in $\mathbf{h}$, resulting in a vector of nodes $(\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_c)$;
  Sort the pages with the c largest values in $\mathbf{a}$, resulting in a vector of nodes $(\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_c)$;
  return $((\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_c), (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_c))$;
end

Algorithm 4: Returning the top c hubs and authorities

(having a unique dominant eigenvalue), convergence is immediately shown by the power method.

However, there is a problem here: we can not prove that nonnegative symmetric matrices have a unique dominant eigenvalue (a unique dominant eigenvalue means the largest eigenvalue with multiplicity 1), simply because this is not true in general.[2] In the original paper of Kleinberg [Kle99] he solves this issue by simply imposing that the matrix M has a unique dominant eigenvalue and he doesn't pay any further attention to this problem. He presents it as 'a small, technical assumption for the sake of simplicity'.

Is this justified in practice? Actually it is, because you can prove with probability theory that a random matrix C_n, with a probability tending to 1, has no repeated eigenvalues as the size of the matrix goes to infinity (See for example Thereom 2.2.3 in [DG09]). You can also defend this differently: the only reason why we can't use the Perron-Frobenius theorem (see 1.2.10) here, is because M will have zero entries (not all pages in S_σ will be linked to each other, the graph $\mathscr{G}'_\sigma$ is not strongly connected in general). But, it is intuitively clear that by adding 1 to each entry of M, the final results of the algorithm (a sorted vector with the best hubs and authorities) will not be changed at all, because pages with larger indegrees and outdegrees will continue to get better hub and authority scores (note, however, that the relative hub and authority scores can fluctuate a bit and the algorithm will converge slower because of the lack of zero entries). So, by adding 1 to each entry of M, the matrix becomes a positive, real matrix and we know from the Perron-Frobenius that these matrices have a unique dominant eigenvalue. So yes, the 'small, technical assumption' in the paper of Kleinberg is justified.

Now that this problem is solved, we present the relevant theorems below. We also impose on the matrix M that it has a unique dominant eigenvalue with the preceding explanations in mind. Remember that we will generalize the idea of the HITS algorithm to introduce similarity on graphs. Therefore, we will reconsider the convergence of the (generalized) algorithm again in the following section, and there we prove that there exists also a limit even when the

[2]The matrix $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ is a simple counterexample of a symmetric, nonnegative, real matrix that has no unique dominant eigenvalue.

matrix M has no unique dominant eigenvalue. The reason why we don't present this result immediately, is because we want to present the results as authentic possible and we want to show the evolution of the ideas in the successive papers.

Theorem 2.1.1. *If A is a symmetric, real $n \times n$-matrix, then it has n (not necessarily different) real eigenvalues corresponding to real eigenvectors.*

Proof. First, threat A as complex matrix. The characteristic polynomial $\det(A - \lambda I)$ has n roots in $\mathbb{C}$ and each root is an eigenvalue for A. Let $\lambda \in \mathbb{C}$ be any eigenvalue and $\mathbf{v} \in \mathbb{C}^n$ be a corresponding eigenvector for A. We have:

$$A\mathbf{v} = \lambda\mathbf{v}.$$

As $A = A^t$, we also get:

$$\mathbf{v}^t A = \lambda\mathbf{v}^t.$$

Taking the complex conjugate of both sides we get (A is a real matrix):

$$\bar{\mathbf{v}}^t A = \bar{\lambda}\bar{\mathbf{v}}^t$$

We get:

$$\bar{\mathbf{v}}^t A\mathbf{v} = (\bar{\mathbf{v}}^t A)\mathbf{v} = (\bar{\lambda}\bar{\mathbf{v}}^t)\mathbf{v} = \bar{\lambda}\bar{\mathbf{v}}^t\mathbf{v}.$$

We also have:

$$\bar{\mathbf{v}}^t A\mathbf{v} = \bar{\mathbf{v}}^t(A\mathbf{v}) = \lambda\bar{\mathbf{v}}^t\mathbf{v}.$$

Hence:

$$\bar{\lambda}\bar{\mathbf{v}}^t\mathbf{v} = \lambda\bar{\mathbf{v}}^t\mathbf{v}.$$

We conclude that $\lambda = \bar{\lambda}$ for $\mathbf{v} \neq 0$. We proved that every eigenvalue of A is real. If λ is an eigenvalue of A, then the matrix $(A - \lambda I)$ is not invertible so a vector $\mathbf{s} \in \mathbb{R}^n$ exists with

$$(A - \lambda I)\mathbf{s} = 0,$$

proving that also the corresponding eigenvector is real.

□

Theorem 2.1.2. ***(Symmetric Schur Decomposition)*** *Let A be a real symmetric matrix, then there exist an orthogonal matrix P such that:*

(i) $P^{-1}AP = D$, a diagonal matrix,

(ii) The diagonal entries of D are the eigenvalues of A,

(iii) The column vectors of P are the eigenvectors of the eigenvalues of A.

Proof. By induction on the order of the matrix. For $n = 1$ the theorem is trivial. Let A be a symmetric $n \times n$-matrix. A has at least one eigenvalue λ_1 by the previous theorem. Let $\mathbf{x_1}$ be a corresponding eigenvalue with $\|\mathbf{x}_1\| = 1$ and $A\mathbf{x}_1 = \lambda_1\mathbf{x}_1$. By the Gram-Schmidt procedure, we construct an orthonormal basis $V_1 = \{\mathbf{x}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ of $\mathbb{R}^n$. Let:

$$S_1 = [\mathbf{x}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n],$$

since S_1 is orthonormal, we get $S_1^t = S^{-1}$. Consider the matrix: $S_1^{-1}AS_1$. We have:

$$(S_1^{-1}AS_1)^t = (S_1^tAS_1)^t = S_1^tA^tS_1 = S_1^{-1}AS_1$$

Thus $S_1^{-1}AS_1$ is a symmetric matrix. Since $S_1\mathbf{e}_1 = \mathbf{x}_1$, we get:

$$\begin{aligned} S_1^{-1}AS_1\mathbf{e}_1 &= (S_1^{-1}A)(\mathbf{x}_1) \\ &= S_1^{-1}(\lambda_1\mathbf{x}_1) \\ &= \lambda_1(S_1^{-1}\mathbf{x}_1) \\ &= \lambda_1\mathbf{e}_1 \end{aligned}$$

So we get:

$$S_1^{-1}AS_1 = \left(\begin{array}{c|c} \lambda_1 & \mathbf{0} \\ \hline \mathbf{0}^t & A_1 \end{array}\right),$$

with $\mathbf{0}$ a vector of zero entries of size $n-1$ and A_1 an $(n-1)\times(n-1)$ symmetric matrix. We know by induction that there exist a $(n-1)\times(n-1)$ orthogonal matrix S_2 such that $S_2^{-1}A_1S_2 = D'$ with D' an $(n-1)\times(n-1)$ diagonal matrix. Let:

$$S_2' = \left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2 \end{array}\right),$$

and also S_2' is an orthogonal matrix, we get:

$$\begin{aligned} \left(S_2'\right)^{-1} S_1^{-1}AS_1S_2' &= \left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2^t \end{array}\right)\left(S_1^{-1}AS_1\right)\left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2^t \end{array}\right)\left(\begin{array}{c|c} \lambda_1 & \mathbf{0} \\ \hline \mathbf{0}^t & A_1 \end{array}\right)\left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} \lambda_1 & \mathbf{0} \\ \hline \mathbf{0}^t & S_2^tA_1S_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} \lambda_1 & \mathbf{0} \\ \hline \mathbf{0}^t & D' \end{array}\right) \end{aligned}$$

Thus, if we put

$$\begin{aligned} P &= S_1S_2' \\ D &= \left(\begin{array}{c|c} \lambda_1 & \mathbf{0} \\ \hline \mathbf{0}^t & D' \end{array}\right), \end{aligned}$$

we have proved (1). From the definition of diagonalizable matrices and the fact that a square matrix is diagonalizable if and only if it has an eigenbasis (a basis containing only lineair independent eigenvectors), (ii) and (iii) immediately follow. □

Theorem 2.1.3. *Giving a graph $\mathscr{G}$ with n linked pages, the sequence as defined in the previous paragraph:*

$$\mathbf{z}^{(0)} = \mathbf{1} \in \mathbb{R}^n, \mathbf{z}^{(k+1)} = \frac{M\mathbf{z}^{(k)}}{||M\mathbf{z}^{(k)}||_2}, \quad k = 0, 1, \ldots,$$

converges when M has a unique dominant eigenvalue.

Proof. Since 1) M is diagonalizable as symmetric matrix by Theorem 2.1.2 and 2) M has a unique dominant eigenvalue, it follows from the power method that the sequence will converge to a corresponding dominating eigenvector $(\mathbf{h}', \mathbf{a}')$. This eigenvector contains the hub and authority scores. □

We conclude with a nice corollary.

Corollary 2.1.4. *The second power of the matrix M has the form:*

$$M^2 = \begin{pmatrix} BB^T & 0 \\ 0 & B^TB \end{pmatrix},$$

and the normalized hub and authority scores are given by the dominant eigenvectors of BB^T and B^TB.

Proof. By the compact form given in equation 2.1, we see that $\mathbf{h_k} \leftarrow (BB^T)^{k-1}B\mathbf{a_0}$ and $\mathbf{a_k} \leftarrow (B^TB)^k\mathbf{a_0}$. Let $\mathbf{a_0}$ be $\mathbf{1} \in \mathbb{R}^n$. From the previous theorem we also know that:

$$\lim_{k\to\infty} \mathbf{h_k} = \mathbf{h} \quad \text{and} \quad \lim_{k\to\infty} \mathbf{a_k} = \mathbf{a},$$

and also from the previous proof we know that $(\mathbf{h}, \mathbf{a})$ is the dominant eigenvector of M. It follows immediately that also $\mathbf{h}$ is the dominant eigenvector of BB^T and $\mathbf{a}$ is the dominant eigenvector of B^TB. □

2.1.6 Examples

Searching for math professors at the VUB

Example 2.1.5. We conclude this section with a fictitious example of the HITS-algorithm. Suppose you are looking for `math professors vub` with a text-based search engine and you get the following results:

- The website of the mathematics department of the VUB,
- The website of the faculty of science of the VUB,
- The websites of 4 math professors,
- The website of 10 PhD students at the the mathematics department of the VUB.

Lets take a look at the link structure of these web pages (remember that it is a fictitious example):

- The website of the mathematics department at the VUB links to the websites of all the 4 professors, the 10 PhD students and the faculty of science,
- The website of the faculty of science of the VUB links to the websites of all the 4 math professors and the mathematics department,
- The websites of the 4 math professors link to the website of the Mathematics department and the faculty of science,

- The websites of the 10 PhD students at the the VUB link to the the website of their promotor. 1 professor has 4 PhD students, the other 3 professors have 2 PhD students.

We can now construct the graph $\mathscr{G}_\sigma$ (of course this graph is not completely made according to Algorithm 2) and we have the following adjacency matrix of $\mathscr{G}_\sigma$:

- **Row 1**: website of the mathematics department,
- **Row 2**: website of the faculty of science,
- **Row 3**: website of the professor with the 4 PhD students,
- **Row 4, 5, 6**: websites of the professors with the 2 PhD students,
- **Row 7, 8, 9, 10**: websites of the 4 PhD students of the professor on row 3,
- **Row 11, 12**: websites of the 2 PhD students of the professor on row 4,
- **Row 12, 14**: websites of the 2 PhD students of the professor on row 5,
- **Row 14, 16**: websites of the 2 PhD students of the professor on row 6,

Leads to:

$$B = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

Intuitively, we expect that the professor with his 4 PhD student will have the largest authority score, immediately followed by the other 3 professors. The website of the mathematics department is clearly the best hub in this example and should get the largest hub score. Also the website of the faculty of science should get a high hub score.

We now apply the HITS-method by calculating the dominant eigenvector of BB^T (this returns the hub scores) and the dominant eigenvector of B^TB (this returns the authority scores) with the power method (see 1.4.2). We get:

$$\mathbf{a} = \begin{pmatrix} 0.1979 \\ 0.3162 \\ \mathbf{0.3688} \\ 0.3231 \\ 0.3231 \\ 0.3231 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \\ 0.2029 \end{pmatrix} \quad \text{and} \quad \mathbf{h} = \begin{pmatrix} \mathbf{0.8645} \\ 0.3605 \\ 0.1207 \\ 0.1207 \\ 0.1207 \\ 0.1207 \\ 0.0866 \\ 0.0866 \\ 0.0866 \\ 0.0866 \\ 0.0758 \\ 0.0758 \\ 0.0758 \\ 0.0758 \\ 0.0758 \\ 0.0758 \end{pmatrix}$$

We see that the websites of the 4 math professors are indeed the best authorities for the search query `math professors vub` and that the website of the mathematics department is an extremely good hub (this is very logic because it links to all the other relevant websites). The professor with his 4 PhD students would be ranked first in the search results (he has the highest authority score), the other professors would appear just underneath him. Obviously, a hub score of 0.8645 is so high that it would be quite exceptional in a graph containing a lot more websites (it's very unlikely that you find a website containing links to all the other pages/nodes in the graph). Nevertheless, we conclude that the HITS-algorithm returns the results we wanted intuitively.

Predictors in the Eurovision Song Contest 2009-2015

The Eurovision Song Contest is an annual competition between countries whose public broadcaster is part of the EBU-network. The contest is the biggest music competition in the world, reaching about 200 million annually.

The contest consists of three shows: 2 semi-finals and 1 grand final. From each semi-final, 10 countries proceed the the grand final. Italy, Germany, Spain, United Kingdom and France are always qualified for the grand final because they are the main funders of the event. Also the winner of last year participates automatically in the final. Each country, also those who dropped out during the semi-finals, gives points during the voting of the grand final. The voting during the grand final takes place after all the countries have performed their song. Each country is called and and awards 12 points to their favorite song, 10 points to their second favorite, and then points from 8 down to 1 to eight other songs. Countries can not vote for themselves.

The voting system is in fact a positional voting system that is very similar to the Borda count method (see [Saa00] for a scientific explanation of Borda count): the list of points of a country represents the ranking of the 10 best countries in the voting of that country. So the points are values on an ordinal scale.

The complete voting procedure during the Eurovision Song Contest can be seen as a directed graph: all the participating countries are the nodes and the edges represent the

points between the countries (when country a assigns 3 points to country b, then there are 3 edges from a to b).

Let A be the adjacency matrix of the voting during a song contest (A will in fact be just a points table). If we take A as input for the HITS-algorithm, we expect that the country with the highest authority score will be the winner of the competition. Actually we expect a lot more: when we order the countries based on their authority score, we expect that this ordering will be practically equal to the final ranking of the contest. This is based on the simple fact that Borda count just sums up points, and we only expect very small differences when the difference in points is low between two countries. This small differences are then be caused by the algorithm: remember that the HITS-algorithm does not simply give a high authority score to nodes with a large indegree, but also takes the hub scores into account, but we will see that the hub scores will be low so their influence will indeed be limited.

But what is very interesting now, is the role that the hub scores are playing. In fact these scores can be seen as a kind of 'predictive value' of a country: a country with a high hub score will have assigned points in such way that it is seen as a reliable source, meaning that the points of that country will match well with the final result of the contest.

Let's take the Eurovision Song Contest 2014 as an example.

The final results of the contest where (the complete result table can be found in Appendix B):

1. Austria (290 points)
2. The Netherlands (238 points)
3. Sweden (218 points)
4. Armenia (174 points)
5. Hungary (143 points)
6. Ukraine (113 points)
7. Russia (89 points)
8. Norway (88 points)
9. Denmark (74 points)
10. Spain (74 points)
11. Finland (72 points)
12. Romania (72 points)
13. Switzerland (64 points)
14. Poland (62 points)
15. Iceland (58 points)
16. Belarus (43 points)
17. United Kingdom (40 points)
18. Germany (39 points)
19. Montenegro (37 points)
20. Greece (35 points)
21. Italy (33 points)

22. Azerbaijan (33 points)
23. Malta (32 points)
24. San Marino (14 points)
25. Slovenia (9 points)
26. France (2 points)

Now we calculate the hub and authority scores, based on the full scoreboard and the results are presented in Table 2.1.6.

Notice that ordering the countries by their authority scores indeed returns the final ranking of the contest. The countries who have an authority score equal to 0 are the countries that didn't make it to the final and couldn't therefore receive any points. Also the complete losers receiving the famous 'nul points' will have an authority score equal to 0, but that didn't occur during the final of 2014. When looking at the hub scores, it appears that Portugal 'predicted' the final ranking the best. When we look at the points awarded by Portugal during the final, this is indeed true:

- **12 points**: Austria
- **10 points**: The Netherlands
- **8 points**: Sweden
- **7 points**: Switzerland
- **6 points**: Hungary
- **5 points**: Denmark
- **4 points**: Armenia
- **3 points**: Norway
- **2 points**: Russia
- **1 point**: Romania

No less than 7 countries in the points of Portugal have achieved the final top 10, the top 3 results of Portugal are even equal to the final top 3! Russia, Romania and Switzerland are the 3 countries where Portugal awarded points to but didn't achieve the top 10, but still they are placed 1^{th}, 12^{th} and 13^{th} in the final ranking. So it is absolutely not surprising that Portugal is the country with the highest hub score.

Note that countries who scored very well during the final, usually have a moderate hub score: this is due to the fact that countries can not vote for themselves. The average hub score is, in general, also lower than the average authority score, this is because countries can award points to only 10 countries, but (qualified) countries can receive points from every country, except themselves. The 'predictive value' of a country is therefore limited to only 10 countries, while the voting produces a complete final ranking of 26 countries.

Of course, it's tempting to research the predictive value of countries during a couple of years. We opted for the period 2009-2015 because during the last seven editions the voting procedure remained unchanged: the points awarded by each country are based on 50% televoting and 50% jury vote. So we take the average of the hub scores of the last seven years of each country. The result is presented in Table 2.1.6. The complete results of each year can be found in Appendix B.

Participant	Authority Score	Participant	Hub Score
Austria	0.285110029	Portugal	0.173610946
The Netherlands	0.235720093	Finland	0.172574335
Sweden	0.212949392	Belgium	0.169584694
Armenia	0.156091783	Latvia	0.166295198
Hungary	0.124453384	Spain	0.165764986
Ukraine	0.095410297	Hungary	0.165699760
Norway	0.086504385	Iceland	0.165062483
Denmark	0.074474911	Estonia	0.162056401
Finland	0.070675128	Denmark	0.160549539
Spain	0.068432379	Lithuania	0.159920120
Russia	0.065352465	Greece	0.159278464
Romania	0.062560768	Norway	0.156676045
Switzerland	0.055868228	Slovenia	0.156533823
Iceland	0.053973263	Sweden	0.155725338
Poland	0.052246035	Romania	0.155212094
United Kingdom	0.037978889	France	0.153993935
Germany	0.029958317	Switzerland	0.153864143
Belarus	0.027945852	Israel	0.153059334
Malta	0.026911408	Ireland	0.147584917
Italy	0.023817428	The Netherlands	0.145211066
Montenegro	0.023425296	United Kingdom	0.144342619
Azerbaijan	0.022613716	Austria	0.137956312
Greece	0.021816155	Germany	0.136869443
San Marino	0.009315756	Ukraine	0.135769452
Slovenia	0.005843162	Italy	0.121638547
France	0.002167387	Malta	0.119063784
Albania	0	Georgia	0.117423087
Belgium	0	Moldova	0.115874904
Estonia	0	Poland	0.112296570
FYR Macedonia	0	FYR Macedonia	0.107531180
Georgia	0	Russia	0.102149330
Ireland	0	San Marino	0.098204303
Israel	0	Montenegro	0.097193444
Latvia	0	Albania	0.091897617
Lithuania	0	Belarus	0.089227164
Moldova	0	Azerbaijan	0.068005452
Portugal	0	Armenia	0.050899422

Table 2.1: The authority and hub scores of the countries during the Final of the Eurovision Song Contest 2014

We have to put a condition on the results in Table 2.1.6: countries should have participated 6 out of 7 times, more than 1 absence would possibly make the average misleading. In that case, the best predictors are:

1. Hungary

2. Cyprus
3. The Netherlands
4. Belgium
5. Spain

And the worst predictors are:

1. Armenia
2. Azerbaijan
3. FYR Macedonia
4. Georgia
5. Albania

We see that the best predictors are indeed not the most succesfull countries at the contest itself: Hungary and Cyprus never reached the top 10 in the last seven years, The Netherlands only reached the final twice, Belgium qualified three times. Spain is as a main funder directly qualified for the grand final, but was 5 out of 7 times placed lower than the 20th place. The bottom is also not very surprising: in 2010, 2012 and 2013 Albania did not receive enough televotes so the jury decided their points (see [14] for more information), making their judging process vary from one year to another. Azerbaijan and Armenia scored very high in the last five years, (e.g. Azerbaijan reached the top 5 every year except 2014 and 2015, won in 2011 and became second in 2013) and due to their dispute about the Nagorno-Karabakh region, they never exchanged a single point during the last five years. Also cultural differences are probably an explanation: Georgia, Azerbaijan and Armenia are located in the Caucasus, a remote corner of Europe, with many Asian influences (the region is sometimes refered to as Eurasia).

Table 2.2: The average of the hub scores between 2009-2015

#	Participant	2015	2014	2013	2012	2011	2010	2009	Average
1.	**Australia**	0.157516	-	-	-	-	-	-	**0.157516**
2.	**Hungary**	0.146121	0.165700	0.150320	0.151291	0.130650	-	0.158260	**0.150390**
3.	**Cyprus**	0.145637	-	0.161413	0.140385	0.145691	0.144175	0.146567	**0.147311**
4.	**The Netherlands**	0.155502	0.145211	0.133926	0.147638	0.137637	0.139085	0.163395	**0.146056**
5.	**Belgium**	0.145478	0.169585	0.159189	0.153462	0.118487	0.147362	0.122234	**0.145114**
6.	**Spain**	0.158995	0.165765	0.153609	0.133174	0.114255	0.162626	0.122739	**0.144452**
7.	**Israel**	0.146691	0.153059	0.157744	0.139948	0.131677	0.118970	0.157031	**0.143589**
8.	**Latvia**	0.149527	0.166295	0.134627	0.138436	0.121329	0.142837	0.143788	**0.142406**
9.	**Slovakia**	-	-	-	0.138399	0.141130	0.145328	0.142350	**0.141802**
10	**Estonia**	0.145329	0.162056	0.141096	0.131495	0.143254	0.136098	0.131401	**0.141533**
11.	**Lithuania**	0.118747	0.159920	0.141552	0.144121	0.127425	0.144634	0.152601	**0.141286**
12.	**Denmark**	0.160507	0.160550	0.112496	0.139704	0.104782	0.144174	0.156365	**0.139797**
13.	**Malta**	0.145890	0.119064	0.146482	0.122804	0.162192	0.138073	0.139682	**0.139170**
14.	**Poland**	0.159925	0.112297	-	-	0.127825	0.159433	0.135973	**0.139090**
15.	**Slovenia**	0.135649	0.156534	0.141375	0.148665	0.118682	0.137277	0.133178	**0.138766**
16.	**Iceland**	0.145817	0.165062	0.149906	0.129060	0.130358	0.137492	0.113493	**0.138741**
17.	**Austria**	0.150454	0.137956	0.127058	0.153796	0.129429	-	0.132217	**0.138485**
18.	**Germany**	0.157865	0.136869	0.126650	0.157167	0.103942	0.115909	0.155134	**0.136220**
19.	**Croatia**	-	-	0.169310	0.132023	0.129980	0.114014	0.134540	**0.135974**
20.	**Romania**	0.153499	0.155212	0.141742	0.119829	0.131675	0.128084	0.119466	**0.135644**
21.	**France**	0.142272	0.153994	0.135187	0.151435	0.137268	0.118194	0.109034	**0.135341**
22.	**Greece**	0.125544	0.159278	0.144943	0.124054	0.145912	0.095629	0.148054	**0.134774**
23.	**Finland**	0.149564	0.172574	0.108008	0.141472	0.101891	0.133330	0.132773	**0.134231**
24.	**Ireland**	0.141492	0.147585	0.137790	0.132864	0.103704	0.147411	0.128126	**0.134139**
25.	**United Kingdom**	0.153125	0.144343	0.143615	0.121922	0.096917	0.136172	0.133630	**0.132818**
26.	**Russia**	0.128412	0.102149	0.130277	0.129907	0.143360	0.141861	0.152992	**0.132708**
27.	**Sweden**	0.130248	0.155725	0.122816	0.099063	0.106864	0.153858	0.160329	**0.132701**
28.	**Bulgaria**	-	-	0.136342	0.150772	0.110776	0.136208	0.122794	**0.131378**
29.	**Norway**	0.144331	0.156676	0.098933	0.154850	0.099012	0.155455	0.107546	**0.130972**

Table 2.2: The average of the hub scores between 2009-2015 (continued)

#	Participant	2015	2014	2013	2012	2011	2010	2009	Average
30.	**Bosnia & Herzegovina**	-	-	-	0.135192	0.127186	0.139054	0.119158	**0.130147**
31.	**Portugal**	0.144985	0.173611	-	0.116836	0.130723	0.107674	0.105899	**0.129955**
32.	**Ukraine**	-	0.135769	0.106438	0.123998	0.120992	0.139278	0.142826	**0.128217**
33.	**Belarus**	0.147342	0.089227	0.143351	0.120914	0.138934	0.104108	0.149588	**0.127638**
34.	**Serbia**	0.128584	-	0.165969	0.113907	0.095025	0.132455	0.123080	**0.126503**
35.	**Switzerland**	0.146739	0.153864	0.117585	0.125128	0.092621	0.127077	0.117733	**0.125821**
36.	**Moldova**	0.130904	0.115875	0.139519	0.119439	0.126125	0.119696	0.119161	**0.124388**
37.	**Turkey**	-	-	-	0.113535	0.145608	0.137704	0.094715	**0.122890**
38.	**San Marino**	0.131533	0.098204	0.092521	0.130841	0.158559	-	-	**0.122332**
39.	**Italy**	0.146217	0.121639	0.134389	0.109725	0.098845	-	-	**0.122163**
40.	**Montenegro**	0.085579	0.097193	0.156165	0.137301	-	-	0.126144	**0.120476**
41.	**Albania**	0.124384	0.091898	0.102282	0.098185	0.142108	0.140538	0.141493	**0.120127**
42.	**Georgia**	0.111867	0.117423	0.147386	0.123067	0.132744	0.086281	-	**0.119795**
43.	**FYR Macedonia**	0.085414	0.107531	0.138521	0.132612	0.116197	0.127626	0.124228	**0.118876**
44.	**Czech Republic**	0.130862	-	-	-	-	-	0.103917	**0.117390**
45.	**Azerbaijan**	0.125808	0.068005	0.109553	0.113396	0.119173	0.120818	0.109357	**0.109444**
46.	**Armenia**	0.119244	0.050899	0.125792	-	0.128531	0.101509	0.109782	**0.105960**

2.1.7 Final reflection

The HITS-algorithm is one of the few algorithms that has the ability to rank pages according to a specific search query. Also the computational cost of the HITS-algorithm, which equals the cost of the power method (see 1.4.2), is not excessive and feasible for most servers. The result of the HITS-algorithm for popular queries will also be cached by most search engines, which reduces the computational cost even more because the saved results can be served directly to the user without any new calculations.

The biggest disadvantage of the HITS-algorithm is that it suffers from *topic drift*: the graph $\mathscr{G}'_\sigma$ could contain nodes which have high authority scores for the query but are completely irrelevant. E.g. `Facebook` is nowadays a universally popular website, almost every website contains a 'like' or 'share' button linking to Facebook, and Facebook itself contains tons of posts linking to other webpages. This means that Facebook has a great chance to appear in almost any $\mathscr{G}'_\sigma$ and receive a high authority score because the original HITS-algorithm as presented here cannot detect such 'universally popular' websites. The same goes for other social media websites and some advertisements.

Nowadays, we know that `Ask.com` uses this algorithm. In fact, most search engines are very secretive about their search algorithm (e.g. `Google`) to make profit and avoid cheating by webmasters. Still, the chances are that other search engines use some variant of the algorithm as well, in combination with a lot of other procedures.

2.2 Node similarity

This section provides a detailed overview of the paper '*A Measure of Similarity between Graph Vertices: Applications to Synonym Extraction and Web Searching*' [BGH+04] of V. D. Blondel and others. The paper generalizes the HITS-algorithm leading to the concept of (node) similarity on graphs. In the following chapters, we will call this method often the 'method of Blondel' . This method is explained in detail and will serve as a framework for the following sections and chapters. The method allows a mathematical rigorous approach, leading to a very important convergence theorem. Thanks to it's generality, this convergence theorem is extremely powerful and will be used to prove convergence for all the presented algorithms in the next sections and chapters. This theorem also solves the issue from the previous section where the matrix M has to have a unique dominant eigenvalue. Once we constructed the algorithm, proved the convergence of it and introduced the compact form, we also look at some special cases of node similarity between graphs.

Introduction

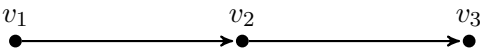
In this introduction, we explain the main idea behind the method of Blondel. Because we are focussing on the intuitive ideas here, we don't prove any theorems in this paragraph yet. The theoretical results are presented in the following paragraphs. Notice that all the graphs in this section are directed graphs, although this is not always explicitely stated. However, the method works for undirected graphs too, often leading to much easier formulas since the graphs have symmetric adjacency matrices in that case. We will give explicit attention to undirected graphs in the section about 'Special cases' (see 2.2.4).

The method of Blondel is a pure generalization of the HITS-algorithm. Remember from the previous section that we constructed a graph $\mathscr{G}'_\sigma$ and calculated hub and authorities scores for each vertex. Well, the authors of [BGH+04] found a way to replace these hub and authority scores by just any other graph, allowing us to compare $\mathscr{G}'_\sigma$ to any other graf $\mathscr{H}$. We will call the graph that replaces the hub and authority scores the *structure graph*.

How does this work? For $\mathscr{G}'_\sigma$, the authority score of vertex v_j of $\mathscr{G}'_\sigma$ can be thought of as a *score* between v_j of $\mathscr{G}$ and the vertex denoted as *authority* of the graph:

and, conversely, the hub score of vertex v_j of $\mathscr{G}'_\sigma$ can be thought of as a score between v_j and the vertex denoted as *hub*. We can replace $\mathscr{G}'_\sigma$ with any other graph $\mathscr{G}$. We call the hub-authority graph a *structure graph* and we already know the resulting iterative method from the previous section. We will call this scores *similarity scores*.

The central question is now: which *mutually reinforcing relation* do we get when using another structure graph, different from the hub-authority structure graph? If we take, for example, as structure graph a path graph with three vertices v_1, v_2, v_3:



Let $\mathscr{G}(W, \rightarrow)$ be a graph. With each vertex w_i of $\mathscr{G}$ we now associate three scores z_{i1}, z_{i2} and z_{i3}, one for each vertex of the structure graph. We initialize these scores with a positive value and then update them according to the mutually reinforcing relation:

$$\begin{cases} z_{i1} := & \sum_{j:(w_i,w_j)\in\rightarrow} z_{j2}, \\ z_{i2} := \sum_{j:(w_j,w_i)\in\rightarrow} z_{j1} & + \sum_{j:(w_i,w_j)\in\rightarrow} z_{j3}, \\ z_{i3} := & \sum_{j:(w_j,w_i)\in\rightarrow} z_{j2}, \end{cases}$$

or, in matrix form ($\mathbf{z}_j$ denotes the column vector with entries z_{ij}, B is the adjacency matrix of graph $\mathscr{G}$),

$$\begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \mathbf{z}_3 \end{pmatrix}^{(k+1)} = \begin{pmatrix} 0 & B & 0 \\ B^T & 0 & B \\ 0 & B^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \mathbf{z}_3 \end{pmatrix}^{(k)}$$

which we, again, can denote by

$$\mathbf{z}^{(k+1)} = M\mathbf{z}^{(k)}. \tag{2.3}$$

The principle is now exactly the same as the previous example with hubs and authorities. The matrix M is symmetric and nonnegative, and again the result is the limit of the normalized vector sequence:

$$\mathbf{s}^{(0)} = \mathbf{z}^{(0)} > 0, \mathbf{s}^{(k+1)} = \frac{M\mathbf{s}^{(k)}}{||M\mathbf{s}^{(k)}||_2}, \quad k = 0, 1, \ldots, \tag{2.4}$$

Remember that the HITS-algorithm assumed that M has a unique dominant eigenvalue but we don't want to make this assumption in this section because we want a concept that can be applied to all kinds of directed graphs. We will see that without this assumption, the sequence 2.4 does not always converge but oscillates between the limits:

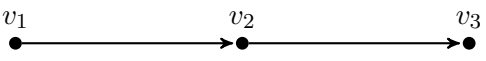
$$\mathbf{s}_{\text{even}} = \lim_{k\to\infty} \mathbf{s}^{(2k)} \quad \text{and} \quad \mathbf{s}_{\text{odd}} = \lim_{k\to\infty} \mathbf{s}^{(2k+1)}$$

The limit vectors $\mathbf{s}_{\text{even}}$ and $\mathbf{s}_{\text{odd}}$ do in general depend on the initial vector $\mathbf{z}^{(0)}$. We will prove later on that the vector $\mathbf{z}^{(0)} = J$, with J a matrix of ones, is a good choice.

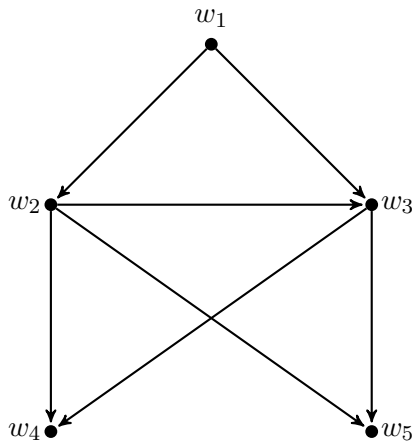
The extremal limit $\mathbf{s}_{\text{even}}(J)$ will be defined as the *similarity matrix*. The element s_{ij} is called the *similarity score* between vertex w_i of $\mathscr{G}$ and vertex v_j of the structre graph.

We now give a numerical example.

Example 2.2.1. Take as structure graph again the path graph with three vertices v_1, v_2, v_3:



Let $\mathscr{G}(W, \rightarrow)$ be the following graph:



Then the adjacency matrix B is:

$$B = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

By using the described mutually reinforcing updating iteration we get the following similarity matrix (a numerical algorithm to calculate this is presented later on in this section):

$$S = \begin{pmatrix} 0.3557 & 0.1265 & 0 \\ 0.3102 & 0.3451 & 0.0557 \\ 0.2732 & 0.4619 & 0.4115 \\ 0 & 0.1579 & 0.3557 \\ 0 & 0.0840 & 0.1521 \end{pmatrix}$$

The similarity score of w_4 with v_2 of the structure graph is equal to 0.1579.

We now construct the general case. Take two (directed) graphs $\mathscr{G} = (U, \to)$ and $\mathscr{H} = (V, \to')$ with $n_{\mathscr{G}}$ and $n_{\mathscr{H}}$ the order of the graphs. We think of $\mathscr{G}$ as the structure graph (such as the graphs hub $\to$ authority and the graph $1 \to 2 \to 3$ in the previous paragraphs). We get the following mutually reinforcing updating iteration with as updating equations:

$$z_{ij}^{(k+1)} := \sum_{p:(v_p,v_i)\in\to',q:(u_q,u_j)\in\to} x_{pq}^{(k)} + \sum_{p:(v_i,v_p)\in\to',q:(u_j,u_q)\in\to} z_{pq}^{(k)} \tag{2.5}$$

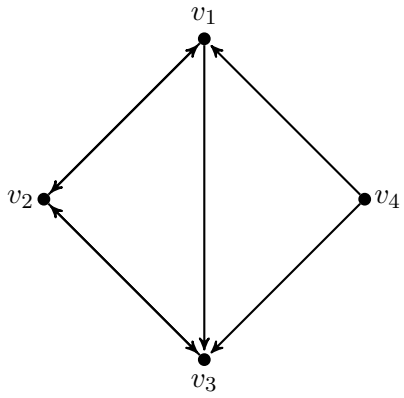
Consider the product graph $\mathscr{G} \times \mathscr{H}$ (see Definition 1.5.17). The above updating equation is equivalent to replacing the scores of all vertices of the product graph by the sum of the scores of the vertices linked by an incoming or outgoing edge.

Equation (2.4) can be rewritten in a more compact matrix form. Let Z_k be the $n_{\mathscr{H}} \times n_{\mathscr{G}}$ matrix of entries z_{ij} at iteration k, and A and B are the adjacency matrices of $\mathscr{G}$ and $\mathscr{H}$. Then the updating equations can be written as:

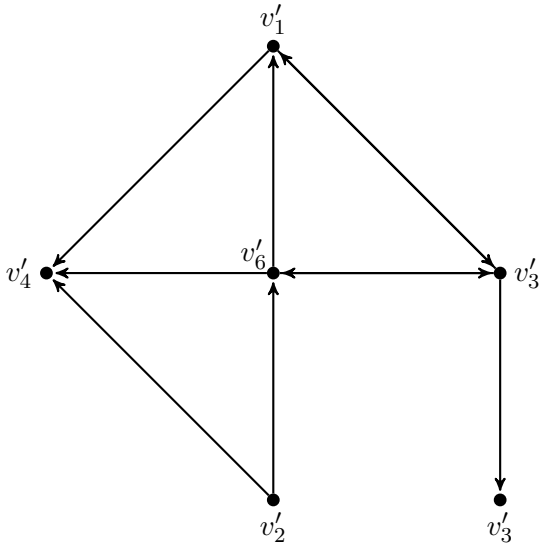
$$Z^{(k+1)} = BZ^{(k)}A^T + B^T Z^{(k)} A, \quad k = 0, 1, \ldots, \tag{2.6}$$

We'll prove that the normalized even and odd iterates of this updating equation converge when using as start matrix J, the matrix of ones. This normalized result will be denoted by S, the similarity matrix. This compact form also shows the true meaning of similarity between nodes: two similarity score of a vertex in a graph is lare when the simialrity sore of the adjacencent vertices is large. We will dive deeper into the meaning of the algorithm in Chapter 3. The following example shows a calculated similarity matrix of two directed graphs.

Example 2.2.2. Let $\mathscr{H}(V, \rightarrow)$ be the following graph:



Let $\mathscr{G}(V', \rightarrow')$ be the following graph:



This results in the following similarity matrix (a numerical algorithm to calculate this matrix is introduced later in this section):

$$S = \begin{pmatrix} 0.2636 & 0.2786 & 0.2723 & 0.1289 \\ 0.1286 & 0.1286 & 0.0624 & 0.1268 \\ 0.2904 & 0.3115 & 0.2825 & 0.1667 \\ 0.1540 & 0.1701 & 0.2462 & 0 \\ 0.0634 & 0.0759 & 0.1018 & 0 \\ 0.3038 & 0.3011 & 0.2532 & 0.1999 \end{pmatrix}$$

We see for example, that vertex v_2 of $\mathscr{H}$ is most similar to vertex v_3' in $\mathscr{G}$ because the similarity score s_{32} is the highest among all the similarity scores of v_2.

2.2.1 Convergence theorem

In the introduction, we mentioned already that the sequence in Equation (2.4) converges for even and odd iterates. We will prove this at the end of this subsection. But before we arrive there, we first need some results on the eigenvectors and eigenvalues of nonnegative matrices. The Perron-Frobenius applies only to nonnegative, *irreducible* matrices, but since one is confronted in practice with nonnegative matrices that are not necessary irreducible, we extend the Perron-Frobenius and see what remains without this assumption. We will prove that the spectral radius $\rho(M)$ of a nonnegative matrix M is an eigenvalue of M, also called in this case the the Perron root (see Definition 1.2.11). Moreover, there exists an associated nonnegative eigenvector $\mathbf{x} \geq 0 (\mathbf{x} \neq 0)$, the Perron vector, such that $M\mathbf{x} = \rho\mathbf{x}$. We will also investigate in more specific results in the case M is not only nonnegative, but also symmetric.

Lemma 2.2.3. *Let A, B be $n \times n$-matrices, if $|A| \leq B$, then $\rho(A) \leq \rho(|A|) \leq \rho(B)$. (See Definition 1.1.8 for the definition of $|\cdot|$).*

Proof. For every $m = 1, 2, \ldots$ we have

$$|A^m| \leq |A|^m \leq B^m$$

by using some trivial properties of the absolute value function. Let $||\cdot||_2$ be the matrix 2-norm induced by the Euclidean vector norm: for any matrix M, we have $\|M\|_2 = \max_{\|\mathbf{x}\|_2=1} \|M\mathbf{x}\|_2$. For this matrix norm it is trivial to see that if $|M| \leq |M'|$ (see Definition 1.1.5) it follows that $\|M\|_2 \leq \|M'\|_2$ and also $\|M\|_2 = \| \, |M| \, \|_2$, we get:

$$\|A^m\|_2 \leq \| \, |A|^m \, \|_2 \leq \|B^m\|_2$$

and

$$\|A^m\|_2^{1/m} \leq \| \, |A|^m \, \|_2^{1/m} \leq \|B^m\|_2^{1/m}$$

for all $m = 1, 2, \ldots$ If we now let $m \to \infty$ and apply the spectral radius formula from Theorem 1.3.12 we get:

$$\rho(A) \leq \rho(\|A\|) \leq \rho(B).$$

□

Theorem 2.2.4. *If $A \geq 0$ is an $n \times n$-matrix, then $\rho(A)$ is an eigenvalue of A and there is a nonnegative vector $\mathbf{x} \geq \mathbf{0}, \mathbf{x} \neq \mathbf{0}$, such that $A\mathbf{x} = \rho(A)\mathbf{x}$.*

Proof. For any $\epsilon > 0$ define $A(\epsilon) = (a_{ij} + \epsilon) > 0$, so in $A(\epsilon)$ we add ϵ to each entry of A. Denote by $\mathbf{x}(\epsilon)$ the Perron vector of $A(\epsilon)$, this exists by the Perron-Frobenius as $A(\epsilon)$ is a strictly positive matrix (see Theorem 1.2.10), so $\mathbf{x}(\epsilon) > 0$ and $\sum_{i=1}^n x(\epsilon)_i = 1$. Since the set of vectors $\{\mathbf{x}(\epsilon) : \epsilon > \mathbf{0}\}$ is contained in the compact set $\{\mathbf{x} : \mathbf{x} \in \mathbb{R}^n, \|\mathbf{x}\|_1 \leq 1\}$, we know from the Bolzano-Weierstrass theorem (see Theorem 2.1 in [Col12]) that there is a monotone decreasing sequence $\epsilon_1 > \epsilon_2 > \ldots$ with $\lim_{k\to\infty} \epsilon_k = 0$ such that $\lim_{k\to\infty} \mathbf{x}(\epsilon_k) = \mathbf{x}$ exists. Since $\mathbf{x}(\epsilon_k) > \mathbf{0}$ for all $k = 1, 2, \ldots$, it must be that $\mathbf{x} = \lim_{k\to\infty} \mathbf{x}(\epsilon_k) \geq \mathbf{0}$; $\mathbf{x} = \mathbf{0}$ is impossible because:

$$\sum_{i=1}^n x_i = \lim_{k\to\infty} \sum_{i=1}^n x(\epsilon_k)_i = 1$$

By Lemma 2.2.3, $\rho(A(\epsilon_k)) \geq \rho(A(\epsilon_{k+1}) \geq \cdots \geq \rho(A)$ for all $k = 1, 2, \ldots$, so the sequence $\{\rho(A(\epsilon_k))\}_{k=1,2,\ldots}$ is a monotone decreasing sequence. Thus, $\rho = \lim_{k\to\infty} \rho(A(\epsilon_k))$ exists and $\rho \geq \rho(A)$. From the fact that

$$\begin{aligned} A\mathbf{x} &= \lim_{k\to\infty} A(\epsilon_k)\mathbf{x}(\epsilon_k) \\ &= \lim_{k\to\infty} \rho(A(\epsilon_k))\mathbf{x}(\epsilon_k) \\ &= \lim_{k\to\infty} \rho(A(\epsilon_k)) \lim_{k\to\infty} \mathbf{x}(\epsilon_k) \\ &= \rho\mathbf{x}, \end{aligned}$$

and the fact that $\mathbf{x} \neq \mathbf{0}$ we conclude that ρ is an eigenvalue of A. But then $\rho \leq \rho(A)$ so it must be that $\rho = \rho(A)$. □

Now that we know that any nonnegative matrix M has his spectral radius as an eigenvalue and there exists an associated nonnegative eigenvector, we will see if we can get more specific results when handling nonnegative, symmetric matrices. First we introduce the orthogonal projection.

Definition 2.2.5. *A **projection of a vector** $\mathbf{x}$ on a vector $\mathbf{y}$ is defined as*[3]*:*

$$proj_{\mathbf{y}}\mathbf{x} = \frac{\langle \mathbf{y}, \mathbf{x} \rangle}{\|\mathbf{y}\|}\mathbf{y},$$

the projection of $\mathbf{x}$ on a subspace Y zith orthonormal basis $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m\}$ is defined as the sum of projections:

$$proj_Y\mathbf{x} = \sum_{i=1}^{m} \frac{\langle \mathbf{y}_i, \mathbf{x} \rangle}{\|\mathbf{y}_i\|}\mathbf{y}_i$$

We now combine the above results to get a new result of the Perron-Frobenius when handling nonnegative, symmetric matrices.

Theorem 2.2.6. *Let $\mathcal{V}$ be a linear subspace of $\mathbb{R}^n$ with orthonormal basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}$. Arrange the column vectors $\mathbf{v}_i$ in a matrix V and let $\mathbf{x} \in \mathbb{R}^n$. The **orthogonal projection** of $\mathbf{x}$ on $\mathcal{V}$ is then given by:*

$$\Pi\mathbf{x} = VV^T\mathbf{x},$$

*the matrix $\Pi = VV^T$ is the **orthogonal projector**. Projectors have the property that $\Pi^2 = \Pi$*

Proof. We use the connection between transposes and the standard inner product to find the matrix of the orthogonal projection on the subspace $\mathcal{V}$:

$$\begin{aligned} \text{proj}_{\mathcal{V}}\mathbf{x} &= \sum_{i=1}^{m} \frac{\langle \mathbf{v}_i, \mathbf{x} \rangle}{\|\mathbf{v}_i\|}\mathbf{v}_i \\ &= \sum_{i=1}^{m} \langle \mathbf{v}_i, \mathbf{x} \rangle \mathbf{v}_i \\ &= \sum_{i=1}^{m} \mathbf{v}_i \langle \mathbf{v}_i, \mathbf{x} \rangle \end{aligned}$$

[3] $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i$ with $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is the standard inner product of two vectors in $\mathbb{R}^n$.

Remember that $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T \mathbf{y}$ so:

$$\begin{aligned} &= \sum_{i=1}^{m} \mathbf{v}_i (\mathbf{v}_i^T \mathbf{x}) \\ &= \sum_{i=1}^{m} (\mathbf{v}_i \mathbf{v}_i^T) \mathbf{x} \\ &= VV^T \mathbf{x} \end{aligned}$$

Proving that $\Pi^2 = \Pi$ is also trivial, remember that for an orthogonal matrix A it holds that $A^T = A^{-1}$:

$$\begin{aligned} \Pi^2 &= (VV^T)^2 \\ &= VV^T VV^T \\ &= VV^{-1} VV^T \\ &= I_n VV^T \\ &= VV^T \\ &= \Pi \end{aligned}$$

□

Lemma 2.2.7. *Let Π be an orthogonal projecter, then:*

$$\Pi^T = \Pi$$

Proof.

$$\Pi^T = (VV^T)^T = (V^T)^T V^T = VV^T = \Pi$$

□

We already introduced the Jordan normal form in Theorem 1.3.10, but we also need the following theorem. The proof of this theorem is highly technical and falls behind the scope of this master thesis. The seperate results can be found in Theorems 1.3.15, 1.3.16, 1.3.17 in [**?**].

Theorem 2.2.8. *Let A be a $n \times n$-matrix with s distinct eigenvalues $\lambda_1, \ldots, \lambda_s$. Let each λ_i have algebraic multiplicity m_i and geometric multiplicity μ_i. The matrix A is similar to a Jordan matrix*

$$J = \begin{pmatrix} J_1 & & \\ & \ddots & \\ & & J_\mu \end{pmatrix}$$

where

1. *$\mu = \mu_1 + \mu_2 + \cdots + \mu_s$,*
2. *For each λ_i, the number of Jordan blocks in J with value λ_i is equal to μ_i,*
3. *λ_i appears on the diagonal of J exactly m_i times.*

Further, the matrix J is unique, up to re-ordering the Jordan blocks on the diagonal.

We are now ready for genealizing the Perron-Frobenius to nonnegative, symmetric matrices.

Theorem 2.2.9. *Let M be a symmetric nonnegative matrix with spectral radius ρ. Then the algebraic and geometric multiplicity of the Perron root ρ are equal; there is a nonnegative matrix X whose columns span the eigenspace associated with the Perron root; and the elements of the orthogonal projector Π on the vector space associated with the Perron root of M are all nonnegative.*

Proof. We know that any symmetric nonnegative matrix M can be permuted to a Jordan canonical form J. So from the previous Theorem 2.2.8, we know that ρ will appear exactly m_ρ (= the algebraic multiplicity of ρ) times on the diagonal and we also know that there will be exactly μ_ρ (= the geometric multiplicity of ρ) Jordan blocks with value ρ on their diagonal. But from Theorem 2.1.2 we already know that J is in fact a diagonal matrix (all Jordan blocks have size 1×1) with all eigenvalues on the diagonal, so $m_\rho = \mu_\rho$. To construct the nonnegative matrix X, we use again Theorem 2.1.2: $P^{-1}MP = D$ and we know from the theorem that the column vectors of P are the eigenvectors corresponding to the eigenvalues of M. Looking at the proof of Theorem 2.1.2 we know that the eigenvector $\mathbf{x}_i$ corresponding to the eigenvalue d_{ii} (the i'th diagonal element of D), can be found at the i'th column of P. We know that these eigenvectors are linear independent because of the definition of diagonalizable matrices (A $n \times n$ matrix A is diagonalizable if and only if it has an eigenbasis). So if we select all the columns of P corresponding to ρ, we found the eigenspace of the Perron root.

To see that the resulting matrix X is nonnegative, notice that the eigenspace of ρ can also be found by taking the Perron vectors of the (1×1) Jordan blocks J_i that have ρ on the diagonal appropriately padded with zeros. All these blocks J_i are irreducible and nonnegative and it follows from the Perron-Frobenius that the corresponding eigenvectors are nonnegative.

To see that the orthogonal projector Π associated with the eigenspace of ρ is nonnegative, notice that normalizing the vectors in X gives an orthonormal basis because by Theorem 2.1.2 we know that P is orthogonal and we picked some columns of P to form our eigenbasis of ρ. Normalizing doesn't affect the sign of the vectors, so the normalized X' is also nonnegative, so $\Pi = X'X'^T$ will also be nonnegative. □

Theorem 2.2.10. *Let M be a $n \times n$-symmetric, nonnegative matrix of spectral radius ρ. Let $\mathbf{s}^{(0)} > 0$ and consider the sequence*

$$\mathbf{s}^{(k+1)} = \frac{M\mathbf{s}^{(k)}}{||M\mathbf{s}^{(k)}||_2}, k = 0, 1, \ldots$$

Two convergence cases can occur depending on whether or not $-\rho$ is an eigenvalue of M. When $-\rho$ is not an eigenvalue of M, then the sequence of $\mathbf{s}^{(k)}$ simply converges to $\frac{\Pi\mathbf{s}^{(0)}}{||\Pi\mathbf{s}^{(0)}||_2}$, where Π is the orthogonal projector on the eigenspace associated with the Perron root ρ. When $-\rho$ is an eigenvalue of M, then the subsequences $\mathbf{s}^{(2k)}$ and $\mathbf{s}^{(2k+1)}$ converge to the limits

$$\mathbf{s}_{even}(\mathbf{s}^{(0)}) = \lim_{k\to\infty} \mathbf{s}^{(2k)} = \frac{\Pi\mathbf{s}^{(0)}}{||\Pi\mathbf{s}^{(0)}||_2} \quad \text{and} \quad \mathbf{s}_{odd}(\mathbf{s}^{(0)}) = \lim_{k\to\infty} \mathbf{s}^{(2k+1)} = \frac{\Pi M\mathbf{s}^{(0)}}{||\Pi M\mathbf{s}^{(0)}||_2}.$$

where Π is the orthogonal projector on the sums of the invariant subspaces associated with ρ and $-\rho$. In both cases the set of all possible limits is given by:

$$\left\{\mathbf{s}_{even}(\mathbf{s}^{(0)}), \mathbf{s}_{odd}(\mathbf{s}^{(0)}) : \mathbf{s}^{(0)} > 0\right\} = \left\{\frac{\Pi\mathbf{s}}{||\Pi\mathbf{s}||_2} : \mathbf{s} > 0\right\}$$

and the vector $\mathbf{s}_{even}(\mathbf{1})$ *is the unique vector of largest possible Manhattan norm in this set.*

Proof. We prove only the case where $-\rho$ is an eigenvalue, the other case is an easy adaption. Denote the invariant subspaces of M corresponding to ρ, $-\rho$ and to the rest of the spectrum, respectively by $\mathcal{V}_\rho, \mathcal{V}_{-\rho}$ and $\mathcal{V}_\mu$. From the previous theorem, we know that $\mathcal{V}_\rho, \mathcal{V}_{-\rho}$ are certainly nontrivial because ρ and $-\rho$ have at least multiplicity 1, so the eigenspace contains also at least 1 vector. We also suppose that $\mathcal{V}_\mu$ is nontrivial (if $\mathcal{V}_\mu$ were trivial, the proof becomes would get only easier). Let V_ρ be the $n \times u$-matrix whose columns span $\mathcal{V}_\rho$, $V_{-\rho}$ the $n \times v$ matrix spanning $\mathcal{V}_{-\rho}$, V_μ the $n \times w$- matrix spanning $\mathcal{V}_\mu$. We now have:

$$MV_\rho = \rho V_\rho, \quad MV_{-\rho} = -\rho V_{-\rho},$$

for M we can write:

$$MV_\mu = V_\mu M_\mu,$$

where M_μ is a $w \times w$-matrix with spectral radius μ strictly less than ρ because $\mathcal{V}_\mu$ is nontrivial by assumption.

Remember from Theorem 2.1.2 that $P^{-1}MP = D$, with D a diagonal matrix, this can be rewritten as $M = PDP^{-1}$ or $M = PDP^T$ (P is an orthogonal matrix), we can rewrite this in this case as (this is the so called *eigendecomposition* for symmetric matrices):

$$\begin{aligned} M &= \begin{pmatrix} V_\rho & V_{-\rho} & V_\mu \end{pmatrix} \begin{pmatrix} \rho I & & \\ & -\rho I & \\ & & M_\mu \end{pmatrix} \begin{pmatrix} V_\rho & V_{-\rho} & V_\mu \end{pmatrix}^T \\ &= \rho V_\rho V_\rho^T - \rho V_{-\rho} V_{-\rho}^T + V_\mu M_\mu V_\mu^T \end{aligned}$$

Let $A = \begin{pmatrix} V_\rho & V_{-\rho} & V_\mu \end{pmatrix}$, it follows that (remember that A is orthogonal from Theorem 2.1.2):

$$\begin{aligned} M^2 &= A \begin{pmatrix} \rho I & & \\ & -\rho I & \\ & & M_\mu \end{pmatrix} A^T A \begin{pmatrix} \rho I & & \\ & -\rho I & \\ & & M_\mu \end{pmatrix} A^T \\ &= A \begin{pmatrix} \rho I & & \\ & -\rho I & \\ & & M_\mu \end{pmatrix} A^{-1} A \begin{pmatrix} \rho I & & \\ & -\rho I & \\ & & M_\mu \end{pmatrix} A^T \\ &= A \begin{pmatrix} \rho^2 I & & \\ & (-\rho)^2 I & \\ & & M_\mu^2 \end{pmatrix} A^T \\ &= \rho^2 V_\rho V_\rho^T + \rho^2 V_{-\rho} V_{-\rho}^T + V_\mu M_\mu^2 V_\mu^T \\ &= \rho^2 \Pi + V_\mu M_\mu^2 V_\mu^T \end{aligned}$$

where

$$\Pi = V_\rho V_\rho^T + V_{-\rho} V_{-\rho}^T$$

is the orthogonal projector on the invariant subspace $\mathcal{V}_\rho \oplus \mathcal{V}_{-\rho}$ of M^2 corresponding to ρ^2. Analogously, we also have:

$$M^{2k} = \rho^{2k} \Pi + V_\mu M_\mu^{2k} V_\mu^T, \tag{2.7}$$

and since M_μ has as spectral radius μ, strictly below ρ, we multiply (2.7) by $\mathbf{s}^{(0)}$:

$$\begin{aligned} M^{2k}\mathbf{s}^{(0)} &= \rho^{2k}\Pi\mathbf{s}^{(0)} + V_\mu M_\mu^{2k} V_\mu^T \mathbf{s}^{(0)} \\ \Leftrightarrow \frac{M^{2k}\mathbf{s}^{(0)}}{\|M^{2k}\mathbf{s}^{(0)}\|_2} &= \frac{\rho^{2k}\Pi\mathbf{s}^{(0)} + V_\mu M_\mu^{2k} V_\mu^T \mathbf{s}^{(0)}}{\|\rho^{2k}\Pi\mathbf{s}^{(0)} + V_\mu M_\mu^{2k} V_\mu^T \mathbf{s}^{(0)}\|_2} \\ \Leftrightarrow \mathbf{s}^{(2k)} &= \frac{\Pi\mathbf{s}^{(0)} + \frac{1}{\rho^{2k}} V_\mu M_\mu^{2k} V_\mu^T \mathbf{s}^{(0)}}{\|\Pi\mathbf{s}^{(0)} + \frac{1}{\rho^{2k}} V_\mu M_\mu^{2k} V_\mu^T \mathbf{s}^{(0)}\|_2} \\ \Leftrightarrow \mathbf{s}^{(2k)} &= \frac{\Pi\mathbf{s}^{(0)}}{\|\Pi\mathbf{s}^{(0)}\|_2} + O\left(\frac{\mu}{\rho}\right)^{2k} \qquad (2.8) \\ \Leftrightarrow \lim_{k\to\infty} \mathbf{s}^{(2k)} &= \frac{\Pi\mathbf{s}^{(0)}}{\|\Pi\mathbf{s}^{(0)}\|_2} \qquad (2.9) \end{aligned}$$

Analogously, when multiplying (2.7) by $M\mathbf{s}^{(0)}$ we have:

$$\lim_{k\to\infty} \mathbf{s}^{(2k+1)} = \frac{\Pi M\mathbf{s}^{(0)}}{\|\Pi M\mathbf{s}^{(0)}\|_2} \qquad (2.10)$$

The expressions (2.9) and (2.10) bring up the question whether $\|\Pi\mathbf{s}^{(0)}\|_2$ and $\|\Pi M\mathbf{s}^{(0)}\|_2$ are always nonzero, from Definition 1.3.2 we know these norms can be calculated as:

$$(\Pi\mathbf{s}^{(0)})^T\Pi\mathbf{s}^{(0)} \quad \text{and} \quad (\Pi M\mathbf{s}^{(0)})^T(\Pi M\mathbf{s}^{(0)}),$$

since $\Pi^2 = \Pi$ and $\Pi^T = \Pi$, this equals:

$$\mathbf{s}^{(0)T}\Pi\mathbf{s}^{(0)} \quad \text{and} \quad \mathbf{s}^{(0)T}M\Pi M\mathbf{s}^{(0)},$$

these norms are both nonzero since $\mathbf{s}^{(0)} > 0$ and we know from the previous Theorem 2.2.9 that both Π and $M\Pi$ are nonnegative and nonzero.

From the fact that M is nonnegative and the formula for $\mathbf{s}_{\text{even}}(\mathbf{s}^{(0)})$ and $\mathbf{s}_{\text{odd}}(\mathbf{s}^{(0)})$ we conclude that both limits lie in $\{\Pi\mathbf{s}/\|\Pi\mathbf{s}\|_2 : \mathbf{s} > 0\}$. We now show that every element $\tilde{\mathbf{s}}^{(0)} \in \{\Pi\mathbf{s}/\|\Pi\mathbf{s}\|_2 : \mathbf{s} > 0\}$ can be obtained as $\mathbf{s}_{\text{even}}(\mathbf{s}^{(0)})$ for some $\mathbf{s}^{(0)} > 0$. Since the entries of Π are nonnegative, so are those of $\tilde{\mathbf{s}}^{(0)}$. This vector may have some zero entries. From $\tilde{\mathbf{s}}^{(0)}$ we construct $\mathbf{s}^{(0)}$ by adding $\epsilon > 0$ to all the zero entries of $\tilde{\mathbf{s}}^{(0)}$. For $\epsilon \to 0$, the vector $\mathbf{s}^{(0)} - \tilde{\mathbf{s}}^{(0)}$ is clearly orthogonal to $\mathcal{V}_\rho \oplus \mathcal{V}_{-\rho}$ and will vanish in the iteration of M^2. Thus we have $\mathbf{s}_{\text{even}}(\mathbf{s}^{(0)}) = \tilde{\mathbf{s}}^{(0)}$ for $\mathbf{s}^{(0)} > 0$, proving our statement.

We now prove the last statement. We actually want to prove that:

$$\left\| \frac{\Pi\mathbf{1}}{\|\Pi\mathbf{1}\|_2} \right\|_1 \geq \left\| \frac{\Pi\mathbf{s}^{(0)}}{\|\Pi\mathbf{s}^{(0)}\|_2} \right\|_1 \quad \forall \mathbf{s}^{(0)} > 0.$$

To prove this we rewrite both sides. Remember that Π is nonnegative, then we get with the norm properties:

$$\begin{aligned} \left\| \frac{\Pi\mathbf{1}}{\|\Pi\mathbf{1}\|_2} \right\|_1 &= \mid \frac{1}{\|\Pi\mathbf{1}\|_2} \mid \|\Pi\mathbf{1}\|_1 \\ &= \frac{\|\Pi\mathbf{1}\|_1}{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}} \end{aligned}$$

$$
\begin{aligned}
&= \frac{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\|\Pi\mathbf{1}\|_1}{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}} \\
&= \frac{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\|\Pi\mathbf{1}\|_1}{\mathbf{1}^T\Pi\mathbf{1}} \\
&= \frac{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\left(|(\Pi\mathbf{1})_1| + |(\Pi\mathbf{1})_2| + \cdots + |(\Pi\mathbf{1})_n|\right)}{\mathbf{1}^T\Pi\mathbf{1}} \\
&= \frac{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\left((\Pi\mathbf{1})_1 + (\Pi\mathbf{1})_2 + \cdots + (\Pi\mathbf{1})_n\right)}{\mathbf{1}^T\Pi\mathbf{1}} \\
&= \frac{\sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}\left((\Pi\mathbf{1})_1 + (\Pi\mathbf{1})_2 + \cdots + (\Pi\mathbf{1})_n\right)}{(\Pi\mathbf{1})_1 + (\Pi\mathbf{1})_2 + \cdots + (\Pi\mathbf{1})_n} \\
&= \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}
\end{aligned}
$$

and also:

$$
\begin{aligned}
\left\| \frac{\Pi\mathbf{s}^{(0)}}{\|\Pi\mathbf{s}^{(0)}\|_2} \right\|_1 &= \frac{\|\Pi\mathbf{s}^{(0)}\|_1}{\|\Pi\mathbf{s}^{(0)}\|_2} \\
&= \frac{|(\Pi\mathbf{s}^{(0)})_1| + |(\Pi\mathbf{s}^{(0)})_2| + \cdots + |(\Pi\mathbf{s}^{(0)})_n|}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}} \\
&= \frac{(\Pi\mathbf{s}^{(0)})_1 + (\Pi\mathbf{s}^{(0)})_2 + \cdots + (\Pi\mathbf{s}^{(0)})_n}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}} \\
&= \frac{\mathbf{1}^T\Pi\mathbf{s}^{(0)}}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}} \\
&= \frac{\mathbf{1}^T\Pi^2\mathbf{s}^{(0)}}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}}
\end{aligned}
$$

We apply the Cauchy-Schwarz inequality to $\Pi\mathbf{1}$ and $\Pi\mathbf{s}^{(0)}$, remember that $\Pi = WW^T$ for some W, so $\Pi^T = (WW^T)^T = (W^T)^TW^T = WW^T = \Pi$, that the matrix Π and all vectors are nonnegative and $\Pi^2 = \Pi$:

$$
\begin{aligned}
|\langle \Pi\mathbf{1}, \Pi\mathbf{s}^{(0)}\rangle| &\leq \|\Pi\mathbf{1}\|.\|\Pi\mathbf{s}^{(0)}\| \\
\Leftrightarrow |(\Pi\mathbf{1})^T\Pi\mathbf{s}^{(0)}| &\leq \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}.\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}} \\
\Leftrightarrow |\mathbf{1}^T\Pi^T\Pi\mathbf{s}^{(0)}| &\leq \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}.\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}} \\
\Leftrightarrow |\mathbf{1}^T\Pi^2\mathbf{s}^{(0)}| &\leq \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}.\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}} \\
\Leftrightarrow \frac{|\mathbf{1}^T\Pi^2\mathbf{s}^{(0)}|}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}} &\leq \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}} \\
\Leftrightarrow \frac{\mathbf{1}^T\Pi^2\mathbf{s}^{(0)}}{\sqrt{\mathbf{s}^{(0)T}\Pi^2\mathbf{s}^{(0)}}} &\leq \sqrt{\mathbf{1}^T\Pi^2\mathbf{1}}
\end{aligned}
$$

The last equation is true since $\Pi\mathbf{s}^{(0)}$ and $\Pi\mathbf{1}$ are both real and nonnegative. □

2.2.2 Similarity matrices

We now come to the formal definition of the similarity matrix of two graphs $\mathscr{G}(U, \to)$ and $\mathscr{H}(V, \to')$, by proving that the mutually reinforcing relation is given by (A and B are the adjacency matrices of $\mathscr{G}$ and $\mathscr{H}$):

$$Z^{(k+1)} = BX^{(k)}A^T + B^T X^{(k)} A, \quad k = 0, 1, \ldots, \tag{2.11}$$

which we already introduced in (2.6). To prove this relation, we take a detour via a nice property of the Kronecker product and the vec-operator. The vec-operator is a convenient way of stacking columns of a matrix left to right. With this operator we can rewrite (2.11) in $\mathbf{z}^{(k+1)} = M\mathbf{z}^{(k)}$ with M equal to a sum of Kronecker products. This will allow us the apply Theorem 2.2.10.

Definition 2.2.11. *With each $m \times n$-matrix $A = [a_{ij}]$ we can associate the vector* $\mathrm{vec}(A) \in \mathbb{R}^{mn}$ *defined by:*

$$\mathrm{vec}(A) = \begin{pmatrix} a_{11} & \ldots & a_{m1} & a_{12} & \ldots & a_{m2} & a_{1n} & \ldots & a_{mn} \end{pmatrix}^T$$

Definition 2.2.12. *The **Kronecker product** (or **tensor product**) of an $m \times n$-matrix $A = (a_{ij})$ and a $p \times q$-matrix $B = (b_{ij})$ is denoted by $A \otimes B$ and is defined by the matrix:*

$$A \otimes B = \begin{pmatrix} a_{11}B & \ldots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \ldots & a_{mn}B \end{pmatrix} \in \mathbb{R}^{mp \times nq}$$

Lemma 2.2.13. *Consider a $m \times n$-matrix $A = (a_{ij})$ and a $p \times q$-matrix $B = (b_{ij})$, we have:*

$$A^T \otimes B^T = (A \otimes B)^T$$

Proof. By direct computation, we get:

$$\begin{aligned}
(A \otimes B)^T &= \begin{pmatrix} a_{11}\begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots \\ b_{p1} & b_{p2} & \ldots & {}_{pq} \end{pmatrix} & \ldots & a_{1n}\begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots \\ b_{p1} & b_{p2} & \ldots & {}_{pq} \end{pmatrix} \\ \vdots & \ddots & \vdots \\ a_{m1}\begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots \\ b_{p1} & b_{p2} & \ldots & {}_{pq} \end{pmatrix} & \ldots & a_{mn}\begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots \\ b_{p1} & b_{p2} & \ldots & {}_{pq} \end{pmatrix} \end{pmatrix}^T \\
&= \begin{pmatrix} a_{11}\begin{pmatrix} b_{11} & b_{21} & \ldots & b_{p1} \\ \vdots & \ddots & \vdots & \vdots \\ b_{1q} & b_{2q} & \ldots & b_{pq} \end{pmatrix} & \ldots & a_{m1}\begin{pmatrix} b_{11} & b_{21} & \ldots & b_{p1} \\ \vdots & \ddots & \vdots & \vdots \\ b_{1q} & b_{2q} & \ldots & b_{pq} \end{pmatrix} \\ \vdots & \ddots & \vdots \\ a_{1n}\begin{pmatrix} b_{11} & b_{21} & \ldots & b_{p1} \\ \vdots & \ddots & \vdots & \vdots \\ b_{1q} & b_{2q} & \ldots & b_{pq} \end{pmatrix} & \ldots & a_{mn}\begin{pmatrix} b_{11} & b_{21} & \ldots & b_{p1} \\ \vdots & \ddots & \vdots & \vdots \\ b_{1q} & b_{2q} & \ldots & b_{pq} \end{pmatrix} \end{pmatrix}
\end{aligned}$$

$$\begin{aligned} &= \begin{pmatrix} a_{11}B^T & \dots & a_{m1}B^T \\ \vdots & \ddots & \vdots \\ a_{1n}B^T & \dots & a_{mn}B^T \end{pmatrix} \\ &= A^T \otimes B^T \end{aligned}$$

□

Theorem 2.2.14. *Let A a $m \times n$-matrix , B a $p \times q$-matrix and C a $m \times p$ matrix, the matrix equation:*

$$AXB = C$$

with X an unknown $n \times p$-matrix, is equivalent tot the system of qm equations and np unkonws given by:

$$(B^T \otimes A)\,\mathrm{vec}(X) = \mathrm{vec}(C)$$

so:

$$\mathrm{vec}(AXB) = (B^T \otimes A)\,\mathrm{vec}(X)$$

Proof. Let Q_k be the kth column of a given matrix Q, we get:

$$\begin{aligned} (AXB)_k &= A(XB)_k \\ &= AXB_k \\ &= A\sum_{i=1}^{p} b_{ik}X_i \\ &= Ab_{1k}X_1 + Ab_{2k}X_2 + \cdots + Ab_{pk}X_p \\ &= \begin{pmatrix} b_{1k}A & b_{2k}A & \dots & b_{pk}A \end{pmatrix} \mathrm{vec}(X) \\ &= (B_k^T \otimes A)\,\mathrm{vec}(X) \end{aligned}$$

We get:

$$\mathrm{vec}(AXB) = \begin{pmatrix} B_1^T \otimes A \\ \vdots \\ B_q^T \otimes A \end{pmatrix} \mathrm{vec}(X)$$

But it follows immediately that:

$$\mathrm{vec}(AXB) = (B^T \otimes A)\,\mathrm{vec}(X)$$

because the transpose of a column of B is a row of B^T. □

Theorem 2.2.15. *Let $\mathscr{G}$ and $\mathscr{H}$ be two graphs with adjacency matrices $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ and $B = [b_{ij}] \in \mathbb{R}^{m\times m}$ and let $S^{(0)} > 0$ be an initial positive matrix, and define:*

$$S^{(k+1)} = \frac{BS^{(k)}A^T + B^TS^{(k)}A}{\|BS^{(k)}A^T + B^TS^{(k)}A\|_F}, \quad k = 0, 1, \dots.$$

Then the matrix subsequences $S^{(2k)}$ and $S^{(2k+1)}$ converge to S_{even} and S_{odd} and among all the matrices in the set of all possible limits:

$$\{S_{even}(S^{(0)}), S_{odd}(S^{(0)}) : S^{(0)} > 0\}$$

the matrix $S_{even}(J)$, with J the all-one matrix, is the unique matrix of largest 1-norm.

Proof. We first rewrite the compact form of 2.11:

$$\begin{aligned} Z^{(k+1)} &= BZ^{(k)}A^T + B^TZ^{(k)}A \\ \Leftrightarrow \operatorname{vec}(Z^{(k+1)}) &= \operatorname{vec}(BZ^{(k)}A^T + B^TZ^{(k)}A) \\ \Leftrightarrow \operatorname{vec}(Z^{(k+1)}) &= \operatorname{vec}(BZ^{(k)}A^T) + \operatorname{vec}(B^TZ^{(k)}A) \\ \Leftrightarrow \operatorname{vec}(Z^{(k+1)}) &= \left((A^T)^T \otimes B\right)\operatorname{vec}(Z^{(k)}) + \left(A^T \otimes B^T\right)\operatorname{vec}(Z^{(k)}) \\ \Leftrightarrow \operatorname{vec}(Z^{(k+1)}) &= (A \otimes B + A^T \otimes B^T)\operatorname{vec}(Z^{(k)}) \end{aligned}$$

If we define $\mathbf{z}^{(k)} = \operatorname{vec}(Z^{(k)})$ and $M = A \otimes B + A^T \otimes B^T$ we get:

$$\mathbf{z}^{(k+1)} = M\mathbf{z}^{(k)},$$

exactly the same compact form as introduced in the beginning of this section in 2.3. If we can prove that M is symmetric and nonnegative, then we can apply Theorem 2.2.10 and the result follows. M is, of course, nonnegative because the adjacency matrices A and B are always nonnegative. To see that M is symmetric, first notice that M can be rewritten using Lemma 2.2.13 to:

$$M = A \otimes B + (A \otimes B)^T, \tag{2.12}$$

because $A \otimes B$ is a $nm \times nm$ square matrix, we know that M is symmetric because it's the sum of a square matrix and it's transpose. In order to stay consistent with the Eucledian norm appearing in Theorem 2.2.10, we use as matrix norm the Frobenius norm. After all we know from section 1.3.2 that for $A \in \mathbb{R}^{1\times n}$ the frobenius norm equals the 2-norm. We can now apply Theorem 2.2.10 and the result follows. □

Definition 2.2.16. *Let $\mathscr{G}, \mathscr{H}$ be two graphs, then the unique matrix*

$$S = S_{even}(\mathbf{1}) = \lim_{k\to+\infty} S^{(2k)}$$

(with the notations of the previous theorem) is the ***similarity matrix*** *between $\mathscr{G}$ and $\mathscr{H}$.*

Notice that it follows from 2.11 that the similarity matrix between $\mathscr{H}$ and $\mathscr{G}$ is the transpose of the similarity matrix between $\mathscr{G}$ and $\mathscr{H}$.

2.2.3 Algorithm

Data:
A: the $n \times n$ adjacency matrix of a directed graph $\mathscr{G}$
B: the $m \times m$adjacency matrix of a directed graph $\mathscr{H}$.
TOL: tolerance for the estimation error.
Result:
S: the similarity matrix between $\mathscr{G}$ and $\mathscr{H}$.
begin `similarity_matrix`(A,B,TOL)
  $k = 1$;
  $Z^{(0)} = \mathbf{1}$ ($n \times m$-matrix with all entries equal to 1);
  $\mu = n \times m$-matrix with all entries equal to TOL;
  repeat
    $Z^{(k)} = \frac{BZ^{(k-1)}A^T + B^T Z^{(k-1)}A}{\|BZ^{(k-1)}A^T + B^T Z^{(k-1)}A\|_F}$;
    $k = k + 1$;
  until k *is even and* $|Z^{(k)} - Z^{(k-2)}| < \mu$;
  return $Z^{(k)}$;
end

Algorithm 5: Algorithm for calculating the similarity matrix between $\mathscr{G}$ and $\mathscr{H}$

The compact form introduced in the previous section leads directly to the approximation algorithm 5. Note that the algorithm must iterate an even number of times and to check if the tolerance limit is reached, only the even iteration steps are considered because the results will oscillate between the even and odd iterates. This is of course an obvious consequence of the definition of the similarity matrix and Theorem 2.2.10. Remember that the absolute value that is mentioned is the same as in Notation 1.1.8. A Matlab implementation of the algorithm can be found in Listing A.2 in Appendix A.

Equivalence with the Power Method

Algorithm 5 is in fact completely equivalent to the Power Method of section 1.4.2. To see this, remind that we are in fact calculating the expression of Theorem 2.2.10:

$$\mathbf{s}_{\text{even}}(\mathbf{s}^{(0)}) = \lim_{k\to\infty} \mathbf{s}^{(2k)} = \frac{\Pi \mathbf{s}^{(0)}}{||\Pi \mathbf{s}^{(0)}||_2},$$

when we compare this with equation (1.21) from the Power method section, we see that in both cases we are in fact calculating the largest eigenvector of a matrix, but with the difference that in this Π is constructed in such way that it oscillates and the even iterates always converge. The attentive reader will note that Π doesn't appear directly in Algorithm 5, but this is, of course, a consequence of the vectorization defined in Definition 2.2.11. With this operation, we are able to work directly with matrices while proving convergence through Theorem 2.2.10. With this in mind, it's clear that Algorithm 5 can be seen as a *matrix analogue* of the Power method.

This brings to one of the main ideas of this master thesis: when we use the important Theorem 2.2.10 to prove convergence of some algorithm (all the algorithms that will follow are using this theorem for this purpose), we know for sure that the resulting algorithm will somehow be equivalent to the Power Method because this theorem shows that the algorithm

essentially comes down to finding the largest eigenvalue of some orthogonal projector Π, altough this Π might not appear directly in the algorithm thanks to vectorization . Depending on the possible characteristics of Π, this makes it possible to derive the computational costs of the constructed algorithm.

Also note that in this case, the equivalence is very clear when looking at the (pseudo)code for both algorithms, but as this master thesis proceeds, the algorithms will get more complicated and the equivalence with the Power Method will not be as clear.

Computational cost

We already showed in Theorem 2.2.10 that the convergence of the even iterates in the above recurrence relation is linear with ratio $(\mu/\rho)^{2n}$, where ρ is the spectral radius of M and μ is the spectral radius of the matrix M_μ, a matrix where the columns span the subspaces of all eigenvectors besides ρ and also $-\rho$ when $-\rho$ is an eigenvalue of M.

With the accuracy level TOL, we write:

$$\left|\frac{\mu}{\rho}\right|^{2n} \approx \text{TOL}$$

So, for example, for $\text{TOL} = 10^{-5}$ we get $n = -5/(2\log\mu/\lambda)$, we become:

$$\texttt{similarity\_matrix} \in O\left(\frac{1}{\log\mu - \log\rho}\right)$$

Note that this O holds for any estimation error TOL of the form 10^{-e} with $e \in \mathbb{N}$. Also note that this is completely analogous to the computational cost of the power method, which comes as no surprise since the algorithms are similar, but remember from Theorem 2.2.4 that we cannot encounter a situation in which $\mu = \rho$, as μ is the spectral radius of the spectrum without ρ and $-\rho$ (if $-\rho$ is an eigenvalue). When M_μ is trivial (meaning that M has only eigenvalues ρ and possibly $-\rho$), we conclude from equation (2.8) in Theorem 2.2.10 that in this case $\left|\frac{1}{\rho}\right|^{2n} \approx \text{TOL}$ so:

$$\texttt{similarity\_matrix} \in O\left(\frac{1}{\log 1 - \log\rho}\right).$$

The algorithm can always be calculated in a finite number of steps.

2.2.4 Special cases

We now consider some special cases of of similarity scores between two vertices of graphs.

HITS-algorithm

Remember that the HITS-algorithm assumed that

$$M = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix},$$

has a unique dominant eigenvalue to calculate the hub and authority scores of nodes in a graph $\mathscr{G}_\sigma$ with adjacency matrix B. This assumption is a direct result of the power

method, because it's one of the conditions to apply the algorithm. One of our goals was to drop this assumption. We finally arrived at this result, that also returns a solution when the Perron root has a multiplicity larger then 1. To prive this result, we need the singular value decomposition of real matrices. A comprehensive overview of this decomposition can be found in [Tom13], we restrict ourself to the main result of this decomposition, without the intuitive explanations, otherwise we would eviate too much from the main subject.

Theorem 2.2.17. ***(Singular value decomposition)*** *If A is a real $m \times n$-matrix, then there exist orthogonal matrices:*

$$\begin{aligned} U &= [\mathbf{u}_1, \ldots, \mathbf{u}_m] \in \mathbb{R}^{m \times m} \\ V &= [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in \mathbb{R}^{n \times n}, \end{aligned}$$

such that

$$U^T A V = \Sigma = diag(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m \times n},$$

meaning that Σ is a so called pseudo-diagonal matrix meaning that the first diagonal elements are the ***singular values*** *$\sigma_1, \ldots, \sigma_p$ and all other elements of Σ are zero, $p = \min(m, n)$ and $\sigma_1 \geq \cdots \geq \sigma_p$. Equivalently:*

$$A = U \Sigma V^T.$$

The columns of V are ***right singular vectors*** *of A, and those of U are* ***left singular vectors****.*

Proof. Let $\mathbf{x}$ and $\mathbf{y}$ be unit vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, and consider the bilineair form:

$$z = \mathbf{y}^T A \mathbf{x}.$$

The set

$$\mathcal{S} = \{\mathbf{x}, \mathbf{y} \mid \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m, \|\mathbf{x}\|_2 = \|\mathbf{y}\|_2 = 1\}$$

is compact, so that the scalar function $z(\mathbf{x}, \mathbf{y})$ must achieve a maximum value on S (possibly at more than one point). Let $\mathbf{u}_1$, $\mathbf{v}_1$ be two unit vectors in $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively where this maximum is achieved, and let σ_1 be the corresponding value of z:

$$\max_{\|\mathbf{x}\|_2 = \|\mathbf{y}\|_2 = 1} \mathbf{y}^T A \mathbf{x} = \mathbf{u}_1^T A \mathbf{v}_1 = \sigma_1.$$

We want to show that $\mathbf{u}_1$ is parallel to the vector $A\mathbf{v}_1$: if this were not the case, the inner product $\mathbf{u}_1^T A \mathbf{v}_1$ could be increased by rotating $\mathbf{u}_1$ towards the direction of $A\mathbf{v}_1$, thereby contradicting the fact that $\mathbf{u}_1^T A \mathbf{v}_1$ is a maximum. Similarly, notice that:

$$\mathbf{u}_1^T A \mathbf{v}_1 = \mathbf{v}_1^T A^T \mathbf{u}_1$$

and repeating the argument above, we see that $\mathbf{v}_1$ is parallel to $A^T \mathbf{u}_1$. The vectors $\mathbf{u}_1$ and $\mathbf{v}_1$ can be extended into orthonormal bases for $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively. Collect these orthonormal basis vectors into orthogonal matrices U_1 and V_1. Then:

$$U_1^T A V_1 = S_1 = \begin{pmatrix} \sigma_1 & \mathbf{0}^T \\ \mathbf{0} & A_1 \end{pmatrix}.$$

In fact, the first column of AV_1 is $A\mathbf{v}_1 = \sigma_1 \mathbf{u}_1$, so the first entry of $U_1^T A V_1$ is $\mathbf{u}_1^T \sigma_1 \mathbf{u}_1 = \sigma_1$, and its other entries are $\mathbf{u}_j^T A \mathbf{v}_1 = 0$ because $A\mathbf{v}_1$ is parallel to $\mathbf{u}_1$ and therefore orthogonal,

by construction, to $\mathbf{u}_2, \ldots, \mathbf{u}_m$. A similar argument shows that the entries after the first row of S_1 are zero: the row vector $\mathbf{u}_1^T A$ is parallel to $\mathbf{v}_1^T$ and therefore orthogonal to $\mathbf{v}_2, \ldots, \mathbf{v}_n$, so that $\mathbf{u}_1^T A\mathbf{v}_2 = \cdots = \mathbf{u}_1^T A\mathbf{v}_n = 0$. The matrix A_1 has one fewer row and column than A. We can repeat the same construction on A_1 and write

$$U_2^T A_1 V_2 = S_2 = \begin{pmatrix} \sigma_2 & \mathbf{0}^T \\ \mathbf{0} & A_2 \end{pmatrix}$$

so that

$$\begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & U_2^T \end{pmatrix} U_1^T A V_1 \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & V_2^T \end{pmatrix} = \begin{pmatrix} \sigma_1 & 0 & \mathbf{0}^T \\ 0 & \sigma_2 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & A_2 \end{pmatrix}.$$

this procedure can be repeated until A_k vanishes to obtain:

$$U^T A V = \Sigma,$$

where U^T and V are orthogonal matrices obtained by multiplying together all the orthogonal matrices used in the procedure, and:

$$\Sigma = (\sigma_1, \ldots, _p).$$

Since matrices U and V are orthogonal, we can multiply with U and V^T to obtain:

$$A = U\Sigma V^T,$$

which is the desired result.

It remains to show that the elements on the diagonal of Σ arrange in decreasing order. To see that $\sigma_1 \geq \cdots \geq \sigma$ where $p = \min(m, n)$ we can observe that the successive maximization problems that yield $\sigma_1, \ldots, \sigma_p$ are performed on a sequence of sets each of which contains the next. □

Theorem 2.2.18. *Let $\mathscr{G}$ be a graph with adjacency matrix B. The normalized hub and authority scores of the vertices are given by the normalized dominant eigenvectors of the matrices BB^T and B^TB, provided the corresponding Perron root is of multiplicity 1. Otherwise, it is the normalized projection of the vector $\mathbf{1}$ on the eigenspace of the Perron root.*

Proof. The corresponding matrix M is:

$$M = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix}$$

so:

$$M^2 = \begin{pmatrix} BB^T & 0 \\ 0 & B^TB \end{pmatrix},$$

and the result follows from Theorem 2.2.10 under the condition that the matrix M^2 has a dominant root ρ^2 . This can be seen as follows: let V and U be the dominant right and left singular vectors of B, so:

$$BV = \rho U, \quad B^T U = \rho V$$

then clearly V and U are also the dominant right and left singular vectors of B^TB and BB^T so:

$$B^TBV = \rho^2 V, \quad BB^TU = \rho^2 U.$$

The projectors associated with the dominant eigenvalues of BB^T and B^TB are, respectively:

$$\Pi_v = VV^T \quad \text{and} \quad \Pi_u = UU^T,$$

the projector of Π of M^2 is then:

$$\Pi = \text{diag}(\Pi_v, \Pi_u),$$

and hence the subvectors of $\Pi\mathbf{1}$ are the vectors $\Pi_v\mathbf{1}$ and $\Pi_u\mathbf{1}$, which can be computed with B^TB and BB^T. □

Central scores

As for the hub and authority scores we can give an explicit expression for the similarity score with vertex 2, which we will call the *central score*.

Theorem 2.2.19. *Let $\mathscr{G}$ be a graph with adjacency matrix B. The normalized similarity scores of vertex 2 of the path graph $1 \to 2 \to 3$ with the vertices of graph $\mathscr{G}$ are called the central scores and are given by the normalized dominant eigenvector of the matrix B^TB+BB^T, provided the corresponding Perron root is of multiplicity 1. Otherwise, it is the normalized projection of the vector $\mathbf{1}$ on the dominant invariant subspace.*

Proof. Let

$$M = \begin{pmatrix} 0 & B & 0 \\ B^T & 0 & B \\ 0 & B^T & 0 \end{pmatrix}$$

so:

$$M^2 = \begin{pmatrix} BB^T & 0 & BB \\ 0 & B^TB + BB^T & 0 \\ B^TB^T & 0 & B^TB \end{pmatrix}$$

because M is nonnegative and symmetric, the result follows from Theorem 2.2.10 under the condition that the central matrix $B^TB + BB^T$ has a dominant root ρ^2 of M^2. We can state this condition otherwise, because M can be permuted to:

$$M = P^T \begin{pmatrix} 0 & E \\ E^T & 0 \end{pmatrix} P, \quad \text{where } E = \begin{pmatrix} B \\ B^T \end{pmatrix}$$

Now let V and U be the dominant right and left singular vectors of E:

$$EV = \rho U, E^TU = \rho V,$$

then clearly V and U are also dominant right and left singular vectors of E^TE and EE^T, since:

$$E^TEV = \rho^2 V, \quad EE^TU = \rho^2 U.$$

Moreover,

$$PM^2P^T = \begin{pmatrix} EE^T & 0 \\ 0 & E^TE \end{pmatrix},$$

and let

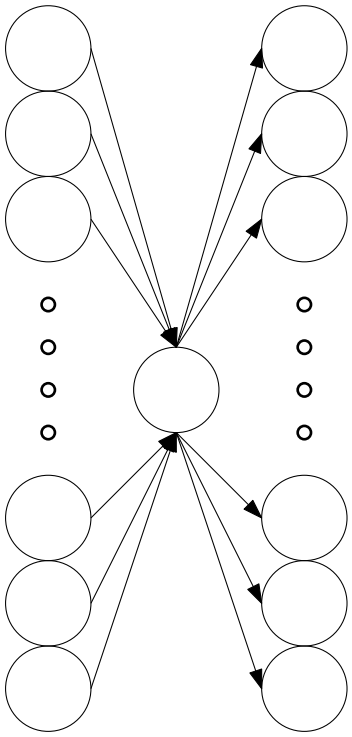
$$\Pi_v = VV^T \quad \text{and} \quad \Pi_u = UU^T$$

be the projectors associated with the dominant eigenvalues of EE^T and E^TE. The projector Π of M^2 is then equal to:

$$\Pi = P^T \text{diag}(\Pi_v, \Pi_u)P,$$

and it follows that the subvectors of $\Pi\mathbf{1}$ are te vectors $\Pi_v\mathbf{1}$ and $\mathbf{\Pi_u 1}$, which can be computed from E^TE or EE^T. Since $E^TE = B^TB + BB^T$, the central vector $\Pi_v\mathbf{1}$ is the central vector of $\Pi\mathbf{1}$. □

Example 2.2.20. In order to illustrate the intuitive meaning of calculating a similarity matrix where the path graph $1 \to 2 \to 3$ is the structure graph, consider the following directed bow-tie graph:



Label the center vertex first, then the n right vertices and finally the m left vertices, then the adjacency matrix of the bow-tie graph becomes:

$$B = \left(\begin{array}{c|ccc|ccc} 0 & 0 & \dots & 0 & 1 & \dots & 1 \\ \hline 1 & & & & & & \\ \vdots & & 0_{n\times n} & & & 0_{n\times m} & \\ 1 & & & & & & \\ \hline 0 & & & & & & \\ \vdots & & 0_{m\times n} & & & 0_{m\times m} & \\ 0 & & & & & & \end{array}\right)$$

By direct computation, we get that the matrix $B^TB + BB^T$ is equal to:

$$B^TB + BB^T = \begin{pmatrix} m+n & 0 & 0 \\ 0 & 1_{n\times n} & 0 \\ 0 & 0 & 1_{m\times m} \end{pmatrix}$$

By Theorem 2.2.19, the Perron root of M is equal to $\rho = \sqrt{n+m}$ because the central matrix $B^TB + BB^T$ has a dominant root ρ^2 of M^2 and we clearly see that if we would take the characteristic polynomial of $B^TB + BB^T$, we would get $m+n$ as Perron root. The dominant eigenvector of $B^TB + BB^T$ is equal to the similarity score $\mathbf{s}_2$ by Theorem 2.2.19. It's now easy to see that:

$$\mathbf{s}_2 = \begin{pmatrix} 1 \\ \hline 0_n \\ \hline 0_m \end{pmatrix}$$

If we see vertex 2 of the path graph $1 \to 2 \to 3$ as the *center*, which can be seen as a vertex through which much information is passed, then it is not surprising that $\mathbf{s}_2$ indicates that vertex 1 of the directed bow-tie graph is the only one that looks like a center. The left vertices of the bow-tie graph look like vertex 1 of the path graph and the right vertices of the bow-tie graph look like vertex 3. This is a beautiful example because it confirms our intuition.

Self-similarity of a graph

When we compute the similarity matrix of two equal graphs $\mathscr{G} = \mathscr{H}$, the similarity matrix S is square matrix with as entries the similarity scores between vertices of $\mathscr{G}$, we call S the *self-similarity matrix of* $\mathscr{G}$ in this case.

Intuitively, we expect that vertices have the highest similarity scores with themselves, which means that the largest entries are located on the diagonal of S. We prove in the next theorem that the largest entry of a self-similarity matrix appears always on the diagonal and that, except for some special cases, the diagonal elements of a self-similarity matrix are nonzero. This doesn't mean that the diagonal elements are always larger than the other elements on the same row or column and we conclude this paragraph with some easy examples the show this.

That the similarity matrix of a graph with itself is not always diagonally dominant is a serious limitation on the intuitive concept of similarity between graphs! It shows that the algorithm is sometimes not capable to detect the essential link between a vertex and itself, which is one of the reasons why the presented algorithm of similarity is still not totally satisfactory.

To prove this statements, we first need to dive a little bit in the world of the symmetric, positive semidifinite matrices with some definitions and some proofs.

Definition 2.2.21. *Let A be an $n \times n$-matrix, the determinant of a $k \times k$ principal submatrix (see Definition 1.2.7) is called a* ***principal minor of order k****.*

Definition 2.2.22. *A $n \times n$, symmetric matrix A is called* ***positive semidefinite*** *if all eigenvalues of A are nonnegative.*

Theorem 2.2.23. *For a $n \times n$, symmetric matrix A, the following properties are equivalent:*

1. *A is positive semidefinite,*
2. *$A = U^T U$ for some matrix U,*
3. *$\mathbf{x}^T A\mathbf{x} \geq 0$ for every $\mathbf{x} \in \mathbb{R}^n$,*
4. *All principal minors A are nonnegative.*

Proof. (1) $\Rightarrow$ (2) Write $A = UDU^T$ by Theorem 2.1.2 where U is orthogonal and D diagonal. Now, we know from that theorem, that the entries on the diagonal of D are the eigenvalues of A (which are nonnegative), so we may write $D = C^2$ where C will also be a diagonal matrix. Then we have

$$A = UCCU^T = UC(UC)^T,$$

as desired.

(2) $\Rightarrow$ (3) For every $\mathbf{x} \in \mathbb{R}^n$ we have

$$\mathbf{x}^T A\mathbf{x} = \mathbf{x}^T U^T U\mathbf{x} = (U\mathbf{x})^T (U\mathbf{x}) \geq 0.$$

(3) $\Rightarrow$ (1) If $\mathbf{v}$ is an eigenvector of A with eigenvalue λ then $0 \leq \mathbf{v}^T A\mathbf{v} = \lambda \mathbf{v}^T \mathbf{v}$ which implies that $\lambda \geq 0$.

(2) $\Rightarrow$ (4) Let B be a submatrix of A formed by deleting the rows and columns with indices in the set S. Then modify U to form V by deleting the columns with indices in the set S. Now: $\det(B) = \det(V^T V) = \det(V)^2 \geq 0$. Remember that U is orthogonal from and thus V is orthogonal too.

(4) $\Rightarrow$ (1) By contraposition, we may assume that A has a unit eigenvector $\mathbf{v}$ with eigenvalue $\lambda \leq 0$. If A has only one eigenvalue ≤ 0, then $\det(A) < 0$ and we are done. Otherwise, choose a unit eigenvector $\mathbf{u}$ orthogonal to $\mathbf{v}$ with eigenvalue $\mu \leq 0$. Now choose $\alpha \in \mathbb{R}$ so that the vector $\mathbf{w} = \mathbf{v} + \alpha\mathbf{u}$ has at least one zero coordinate, say w_i. If A' is the matrix obtained from A by removing the column i and row i and $\mathbf{w}'$ is obtained by removing coordinate i of $\mathbf{w}$, then we have

$$(\mathbf{w}')^T A'\mathbf{w}' = \mathbf{w}^T A\mathbf{w} = \lambda + \alpha^2 \mu < 0,$$

so A' is not semidefinite since we already demonstrated the equivalence between (3) and (1), and the result follows by induction. □

Lemma 2.2.24. *The scaled sum of two positive semidefinite matrix is again a positive semidefinite matrix.*

Proof. Assume that A and B are $n \times n$, positive semidefinite matrices and $\alpha, \beta \in \mathbb{R}^+$, so for all $\mathbf{x} \in \mathbb{R}^n$ we have $\mathbf{x}^T A\mathbf{x} \leq 0$, so then:

$$\mathbf{x}^T(\alpha A + \beta B)\mathbf{x} = \alpha(\mathbf{x}^T A\mathbf{x}) + \beta(\mathbf{x}^T B\mathbf{x}) \geq 0.$$

□

Theorem 2.2.25. *The self-similarity matrix of a graph $\mathscr{G}$ is positive semidefinite. The largest entry of the matrix appears on the diagonal and if a diagonal entry is equal to zero, all the entries of the corresponding row and column are also equal to zero.*

Proof. Since $A = B$, the compact form of Theorem 2.2.15 becomes:

$$S^{(k+1)} = \frac{AS^{(k)}A^T + A^TS^{(k)}A}{\|AS^{(k)}A^T + A^TS^{(k)}A\|_F}, \quad S^{(0)} = J,$$

First notice that $S^{(k+1)}$ is in this case always symmetric because $s_{ij} = s_{ji}$ (the similarity scores of vertex v_i and v_j are the same as the similarity scores of vertex v_j and v_i). The all-one matrix J is clearly positive semidefinite as it has only nonnegative principal minors. So $S^{(0)}$ is postive semidefinite, so it can be decomposed as $S^{(0)} = W^TW$. We write for some vector $\mathbf{x} \in \mathbb{R}^n$:

$$\begin{aligned} \mathbf{x}^TA^TS^{(0)}A\mathbf{x} &= \mathbf{x}^TA^TW^TWA\mathbf{x} \\ &= \|WA\mathbf{x}\|_2 \\ &\geq 0, \end{aligned}$$

similarly, $\mathbf{x}^TAS^{(0)}A^T\mathbf{x} \geq 0$, thus all matrices $S^{(k)}$ are positive semidefinite because the (scaled) sum of two positive semidefinite matrices is also positive semidefinite. Now, the rest of the proof follows from the following observation: let α be a real number. If $\mathbf{x} = \alpha e_i - e_j$ then:

$$\mathbf{x}^tS^{(k)}\mathbf{x} = \alpha^2 s_{ii}^{(k)} - 2\alpha s_{ij}^{(k)} + s_{jj}^{(k)}, \tag{2.13}$$

so to prove that the largest entry of the matrix $S^{(k)}$ appears on the diagonal, assume, contrapositively, that the strictly largest entry is $s_{ij}^{(k)} = s_{ji}^{(k)}$ with $i \neq j$. Taking $\alpha = 1$, equation (2.13) becomes:

$$\mathbf{x}^tS^{(k)}\mathbf{x} = s_{ii}^{(k)} - 2s_{ij}^{(k)} + s_{jj}^{(k)} = (s_{ii}^{(k)} - s_{ij}^{(k)}) + (s_{jj}^{(k)} - s_{ij}^{(k)}) < 0.$$

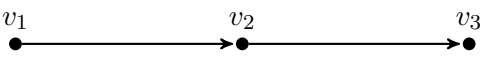
To prove the last statement, if $s_{ii}^{(k)} = 0$ but $s_{ij}^{(k)} \neq 0$ for some j gives for (2.13):

$$\mathbf{x}^tS^{(k)}\mathbf{x} = -2\alpha s_{ij}^{(k)} + s_{jj}^{(k)},$$

which is negative for α large enough.

□

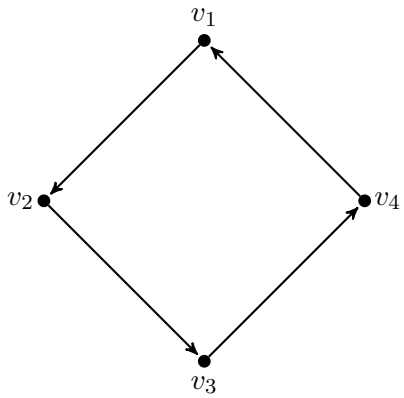
Example 2.2.26. The self-similaritiy matrix of the graph:



is equal to:

$$\begin{pmatrix} 0.5774 & 0 & 0 \\ 0 & 0.5774 & 0 \\ 0 & 0 & 0.5774 \end{pmatrix}$$

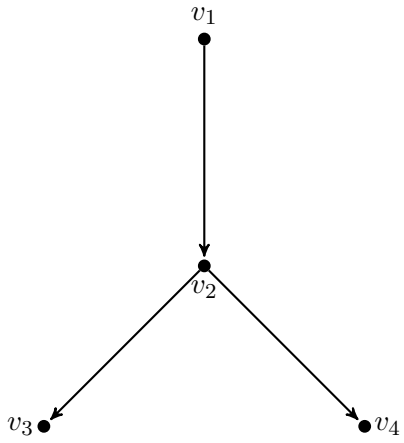
Example 2.2.27. The self-similartiy matrix of the graph:



is equal to:

$$\begin{pmatrix} 0.250 & 0.250 & 0.250 & 0.250 \\ 0.250 & 0.250 & 0.250 & 0.250 \\ 0.250 & 0.250 & 0.250 & 0.250 \\ 0.250 & 0.250 & 0.250 & 0.250 \end{pmatrix}$$

Example 2.2.28. The self-similartiy matrix of the graph:



is equal to:

$$\begin{pmatrix} 0.4082 & 0 & 0 & 0 \\ 0 & 0.4082 & 0 & 0 \\ 0 & 0 & 0.4082 & 0.4082 \\ 0 & 0 & 0.4082 & 0.4082 \end{pmatrix}$$

Undirected graphs

In the case of undirected graphs, the compact form 2.2.15 only becomes easier and equals:

Theorem 2.2.29. *Let $\mathscr{G}$ and $\mathscr{H}$ be two undirected graphs with adjacency matrices $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ and $B = [b_{ij}] \in \mathbb{R}^{m\times m}$ and let $S^{(0)} > 0$ be an initial positive matrix, and define:*

$$S^{(k+1)} = \frac{BS^{(k)}A}{\|BS^{(k)}A\|_F}, \quad k = 0, 1, \ldots.$$

Then the matrix subsequences $S^{(2k)}$ and $S^{(2k+1)}$ converge to S_{even} and S_{odd} and among all the matrices in the set of all possible limits:

$$\{S_{even}(S^{(0)}), S_{odd}(S^{(0)}) : S^{(0)} > 0\}$$

the matrix $S_{even}(J)$ is the unique matrix of largest 1-norm.

Proof. This follows easily from the fact that in the case of undirected graphs, A and B are symmetric:

$$\begin{aligned} S^{(k+1)} &= \frac{BS^{(k)}A^T + B^TS^{(k)}A}{\|BS^{(k)}A^T + B^TS^{(k)}A\|_F} \\ &= \frac{BS^{(k)}A + BS^{(k)}A}{\|BS^{(k)}A + BS^{(k)}A\|_F} \\ &= \frac{2BS^{(k)}A}{\|2BS^{(k)}A\|_F} \\ &= \frac{BS^{(k)}A}{\|BS^{(k)}A\|_F} \end{aligned}$$

□

2.3 Node-edge similarity

The paper of Blondel et al. [BGH$^+$04] caused a flow of successive papers which build upon the concept of similarity between two graphs. For instance, the paper 'Graph similarity scoring and matching' of Laura Zager and George Verghese [ZV08] expands the notion of node similarity presented in the previous section to similarity between the edges of two graphs. The paper was presented in 2006. Intuitively an edge of a graph $\mathscr{G}$ is similar to an edge of graph $\mathscr{H}$ if their *source* and *terminal nodes* are similar. As a consequence, the notion of similarity between edges introduces a coupling between edge and node similarity scores. The algorithm presented in this paper is therefore an extension of the algorithm presented in the previous section.

2.3.1 Coupled node-edge similarity scores

We now present the extended algorithm allowing us to calculate not only a node similarity scores, but also edge similarity scores. The algorithm will use a new sort of matrices that represent a graph. Recall Definition 1.5.6 of source and terminal nodes.

Definition 2.3.1. *Let be $\mathscr{G} = (V, \to)$ be a graph with adjacency matrix A, numbered vertices $v_1, v_2, \ldots, v_n \in V$ and numbered edges $e_1, e_2, \ldots e_m \in \to$. We define the* ***source-edge matrix*** *A_S as a $n \times m$ matrix with entries:*

$$(A_S)_{ij} = \begin{cases} 1 & \text{if } s_{\mathscr{G}}(e_j) = v_i \\ 0 & \text{otherwise} \end{cases},$$

the notation A_S is derived from the adjacency matrix A.

Definition 2.3.2. *Let be $\mathscr{G} = (V, \to)$ be a graph with adjacency matrix A, numbered vertices $v_1, v_2, \ldots, v_n \in V$ and numbered edges $e_1, e_2, \ldots e_m \in \to$. We define the* ***terminus-edge matrix*** *A_T as a $n \times m$ matrix with entries:*

$$(A_T)_{ij} = \begin{cases} 1 & \text{if } t_{\mathscr{G}}(e_j) = v_i \\ 0 & \text{otherwise} \end{cases},$$

the notation A_T is derived from the adjacency matrix A.

Property 2.3.3. *Let $\mathscr{G} = (V, \to)$ be a graph, then $A_S A_S^T$ is a diagonal matrix where the ith diagonal entry is equal the outdegree of vertex v_i.*

Proof. By direct computation, we get:

$$(A_S A_S^T)_{ij} = \sum_{k=1}^{m} (A_S)_{ik} (A_S)_{kj}^T = \sum_{k=1}^{m} (A_S)_{ik} (A_S)_{jk}$$

Assume $i \neq j$. Then for each k, $(A_S)_{ik}(A_S)_{jk} = 0$ since vetrex v_i and vertex v_j can't be both the source node of edge k.

When $i = j$, then for each k, $(A_S)_{ik}(A_S)_{jk} = ((A_S)_{ik})^2$ equals 0 or 1 depending on whether v_i is a starting point or not, so 1 is added to $(A_S A_S^T)_{ii}$ each time an edge 'departs' from v_i, this is exactly the outdegree of vertex v_i. □

In an analogous way, we prove:

Property 2.3.4. *Let $\mathscr{G} = (V, \rightarrow)$ be a graph, then $A_T A_T^T$ is a diagonal matrix where the ith diagonal entry is equal the indegree of vertex v_i.*

Property 2.3.5. *Let $\mathscr{G} = (V, \rightarrow)$ be a graph, then the adjacency matrix A is equal to $A_S A_T^T$.*

Proof. By direct computation, we get:

$$(A_S A_T^T)_{ij} = \sum_{k=1}^{m} (A_S)_{ik} (A_T)_{kj}^T = \sum_{k=1}^{m} (A_S)_{ik} (A_T)_{jk}$$

The terms $(A_S)_{ik}(A_T)_{jk}$ equal 1 if edge k goes from v_i to v_j and since the sum is taken over all edges we conclude:

$$(A_S A_T^T)_{ij} = A_{ij}.$$

□

Let $\mathscr{G}(V, \rightarrow)$, $\mathscr{H}(U, \rightarrow')$ be two (directed) graphs, $\mathscr{G}$ has $n_{\mathscr{G}}$ vertices and $m_{\mathscr{G}}$ edges and $\mathscr{H}$ has $n_{\mathscr{H}}$ vertices and $m_{\mathscr{H}}$. Remember the following updating equation from (2.5), which returns the (node) similarity score between vertices u_i from $\mathscr{H}$ and v_j from $\mathscr{G}$:

$$x_{ij}^{(k+1)} = \sum_{r:(u_r,u_i)\in\rightarrow', w:(v_w,v_j)\in\rightarrow} x_{rw}^{(k)} + \sum_{r:(u_i,u_r)\in\rightarrow', w:(v_j,v_w)\in\rightarrow} x_{rw}^{(k)},$$

if we number the edges of $\mathscr{G}$ and $\mathscr{H}$ as $e_1, e_2, \ldots e_{m_{\mathscr{G}}} \in \rightarrow$ and $e'_1, e'_2, \ldots, e'_{m_{\mathscr{H}}} \in \rightarrow'$, this can be rewritten as:

$$x_{ij}^{(k+1)} = \sum_{t_{\mathscr{H}}(e'_p)=u_i, t_{\mathscr{G}}(e_q)=v_j} x^{(k)}_{s_{\mathscr{H}}(e'_p) s_{\mathscr{G}}(e_q)} + \sum_{s_{\mathscr{H}}(e'_p)=u_i, s_{\mathscr{G}}(e_q)=v_j} x^{(k)}_{t_{\mathscr{H}}(e'_p) t_{\mathscr{G}}(e_q)}.$$

We now extend this mutually reinforcing relation with the notion of edge similarity. x_{ij} denotes again the node similarity score between vertex u_i from $\mathscr{H}$ and vertex v_j from $\mathscr{G}$ and y_{pq} denotes the edge similarity score between edge p from $\mathscr{H}$ and edge q in $\mathscr{G}$, the update equations for edge and node similarity scores take now the following form:

$$y_{pq}^{(k+1)} = x^{(k)}_{s_{\mathscr{H}}(e'_p) s_{\mathscr{G}}(e_q)} + x^{(k)}_{t_{\mathscr{H}}(e'_p) t_{\mathscr{G}}(e_q)} \tag{2.14}$$

$$x_{ij}^{(k+1)} = \sum_{t_{\mathscr{H}}(e'_r)=u_i, t_{\mathscr{G}}(e_w)=v_j} y_{rw}^{(k)} + \sum_{s_{\mathscr{H}}(e_t)=u_i, s_{\mathscr{G}}(e_w)=v_j} y_{rw}^{(k)} \tag{2.15}$$

With the same reasoning as presented in the previous section, these scores can be assembled into matrices $Y^{(k)}$ and $X^{(k)}$ by using the source-edge matrices A_S and the terminus-edge matrices A_T. Let A be the adjacency matrix of $\mathscr{G}$ and B be the adjacency matrix of $\mathscr{H}$, let $X^{(k)}$ be the $n_{\mathscr{H}} \times n_{\mathscr{G}}$-matrix with entries x_{ij}, the node similarity scores at iteration step k and let $Y^{(k)}$ be the $m_{\mathscr{H}} \times m_{\mathscr{G}}$-matrix with entries y_{pq}, the edge similarity scores at step k. The equations (2.14) and (2.15) can be rewritten as:

$$Y'^{(k+1)} = B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T \tag{2.16}$$

$$X'^{(k+1)} = B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T \tag{2.17}$$

for $k = 0, 1, \ldots$

Of course we want to customize these equations in a way that we can apply Theorem 2.2.10 to prove convergence. This will be completely analogous to Theorem 2.2.15, but we can achieve a slightly better result: not only the even and odd iterates will converge, the iteration converges as a whole. Lastly, remember that one of the conditions of Theorem 2.2.10 is to normalize the results at each iteration step. Therefore, the following theorem comes as no surprise:

Theorem 2.3.6. *Let $\mathscr{G}$ and $\mathscr{H}$ be two graphs with adjacency matrices A and B, $\mathscr{G}$ has $n_{\mathscr{G}}$ vertices and $m_{\mathscr{G}}$ edges and $\mathscr{H}$ has $n_{\mathscr{H}}$ vertices and $m_{\mathscr{H}}$ edges, define:*

$$Y^{(k+1)} = \frac{B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \tag{2.18}$$

$$X^{(k+1)} = \frac{B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \tag{2.19}$$

for $k = 0, 1, \ldots$.

Then the matrix subsequences $X^{(2k)}$, $Y^{(2k)}$ and $X^{(k+1)}$, $Y^{(k+1)}$ converge to X_{even}, Y_{even} and X_{odd}, Y_{odd}. If we take[4]:

$$X^{(0)} = J \in \mathbb{R}^{n_{\mathscr{H}} \times n_{\mathscr{G}}}$$
$$Y^{(0)} = J \in \mathbb{R}^{m_{\mathscr{H}} \times m_{\mathscr{G}}}$$

as initial matrices, then $X_{even}(\mathbf{1}) = X_{odd}(\mathbf{1})$, $Y_{even}(\mathbf{1}) = Y_{odd}(\mathbf{1})$ are the unique matrices of largest 1-norm among all possible limits with positive initial matices and the matrix sequence $X^{(k)}$, $Y^{(k)}$ converges has a whole.

Proof. By Theorem 2.2.14 we can rewrite (2.16) as follows:

$$\begin{aligned}
Y'^{(k+1)} &= B_S^T X'^{(k)} A_S + B_T^T X'^{(k)} A_T \\
\Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= \operatorname{vec}(B_S^T X'^{(k)} A_S + B_T^T X'^{(k)} A_T) \\
\Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= \operatorname{vec}(B_S^T X'^{(k)} A_S) + \operatorname{vec}(B_T^T X'^{(k)} A_T) \\
\Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= (A_S^T \otimes B_S^T) \operatorname{vec}(X'^{(k)}) + (A_T^T \otimes B_T^T) \operatorname{vec}(X'^{(k)}) \\
\Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= (A_S^T \otimes B_S^T + A_T^T \otimes B_T^T) \operatorname{vec}(X'^{(k)})
\end{aligned}$$

Completely analogously we can also rewrite (2.17):

$$\operatorname{vec}(X'^{(k+1)}) = (A_S \otimes B_S + A_T \otimes B_T) \operatorname{vec}(Y^k),$$

define $\mathbf{y}^{(k)} = \operatorname{vec}(Y'^{(k+1)})$ and $\mathbf{x}^{(k)} = \operatorname{vec}(X'^{(k+1)})$, we get:

$$\begin{aligned}
\mathbf{y}^{(k+1)} &= (A_S^T \otimes B_S^T + A_T^T \otimes B_T^T)\mathbf{x}^{(k)} \\
\mathbf{x}^{(k+1)} &= (A_S \otimes B_S + A_T \otimes B_T)\mathbf{y}^{(k)}
\end{aligned}$$

[4] J is a matrix of all ones

If we define $G = A_S^T \otimes B_S^T + A_T^T \otimes B_T^T$, then with Lemma 2.2.13 and well known properties of transpose matrices:

$$\begin{aligned} G^T &= (A_S^T \otimes B_S^T + A_T^T \otimes B_T^T)^T \\ &= ((A_S \otimes B_S)^T + (A_T \otimes B_T)^T)^T \\ &= ((A_S \otimes B_S)^T)^T + ((A_T \otimes B_T)^T)^T \\ &= A_S \otimes B_S + A_T \otimes B_T \end{aligned}$$

So we get:

$$\mathbf{y}^{(k+1)} = G\mathbf{x}^{(k)} \tag{2.20}$$
$$\mathbf{x}^{(k+1)} = G^T\mathbf{y}^{(k)}, \tag{2.21}$$

G is a $m_{\mathscr{G}}m_{\mathscr{H}} \times n_{\mathscr{G}}n_{\mathscr{H}}$-matrix, the previous expressions can be concatenated to a single matrix update equation (we define matrix M and $\mathbf{z}^{(k+1)}$):

$$\mathbf{z}^{(k+1)} = \begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^{(k+1)} = \begin{pmatrix}\mathbf{0}_{m_{\mathscr{G}}m_{\mathscr{H}}} & G^T \\ G & \mathbf{0}_{n_{\mathscr{G}}n_{\mathscr{H}}}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^{(k)} = M\mathbf{z}^{(k)},$$

M is clearly nonnegative because G and G^T consists of sums of Kronecker products of A_S, B_S, A_T and/or B_T, all matrices with entries equal to zero or one, M is as a $(n_{\mathscr{G}}n_{\mathscr{H}} + m_{\mathscr{G}}m_{\mathscr{H}}) \times (n_{\mathscr{G}}n_{\mathscr{H}} + m_{\mathscr{G}}m_{\mathscr{H}})$-matrix clearly symmetric, so the result follows immediately from Theorem 2.2.10. The appearance of the Perron norm can be explained in the same way as in Theorem 2.2.15, note that we write $Y^{(k)}$ and $X^{(k)}$ for the normalized results at iteration step k. We still have to prove that odd and even iterates are the same and that M has only positive eigenvalues, meaning that the sequence converges to $\frac{\Pi\mathbf{z}^{(0)}}{\|\Pi\mathbf{z}^{(0)}\|_2}$, this can be done easily by expanding the matrix equation:

$$\mathbf{x}^{(k)} = \begin{cases}(G^TG)^{\frac{k}{2}}\mathbf{x}^{(0)} & \text{k even}\\ (G^TG)^{\frac{k-1}{2}}G^T\mathbf{y}^{(0)} & \text{k odd}\end{cases} \quad \text{and} \quad \mathbf{y}^{(k)} = \begin{cases}(GG^T)^{\frac{k}{2}}\mathbf{y}^{(0)} & \text{k even}\\ (GG^T)^{\frac{k-1}{2}}G\mathbf{x}^{(0)} & \text{k odd}\end{cases}$$

The matrix G is the sum of two matrices $(A_S^T \otimes B_S^T)$ and $(A_T^T \otimes B_T^T)$. Now, A_S^T is a $m_{\mathscr{G}} \times n_{\mathscr{G}}$-matrix and has in each row a single '1'-entry, simply because an edge has at most one source node . The same holds for B_S^T. Therefore, taking the Kronecker product of those two matrices results in the matrix $A_S^T \otimes B_S^T$ which also has just a single '1' entry in each row. With the same reasoning, we conclude that also $A_T^T \otimes B_T^T$ has also just a single '1'-entry in each row. Taking the sum of $(A_S^T \otimes B_S^T)$ and $(A_T^T \otimes B_T^T)$ results thus in the matrix G with exactly two 1 entries in each row. If we now choose the initial conditions as $\mathbf{x}^{(0)} = \mathbf{1} \in \mathbb{R}^{n_{\mathscr{H}}n_{\mathscr{G}}}$ and $\mathbf{y}^{(0)} = \mathbf{1} \in \mathbb{R}^{m_{\mathscr{H}}m_{\mathscr{G}}}$, we conclude:

$$G\mathbf{x}^{(0)} = 2\mathbf{y}^{(0)},$$

we get:

$$\mathbf{x}^{(k)} = \begin{cases}(G^TG)^{\frac{k-2}{2}}G^TG\mathbf{x}^{(0)} = \frac{1}{2}(G^TG)^{\frac{k-2}{2}}G^T\mathbf{y}^{(0)} & k \text{ even}\\ (G^TG)^{\frac{k-1}{2}}G^T\mathbf{y}^{(0)} & k \text{ odd}\end{cases}$$

$$\mathbf{y}^{(k)} = \begin{cases}(GG^T)^{\frac{k}{2}}\mathbf{y}^{(0)} & k \text{ even}\\ (GG^T)^{\frac{k-1}{2}}G\mathbf{x}^{(0)} = (GG^T)^{\frac{k-1}{2}}\frac{1}{2}\mathbf{y}^{(0)} & k \text{ odd}\end{cases}$$

First, observe that the matrices GG^T and G^TG are besides being symmetric (for example, $(GG^T)^T = GG^T$) and nonnegative, are also positive semi-definite and thus have only nonnegative eigenvalues. Note that the factor $\frac{1}{2}$ that appears in the odd iterates will be eliminated by the normalization in each step. So in the limit as $k \to \infty$, the even and odd iterates are the same.

□

We now define $X_{\text{even}}(\mathbf{1})$ as the node similarity matrix and $Y_{\text{even}}(\mathbf{1})$ as the edge similarity matrix.

2.3.2 The algorithm

Data:
A_S: the $n_{\mathscr{G}} \times m_{\mathscr{G}}$ source-edge matrix of a graph $\mathscr{G}$
A_T: the $n_{\mathscr{G}} \times m_{\mathscr{G}}$ terminal-edge matrix of a graph $\mathscr{G}$
B_S: the $n_{\mathscr{H}} \times m_{\mathscr{H}}$ source-edge matrix of a graph $\mathscr{G}$
B_T: the $n_{\mathscr{H}} \times m_{\mathscr{H}}$ terminal-edge matrix of a graph $\mathscr{G}$
TOL: tolerance for the estimation error.
Result:
X: the node similarity matrix between $\mathscr{G}$ and $\mathscr{H}$
Y: the edge similarity matrix between $\mathscr{G}$ and $\mathscr{H}$
begin `node_edge_similarity_matrix(`A_S,A_T, B_S,B_T,TOL`)`
  $k = 1$;
  $X^{(0)} = \mathbf{1}$ ($n_{\mathscr{H}} \times n_{\mathscr{G}}$-matrix with all entries equal to 1);
  $Y^{(0)} = \mathbf{1}$ ($m_{\mathscr{H}} \times m_{\mathscr{G}}$-matrix with all entries equal to 1);
  $\mu_X = n_{\mathscr{H}} \times n_{\mathscr{G}}$-matrix with all entries equal to TOL;
  $\mu_Y = m_{\mathscr{H}} \times m_{\mathscr{G}}$-matrix with all entries equal to TOL;
  repeat
    $Y^{(k)} = \frac{B_S^T X^{(k-1)} A_S + B_T^T X^{(k-1)} A_T}{\|B_S^T X^{(k-1)} A_S + B_T^T X^{(k-1)} A_T\|_F}$;
    $X^{(k)} = \frac{B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F}$;
    $k = k + 1$;
  until $|X^{(k)} - X^{(k-1)}| < \mu_X$ *and* $|Y^{(k)} - Y^{(k-1)}| < \mu_Y$;
  return $X^{(k)}, Y^{(k)}$;
end

Algorithm 6: Algorithm for calculating the node and edge similarity matrix X and Y between $\mathscr{G}$ and $\mathscr{H}$.

The algorithm to calculate the node and edge similarity scores is presented in pseudocode in Algorithm 6. Remember that the absolute value that is mentioned is the same as in Notation 1.1.8. Besides the different calculations, the main difference with the algorithm of the previous section (Algorithm 5) is that the whole sequence converges, not only the even (or odd) iterates, making this algorithm twice as fast. This is because in Algorithm 5 we take the limit of the even iterates but we need the calculations of the odd iterates too to achieve a result. In this implementation, the algorithm will not oscillate and both the even and odd iterates converge to the same limit, so the tolerance level will be satisfied with half

of the number of steps we would need for Algorithm 5. Also notice that we implement a *sequential* update: when $Y^{(k)}$ is calculated,the result is immediately used in $X^{(k)}$. This is not according to Theorem 2.3.6, which suggest *simultaneous* updating equations: in that case $X^{(k)}$ uses $Y^{(k-1)}$ in it's calculations. It is not difficult to see, by an analogous argument of Theorem 2.3.6, that both approaches yield the same set of similarity scores. Numerical analysts, however, will always prefer the sequential update procedure because it performs slightly better (see section 3.2 in [**?**]) as you directly use a more accurate result for $Y^{(k)}$ in the calculation of $X^{(k)}$.

A Matlab implementation can be found in Listing A.3 in Appendix A. Because giving source-edge matrices and terminal-edge matrices as input is quite unnatural, in Listing A.4 and Listing A.5 some Matlab code is also presented to transform an adjacency matrix into a source-edge matrix and a terminal-edge matrix. The resulting matrices represent an edge numbering left-to-right, based on the entries of the adjacency matrix. The algorithm to calculate the source-edge matrix based on the adjacency matrix is also written in pseudo-code in Algorithm 7, the calculation of the terminal-edge matrix is completely analogous. A Matlab implementation of Algorithm 6 that takes two adjacency matrices as input can be found in Listing A.6.

Data:
A: the $n \times n$ adjacency matrix of a graph $\mathscr{G}$
Result:
A_S: the source-edge matrix of graph $\mathscr{G}$
begin `source_edge_matrix(`A`)`
  m = sum of all entries of A ;
  A_S = initialize a $n \times m$-matrix with all entries equal to 0;
  current_edge = 1;
  for $i : 1$ *to* n **do**
    for $j : 1$ *to* n **do**
      if $(A)_{ij} > 0$ **then**
        for $e : 1$ *to* $(A)_{ij}$ **do**
          $(A_S)_{i,\text{current\_edge}} = 1$;
          current_edge = current_edge + 1 ;
        end
      end
    end
  end
end
return A_S;

Algorithm 7: Algorithm for calculating the source-edge matrix A_S based on the adjacency matrix A of a graph $\mathscr{G}$.

2.3.3 Difference with node similarity

It is clear that the node-edge similarity algorithm is different from Algorithm 5 from the previous section. We will make this a little more explicit and show the difference in the resulting node similarity matrix. We first need another property of the Kronecker product.

Property 2.3.7. ***(mixed-product property of Kronecker products)*** *Let $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{r\times s}, C \in \mathbb{R}^{n\times p}$ and $D \in \mathbb{R}^{s\times t}$ then:*

$$(A \otimes B)(C \otimes D) = AC \otimes BD$$

Proof.

$$\begin{aligned}
(A \otimes B)(C \otimes D) &= \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix} \begin{pmatrix} c_{11}D & \dots & c_{1p}D \\ \vdots & \ddots & \vdots \\ a_{n1}D & \dots & a_{np}D \end{pmatrix} \\
&= \begin{pmatrix} \sum_{k=1}^{n} a_{1k}c_{k1}BD & \dots & \sum_{k=1}^{n} a_{1k}c_{kp}BD \\ \vdots & \ddots & \vdots \\ \sum_{k=1}^{n} a_{mk}c_{k1}BD & \dots & \sum_{k=1}^{n} a_{mk}c_{kp}BD \end{pmatrix} \\
&= AC \otimes BD.
\end{aligned}$$

□

Now take equations (3.10) and (3.11) and consider only the even iterates. We consider only the even iterates because Algorithm 5 does, remember that in Algorithm 6 the even and odd iterates yield the same set of similarity scores. We get:

$$\begin{aligned}
\mathbf{x}^{(k)} &= (G^T G)\mathbf{x}^{(k-2)} \\
&= ((A_S \otimes B_S + A_T \otimes B_T)(A_S^T \otimes B_S^T + A_T^T \otimes B_T^T))\mathbf{x}^{(k-2)} \\
&= ((A_S \otimes B_S)(A_S^T \otimes B_S^T) + (A_S \otimes B_S)(A_T^T \otimes B_T^T) \\
&\quad +(A_T \otimes B_T)(A_S^T \otimes B_S^T) + (A_T \otimes B_T)(A_T^T \otimes B_T^T))\mathbf{x}^{(k-2)} \\
&= (A_S A_S^T \otimes B_S B_S^T + A_S A_T^T \otimes B_S B_T^T \\
&\quad +A_T A_S^T \otimes B_T B_S^T + A_T A_T^T \otimes B_T B_T^T)\mathbf{x}^{(k-2)} \\
&= (A_S A_S^T \otimes B_S B_S^T + A \otimes B + A^T \otimes B^T + A_T A_T^T \otimes B_T B_T^T)\mathbf{x}^{(k-2)} \\
&= (A \otimes B + A^T \otimes B^T + A_S A_S^T \otimes B_S B_S^T + A_T A_T^T \otimes B_T B_T^T)\mathbf{x}^{(k-2)}
\end{aligned}$$

Where A, B are the adjacency matrices of $\mathscr{G}, \mathscr{H}$ (Property 2.3.5), $A_S A_S^T, B_S B_S^T$ are the diagonal matrices with the outdegrees of the vertices on the diagonal (Property 2.3.3) and $A_T A_T^T, B_T B_T^T$ are the diagonal matrices with the indegree of the vertices on the diagonal (Property 2.3.4). The iteration sugested in Theorem 2.2.15 in the previous section has the following form (see equation (2.12)):

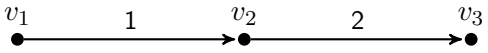
$$\mathbf{x}^{(k)} = A \otimes B + A^T \otimes B^T \mathbf{x}^{(k-2)},$$

thus Algorithm 6 differs from Algorithm 5 in three important ways:

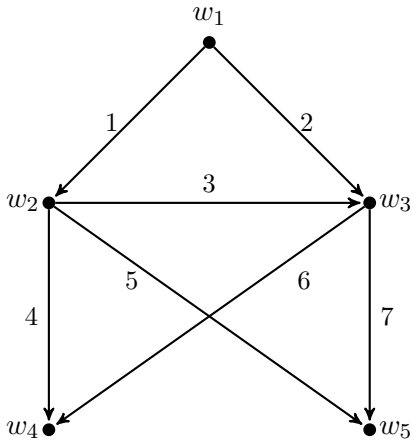
- In Algorithm 6 the inclusion of additional diagonal terms serve to amplify the scores of nodes that are highly connected in the node similarity matrix,
- Algorithm 6 returns a node similarity matrix and an edge similarity matrix, Algorithm 5 returns only a node similarity matrix,
- The whole sequence in Algorithm 6 converges, not only the even and odd iterates.

2.3.4 Example

Example 2.3.8. We retake Example 2.2.2 and number the edges as follows, let $\mathscr{G} = (V, \rightarrow)$ be :



Let $\mathscr{H} = (W, \rightarrow)$ be the following graph:



Then the node similarity matrix is:

$$X = \begin{pmatrix} 0.2338 & 0.0718 & 0 \\ 0.2472 & 0.3230 & 0.0128 \\ 0.1841 & 0.7553 & 0.3185 \\ 0 & 0.0935 & 0.2338 \\ 0 & 0.0441 & 0.0576 \end{pmatrix},$$

and the edge similarity matrix equals:

$$Y = \begin{pmatrix} 0.2166 & 0.0329 \\ 0.3847 & 0.1518 \\ 0.3899 & 0.2495 \\ 0.1325 & 0.2166 \\ 0.1133 & 0.1480 \\ 0.3653 & 0.4176 \\ 0.1080 & 0.3847 \end{pmatrix}$$

If you look at the structure of $\mathscr{H}$ and compare it with the structure of $\mathscr{G}$, then intuitively it is not surprising that edge 3 of $\mathscr{H}$ is most similar to edge 1 of $\mathscr{G}$ and edge 6 of $\mathscr{H}$ is most similar to edge 2 of $\mathscr{G}$ because they are both very central edges with source nodes and terminal nodes that are very similar (they both have high similarity scores). Also note that w_7 is considered as the most 'central node' (see subsection 2.2.4) as it has the largest similarity score for v_2, also this can be logically explained because w_3 has 2 incoming edges and 2 outgoing eges, no other edge has this in $\mathscr{H}$.

2.4 Colored graphs

In this last section, we extend the notion of similarity to colored graphs. The paper [VDF09] is written by two Belgian professors Paul Van Dooren and Catherine Fraikin from the Catholic University of Louvain in 2009.

Graph coloring is already introduced in paragraph 1.5.1 and we will construct a method that returns similarity matrices for two graphs where the coloring is on the nodes or on the edges of both graphs. If you would compare the orignal paper to explanation in this section, you will see that there are lot of differences. This has two reasons: first, with the previous sections we have already a broad overview of similarity on graphs so we can achieve results in a more detailed way, second, to make the notations uniform in the whole master thesis, this paper had to be rewritten.

2.4.1 Colored nodes

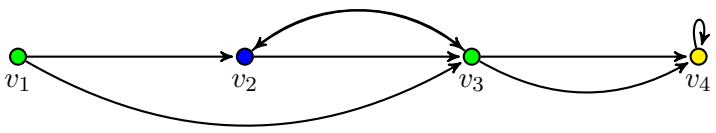
Method

We first extend the node similarity introduced in section 2.2 to node colored graphs. Take two node colored graphs $\mathscr{G} = (V, \to, C, a)$ and $\mathscr{H} = (U, \to', C, a')$ with $|C|$ (remember that a, a' are surjective) different colors and assume that the nodes in both graphs are renumbered such that those of color 1 come first, then those of color 2,... The adjacency matrices A (of graph $\mathscr{G}$) and B (of graph $\mathscr{H}$) can then be partitioned as follows:

$$A = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1|C|} \\ A_{11} & A_{12} & \dots & A_{1|C|} \\ \vdots & \vdots & \ddots & \vdots \\ A_{|C|1} & A_{|C|2} & \dots & A_{|C||C|} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} B_{11} & B_{12} & \dots & B_{1|C|} \\ B_{11} & B_{12} & \dots & B_{1|C|} \\ \vdots & \vdots & \ddots & \vdots \\ B_{|C|1} & B_{|C|2} & \dots & B_{|C||C|} \end{pmatrix}$$

Remember that $c_{\mathscr{G}}(V, i)$ denotes the number of vertices of color i in graph $\mathscr{G}$, then each block $A_{ij} \in \mathbb{R}^{c_{\mathscr{G}}(V,i) \times c_{\mathscr{G}}(V,j)}$ and $B_{ij} \in \mathbb{R}^{c_{\mathscr{H}}(U,i) \times c_{\mathscr{H}}(U,j)}$ describes the adjacency relations between the nodes of color i to the vertices of color j in both A and B. In fact, by just renumbering the vertices such that the nodes with the same color succeed each other, you immediately get such partitioning. If you see this renumbering as a permutation on the nodes, then performing on the original adjacency matrix a left and right multiplication with the corresponding permutation matrix (see Definition 1.1.1) leads to this adjusted adjacency matrix. To make the idea more comprehensible, we give an example.

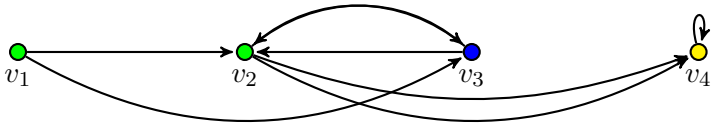
Example 2.4.1. Let $\mathscr{G}$ be following graph:



The adjacency matrix equals:

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

If we order the colors as: {green, blue, yellow} (so color 1 = green, color 2 = blue, color 3 = yellow), we can renumber the vertices and get the following graph:



The adjacency matrix equals:

$$A' = PAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

which can be partitioned in blocks as follows (we have 3 colors: 1 = green, 2 = blue, 3 = yellow):

$$A' = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} & \begin{pmatrix} 1 \\ 1 \end{pmatrix} & \begin{pmatrix} 0 \\ 2 \end{pmatrix} \\ \begin{pmatrix} 0 & 2 \end{pmatrix} & \begin{pmatrix} 0 \end{pmatrix} & \begin{pmatrix} 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 \end{pmatrix} & \begin{pmatrix} 1 \end{pmatrix} \end{pmatrix}.$$

Now, we will only compare the nodes of the same color in both graphs, so that we define similarity matrices $S_{ii} \in \mathbb{R}^{c_{\mathscr{G}}(V,i) \times c_{\mathscr{H}}(v_i)}$, with $i = 1, \ldots, |C|$, which we can put in a block-diagonal similarity matrix:

$$S = \begin{pmatrix} S_{11} & 0 & \ldots & 0 \\ 0 & S_{12} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & S_{|C||C|} \end{pmatrix}$$

Theorem 2.4.2. *Let $\mathscr{G} = (V, \to, C, a)$ and $\mathscr{H} = (U, \to', C, a')$ be two node colored graphs and define:*

$$\begin{aligned} Z_1^{(k+1)} &= \sum_{i \in \{1,\ldots,|C|\}} B_{1i} S_{ii}^{(k)} A_{1i}^T + B_{i1}^T S_{ii}^{(k)} A_{i1} \\ Z_2^{(k+1)} &= \sum_{i \in \{1,\ldots,|C|\}} B_{2i} S_{ii}^{(k)} A_{2i}^T + B_{i2}^T S_{ii}^{(k)} A_{i2} \\ &\vdots \\ Z_{|C|}^{(k+1)} &= \sum_{i \in \{1,\ldots,|C|\}} B_{|C|i} S_{ii}^{(k)} A_{|C|i}^T + B_{i|C|}^T S_{ii}^{(k)} A_{i|C|} \\ (S_{11}, S_{22}, \ldots, S_{|C||C|})^{(k+1)} &= \frac{\left(Z_1^{(k+1)}, Z_2^{(k+1)}, \ldots, Z_{|C|}^{(k+1)}\right)}{\left|\left|\left(Z_1^{(k+1)}, Z_2^{(k+1)}, \ldots, Z_{|C|}^{(k+1)}\right)\right|\right|_F} \end{aligned}$$

for $k = 0, 1, \ldots$

Then the the matrix subsequences $Z_j^{(2k)}$ *for every* $j \in \{1, \ldots, |C|\}$ *and* $(S_{11}, S_{22}, \ldots, S_{|C||C|})^{(2k)}$ *converge to* Z_j^{even} *and* $(S_{11}, S_{22}, \ldots, S_{|C||C|})^{even}$*. Also the odd subsequences converge. If we take:*

$$S_{jj}^{(0)} = J \in \mathbb{R}^{c_{\mathscr{H}}(U,j) \times c_{\mathscr{G}}(V,j)}$$

as initial matrices, then the resulting $S_{jj}^{even}(\mathbf{1})$ *are the unique matrices of largest 1-norm among all possible limits with positive start vector.*

Proof. By induction on $|C|$. Remember from Definition 1.5.18 that the function a is surjective, meaning that C only consists of colors that are actually used.

For $|C| = 1$, we have a graph with all vertices having the same color, which can be seen as an uncolored graph. This is just the normal case as proved in Theorem 2.2.15.

Although redundant, it is instructive to prove the case $|C| = 2$ separately, because the generalization in the induction step is then immediately clear, so consider the partitioned adjacency matrices:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix},$$

the equations of the theorem are in this case:

$$\begin{aligned} Z_1'^{(k+1)} &= B_{11} S_{11}^{(k)} A_{11}^T + B_{11}^T S_{11}^{(k)} A_{11} + B_{12} S_{22}^{(k)} A_{12}^T + B_{21}^T S_{22}^{(k)} A_{21} \\ Z_2'^{(k+1)} &= B_{21} S_{11}^{(k)} A_{21}^T + B_{12}^T S_{11}^{(k)} A_{12} + B_{22} S_{22}^{(k)} A_{22}^T + B_{22}^T S_{22}^{(k)} A_{22} \\ (S_{11}, S_{22})^{(k+1)} &= \frac{(Z_1'^{(k+1)}, Z_2'^{(k+1)})}{\|(Z_1'^{(k+1)}, Z_2'^{(k+1)})\|_F} \end{aligned}$$

We can write with Theorem 2.2.14:

$$\begin{aligned} Z_1'^{(k+1)} &= B_{11} Z_1'^{(k)} A_{11}^T + B_{11}^T Z_1'^{(k)} A_{11} + B_{12} Z_2'^{(k)} A_{12}^T + B_{21}^T Z_2'^{(k)} A_{21} \\ \Leftrightarrow \operatorname{vec}(Z_1'^{(k+1)}) &= \operatorname{vec}(B_{11} Z_1'^{(k)} A_{11}^T + B_{11}^T Z_1'^{(k)} A_{11} + B_{12} Z_2'^{(k)} A_{12}^T + B_{21}^T Z_2'^{(k)} A_{21}) \\ \Leftrightarrow \operatorname{vec}(Z_1'^{(k+1)}) &= (A_{11} \otimes B_{11}) \operatorname{vec} Z_1'^{(k)} + (A_{11}^T \otimes B_{11}^T) \operatorname{vec}(Z_1'^{(k)}) \\ &\quad + (A_{12} \otimes B_{12}) \operatorname{vec}(Z_2'^{(k)}) + (A_{21}^T \otimes B_{21}^T) \operatorname{vec}(Z_2'^{(k)}) \\ \Leftrightarrow \operatorname{vec}(Z_1'^{(k+1)}) &= (A_{11} \otimes B_{11} + A_{11}^T \otimes B_{11}^T) \operatorname{vec}(Z_1'^{(k)}) \\ &\quad + (A_{12} \otimes B_{12} + A_{21}^T \otimes B_{21}^T) \operatorname{vec}(Z_2'^{(k)}) \end{aligned}$$

In a similar way, we can write $Z_2'^{(k+1)}$, which is the not normalized version of $Z_2^{(k+1)}$, as:

$$\operatorname{vec}(Z_2'^{(k+1)}) = (A_{21} \otimes B_{21} + A_{12}^T \otimes B_{12}^T) \operatorname{vec} Z_1'^{(k)} + (A_{22} \otimes B_{22} + A_{22}^T \otimes B_{22}^T) \operatorname{vec} Z_2'^{(k)}$$

If we define M, $\mathbf{z}^{k+1}$ as follows, the previous expressions concatenate to a single matrix update equation:

$$\begin{aligned} \mathbf{z}^{k+1} &= \begin{pmatrix} \operatorname{vec}(Z_1') \\ \operatorname{vec}(Z_2') \end{pmatrix}^{(k+1)} \\ &= \begin{pmatrix} A_{11} \otimes B_{11} + A_{11}^T \otimes B_{11}^T & A_{12} \otimes B_{12} + A_{21}^T \otimes B_{21}^T \\ A_{21} \otimes B_{21} + A_{12}^T \otimes B_{12}^T & A_{22} \otimes B_{22} + A_{22}^T \otimes B_{22}^T \end{pmatrix} \begin{pmatrix} \operatorname{vec}(Z_1') \\ \operatorname{vec}(Z_2') \end{pmatrix}^{(k)} \\ &= M \mathbf{z}^{(k)} \end{aligned}$$

Notice that the diagonal blocks in M are related to links between the nodes with the same color, while the off diagonal blocks refer to links between nodes of another color. As always, we want to use Theorem 2.2.10 to get the result. M is clearly nonnegative, because every block A_{ij} and B_{ij} is nonnegative (it's derived from the nonnegative adjacency matrices A and B). Proving that M is symmetric a bit trickier, but notice that $A_{11} \otimes B_{11} + (A_{11} \otimes B_{11})^T$ is a symmetric $c_{\mathscr{G}}(1)c_{\mathscr{H}}(1) \times c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$ matrix (because it's the sum of a matrix with his transpose), and $A_{22} \otimes B_{22} + (A_{22} \otimes B_{22})^T$ is a symmetric $c_{\mathscr{G}}(2)c_{\mathscr{H}}(2) \times c_{\mathscr{G}}(2)c_{\mathscr{H}}(2)$. If we define $G = A_{21} \otimes B_{21} + A_{12}^T \otimes B_{12}^T$, G is a $c_{\mathscr{G}}(2)c_{\mathscr{H}}(2) \times c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$-matrix. Now, notice the following relation between the off diagonal blocks:

$$\begin{aligned} G^T &= (A_{21} \otimes B_{21} + A_{12}^T \otimes B_{12}^T)^T \\ &= (A_{21} \otimes B_{21})^T + ((A_{12}^T \otimes B_{12}^T)^T)^T \\ &= A_{12} \otimes B_{12} + A_{21}^T \otimes B_{21}^T \end{aligned}$$

G^T is a $c_{\mathscr{G}}(1)c_{\mathscr{H}}(1) \times c_{\mathscr{G}}(2)c_{\mathscr{H}}(2)$-matrix. So we can rewrite M as:

$$M = \begin{pmatrix} A_{11} \otimes B_{11} + A_{11}^T \otimes B_{11}^T & G^T \\ G & A_{22} \otimes B_{22} + A_{22}^T \otimes B_{22}^T \end{pmatrix}$$

M is a $(c_{\mathscr{G}}(1)c_{\mathscr{H}}(1) + c_{\mathscr{G}}(2)c_{\mathscr{H}}(2)) \times (c_{\mathscr{G}}(1)c_{\mathscr{H}}(1) + c_{\mathscr{G}}(2)c_{\mathscr{H}}(2))$-matrix, we want to prove that $(M)_{ij} = (M)_{ji}$ and to do so we distinguish all possible cases:

- If $i \leq c_{\mathscr{G}}(1)c_{\mathscr{H}}(1), j \leq c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$, then $(M)_{ij}$ and $(M)_{ji}$ will both be entries of $A_{11} \otimes B_{11} + A_{11}^T \otimes B_{11}^T$ and this submatrix was symmetric.
- If $i \leq c_{\mathscr{G}}(1)c_{\mathscr{H}}(1), j > c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$, then $(M)_{ij}$ will be an entry of G^T and $(M)_{ji}$ will be an entry of G, so they are equal.
- If $i > c_{\mathscr{G}}(1)c_{\mathscr{H}}(1), j > c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$, then $(M)_{ij}$ and $(M)_{ji}$ will both be entries of $A_{22} \otimes B_{22} + A_{22}^T \otimes B_{22}^T$ and this submatrix was symmetric.
- If $i > c_{\mathscr{G}}(1)c_{\mathscr{H}}(1), j \leq c_{\mathscr{G}}(1)c_{\mathscr{H}}(1)$, then $(M)_{ij}$ will be an entry of G and $(M)_{ji}$ will be an entry of G^T, so they are equal.

The result immediately follows from Theorem 2.2.10. We motivated the usage of the Frobenius norm already in the proof of Theorem 2.2.15. Notice that the normalization after each iteration step happens 'together' by dividing by $\|(Z_1^{(k+1)}, Z_2^{(k+1)})\|_F$, because this is in accordance to the conditions of Theorem 2.2.10. Normalizing $Z_1^{(k+1)}, Z_2^{(k+1)}$ separately after the expressions are calculated is a bad idea: it gives an iterative process that is different from the one described in Theorem 2.2.10 and we can not prove convergence in this case.

We now prove the induction step $|C| = n - 1 \Rightarrow |C| = n$. The only crucial thing to prove is that M is again symmetric, the rest of the steps consist of an easy expansion of the case

$|C| = 2$. M is in this case equal to:

$$M = \left(\begin{array}{ccc|c} A_{11} \otimes B_{11} & \cdots & A_{1(n-1)} \otimes B_{1(n-1)} & A_{1n} \otimes B_{1n} \\ +A_{11}^T \otimes B_{11}^T & & +A_{(n-1)1}^T \otimes B_{(n-1)1}^T & +A_{n1}^T \otimes B_{n1}^T \\ \vdots & \ddots & \vdots & \vdots \\ A_{(n-1)1} \otimes B_{(n-1)1} & \cdots & A_{(n-1)(n-1)} \otimes B_{(n-1)(n-1)} & A_{(n-1)n} \otimes B_{(n-1)n} \\ +A_{1(n-1)}^T \otimes B_{1(n-1)}^T & & +A_{(n-1)(n-1)}^T \otimes B_{(n-1)(n-1)}^T & +A_{n(n-1)}^T \otimes B_{n(n-1)}^T \\ \hline A_{n1} \otimes B_{n1} & \cdots & A_{n(n-1)} \otimes B_{n(n-1)} & A_{nn} \otimes B_{nn} \\ +A_{1n}^T \otimes B_{1n}^T & & +A_{(n-1)n}^T \otimes B_{(n-1)n}^T & +A_{nn}^T \otimes B_{nn}^T \end{array}\right)$$

which can be seen as:

$$M = \left(\begin{array}{ccc|c} & & & A_{1n} \otimes B_{1n} \\ & & & +A_{n1}^T \otimes B_{n1}^T \\ & M' & & \vdots \\ & & & A_{(n-1)n} \otimes B_{(n-1)n} \\ & & & +A_{n(n-1)}^T \otimes B_{n(n-1)}^T \\ \hline A_{n1} \otimes B_{n1} & \cdots & A_{n(n-1)} \otimes B_{n(n-1)} & A_{nn} \otimes B_{nn} \\ +A_{1n}^T \otimes B_{1n}^T & & +A_{(n-1)n}^T \otimes B_{(n-1)n}^T & +A_{nn}^T \otimes B_{nn}^T \end{array}\right)$$

From the induction hypothesis we now that M' is symmetric. It's clear that the entries in the last column of M are the transpose of the entries in the last row. It follows that M is again symmetric.

□

Algorithm

We present an algorithmic implemtation of the described method. First, we have to find a way to easily calculate a partitioned adjacency matrix. This is presented in Algorithm 8. This algorithm expects as input an adjancency matrix where the vertices are arranged by color and an ordered list that tells how many vertices belong to each color. In Algorithm 9 we calculate the similarity matrix of two node colored graphs based on the adjacency matrix patitioning algorithm. A Matlab implementation of both algoritmhs can be found in Listing A.7 and Listing A.8 in Appendix A.

Data:
C: a list with the number of vertices sharing the same colors in $\mathscr{G}$ (both the vertices as the colors must be ordered appropriately)
A: the 'normal' adjacence matrix of the graph $\mathscr{G}$ (the vertices must be ordered approprately: the first vertices must belong to the first color)
Result:
Z: a 2-dimensional list where the element on place (i, j) represents the partition of adjancency matrix A, $\mathrm{A}_{ij}, between the vertices of color i and the vertices of color \mathrm{j}$
begin `colored_node_adjacency_matrix_partitioning(`C`,`A`)`
  $Z = \prod_{i,j} C(i).C(j) \times \prod_{i,j} C(i).C(j)$-matrix with all entries equal to 0;
  for $i : 1$ *to* $|C|$ **do**
    for $j : 1$ *to* $|C|$ **do**
      C_i = the number of vertices of color i $(= C(i))$;
      C_j = the number of vertices of color j $(= C(j))$;
      B = a $C_i \times C_j$-matrix with all entries equal to 0;
      start_vertex_i = 0;
      start_vertex_j = 0;
      for $k : 1$ *to* $i - 1$ **do**
        C_k = the number of vertices of color k $(= C(k))$;
        start_vertex_i = start_vertex_i + C_k;
      end
      for $k : 1$ *to* $i - 1$ **do**
        C_k = the number of vertices of color k $(= C(k))$;
        start_vertex_j = start_vertex_j + C_k;
      end
      for $r : 1$ *to* c_i **do**
        for $k : 1$ *to* c_j **do**
          $B(r, k) = A$(start_vertex_i $+r$, start_vertex_j $+k$);
        end
      end
      $Z(i, j) = B$;
    end
  end
  return Z;
end

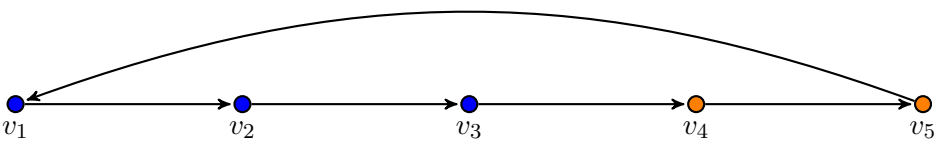
Algorithm 8: Algorithm that takes an ordered list with the number of vertices and a normal adjacency matrix as input and returns a partitioned adjacency matrix.

Data:
C_A: a list with the number of vertices sharing the same colors in $\mathscr{G}$,
A: the 'normal' adjacence matrix of the graph $\mathscr{G}$,
C_B: a list with the number of vertices sharing the same colors in $\mathscr{H}$,
B: the 'normal' adjacence matrix of the graph $\mathscr{H}$.
TOL: tolerance for the estimation error.
Result:
S: a 1-dimensional list with on place (i) the similarity matrix of the vertices with color i, S_{ii}.
begin `colored_node_similarity_matrix`(C_A, A, C_B, B, TOL)
  A_P = `colored_node_adjacency_matrix_partitioning` (C_A, A);
  B_P = `colored_node_adjacency_matrix_partitioning` (C_B, B);
  $k = 1$;
  for $i : 1$ *to* $|C|$ **do**
    $S(i)$ = a $C_B(i) \times C_A(i)$-matrix with all entries equal to 1;
    $\mu(i)$ = a $C_B(i) \times C_A(i)$-matrix with all entries equal to TOL;
  end
  while *True* **do**
    norm = 0;
    for $i : 1$ *to* $|C_A|$ **do**
      Z(i) = $\sum_{j\in\{1,\ldots,|C_A|\}} B_P(1,j)S_{\text{previous}}(i)A_P^T(1,i) + B_P^T(i,1)S_{\text{previous}}A_P(i,1)$;
      norm = norm + `trace` $(Z^T(i).Z(i))$;
    end
    S = $\frac{Z}{\text{norm}}$;
    if k *even* **then**
      if $|S - S_{previous\_even}| < TOL$ **then**
        break;
      else
        $S_{\text{previous\_even}} = S$
      end
    end
  end
  return S;
end

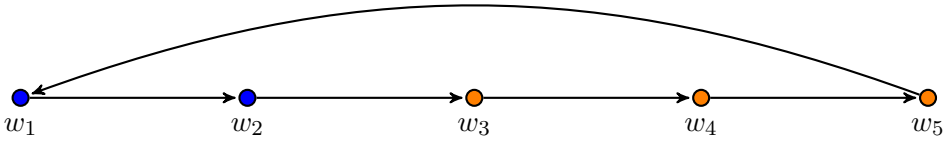
Algorithm 9: Algorithm that takes an ordered list with the number of vertices and a normal adjacency matrix as input and returns a partitioned adjacency matrix.

Example

Example 2.4.3. Let $\mathscr{G}$ be the graph:



and $\mathscr{H}$ be the graph:



The similarity matrix is given by:

$$S = \begin{pmatrix} 0.5969 & 0.3791 & 0 & 0 & 0 \\ 0 & 0.3791 & 0.5969 & 0 & 0 \\ 0 & 0 & 0 & 0.5986 & 0 \\ 0 & 0 & 0 & 0.3764 & 0.3764 \\ 0 & 0 & 0 & 0 & 0.5986 \end{pmatrix}$$

As one could expect, the highest similarity scores (0.59) are obtained for the nodes that are the transitions between two type of nodes: all the pairs $(v_1, w_1), (v_3, w_2), (v_4, w_3)(v_5, w_5)$ share this similarity score.

2.4.2 Colored edges

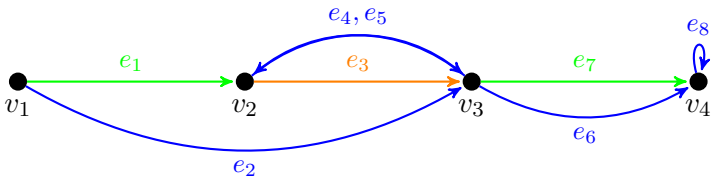
Method

We now extend the node-edge similarity method from Section 2.3 to edge colored graph. Take two edge colored graphs $\mathscr{G} = (V, \to, C, b)$ and $\mathcal{G} = (U', \to', C, b')$ with $|C|$ different colors (remember that b, b' are surjective). The edges can be renumbered such that those of the same color are next to each other in the source-edge and terminal-edge matrices, so A_S, A_T from $\mathscr{G}$ and B_S, B_T from $\mathscr{H}$ can be partitioned as follows:

$$A_S = \begin{pmatrix} A_{S_1} & \dots & A_{S_{|C|}} \end{pmatrix} \quad \text{and} \quad A_T = \begin{pmatrix} A_{T_1} & \dots & A_{T_{|C|}} \end{pmatrix}$$
$$B_S = \begin{pmatrix} B_{S_1} & \dots & B_{S_{|C|}} \end{pmatrix} \quad \text{and} \quad B_T = \begin{pmatrix} B_{T_1} & \dots & B_{T_{|C|}} \end{pmatrix}$$

Again, this can easily be achieved by multiplying the original A_S by the permutation matrix that represents the renumbering of the edges. The blocks $A_{S_i}, A_{T_i} \in \mathbb{R}^{n_{\mathscr{G}} \times c_{\mathscr{G}}(\to, i)}$ with $n_{\mathscr{G}}$ the number of vertices of $\mathscr{G}$ and $c_{\mathscr{G}}(\to, i)$ the number of edges of color i. So for $\mathscr{H}$ we have: $B_{S_i}, B_{T_i} \in \mathbb{R}^{n_{\mathscr{H}} \times c_{\mathscr{H}}(\to', i)}$. We give a small example.

Example 2.4.4. Let $\mathscr{G}$ be following graph:

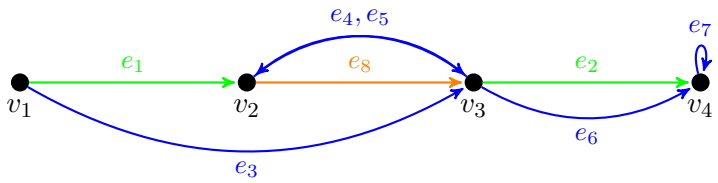


When we calculate the source-edge matrix with Algorithm 7 (the resulting matrices represent an edge numbering left-to-right, the same as indicated in the graph) and terminal-edge

matrices in an equal way, we get:

$$A_S = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad A_T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

If we order the colors as: {green, blue, orange} (so color 1 = green, color 2 = blue, color 3 = yellow), we can renumber the edges:



A'_S and A'_T are now:

$$A'_S = A_S P = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

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

$$A'_T = A_T P = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

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

which can be partitioned in blocks as follows (we have 3 colors: 1 = green, 2 = blue, 3 =

orange):

$$A'_S = \begin{pmatrix} A'_{S_1} & A'_{S_2} & A'_{S_3} \end{pmatrix} = \left(\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \right)$$

$$A'_T = \begin{pmatrix} A'_{T_1} & A'_{T_2} & A'_{T_3} \end{pmatrix} = \left(\begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right)$$

Just like the colored node method, the edge similarity matrix will be block-diagonal because we compare only the edges of the same type. The edge similarity matrix has thus a block diagonal structure with blocks $Y_{ii} \in \mathbb{R}^{c_{\mathcal{G}}(\to,i) \times c_{\mathcal{H}}(\to',i)}$.

$$Y = \begin{pmatrix} Y_{11} & 0 & \dots & 0 \\ 0 & Y_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Y_{|C||C|} \end{pmatrix}$$

The node similarity matrix X, on the other hand, is no different from the one of Theorem 2.3.6. To adapt the method of Theorem 2.3.6 to colored edges we have to rewrite the equations (2.18) and (2.19) in a decoupled form such that $X^{(k)}$ and $Y^{(k)}$ can be calculated independently of each other. To make this paragraph more readable, we write our orignal equations again:

$$\begin{aligned} Y^{(k+1)} &= \frac{B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \\ X^{(k+1)} &= \frac{B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T}{\|B_S Y^{(k)} A_S^T + B'_T Y^{(k)} A_T^T\|_F} \end{aligned}$$

Remember that $G = A_S^T \otimes B_S^T + A_T^T \otimes B_T^T$ and $G^T = A_S \otimes B_S + A_T \otimes B_T$. From equations (3.10) and (3.11) we can write (see the proof of Theorem 2.3.6):

$$\mathbf{x}^{(k+1)} = \frac{G^T G(\mathbf{x}^{(k)})}{\|G^T G(\mathbf{x}^{(k)})\|_F} \quad \text{and} \quad \mathbf{y}^{(k+1)} = \frac{G G^T(\mathbf{y}^{(k)})}{\|G G^T(\mathbf{y}^{(k)})\|_F} \tag{2.22}$$

Remember that $\mathbf{x}^{(k)} = \text{vec}(X^{(k)})$, with Lemma 2.2.13 we can rewrite this to decoupled notations (see the first part of the proof Theorem 2.3.6):

$$\begin{aligned} X^{(k+1)} &= \frac{B_S(B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T) A_S^T + B_T(B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T) A_T^T}{\|B_S(B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T) A_S^T + B_T(B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T) A_T^T\|_F} \\ Y^{(k+1)} &= \frac{B_S^T(B_S Y^{(k)} A_S^T + B_T Y^{(k)} + A_T^T) A_S + B_T^T(B_S Y^{(k)} A_S^T + B_T Y^{(k)} + A_T^T) A_T}{\|B_S^T(B_S Y^{(k)} A_S^T + B_T Y^{(k)} + A_T^T) A_S + B_T^T(B_S Y^{(k)} A_S^T + B_T Y^{(k)} + A_T^T) A_T\|_F} \end{aligned}$$

To keep the notation understandable, we will keep on using the decoupled equations of (2.22). We are ready for the theorem that describes the method of edge similarity on colored edges:

Theorem 2.4.5. *Let $\mathscr{G} = (V, \to, C, b)$ and $\mathscr{H} = (U, \to', C, b')$ be two edge colored graphs and define:*

$$\begin{aligned}
X'^{(k+1)} &= \sum_{i\in\{1,\dots,|C|\}} B_{S_i} Y_{ii}^{(k)} A_{S_i}^T + B_{T_i} Y_{ii}^{(k)} A_{T_i}^T \\
Y_{11}'^{(k+1)} &= B_{S_1}^T X^{(k)} A_{S_1} + B_{T_1}^T X^{(k)} A_{T_1} \\
&\vdots \\
Y_{ii}'^{(k+1)} &= B_{S_i}^T X^{(k)} A_{S_i} + B_{T_i}^T X^{(k)} A_{T_i} \\
&\vdots \\
Y_{|C||C|}'^{(k+1)} &= B_{S_{|C|}}^T X^{(k)} A_{S_{|C|}} + B_{T_{|C|}}^T X^{(k)} A_{T_{|C|}} \\
(X, Y_{11}, \dots, Y_{|C||C|}) &= \frac{\left(X'^{(k+1)}, Y_{11}'^{(k+1)}, \dots, Y_{|C||C|}'^{(k+1)}\right)}{\left\|\left(X'^{(k+1)}, Y_{11}'^{(k+1)}, \dots, Y_{|C||C|}'^{(k+1)}\right)\right\|_F}
\end{aligned}$$

for $k = 0, 1, \dots$ Then the the matrix subsequences $(X, Y_{11}, \dots, Y_{|C||C|})^{(2k)}$ converge to $(X, Y_{11}, \dots, Y_{|C||C|})^{even}$ Also the odd subsequences converge. If we take:

$$\begin{aligned}
X^{(0)} &= J \in \mathbb{R}^{n_{\mathscr{H}} \times n_{\mathscr{G}}} \\
Y_{jj}^{(0)} &= J \in \mathbb{R}^{c_{\mathscr{G}}(\to, i) \times c_{\mathscr{H}}(\to', i)}
\end{aligned}$$

as initial matrices, then the resulting $S_{jj}^{even}(\mathbf{1})$ are the unique matrices of largest 1-norm among all possible limits with positive start vector.

Proof. By induction on $|C|$. Remember from Defintion 1.5.20 that the function b is surjective, meaning that C only consist of colors that are actually used.

For $|C| = 1$, we have a graph with all edges having the same color, which can be seen as an uncolored graph. This is just the normal case as proved in Theorem 2.3.6.

Altough again redundant, it is instructuve to prove the case $|C|$ seprately, because the generalization in the indcution step is then immediately clear, so consider the partitioned source-edge and terminal-edge matrices:

$$\begin{aligned}
A_S &= \begin{pmatrix} A_{S_1} & A_{S_2} \end{pmatrix} \quad \text{and} \quad A_T = \begin{pmatrix} A_{T_1} & A_{T_2} \end{pmatrix} \\
B_S &= \begin{pmatrix} B_{S_1} & B_{S_2} \end{pmatrix} \quad \text{and} \quad B_T = \begin{pmatrix} B_{T_1} & A_{B_2} \end{pmatrix}
\end{aligned}$$

the equations of the theorem are in this case:

$$\begin{aligned}
X'^{(k+1)} &= B_{S_1} Y_{11}'^{(k)} A_{S_1}^T + B_{T_1} Y_{11}'^{(k)} A_{T_1}^T + B_{S_2} Y_{22}'^{(k)} A_{S_2}^T + B_{T_2} Y_{22}'^{(k)} A_{T_2}^T \\
Y_{11}'^{(k+1)} &= B_{S_1}^T X'^{(k)} A_{S_1} + B_{T_1}^T X'^{(k)} A_{T_1} \\
Y_{22}'^{(k+1)} &= B_{S_2}^T X'^{(k)} A_{S_2} + B_{T_2}^T X'^{(k)} A_{T_2}
\end{aligned}$$

which can be rewritten using Theorem 2.2.13 as:

$$\begin{aligned}
X'^{(k+1)} &= B_{S_1} Y_{11}'^{(k)} A_{S_1}^T + B_{T_1} Y_{11}'^{(k)} A_{T_1}^T + B_{S_2} Y_{22}'^{(k)} A_{S_2}^T + B_{T_2} Y_{22}'^{(k)} A_{T_2}^T \\
\Leftrightarrow \operatorname{vec}(X'^{(k+1)}) &= (A_{S_1} \otimes B_{S_1} + A_{T_1} \otimes B_{T_1}) \operatorname{vec}(Y_{11}'^{(k)}) + (A_{S_2} \otimes B_{S_2} + A_{T_2} \otimes B_{T_2}) \operatorname{vec}(Y_{22}'^{(k)})
\end{aligned}$$

$Y_{11}^{\prime(k)}$ and $Y_{22}^{\prime(k)}$ can also be rewritten:

$$\begin{aligned}\operatorname{vec}(Y_{11}^{\prime(k+1)}) &= (A_{S_1}^T \otimes B_{S_1}^T + A_{T_1}^T \otimes B_{T_1}^T)\operatorname{vec}(X^{\prime(k)}) \\ \operatorname{vec}(Y_{22}^{\prime(k+1)}) &= (A_{S_2}^T \otimes B_{S_2}^T + A_{T_2}^T \otimes B_{T_2}^T)\operatorname{vec}(X^{\prime(k)})\end{aligned}$$

We define $\mathbf{z}^{(k+1)}$ and M as follows, and again the previous expressions concatenate to a single matrix equation:

$$\begin{aligned}\mathbf{z}^{(k+1)} &= \begin{pmatrix}\operatorname{vec}(X)\\ \operatorname{vec}(Y_{11})\\ \operatorname{vec}(Y_{22})\end{pmatrix}^{(k+1)} \\ &= \begin{pmatrix} 0 & \begin{matrix}A_{S_1}\otimes B_{S_1}\\ +A_{T_1}\otimes B_{T_1}\end{matrix} & \begin{matrix}A_{S_2}\otimes B_{S_2}\\ +A_{T_2}\otimes B_{T_2}\end{matrix} \\ \begin{matrix}A_{S_1}^T\otimes B_{S_1}^T\\ +A_{T_1}^T\otimes B_{T_1}^T\end{matrix} & 0 & 0 \\ \begin{matrix}A_{S_2}^T\otimes B_{S_2}^T\\ +A_{T_2}^T\otimes B_{T_2}^T\end{matrix} & 0 & 0 \end{pmatrix}\begin{pmatrix}\operatorname{vec}(X)\\ \operatorname{vec}(Y_{11})\\ \operatorname{vec}(Y_{22})\end{pmatrix}^{(k)} \\ &= M\mathbf{z}^{(k)}\end{aligned}$$

M is cleary nonnegative as it exists of zero elements or sums of Kronecker products of nonnegative matrices. To see that is symmetric, rewrite M with G and G^T:

$$G = \begin{pmatrix}A_{S_1}^T\otimes B_{S_1}^T + A_{T_1}^T\otimes B_{T_1}^T\\ A_{S_2}^T\otimes B_{S_2}^T + A_{T_2}^T\otimes B_{T_2}^T\end{pmatrix}$$

With Lemma 2.2.13 we calculate G^T:

$$G^T = \begin{pmatrix}A_{S_1}\otimes B_{S_1} + A_{T_1}\otimes B_{T_1} & A_{S_2}\otimes B_{S_2} + A_{T_2}\otimes B_{T_2}\end{pmatrix}$$

G is a $n_{\mathscr{G}}n_{\mathscr{H}} \times (c_{\mathscr{G}}(\to,1)c_{\mathscr{H}}(\to',1) + c_{\mathscr{G}}(\to,1)c_{\mathscr{H}}(\to',2))$-matrix, so:

$$M = \begin{pmatrix}0_{c_{\mathscr{G}}(\to,1)c_{\mathscr{H}}(\to',1)+c_{\mathscr{G}}(\to,1)c_{\mathscr{H}}(\to',2)} & G^T\\ G & 0_{n_{\mathscr{G}}n_{\mathscr{H}}}\end{pmatrix}$$

which is clearly a symmetric matrix and the result follows.

We now prove the induction step $|C| = n-1 \Rightarrow |C| = n$. The only thing to show is that M stays symmetric (nonnegative is clear), but this is obvious as G can be rewritten as:

$$G = \begin{pmatrix}A_{S_1}^T\otimes B_{S_1}^T + A_{T_1}^T\otimes B_{T_1}^T\\ A_{S_2}^T\otimes B_{S_2}^T + A_{T_2}^T\otimes B_{T_2}^T\\ \\ A_{S_{|C|}}^T\otimes B_{S_{|C|}}^T + A_{T_{|C|}}^T\otimes B_{T_{|C|}}^T\end{pmatrix}$$

and you can rewrite M with G and G^T.

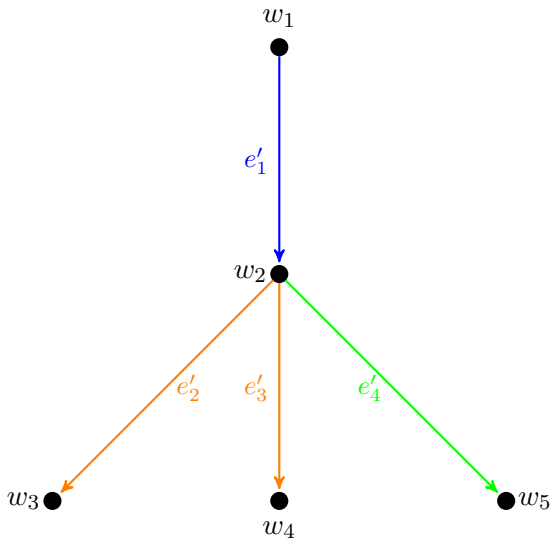
□

Algorithm

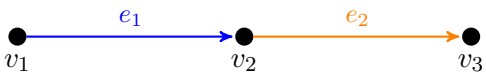
With the same reasoning as for colored nodes, we developed an algorithm that takes a source-edge matrix or a terminal-edge matrix as input together with a an ordered list with the number of edges belonging to each color. Under the condition that the edges are arranged by color in the source-edge or terminal-edge matrix, this algorithm returns a partitioned source or terminal-edge matrix. Based on this algorithm, we can construct an algorithm that returns the node similarity matrix X and the (colored) edge similarity matix Y. The Matlab listings of both algorithms can be found in Listing A.9 and Listing A.10 in Appendix A. Because the algorithms resemble a lot to the previous algorithms, we didn't write them in pseudocode.

Example

Example 2.4.6. Let $\mathscr{G}$ be the graph:



and $\mathscr{H}$ be the graph:



The node and edge similarity matrix are given by:

$$X = \begin{pmatrix} 0.2295 & 0 & 0 & 0 & 0 \\ 0 & 0.8903 & 0 & 0 & 0 \\ 0 & 0 & 0.2284 & 0.2284 & 0.2284 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0.5893 & 0 & 0 & 0 \\ 0 & 0.5812 & 0.5812 & 0 \end{pmatrix}$$

In fact these two graphs are very similar in the sense that node v_3 is replaced by three nodes w_3, w_4, w_5. This is also detected by the node and edge similarity matrices, giving equal similarity scores for (e'_2, e_2) and (e'_3, e_2), (e'_4, e_2) is equal to zero because e'_4 has a different color.

2.4.3 Full colored graphs

We can also consider the combination of colored nodes and colored edges. This in fact the same method as with the colored edges, buth the matrices X and Y will in this case both be block diagonal (possibly with a different number of blocks). The iteration matrix for Y will be completely equal to the iteration matrix for colored edges and it's easy to adapt the iteration matrix for X so it can handle with colored nodes. We will not discuss this case in detail, as it would be an extensive repetition of the previous subsections.

2.5 Applications

Similarity on graphs is a fairly new concept, so there are not so many applications developed yet. In this small subsection we give a concise overview of the already existing applications. We just give an intuitive idea to give the reader a notion of which kind of applications are already possible. For the more detailed, mathematical approach we refer to relevant papers.

2.5.1 Synonym Extraction

In [BGH+04], they use the structure graph $1 \to 2 \to 3$ to substract synonyms in graphs constructed from a dictionary. the method is based on the assumption that synonyms have many words in common in their definitions and appear together in the defintion of many words. So to construct a *dictionary graph* $\mathscr{G}$, each word of the dictionary is a vertex and there is an edge from v_i to v_j if v_i appears in the defintion of v_j. Now, with a given word w, we constrcut a neighboorhood graph $\mathscr{G}'$, which is the subgraph of $\mathscr{G}$ whose vertices are pointed to by w or are pointing to w. Finally, we rank the words by decreasing central score and the words with the highest central scores will possibly be good synonyms.

2.5.2 Graph matching

The mean usage of the current graph similarity algorithms as presented in this section, is to find some kind of an optimal matching between the elements (vertices and edges) between two graphs. Graph matching has received a lot of popularity because it's useful for data mining. By example, in [ZV08] they use brute-force algorithm that calculates the similarity scores between a graph $\mathscr{G}$ and any possible subgraph of $\mathscr{G}$. By putting some conditions about the desired result, it is possible to detect the subgraph that is most similar to the original graph. Practical uses of graph matching are:

- Comparing two databases,
- Finding clusters in a set of data,
- Align sequences of DNA,
- ...

Chapter 3

Similarity on hypergraphs

3.1 Introduction

In this section, we explore similarity on hypergraphs. This is exploration is new as no papers can be found on the subject. In order to generalize this concept of similarity successfully and give it a correct meaning, we need to formulate some conditions that a possible method for similarity on hypergraphs has to fulfill. Intuitively, all methods from the previous chapter work in the same way: in the first iteration step only the adjacency relations of vertices (or edges) in two graphs are used and in each following iteration step, the relationships between these adjaceny relations are calculated, which will result in high similarity scores (compared to the others in the similarity matrix) for a vertex when the adjacent vertices have a high similarity score. When calculating an edge similarity matrix, an edge will have a high similarity score (compared to others) if the vertices that an edge connect have a high similarity score.

Based on this, we now present our conditions. These conditions must ensure that a similarity method for hypergraphs has the same behaviour as the similarity methods for graphs:

(C1) When the method is applied on two undirected graphs, it must return the same similarity scores (up to a constant) as the methods from Section 2.

(C2) Adding a non-isolated node to one of the hypergraphs must influence the similarity scores.

(C3) Adding an edge that is not a hyperloop must influence the similarity scores.

(C4) The similarity score of a vertex in an hypergraph is large (compared to others) when the similarity score of the adjacent vertices in the hypergraph is large.

(C5) The similarity score of an edge is large (compared to others) when it connects vertices with large similarity scores.

(C6) When two vertices have the same relationships to all other vertices, we say that these vertices are *interchangeable* , sometimes these nodes are also called *structural equivalent*. Interchangeable vertices must have the same similarity scores because, intuitively, they play the same role. Actually, we can give a more general definition of interchangeability in hypergraphs:

Definition 3.1.1. *Two vertices v_i, v_j of a hypergraph $\mathcal{G}$ are* ***interchangeable vertices*** *if for each edge of size $k+1$ that connect v_i to $v_{p1}, v_{p2}, \ldots, v_{pk}$, there is an edge of size $k+1$ (possibly the same) that connects v_j to $v_{p1}, v_{p2}, \ldots, v_{pk}$ or an edge of size $k+1$ that connects v_j to $v'_{p1}, v'_{p2}, \ldots, v'_{pk}$, a group of interchangeable vertices of $v_{p1}, v_{p2}, \ldots, v_{pk}$ (in this case v'_{pi} can be equal to v_{pi}).*

This definition is slightly problematic as it is a circular definition. To keep things easy to understand and easy to prove, we will always limit ourselves to the non-circular defintion:

Definition 3.1.2. *Two vertices v_i, v_j of a hypergraph $\mathcal{G}$ are* ***interchangeable vertices*** *if for each edge of size $k+1$ that connect v_i to $v_{p1}, v_{p2}, \ldots, v_{pk}$, there is an edge of size $k+1$ (possibly the same) that connects v_j to $v_{p1}, v_{p2}, \ldots, v_{pk}$.*

It's easy to see that if the condition is met for the non-circular version, the condition is also met for the circular version: in the non-circular version the 'first-grade' interchangeable vertices are spotted. They will have the same similarity scores when the condition is met for the non-circular version. In the second step, you can spot the higher grade interchangeable vertices where we already know that the lower grade interchangeable vertices are ok.

When calculating an edge similarity matrix, interchangeable edges must also have the same similarity scores. Interchangeable edges are edges that contain exactly the same vertices.

(C7) The cardinality of an edge E in a hypergraph ($|E|$) must influence the edge similarity scores: edges with a high cardinality must have a higher similarity score than two edges with a lower cardinality.

(C8) A vertex can only have a similarity score equal to zero when the hypergraph is not connected. In that case, it can occur that a group of connected vertices dominates all vertices that are not adjacent to this group. Also edge can only have a similarity score equal to zero when the hypergraph is not connected. An isolated vertex (that is not a loop) will always have a similarity score equal to zero.

The attentive reader may ask himself how we collected this set of conditions. These conditions are established in a heuristic way, based on observations from various tryouts on the methods of similarity on graphs. To give these conditions a better foundation, we now give an explanation on how the methods of similarity on graphs of section 2 meet each condition for undirected graphs. Although most conditions are also met for directed graphs, we only have to consider undirected graphs because hypergraphs itself are undirected. The explanation can either be a proof of a simple theorem or a more heuristic explanation. The explanations are numbered (E1,.., E8) in the same way as the conditions because we will often refer to this explanations in the rest of this chapter.

(E1) Undirected graphs are equal to 2-hypergraphs. It is evident that in the case of a 2-hypergraph the results obtained by a similarity method for hypergraphs must return the same results (up to a constant).

(E2) It's easy to see that (C2) holds for every (node) similarity method on graphs as adding a extra node to a graph generates an additional row and column in the adjacency matrix. This row and column are thus included in the calculation of the similarity matrix S and after the calculation, similarity scores for the added node compared to all the other nodes from the other graph are obtained.

(E3) This holds with the same reasoning as (E2) but now applied to the (edge) similarity method.

(E4) We can explain this in a heuristic way: for example, in the method of Blondel, the compact form equals (A is the adjacency matrix of $\mathscr{G}_A$ and B is the adjacency matrix of $\mathscr{G}_B$):

$$S^{(k+1)} = \frac{BS^{(k)}A^T + B^TS^{(k)}A}{\|BS^{(k)}A^T + B^TS^{(k)}A\|_F}, \quad k = 0, 1, \ldots,$$

first note that $BS^{(k)}A^T + B^TS^{(k)}A$ is in fact the sum of the similarities of the children and parents of node v_i of $\mathscr{G}_B$ and vertex v_j of $\mathscr{G}_A$. We see that $s_{i,j}$, the similarity score between v_i of B and v_j of A, is in fact equal to a scalar times the element (i, j) of $BS^{(k)}A^T + B^TS^{(k)}A$. The same can be said about the even iterates and hence about the complete similarity matrix S. So we see that this implicit relation means that the similarity scores of a vertex in an graph is large (compared to others) when the similarity score of the adjacent vertices in the graph is large.

(E5) This holds with the same reasoning as (E4) but now applied to the (edge) similarity method.

(E6) To see that (C6) also holds in graphs, we define interchangeable vertices in graphs as follows:

Definition 3.1.3. *In an undirected graph $\mathscr{G}$, two vertices v_i, v_j are **interchangeable** if they have exactly the same adjacency structure. Meaning that for each edge that connects v_i to a vertex v_q, there must also be an edge that connects v_j to v_q or an edge that connects v_j to an interchangeable vertex v_q. In a directed graph, the same holds in the sense that for each edge $v_i \to v_q$ there must exist an edge $v_j \to v_q$ or an edge $v_j \to v'_q$ with v_q, v'_q interchangeable vertices. and for each edge $v_p \to v_i$ there must exist an edge $v_p \to v_j$ or an edge $v'_p \to v_j$ with v_q, v'_q interchangeable vertices.*

The non-circular version is:

Definition 3.1.4. *In an undirected graph $\mathscr{G}$, two vertices v_i, v_j are **interchangeable** if they have exactly the same adjacency structure. Meaning that for each edge that connects v_i to a vertex v_q, there must also be an edge that connects v_j to v_q. In a directed graph, the same holds in the sense that for each edge $v_i \to v_q$ there must exist an edge $v_j \to v_q$ and for each edge $v_p \to v_i$ there must exist an edge $v_p \to v_j$.*

We will show that the non-circular version of this defintion holds for the method of Blondel on graphs by proving the following theorem. Although we would be fine to prove it just for undirected graphs as hypergraphs will always have undirected graph representations, we also consider the directed case. Notice that all isolated vertices in a graph are always interchangeable.

Theorem 3.1.5. *Let $\mathscr{G}_A$ be a graph with interchangeable vertices and $\mathscr{G}_B$ another graph, then by calculating the similarity matrix between $\mathscr{G}_A$ and $\mathscr{G}_B$ the interchangeable vertices of $\mathscr{G}_A$ will have the same similarity scores for every vertex of $\mathscr{G}_B$.*

Proof. Let $\mathscr{G}_A = (V, \to), \mathscr{G}_B = (W, \to')$ with $|V| = n, |W| = m$. First, it's easy to see that the interchangeable vertices form an equivalence relation $\sim$ on V with $v_p \sim v_q$ if and only if v_p and v_q are interchangeable. Take $\overline{v_p}$, an equivalency class with more than one vertex (this exists as $\mathscr{G}_A$ has interchangeable vertices).

Now, from the definition of interchangeable vertices in graphs, we conclude that two vertices v_p and v_q are interchangeable if and only if for all $i \in \{1, \ldots, n\}$ holds that $a_{ip} = a_{iq}$ and that $a_{pi} = a_{qi}$. This holds also for directed graphs, in the undirected case we even have that $a_{ip} = a_{pi} = a_{qi} = a_{iq}$. So all vertices in $\overline{v_p}$ have the same entries in the adjacency matrix A of $\mathscr{G}_A$.

The similarity scores of v_p and any other vertex w_j of $\mathscr{G}_B$ at iteration step $k+1$ can be calculated as $(A = (a_{ij}), B = (b_{ij}), S = (s_{ij}))$:

$$s_{jp}^{(k+1)} = \frac{\sum_{f=1}^{m}\sum_{g=1}^{n} b_{jf} s_{fg}^{(k)} a_{gp}^T + b_{jf}^T s_{fg}^{(k)} a_{gp}}{\|BS^{(k)}A^T + B^T S^{(k)} A\|_F} \tag{3.1}$$

$$= \frac{\sum_{f=1}^{m}\sum_{g=1}^{n} b_{jf} s_{fg}^{(k)} a_{pg} + b_{fj}^{(k)} s_{fg}^{(k)} a_{gp}}{\|BS^{(k)}A^T + B^T S^{(k)} A\|_F} \tag{3.2}$$

So, if we consider this iteratively we get that the similarity scores (at iteration step k) between a vertex in $\overline{v_p}$ and a vertex w_j of $\mathscr{G}_B$ are equal to $s_{jp}^{(k)}$ because for all $g \in \{1, \ldots, n\}$, a_{pg} is the same for all vertices in $\overline{v_p}$ and the same holds for the a_{gp}'s. So all the vertices in $\bar{v_p}$ undergo the same calculations with the entries b_{jf}, b_{fj} $(f \in \{1, \ldots, m\})$. Note that we start with $s_{jp}^{(0)}$ equal to 1.

Because we proved that for any equivalence class in $\sim$ (with more than one element) the similarity scores are equal at each iteration step k, the theorem follows. □

(E7) This condition arose based on intuition. A hypergraph that is not a k-hypergraph, has at least one edge that has a different size. Now, intuitively, when an edge E_i in a hypergraph is compared to an edge E_i' in a hypergraph , it is somehow clear that they must have a higher (edge) similarity score when they connect more vertices. Of course, this only holds in connected hypergraphs (see Condition 8).

One may wonder whether graphs too meet this condition, but having a different edge size only arises in the pathetic case the graph contains a loop. We can show that comparing two loops will normally result in a lower edge similarity score than comparing two edges that aren't loops. We say 'normally', because of course, (C4) or (C8) can always interfere in some graphs. Now, to show this, look at the node-edge similarity method, the edge similarity matrix Y and the node similarity matrix X are calculated as:

$$Y^{(k+1)} = \frac{B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \tag{3.3}$$

$$X^{(k+1)} = \frac{B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \tag{3.4}$$

for $k = 0, 1, \ldots$.

Element-wise, we get:

$$\begin{aligned} y_{ji}^{(k+1)} &= \frac{\sum_{f=1}^{n_{\mathscr{H}}} \sum_{g=1}^{n_{\mathscr{G}}} b_{s_{jf}}^T x_{fg}^{(k)} a_{s_{gi}} + b_{t_{jf}}^T x_{fg}^{(k)} a_{t_{gi}}}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \\ x_{ji}^{(k+1)} &= \frac{\sum_{f=1}^{m_{\mathscr{H}}} \sum_{g=1}^{m_{\mathscr{G}}} b_{s_{jf}} y_{fg}^{(k)} a_{s_{gi}}^T + b_{t_{jf}} y_{fg}^{(k)} a_{st_{gi}}^T}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \end{aligned}$$

which is equal to:

$$\begin{aligned} y_{ij}^{(k+1)} &= \frac{\sum_{f=1}^{n_{\mathscr{H}}} \sum_{g=1}^{n_{\mathscr{G}}} b_{s_{fi}} x_{fg}^{(k)} a_{s_{gj}} + b_{t_{fi}} x_{fg}^{(k)} a_{t_{gj}}}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \\ x_{ij}^{(k+1)} &= \frac{\sum_{f=1}^{m_{\mathscr{H}}} \sum_{g=1}^{m_{\mathscr{G}}} b_{s_{if}} y_{fg}^{(k)} a_{s_{jg}} + b_{t_{if}} y_{fg}^{(k)} a_{st_{gj}}}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \end{aligned}$$

Now let e_j of $\mathscr{G}$ be a loop on vertex v_p and e_i' of $\mathscr{H}$ a loop of vertex v_q', the edge similarity score between e_j and e_i' equals:

$$y_{ji}^{(k+1)} = \frac{b_{s_{qi}} x_{qp}^{(k)} a_{s_{pi}} + b_{t_{qi}} x_{qp}^{(k)} a_{t_{pj}}}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \tag{3.5}$$

$$\Leftrightarrow y_{ji}^{(k+1)} = \frac{2x_{qp}^{(k)}}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \tag{3.6}$$

$$x_{qp}^{(k+1)} = \frac{\sum_{f=1}^{m_{\mathscr{H}}} \sum_{g=1}^{m_{\mathscr{G}}} b_{s_{qf}} y_{fg}^{(k)} a_{s_{pg}} + b_{t_{qf}} y_{fg}^{(k)} a_{st_{gp}}}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \tag{3.7}$$

where we indeed see that the result of y_{ji} is only based on x_{qp} which will be high if both v_p and v_q' are heavy connected to other vertices (see C4). In the case of e_m of $\mathscr{G}_A$ connecting the vertices v_p and v_o and e_n' of $\mathscr{G}_B$ connecting the vertices v_q', v_r' we get the following edge similarity score y_{nm} ($\mathscr{G}_A$ and $\mathscr{G}_B$ are undirected):

$$y_{nm}^{(k+1)} = \frac{2(x_{qp}^{(k)} + x_{q_o}^{(k)} + x_{rp}^{(k)} + x_{ro}^{(k)})}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F},$$

which will normally be higher than (3.6) (keep in mind that condition (C4) and (C8) can occur).

(E8) To see (C8) on undirected graphs, we first notice that indeed isolated vertices have always similarity scores equal to 0 in the method of Blondel (also in the case of directed graphs), for example, let w_j of $\mathscr{H}$ with adjacency matrix B be an isolated vertex and we calculate the similarity scores with the vertices of $\mathscr{G}$, elementwise we write:

$$s_{jp}^{(k+1)} = \frac{\sum_{f=1}^{m} \sum_{g=1}^{n} b_{jf} s_{fg}^{(k)} a_{pg} + b_{fj}^{(k)} s_{fg}^{(k)} a_{gp}}{\|B S^{(k)} A^T + B^T S^{(k)} A\|_F}$$

But since w_j is an isolated vertex we know that all the $b'_{jf}s$ and b_{fj} are equal to zero, so:

$$s_{jp}^{(k+1)} = \frac{\sum_{f=1}^{m}\sum_{g=1}^{n} 0 s_{fg}^{(k)} a_{pg} + 0 s_{fg}^{(k)} a_{gp}}{\|BS^{(k)}A^T + B^T S^{(k)} A\|_F} = 0$$

The condition about connectivity is easy to see: when a matrix is not connected, there exists zeros in the adjacency matrix. Each zero decreases the similarity score, if this results in a very low similarity score in the first iteration steps, it could be that in the limit - caused by the zeros in the adjacency matrix - the similarity scores are only getting lower which will finally result in a similarity score equal to zero.

3.2 Similarity through corresponding graph representations

In the section, we explore representations of hypergraphs as (classical) graphs and use them to calculate the similarity between two hypergraphs by simply using the methods of the previous chapter. Intuitively, we want a characteristic graph representation of a hypergraph that is faithful (meaning that a graph represents only 1 hypergraph), because only a faithful characteristic graph will preserve all the information that the structure of a hypergraph contains. When the characteristic graph of a hypergraph is not faithful, the resulting similarity methods can do a fair job under certain circumstances, but we will see that we can not fulfill all the conditions at the same time because some information of the original hypergraph is lost. To prove that several graph representations of hypergraphs meet certain conditions, we will often refer to the explanations (E_i) in the following way: we explain or prove that the graph representation preserves a certain characteristic of the hypergraph and then we just use the explanations for graphs.

3.2.1 Line-graphs

General definitions and properties

Definition 3.2.1. *Let $\mathcal{H} = (V, E)$ be a hypergraph with $E \neq \emptyset$ and $E = \{E_1, \dots E_m\}$. The **line-graph** of $\mathcal{H}$ is the undirected graph often denoted by $\ell(\mathcal{H}) = (V', \leftrightarrow)$ with:*

1. *$V' = E$,*
2. *$E_i \leftrightarrow E_j$ if and only if $E_i \cap E_j \neq \emptyset$ and $i \neq j$.*

It's immediately clear that line-graphs are simple graphs (see Definition 1.5.14). An important property of a hypergraphs can be seen on a line-graph:

Property 3.2.2. *If $\mathcal{H} = (V, E)$ is connected, then the line-graph $\ell(\mathcal{H})$ is also connected.*

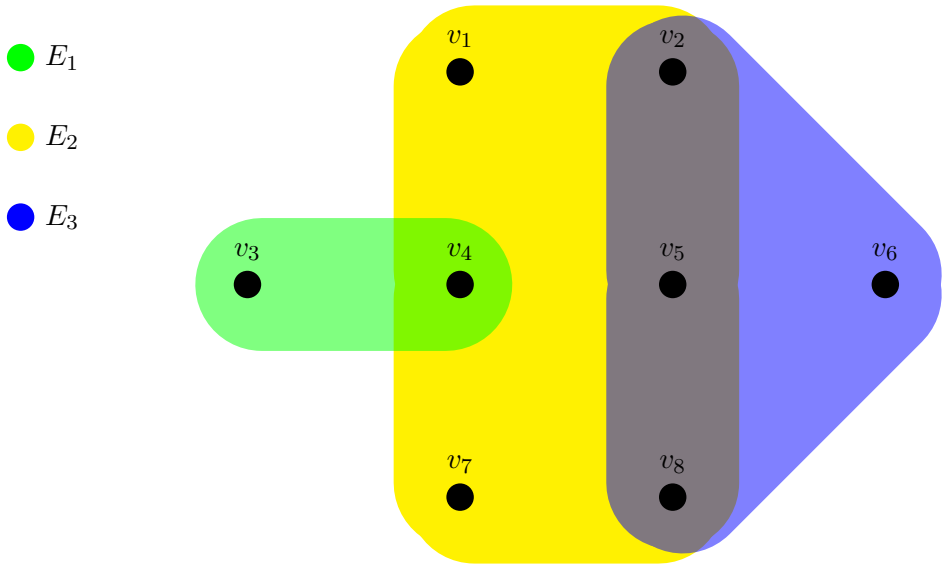
Proof. By contraposition, if $\ell(\mathcal{H})$ is not connected, take E_p, E_q, two vertices of $\ell(\mathcal{H})$ that doesn't have a path between each other. This means that there is no sequence

$$E_p = E_{k_0}, E_{k_1}, \dots, E_{k_{l-1}}, E_q = E_{k_l}$$

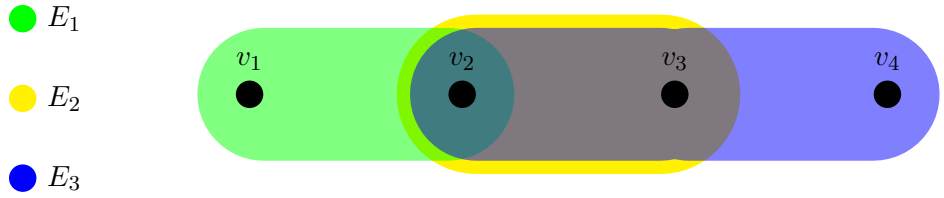
such that $E_{k_{i-1}} \cap E_{k_i} \neq \emptyset$ for $i \in \{0, \dots, l\}$, if $v_p \in E_p$ and $v_q \in E_q$ with v_p, v_q vertices of $\mathcal{H}$, this means that no path exists between v_p and v_q and therefore $\mathcal{H}$ is not connected. □

The following example shows that characteristic line-graph of a hypergraph is not faithful:

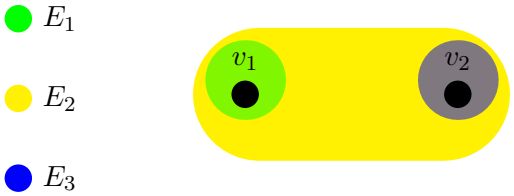
Example 3.2.3. Let $\mathcal{H}_1$ be the following hypergraph:



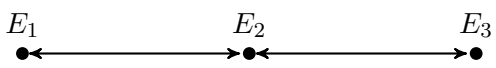
Let $\mathcal{H}_2$ be the following hypergraph:



And let $\mathcal{H}_3$ be the following hypergraph ($\mathcal{H}_3$ is in fact also an undirected graph):



All off $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ lead to the same line-graph $\mathcal{G}$:



Algorithm for similarity

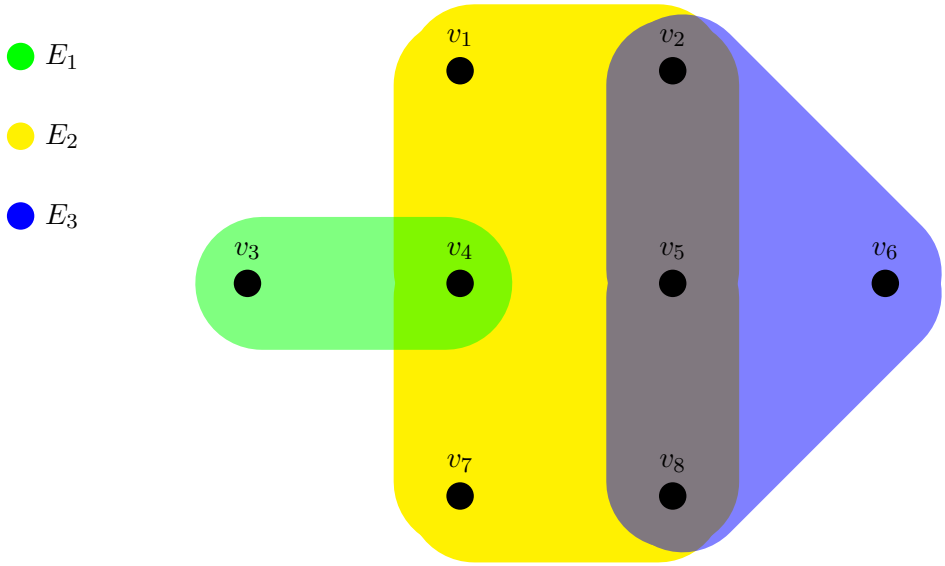
We want to apply Algorithm 5 to produce a similarity matrix between two hypergraphs. This will produces an edge similarity matrix, because the vertices are not represented in the line-graph representation.

To use Algorithm 5 from section 2.2, we first take a hypergraph as input and calculate the adjacency matrix of the corresponding line-graph. The Matlab implementation of this step can be found in Appendix A in Listing A.11. By applying this algorithm to two hypergraphs, we can use Algorithm 5.

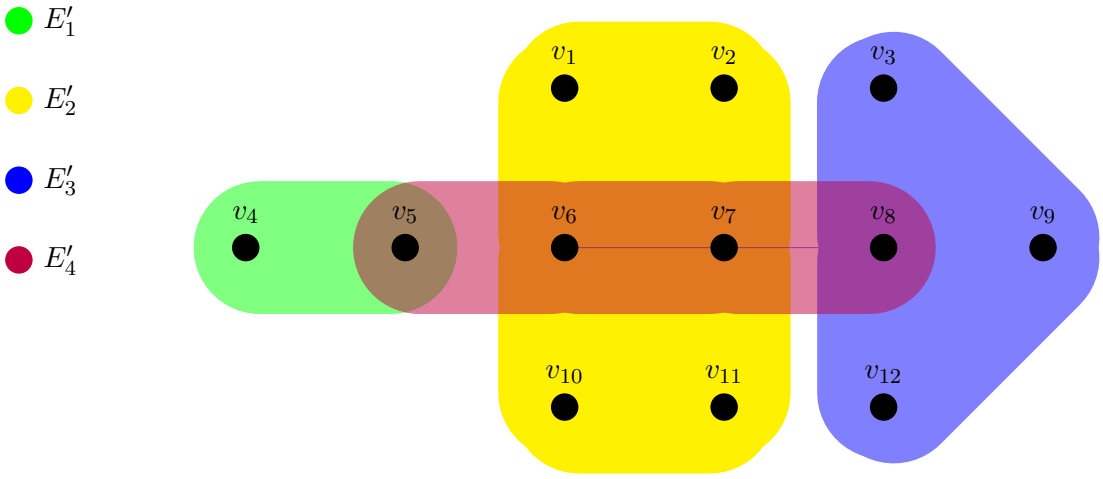
Examples

We now give some examples of the similarity of two hypergraphs by using line-graphs:

Example 3.2.4. Let $\mathcal{H}_1$ be the following hypergraph:



Let $\mathcal{H}_2$ be the following hypergraph:



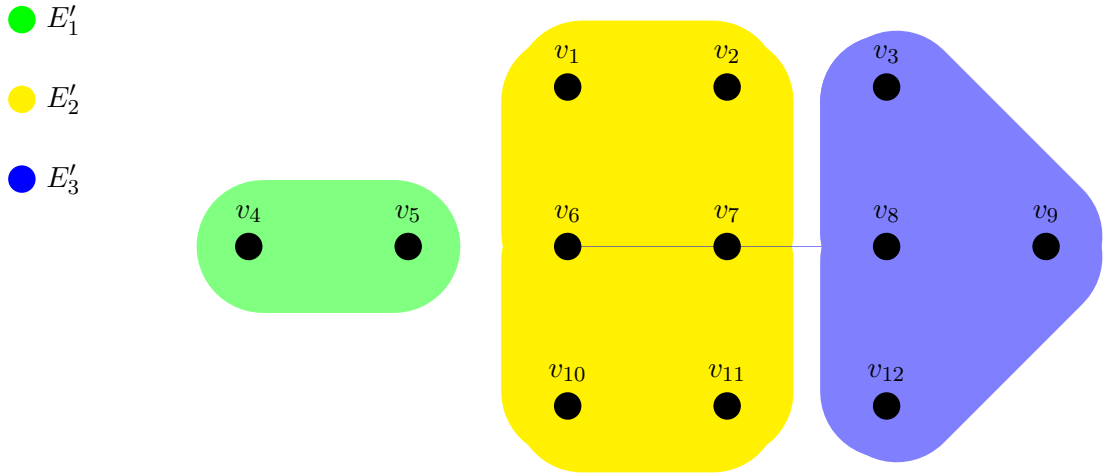
By applying the algorithm we get the adjacency matrices A_1 for $\mathcal{H}_1$ and A_2 for $\mathcal{H}_2$:

$$A_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad \text{and} \quad A_2 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Now we use Algorithm 5 with A_1 and A_2 and get the following similarity matrix:

$$S = \begin{pmatrix} 0.2887 & 0.2887 & 0.2887 \\ 0.2887 & 0.2887 & 0.2887 \\ 0.2887 & 0.2887 & 0.2887 \\ 0.2887 & 0.2887 & 0.2887 \end{pmatrix}$$

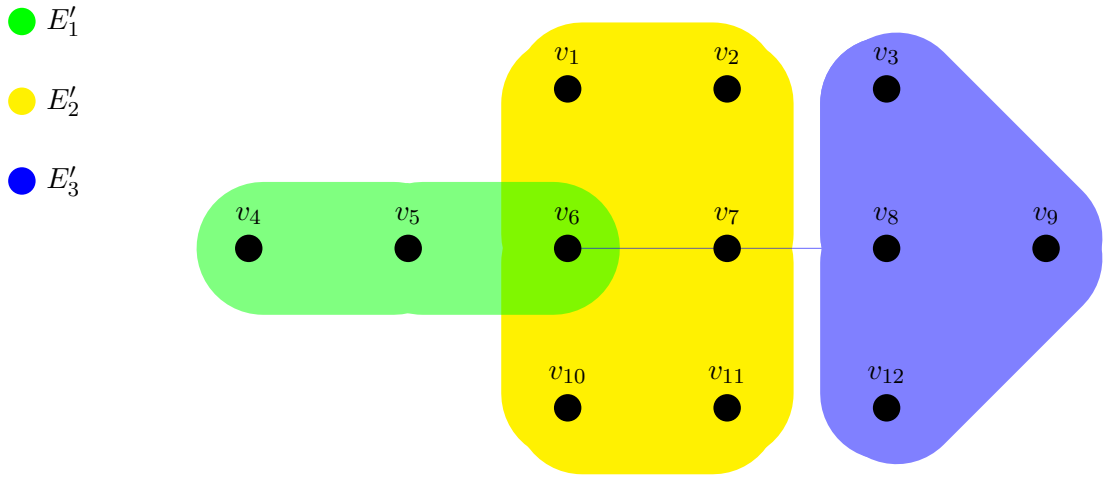
Example 3.2.5. $\mathcal{H}_1$ is the same as in the previous example, but now, $\mathcal{H}_2$ is the following hypergraph:



The similarity score of these two hypergraphs by using their line-graph representation becomes:

$$S = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

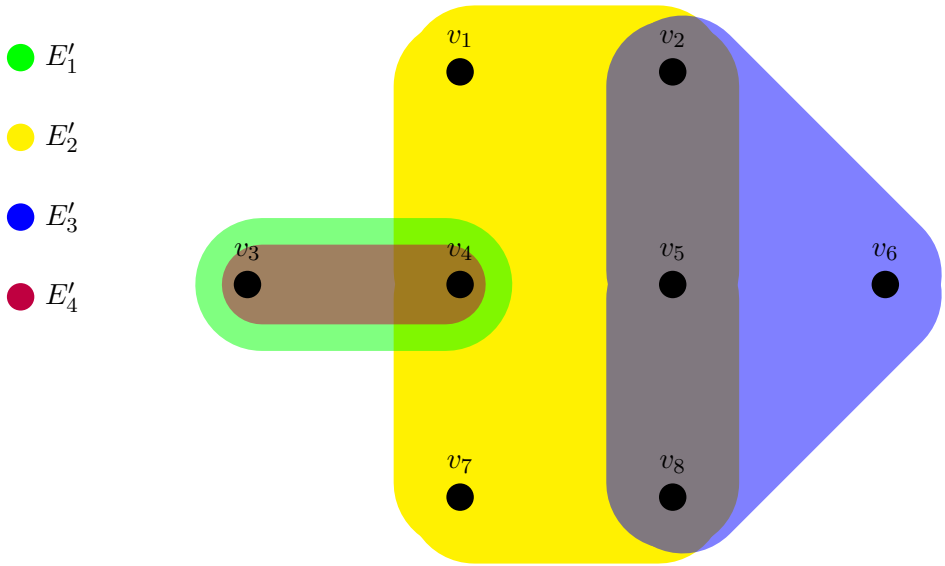
Example 3.2.6. $\mathcal{H}_1$ is the same as in the previous example, but now, $\mathcal{H}_2$ is the following hypergraph:



The similarity score of these two hypergraphs by using their line-graph representation becomes:

$$S = \begin{pmatrix} 0.4082 & 0.4082 & 0.4082 \\ 0.4082 & 0.4082 & 0.4082 \\ 0 & 0 & 0 \end{pmatrix}$$

Example 3.2.7. $\mathcal{H}_1$ is the same as in the previous example, but now, $\mathcal{H}_2$ is the following hypergraph:



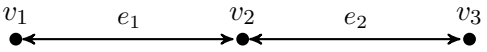
The similarity score of these two hypergraphs by using their line-graph representation becomes:

$$S = \begin{pmatrix} 0.4082 & 0.4082 & 0 & 0.4082 \\ 0.4082 & 0.4082 & 0 & 0.4082 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

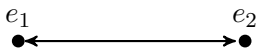
Interpretation

We now discuss the conditions from the introduction:

(C1) Not fulfilled: first, the method will only result an edge similarity matrix and second, the line-graph of an undirected graph is not necessary the undirected graph itself:



Has as line-graph:



which will clearly not result in the same edge similarity scores as no information about the vertex adjacency is saved.

(C2) Not fulfilled: the vertices of the hypergraph doesn't play a role in the line-graph. No information about them is saved in the line-graph representation.

(C3) Fulfilled: Adding an edge to one of the two hypergraphs will result in an extra vertex in the line-graph. Adding this vertex will take the edge into account when calculating the similarity scores and therefore we conclude on a heuristic base that this condition is fulfilled.

(C4) Not fulfilled: in Examples 3.2.4, 3.2.5, 3.2.6 and 3.2.7 all positive similarity scores are the same, regardless of the adjacency structure of the hypergraph.

(C5) Not fulfilled: there is no information about the vertices saved in the line-graph representation of a hypergraph.

(C6) Fulfilled: from Example 3.2.7 we see that the E_1' and E_4' are interchangeable and have indeed the same similarity scores. We will prove this.

Theorem 3.2.8. *The line-graph representation of a hypergraph preserves interchangeable edges.*

Proof. Two edges are interchangeable in a hypergraph if they contain exactly the same vertices. They form an equivalence class on E where $E_i \sim E_j$ if they are interchangeable. Let $\overline{E_i}$ be the class of interchangeable edges with E_i, we assume that this class contains at least 2 edges. Now from Defintion 3.2.1 we see that $E_i \leftrightarrow E_j$ if and only if $E_i \cap E_j \neq \emptyset$ and $i \neq j$ in the linegraph representation, thus all the edges in $\overline{E_i}$ will have exactly the same adjacency structure, making them interchangeable as vertices in the line-graph representation too. □

Because the line-graph representation preserves interchangeable edges and we use the method of Blondel for which we already proved in (E6) that this condition holds, the result follows.

(C7) Not fulfilled: no information on the number of vertices an edge contains is preserverd by the line-graph representation.

(C8) Fulfilled: but it is also the only thing this representation tells us when used for the calculation of similarity: two edges of the two line-graphs have a positive similarity score when these edges are connected to other edges, meaning that for any E_p, E_q there is a sequence

$$E_p = E_{k_0}, E_{k_1}, \ldots, E_{k_{l-1}}, E_q = E_{k_l}$$

such that $E_{k_{i-1}} \cap E_{k_i} \neq \emptyset$.

This follows immediately from Property 3.2.2. The connectivity of edges is indeed represented in the line-graph representation by definition.

Conclusion

The line-graph fails to satisfy lots of the conditions and is therefore not a good representation to calculate similarity between two hypergraphs. Similarity between hypergraphs through line-graphs only allows us to discover groups of connected edges in both hypergraphs. The fact that the line-graph representation is a bit disappointing, was also predictable as a lot of hypegraphs share the same line-graph representation.

3.2.2 2-section of a hypergraph

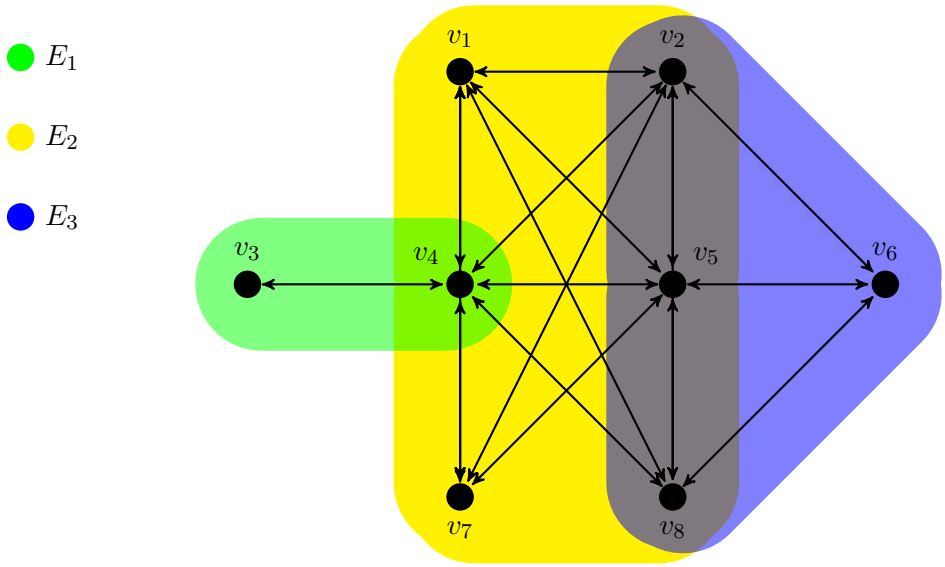
General definitions and properties

We now look at another graph representation of a hypergraph. In contrast to the line-graph representation, the 2-section saves information about the vertices which will introduce a more sophisticated way to say something about similarity between two hypergraphs by using their 2-section.

Definition 3.2.9. *The* 2-***section*** *of a hypergraph* $\mathcal{H} = (V, E)$ *is the (undirected) graph denoted by* $\mathcal{H}_2 = (V, \leftrightarrow)$ with:

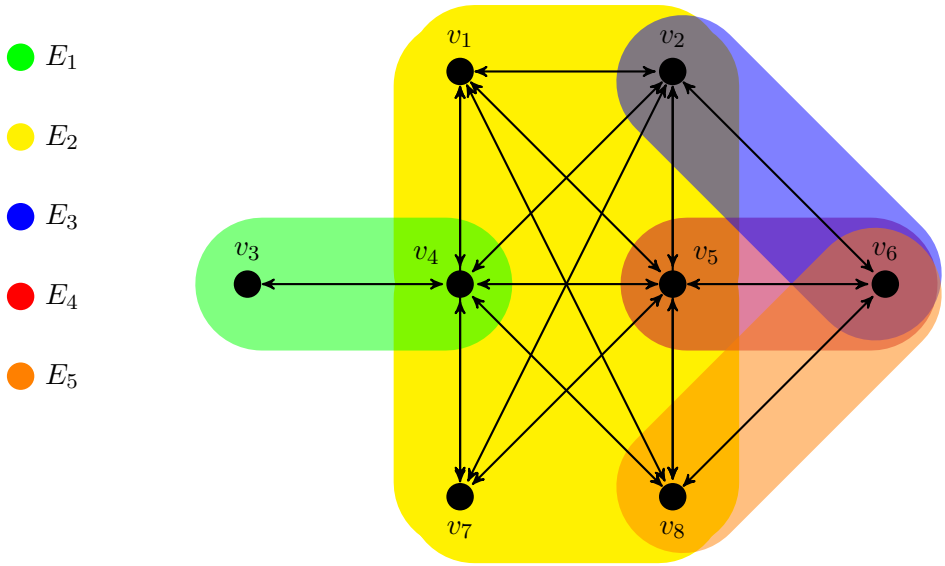
- The same vertex set as the hypergraph,
- $v_i \leftrightarrow v_j$ if and only if $v_i, v_j \in E_k$ for some $E_k \in E$ and $i \neq j$.

Example 3.2.10. The 2-section of the following hypergraph $\mathcal{H}$ is drawn on top:



Also the 2-section of a hypergraph is not a faithful graph characteristic of a hypergraph, as the following example shows:

Example 3.2.11. The 2-section of the following hypergraph $\mathcal{H}'$ is drawn on top and is the same as the previous example:



Algorithm for similarity

In Listing A.12 in Appenix A, we introduce an algorithm that takes a hypergraph as input and returns the adjacency matrix of the corresponding 2-section of the hypergraph. The resulting adjacency matrix can then be used in the node similarity method from Algorithm 5.

Examples

We now use the same examples as in the previous subsection and look at what the similarity of two hypergraphs becomes by using their 2-section.

Example 3.2.12. Take the same $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.4, by calculating the adjacency matrices of the 2-section, and we can apply Algorithm 5 which returns the similarity matrix:

$$S = \begin{pmatrix} 0.1347 & 0.1479 & 0.0261 & 0.1388 & 0.1479 & 0.0833 & 0.1347 & 0.1479 \\ 0.1347 & 0.1479 & 0.0261 & 0.1388 & 0.1479 & 0.0833 & 0.1347 & 0.1479 \\ 0.0263 & 0.0288 & 0.0051 & 0.0271 & 0.0288 & 0.0162 & 0.0263 & 0.0288 \\ 0.0147 & 0.0161 & 0.0028 & 0.0151 & 0.0161 & 0.0091 & 0.0147 & 0.0161 \\ 0.0790 & 0.0867 & 0.0153 & 0.0814 & 0.0867 & 0.0489 & 0.0790 & 0.0867 \\ 0.1610 & 0.1768 & 0.0312 & 0.1659 & 0.1768 & 0.0996 & 0.1610 & 0.1768 \\ 0.1610 & 0.1768 & 0.0312 & 0.1659 & 0.1768 & 0.0996 & 0.1610 & 0.1768 \\ 0.0890 & 0.0977 & 0.0172 & 0.0917 & 0.0977 & 0.0551 & 0.0890 & 0.0977 \\ 0.0263 & 0.0288 & 0.0051 & 0.0271 & 0.0288 & 0.0162 & 0.0263 & 0.0288 \\ 0.1347 & 0.1479 & 0.0261 & 0.1388 & 0.1479 & 0.0833 & 0.1347 & 0.1479 \\ 0.1347 & 0.1479 & 0.0261 & 0.1388 & 0.1479 & 0.0833 & 0.1347 & 0.1479 \\ 0.0263 & 0.0288 & 0.0051 & 0.0271 & 0.0288 & 0.0162 & 0.0263 & 0.0288 \end{pmatrix}$$

Example 3.2.13. Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be the same as in Example 3.2.5, the similarity matrix using the 2-section of the hypergraphs becomes:

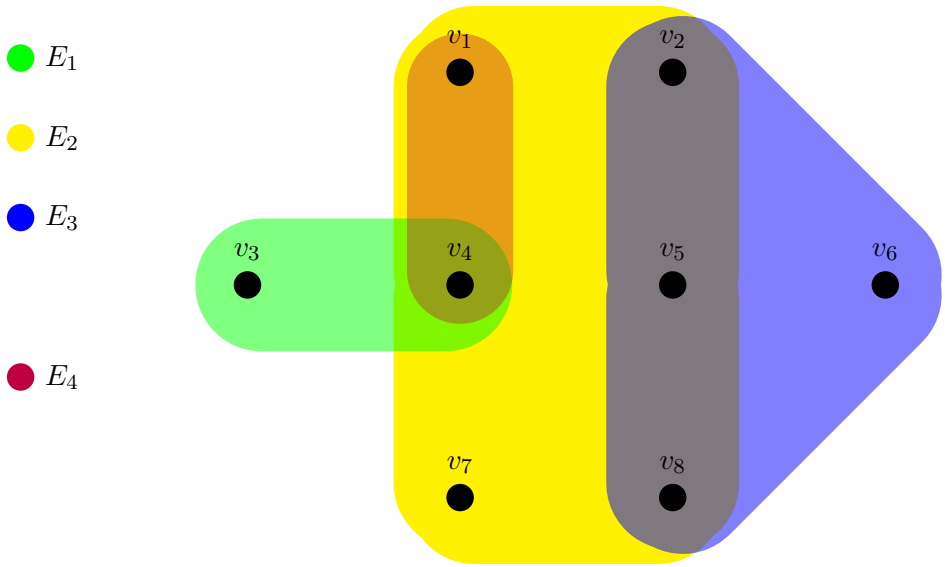
$$S = \begin{pmatrix} 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0.1532 & 0.1682 & 0.0297 & 0.1579 & 0.1682 & 0.0948 & 0.1532 & 0.1682 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Example 3.2.14. Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be the same as in Example 3.2.6, the similarity matrix

using the 2-section of the hypergraphs becomes:

$$S = \begin{pmatrix} 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0397 & 0.0436 & 0.0077 & 0.0409 & 0.0436 & 0.0246 & 0.0397 & 0.0436 \\ 0.0397 & 0.0436 & 0.0077 & 0.0409 & 0.0436 & 0.0246 & 0.0397 & 0.0436 \\ 0.1623 & 0.1782 & 0.0314 & 0.1673 & 0.1782 & 0.1004 & 0.1623 & 0.1782 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Example 3.2.15. Let $\mathcal{H}_2$ be the same as in the previous example (Example 3.2.14), but take for $\mathcal{H}_1$ the following hypergraph:

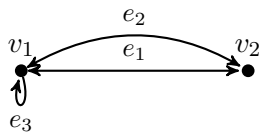


The similarity matrix using the 2-section of the hypergraph becomes the same as in the previous example:

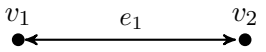
$$S = \begin{pmatrix} 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0397 & 0.0436 & 0.0077 & 0.0409 & 0.0436 & 0.0246 & 0.0397 & 0.0436 \\ 0.0397 & 0.0436 & 0.0077 & 0.0409 & 0.0436 & 0.0246 & 0.0397 & 0.0436 \\ 0.1623 & 0.1782 & 0.0314 & 0.1673 & 0.1782 & 0.1004 & 0.1623 & 0.1782 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0.1493 & 0.1639 & 0.0289 & 0.1538 & 0.1639 & 0.0923 & 0.1493 & 0.1639 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Interpretation

(C1) Not fulfilled: an undirected graph with multiple edges between two vertices v_p, v_q will 'lose' this edges in its 2-section. Also loops are not taken into consideration. Take for example the following graph $\mathscr{G}$:



$\mathscr{G}$ will have as 2-section:



(C2) Fulfilled: the number of vertices is preserved in the 2-section of a hypergraph, so each vertex is taken into account when calculating the similarity matrix. We conclude with the same reasoning as (E2) that the condition is fulfilled.

(C3) Not fulfilled: introducing an edge in a hypergraph that connects vertices who are already connected, doesn't change anything to the 2-section of the hypergraph (by definition), for instance, we see in Example 3.2.15 that get the same similarity matrix as in Example 3.2.14 as introducing edge E_4 doesn't influence the similarity scores at all because v_1, v_4, v_7 were already adjacent to each other in the 2-section of $\mathcal{H}_1$.

(C4) Fulfilled: take for Example 3.2.12, we see that the largest similarity scores occurs between vertices v_6, v_7 in $\mathcal{H}_2$ and vertices v_2, v_5 and v_8 from $\mathcal{H}_1$. This can be explained from the fact that indeed v_6, v_7 are all adjacent to the vertices in E_2. And that E_2 is the most central set of vertices in the whole hypergraph.

This can be explained generally from the fact that the 2-section preserves the adjacency relations between vertices of the hypergraph by definition (namely: a vertex that is adjacent to another vertex in the hypergraph, will also be adjacent in the 2-section). By preserving this relations, we can use the information in (E4) to conclude that this condition is fulfilled.

(C5) This condition does not apply on this method because we don't calculate edge similarity scores.

(C6) Fulfilled: all interchangeable vertices have the same similarity scores in the Examples 3.2.12, 3.2.13, 3.2.14. This is can be explained from the fact that the interchangeable vertices of a hypergraph are also interchangeable vertices in the 2-section graph. We will prove this:

Theorem 3.2.16. *The 2-section of a hypergraph preserves interchangeable vertices.*

Proof. The interchangeable vertices of a hypergraph $\mathcal{G}$ form an equivalence class on the set of vertices V, take an equivalence $\overline{v_i}$ with $|\overline{v_i}| > 1$ (we assume that $\mathcal{G}$ has at least two interchangeable vertices), this means that all vertices in $\overline{v_i}$ have exactly the same

adjacency structure in the hypegraph $\mathcal{G}$. So, when a vertex in $\overline{v_i}$ is adjacent to a vertex v_p, all vertices in $\overline{v_i}$ are adjacent to v_p in the hypergraph $\mathcal{G}$. By the definition of the 2-section, this means that all vertices in $\overline{v_i}$ will also have an edge connecting them to v_p in the 2-section. So in the 2-section, all vertices in $\overline{v_i}$ will be adjacent to the same vertices, making them also interchangeable by the definition of interchangeable vertices in graphs.

□

Because the 2-section preserves interchangeable vertices and we use the method of Blondel for which we already proved in (E6) that this condition holds, the result follows.

(C7) Not fulfilled: the 2-section doesn't save any information about the number of vertices in each edge.

(C8) Fulfilled from the fact that isolated vertices in a hypergraph will also be isolated in the 2-section by definition. Because we already know from (E8) that this condition holds for the method of Blondel, we conclude that this condition is fulfilled because the 2-section preserves connectivy by definition.

Conclusion

The 2-section of a hypergraph is a rich structure that saves a lot more information compared to the the line-graph representation. The biggest drawback for this method is that adding an edge to a hypergraph can sometimes have no effect at all. This happens when the added edge connects vertices which were already connected. This is bad, because adding an edge always should have an impact on the similarity scores as it expresses an additional union between vertices. As a consequence, adjacent vertices that aren't interchangeable can still have the same similarity scores. We saw in Example 3.2.11 that the 2-section of a hypergraph is not unique, meaning that we are loosing certain information on the hypergraph. In this case, we can lose information about the edges as some edges will not be represented in the 2 section and we also lose all information about the number of vertices contained in each edge. Meaning that the number of vertices contained in an edge doesn't play any role when calculating the similarity scores.

We can resolve the problems in conditions (C1) and (C3) by allowing multiple edges between vertices and loops: every edge in the hypergraph is then also represented in this *extended 2-section*.

3.2.3 Extended 2-section of a hypergraph

General definitions and properties

The **extended 2-section** of a hypergraph $\mathcal{H} = (V, E)$ is the (undirected) graph denoted by $\mathcal{H}'_2 = (V, \leftrightarrow)$ with:

- The same vertex set as the hypergraph,
- for every $E_i \in E$ with $|E_i| > 1$: $v_k \leftrightarrow v_l$ for every $v_k, v_l \in E_i and k \neq l$,
- for every $E_i \in E$ with $|E_i| = 1$: $v_k \leftrightarrow v_k$ for $v_k \in E_i$.

Algorithm for similarity

We introduce Algorithm 10 that takes a hypergraph as input and returns the adjacency matrix of the corresponding extended 2-section of the hypergraph. A Matlab implementation can be found in Listing A.13 in Appenix A.

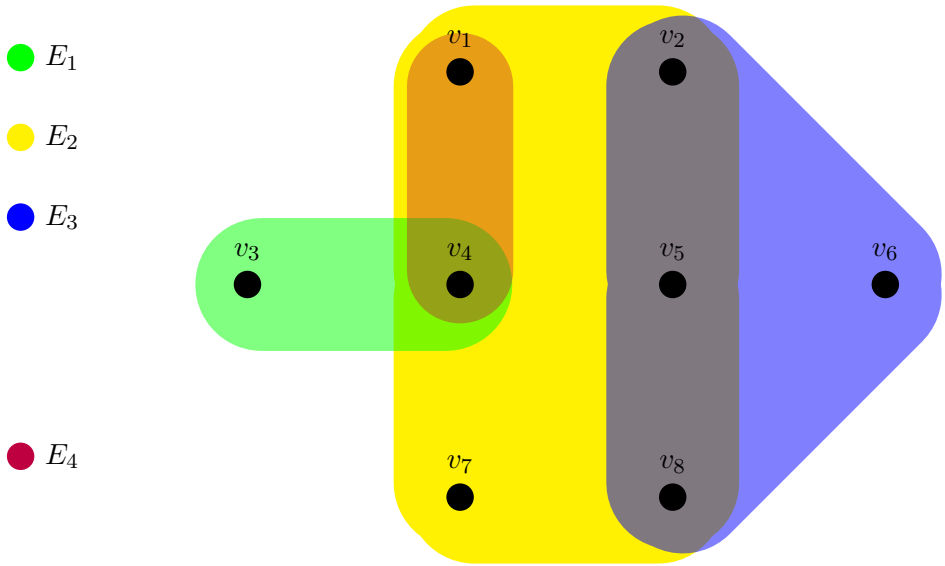
Data:
n: the number of vertices of hypergraph $\mathcal{H}$
E: a set of subsets E_i of $\{1, \ldots, n\}$ that represent the edges of hypergraph $\mathcal{H}$
Result:
A: the adjacency matrix of the corresponding extended 2-section
begin `hypergraph_to_extended2section`(n, E)
    A = initialize a $n \times n$-matrix with all entries equal to 0;
    m = number of edges;
    for $i : 1$ *to* m **do**
        if $|E_m| = 1$ **then**
            k = vertex in E_m;
            $(A)_{kk} = A_{kk} + 1$;
        else
            for $j : 1$ *to* $|E_m|$ **do**
                C = all possible combinations of elements in $E_m \backslash j$;
                for $l : 1$ *to* $|C|$ **do**
                    $A_{jl} = A_{jl} + 1$
                end
            end
        end
    end
end
return A;

Algorithm 10: Algorithm to calculate the adjacency matrix of the extended 2-section of a hypergraph.

Example

We take an example that really shows the power of extended 2-sections:

Example 3.2.17. Take $\mathcal{H}_1$ as in Example 3.2.4 and $\mathcal{H}_2$:



The extended 2-section has as adjacency matrices for $\mathcal{H}_1$ and $\mathcal{H}_2$:

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 2 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 2 & 0 & 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 2 & 0 & 1 & 2 & 1 & 1 & 0 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & 1 & 0 & 2 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 2 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 2 & 0 & 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 2 & 0 & 1 & 2 & 1 & 1 & 0 \end{pmatrix}$$

Remember that the 'normal' 2-section for $\mathcal{H}_1$ and $\mathcal{H}_2$ would return the following adjacency matrices:

$$A' = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \quad \text{and} \quad B' = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

The node similarity matrix with the extended 2-section is:

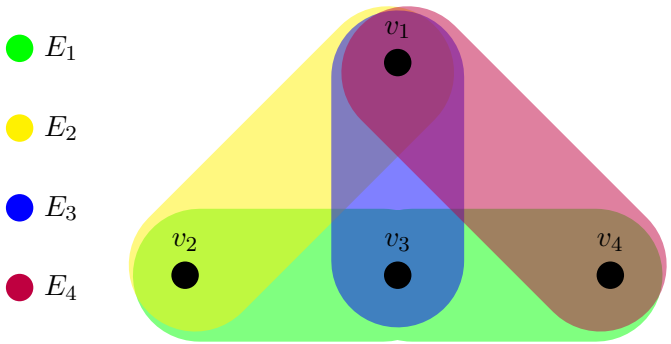
$$S = \begin{pmatrix} \mathbf{0.1108} & \mathbf{0.1651} & \mathbf{0.0174} & \mathbf{0.1132} & \mathbf{0.1651} & \mathbf{0.0763} & \mathbf{0.1108} & \mathbf{0.1651} \\ 0.1409 & 0.2099 & 0.0222 & 0.1438 & 0.2099 & 0.0970 & 0.1409 & 0.2099 \\ 0.0168 & 0.0250 & 0.0026 & 0.0171 & 0.0250 & 0.0116 & 0.0168 & 0.0250 \\ 0.1128 & 0.1680 & 0.0177 & 0.1151 & 0.1680 & 0.0776 & 0.1128 & 0.1680 \\ 0.1409 & 0.2099 & 0.0222 & 0.1438 & 0.2099 & 0.0970 & 0.1409 & 0.2099 \\ 0.0629 & 0.0937 & 0.0099 & 0.0643 & 0.0937 & 0.0433 & 0.0629 & 0.0937 \\ \mathbf{0.0962} & \mathbf{0.1433} & \mathbf{0.0151} & \mathbf{0.0982} & \mathbf{0.1433} & \mathbf{0.0663} & \mathbf{0.0962} & \mathbf{0.1433} \\ 0.1409 & 0.2099 & 0.0222 & 0.1438 & 0.2099 & 0.0970 & 0.1409 & 0.2099 \end{pmatrix}$$

Conversely, the node similarity matrix with the 'normal' 2-section would return:

$$S' = \begin{pmatrix} \mathbf{0.1409} & \mathbf{0.1547} & \mathbf{0.0273} & \mathbf{0.1452} & \mathbf{0.1547} & \mathbf{0.0872} & \mathbf{0.1409} & \mathbf{0.1547} \\ 0.1547 & 0.1698 & 0.0299 & 0.1594 & 0.1698 & 0.0957 & 0.1547 & 0.1698 \\ 0.0273 & 0.0299 & 0.0053 & 0.0281 & 0.0299 & 0.0169 & 0.0273 & 0.0299 \\ 0.1452 & 0.1594 & 0.0281 & 0.1496 & 0.1594 & 0.0898 & 0.1452 & 0.1594 \\ 0.1547 & 0.1698 & 0.0299 & 0.1594 & 0.1698 & 0.0957 & 0.1547 & 0.1698 \\ 0.0872 & 0.0957 & 0.0169 & 0.0898 & 0.0957 & 0.0539 & 0.0872 & 0.0957 \\ \mathbf{0.1409} & \mathbf{0.1547} & \mathbf{0.0273} & \mathbf{0.1452} & \mathbf{0.1547} & \mathbf{0.0872} & \mathbf{0.1409} & \mathbf{0.1547} \\ 0.1547 & 0.1698 & 0.0299 & 0.1594 & 0.1698 & 0.0957 & 0.1547 & 0.1698 \end{pmatrix}$$

The most important thing to notice here is the difference in similarity scores of vertices v_1, v_7 of $\mathcal{H}_2$: in the extended 2-section these vertices have different similarity scores which is correct as v_1 is also contained in edge E_4 and therefore, v_1 and v_7 are not interchangeable. Conversely, in the 'normal' 2-section, the representation doesn't take E_4 into account, leading to the same similarity scores for v_1, v_7.

Example 3.2.18. Take $\mathcal{H}_1$ as in Example 3.2.4 and $\mathcal{H}_2$ as:



The similarity matrix becomes:

$$S = \begin{pmatrix} 0.1566 & 0.2332 & 0.0246 & 0.1599 & 0.2332 & 0.1078 & 0.1566 & 0.2332 \\ 0.1566 & 0.2332 & 0.0246 & 0.1599 & 0.2332 & 0.1078 & 0.1566 & 0.2332 \\ 0.1566 & 0.2332 & 0.0246 & 0.1599 & 0.2332 & 0.1078 & 0.1566 & 0.2332 \\ 0.1566 & 0.2332 & 0.0246 & 0.1599 & 0.2332 & 0.1078 & 0.1566 & 0.2332 \end{pmatrix}$$

Based on this results, we could conclude that all vertices in $\mathcal{G}_2$ are interchangeable (C7), which is not the case: following Definition 3.1.2, only v_2, v_3, v_4 are interchangeable.

Interpretation

(C1) Fulfilled: it's trivial to see that by definition of the extended 2-section, an undirected graph will have exactly the same vertices and exactly the same edges in it's extended 2-section.

(C2) Fulfilled: the number of vertices is preserved in the extended 2-section of a hypergraph, so each vertex is taken into account when calculating the similarity matrix. We conclude with the same reasoning as (E2) that the condition is fulfilled *(equal to the 'normal' 2-section)*.

(C3) Fulfilled: the number of edges is preserved in the extended 2-section of a hypergraph, so each edge is taken into account when calculating the similarity matrix. We conclude with the same reasoning as (E3) that the condition is fulfilled.

(C4) Fulfilled: the extended 2-section preserves the adjacency relations between vertices of the hypergraph by definition (namely: a vertex that is adjacent to another vertex in the hypergraph, will also be adjacent in the extended 2-section). By preserving this relations, we can use the information in (E4) to conclude that this condition is fulfilled. *(equal to the 'normal' 2-section)*

(C5) This condition does not apply on this method because we don't calculate edge similarity scores.

(C6) Fulfilled.

Theorem 3.2.19. *The extended 2-section of a hypergraph preserves interchangeable vertices.*

Proof. The interchangeable vertices of a hypergraph $\mathcal{G}$ form an equivalence class on the set of vertices V, take an equivalence class $\overline{v_i}$ with $|\overline{v_i}| > 1$ (we assume that $\mathcal{G}$ has at least two interchangeable vertices), this means that all vertices in $\overline{v_i}$ have exactly the same adjacency structure in the hypegraph $\mathcal{G}$. So, when a vertex in $\overline{v_i}$ is adjacent to a vertex v_p with k edges that connect both vertices (the size of these edges doesn't matter), all vertices in $\overline{v_i}$ are adjacent with k edges to v_p in the hypergraph $\mathcal{G}$. By the definition of the extended 2-section, this means that all vertices in $\overline{v_i}$ will also have k edges connecting them to v_p in the 2-section. So in the extended 2-section, all vertices in $\overline{v_i}$ will be adjacent to the same vertices, and connecting each vertex $\overline{v_i}$ with the same amount of edges to these vertices, making them also interchangeable in the extended 2-section by the definition of interchangeable vertices in graphs.

□

Because the 2-section preserves interchangeable vertices and we use the method of Blondel for which we already proved in (E6) that this condition holds, the result follows. *(equal equal to the 'normal' 2-section)*

(C7) Not fulfilled: the extended 2-section doesn't save any information about the number of vertices contained in an edge.

(C8) Fulfilled from the fact that isolated vertices in a hypergraph will also be isolated in the extended 2-section by definition. Because we already know from (E9) that this condition holds for the method of Blondel, we conclude that this condition is fulfilled because the extended 2-section preserves connectivity by defintion. *(equal to the 'normal' 2-section)*

Conclusion

The extended 2-section solves all the issues with (C1) and (C3) as it preserves all the information about the number of edges connecting vertices and the adjacency of the vertices. Still, this representation is not unique as no information about the cardinality of the edges is saved, therefore, the method fails to (C8). Therefore, the method preforms much better than

the normal 2-section, but is unreliable in detecting differences between edges. Still, if one is willing to accept this limitation, by example in the case of an application that is not focussed on detecting differences in number of vertices in th edges, the extended 2-section can be an option. Also note that it is impossible to calculate edge similarity scores with the extended 2-section: an edge of a hypergraph is translated into multiple vertices in the 2-section so it would be impossible to satisfy (C5) and (C7).

3.2.4 The incidence graph of a hypergraph

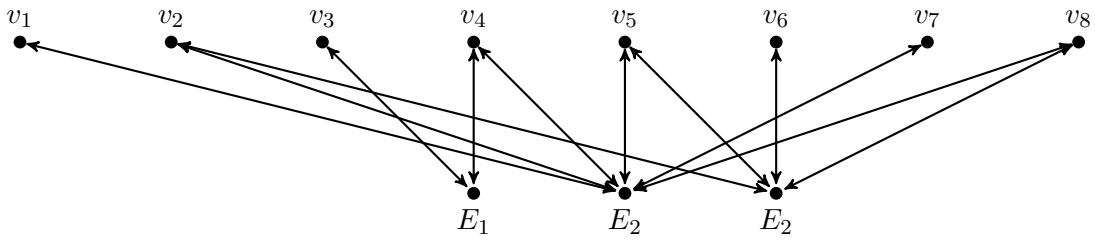
General definitions and properties

Definition 3.2.20. *Let $\mathcal{H} = (V, E)$ be a hypergraph, then the **incidence graph** $\mathscr{G}_i$ of $\mathcal{H}$ is the undirected graph with:*

1. $V' = V \cup E$,
2. $\forall v_i \in V, \forall E_j \in E : v_i \leftrightarrow E_j$ *if* $v_i \in E_j$.

Because all edges in $\mathscr{G}_i$ are between an element of V and E, $_i$ is a bipartite graph and we write $_i = ((V, E), \leftrightarrow)$.

Example 3.2.21. Take the same hypergraph $\mathcal{H}$ as in Example 3.2.10, then the incidence graph $\mathscr{G}_i$ equals:



We can prove that the incidence graph $\mathscr{G}_i = (W = (V, E), \leftrightarrow)$ of a hypergraph is a faitfhul characteristic graph representation if we know it's bipartite structure.

Theorem 3.2.22. *The incidence graph $\mathscr{G}_i = (W = (V, E), \rightarrow)$ of a hypergraph is a faithful representation: the incidence graph represents only one hypergraph under the condition that we know the bipartite structure, which means that for the vertex set W, we know which disjoint subset represents the vertices V of the hypergraph, respectively the edges E of the hypergraph.*

Proof. Suppose that $\mathcal{G}$ and $\mathcal{H}$ are two hypergraphs that share the same incidence graph $\mathscr{G}_i = ((V, E), \rightarrow)$. Then $\mathcal{G}$ and $\mathcal{H}$ have the same set of vertices V and the same set of edges E. Also the adjacency relations in $\mathcal{G}$ and $\mathcal{H}$ are the same by the edge set $\rightarrow$ of $\mathscr{G}_i$. We conclude that $\mathcal{G} = \mathcal{H}$. □

Algorithm for similarity

In Listing A.14 in Appenix A, we introduce an algorithm that takes a hypergraph as input and returns the adjacency matrix of the corresponding incidence graph. The resulting adjacency matrix can then be used in the node similarity method from Algorithm 5.

An important note has to be made here: this algorithm will return both the node and edge similarity scores in the same matrix. Also similarity scores will be available to compare an edge to a node and vice versa. Since we can not give a correct meaning to such similarity scores, we regard them as redundant, intermediate results. In the examples, we will always draw lines in order to clearly separate the node similarity submatrix and the edge similarity submatrix.

Examples

Example 3.2.23. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.4, the similarity matrix with the incidence graph representation becomes:

$$\left(\begin{array}{cccccccc|ccc}
0.0631 & 0.1075 & 0.0101 & 0.0732 & 0.1075 & 0.0444 & 0.0631 & 0.1075 & 0.0170 & 0.1065 & 0.0748 \\
0.0631 & 0.1075 & 0.0101 & 0.0732 & 0.1075 & 0.0444 & 0.0631 & 0.1075 & 0.0170 & 0.1065 & 0.0748 \\
0.0129 & 0.0220 & 0.0021 & 0.0150 & 0.0220 & 0.0091 & 0.0129 & 0.0220 & 0.0035 & 0.0217 & 0.0153 \\
0.0081 & 0.0138 & 0.0013 & 0.0094 & 0.0138 & 0.0057 & 0.0081 & 0.0138 & 0.0022 & 0.0137 & 0.0096 \\
0.0517 & 0.0880 & 0.0082 & 0.0599 & 0.0880 & 0.0363 & 0.0517 & 0.0880 & 0.0139 & 0.0871 & 0.0612 \\
0.1067 & 0.1817 & 0.0170 & 0.1237 & 0.1817 & 0.0750 & 0.1067 & 0.1817 & 0.0287 & 0.1799 & 0.1265 \\
0.1067 & 0.1817 & 0.0170 & 0.1237 & 0.1817 & 0.0750 & 0.1067 & 0.1817 & 0.0287 & 0.1799 & 0.1265 \\
0.0565 & 0.0961 & 0.0090 & 0.0655 & 0.0961 & 0.0397 & 0.0565 & 0.0961 & 0.0152 & 0.0952 & 0.0669 \\
0.0129 & 0.0220 & 0.0021 & 0.0150 & 0.0220 & 0.0091 & 0.0129 & 0.0220 & 0.0035 & 0.0217 & 0.0153 \\
0.0631 & 0.1075 & 0.0101 & 0.0732 & 0.1075 & 0.0444 & 0.0631 & 0.1075 & 0.0170 & 0.1065 & 0.0748 \\
0.0631 & 0.1075 & 0.0101 & 0.0732 & 0.1075 & 0.0444 & 0.0631 & 0.1075 & 0.0170 & 0.1065 & 0.0748 \\
0.0129 & 0.0220 & 0.0021 & 0.0150 & 0.0220 & 0.0091 & 0.0129 & 0.0220 & 0.0035 & 0.0217 & 0.0153 \\
\hline
0.0123 & 0.0209 & 0.0020 & 0.0143 & 0.0209 & 0.0086 & 0.0123 & 0.0209 & 0.0033 & 0.0207 & 0.0146 \\
0.0958 & 0.1632 & 0.0153 & 0.1111 & 0.1632 & 0.0674 & 0.0958 & 0.1632 & 0.0258 & 0.1616 & 0.1136 \\
0.0196 & 0.0333 & 0.0031 & 0.0227 & 0.0333 & 0.0138 & 0.0196 & 0.0333 & 0.0053 & 0.0330 & 0.0232 \\
0.0661 & 0.1126 & 0.0106 & 0.0767 & 0.1126 & 0.0465 & 0.0661 & 0.1126 & 0.0178 & 0.1115 & 0.0784
\end{array}\right)$$

Example 3.2.24. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.5, the similarity matrix with the incidence graph representation becomes:

$$\left(\begin{array}{cccccccc|ccc}
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
\hline
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0920 & 0.1566 & 0.0147 & 0.1067 & 0.1566 & 0.0647 & 0.0920 & 0.1566 & 0.0247 & 0.1551 & 0.1090 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000
\end{array}\right)$$

Example 3.2.25. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.6, the similarity matrix with the incidence graph representation becomes:

$$\left(\begin{array}{cccccccc|ccc}
0.0839 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0839 & 0.1428 & 0.0226 & 0.1414 & 0.0994 \\
0.0839 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0839 & 0.1428 & 0.0226 & 0.1414 & 0.0994 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0254 & 0.0432 & 0.0041 & 0.0294 & 0.0432 & 0.0178 & 0.0254 & 0.0432 & 0.0068 & 0.0428 & 0.0301 \\
0.0254 & 0.0432 & 0.0041 & 0.0294 & 0.0432 & 0.0178 & 0.0254 & 0.0432 & 0.0068 & 0.0428 & 0.0301 \\
0.1092 & 0.1860 & 0.0174 & 0.1267 & 0.1860 & 0.0768 & 0.1092 & 0.1860 & 0.0294 & 0.1842 & 0.1295 \\
0.0839 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0839 & 0.1428 & 0.0226 & 0.1414 & 0.0994 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0839 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0839 & 0.1428 & 0.0226 & 0.1414 & 0.0994 \\
0.0839 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0839 & 0.1428 & 0.0226 & 0.1414 & 0.0994 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
\hline
0.0302 & 0.0514 & 0.0048 & 0.0350 & 0.0514 & 0.0212 & 0.0302 & 0.0514 & 0.0081 & 0.0509 & 0.0358 \\
0.0997 & 0.1697 & 0.0159 & 0.1156 & 0.1697 & 0.0701 & 0.0997 & 0.1697 & 0.0268 & 0.1681 & 0.1181 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000
\end{array}\right)$$

Example 3.2.26. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.7, the similarity matrix with the incidence graph representation becomes:

$$\left(\begin{array}{cccccccc|ccc}
0.0565 & 0.0961 & 0.0090 & 0.0655 & 0.0961 & 0.0397 & 0.0565 & 0.0961 & 0.0152 & 0.0952 & 0.0669 \\
0.0944 & 0.1608 & 0.0151 & 0.1095 & 0.1608 & 0.0664 & 0.0944 & 0.1608 & 0.0254 & 0.1592 & 0.1119 \\
0.0253 & 0.0431 & 0.0040 & 0.0293 & 0.0431 & 0.0178 & 0.0253 & 0.0431 & 0.0068 & 0.0427 & 0.0300 \\
0.0818 & 0.1392 & 0.0130 & 0.0948 & 0.1392 & 0.0575 & 0.0818 & 0.1392 & 0.0220 & 0.1379 & 0.0969 \\
0.0944 & 0.1608 & 0.0151 & 0.1095 & 0.1608 & 0.0664 & 0.0944 & 0.1608 & 0.0254 & 0.1592 & 0.1119 \\
0.0379 & 0.0646 & 0.0061 & 0.0440 & 0.0646 & 0.0267 & 0.0379 & 0.0646 & 0.0102 & 0.0640 & 0.0450 \\
0.0565 & 0.0961 & 0.0090 & 0.0655 & 0.0961 & 0.0397 & 0.0565 & 0.0961 & 0.0152 & 0.0952 & 0.0669 \\
0.0944 & 0.1608 & 0.0151 & 0.1095 & 0.1608 & 0.0664 & 0.0944 & 0.1608 & 0.0254 & 0.1592 & 0.1119 \\
\hline
0.0237 & 0.0403 & 0.0038 & 0.0275 & 0.0403 & 0.0166 & 0.0237 & 0.0403 & 0.0064 & 0.0399 & 0.0281 \\
0.1057 & 0.1800 & 0.0169 & 0.1226 & 0.1800 & 0.0743 & 0.1057 & 0.1800 & 0.0284 & 0.1783 & 0.1253 \\
0.0710 & 0.1210 & 0.0113 & 0.0824 & 0.1210 & 0.0499 & 0.0710 & 0.1210 & 0.0191 & 0.1198 & 0.0842 \\
0.0237 & 0.0403 & 0.0038 & 0.0275 & 0.0403 & 0.0166 & 0.0237 & 0.0403 & 0.0064 & 0.0399 & 0.0281
\end{array}\right)$$

Example 3.2.27. Take $\mathcal{G}_1$ and $\mathcal{G}_2$ as in Example 3.2.18, the similarity matrix becomes:

$$\left(\begin{array}{cccccccc|ccc}
0.1034 & 0.1761 & 0.0165 & 0.1199 & 0.1761 & 0.0727 & 0.1034 & 0.1761 & 0.0278 & 0.1744 & 0.1226 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941 \\
\hline
0.1034 & 0.1761 & 0.0165 & 0.1199 & 0.1761 & 0.0727 & 0.1034 & 0.1761 & 0.0278 & 0.1744 & 0.1226 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941 \\
0.0794 & 0.1352 & 0.0127 & 0.0921 & 0.1352 & 0.0558 & 0.0794 & 0.1352 & 0.0214 & 0.1339 & 0.0941
\end{array}\right)$$

Interpretation

(C1) Fulfilled: we will prove later in Corallary 3.3.7 that using this method with two undirected graphs returns the same results as the node-edge similarity matrix of section

2.3.

(C2) Fulfilled: the number of vertices is preserved in the incidence graph a hypergraph, so each vertex is taken into account when calculating the similarity matrix. We conclude with the same reasoning as (E2) that the condition is fulfilled.

(C3) Fulfilled from the fact that all edges are represented in the incidence graph of the hypergraph, so each edge is taken into account when calculating the similarity matrix. We conclude with the same reasoning as (E3) that the condition is fulfilled.

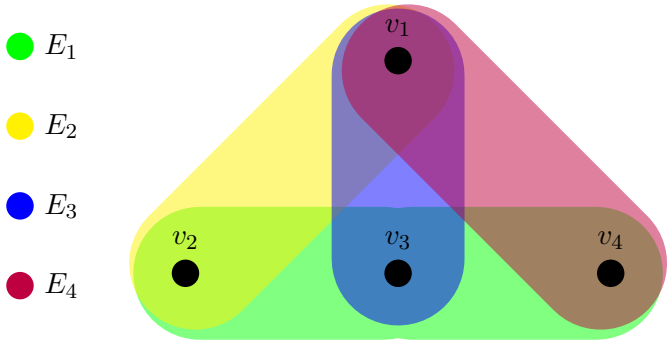
(C4) Fulfilled from the fact that the adjacency relations of the original hypergraph are represented in it's incidence graph and that we use Algorithm 5, for which this statement already holds by (E4).

An example can be found in Example 3.2.23 where v_2, v_5, v_8 of $\mathcal{H}_1$ have the largest similarity score with v_6, v_7 of $\mathcal{H}_2$. This is not surprising as v_6, v_7 are contained in all possible edges.

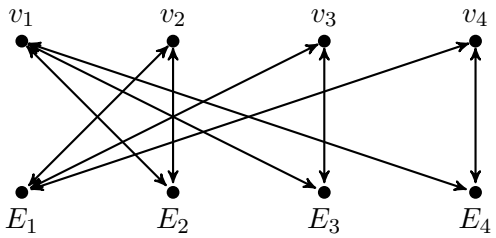
(C5) Fulfilled from the fact that the adjacency relations of the original hypergraph are represented in it's incidence graph, that the edges are considered as normal vertices and that we use Algorithm 5, for which this statement already holds by (E4).

An example can be found in Example 3.2.23 where E_2 of $\mathcal{H}_1$ has the largest similarity score with E_2' of $\mathcal{H}_2$. This is not surprising as these edges are almost equal.

(C6) Fulfilled. Consider $\mathcal{G}_2$ from Example 3.2.18:



The incidence graph becomes:



we see that interchangeable vertices are also interchangeable in the incidence graph (using the circular definition) in the sense that $E2$, $E3$ and $E4$ are interchangeable edges and v_2, v_3, v_4 are all connected to $E1$ and one of these interchangeable edges.

An example of interchangeable edges can be found in Example 3.2.26, where E_1' and E_4' of $\mathcal{H}_2$ have the same similarity scores. In the same example, the vertices v_1, v_7 of $\mathcal{H}_2$ as well as v_2, v_5, v_8.

We know prove that the incidence graph preserves the interchangeability of edges and vertices:

Theorem 3.2.28. *The incidence graph of a hypergraph preserves interchangeable vertices.*

Proof. The interchangeable vertices of a hypergraph $\mathcal{G}$ form an equivalence class on the set of vertices V, take an equivalence $\overline{v_i}$ with $|\overline{v_i}| > 1$ (we assume that $\mathcal{G}$ has at least two interchangeable vertices), this means that all vertices in $\overline{v_i}$ have exactly the same adjacency structure in the hypegraph $\mathcal{G}$. So, when a vertex in $\overline{v_i}$ is adjacent to a vertex v_p, all vertices in $\overline{v_i}$ are adjacent with the same edge E_k to v_p in the hypergraph $\mathcal{G}$. By the definition of the incidence graph, this means that all vertices in in $\overline{v_i}$ will be connected to the edge E_k connecting them to v_p in the incidence graph. So in the incidence graph, all vertices in $\overline{v_i}$ will be adjacent to the same vertices and the same edges, making them also interchangeable by the definition of interchangeable vertices in graphs. With the same reasoning, we conlcude that also the interchangeable edges are preserved.

□

(C7) Fulfilled: the adjancency relations in the incidence graph are determined by the number of vertices contained in an edge in the hypergraph.

(C8) Fulfilled. An example is Example 3.2.6 where the vertices v_3, v_8, v_9 and v_12 have all similarity scores equal to zero as they form a clique that is not connected to the other vertices. The incidence graph also preserves the adjancency relations and thus connectivity, so we know from (E8) that this condition holds.

Conclusion

We conclude that using the incidence graph returns similarity scores that satisfy all the conditions and can be seen as a very good way to calculate the similarity scores of a hypergraph. By Theorem 3.2.22 this is not so surprising: the incidence graph is a faithful representation. Intuitively, this means that this representation saves all information of the represented hypergraph, meaning that it is possible to reconstruct the hypergraph based on its incidece graph. As a result, no information of the hypergraph is lost when using Algorithm 5, so it is possible to satisfy all the conditions.

3.3 Similarity by using the incidence matrix

We already showed that using the incidence graph representation of a hypergraph is a very good way to calculate similarity between hypergraphs. It would be somehow logic to stop searching for similarity methods for hypergraphs now, but there is one very natural generalization that is worth mentioning: the node-edge similarity method described in Section 2.3 uses a source-edge matrix and a terminus-edge matrix. When we would use this method (see Theorem 2.3.6) with undirected graphs $\mathscr{G}$ and $\mathscr{H}$, the source-edge matrix and terminus-edge matrix are the same and are equal to the *incidence matrices* of $\mathscr{G}$ and $\mathscr{H}$. The question is now: is there any problem if we enter not the incidence matrices of two graphs, but of two hypergraphs (see Definition 1.6.10)? The only difference is that each column can have more than 2 entries equal to 1. The answer is no.

Even better, we will be able to prove that this method is equal to the method with the incidence graph up to a constant! By this 'concluding theorem', we know that also this methods meets all conditions imposed in the introduction.

We first prove the compact form of the method:

3.3.1 Compact form

Theorem 3.3.1. *Let $\mathcal{G} = (V, E)$ and $\mathcal{H} = (V', E')$ be two hypergraphs, $\mathcal{G}$ has $n_{\mathcal{G}}$ vertices and $m_{\mathcal{G}}$ eges and $\mathcal{H}$ has $n_{\mathcal{H}}$ vertices and $m_{\mathcal{H}}$ edges. Let A and B be the incidence matrices of $\mathcal{G}$ and $\mathcal{H}$ and define:*

$$Y^{(k+1)} = \frac{B^T X^{(k)} A}{\|B^T X^{(k)} A\|_F} \tag{3.8}$$

$$X^{(k+1)} = \frac{B Y^{(k)} A^T}{\|B Y^{(k)} A^T\|_F} \tag{3.9}$$

for $k = 0, 1, \ldots$.

Then the matrix subsequences $X^{(2k)}$, $Y^{(2k)}$ and $X^{(k+1)}$, $Y^{(k+1)}$ converge to X_{even}, Y_{even} and X_{odd}, Y_{odd}. If we take:

$$X^{(0)} = J \in \mathbb{R}^{n_{\mathscr{H}} \times n_{\mathscr{G}}}$$
$$Y^{(0)} = J \in \mathbb{R}^{m_{\mathscr{H}} \times m_{\mathscr{G}}}$$

as initial matrices, then $X_{even}(\mathbf{1}) = X_{odd}(\mathbf{1})$, $Y_{even}(\mathbf{1}) = Y_{odd}(\mathbf{1})$ are the unique matrices of largest 1-norm among all possible limits with positive initial matices and the matrix sequence $X^{(k)}$, $Y^{(k)}$ converges has a whole.

Proof. The only thing we have to prove is that we can construct a matrix M that is nonnegative and symmetric, the rest of the proof is completely analogous to the proof of Theorem 2.3.6.

So, by Theorem 2.2.14 we can rewrite (2.16) as follows:

$$\begin{aligned} Y'^{(k+1)} &= B^T X^{(k)} A \\ \Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= \operatorname{vec}(B^T X'^{(k)} A) \\ \Leftrightarrow \operatorname{vec}(Y'^{(k+1)}) &= (A^T \otimes B^T) \operatorname{vec}(X'^{(k)}) \end{aligned}$$

Completely analogous we can also rewrite (2.17):

$$\text{vec}(X'^{(k+1)}) = (A \otimes B)\,\text{vec}(Y'^{k}),$$

define $\mathbf{y}^{(k)} = \text{vec}(Y'^{(k+1)})$ and $\mathbf{x}^{(k)} = \text{vec}(X'^{(k+1)})$, we get:

$$\begin{aligned} \mathbf{y}^{(k+1)} &= (A^T \otimes B^T)\mathbf{x}^{(k)} \\ \mathbf{x}^{(k+1)} &= (A \otimes B)\mathbf{y}^{(k)} \end{aligned}$$

If we define $G = A^T \otimes B^T$, then with Lemma 2.2.13:

$$\begin{aligned} G^T &= (A^T \otimes B^T)^T \\ &= A \otimes B \end{aligned}$$

So we get:

$$\mathbf{y}^{(k+1)} = G\mathbf{x}^{(k)} \tag{3.10}$$

$$\mathbf{x}^{(k+1)} = G^T\mathbf{y}^{(k)}, \tag{3.11}$$

G is a $m_{\mathcal{G}}m_{\mathcal{H}} \times n_{\mathcal{G}}n_{\mathcal{H}}$-matrix, the previous expressions can be concatenated to a single matrix update equation (we define matrix M and $\mathbf{z}^{(k+1)}$):

$$\mathbf{z}^{(k+1)} = \begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^{(k+1)} = \begin{pmatrix}\mathbf{0}_{m_{\mathcal{G}}m_{\mathcal{H}}} & G^T \\ G & \mathbf{0}_{n_{\mathcal{G}}n_{\mathcal{H}}}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^{(k)} = M\mathbf{z}^{(k)},$$

M is clearly nonnegative because G and G^T are nonnegative and M is also clearly symmetric, so the result follows immediately from Theorem 2.2.10. The rest of the proof is now completely analogous to the proof of Theorem 2.3.6.

□

We define now $X_{\text{even}}(\mathbf{1})$ as the node similarity matrix and $Y_{\text{even}}(\mathbf{1})$ as the edge similarity matrix.

Note that the equations in compact form in the node-edge similarity method where defined as:

$$\begin{aligned} Y^{(k+1)} &= \frac{B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T}{\|B_S^T X^{(k)} A_S + B_T^T X^{(k)} A_T\|_F} \\ X^{(k+1)} &= \frac{B_S Y^{(k)} A_S^T + B_T Y^{(k)} A_T^T}{\|B_S Y^{(k)} A_S^T + B_T' Y^{(k)} A_T^T\|_F} \end{aligned}$$

But since $A_S = A_T = A$ and $B_S = B_T = B$ in this case, the second term of the sum is redundant and would be eliminated by the normalization in each step. This shows that the compact form as presented in the previous theorem is not different from the compact form of Theorem 2.3.6.

3.3.2 The algorithm

The algorithm is completely analogous to Algorithm 6. A Matlab implementation can be found in Listing A.16 in Appendix A. We also present Algorithm 12, an algorithm that takes an hypergraph as input and returns the incidence matrix of the hypergraph, a Matlab implementation of this algorithm can be found in Listing A.17.

Data:
A: the $n_{\mathcal{G}} \times m_{\mathcal{G}}$ incidence matrix of a hypergraph $\mathcal{G}$
B: the $n_{\mathcal{H}} \times m_{\mathcal{H}}$ incidence matrix of a hypergraph $\mathcal{G}$
TOL: tolerance for the estimation error.
Result:
X: the node similarity matrix between $\mathscr{G}$ and $\mathscr{H}$
Y: the edge similarity matrix between $\mathscr{G}$ and $\mathscr{H}$
begin `node_edge_similarity_matrix_hypergraphs`(A,B,TOL)
  $k = 1$;
  $X^{(0)} = \mathbf{1}$ ($n_{\mathscr{H}} \times n_{\mathscr{G}}$-matrix with all entries equal to 1);
  $Y^{(0)} = \mathbf{1}$ ($m_{\mathscr{H}} \times m_{\mathscr{G}}$-matrix with all entries equal to 1);
  $\mu_X = n_{\mathscr{H}} \times n_{\mathscr{G}}$-matrix with all entries equal to TOL;
  $\mu_Y = m_{\mathscr{H}} \times m_{\mathscr{G}}$-matrix with all entries equal to TOL;
  repeat
    $Y^{(k)} = \frac{B^T X^{(k-1)} A}{\|B^T X^{(k-1)} A\|_F}$;
    $X^{(k)} = \frac{B Y^{(k)} A^T}{\|B Y^{(k)} A^T\|_F}$;
    $k = k + 1$;
  until $|X^{(k)} - X^{(k-1)}| < \mu_X$ *and* $|Y^{(k)} - Y^{(k-1)}| < \mu_Y$;
end
return $X^{(k)}, Y^{(k)}$;

Algorithm 11: Algorithm for calculating the node and edge similarity matrix X and Y between $\mathcal{G}$ and $\mathcal{H}$.

Data:
n: the number of vertices of hypergraph $\mathcal{H}$
E: a set of subsets E_i of $\{1, \ldots, n\}$ that represent the edges of hypergraph $\mathcal{H}$
Result:
A: the incidence matrix of the hypergraph
begin `hypergraph_to_incidencematrix(`n, E`)`
  $m = |E|$;
  $A =$ initialize a $n \times n$-matrix with all entries equal to 0;
  for $E_j \in E$ **do**
    for $v_i \in E_j$ **do**
      $(A)_{ij} = 1$;
    end
  end
  return A;
end

Algorithm 12: Algorithm to calculate the adjacency matrix of the 2-section of a hypergraph.

3.3.3 Examples

We now calculate the same example as in the previous section:

Example 3.3.2. Let $\mathcal{H}_1$, $\mathcal{H}_2$ be the same hypergraphs as in Example 3.2.4, we get as incidence matrix A of $\mathcal{H}_1$ and B of $\mathcal{H}_2$:

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

which can be used to apply Algorithm 11 which results in the node similarity matrix X and

the edge similarity matrix Y:

$$X = \begin{pmatrix} 0.0838 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0838 & 0.1428 \\ 0.0838 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0838 & 0.1428 \\ 0.0171 & 0.0291 & 0.0027 & 0.0198 & 0.0291 & 0.0120 & 0.0171 & 0.0291 \\ 0.0108 & 0.0183 & 0.0017 & 0.0125 & 0.0183 & 0.0076 & 0.0108 & 0.0183 \\ 0.0686 & 0.1168 & 0.0109 & 0.0796 & 0.1168 & 0.0482 & 0.0686 & 0.1168 \\ 0.1417 & 0.2413 & 0.0226 & 0.1643 & 0.2413 & 0.0996 & 0.1417 & 0.2413 \\ 0.1417 & 0.2413 & 0.0226 & 0.1643 & 0.2413 & 0.0996 & 0.1417 & 0.2413 \\ 0.0750 & 0.1277 & 0.0120 & 0.0869 & 0.1277 & 0.0527 & 0.0750 & 0.1277 \\ 0.0171 & 0.0291 & 0.0027 & 0.0198 & 0.0291 & 0.0120 & 0.0171 & 0.0291 \\ 0.0838 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0838 & 0.1428 \\ 0.0838 & 0.1428 & 0.0134 & 0.0972 & 0.1428 & 0.0589 & 0.0838 & 0.1428 \\ 0.0171 & 0.0291 & 0.0027 & 0.0198 & 0.0291 & 0.0120 & 0.0171 & 0.0291 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0.0134 & 0.0840 & 0.0590 \\ 0.1045 & 0.6549 & 0.4603 \\ 0.0213 & 0.1337 & 0.0940 \\ 0.0721 & 0.4519 & 0.3177 \end{pmatrix}$$

Example 3.3.3. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.5, the node similarity matrix X and the edge similarity matrix Y become:

$$X = \begin{pmatrix} 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.1152 & 0.1961 & 0.0184 & 0.1336 & 0.1961 & 0.0810 & 0.1152 & 0.1961 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0.0000 & 0.0000 & 0.0000 \\ 0.1294 & 0.8112 & 0.5702 \\ 0.0000 & 0.0000 & 0.0000 \end{pmatrix}$$

Example 3.3.4. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.6, the node similarity matrix X and the

edge similarity matrix Y become:

$$X = \begin{pmatrix} 0.1076 & 0.1832 & 0.0172 & 0.1247 & 0.1832 & 0.0756 & 0.1076 & 0.1832 \\ 0.1076 & 0.1832 & 0.0172 & 0.1247 & 0.1832 & 0.0756 & 0.1076 & 0.1832 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0326 & 0.0555 & 0.0052 & 0.0378 & 0.0555 & 0.0229 & 0.0326 & 0.0555 \\ 0.0326 & 0.0555 & 0.0052 & 0.0378 & 0.0555 & 0.0229 & 0.0326 & 0.0555 \\ 0.1401 & 0.2386 & 0.0224 & 0.1625 & 0.2386 & 0.0985 & 0.1401 & 0.2386 \\ 0.1076 & 0.1832 & 0.0172 & 0.1247 & 0.1832 & 0.0756 & 0.1076 & 0.1832 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.1076 & 0.1832 & 0.0172 & 0.1247 & 0.1832 & 0.0756 & 0.1076 & 0.1832 \\ 0.1076 & 0.1832 & 0.0172 & 0.1247 & 0.1832 & 0.0756 & 0.1076 & 0.1832 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0.0375 & 0.2351 & 0.1652 \\ 0.1239 & 0.7764 & 0.5457 \\ 0.0000 & 0.0000 & 0.0000 \end{pmatrix}$$

Example 3.3.5. Take $\mathcal{H}_1$ and $\mathcal{H}_2$ as in Example 3.2.7, the node similarity matrix X and the edge similarity matrix Y become:

$$X = \begin{pmatrix} 0.0778 & 0.1326 & 0.0124 & 0.0903 & 0.1326 & 0.0547 & 0.0778 & 0.1326 \\ 0.1302 & 0.2216 & 0.0208 & 0.1509 & 0.2216 & 0.0915 & 0.1302 & 0.2216 \\ 0.0349 & 0.0594 & 0.0056 & 0.0404 & 0.0594 & 0.0245 & 0.0349 & 0.0594 \\ 0.1127 & 0.1919 & 0.0180 & 0.1307 & 0.1919 & 0.0792 & 0.1127 & 0.1919 \\ 0.1302 & 0.2216 & 0.0208 & 0.1509 & 0.2216 & 0.0915 & 0.1302 & 0.2216 \\ 0.0523 & 0.0891 & 0.0083 & 0.0607 & 0.0891 & 0.0368 & 0.0523 & 0.0891 \\ 0.0778 & 0.1326 & 0.0124 & 0.0903 & 0.1326 & 0.0547 & 0.0778 & 0.1326 \\ 0.1302 & 0.2216 & 0.0208 & 0.1509 & 0.2216 & 0.0915 & 0.1302 & 0.2216 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0.0233 & 0.1459 & 0.1025 \\ 0.1039 & 0.6512 & 0.4577 \\ 0.0698 & 0.4376 & 0.3076 \\ 0.0233 & 0.1459 & 0.1025 \end{pmatrix}$$

3.3.4 Concluding theorem

Before heading to an interpretation of the examples and the method, we know present a 'concluding theorem' that shows that the method of similarity using the incidence matrix is completely equal to the method with the incidence graph of the matrix. This gives us an immediate guarantee that all conditions of the introduction are met.

Theorem 3.3.6. *The similarity method using the incidence graphs of two hypergraphs $\mathcal{G}, \mathcal{H}$ returns the same results as the method with the incidence matrices of $\mathcal{G}, \mathcal{H}$, up to a constant, so:*

$$\begin{aligned} X^{(k)} &= \alpha S^{(k)}[1, \ldots, n_{\mathcal{H}}; 1, \ldots, n_{\mathcal{G}}] \\ Y^{(k)} &= \beta S^{(k)}[n_{\mathcal{H}}, \ldots, m_{\mathcal{H}}; n_{\mathcal{G}}, \ldots, m_{\mathcal{G}}] \end{aligned}$$

with X, Y, the node and similarity matrices of the method with the incidence matrices of $\mathcal{H}, \mathcal{G}$, S the similarity matrix of method of Blondel used with the incidence graphs of $\mathcal{H}, \mathcal{G}$ and $\alpha, \beta \in \mathbb{R}$ and k even. $S[R, C]$ denotes the submatrix of S with rows R and columns C.

Proof. To calculate the similarity matrix using the incidence graph representation with the method of Blondel, an element s_{ij} at iteration step k is calculated as follows (remember that A, B are symmetric):

$$s_{ij}^{(k+1)} = \frac{\sum_{f=1}^{n_\mathcal{H}+m_\mathcal{H}} \sum_{g=1}^{n_\mathcal{G}+m_\mathcal{G}} b_{if} s_{fg}^{(k)} a_{gj}}{\|BS^{(k)}A^T\|_F} \tag{3.12}$$

Now, remember that the incidence graph is a bipartite graph and thus, the adjancecy matrices A and B have the following property (remember our convention to order the columns and rows in a way that the vertices come first, followed by the edges):

$$b_{if} = 0 \quad \text{when} \quad 0 \leq i, f \leq n_\mathcal{H} \text{ or } n_\mathcal{H} \leq i, f \leq n_\mathcal{H} + m_\mathcal{H} \tag{3.13}$$

$$a_{gj} = 0 \quad \text{when} \quad 0 \leq g, j \leq n_\mathcal{G} \text{ or } n_\mathcal{G} \leq g, j \leq n_\mathcal{G} + m_\mathcal{G} \tag{3.14}$$

We are only interested in the case where s_{ij} is a calculation between nodes and between edges, as the method with the incidence matrix also only calculates this and it is rather hard to give a correct meaning to a comparision between an edge and an a node.

For the rest of the proof, we only consider the calculating of node similarity, the calculation of edge similarity is completely analogous. In this case, we have that $0 \leq i \leq n_\mathcal{H}$ and $0 \leq j \leq n_\mathcal{G}$, now (3.12) becomes with property (3.13):

$$s_{ij}^{(k+1)} = \frac{\sum_{f=n_\mathcal{H}}^{n_\mathcal{H}+m_\mathcal{H}} \sum_{g=n_\mathcal{G}}^{n_\mathcal{G}+m_\mathcal{G}} b_{if} s_{fg}^{(k)} a_{gj}}{\|BS^{(k)}A^T\|_F} \tag{3.15}$$

So in the case of calculating similarities between nodes, only the $s_{ij}^{(k)}$ of the edges play a role! Now, remember the compact form with the method using the incidence matrices (B' is the incidence matrix of $\mathcal{H}$, A' of $\mathcal{G}$, the rows in the incidence graph represent the vertices, the columns the edges):

$$X^{(k+1)} = \frac{B'Y^{(k)}A'^T}{\|B'Y^{(k)}A'^T\|_F} \tag{3.16}$$

$$Y^{(k+1)} = \frac{B'^T X^{(k)} A'}{\|B'^T X^{(k)} A'\|_F} \tag{3.17}$$

Elementwise, we write:

$$x_{ij}^{(k+1)} = \frac{\sum_{p=1}^{m_\mathcal{H}} \sum_{q=1}^{m_\mathcal{G}} b'_{ip} y_{pq}^{(k)} a'_{jq}}{\|B'^T X^{(k)} A'\|_F} \tag{3.18}$$

$$y_{cd}^{(k+1)} = \frac{\sum_{h=1}^{n_\mathcal{H}} \sum_{e=1}^{n_\mathcal{G}} b'_{hc} x_{he}^{(k)} a'_{ed}}{\|B'Y^{(k)}A'^T\|_F} \tag{3.19}$$

Now every non-zero element of the adjacency matrices A, B, can be found in the incidence matrices A', B' too (we assume that the vertices and edges have the same numbering in both matrices):

$$a_{ij} = \begin{cases} a'_{i,j-n_\mathcal{H}} & \text{if } 0 \leq i \leq n_\mathcal{H} \text{ and } n_\mathcal{H} < j \leq n_\mathcal{H} + m_\mathcal{H} \\ a'_{j,i-n_\mathcal{H}} & \text{if } n_\mathcal{H} < i \leq n_\mathcal{H} + m_\mathcal{H} \text{ and } 0 \leq j \leq n_\mathcal{H} \end{cases}$$

So we rewrite 3.15 an we get (remember that $0 \leq i \leq n_{\mathcal{H}}$ and $0 \leq j \leq n_{\mathcal{G}}$):

$$s_{ij}^{(k+1)} = \frac{\sum_{f=n_{\mathcal{H}}}^{n_{\mathcal{H}}+m_{\mathcal{H}}} \sum_{g=n_{\mathcal{G}}}^{n_{\mathcal{G}}+m_{\mathcal{G}}} b'_{i,f-n_{\mathcal{H}}} s_{fg}^{(k)} a'_{j,g-n_{\mathcal{G}}}}{\|BS^{(k)}A^T\|_F} \tag{3.20}$$

$$= \frac{\sum_{f=1}^{m_{\mathcal{H}}} \sum_{g=1}^{m_{\mathcal{G}}} b'_{i,f} s_{f+n_{\mathcal{H}},g+n_{\mathcal{G}}}^{(k)} a'_{j,g}}{\|BS^{(k)}A^T\|_F} \tag{3.21}$$

The equivalence of (3.21) and (3.18) is now clear: (3.21) depends on the edge similarity scores (to calculate the node similarity scores) and every edge similarity score is multiplied with exactly the same entries of the incidence graph. The result follows by proving that also the edge similarity scores y_{cd} are equal to the s_{ij} with $n_{\mathcal{H}} < i \leq n_{\mathcal{H}} + m_{\mathcal{H}}$ and $n_{\mathcal{G}} < j \leq n_{\mathcal{G}} + m_{\mathcal{G}}$, this is possible with the exact same reasoning. The difference up to a constant is caused by the normalization in each iteration. We conclude that the theorem is true for each even k. □

Corollary 3.3.7. *The node-edge similarity method (see 2.3) returns the same results for undirected graphs $\mathscr{G}, \mathscr{H}$ as the node similarity method of Blondel (see 2.1) using incidence graphs of $\mathscr{G}, \mathscr{H}$, up to constant.*

Proof. This follows immediately from the fact that in the node-edge similarity method for undirected graphs, the source-edge matrix and terminal-edge matrix are the same and are both equal to the incidence matrix for undirected graphs (see Definition 1.5.27). □

Remark 3.3.8. *One may wonder whether the node-edge similarity method of 2.3 also returns the same results for directed graphs $\mathscr{G}, \mathscr{H}$ as the node similarity method of Blondel using the incidence graphs of $\mathscr{G}, \mathscr{H}$. This is the case, but is a little harder to prove. First, the node-edge similarity method stays the same, it's impossible to somehow replace the soure-edge or terminal-edge matrices by the incidence matrix as it will contain negative entries (see Definition 1.5.27). The incidence graph of a directed graph is defined as:*

Definition 3.3.9. *Let $\mathscr{G} = (V, \rightarrow)$ be a directed graph, then the **incidence graph** $\mathscr{G}'_i = (V', \rightarrow')$ of $\mathscr{G}$ is the directed graph with:*

1. $V' = V \cup \rightarrow$,
2. $\forall v_i \in V, \forall e_j \in \rightarrow$: $v_i \rightarrow e_j$ *if* v_i *is the source node of* e_j,
3. $\forall v_i \in V, \forall e_j \in \rightarrow$: $e_j \rightarrow v_i$ *if* v_i *is the terminal node of* e_j.

With this setting, it is possible to adapt our proof to see that the method of Blondel with two incidence graphs of directed graphs is equal the the node-edge similarity method. The proof will have exactly the same reasoning, but will be a little bit more extensive as you have to consider both the equalities with the source-edge matrix and the terminal edge matrix.

3.3.5 Interpretation

By the concluding theorem, we know that the similarity method for hypergraphs using the incidence matrix is the same (up to constant) as the method with the incidence graph representation of hypergraphs. Since the latter method meets all the conditions, this method satisifies all the requirements too.

Appendix A

Listings

Listing A.1: The MatLab code for the Power Method described in algorithm 1

```
function [y, mu] = power_metho(A, y, TOL)
k = 1;
mu = 0;
while true
    z = A*y;
    mu_previous = mu;
    mu = norm(z,2);
    y_previous = y;
    y = z/mu;
    k = k + 1;
    if and(k>2,abs(mu - mu_previous)<TOL)
        break;
    end
end
if (y(1) == -y_previous(1))
    mu = -mu;
end
return;
end
```

Listing A.2: The MatLab code for algorithm 5.

```
function [Z] = similarity_matrix(A,B, TOL)
  Z_0 = ones(size(B,1),size(A,1));
  mu(1:size(B,1),1:size(A,1)) = TOL;
  Z = Z_0;
  Z_previouseven = Z_0;
  k=1;
  while true;
      Y = norm((B*Z*transpose(A)+transpose(B)*Z*A),'fro');
      X = B*Z*transpose(A)+transpose(B)*Z*A;
      Z = X/Y;
      if mod(k,2) == 0
        difference = abs(Z-Z_previouseven);
        disp(difference);
        Z_previouseven = Z;
```

```
        if (difference < mu)
            break;
        end
      end
      k = k + 1;
  end
  return;
end
```

Listing A.3: The MatLab code for Algorithm 6.

```
function [X, Y] = node_edge_similarity_matrices(AS, AT, BS, BT, TOL)
    X = ones(size(BS,1),size(AS,1));
    Y = ones(size(BS,2),size(AS,2));
    mu_X(1:size(BS,1),1:size(AS,1)) = TOL;
    mu_Y(1:size(BS,2),1:size(AS,2)) = TOL;
    X_previouseven = X;
    Y_previouseven = Y;
    k=1;
  while true;
      Y = (transpose(BS)*X*AS+transpose(BT)*X*AT)
            /norm(transpose(BS)*X*AS+transpose(BT)*X*AT, 'fro');
      X = (BS*Y*transpose(AS)+BT*Y*transpose(AT))
            /norm(BS*Y*transpose(AS)+BT*Y*transpose(AT), 'fro');

      difference_X = abs(X-X_previous);
      difference_Y = abs(Y-Y_previous);
      X_previous = X;
      Y_previous = Y;
        if difference_X < mu_X
            if difference_Y < mu_Y
                break;
            end
        end
  end
  return;
end
```

Listing A.4: The MatLab code to converse an adjacency matrix to a source-edge matrix (edges are numbered left-to-right based on the adjacency matrix).

```
function [AS] = source_edge_matrix(A)
  number_of_edges = sum(A(:));
  AS = zeros(size(A,1), number_of_edges); %initialize the souce-edge matrix A_S
  current_edge = 1;
  for i=1:size(A,1)
      for j=1:size(A,2)
          if A(i,j) > 0
              for e=1:A(i,j)
                  AS(i,current_edge) = 1;
                  current_edge = current_edge + 1;
              end
          end
      end
  end
```

```
end
```

Listing A.5: The MatLab code to converse an adjacency matrix to a terminal-edge matrix (edges are numbered left-to-right based on the adjacency matrix).

```
function [AS] = terminal_edge_matrix(A)
  number_of_edges = sum(A(:));
  AS = zeros(size(A,1), number_of_edges); %initialize the terminal-edge matrix A_T
  current_edge = 1;
  for i=1:size(A,1)
      for j=1:size(A,2)
          if A(i,j) > 0
              for e=1:A(i,j)
                  AS(j,current_edge) = 1;
                  current_edge = current_edge + 1;
              end
          end
      end
  end
end
```

Listing A.6: The MatLab code for Algorithm 6 but with adjacency matrices as input (edges are numbered left-to-right based on the adjacency matrices).

```
function [X,Y] = node_edge_similarity_matrices_with_adjacency_matrix(A, B, TOL)
  AS = source_edge_matrix(A);
  AT = terminal_edge_matrix(A);
  BS = source_edge_matrix(B);
  BT = terminal_edge_matrix(B);
  [X,Y] = node_edge_similarity_matrix(AS, AT, BS, BT, TOL);
end
```

Listing A.7: The MatLab code for Algorithm 8 that takes an ordered list with the number of vertices of the same color and a normal adjacency matrix as input and returns a partitioned adjacency matrix.

```
function [Z] = colored_node_adjacency_matrix_partitioning(colored_vertices,A)
 number_of_colors = size(colored_vertices,2);

 for i = 1:number_of_colors
     for j = 1:number_of_colors
         c_i = colored_vertices(i);
         c_j = colored_vertices(j);
         B = zeros(c_i,c_j);
         start_vertex_i = 0;
         start_vertex_j = 0;
         for k = 1:i-1
             start_vertex_i = start_vertex_i + colored_vertices(k);
         end
         for k = 1:j-1
             start_vertex_j = start_vertex_j + colored_vertices(k);
         end
         for r = 1:c_i
```

```
            for k = 1:c_j
                B(r,k) = A(start_vertex_i + r, start_vertex_j + k);
            end
        end
        Z{i,j} = B;
    end
 end
 return;
end
```

Listing A.8: The MatLab code for Algorithm 9 that calculates the node similarity matrix for colored nodes.

```
function [Z] = colored_node_similarity_matrix(colored_verticesA, A,
    colored_verticesB, B, TOL)
 partitionedA = colored_node_adjacency_matrix_partitioning(colored_verticesA, A);
 partitionedB = colored_node_adjacency_matrix_partitioning(colored_verticesB, B);
 number_of_colors = size(colored_verticesA, 2);
 k = 1;
 for i=1:number_of_colors
     Z{i} = ones(colored_verticesB(i), colored_verticesA(i));
 end
 Z_previouseven = Z;
 while true;
     norm = 0;
     for i=1:number_of_colors
         Z_temp = 0;
         for l = 1:number_of_colors
             Z_temp = Z_temp + partitionedB{i,l}*Z{l}*transpose(partitionedA{i,l
                 })+ transpose(partitionedB{l,i})*Z{l}*partitionedA{l,i};
         end
         Z{i} = Z_temp;
         norm = norm + trace(transpose(Z{i})*(Z{i}));
     end

     norm = norm(Z, 'fro');

     for i=1:number_of_colors
         Z{i} = Z{i}/norm;
     end
     if mod(k,2) == 0
        have_to_stop = 1;
        for i = 1:number_of_colors
            difference_i = abs(Z{i}-Z_previouseven{i});
            if not(all(difference_i) < TOL)
                have_to_stop = 0;
            end
        end
        if(have_to_stop == 1)
            break;
        else
            Z_previouseven = Z;
        end
      end
      k = k + 1;
 end
```

```
end
```

Listing A.9: The MatLab code for the algorithm that takes an ordered list with the number of edges of the same color and a source-edge matrix or terminal-edge matrix as input and returns a partitioned source-edge or terminal-edge matrix.

```
function [Z] = colored_edge_matrix_partitioning(colored_edges,A)
 number_of_colors = size(colored_edges,2);
 number_of_nodes = size(A,1);
 for i = 1:number_of_colors
     c_i = colored_edges(i);
     B = zeros(number_of_nodes,c_i);
     start_vertex_i = 0;
     for k = 1:i-1
         start_vertex_i = start_vertex_i + colored_edges(k);
     end

     for r = 1:number_of_nodes
         for k = 1:c_i
             B(r,k) = A(r, start_vertex_i + k);
         end
     end
     Z{i} = B;
 end
 return;
end
```

Listing A.10: The MatLab code for the algorithm that calculates the node and (colored) edge similarity matrix for colored edges.

```
function [S] = colored_edge_similarity_matrix(colored_edgesA, AS,AT, colored_edgesB, BS,BT, TOL)
 partitionedAS = colored_edge_matrix_partitioning(colored_edgesA,AS);
 partitionedAT = colored_edge_matrix_partitioning(colored_edgesA,AT);
 partitionedBS = colored_edge_matrix_partitioning(colored_edgesB,BS);
 partitionedBT = colored_edge_matrix_partitioning(colored_edgesB,BT);
 number_of_colors = size(colored_edgesA, 2);
 k = 1;
 for i=1:number_of_colors
     Z{i,i} = ones(colored_edgesB(i), colored_edgesA(i));
 end
 Z_previouseven = Z;
 while true;
     norm2 = 0;
     for i=1:number_of_colors
         Z_temp = 0;
         for l = 1:number_of_colors
             Z_temp = Z_temp + partitionedBS{l}*Z{l,l}*transpose(partitionedAS{l})+ partitionedB
         end
         Z{i,i} = Z_temp;
         norm2 = norm2 + trace(transpose(Z{i,i})*(Z{i,i}));
         Z{i,i} = Z{i,i}/norm(Z_temp,'fro');
     end
```

```
    if mod(k,2) == 0
       have_to_stop = 1;
       for i = 1:number_of_colors
           difference_i = abs(Z{i,i}-Z_previouseven{i,i});
           if not(all(difference_i) < TOL)
               have_to_stop = 0;
           end
       end
       if(have_to_stop == 1)
           break;
       else
           Z_previouseven = Z;
       end
     end
     k = k + 1;
 end
  disp(Z);
  for i = 1:number_of_colors
  for j = 1:number_of_colors
      if not(i == j)
         Z{i,j} = zeros(size(Z{i,i},1),size(Z{j,j},2));
      end
  end
  end

 S = cell2mat(Z);
 return;
end
```

Listing A.11: The MatLab code for to calculate the adjacency matrix of the representing line-graph of a hypergraph.

```
function [A] = hypergraph_to_linegraph(n,E)
    m = numel(E);
    A = zeros(m,m);
  for i=1:m
      for j=1:m
          if or(i==j,isempty(intersect(E{i},E{j})))
              A(i,j) = 0;
          else
              A(i,j) = 1;
          end
      end
  end
end
```

Listing A.12: The MatLab code for to calculate the adjacency matrix of the 2-section of a hypergraph.

```
function [A] = hypergraph_to_2section(n,E)
 A = zeros(n,n);
  for i=1:n
      for j=1:n
          for idx = 1:numel(E)
```

```
            if i==j
                A(i,j) = 0;
                break;
            elseif not(or(isempty(intersect(E{idx}, [i])),isempty(intersect(E{idx}, [j]))))
                A(i,j) = 1
                break;
            else
                A(i,j) = 0;
            end
        end
    end
  end
end
```

Listing A.13: The MatLab code for Algorithm to calculate the adjacency matrix of the extended 2-section of a hypergraph.

```
function [A] = hypergraph_to_extended2section(n,E)
 A = zeros(n,n);
  for k = 1:numel(E)
      if size(E{k},2) == 1
          A(cell2mat(E(k)), cell2mat(E(k))) = A(cell2mat(E(k)), cell2mat(E(k))) + 1;
      else
          p = perms(E{k});
          already_done = [];
          for i = 1:size(p,1)
              if isempty(intersect([p(i,1)], already_done))
                  for j = 2:size(p,2)
                      A(p(i,1),p(i,j)) = A(p(i,1),p(i,j)) + 1;
                  end
                  already_done = [already_done, p(i,1)];
              end
          end
            disp(A);
      end
  end
  return;
end
```

Listing A.14: The MatLab code for Algorithm to calculate the adjacency matrix of the incidence graph of a hypergraph.

```
function [A] = hypergraph_to_incidencegraph(n,E)
 m = numel(E);
 A = zeros(n+m,n+m);
    for idx = 1:numel(E)
        edge = cell2mat(E(idx));
        for idx_2 = 1:numel(edge)
            A(edge(idx_2), n + idx) = 1;
            A(n + idx,edge(idx_2)) = 1;

        end

    end
end
```

Listing A.15: The MatLab code for Algorithm 12 to calculate the incidence matrix of a hypergraph.

```
function [A] = hypergraph_to_incidencematrix(n,E)
 m = numel(E);
 A = zeros(n,m);
    for idx = 1:numel(E)
        edge = cell2mat(E(idx));
        for idx_2 = 1:numel(edge)
            A(edge(idx_2), idx) =1
        end

    end
end
```

Listing A.16: The MatLab code for Algorithm 11 to calculate the node-edge similarity scores of a hypergraph.

```
function [A] = hypergraph_to_2section(n,E)
 A = zeros(n,n);
  for i=1:n
      for j=1:n
          for idx = 1:numel(E)
              if i==j
                  A(i,j) = 0;
                  break;
              elseif not(or(isempty(intersect(E{idx}, [i])),isempty(intersect(E{idx}, [j]))))
                  A(i,j) = 1
                  break;
              else
                  A(i,j) = 0;
              end
          end
      end
  end
end
```

Appendix B

Results of the Eurovision Song Contest 2009-2015

Table B.1: Eurovision Song Contest 2009

	Albania	Andorra	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Czech Republic	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Germany	Greece	Hungary	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Norway	7	10	8	8	12	10	10	2	8	10	3	12	12	8	8	8	12	10	12	12	8	12	12	12	8	8	10		12	5	5	12	10	10	12	12	12	8	12	3	12	10	387	1	0.311382108	0.107545693
Iceland	6	8	5		2			5	2	5	2	10	8	2	10		7	4	7		12	10	8	8	12	3	5	12	1	8	10	3		6	5		10	5	7	2		8	218	2	0.175683413	0.113493313
Azerbaijan	4		1		10	3	3	8	10	8	10	8	7	3	2	1		8	10			6	4	5	1	10	6	10	6		4	7	4	4	1		8		10	12	10	3	207	3	0.165946697	0.109357170
Turkey	10		4	12		12	7	10	1		1	6		12	5	12	10	3	5			3			5		3	7		3	6					2	6	12	8			12	177	4	0.136634156	0.094714828
United Kingdom	8	4	7		3		4	7	4	7	6	3		6		4	8	12	1		10	4	2	3	10	1		2	4	10		6	8	7	3	10			3		6		173	5	0.137107599	0.133630498
Estonia				4	4				6	3		5			12		1	5	6	10	6		10	10		7			8		1	8		12			7				4		129	6	0.106793281	0.131401482
Greece	12		10		5	5	5	12	7	12							6		4						7		2				8	4	6		4	1	2	2	1			5	120	7	0.096460793	0.148053884
France	2	3	6		7	1	6					2			4		3	6		6	3	5	5	6			1	1			10	3		7	3		7	6		3	1	107	8	0.087462271	0.109033937

Table B.1: Eurovision Song Contest 2009 (continued)

	Albania	Andorra	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Czech Republic	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Germany	Greece	Hungary	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Bosnia & Herzegovina	5			2					4					10	6		2			2							12						12	8	10									106	9	0.081427395	0.119158387
Armenia					1	7		6		4	12			1	1	6				5		8								2	4	7	5			3		4	3		5	6	2	92	10	0.069134382	0.132217298
Russia			12	6	8					1	8		10			5						7	6	7		6			3			5								4	8		91	11	0.072122471	0.152992047
Ukraine			3	10	6						5									8		2		4	2	4		5	10	6	2	1	6								2	76	12	0.058251749	0.142826304
Denmark		5								6						5				3	4	4	3	2	6	5		8	7	2	1	8								5		74	13	0.056476607	0.156365431
Moldova			2	7		4			3														1					3	5	12	12	1			5				7	7		69	14	0.048643191	0.119160533
Portugal		6				6	1				7		3		7					7														2	8	10						57	15	0.039280535	0.105898749
Israel		7				8	8				4		5		10												4						5			1				1		53	16	0.037533919	0.157031429
Albania									5				1	7				7	2								7									2		1	6		10		48	17	0.035463621	0.141493351
Croatia			5				12						2	4				2		1							2	8					5	6								45	18	0.032830769	0.134539946
Romania			3							2				5							2						12			2			2		7					5		40	19	0.027940824	0.119466004
Germany	3					2	2					7				3						1						6	1										2	1	7	35	20	0.027028531	0.155134199
Sweden	1	2										4	6		7	2				3					4		4														33	21	0.025173339	0.160328666
Malta		1													3	4					5	1	1							3	7									6	31	22	0.024200005	0.139682295
Lithuania				1				1				1	2								7	7																			4	23	23	0.018290261	0.152600566
Spain		12																1											7						3						23	24	0.015156376	0.122739159
Finland								3					4					8						3												4						22	25	0.016946597	0.132773031
Andorra																																										0	0.109781594	
Belarus																																										0	0.149587578	
Belgium																																										0	0.122233834	
Bulgaria																																										0	0.122793535	
Cyprus																																										0	0.146566820	
Czech Republic																																										0	0.103917420	
FYR Macedonia																																										0	0.124227588	

Table B.1: Eurovision Song Contest 2009 (continued)

	Albania	Andorra	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Czech Republic	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Germany	Greece	Hungary	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Hungary																																															0	0.158260146
Ireland																																															0	0.128126358
Latvia																																															0	0.143787849
Montenegro																																															0	0.126144148
Poland																																															0	0.135972955
Serbia																																															0	0.123080364
Slovakia																																															0	0.142349752
Slovenia																																															0	0.133178441
Switzerland																																															0	0.117733465
The Netherlands																																															0	0.163394917

Table B.2: Eurovision Song Contest 2010

	Albania	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Germany	10		1		10	8	3	6	4	12	12	8	12	3			2	3	8		12	10	4		12	7	1	3	6	8	12	10	12	12	12	4	10	5	4	246	1	0.263781309	0.115909290
Turkey	8		12	3	6	10	10	12		6	6	10	3	12	5	10			1			4			2		8		3			2	3	5	3	8		10	10	170	2	0.169318656	0.137704079
Romania	5	1	7	1		2			8	8						4	5	7	8	3	2	5	12	10	6	10		3	6	1	7	10	10	4	5	2	2	8	162	3	0.168898361	0.128083998

Table B.2: Eurovision Song Contest 2010 (continued)

	Albania	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Denmark	2	5	4		2			2										12		4	10	3	8		8	12			1			7	12	4	8	2				6	149	4	0.158250811	0.144174170
Azerbaijan				6	5	7	12		10	2		3			1	4	3	7	2			12		7	8			8			2		7		2	12	12		145	5	0.150037583	0.120818451
Belgium										10	3		6	7	12	6	10	10		4	7	10		3	10	5	4	5	5	10				2	7	6		1		143	6	0.147556763	0.147362420
Armenia			5	7	8		7					4		6	10	7	7		12	1				6				6	12	1	4		8	1		12	6	6		141	7	0.139231319	0.101508660
Greece	12	3		12	6	5	12			12		7	4		8		8					7	2		1	8	7		10	5		5			3	3		12	140	8	0.143954029	0.095629350
Georgia		12	8	7	4	4	6	7	5		8	5			5	2	5	5			12		5	1	4			10			1	1	6	1		5	7		136	9	0.135392862	0.086281121
Ukraine		8	10	10			7		6					2	7			2	7	6		8		3		5	7	7						7	1		5	108	10	0.103750143	0.139277675
Russia		10	3	12							10						10	8	5		10				2						6					4	10		90	11	0.085893582	0.141861113
France			6				3				10				3	8	6	6	3			2		4	7			4			3	2							2	82	12	0.080095341	0.118193639
Serbia	3				12		8	1			7		10			5															8			7	10	1				72	13	0.071530583	0.132454774
Israel	4		5	8	1		2				10	1	4											1	5						8	6				10		3	3	71	14	0.071051244	0.118969983
Spain	7	7		1		1	2				4	2					1	5	8	4		12	2	4		5											4		68	15	0.065904340	0.162626170
Albania				3	5			5			12					1	10	7					2		1									8		7		1	62	16	0.059724471	0.140537652
Bosnia & Herzegovina	6						10				6	5																12				4			8			51	17	0.050145707	0.139053777
Portugal		4								4		8				6	6	1		6	1							2				6	5					43	18	0.042922001	0.107674053
Iceland				2	8					3	4	5				4	3						6	6														41	19	0.042236126	0.137492185
Norway		2								5	7			6			2						3			3					3		4					35	20	0.034123011	0.155454891
Cyprus							4	4	1						12						1										2		3					27	21	0.024305724	0.144174824
Moldova		6	4									1				2			6									6	10										27	22	0.024039196	0.119696370
Ireland								1		2		1		2			6																6					7	25	23	0.024239863	0.147411097
Belarus							1				2				12										3					2										18	24	0.013826739	0.104107756
United Kingdom	1		2													3				4																			10	25	0.009378369	0.136171561
Bulgaria																																									0	0	0.136208081	
Croatia																																									0	0	0.114014418	
Estonia																																									0	0	0.136097782	
FYR Macedonia																																									0	0	0.127626178	
Finland																																									0	0	0.133330433	

Table B.2: Eurovision Song Contest 2010 (continued)

	Albania	Armenia	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Iceland	Ireland	Israel	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Romania	Russia	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Latvia																																										0	0.142837341
Lithuania																																										0	0.144633755
Malta																																										0	0.138072738
Poland																																										0	0.159433083
Slovakia																																										0	0.145328357
Slovenia																																										0	0.137277314
Sweden																																										0	0.153858352
Switzerland																																										0	0.127076574
The Netherlands																																										0	0.139085180

Table B.3: Eurovision Song Contest 2011

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Romania	Russia	San Marino	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Azerbaijan	8		8		6	3	8		10	8		8		5	6	8		5	7	8		4		2	8	12	10		8	8	10	12	10		7			3	1	6	12	10		221	1	0.239456546	0.119172772
Italy	12	6	6	1	3	6	6			1		6	1	3	8	7	3	10	4	3	5			12	10	10			10	10	6		12	2		3	12		4				7	189	2	0.196060135	0.098845242
Sweden		4		3	4	4	5		4	10	10	12	6	6	10	1		6	10	7	4	12				6	3	10			3	1	6	1	10	4	5			10	4	1	3	185	3	0.191434165	0.106864013
Ukraine	3	12		12	10			8	5	5			7			10		7				7	7			4	8		2	7	2	10		6	12	6		2			7			159	4	0.164438109	0.120992086
Denmark					1			7		3		10			7		6			12	12	10		6		5		7	3				4		6	8		10		12			5	134	5	0.133906705	0.104782124

Table B.3: Eurovision Song Contest 2011 (continued)

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Romania	Russia	San Marino	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Bosnia & Herzegovina	7		12					2	7				12		5			7	3					4					4					12			12								125	6	0.118392353	0.127186019
Greece	10	7		8	8	8		10		12			3			6	10		8				2						1			8	8	3					2			6	120	7	0.122325490	0.145912383
Ireland				4		7	2	3			12		10				8				4			10		8			1	6				8		7	12		5		8	12	119	8	0.112343966	0.103703681
Georgia	10			10	12			1					3				8	5				2		12		7			7			6	7									110	9	0.117252558	0.132743840
Germany		10			8	5	3	1		8					3			4		6			6	8	2			5	4						7	3	6	8	7	3				107	10	0.103211617	0.103941997
United Kingdom	6	2		5	2			12		4	3		5		1		2		2	6	1	10	5	3	7	4	1		3			4	2			1				6	3			100	11	0.100671386	0.096916917
Moldova		5	7	7	7	1									4	5	4					8	8	4						5	12	7			5						7	8	97	12	0.095132589	0.126124831
Slovenia		3					12		12	6	7	2	10				1		6			2	3		3					1	4	5	1	10	1				2	2			96	13	0.096457469	0.118682326
Serbia	1	1	7				10		8				8	2					3		5			5						6	6			5		10	4		6	3			85	14	0.082541126	0.095024968
France	2	5				12	4		6	7				4		2		12				1	1		1	2				2	3	3			10								82	15	0.085010939	0.137268491
Russia	4	8		4	5			4		2		5				4	5	1	3	1		8		6		5						4											77	16	0.077844320	0.143360223
Romania			1	6		10		6			4	1						1		1	6	12				12									8				4	5			77	17	0.073558844	0.131674852
Austria	3						7	5	3		1		2	1		3	12		2						2									3	5								64	18	0.059654291	0.129428733
Lithuania									2							12						10		7			3	12		1	2	7			1								63	19	0.058773021	0.127424625
Iceland							1				6		8				12						5				8	4								2	1	10						61	20	0.052757629	0.130358202
Finland											5	7					2		10	3				1			12	5									7	5					57	21	0.052100575	0.101891325
Hungary		2	2											12	2				5			7											8		6	5		4					53	22	0.048236982	0.130649629
Spain	5											4	4	12														12	5									3		1		50	23	0.052115936	0.114254584
Estonia						2					2			7								7	5	4	7	6	2									2							44	24	0.040929137	0.143253587
Switzerland																																		5	4							10	19	25	0.016064278	0.092620742
Albania																																													0	0.142107551	
Armenia																																													0	0.128531290	
Belarus																																													0	0.138933798	
Belgium																																													0	0.118486508	
Bulgaria																																													0	0.110776143	

Table B.3: Eurovision Song Contest 2011 (continued)

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Norway	Poland	Portugal	Russia	San Marino	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Croatia																																													0	0.129980068
Cyprus																																													0	0.145690798
FYR Macedonia																																													0	0.116197406
Israel																																													0	0.131676797
Lativa																																													0	0.121328888
Malta																																													0	0.162191624
Norway																																													0	0.099011716
Poland																																													0	0.127824729
Portugal																																													0	0.130723491
San Marino																																													0	0.158559128
Slovakia																																													0	0.141130328
The Netherlands																																													0	0.137636506
Turkey																																													0	0.145607545

Table B.4: Eurovision Song Contest 2012

	Albania	Austria	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Portugal	Romania	Russia	San Marino	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Sweden	5	12	7	6	12	8	8	7	10	12	12	6	12	12	8	12	6	12	12	8	12	10	12	10	6	7	4	12	10	12	12	10	12	12	10	12	12	7	12	6	12	372	1	0.305639378	0.099063059
Russia	3	5	10	12	8	3	6	6	5	8	4	4	8	4	5	7	4	7	6	12	8	10	7	10	6	3	6	4	8	8	4	12	7	3	8	8	8	7	4	7	10	259	2	0.206572384	0.129906575
Serbia	1	10			5	10	12	12	7			10	2	8		10	8	4	3		6		5	5	3	12	10	5	5	4	6		7	12			10	10	10		2		214	3	0.176946532	0.113907146
Azerbaijan	4		7		7	7	10		8			2			10									12	12	2	150	4	0.117051052	0.113395687
Albania		8		10	6	4	5	4	5			12	6		3	6	10	6	4		12	1		12	10	5		1	12	1		3		1	12		5			146	5	0.118949905	0.098184652
Estonia			4									10	10			4				10	8		8	8		2		7	7		6			10		6	8		7	1	4	120	6	0.097934280	0.131495479
Turkey	10	3	12		7	4	7			2		8		5	7	8		3			1		1	8					3		5				6	3	8		1	112	7	0.088872329	0.113535313
Germany	2	4						4		10	10		1	7	2		10		10		8	7	3	2			3	10							2	3		4	2		6	110	8	0.089024803	0.157167197
Italy	7							2	2	3	7	5		1	4	2	3	5		10	2	4		4	10	5	2	4	2					7	5	5	1		4			101	9	0.080314920	0.109725154
Spain		6			6	5	3		6		4		3	6			1	2		10				5	4				12	6	1						4	8	6			8	97	10	0.078561620	0.133173657
Moldova	1		5				1		7								2	2									3		6	12	7	8		1	7	2			8			81	11	0.062811881	0.119439246
Romania		6			3				3	1				2			7				5	6	7		12		4	1					10						4			71	12	0.054187099	0.119828631
FYR Macedonia		8			2		12	2	8														1				8					12	1	6						8	3		71	13	0.053773867	0.132611623
Lithuania			4	8			5					3			3	12					7		4		4		1	6		5									1			7	70	14	0.055584045	0.14412079
Ukraine			3	10								1	3		6		1				3		3	6	2	7	8	1	1	8				2								65	15	0.049463410	0.123998028
Cyprus	6		2														12		8		3	5							2	2	8		5	12								65	16	0.045779111	0.140384735
Greece	12		5		2	1	1	1	12						1	1					2						4			8	3	4								3	5	64	17	0.047316840	0.124054195
Bosnia & Herzegovina		7						10				7															6					5	2	7		1	10					55	18	0.044119285	0.135191508
Ireland			1	1	4			3		4			4							4			5												5	5		10				46	19	0.036462417	0.132863608
Iceland									1	6	6		7			3		6										5						4	4	4							46	20	0.039862230	0.129060084
Malta		8	3									1													7						2	6							2	7	5	41	21	0.030603570	0.122803553
France		2				2															6			3													2	6				21	22	0.016426246	0.151434886
Denmark												2	5			5		5					2					2														21	23	0.017720863	0.139703998
Hungary																											1		7		2	8								1		19	24	0.014553209	0.151290741
United Kingdom					1							5									4		2																				12	25	0.009791307	0.121921955

Table B.4: Eurovision Song Contest 2012 (continued)

	Albania	Austria	Azerbaijan	Belarus	Belgium	Bosnia & Herzegovina	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Portugal	Russia	San Marino	Serbia	Slovakia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Turkey	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Norway																			1																		3					7	26	0.005255627	0.154850199
Austria																																											0	0.153796281
Belarus																																											0	0.120913553
Belgium																																											0	0.153461988
Bulgaria																																											0	0.150772307
Croatia																																											0	0.132022928
Finland																																											0	0.141472175
Georgia																																											0	0.123066528
Israel																																											0	0.139948084
Latvia																																											0	0.138435935
Montenegro																																											0	0.137300587
Portugal																																											0	0.116835694
San Marino																																											0	0.130840589
Slovakia																																											0	0.138398727
Slovenia																																											0	0.148664554
Switzerland																																											0	0.125128002
The Netherlands																																											0	0.147638419

Table B.5: Eurovision Song Contest 2013

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Romania	Russia	San Marino	Serbia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Denmark	1	4	5	5	1	10	2	10	7		8	12	7	12	7	10	7	10	12	12	8	12	6	2	6	6	10	7	6	4		12	12	8	10	3	10	5	12	281	1	0.251738561	0.112496381
Azerbaijan	7		12		10	5	12	7	8				8	12	4	12	7	4	12	7	2	12	12	3	12	8	12	10	12	10	12	2	5	3	7		6	2	10	234	2	0.211075596	0.109552608
Ukraine		12	1	12	12	8	10	12	10	3	10				8		6		1		10	5	7	10	10	12		1	4	1		10		10			5		5	214	3	0.196324466	0.106438081
Norway	2	3		2	3	7	1	3	4	12	3	8	12		4	7	4	2	10		6	8	8	6	3	2	4		8	7	7	5	12	7	6	3		191	4	0.164027896	0.098933496
Russia		7			8	4	5	6	5	7	12	6	2	6	6	2			1	10	7			12	7	7							8	10	6	5		4	4	10	174	5	0.157129979	0.130276520
Greece	10	8	7	4	6	2	7	5	12	6		4	1		6			1		2	7	1	4		8	5	7	10	12							8	1		8	152	6	0.126771445	0.144942656
Italy	12	1	10			6		8	6			10		10	2		6					3	10		8	6	1	4	4	8	12		12						126	7	0.111480415	0.134388723
Malta		6		8	7			3	4	5	3			2	3	5	3	8	5		10	5			10	5		10			1				8	2	7	120	8	0.098929720	0.146482452
The Netherlands	4		8		12		6	2	10	7	2	8					5	8	6			4			2	3		8	5	3		8	4		7	12		6	114	9	0.095968817	0.133925774
Hungary	8						6	4		4		10			12	2			3	5					6	2			2			6	2		3	10	7		84	10	0.066641891	0.150320451
Moldova			2	1	4	3	3			7			4							3	1	4				5		12	6		6		2				8		71	11	0.062911597	0.139519096
Belgium			3								5	2		3	5			4		4		4		2				3		8	5	3	2		7		12		3	71	12	0.058233658	0.159189216
Romania	5		4	6										1			10		6		1		7	10	1	6									4				4	65	13	0.054431203	0.141742263
Sweden					1	4	3			8	1		4			3			4	5			4		5	12					1	6				3			62	14	0.050114034	0.122816263
Georgia		10		10											5									8	3	2			5								7			50	15	0.040436276	0.147386391
Belarus		5		7								5			5		1				3			1	2	4	3										12			48	16	0.039440804	0.143351285
Iceland										1	6	5				8		6									4					4	6	5		2			47	17	0.038200761	0.149905594
Armenia					2		8							7	10			3								1	1		2	1							6			41	18	0.035257962	0.125792175
United Kingdom																7						5				5		3				1	4	1	2				23	19	0.020645649	0.143614616
Estonia												6											10	3															19	20	0.015433229	0.141096214
Germany	3		6															5															3	1				18	21	0.015543936	0.126649981
Lithuania				3	5									1					1		6														1			17	22	0.014312777	0.141552096
France		2							1			1			2															8									14	23	0.010153847	0.135186986
Finland										2				3	1			4												3									13	24	0.010627535	0.108008031
Spain	6																				2																		8	25	0.005630226	0.153609206
Ireland									2																							2						1	5	26	0.004543069	0.137789556
Albania																																										0	0.102282457	
Austria																																										0	0.127058480	
Bulgaria																																										0	0.136341600	

Table B.5: Eurovision Song Contest 2013 (continued)

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Bulgaria	Croatia	Cyprus	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Romania	Russia	San Marino	Serbia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Croatia																																											0	0.169310202
Cyprus																																												0	0.161412928
FYR Macedonia																																												0	0.138521061
Israel																																												0	0.157743866
Latvia																																												0	0.134627381
Montenegro																																												0	0.156164697
San Marino																																												0	0.092520925
Serbia																																												0	0.165969147
Slovenia																																												0	0.141375070
Switzerland																																												0	0.117585074

Table B.6: Eurovision Song Contest 2014

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	San Marino	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Austria	5			1		12	8	4	3	12	10	10	7	12	10	10	12	12	6	10	10	7	6	3	2	10		12	8	5		12	12	12	12	8	12	290	1	0.285110029	0.137956312
The Netherlands		4	10		2	8	10	12	7	8	8		12	8	12	12	10		4	12	12				12	12	10	3	3	2	10	7	10	10		8	238	2	0.235720093	0.145211066
Sweden	7		6			10	12	10		10	4	2		2	8	7	12	10	8	7	7	6	3	10		8	4	8	12	6	3	10	8	10		6	8	12	7	218	3	0.212949392	0.155725338
Armenia			12		10		2	5	8	4	12	12	6	7	7	2		10		6		1	4	7	8	6		4	5		7	10		174	4	0.156091783	0.050899422
Hungary	8		7	8	5	7	3	7	10	5				6		6	1	7		1		4	12			6	10	6	7		2	7	1	4	3		143	5	0.124453384	0.16569976
Ukraine			5	10	8	4	1	8		2		6		5	2			5	10	7	5		10	7		5		7		6						113	6	0.095410297	0.135769452
Russia		10		12	12			1	6			8	10					3		2	6	5	8			2								4		89	7	0.086504385	0.10214933

Table B.6: Eurovision Song Contest 2014 (continued)

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	San Marino	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Norway			1		4		6	3		7	2		5	3		1	7				5	8	2				7	3	1			5		3	5	10			88	8	0.074474911	0.156676045
Denmark		1				6				6	3	8			8						1				1	6	5	4		1	6	3	8	3	1		3	74	9	0.070675128	0.160549539
Spain	12	2				2		2			6		1		6	4										5	2	5			4			8		2	5	74	10	0.068432379	0.165764986
Finland			4			3	4	6					4		6	5			6	3					7	3				3			6	4	2		6	72	11	0.065352465	0.10753118
Romania			8	6	1	5			4						2	8	5					8	12		4	1						8						72	12	0.062560768	0.155212094
Switzerland			5	3								1	3	4	1		5		2	2	3				5		10	7	6		3				3		1	64	13	0.055868228	0.153864143
Poland				2	7				5		5		10	1	3	3			8			2	4	2							1		2			7		62	14	0.053973263	0.11229657
Iceland			2			5					7		2		5				7					6					1	8		1	4		6	4	58	15	0.052246035	0.165062483
Belarus		8		7														1		3		5	1					12									6	43	16	0.037978889	0.089227164
United Kingdom						1	7						3			4	8					4		3							5	5						40	17	0.029958317	0.144342619
Germany	4	6											5													8		2						7	5			39	18	0.027945852	0.136869443
Montenegro	6	12							12																													37	19	0.026911408	0.097193444
Greece	2	7		4	6								4					2	3			1						4									2	35	20	0.023817428	0.159278464
Italy	10								2		1											12	6											2					33	21	0.023425296	0.121638547
Azerbaijan					3								7																10	12						1		33	22	0.022613716	0.068005452
Malta	1			5						3							3		1											4					5	10	32	23	0.021816155	0.119063784
San Marino	3	3		3											4							1															14	24	0.009315756	0.098204303
Slovenia									1														8															9	25	0.005843162	0.156533823
France										1																								1				2	26	0.002167387	0.172574335
Albania																																						0	0	0.091897617	
Belgium																																						0	0	0.169584694	
Estonia																																						0	0	0.162056401	
FYR Macedonia																																						0	0	0.153993935	
Georgia																																						0	0	0.117423087	
Ireland																																						0	0	0.147584917	
Israel																																						0	0	0.153059334	
Latvia																																						0	0	0.166295198	
Lithuania																																						0	0	0.15992012	
Moldova																																						0	0	0.115874904	

Table B.6: Eurovision Song Contest 2014 (continued)

	Albania	Armenia	Austria	Azerbaijan	Belarus	Belgium	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	San Marino	Slovenia	Spain	Sweden	Switzerland	The Netherlands	Ukraine	United Kingdom	Points	Place	Authority Score	Hub Score
Portugal																																								0	0.173610946

Table B.7: Eurovision Song Contest 2015

	Albania	Armenia	Australia	Austria	Azerbaijan	Belarus	Belgium	Cyprus	Czech Republic	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	San Marino	Serbia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	United Kingdom	Points	Place	Authority Score	Hub Score
Sweden	7	7	12	7	6	10	12	10	10	12	10	5	12	8	7	10	4	10	12	10	10	12	12	10	10	8	5	12	12	8	8	8	7	8	12	8		12	10	12	365	1	0.247602	0.130248
Russia	8	12	10	8	12	12	10	5	8	10	12	6	8	10	5	12	8	6	3	8	8	10	10		7	10	7	2	6	10	10			10	5	10	6	7	6	6	303	2	0.203561	0.128412
Italy	12	6	8	10	8	1	8	12	7	5	3	7	2	6	8	3	12	2	6	6	12		8	1	12	7	6	5	7	12	12	12	10	7	8	12	8	6	7	8	292	3	0.193495	0.146217
Belgium	1	4	6	5		8		7	6	7	7	1	7	12	6	8	7	12	4	2	4	7	4	7		6		7	5	5	7	10	5	4	3	6	10	2	12	3	217	4	0.147482	0.145478
Australia	3	1		12		6	4	4		8	5		5	2		7	5	8	8	5	7	6	5	3	6	4		10	8		2	4	8	3	2	7	12	8	8	10	196	5	0.135929	0.157516
Latvia		2	7	1	5	5	7	3	5	4	6		6	7	4	6	3	5	7	12		4		12	4	2		8	10	2	5	2	12	1	7	4	5	4	2	7	186	6	0.126286	0.149527
Estonia	4		3	6	4	7	2	2		6		2	10		2	2		7	1		3	2	6	8	3		1	3	2	1		7		2		3	3		4		106	7	0.071542	0.145329
Norway			4	4						3	4		4			4		4	10	4		5	2	5	2				3	6	6		6		4	2	7	10	3		102	8	0.070783	0.144331
Israel	5		2	2	7	2		6					3		1	5			5			8	1		5		3	4	4	7	1		2	6		1	4	3	5	5	97	9	0.065718	0.146691
Serbia	2		5	3					3			10									2	3					12								6			5	1	1	53	10	0.030338	0.128584
Georgia		10	1		10	3	1		4		1						2	3						6		5						5									51	11	0.031356	0.111867
Azerbaijan									12						10									2	8	3	8				3	3									49	12	0.028747	0.125808
Montenegro	6	8			2							4																						12	10		2				44	13	0.026056	0.085579
Slovenia					3							8	1						2		6		3	4			4	1	1					5			1				39	14	0.022577	0.135649
Romania							5	1		2								1			5					12				4						5					35	15	0.023982	0.153499
Armenia						4	3		2			3		3	12		1															6									34	16	0.020110	0.119244

Table B.7: Eurovision Song Contest 2015 (continued)

	Albania	Armenia	Australia	Austria	Azerbaijan	Belarus	Belgium	Cyprus	Czech Republic	Denmark	Estonia	F.Y.R. Macedonia	Finland	France	Georgia	Germany	Greece	Hungary	Iceland	Ireland	Israel	Italy	Latvia	Lithuania	Malta	Moldova	Montenegro	Norway	Poland	Portugal	Romania	Russia	San Marino	Serbia	Slovenia	Spain	Sweden	Switzerland	The Netherlands	United Kingdom	Points	Place	Authority Score	Hub Score
Albania							6					12											6				10															34	17	0.016755	0.124384
Lithuania										1	2				3						7		7																		4	30	18	0.020556	0.118747
Greece	10	5						8																																	23	19	0.014358	0.125544
Hungary									1		8			1	1															4											19	20	0.013061	0.146121
Spain					1									5							1						1	2		3		1								15	21	0.009537	0.158995
Cyprus																	10																		1					11	22	0.006646	0.145637
Poland														4						3		1																	2	10	23	0.006909	0.159925
United Kingdom																				1				1			1					3								5	24	0.003258	0.153125
France	3																															1								4	25	0.002338	0.142272
Austria																																								0	26	0	0.150454
Germany																																									0	27	0	0.157865
Belarus																																									0		0	0.147342
Czech Republic																																									0		0	0.130862
Denmark																																									0		0	0.160507
FYR Macedonia																																									0		0	0.085414
Finland																																									0		0	0.149564
Iceland																																									0		0	0.145817
Ireland																																									0		0	0.141492
Malta																																									0		0	0.145890
Moldova																																									0		0	0.130904
Portugal																																									0		0	0.144985
San Marino																																									0		0	0.131533
Switzerland																																									0		0	0.146739
The Netherlands																																									0		0	0.155502

Bibliography

[Ber85] C. Berge. *Graphs and Hypergraphs.* Elsevier Science Ltd., Oxford, UK, UK, 1985.

[BGH+04] Vincent D. Blondel, Anahí Gajardo, Maureen Heymans, Pierre Senellart, and Paul Van Dooren. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Rev.*, 46(4):647–666, April 2004.

[Bin82] K.G. Binmore. *Mathematical Analysis: A Straightforward Approach.* Cambridge University Press, 1982.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[BR91] R.A. Brualdi and H.J. Ryser. *Combinatorial Matrix Theory.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1991.

[BR97] R.B. Bapat and T.E.S. Raghavan. *Nonnegative Matrices and Applications.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1997.

[Bul06] A. Bultheel. *Inleiding tot de numerieke wiskunde.* Acco, 2006.

[Cae11] S. Caenepeel. *Analyse: afleiden, integreren, wiskundige software.* Vrije Universiteit Brussel, 2011.

[car11] *Discrete Wiskunde.* Vrije Universiteit Brussel, 2011.

[Col12] E. Colebunders. *Topologie.* Vrije Universiteit Brussel, 2012.

[Dev09] M. Devos. Semidefinite programming. *Simon Fraser University*, 2009.

[DG09] P. Deift and D. Gioev. *Random Matrix Theory: Invariant Ensembles and Universality.* Courant Lecture Notes. American Mathematical Soc., 2009.

[DM11] W. De Meuter. *Algoritmen en Datastructuren I.* Vrije Universiteit Brussel, 2011.

[14] Eurovision Broadcasting Union (EBU/UER). Official website of the eurovision song contest.

[Eve80] H.W. Eves. *Elementary Matrix Theory.* Dover Books on Mathematics Series. Dover, 1980.

[Fou10] S. Foucart. *Linear Algebra and Matrix Analysis.* Drexel University, 2010.

[Fri04] S. Friedl. *Linear Algebra.* Rice University, 2004.

[GLPN93] Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. Directed hypergraphs and applications. *Discrete Appl. Math.*, 42(2-3):177–201, April 1993.

[GVL12] G.H. Golub and C.F. Van Loan. *Matrix Computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2012.

[HJ86] Roger A. Horn and Charles R. Johnson, editors. *Matrix Analysis.* Cambridge University Press, New York, NY, USA, 1986.

[Hor86] Roger A Horn. *Topics in Matrix Analysis.* Cambridge University Press, New York, NY, USA, 1986.

[Kle99] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[Lau04] Alan J. Laub. *Matrix Analysis For Scientists And Engineers.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004.

[Lif07] Y. Lifshits. Four results of jon kleinberg: A talk for st. petersburg mathematical society. *Steklov Insitute of Mathematics at St. Petersburg*, May 2007.

[LT85] P. Lancaster and M. Tismenetsky. *The Theory of Matrices: With Applications.* Computer Science and Scientific Computing Series. Academic Press, 1985.

[Mor13] J. Morrow. *Advanced Calculus.* University of Washington, 2013.

[Nou08] D. Noutsos. Perron-frobenius theory and some extensions. *Department of Mathematics, University of Ioannina*, May 2008.

[PZ14] KellyJ. Pearson and Tan Zhang. On spectral hypergraph theory of the adjacency tensor. 30(5):1233–1248, 2014.

[Saa00] Donald G. Saari. Mathematical structure of voting paradoxes. 15(1):1–53, 2000.

[Ste14] S. Sternberg. *Dynamical Systems.* Dover Publications, 2014.

[Tom13] C. Tomasi. Orthogonal matrices and the singual value decomposition. *Duke Universisty*, 2013.

[Tro05] J.A. Tropp. *Introduction to Numerical Methods.* California Institute of Technology, 2005.

[Var62] R.S. Varga. *Matrix iterative analysis.* Prentice-Hall series in automatic computation. Prentice-Hall, 1962.

[VDF09] Paul Van Dooren and Catherine Fraikin. Similarity matrices for colored graphs. *Bull. Belg. Math. Soc. Simon Stevin*, 16(4):705–722, 11 2009.

[Zag05] L. Zager. Graph similarity and matching. Master's thesis, Massachusetts Instiute of Technology, 2005.

[ZV08] Laura A. Zager and George C. Verghese. Graph similarity scoring and matching. *Applied Mathematics Letters*, 21(1):86 – 94, 2008.

Index