LOUVAIN
School of Management

UNIVERSITE CATHOLIQUE DE LOUVAIN

LOUVAIN SCHOOL OF MANAGEMENT

Communities in Networks: An empirical and comparative Study of some Detection Methods

Promoteur : Prof. Marco Saerens

Mémoire-recherche présenté par
Baptiste Jungers

en vue de l'obtention du titre de
Master 120 crédits en ingénieur de gestion

ANNEE ACADEMIQUE 2015-2016

I would like to thank my thesis supervisor, Professor Marco Saerens, for valuable assistance, conversations, and comments as well as for his time and availability that were valuable to the writing of this thesis.

I also wish to thank everyone that indirectly contributed to it. Their help and support over the last years were above everything a great source of motivation and inspiration.

Table of Contents

Part II. Experiments

1. INTRODUCTION

Graph Theory, the study of graphs or networks[1], is a branch of Mathematics and Computer Science and has a long history dating back to the 18th century [24]. Loosely speaking, a graph is a mathematical structure consisting of elements, called nodes, which can be pairwise linked by edges if some sort of interaction exists between them. Graph Theory was long only used in very particular scientific disciplines such as Mathematics, Sociology, Social Anthropology, Biology or Chemistry [63]. But this field experienced a growth of interest in the early 2000's. Because of the increasing availability of accurate network datasets of various sizes and powerful tools to analyze them, Physicists in particular as well as scientists from a wide range of disciplines began to perform empirical studies on networks and discovered statistical properties that are common to many real-world networks, either being natural or man-made [57]: Transitivity, small-worldness, graph diameter and properties related to nodes degree.

Another feature shared by many networks is that they often exhibit internal substructures called communities[2]. Communities are very specific in the sense that they consist of cohesive groups of nodes which are densely connected to each other but only sparsely connected or not connected at all to other communities in the graph [29]. Community detection, the focus of this thesis, is today an active and interdisciplinary field of research. But the need of discovering such structures is not new as communities and more specifically social communities have been an object of study in Sociology, the term community referring directly to the social context. However, because of the large amount of human work required, they often contained only a few dozens of nodes [63].

Often in the literature, the work achieved by two Physicists, namely Michelle Girvan and Mark E. J. Newman (see, e.g., [29, 53, 57, 58, 59]), is regarded as the starting point of the race towards the ideal method for community detection. Furthermore, as graphs can be used as an abstraction to represent many real-world complex systems, several disciplines also entered the race to determine reliable criteria that could define meaningful communities as well as algorithms of low computational complexity that could be applied on graphs of increasingly larger sizes. Indeed, the detection of meaningful communities and then the further investigation of their number, size and respective function in a network can have concrete applications in different fields. It could potentially help to understand the complex structure of a system, how it behaves,

[1] Graphs and networks will be used interchangeably in this thesis, graph being the mathematical term equivalent to network.
[2] Several terms equivalent to the concept of community are used interchangeably in the literature such as classes, groups, clusters or modules.

but also to predict how it will evolve or collapse. However, except the topological definition of community given above, no universally agreed-upon definition has appeared over the last fifteen years [20, 63, 70].

This thesis aims at studying different methods to accurately detect the communities contained inside a graph by pursuing the work of two former students, Augustin Collette and Joëlle Van Damme (see respectively [14, 77]). More specifically, we will address two research questions:

- First, we will compare the performance of two methods that return an optimal number of communities through the optimization of their respective criterion. The first one is the Louvain Method that aims at maximizing the criterion of modularity while the second one is the Relational Clustering based on the Condorcet's criterion. In a second time, as we know the community structure of our datasets, these methods will be compared on their performance when coerced into detecting a quantity of communities equal to the ground truth quantity.
- In a second time, still targeting the true number of communities, we will compare the performance of the two previous methods with a clustering method coming from the field of data mining: The k-means algorithm. We will try to retrieve the true communities by grouping the nodes in clusters in such a way that nodes inside a cluster are more similar or close to each other than to nodes from other clusters. This presupposes that we will have to first determine meaningful pairwise similarities and distances metrics between all the nodes of the graph.

In this perspective, besides this introduction, the content of this thesis is structured as follows:

- In the first part, the reader will be provided with a theoretical background of useful concepts used in this thesis: Graphs, measures of proximity (distance or similarity), kernels, clustering, graph partitioning and community detection. It is assumed that the reader has some basic knowledge of matrix theory, linear algebra and statistics as basic notions would not be explained here. The reader will also find in this part detailed descriptions of the algorithms that will be compared to each other regarding their performance. We will conclude this part by presenting some concrete applications of graphs study, clustering and community detection in the managerial world.
- In the second part, we will present the datasets on which our clustering algorithms will be tested. The respective performance per algorithm and dataset will be quantified through two quality metrics: The Adjusted Rand Index and the Normalized Mutual Information. The methodology used to achieve a quantitative and objective comparison between all methods and to reach reliable conclusions will be presented as well.

- The last part concludes this thesis by summarizing our key findings, discussing the limitations we encountered and proposing further work that could potentially address these limitations and refine our experimental results.

Part I.

Theoretical Background

2. GRAPHS AND NETWORKS

In this chapter, we will give a brief introduction to graphs by introducing related concepts and matrices. We will primarily outline theoretical aspects which are useful for the remainder of this thesis. For further details, we refer the interested reader to [20, 24, 44, 54, 70] as the present chapter is largely inspired by these references and to further references given in this chapter. Let us stress that, in this thesis, we will only consider *weighted undirected* and *unweighted undirected* graphs without *self-loops* which are often called *simple graphs*.

2.1. Introductive concepts

A *network* or *graph* is a non-empty collection of elements, called *vertices* or *nodes*, that can be pairwise connected by *edges* or *links*[3]. In the remainder of this thesis, we will more often use the terms of nodes and links than vertices and edges.

One could directly think to Social Network Sites as a network illustrative example where nodes are people or groups of people and links are friendships or other acquaintances. Yet another obvious example is the *Internet*[4], a network of computers linked together by data exchanges. Any complex system, either natural or human-built, can be represented under the form of a network. But a network being an abstraction of a real-world system, complex in essence, its use will induce a loss of information.

More formally, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a mathematical structure consisting of a non-empty set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$ which is a subset of $\mathcal{V} \times \mathcal{V}$. An edge is then a pair of nodes. Therefore, for a graph $\mathcal{G}$, two basic features are the cardinal of vertices $|\mathcal{V}| = n$ with $\mathcal{V} = \{1, \dots, n\}$, and the cardinal of edges $|\mathcal{E}| = e$ sometimes referred to as the *size* of the graph $\mathcal{G}$ and can yield a maximal value of $\binom{n}{2} = n(n-1)/2$. A graph $\mathcal{G}$ whose size is (nearly) maximal is called a (*near*-)*clique*.

A graph is said to be undirected if the relationship between all pairs of nodes is reciprocal so that the edge (i, j) is equivalent to (j, i) for all $(i, j) \in \mathcal{E}$ and recognized as one single edge. Otherwise, the graph is said to be *directed* (see, e.g., [44]). The absence of self-loops means that no edge connects a node to itself. A *weight* that can take any real value can be associated with

[3] Vertex and edge are mathematical terms because Graph Theory finds its origins in the field of Mathematics.

[4] Not to be confused with the World Wide Web, an undirected network (accessing a page through a hyperlink does not imply that you will find a hyperlink on this page to come back to the previous page) of nodes consisting of Web pages linked by hyperlinks that allow the user to navigate from one page to another one.

each edge making the graph *weighted*. Values and meanings of the weights are application-dependent and often subjective.

Two nodes are said to be *adjacent* or *neighbors* if an edge exists between them. The set of neighbors of node i is $\mathcal{N}(i) = \{j \mid (i,j) \in \mathcal{E}\}$. Furthermore, the *degree* of a node is the number of nodes adjacent to it or the sum of the weights (also called *strength* of a node) on the incident edges in the case of a weighted graph. The sum of the degrees or weights over all nodes is called the *volume* of the graph, $\text{vol}(\mathcal{G})$.

We also refine the definition of *community* as a cluster or group of nodes or as a subset of a graph $\mathcal{G}$, $\mathcal{G}^*(\mathcal{V}^*, \mathcal{E}^*)$ with $\mathcal{V}^* \subseteq \mathcal{V}$ and $\mathcal{E}^* \subseteq \mathcal{E}$, that is internally densely connected but only has sparse connections or none with other communities. A necessary condition for communities is *connectedness* stating that each pair of nodes inside a community must be linked by at least one *path* $\wp$ visiting only nodes belonging to this community. Generally speaking, if a sequence of distinct edges belonging to $\mathcal{E}$ can be found between two nodes, this sequence is called a path and these nodes are said to be *connected*.

Yet an important concept in this thesis and in Graph Theory, the concept of community is not well defined and no universal rigorous answer exists, the same being true for the concept of *partitioning* or *clustering* as it will be discussed later. As stated in [20], the problem of community detection remains an ill-posed problem.

2.2. Matrices related to graphs

In this section, we will present matrices that will be useful for the remainder of this thesis. We will only define matrices for undirected weighted and undirected unweighted graphs. Matrices are represented by uppercase bold characters and vectors by lowercase bold characters.

2.2.1. Adjacency matrix

The *adjacency matrix* $\mathbf{A}$ allows representing the structure of a graph $\mathcal{G}$. This matrix is square and of size $n \times n$. In the case of unweighted graphs, it is defined as

$$a_{ij} = [\mathbf{A}]_{ij} = \begin{cases} 1 \text{ if nodes } i \text{ and } j \text{ are adjacent} \\ 0 \text{ otherwise} \end{cases} \tag{2.1}$$

and $\mathbf{A}$ contains only binary values. We can define a *weighted adjacency matrix* as

$$a_{ij} = [\mathbf{A}]_{ij} = \begin{cases} w_{ij} \text{ if nodes } i \text{ and } j \text{ are adjacent} \\ 0 \text{ otherwise} \end{cases} \tag{2.2}$$

in the case of a weighted graph. Moreover, $\mathbf{A}$ is symmetric, i.e. $\mathbf{A} = \mathbf{A}^{\mathrm{T}} \Leftrightarrow a_{ij} = a_{ji}$ or $w_{ij} = w_{ji}$, if it represents an undirected graph and its diagonal contains only 0's due to the absence of self-loops in the graph.

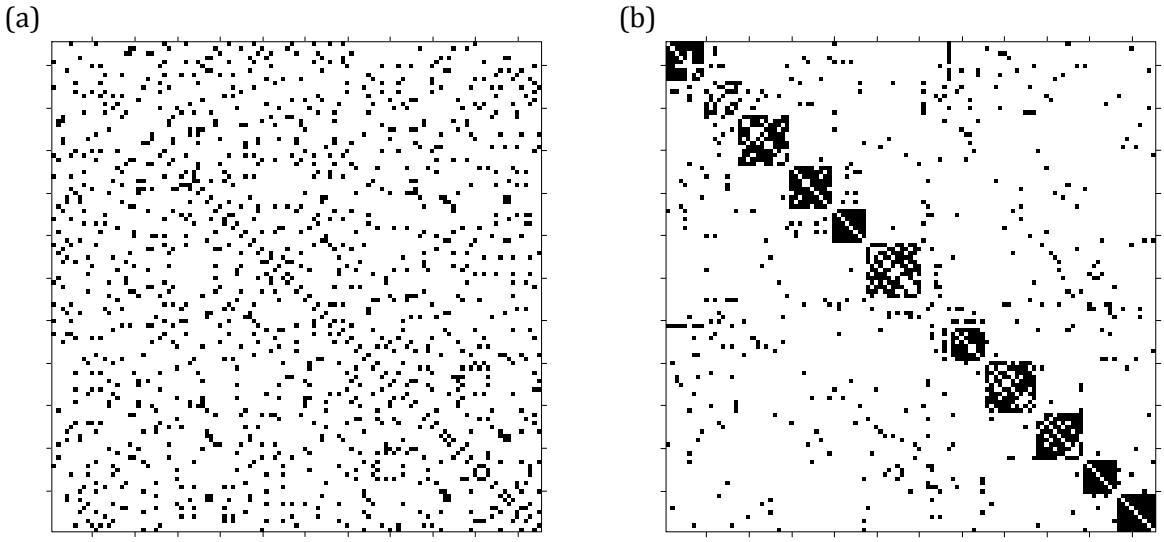

Figure 2.1: (a) Visual representation of the raw adjacency matrix for the 115-node and 12-class unweighted network dataset `foot` with black squares representing entries whose value is equal to 1. (b) The nodes were re-ordered according to the ground truth community labels making graph community structure partly visible.

2.2.2. Degree matrix

For an undirected unweighted graph, the *degree* of a node is the quantity of edges incident to it or the number of adjacent nodes. In the case of an undirected weighted graph, it is the total weight of the incident edges. First we define, in the case of an undirected graph, $\mathbf{d} = \mathbf{Ae}$ as the $n \times 1$ degree vector whose i-th element is the degree of node i and $\mathbf{e}$ is the unit column vector of length n containing only 1's. Then we can define the *degree matrix* $\mathbf{D}$ as $\mathbf{Diag}(\mathbf{d})$ whose entry $[\mathbf{D}]_{ii}$ is the degree of node i. Thus, the degree matrix is a square and diagonal matrix, of size $n \times n$, containing the degrees of the nodes on its diagonal.

For a directed graph, we must make a distinction between the *out-degree* and *in-degree* of a node. The former is the quantity of edges or the total weight starting from the node, the latter is the quantity of edges or the total weight ending at the node. The *out-degree vector* $\mathbf{d}_\mathrm{o}$ is computed as $\mathbf{Ae}$ and the *in-degree vector* $\mathbf{d}_\mathrm{i}$ is computed as $\mathbf{A}^\mathrm{T}\mathbf{e}$. As above, we can define the *out-degree matrix* $\mathbf{D}_\mathrm{o} = \mathbf{Diag}(\mathbf{d}_\mathrm{o})$ and the *in-degree matrix* $\mathbf{D}_\mathrm{i} = \mathbf{Diag}(\mathbf{d}_\mathrm{i})$.

2.2.3. Laplacian matrix

The *unnormalized Laplacian matrix* is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$ whose entries for an unweighted graph are

$$l_{ij} = [\mathbf{L}]_{ij} = \begin{cases} a_{i\bullet} = a_{\bullet j} \text{ if } i = j \\ -a_{ij} \text{ if } i \neq j \end{cases} \tag{2.3}$$

and for a weighted graph

$$l_{ij} = [\mathbf{L}]_{ij} = \begin{cases} w_{i\bullet} = w_{\bullet j} \text{ if } i = j \\ -w_{ij} \text{ if } i \neq j \end{cases} \tag{2.4}$$

Some useful properties of the Laplacian matrix are given in [68]. Among them, we can outline that the unnormalized Laplacian matrix $\mathbf{L}$ is square of size $n \times n$, symmetric because defined on an undirected graph, positive semidefinite and doubly centered with null row and column sums ($\mathbf{Le} = \mathbf{0}$ and $\mathbf{e}^{\mathrm{T}}\mathbf{L} = \mathbf{0}^{\mathrm{T}}$ with $\mathbf{0}$ being a column vector of length n containing only 0's) (see also, e.g., [3]).

Furthermore, the Laplacian matrix $\mathbf{L}$ also provides us with interesting information about its related graph [24]. Since it is positive semidefinite, all its eigenvectors are non-negative and its smallest eigenvalue must be 0. The multiplicity of 0 eigenvalues is equal to the number of connected components of the graph. A graph comprises only one single connected component if the graph is *connected* or, in other words, if any node can be reached from any other node of the graph. Therefore, the Laplacian matrix is always rank-deficient and singular and its inverse is not well-defined.

2.2.4. Common Neighbors matrix

Another matrix that we will consider in this thesis is the Common Neighbors matrix defined as

$$n_{ij} = [\mathbf{N}]_{ij} = \begin{cases} d_{ij} \text{ if } i = j \\ \mathbf{a}_i^{\mathrm{T}}\mathbf{a}_j \text{ if } i \neq j \end{cases} \tag{2.5}$$

where $\mathbf{a}_i = \mathbf{A}\mathbf{e}_i$ with $\mathbf{A}$ being the adjacency matrix of an undirected or directed unweighted graph and $\mathbf{e}_i$ being a column vector of length n containing 0's except element i being 1. The matrix $\mathbf{N}$ is square, of size $n \times n$, and symmetric. It contains on its diagonal the degrees of the nodes while the off-diagonal elements represent the number of adjacent nodes or neighbors two nodes i and j have in common. In matrix form, the matrix $\mathbf{N}$ is simply computed by $\mathbf{AA}$ in the case of undirected graphs. The off-diagonal values also represent the number of length-2 paths between pairs of nodes [24].

2.2.5. Modularity matrix

Modularity is a criterion for community detection that appeared in the field of Physics and was first introduced by Girvan and Newman in [59].

Prior to modularity criterion, Girvan and Newman proposed the *edge betweenness centrality* [29, 59]. The intuition behind it is that, if communities are groups of densely connected nodes but only sparsely connected with the rest of the graph or other communities, a lot of shortest paths (see Section 3.1.1) connecting pairs of nodes should run along the edges which are between the communities. The one-by-one removal of the edges having the highest degrees of betweenness centrality should lead to the emergence of communities by splitting the graph. Nevertheless, this criterion does not show how good, in some sense, the detected structure was. Indeed, even with random graphs (graphs whose nodes degree follows a Poisson distribution instead of a power-

law distribution[5] [54, 55]), meaningless structure tends to be discovered. This drawback triggered the introduction of the modularity criterion.

The first definition of modularity was given in [59]. Considering a partition[6] of k clusters or communities, we define a $k \times k$ symmetric matrix $\mathbf{E}$. The entry $e_{ij} = [\mathbf{E}]_{ij}$ is the quantity of edges that connect nodes belonging to cluster i to nodes belonging to cluster j, thus the entry $e_{ii} = [\mathbf{E}]_{ii}$ is the quantity of within-cluster edges. As for χ^2 statistic tests, the modularity criterion is based on the difference between the quantity of links e_{ii} whose both nodes are inside the cluster i and the quantity of links a_i^2 that have one or both nodes belonging to the same cluster i. If there is no community structure or if the edges distribution among nodes was independent of the cluster structure, we would have $e_{ij} = e_{i\bullet}e_{\bullet j} = a_i a_j$ [24, 59]. The modularity is then defined as the difference between the observed and the expected quantity in a random graph representing a null model [59]:

$$Q = \sum_i (e_{ii} - a_i^2) \tag{2.6}$$

where the sum is taken over all k clusters. Q values lie in the range $[-1,1]$ and the higher Q, the greater the departure from the hypothesis of edges distribution randomness. Typically, a value greater than 0.3 is considered as a significant indicator of community structure in the graph $\mathcal{G}$ [58, 59].

We now introduce the general expression of modularity for directed weighted graphs as [24]:

$$Q = \frac{1}{\mathrm{vol}(\mathcal{G})} \sum_{i,j \in \mathcal{V}} \left[a_{ij} - \frac{a_{i\bullet} a_{\bullet j}}{\mathrm{vol}(\mathcal{G})} \right] \delta(\ell_i, \ell_j) \tag{2.7}$$

where the Kronecker delta function $\delta(\ell_i, \ell_j)$ yields 1 if nodes i and j belong to the same cluster, i.e. $\ell_i = \ell_j$, and 0 otherwise. In matrix form, we have [24]:

$$Q(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m) = \frac{1}{\mathrm{vol}(\mathcal{G})} \sum_{k=1}^{m} \mathbf{u}_k^{\mathrm{T}} \mathbf{Q} \mathbf{u}_k \text{ , with } \mathbf{Q} = \left(\mathbf{A} - \frac{\mathbf{d}_{\mathrm{o}} \mathbf{d}_{\mathrm{i}}^{\mathrm{T}}}{\mathrm{vol}(\mathcal{G})} \right) \tag{2.8}$$

where $\mathbf{u}_k$ $(k = 1, \dots, m)$ is the membership binary vector of length n, i.e. the k-th column of the membership matrix $\mathbf{U}$, and $[\mathbf{u}_k]_i = 1$ if node i belongs to cluster k and 0 otherwise. As stated in [24], contrary to some other criteria the modularity has the nice property to not always increase when the number of communities increases.

[5] Graphs with nodes degree following a power-law distribution are also called *scale-free* networks [55].
[6] Strictly speaking, partition refers to a set of clusters that do not overlap; for a set of overlapping clusters, the term *cover* is often used [42]. Although overlapping communities is a very common feature of real-world networks [20], it will not be covered in this thesis.

3. DISSIMILARITIES AND DISTANCES

In this chapter, we will present the concept of distance or dissimilarity between nodes of a graph while the similarity, the opposite but related concept, will be introduced in the next chapter. Then we will explain how to capture distances between particular objects (e.g., the nodes of a graph which are not naturally represented by feature vectors) and present different distances metrics. We will also explain how useful the concept of distances is as most clustering algorithms such as the k-means algorithm rely on distance-based criteria [31].

It sounds intuitive to cluster nodes that are *similar* or *close*, in some sense, in the same community and to cluster nodes that are *dissimilar* or *distant* in separate communities. As mentioned in the introduction, there is no universal and rigorous definition of the concept of community. Nevertheless, we can make a distinction between three categories of definitions: Local definitions, global definitions and definitions based on the *features* of the nodes [24]. Indeed, to measure proximities (similarities or distances) between pairs of nodes, two sources of information can be considered, simultaneously, sequentially or independently: The *structure* of the graph at the local or global level and the features or *attributes* of the nodes [24]. A graph will be called an *attributed* graph when both sources are available and *plain* when only its structure is known [1].

However, in this thesis, we will only use the latter source, the structure of the graph, from which we will compute hopefully meaningful proximities (i.e., distances or similarities) between each pair of nodes. Nevertheless, we should keep in mind that, as stated in [34], considering only the structural properties of graphs may often not be sufficient to detect meaningful communities corresponding to the ground truth groups.

3.1. Dissimilarities and distances in a graph

Let us first precise the concepts of dissimilarity and distance (see, e.g., [36, 75]). If $\boldsymbol{\Delta}$ is a function taking as input the structure of a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and Δ_{ij} is the real value assigned by $\boldsymbol{\Delta}$ to nodes i and j. If for all pairs of nodes (i, j), i and $j \in \mathcal{V}$, the function $\boldsymbol{\Delta}$ satisfies the following conditions:

- Non-negativity: $\Delta_{ij} \geq 0$;
- Symmetry: $\Delta_{ij} = \Delta_{ji}$;
- Reflexivity: $\Delta_{ij} = 0$ if and only if $i = j$;

it is called a *dissimilarity measure* or *dissimilarity index*. Additionally, if it also satisfies the triangle inequality, $\Delta_{ik} \leq \Delta_{ij} + \Delta_{jk}$, it is called a *distance measure* or *distance metric*.

3.1.1. Shortest-Path distance

Introduced here but not directly used in this thesis, the most intuitive distance measure between two nodes is certainly the length of the shortest path also called *Geodesic* or *Djikstra distance* which is defined as the lower total cost to join these two nodes – with the cost for each edge being defined as $c_{ij} = 1/a_{ij}$ [54]. In the case of unweighted graphs, the length of a path between two nodes i and j is then measured as a quantity of edges since $c_{ij} = 1$ if $a_{ij} = 1$. The shortest path is not necessarily unique as several shortest paths having the same length can exist between two nodes [54, 56]. We could also define the *diameter* of a graph as the longer shortest-path for all the pairs of nodes [54]. Although this distance measure was very popular in different fields, it suffers from limitations as it does not take into account the degree of connectivity of the nodes or, in other words, nodes that are strongly connected should be considered as closer than what the Shortest-Path distance indicates [23].

3.1.2. (Euclidean) Commute-Time distance

The *Commute-Time distance* and variants were broadly investigated by Saerens and co-authors in [22, 23, 25, 68, 71, 81, 82, 83] – upon which this section is largely based – is based on the Markov-chain model of *random walks*, where a Markov-chain is a sequence of nodes visited by a random walker. Contrary to paths, some nodes can be visited several times in a walk. A state is associated with each node and the *transition probability* p_{ij} of moving from current state i to another state corresponding to an adjacent node j is given by [68]

$$\mathrm{P}(s(t+1) = j | s(t) = i) = a_{ij}/a_{i\bullet} = a_{ij}/d_{ii} \tag{3.1}$$

and depends only on the current state and not on any previous ones. From this model, we can extract several quantities of interest [83].

The first one is the *Average First-Passage Time*, $m(j|i)$, also called *Hitting Time*, H_{ij}, the expected time or the average number of steps for a random walker to reach node or state j for the first time when starting from state or node i [68].

The second quantity is the *Average Commute-Time* $n(i,j)$, or simply the *Commute-Time distance* hereafter, defined as the number of steps needed by a random walker starting from node i to reach a node j for the first time and to come back to node i and is defined as [68]

$$n(i,j) = m(j|i) + m(i|j) = H_{ij} + H_{ji} \tag{3.2}$$

This quantity was shown to be a valid distance metric between any pairs of nodes in a graph $\mathcal{G}$ because it satisfies all conditions of the previous conditions [30]. It is equivalent up to a constant factor $\mathrm{vol}(\mathcal{G})$ to the *Resistance distance* (introduced by Klein and Randić in [40]), the effective resistance between two nodes when considering a graph as an electrical network with edges

seen as resistors, which was shown to be a valid distance metric as well [40]. Both distances can be computed from matrix $\mathbf{L}^+$, the Moore-Penrose pseudoinverse (see, e.g., [4]) of the Laplacian matrix $\mathbf{L}$ (also called *Admittance matrix* in [40]), with elements $l_{ij}^+ = [\mathbf{L}^+]_{ij}$. Since $\mathbf{L}^+$ is symmetric, has the same eigenvectors as $\mathbf{L}$ but inverse eigenvalues, and since $\mathbf{L}$ is positive semidefinite, $\mathbf{L}^+$ is also positive semidefinite [23].

The Commute-Time distance matrix contains squared distances between each pair of nodes and its entries are defined as [23, 40, 68]:

$$[\mathbf{\Delta}_{\mathrm{CT}}^{(2)}]_{ij} = n(i,j) = \mathrm{vol}(\mathcal{G})\big(\mathbf{e}_i - \mathbf{e}_j\big)^{\mathrm{T}}\mathbf{L}^+(\mathbf{e}_i - \mathbf{e}_j) \tag{3.3}$$

and in matrix form as [24]:

$$\mathbf{\Delta}_{\mathrm{CT}}^{(2)} = \mathrm{vol}(\mathcal{G})\left(\mathbf{diag}(\mathbf{L}^+)\mathbf{e}^{\mathrm{T}} + \mathbf{e}\big(\mathbf{diag}(\mathbf{L}^+)\big)^{\mathrm{T}} - 2\mathbf{L}^+\right) \tag{3.4}$$

with $\mathbf{e}_i$ and $\mathbf{e}_j$ being the i-th and j-th column of the identity matrix $\mathbf{I}$, that is the basis vectors representing each node in the Euclidean space $\mathbb{R}^n$. Since the Moore-Penrose pseudoinverse of $\mathbf{L}$ is positive semidefinite, the square root of the Commute-Time distance is a Euclidean distance and therefore called *Euclidean Commute-Time distance* [23, 68, 83]. A distance is said to be *Euclidean* if any objects i and j can be represented in a Euclidean space that exactly preserves the distances $\Delta_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ between the objects, with $\{\mathbf{x}_i\}_{i=1}^n$ being the feature vectors of these n objects in this Euclidean space [24].

Moreover, the Commute-Time distance is equivalent to a squared Mahalanobis distance with $\mathbf{L}^+$ playing the role of the covariance matrix, and where matrix $\mathbf{L}^+$ can be computed thanks to [24]:

$$\mathbf{L}^+ = \left(\mathbf{L} - \frac{\mathbf{e}\mathbf{e}^{\mathrm{T}}}{n}\right)^{-1} + \frac{\mathbf{e}\mathbf{e}^{\mathrm{T}}}{n} \tag{3.5}$$

These distances decrease as the size of the set of direct and indirect paths between two nodes increases and when the length of the paths decreases: The more short paths between nodes i and j, the closer the nodes. On the contrary, the Shortest-Path distance does not necessarily decrease when nodes become more connected [23].

Nevertheless, von Luxburg *et al*. [78] strongly discourage the use of the Commute-Time distance on large graphs with size in the order of 1,000 nodes and even on graphs of smaller size for several reasons. First, because it implies the computation of the pseudoinverse Laplacian matrix $\mathbf{L}^+$ which can be computationally expensive for large graphs although approximation approaches can be used[7] [83]. Still in [78], the authors show that, in large graphs, the Commute-Time distance $n(i,j)$ between any pairs of nodes often tends to a limit value of

[7] As we experienced it, the computational complexity of an algorithm always is a matter of importance though not directly treated in this thesis.

$$\frac{1}{\mathrm{vol}(\mathcal{G})} n(i,j) \approx \frac{1}{d_{ii}} + \frac{1}{d_{jj}} \tag{3.6}$$

exclusively determined by local properties, i.e. the degree values of i and j, and not by any global property of the graph. Moreover, the range of values taken by the Commute-Time distances can be wide and also be an issue [78].

To sum up [24], on one hand, we have the Shortest-Path distance which does not take into the degree of connectivity between nodes, but only the length of the shortest path. On the other hand, we have the Commute-Time distance which captures this desirable property but tends to a meaningless limit value in large graphs. These unpleasant outcomes have led the researchers to propose the Corrected Commute-Time distance discussed in the next section, the Sigmoid Commute-Time kernel introduced in the next chapter and other alternatives which will not be covered in this thesis, but we refer the interested reader to [14] for instance.

3.1.3. (Euclidean) Corrected Commute-Time distance

Von Luxburg *et al*. [78] showed that the value of the Commute-Time distance between two nodes is dominated by the degrees of these nodes because the random walker (see Section 3.1.2) "*gets lost*" in large graphs. This is why they introduced the *Corrected Commute-Time distance* (called *Amplified Commute distance* in [78]) where the original Commute-Time distance value is corrected by removing the influence of the nodes degree. It is computed from the raw Commute-Time distance as follows [24]:

$$\boldsymbol{\Delta}_{\mathrm{CCT}}^{(2)} = \boldsymbol{\Delta}_{\mathrm{CT}}^{(2)} - \mathrm{vol}(\mathcal{G}) \left[\mathbf{diag}(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1})\mathbf{e}^{\mathrm{T}} + \mathbf{e}\big(\mathbf{diag}(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1})\big)^{\mathrm{T}} - 2(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}) \right] \tag{3.7}$$

As for the Euclidean Commute-Time distance, the Euclidean Corrected Commute-Time is computed by taking the square root of the Corrected Commute-Time distance.

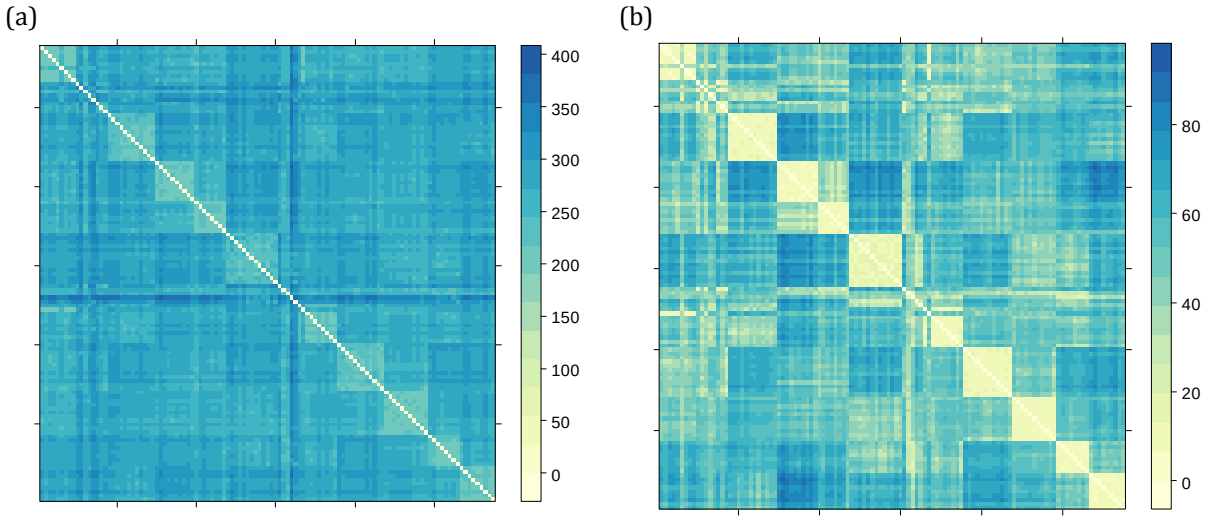

Figure 3.1: A graphical representation of the matrices containing the pairwise (a) Commute-Time and (b) Corrected Commute-Time distances for the 115-node and 12-class dataset `foot` with nodes reordered according to the ground truth partition. Low values indicate pairs of nodes that are close according to these two distance metrics.

4. SIMILARITIES AND KERNELS

Kernels are a famous tool used in different disciplines of data science to produce a $n \times n$ pairwise similarity matrix between n objects called a kernel matrix $\mathbf{K}$. When these objects are the nodes of a graph $\mathcal{G}$, the kernel matrix $\mathbf{K}$ is then called *kernel on a graph matrix*. As for distances, similarities between nodes can be captured either from the features of the nodes regardless of their connectivity or from the structure of the graph at a local or a global level [24].

4.1. Kernel definition

A *kernel* is a function that returns a real number representing the similarity between two objects i and j. More precisely, the similarity between two objects that are defined in an *input space* $\mathcal{I}$ is the inner product between the feature vectors of these objects in a space $\mathcal{F}$ called *vector*, *feature* or *inner product space* [88].

When objects cannot be naturally represented as feature vectors as it is encountered, inter alia, with graphs, we must first extract a set of features through a mapping function from the input space to the feature space, $\phi : \mathcal{I} \to \mathcal{F}$. As this step can be difficult to perform or even impossible, kernel methods allow us to compute the inner product similarity without having to explicitly compute the feature vectors space through the mapping function $\phi(\cdot)$ [88].

Further work on $\mathbf{K}$ will be equivalent to performing it on a data matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^{\mathrm{T}}$ containing the transposed feature vectors on its rows. If we are working with graphs, the objects are the n nodes of the graph represented by n feature vectors called *node vectors* and defined in the feature space [22, 24, 88].

4.2. Kernel matrices

Let $K : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$ be a function that maps pairs of objects from their input space $\mathcal{I}$ to their inner product value in the inner product space or feature space $\mathcal{F}$, that is $K(i,j) = \phi(i)^{\mathrm{T}}\phi(j)$ with $\phi(i) = \mathbf{x}_i$ and $\phi(j) = \mathbf{x}_j$, and let $\mathbf{K}$ be the $n \times n$ corresponding kernel matrix of the $n \times p$ data matrix $\mathbf{X}$ [88].

The function $K(i,j)$ is a positive semidefinite kernel function if and only if it is symmetric (i.e., $K(i,j) = K(j,i)$), and the resulting kernel matrix $\mathbf{K}$ is positive semidefinite (i.e., $\mathbf{x}^{\mathrm{T}}\mathbf{K}\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$). Any function, although returning meaningful (intuitive and practically adequate for a given application) similarities between objects, which is not positive semidefinite cannot be called a kernel but only a similarity function [22, 24, 88]. On the contrary, a similarity matrix that is positive semidefinite is called a kernel or sometimes a *valid* kernel to outline its positive

semidefiniteness [24]. A symmetric but not positive semidefinite similarity matrix, and thus not a valid kernel matrix, can be made positive semidefinite by adding $\alpha\mathbf{I}$ to it, with $\alpha \geq |\lambda_{\min}|$ and $\lambda_{\min}$ being the lowest eigenvalue of the similarity matrix [22].

Finally, we should outline that several kernel functions exist and while the inner product function is certainly the most popular similarity measure it is not necessarily the most meaningful [22, 24].

4.3. Distances from kernels and vice versa

Once we have a meaningful valid kernel on a graph matrix $\mathbf{K}$, several other matrices of interest can be automatically deduced from it. Among them, by *spectral decomposition* of the kernel matrix, we could compute the data matrix $\mathbf{X}$ of size $n \times p$, p being the rank of the kernel matrix $\mathbf{K}$, representing the nodes in a p-dimension Euclidean space, i.e. the feature space [22, 25].

A valid and Euclidean distance metric between any pair of nodes of the graph could also be automatically deduced from a valid kernel matrix $\mathbf{K}$ [22, 25]:

$$\begin{aligned} \Delta_{ij}^2 &= \|\mathbf{x}_i - \mathbf{x}_j\|^2 = (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j) = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_i + \mathbf{x}_j^{\mathrm{T}}\mathbf{x}_j - 2\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j \\ &= k_{ii} + k_{jj} - 2k_{ij} \\ &= (\mathbf{e}_i - \mathbf{e}_j)^{\mathrm{T}}\mathbf{K}(\mathbf{e}_i - \mathbf{e}_j) \end{aligned} \tag{4.1}$$

And the distance matrix containing all pairwise squared distances Δ_{ij}^2 is computed as [24]:

$$\boldsymbol{\Delta}^{(2)} = \mathrm{diag}(\mathbf{K})\mathbf{e}^{\mathrm{T}} + \mathbf{e}\big(\mathrm{diag}(\mathbf{K})\big)^{\mathrm{T}} - 2\mathbf{K} \tag{4.2}$$

The other way around, given a symmetric pairwise Euclidean distance matrix and from the theory of Multidimensional Scaling (see, e.g., [9]), we can compute the inner products from the distances [24]. But while distances are invariant with respect to the origin of coordinates, inner products are not. We then fix the origin of the coordinates to the centroid of the node vector which in turn implies that $\sum_{i=1}^{n} k_{ij} = \sum_{j=1}^{n} k_{ij} = 0$ and thus the inner product matrix is centered, i.e. $\mathbf{Ke} = \mathbf{0}$ where $\mathbf{e}$ is vector of length n containing only 0's values. In that case, the centered inner product matrix is provided by [24]

$$\mathbf{K} = -\frac{1}{2}\mathbf{H}\boldsymbol{\Delta}^{(2)}\mathbf{H} \tag{4.3}$$

where $\mathbf{H} = (\mathbf{I} - \mathbf{e}\mathbf{e}^{\mathrm{T}}/n)$ is the centering matrix and $\boldsymbol{\Delta}^{(2)} = \boldsymbol{\Delta} \circ \boldsymbol{\Delta}$ a squared distance matrix with $\circ$ being the Hadamard element-wise product. If the distance metric is Euclidean, the centered inner product matrix is positive semidefinite.

4.4. (Corrected) Commute-Time kernel

Commute-Time kernel was first introduced in [68]. Let us recall (see Section 3.1.2) that the matrix containing the squared Euclidean Commute-Time distances or equivalently the Commute-Time distances is obtained by

$$\mathbf{\Delta}_{\mathrm{ECT}}^{(2)} = \mathrm{vol}(\mathcal{G})\left(\mathbf{diag}(\mathbf{L}^{+})\mathbf{e}^{\mathrm{T}} + \mathbf{e}\big(\mathbf{diag}(\mathbf{L}^{+})\big)^{\mathrm{T}} - 2\mathbf{L}^{+}\right) \tag{4.4}$$

In the previous section, we learnt that a centered inner product matrix and valid kernel $\mathbf{K}$ can be computed from squared Euclidean distances as $\mathbf{K} = -\frac{1}{2}\mathbf{H}\mathbf{\Delta}^{(2)}\mathbf{H}$. From Equation 4.4 and since $\mathbf{He} = \mathbf{0}$ and $\mathbf{L}^{+}$ is already doubly centered ($\mathbf{HL}^{+}\mathbf{H} = \mathbf{L}^{+}$), we have

$$\mathbf{K} = \mathrm{vol}(\mathcal{G})\mathbf{L}^{+} \tag{4.5}$$

The pseudoinverse Laplacian matrix $\mathbf{L}^{+}$ contains the inner products of the node vectors in a Euclidean embedding space where the nodes are exactly separated by the Euclidean Commute-Time distances up to a scaling factor $\mathrm{vol}(\mathcal{G})$ [23, 24]. The Commute-Time kernel is then simply defined as

$$\mathbf{K}_{\mathrm{CT}} = \mathbf{L}^{+} \tag{4.6}$$

As for the Commute-Time distance, the Commute-Time kernel does not scale well on large graphs but seems less affected than the Commute-Time distance by the "getting lost" effect [24]. Moreover, in some applications, inner-product based measures outperform Euclidean-distance based approaches [23].

To compute the Corrected Commute-Time kernel, we apply the same transformation as above which leads to [24]:

$$\mathbf{K}_{\mathrm{CCT}} = \mathbf{L}^{+} + \mathbf{HD}^{-1}\mathbf{AD}^{-1}\mathbf{H} = \mathbf{K}_{\mathrm{CT}} + \mathbf{HD}^{-1}\mathbf{AD}^{-1}\mathbf{H} \tag{4.7}$$

4.5. Sigmoid (Corrected) Commute-Time kernel

Introduced by Yen *et al.* [81], the Sigmoid Commute-Time kernel is obtained by applying an element-wise sigmoid transformation on the Commute-Time kernel $\mathbf{K}_{\mathrm{CT}} = \mathbf{L}^{+}$:

$$\left[\mathbf{K}_{\mathrm{CT}}^{\mathrm{S}}\right]_{ij} = \frac{1}{1 + \exp(-\alpha l_{ij}^{+}/\sigma)} \tag{4.8}$$

where σ is the normalizing term corresponding to the standard deviation of the elements of the Laplacian matrix $\mathbf{L}^{+}$ and α is a tuning parameter to determine the sharpness of the sigmoid curve. Thus the values of the Sigmoid Commute-Time kernel lie in the range [0,1]. But as the Sigmoid Commute-Time kernel is not necessarily positive semidefinite, it cannot be considered as a valid kernel [81].

However, as stated in [82], it is shown that the kernel k-means based on the Sigmoid Commute-Time kernel converges to a minimum of the criterion even if the latter becomes negative. It also

performs better than the Commute-Time kernel $\mathbf{K}_{\mathrm{CT}}$ as it has a smaller range of values. Indeed, this range reduction results from the sigmoid transformation and it is desirable as the k-means is highly sensitive to outliers. Furthermore, as shown in [14, 77], making this kernel positive semidefinite does not significantly enhance the quality of clustering partitions.

Finally, the Sigmoid Corrected Commute-Time $\mathbf{K}^{\mathrm{S}}_{\mathrm{CCT}}$ is computed through the same sigmoid transformation applied this time on the Corrected Commute-Time kernel $\mathbf{K}_{\mathrm{CCT}}$. As the Commute-Time kernel, these kernels do not scale well on large graphs [24].

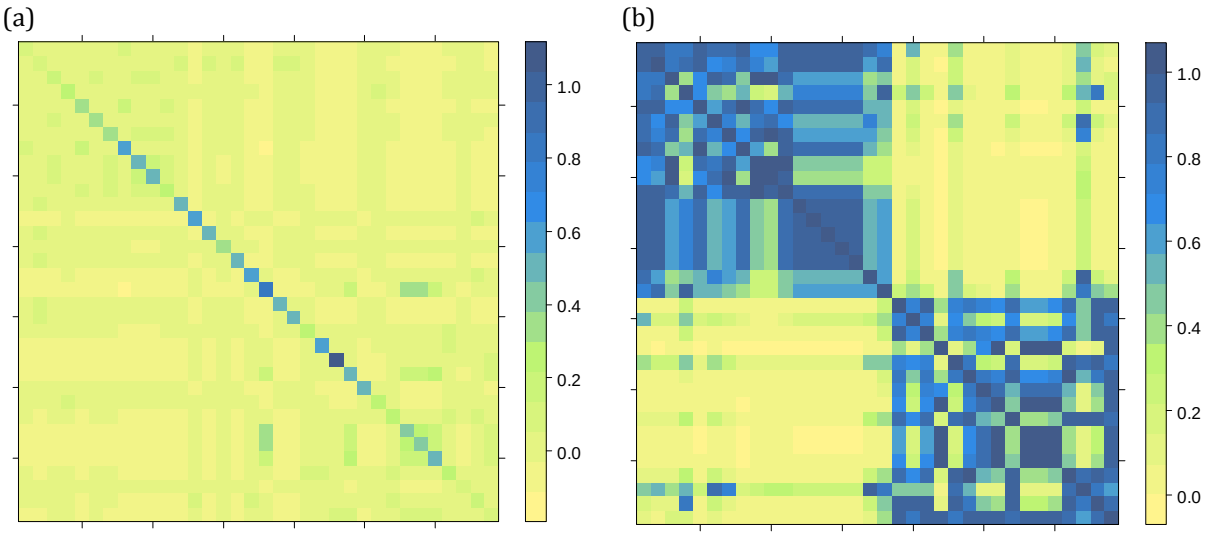

Figure 4.1: A graphical representation of (a) the Commute-Time kernel and (b) Sigmoid Commute-Time kernel ($\alpha = 7$) matrices for the 34-node and 2-class dataset `zachary` with nodes re-ordered according to their true class. Large values indicate highly similar nodes. For this dataset, true classes are more easily identifiable on the diagonal with the Sigmoid Commute-Time kernel.

5. NODE CLUSTERING AND COMMUNITY DETECTION

Strictly speaking, as stated in [20], we will be concerned by *community detection* if the graph is *sparse*, that is the number of edges is roughly equal to the number of nodes. Let us recall that, loosely speaking, communities can be defined as groups of nodes that show a large quantity of internal connections in comparison with the quantity of connections they have with the rest of the graph. Then if the number of edges is too large in comparison to the number of nodes, the distribution of edges among the nodes is too homogeneous for the definition of community to remain valid. Therefore, if the number of edges is significantly larger than the number of nodes, the problem is more related to *data clustering*, i.e. clustering of objects that are close, similar or distant in some sense [20].

We can also make a distinction between *node clustering* or *graph partitioning* (see Section 5.1) and *community structure detection* (see Section 5.2). Indeed, the detection of cohesive groups in networks can be made through two approaches [53]:

- Graph partitioning: With this approach, it is assumed that the number of groups into which to split the graph is known or has to be provided in advance. Even if a graph exhibits no meaningful structure of interest, any clustering algorithm will generate a partition of the nodes consisting of a specified number of clusters.
- Community structure detection: Here we assume that networks exhibit some natural subgroups, whose quantity and size are determined by the structure of the graph, and the objective is then to reveal these communities.

For the reader's convenience, we will consider this categorization but more refined ways to classify methods that aim at finding cohesive groups in graphs can be found in the literature.

5.1. Node clustering

As for the concept of community, it is somewhat difficult to find a rigorous definition of clustering because it directly refers to the concept of *clusters*, which is usually defined with vague terms [36, 75]. Several definitions of this concept are given in [36]. Informally, we can say that clustering is the process of grouping similar objects in the same cluster such that they are more similar to one another than to those of other clusters [36].

Now more specifically, *hard or crisp partitioning clustering methods* aim at producing a partition $\mathcal{C} = \{c_1, \dots c_k \dots, c_m\}$ of n objects (e.g., the nodes set $\mathcal{V}$ of a graph $\mathcal{G}$) into a specified number m clusters, with $m \leq n$, under the following conditions [75]:

- $c_k \neq \emptyset$ for all $k = 1, \ldots, m$;
- $\bigcup_{k=1}^{m} c_k = \mathcal{V}$;
- $c_k \cap c_l = \emptyset$ for $k, l = 1, \ldots, m$ and $k \neq l$.

The partition $\mathcal{C}$ then contains m mutually disjoint and non-empty clusters so that each object belongs to one and only one cluster. This type of clustering is said to be hard or crisp by contrast to *fuzzy clustering* introduced in [87] which allows overlapping clusters, i.e. each object can belong to more than one cluster with a certain level of probability.

5.1.1. k-means algorithm

In this section, we will present the standard formulation of the k-means paradigm whereas variants will be presented in the next sections. The k-means algorithm is among all clustering algorithms the most popular because it can be easily described, implemented and used [75]. Given a set of n objects represented in a Euclidean space by feature vectors $\{\mathbf{x}_i\}_{i=1}^{n}$, the objective is to find a partition of k clusters such that the cost function J, the sum of the clusters within-inertia, is minimized leading to compact and spherical clusters [31, 75]:

$$J = \sum_{k=1}^{m} J_k = \sum_{k=1}^{m} \sum_{i \in c_k} \|\mathbf{x}_i - \mathbf{m}_k\|^2 \tag{5.1}$$

The within-inertia for a cluster c_k is the sum of the squared Euclidean distances $\|\mathbf{x}_i - \mathbf{m}_k\|^2$ between all objects in the cluster to its representative, that is the centroid $\mathbf{m}_k$ defined as the mean value computed from the feature vectors of the objects belonging the cluster.

The clustering procedure to minimize the cost function is performed by an iterative reallocation technique [75]: (1) re-assigning each object to the cluster whose centroid is the closest and (2) re-computing the centroid for each cluster. Initial centroids were randomly chosen among the set of objects. The algorithm iterates until convergence to a stationary value of the cost function $J(t)$, t being the number of iterations, i.e. when the objects membership does not change anymore. As the algorithm is a heuristic with greedy approach, there is no guarantee that it will converge to a global minimum. Indeed, the quality of the final partition depends on the initial prototypes. Moreover, convergence to a minimum, even local, is not guaranteed with distances other than squared Euclidean distances, even the raw Euclidean distance itself [75]. As a rule of thumb, because of the non-deterministic characteristic of this algorithm, one should run it several times with different initial prototypes and keep the partition having the lowest value of J [31, 75].

As we are working on structured objects, we do not have the data matrix containing feature vectors available. Instead, we will directly work with a matrix of distances $\mathbf{\Delta}$ (see Chapter 3) between each pair of nodes and with a kernel on a graph (see Chapter 4).

5.1.1.1. Distance-based k-means

Here, we will work with the distance metrics presented in Chapter 3 and we will use the *total within-cluster sum of distances* J as distance-based criterion. For each cluster c_k, the within-cluster sum of distances J_k defined as [83]

$$J_k = \sum_{i \in c_k} \Delta_{q_k i} \tag{5.2}$$

with q_k being the *prototype*[8] of cluster k, its most typical component, that is the node displaying the lowest sum of distances to all other nodes belonging to the same cluster k. Here, prototypes are used instead of centroids as representatives of the clusters and, contrary to the latter, the former are necessarily a node of the graph and not simply the center of gravity. The choice of nodes as representatives makes clustering process less sensitive to outliers but at the cost of increased computational complexity [31, 75]. The total within-cluster sum of distances can then be defined as the sum over all m clusters [83]:

$$J = \sum_{k=1}^{m} J_k = \sum_{k=1}^{m} \sum_{i \in c_k} \Delta_{q_k i} \tag{5.3}$$

The two steps of the standard definition of the k-means algorithm are repeated iteratively till convergence of the criterion. Using a distance-based k-means instead of the standard k-means is mandatory as we do not work with feature vectors representing the nodes of the graph but it can also allow avoiding inherent limitations of the standard k-means as we do not have to compute the data matrix $\mathbf{X}$.

We will consider the Euclidean Commute-Time and the Euclidean Corrected Commute-Time distances instead of their squared homologues as the range of the metrics is less wide, which is highly desirable as the k-means algorithm is sensitive to wide-range metrics and outliers [31, 75].

5.1.1.2. Kernel-based k-means

First, let us define the different spaces that could be considered when working with kernel-based clustering algorithms [24, 82]:

- The *input space* in which the data are defined, each coordinate in this space representing the measurement of a characteristic on objects (see Section 4.1). As we are working with structured objects, this space is not relevant in our case.
- The *feature space* is the space resulting of a non-linear mapping of the input space (see Sections 4.1 and 4.2). In our case, it is a Euclidean embedding space preserving exactly

[8] As the prototypes are chosen among the set of nodes, they will be identified by an integer index such as the corresponding row or column index of the adjacency matrix.

the distances between the nodes of a graph. The nodes vectors representing the nodes in this space do not need to be explicitly computed as we directly work with kernel or similarity matrices.

- The *sample space* corresponds to the Euclidean space having as dimensionality the number of nodes n. Thus, each coordinate in this space corresponds to a node of the graph.

For the kernel-based k-means algorithm, we will work in the sample space rather than in the feature space as for the standard k-means.

Let us start with the standard optimization problem that the standard k-means algorithm aims at solving and which can be expressed in the feature space as [81]

$$J(\mathbf{g}_1, \dots, \mathbf{g}_m) = \sum_{k=1}^{m} \sum_{i \in c_k} \|\mathbf{x}_i - \mathbf{g}_k\|^2 \tag{5.4}$$

where m is the number of clusters specified by the user, $\mathbf{x}_i$ is the node vector of node i, $\mathbf{g}_k$ the prototype of cluster k in the feature space and $\|\mathbf{x}_i - \mathbf{g}_k\| = \left((\mathbf{x}_i - \mathbf{g}_k)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{g}_k)\right)^{1/2}$ is the L_2- or Euclidean distance (see Section 3.1.2) between the node i and the prototype of the cluster it belongs to. Let us recall that the node vectors represent the nodes in the Euclidean embedding space preserving exactly the distances between the nodes. We can define the $n \times p$ data matrix $\mathbf{X}$ which contains the n nodes vectors as rows, p being the number of features in the p-dimension embedding space [24, 82].

Let us now operate the change of parameter, $\mathbf{g}_k \rightarrow \mathbf{X}^{\mathrm{T}}\mathbf{h}_k$, the so-called "*kernel trick*" which allows to express $\mathbf{g}_k$ as a linear combination of the node vectors $\mathbf{x}_i$ and therefore avoiding to compute the nodes and the prototypes vectors, $\mathbf{x}_i$ and $\mathbf{g}_k$. The vector $\mathbf{h}_k$ is the prototype vector of cluster k in the n-dimension sample space, n being the number of nodes of the graph $\mathcal{G}$. If we compute the expression in terms of $\mathbf{h}_k$, we obtain (see [24, 81, or 82] for the mathematical proof):

$$\begin{aligned} J(\mathbf{h}_1, \dots, \mathbf{h}_m) &= \sum_{k=1}^{m} \sum_{i \in c_k} \|\mathbf{x}_i - \mathbf{g}_k\|^2 \\ &= \sum_{k=1}^{m} \sum_{i \in c_k} (\mathbf{x}_i - \mathbf{g}_k)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{g}_k) \\ &= \sum_{k=1}^{m} \sum_{i \in c_k} (\mathbf{e}_i - \mathbf{h}_k)^{\mathrm{T}}\mathbf{K}(\mathbf{e}_i - \mathbf{h}_k) \end{aligned} \tag{5.5}$$

where $\mathbf{K}$ is the kernel matrix containing the inner products, i.e. $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$. As already mentioned, the criterion is minimized by the iteration of two steps, namely the nodes re-allocation and the

prototypes re-computation steps, till convergence to a local minimum. If we then introduce the binary membership variables u_{ik}, the criterion J becomes

$$J(\mathbf{h}_1, \dots, \mathbf{h}_m) = \sum_{k=1}^{m} \sum_{i \in c_k} u_{ik} (\mathbf{h}_k - \mathbf{e}_i)^{\mathrm{T}} \mathbf{K} (\mathbf{h}_k - \mathbf{e}_i) \tag{5.6}$$

and each node i is reallocated in the cluster k for which $(\mathbf{h}_k - \mathbf{e}_i)^{\mathrm{T}} \mathbf{K} (\mathbf{h}_k - \mathbf{e}_i)$ is minimal. As shown in [81], the prototype vector of cluster k, $\mathbf{h}_k$, defined in the n-dimension sample space is simply computed after the nodes reallocation step as follows

$$\mathbf{h}_k = \frac{1}{n_k} \sum_{i \in c_k} \mathbf{e}_i \tag{5.7}$$

where n_k is the number of nodes belonging to cluster k. Thus, $[\mathbf{h}_k]_i = 1/n_k$ if node i belongs to cluster k and 0 otherwise.

5.1.2. Clustering by Similarity Aggregation or Relational Clustering

The clustering procedure we will discuss in this section is based on the work of François Marcotorchino (see, e.g., [46]) and Pierre Michaud (see, e.g., [50]) on the *Relational Analysis* from which it takes its name. It is also known as *Condorcet*[9] *Method* or *Voting Method* [76]. For the sake of simplicity, we will call it Relational Clustering in the remainder of this thesis. Contrary to the k-means algorithm, the number of clusters does not need to be provided as this method aims at finding an optimal number of clusters according to the Condorcet's criterion.

5.1.2.1. Relational analysis theory principles

A partition of objects [5, 11], i.e. a set of disjoint and mutually exclusive clusters (see Section 5.1), is an *equivalence relationship* $\mathcal{R}$, where $i\mathcal{R}j$ if objects i and j belong to the same cluster. Thus, partitioning a set of n objects implies to define a $n \times n$ square matrix $\mathbf{X}$ whose elements $[\mathbf{X}]_{ij} = x_{ij}$ satisfy the conditions of an equivalence relationship, namely:

- Reflexivity: $x_{ij} = 1$ if $i = j$;
- Symmetry: $x_{ij} = x_{ji}$ for all $i, j \in \mathcal{V}$;
- Transitivity: $x_{ij} + x_{jk} - x_{ik} \leq 1$ for all $i, j, k \in \mathcal{V}$;
- Binarity: $x_{ij} \in \{0,1\}$ for all $i, j \in \mathcal{V}$;

and

$$x_{ij} = \begin{cases} 1 \text{ if } i\mathcal{R}j, \text{ i. e. objects } i \text{ and } j \text{ belong to the same cluster} \\ 0 \text{ otherwise} \end{cases} \tag{5.8}$$

More generally [5], Relational Analysis applications aim at finding a relationship, e.g., an equivalence relationship, that best fits a given symmetric relationship between objects. It then relies on a square symmetric matrix $\mathbf{C}$, called *Condorcet's matrix*, whose elements $[\mathbf{C}]_{ij} = c_{ij}$

[9] Named after the French Philosopher and Mathematician, Nicolas de Caritat, Marquis de Condorcet.

represent the *agreements* or the similarity between objects i and j. This matrix will allow us to make pairwise comparisons between any objects as Relational Analysis is based on pairwise relationships between objects. Let us now explain the mathematical model underlying the Relational Clustering with an illustrative example.

5.1.2.2. Relational methodology

Let us consider the Common Neighbors matrix, introduced in Section 2.2.4, as a Condorcet's matrix. The off-diagonal elements of this matrix contain the number of adjacent nodes that two nodes have in common. The diagonal elements are the nodes degree.

This matrix can be taken as a similarity matrix and plays the role of a Condorcet's matrix. It is also a *valid* kernel (see Section 4.2) [24]. The similarity of an object, e.g., a node of a graph $\mathcal{G}$, with itself must always be greater or equal than the similarity with any other node, $c_{ii} \geq c_{ij}$ for all $i, j \in \mathcal{V}$, or more generally $c_{ij} \leq \min\{c_{ii}, c_{jj}\}$ for all $i, j \in \mathcal{V}$ [4]. This condition is satisfied as the degree of a node is always greater or equal than the number of nodes that two nodes have in common. Therefore, $\min\{c_{ii}, c_{jj}\}$ is the maximum of similarity or agreements between two nodes i and j. Their dissimilarity or *disagreements* is defined as $\bar{c}_{ij} = \min\{c_{ii}, c_{jj}\} - c_{ij}$. Two nodes will be *a priori* classified in the same cluster if $\bar{c}_{ij} < c_{ij}$, that is if they share more agreements than disagreements, following the principle of *majority rule* coming from the works of the Marquis de Condorcet on the voting systems cited in [5].

Then, the equivalence matrix $\mathbf{X}$ representing the best partitioning *consensus* of the objects is determined by maximization of the Condorcet's criterion [5]

$$C(\mathbf{X}) = \sum_{i=1}^{n} \sum_{j=1}^{n} (c_{ij}\, x_{ij} + \bar{c}_{ij} \bar{x}_{ij}) \tag{5.9}$$

with $\bar{x}_{ij} = 1 - x_{ij}$ and under the conditions of an equivalence relation (see Section 5.3.1). It is equivalent to the maximization of the following criterion under the same constraints [5]:

$$C'(\mathbf{X}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(c_{ij} - \frac{\min\{c_{ii}, c_{jj}\}}{2} \right) x_{ij} \tag{5.10}$$

The optimum can be found by linear programming only when n is not greater than a few dozens of objects [61]. For larger quantities, we have to use a heuristic algorithm as the one proposed in [5] and composed of three steps:

- Initialization step: All objects are considered randomly and only once. The first random object is a cluster on its own. Each other object will be classified in the cluster, if any, with which the number of agreements is the largest if this quantity is greater than the number of disagreements. Otherwise, the object is classified in a new cluster. The

number of agreements between a node i and a cluster c_l is equal to $\sum_{j \in c_l} c_{ij}$ and the number of disagreements is $\sum_{j \in c_l} \min\{c_{ii}, c_{jj}\} - \sum_{j \in c_l} c_{ij}$.

- Merge of clusters: Each cluster is taken sequentially and will be merged with the cluster, if any, with which the number of agreements is the largest if and only if this quantity is greater than the disagreements. The number of agreements between two clusters c_l and c_k is equal to $\sum_{i \in c_k} \sum_{j \in c_l} c_{ij}$ and the number of disagreements is $\sum_{i \in c_k} \sum_{j \in c_l} \min\{c_{ii}, c_{jj}\} - \sum_{i \in c_k} \sum_{j \in c_l} c_{ij}$. This step iterates till no more mergers occur.
- Objects reallocation: An object will be moved from its cluster to another cluster, if any, with which the agreements are the largest, larger than their disagreements and larger than the agreements with the original cluster. This step iterates till no more moves occur.

The second and third steps are iterated until $C'(\mathbf{X})$ cannot be improved anymore.

Finally, the majority principle in the Condorcet's criterion can be relaxed by introducing a factor $\gamma \in [0,1]$ such that the classification rule becomes $c_{ij} > \gamma\left(\min\{c_{ii}, c_{jj}\}\right)$ [76]. If $\gamma = 0.5$, it is equivalent to the original Condorcet's criterion. By setting a value higher than 0.5, the homogeneity of the clusters will be increased as well as the quantity returned by the algorithm. On the contrary, setting a value lower than 0.5 will lead to less homogeneous clusters but the returned quantity will be lower. We then modify Equation 4.9 accordingly:

$$C'(\mathbf{X}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(c_{ij} - \gamma\left(\min\{c_{ii}, c_{jj}\}\right)\right) x_{ij} \tag{5.11}$$

In the experimental part, we will apply this clustering method on the Common Neighbors matrix and on a Sigmoid Commute-Time kernel with parameter $\alpha = 7$ (see Section 4.5).

5.2. Community detection

Contrary to node clustering [24], community detection algorithms are often able to detect an optimal number of communities whereas, in clustering, this quantity has to be specified a priori by the user. A second difference is that in clustering, the union of all clusters forms a partition of the set of nodes $\mathcal{V}$ since each node belongs to exactly one cluster. With community detection methods, it is possible that not all nodes will be eventually classified in one of the detected communities or that no meaningful, in some sense, community structure at all will be detected [54].

5.2.1. Louvain Method

Introduced by Blondel *et al.* in [6], the so-called Louvain Method is an agglomerative greedy bottom-up and multi-level algorithm for finding densely connected groups of nodes according to

the modularity criterion (see Section 2.5.2). Thus, it is a community detection method [24]. Bottom-up methods in network science are often seen as the equivalent of agglomerative hierarchical clustering methods in data mining, multivariate statistical analysis and pattern recognition such as Ward's algorithm [14, 85]. But we should outline that hierarchical clustering algorithms only stop when all objects are in a single cluster (for an application of Ward's algorithm in node clustering see, e.g., [14, 71, 81, 82]).

As often, reaching the global maximum of modularity is computationally prohibitive; the Louvain Method was proposed as a heuristic to tackle this issue. First, we create n clusters, i.e. each node is initially a cluster on its own. Then, the algorithm proceeds in two steps [6, 24]:

- Local modularity optimization: Each node is randomly considered and tentatively moved to each adjacent cluster, i.e. a cluster that contains at least one adjacent node. The node will be eventually moved to the adjacent cluster for which the increase in modularity is the highest and strictly positive; in the case of a tie, we use a breaking rule. This step iterates, so each node will be considered several times, and ends when no more moves occur. The result is a partition of the graph into several communities.
- Nodes aggregation or coarsening: This step aims at building a new graph $\mathcal{G}'$ whose nodes are the clusters determined during the first step. The weight between each pair of new nodes will be the sum of the weights between the nodes of the corresponding clusters in $\mathcal{G}$. Furthermore, for each node in $\mathcal{G}'$, a self-loop is computed as the sum of all the weights between the nodes constituting the cluster in $\mathcal{G}$. The adjacency matrix $\mathbf{A}'$ representing the new graph $\mathcal{G}'$ is computed as $\mathbf{A}' = \mathbf{U}^{\mathrm{T}}\mathbf{A}\mathbf{U}$ with $\mathbf{U}$ being the $n \times m$ membership matrix related to graph $\mathcal{G}$ with element $[\mathbf{U}]_{ik} = 1$ if node i belongs to cluster k and 0 otherwise.

A *pass* or *run* is defined as the combination of these two steps. For each new graph $\mathcal{G}'$, we only consider the new composite nodes resulting from the aggregation step of the previous run. Runs are repeated iteratively until the algorithm stops when the modularity cannot be increased anymore, i.e. when no more nodes move during the first step or when no aggregation of the nodes is followed by a local optimization [24]. The algorithm then returns an optimal number of communities according to the maximum of modularity.

For undirected graphs, the change in modularity when moving one node i from a cluster c_k to a cluster c_l with $k \neq l$ is, as shown in [24]:

$$\Delta Q(i, c_l) = \frac{2}{\mathrm{vol}(\mathcal{G})}(\mathbf{e}_i + \mathbf{u}_l - \mathbf{u}_k)^{\mathrm{T}}\mathbf{Q}\mathbf{e}_i \tag{5.12}$$

where $\mathbf{Q}$ is the modularity matrix, $\mathbf{u}_l$ and $\mathbf{u}_k$ the membership vectors for clusters c_l and c_k, respectively, and $\mathbf{e}_i$ the i-th column of the identity matrix $\mathbf{I}$. Thus for each randomly considered

node i in step one, only the quantity $\mathbf{u}_l^{\mathrm{T}}\mathbf{Q}\mathbf{e}_i$ needs to be computed for each target cluster c_l explaining why the Louvain Method is computationally efficient even on large graphs up to dozens of millions of nodes (see, e.g., [24] for a comparison between some methods based on modularity).

Finally, the algorithm proceeds run after run by building communities that are a hierarchy of nested communities. We can then keep track of how and which communities are aggregated during the whole process, the different levels of aggregation, allowing us to detect potentially meaningful sub-communities in large communities and subsequent organizational hierarchy which can be desirable in some applications.

Let us conclude this section by giving some drawbacks related to modularity criterion optimization. In an analysis of modularity by Good, de Montjoye, and Clauset (2010) cited in [20], it is reported that several but very distinct partitions of the graph could lead to a high level of modularity close to the global optimum. A second major drawback is called the *resolution limit* [21] which prevents the detection of small-size communities having a quantity of internal edges in the order of $\sqrt{2e}$ or smaller, e being the size of the graph, that is the number of edges.

Here, we have presented the Louvain Method for modularity optimization in a very simple manner. Further refinements have been suggested to the algorithm such as for instance in [28] which will not be considered here as they go beyond the scope of this thesis.

5.2.2. Generic Louvain Method

The Newman-Girvan modularity criterion has become very popular in the field of community detection and the Louvain Method was specifically developed to efficiently maximize this partitioning quality function. However, several other partitioning criteria exist with a non-exhaustive list presented in [11] where it is also shown that some of them can be efficiently plugged into a *generic* version of the Louvain Method if they satisfy some conditions.

Let us recall from the previous section that what makes the Louvain Method so efficient is that the improvement in the modularity criterion obtained when moving one node to one community is computed locally, i.e. for the considered node and the target community [6, 11, 24]. Therefore, any quality function whose gain can also be locally computed is eligible to be efficiently plugged into the Louvain Method [11]. The generic version of the algorithm is the same as the original one with its two steps; only the computation of the gain in the quality function in step one is different [11, 24].

As show in [11], this is possible for all quality functions that are *linear* functions of the equivalence matrix $\mathbf{X}$ if we consider Relational coding (see Section 5.1.2.1). $F(\mathbf{X})$ is a linear partitioning quality function if it can be written as:

$$F(\mathbf{X}) = \sum_{i,j \in \mathcal{V}} \varphi(a_{ij}) x_{ij} + K \tag{5.13}$$

where $\varphi(a_{ij})$ is a function of the original adjacency matrix **A** and K any constant depending on the same adjacency matrix[10]. While the linearity condition is sufficient to this end, it is not necessary as some nonlinear but *separable* quality functions could also be efficiently incorporated [11]. A function $F(\mathbf{X})$ is separable if it can be written as a scalar product of $\varphi(\mathbf{A})$ and $\psi(\mathbf{X})$:

$$F(\mathbf{X}) = \sum_{i,j \in \mathcal{V}} \varphi(a_{ij}) \psi(x_{ij}) + K \tag{5.14}$$

with $\psi(\mathbf{X})$ being a function of the equivalence matrix. Whether the gain in the quality function when moving one node to another community can be computed locally or whether it must be computed globally and thus not being suitable to the Louvain Method will depend on this latter function [11].

[10] See Equation 2.7 where the Kronecker delta δ is used instead of entries of the equivalence matrix **X** and where $\varphi(a_{ij}) = a_{ij} - \frac{a_{i\bullet} a_{\bullet j}}{a_{\bullet\bullet}}$ showing that Newman-Girvan modularity satisfies the linearity condition [11].

6. MANAGERIAL IMPLICATIONS

As mentioned in the introduction, graph analysis, community detection and clustering techniques can have concrete applications in various fields. Therefore, we will present in this chapter some concrete examples for the managerial world to conclude the first part of this thesis.

6.1. Marketing

We refer the interested reader to [77] for a comprehensive review of some clustering applications in Marketing such as, amongst others, segmentation, whereas in this thesis we will present two contemporary Marketing strategies: *Viral Marketing* and *recommendation systems*.

Viral Marketing aims at the fast spreading of marketing messages among a large number of individuals at low cost by using online social medium [80, 90]; this expression being an analogy of the spreading schemes of pathologies inside networks. In this perspective, the structure of a social network has an impact on the selection of initial individuals as *seeds*, that is the selection of the most *central* individuals in a network for an effective spreading. Several measures of *centrality* are investigated in [39] for spreading simulation purposes in different networks, with the basic out-degree (see Section 2.2.2) centrality achieving competitive results in comparison to other more complex measures. Nevertheless, considering only the structure of a network may not be sufficient when, e.g., diffusion time matters. Nodes attributes such as *human dynamics*, defined by Barabási (2005) as the time for an individual to forward the message according to its perceived priorities and cited in [90], should also be taken into account in addition to the network structure.

Online recommendation systems [32] provide people with suggestions for future purchasing opportunities. Some of them build individual recommendations while other target groups or communities of users from online social medium. But the existence of communities does not automatically imply that all the members share enough similar interests to make this marketing strategy efficient. Therefore, in [32], a measure is developed to compute similarity among a community rather than conventional pairwise similarity measures.

6.2. Logistics and Supply Chain Management

Graph and data mining can be used at all levels of a supply chain. For instance, suppliers selection is a crucial decision as raw material and components parts purchases represent the main costs for a company [33]. Besides this quantitative aspect, other qualitative considerations

must be taken into account to meet clients' expectations at the lowest cost. Therefore, still in [33], the authors introduce a method based on the joint use of fuzzy distances and hierarchical clustering to determine a ranking among potential suppliers.

Clustering methods can also be considered regarding transportation networks and vehicle routing strategies. For instance, in [49] where freight transportation by trucks is analyzed, the Louvain Method is used to identify clusters of demand that should be served in the same tour. Truck vehicles filled to capacity and backhauls (i.e., non-empty returns after deliveries) mean lower unitary costs since any empty trip is unprofitable for truckload companies and, because of externalities, for society.

Analysis of transportation networks and tackling the vehicle routing problem can also be valuable in the case of natural disasters for the design of emergency and humanitarian operations. In [8], a comparison of the urban road network is made before and after a seismic disaster, with edges being streets and nodes being street intersections. It allows the identification of the largest connected component (see Section 2.2.3), a subset of the network before the disaster which contains the largest quantity of undamaged streets, and smaller isolated components, i.e. areas that cannot be reached anymore from the main component because of damaged or blocked roads.

Another concrete example is the assessment of a supply chain *resilience* which, according to the most commonly used definition in the literature coming from Ponomarov and Hocomb (2009) and cited in an extensive review on this topic [37], is "the adaptive capability of the supply chain to prepare for unexpected events, respond to disruptions and recover from them by maintaining continuity of operations at the desired level of connectedness and control over structure and function". In [38], the North American power grid network is analyzed to detect nodes that could be at risk of causing cascading failures and/or serious damage to the network efficiency. Such studies can be applied to various networks in the field of Supply Chain Management and eventually aims at triggering proactive reactions to modify the network in order to improve its resilience or to deploy countermeasures accordingly.

6.3. Accounting and Finance

An important topic in Accounting is the detection of misrepresentations in the financial statements, either due to fraud or error. Clustering techniques and graph mining approaches can be used to detect such *anomalies* in networks of accounting items [12].

What concerns financial networks, resilience is also a matter of importance. This topic, as well as financial networks, was already studied prior to the financial crisis of 2007-2008. As stated in [2], the conclusions slightly differ between the different articles on this subject. In this recent

article, it is found that critical nodes that could compromise the integrity of the network are *hubs* (i.e., nodes with a high degree) whose edges are mostly *contagious links* defined as links with financial counterparties whose default could not be absorbed by the capital of the institution.

Regarding assets allocation and portfolio management, graph-based and clustering approaches can complete traditional methods, such as the Markowitz model, to simplify the process of assets selection and to improve their efficiency [13, 41, 52, 89]. They can also be considered as alternatives to them to tackle the restrictive or invalid assumptions of some models as well as the required inputs that cannot be properly estimated because of the complex and noisy feature of financial data [7, 17, 18, 60].

A contemporary issue with financial data [10], yet not specific to them, is their large dimensionality (e.g., several attributes describing one transaction in financial markets) as well as their size (considering the large amount of transactions per second for one single stock) which classifies them as one instance of *tick data* in the paradigm of *Big Data*. One would like to store this data in an efficient way, such that the required space of storage is reduced as through data compression, but without slowing down the access. Several clustering algorithms can be used to decompose an original matrix in several sub-matrices such that their cumulated size is smaller than the size of the original one.

6.4. Business Process

In the field of Project Management, risks are numerous, complex, dynamic with some of them being interrelated [48]. Stating that these interconnections between risks are often neglected in traditional risk management methods, the authors therefore propose to use clustering methods to identify groups of highly interrelated risks while the interdependencies between clusters are minimized. This approach has the advantage of reducing the dimensionality of the risks making the problem more manageable. It also promotes collaboration and cooperation between individuals that would have never met during the usual course of a project since a cluster is likely to contain risks belonging to different fields of competences.

Cooperation and collaboration between individuals inside an organization can also be studied by building social networks from the *event logs* [72] which are records of business events stored by information systems. Building such networks from other sources, such as e-mail messages exchanges, could lead to more noisy data as they are not always work-related content. When processes are not fully automated, analysis of event logs is invaluable to discover gaps between what should be, the guidelines and the actual reality which is also what makes such data difficult to analyze. Indeed, when performing Business Process Analysis, one has to deal with complex, dynamic, noisy and incomplete data [67, 72]. This is particularly true with sensitive

organizations such as healthcare organizations which are furthermore multidisciplinary environments with ad-hoc processes [67].

Regarding the financial advisory process and more specifically its simplification, in [51] a *Case-Based Reasoning* approach is proposed to meet client's needs and constraints. The idea is to reuse experience gathered with previous similar clients (according to cosine similarity measure based on a set of features) and to retrieve the corresponding effective solutions that were offered. The latter are then clustered with a k-means algorithm to reduce the quantity of possible proposals, with the final propositions being the centroids of the k clusters.

Part II.

Experiments

7. DATASETS DESCRIPTION

In this chapter, we will present the datasets on which the clustering methods will be tested. For more details, we refer the interested reader to the cited references. The quantity of datasets is relatively small because of the scarcity of datasets including the class labels of the nodes. Nevertheless, datasets including nodes label can be created artificially. Among our 16 datasets, we can identify three different categories. The first one is composed of five datasets, the smallest ones, which could be loosely referred to as real-world social networks. The second category includes two artificial datasets generated by the Lancichinetti-Fortunato-Radicchi (LFR) model. The last one contains nine subsets derived from the famous Newsgroup dataset. All the datasets must be considered as of small size, the largest containing only 999 nodes (see Appendix A, Table A.1).

- **Football dataset**[11]**:** Compiled by Girvan and Newman and introduced in [29], it represents the schedule of games between the 115 American college football teams, thus, teams are the nodes and games are the edges. Teams are divided into 12 conferences, the true classes, of 8 to 12 teams, and games involving teams of the same conference are more frequent than inter-conferences games.
- **LFR datasets:** Two 600-node datasets containing respectively 3 and 6 classes and synthetically generated by the LFR model [43] to mimic properties found in real-world graphs by allowing the user to control most of the properties of a network. Nevertheless, artificial datasets do not contain the noise of real datasets and show only small correlation with properties observed in real data [64].
- **Newsgroup dataset**[12]**:** Nine samples were built from the famous Newsgroup dataset which contains around 20,000 unstructured documents coming from 20 different newsgroups. Each discussion group is supposed to be designed for a specific topic. Among the 9 samples, three of them contain 2 topics each, three contain 3 topics and the last three contain 5 topics. As each sample contain 200 documents per topic, we have 3 samples of around 400 documents each, 3 of 600 and 3 of 1,000. Finally, documents were preprocessed to reduce the high dimensionality problem. The nodes are documents and the links are the mutual information understood as terms shared between documents as described in [83] making this network dataset weighted. As some topics could be related (see Appendix A, Figure A.1), these datasets could potentially contain overlapping classes and thus be fuzzy [83].

[11] Available from https://networkdata.ics.uci.edu/data.php?id=5.
[12] Available from http://people.csail.mit.edu/jrennie/20Newsgroups/.

- **Political books dataset**[13]**:** Compiled by Valdis Krebs [73], this dataset represents the purchase pattern of best-selling political books on the retail website Amazon and pairs of them are linked if they are *frequently bought by the same customer*. The network is unweighted, the nodes consist of 105 books that were eventually divided by Newman into three groups according to their stated or apparent political orientation [53]: Conservative, neutral – unidentified trend or bipartisan – and liberal literature. The analysis of such network aims at detecting the political trends among the citizens of a country as opinion pools do. Contrary to the latter, such approach is not invasive and objectivity is not questionable [73].
- **School dataset**[14]**:** This dataset contains 236 nodes consisting of teachers and schoolchildren from a French primary school (see [74] for more details) with two variants:
  - The first one with the different classrooms of each educational level kept separated, thus containing 11 groups in total, 1 of teachers and 10 of schoolchildren;
  - The other one with the classrooms of the same education level grouped together, thus containing only 6 groups, 5 of schoolchildren and 1 of teachers.
- **Zachary dataset:** A famous unweighted dataset depicting the relationships between the members of a karate club studied over a period of two years by the Anthropologist Wayne W. Zachary in the 1970's. The nodes are 34 regular members of the club and pairs of them are linked if they interact outside the club. Studying this network leads to understanding the conflicting forces at work and the latent ideological division among the members and why the club eventually split in two, the "natural" groups of this dataset. We refer the interested reader to [86] for more details of this ethnographic study and its methodology.

[13] Available from http://orgnet.com.
[14] Available from http://www.sociopatterns.org/datasets/primary-school-cumulative-networks/.

8. QUALITY MEASURES

In this chapter, we will introduce two metrics that will be used to assess the quality of the partitions returned by our clustering methods on the datasets introduced in the previous chapter.

When carrying out an *internal evaluation* [36] of a partition, we use information or data that was used during the clustering process in order to assess how well a clustering algorithm achieved a targeted objective such as the optimization of a criterion. Internal evaluation gives no clues about whether a clustering method returns more meaningful partitions than another one. On the other hand, with *external evaluation* [36], clustering results are assessed by matching clustering partitions to a benchmark consisting of a priori and independent external information, that is information that was not used during the clustering process and is therefore not data-dependent. This could be expectations based on the theory, the knowledge of an expert or, in our case, the ground truth partition of the nodes. We will use two external measures that will allow us to make this comparison: The *Adjusted Rand Index* and the *Normalized Mutual Information*.

Here, only the quality of the partitions returned by the algorithms (i.e. their respective effectiveness) will be assessed. Even if the performance in terms of computational complexity, that is the efficiency or scalability of the algorithm, is also a matter of crucial importance, it will not be discussed in depth in this thesis and not in this chapter. Moreover, an overall evaluation of an algorithm should also take into account the number of parameters that have to be specified a priori by the user [16, 44].

8.1. Adjusted Rand Index

The *Rand Index*, initially proposed by Rand in [66], is a pair-counting method which evaluates the quality of the clustering partition with respect to the ground truth partition by considering the $\binom{n}{2} = n(n-1)/2$ possible pair of nodes. We will say that a pair of nodes is a true positive (TP) if both nodes belonging to the same natural class were classified into the same cluster and a true negative (TN) if two nodes that do not have the same natural membership were classified into two different clusters. On the other hand, we can encounter two types of errors, namely the false positive (FP) and the false negative (FN), when two nodes are respectively falsely classified in the same cluster and in separated clusters. The Rand Index is the percentage of correct classifications [45]:

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{8.1}$$

The main issues with the Rand Index are that it does not take a constant value when comparing two random partitions and it tends to 1 when we compare two partitions containing a large number of clusters even if they completely differ. Therefore, Hubert and Arabie [35] proposed the *Adjusted Rand Index* (ARI, henceforth) which can be computed from a *confusion matrix* with rows corresponding to the true classes and columns to the clusters returned by the algorithm. Considering the known set of true groups $\Omega = \{\omega_1, \dots \omega_l \dots, \omega_p\}$ and a clustering partition $\mathcal{C} = \{c_1, \dots c_k \dots, c_m\}$, the related ARI is computed as

$$\frac{\sum_{l,k}\binom{n_{lk}}{2} - \left[\sum_l \binom{n_{l\bullet}}{2} \sum_k \binom{n_{\bullet k}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_l \binom{n_{l\bullet}}{2} + \sum_k \binom{n_{\bullet k}}{2}\right] - \left[\sum_l \binom{n_{l\bullet}}{2} \sum_k \binom{n_{\bullet k}}{2}\right] / \binom{n}{2}} \tag{8.2}$$

with n_{lk} being the number of nodes belonging to true class l and cluster k, $n_{l\bullet}$ and $n_{\bullet k}$ respectively being the number of nodes in class l and cluster k. The ARI has the form of an index with constant expected value 0 and maximum value of 1:

$$\frac{\text{index} - \text{expected index}}{\text{maximum index} - \text{expected index}} \tag{8.3}$$

Moreover, this measure is recommended in [69, 84] and used in [14, 77].

8.2. Normalized Mutual Information

Still given the ground truth partition $\Omega = \{\omega_1, \dots \omega_l \dots, \omega_p\}$ and a clustering partition $\mathcal{C} = \{c_1, \dots c_k \dots, c_m\}$, the *Normalized Mutual Information* (NMI, hereafter) based on information theory is computed from a confusion matrix (see Section 8.1) as follows [16, 45]:

$$NMI(\mathcal{C}, \Omega) = \frac{I(\mathcal{C}, \Omega)}{[H(\mathcal{C}) + H(\Omega)]/2} \tag{8.4}$$

where

$$I(\mathcal{C}, \Omega) = \sum_l \sum_k \frac{|c_k \cap \omega_l|}{n} \log \frac{n|c_k \cap \omega_l|}{|c_k||\omega_l|} \tag{8.5}$$

is the *Mutual Information* between a cluster k and a true class l which tells us how much information the knowledge of the cluster membership of a node gives us about the true class membership. It is equal to 0 when clusters are randomly distributed among classes. But the Mutual Information increases with the number of clusters and reaches a maximum for $k = n$, n being the number of nodes. Thus, $I(\mathcal{C}, \Omega)$ is normalized by the denominator $[H(\mathcal{C}) + H(\Omega)]/2$ where $H(\cdot)$ is the *entropy*, defined – if we consider the estimators of the probabilities – as follows [45]:

$$H(\Omega) = -\sum_l \frac{|\omega_l|}{n} \log \frac{|\omega_l|}{n} \tag{8.6}$$

$$H(\mathcal{C}) = -\sum_k \frac{|c_k|}{n} \log \frac{|c_k|}{n} \tag{8.7}$$

As $[H(\mathcal{C}) + H(\Omega)]/2$ is always slightly larger than $I(\mathcal{C}, \Omega)$, the NMI is hence always ranked between 0 and 1 and comparison between different numbers of clusters is possible. This metric is often used when comparing clustering methods in terms of their performance [20] and also used in [14, 77].

9. EXPERIMENTAL PROCEDURE

9.1. Algorithms implementation

Algorithms required for the purpose of this thesis come from different sources. We implemented the non-existing ones using R programming language and software environment, version 3.2.3. As some algorithms were already implemented in Matlab language, we converted them into R; according to our needs, some of them were subsequently modified. We also used some functions that were already implemented in R packages without any modifications.

Let us recall that the Louvain Method returns an optimal number of clusters corresponding to a maximum of modularity. The algorithm was then modified so that it will stop before reaching the maximum level of modularity if at some point the current number of clusters is equal to the true number. It will be coerced into continuing if the optimal number of clusters is greater than the true quantity and clusters that will be subsequently merged are those leading to the lowest decrease in modularity.

9.2. Parameter tuning

9.2.1. Sigmoid Commute-Time kernel

We must first set the value of the parameter α of the sigmoid transformation before computing the Sigmoid Commute-Time kernel $\mathbf{K}_{\mathrm{CT}}^{\mathrm{S}}$ (see Equation 4.8). In a first time, we will set the value of α to 7, as this value proved to be adequate for our datasets in [14, 82]. Then in a second time, we will use an optimized value per dataset. These values were determined in [77] by running 50 times on each dataset the kernel-based k-means with different values of α ranging from 10^{-6} to 10^{3}. Per dataset, the value leading to the highest average NMI score is recorded as the optimal parameter (see Table 9.1).

Table 9.1: Datasets and corresponding optimal parameter α (if different than 7) for the computation of the Sigmoid Commute-Time kernel (adapted from [77]).

	Optimal value for α		Optimal value for α
zachary	8	news_2cl_1	5
foot	3	news_2cl_2	4
polbooks	1,000	news_2cl_3	10
school_11	9	news_3cl_2	8
school_6	10	news_3cl_3	10
		news_5cl_1	10
		news_5cl_2	3
		news_5cl_3	5

9.2.2. Relational Clustering

When coercing this algorithm into returning a partition containing the true number of clusters, if the optimal (for γ = 0.5) number of clusters returned by the method is larger, we will have to decrement the value of the parameter γ from 0.5 by steps of 0.01 (see Section 5.1.2.2). Clearly, we proceed in the opposite way if the optimal number of clusters is lower than the ground truth quantity.

9.3. Clustering procedure

The starting point is the adjacency matrix **A** corresponding to each one of the 16 datasets. From this matrix, several other matrices must be first computed to be further used as inputs of the clustering and community detection algorithms.

For the distance-based k-means (see Section 5.1.1.1):

- The Euclidean Commute-Time distance matrix $\mathbf{\Delta}_{\mathrm{ECT}}$ (see Section 3.1.2);
- The Euclidean Corrected Commute-Time distance matrix $\mathbf{\Delta}_{\mathrm{ECCT}}$ (see Section 3.1.3).

For the kernel-based k-means (see Section 5.1.1.2):

- The Sigmoid Commute-Time kernel $\mathbf{K}_{\mathrm{CT}}^{\mathrm{S}}$ with α = 7 (see Section 4.5);
- The Sigmoid Commute-Time kernel $\mathbf{K}_{\mathrm{CT}}^{\mathrm{S}}$ with optimized parameters (see Section 8.2.1).

For the Relation Clustering (see Section 5.1.2):

- The Common Neighbors matrix **N** (see Section 2.2.4);
- The Sigmoid Commute-Time kernel $\mathbf{K}_{\mathrm{CT}}^{\mathrm{S}}$ with α = 7 (see Section 4.5).

For the Louvain Method (see Section 5.2.1): The Modularity matrix **Q** (see Section 2.2.5).

Let us make some important remarks:

- For the computation of the Common Neighbors matrix **N**, the graphs must be unweighted. Therefore, for weighted graphs, we will only keep the edges corresponding to the thirty upper percentiles of the weights, set them to 1 to make the graph unweighted and rebuild the adjacency matrices accordingly.
- As all the clustering or community detection methods are non-deterministic. We will then launch each of them 50 times on all datasets and store the average ARI and NMI scores per algorithm and dataset.
- Regarding the Relational Clustering, there is no parameter value γ for which we can be sure that the returned number of clusters will be exactly equal to the true number. As stated in Section 9.2.2, starting from a value of 0.5, we will have to either decrease or increase the parameter value by steps of 0.01. At each step, Relational Clustering is launched 50 times and the procedure is stopped when we have gathered the 50 first partitions containing a number of clusters equal to the ground truth quantity.

- However, the optimal number of clusters returned by the Relational Clustering can be cumbersome leading to large execution times. To tackle this issue, we will start from a parameter value of 0.01 and increment it by steps of 0.01. As previously, at each step the algorithm will be launched 50 times and the procedure will be stopped if at some step all the 50 partitions returned by the algorithm contain a number of clusters strictly greater than the ground truth quantity. ARI and NMI scores will be computed from the last 50 partitions containing the ground truth number of clusters.

9.4. Performance comparison

To achieve reliable performance comparison between our algorithms, we will follow the procedure preconized by Demšar in [15] and used in [14]. Once we have run all the algorithms on all the datasets, we compute the average ARI and NMI scores for each dataset and algorithm. In order to summarize these values, we will execute a *Friedman test* (see Section 9.4.1) to determine whether at least one algorithm performs significantly better than the other ones. If it is actually the case and in order to identify the method(s) that is (are) significantly different, we will perform a *Nemenyi post hoc test* (see Section 9.4.2) to compare each method to each other. Finally, if required, we could also perform a multiple comparison against a control method called *Bonferroni-Dunn test* (see Section 9.4.3). All the tests will be performed at a level of confidence α = 0.05.

9.4.1. Friedman test

The Friedman test named after Milton Friedman [26, 27] is a non-parametric test equivalent of the repeated-measures ANOVA (see [26] for a comparison between ANOVA and Friedman test) and is computed as follows. Considering each quality metric, i.e. the average ARI or NMI score for each dataset, a rank is assigned to each of the k algorithms according to its performance on a dataset. Algorithms are ranked by getting a specific amount of points: The most effective one on the dataset will get the maximum of k points while the less effective will get only one point. Then the average rank sum by algorithm $\bar{R}_j$ is computed over all the D datasets.

The null hypothesis of the Friedman's test is that all the average rank sums are not statistically different at a level of confidence α and the algorithms exhibit the same performance. The Friedman's test statistic value is then computed as follows [15, 79]:

$$\chi_F^2(k-1) = \frac{12D}{k(k+1)}\left[\sum_{j=1}^{k}\bar{R}_j^2 - \frac{k(k+1)^2}{4}\right] \tag{9.1}$$

and is approximately distributed according to a chi-square distribution with $k-1$ degrees of freedom when k or D is "large enough", say $k > 5$ or $D > 5$ [79] or $kD \geq 30$ [47]. Otherwise, tables with exact critical values must be used (see, e.g., [47]). If the Friedman test statistic value

is greater than the critical value of rejection (i.e. $\chi_F^2(k-1) > \chi_{k-1;\,\alpha}^2$), we reject the null hypothesis in favor of the alternative hypothesis. Friedman test can be directly performed with the function `friedman.test` implemented in the R package `stats` [64].

9.4.2. Post hoc Nemenyi test

If the Friedman test shows significance, we can carry out a post hoc test according to Nemenyi (1963) cited in [15] to make $k(k-1)/2$ comparisons between all algorithms against each other. The average rank sums of two methods, as computed for the Friedman test, are significantly different if they differ by at least a critical difference CD:

$$\left|\bar{R}_i - \bar{R}_j\right| > CD = q_{k;\alpha}\sqrt{\frac{k(k+1)}{6D}} \tag{9.2}$$

where critical values $q_{k;\alpha}$ are based on the Studentized range statistic divided by $\sqrt{2}$ (see Table 9.2). Post hoc Nemenyi test can be performed through `posthoc.friedman.nemenyi.test` function implemented in the package `PMCMR` [62] or manually with the critical values given in Table 9.2.

Table 9.2: Critical values for the two-tailed Nemenyi test (adapted from [15]).

k	2	3	4	5	6	7	8	9	10
$q_{k;0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164

9.4.3. Bonferroni-Dunn test

The Bonferroni-Dunn test [19] is used to compare all methods again a control method. From Table 9.3, one can notice that this test is more powerful than the Nemenyi test as the latter adjusts the critical value for $k(k-1)/2$ comparisons while only $k-1$ are made when comparing with a control method [15]. The only difference with the Nemenyi test is therefore the critical values. This test should be considered when comparing a newly introduced method against existing ones [15].

Table 9.3: Critical values for the two-tailed Bonferroni-Dunn test with k including the control method (adapted from [15]).

k	2	3	4	5	6	7	8	9	10
$q_{k;0.05}$	1.960	2.241	2.394	2.498	2.576	2.638	2.690	2.724	2.773

10. RESULTS AND DISCUSSION

In this chapter, we will present, discuss and compare the levels of performance of the different methods of clustering and community detection.

Table 10.1: Acronyms that will be used in this chapter and their corresponding concept/method.

Concepts/methods	Related abbreviation
Euclidean Commute-Time distance	ECT
Euclidean Corrected Commute-Time distance	ECCT
Distance-based k-means	DBKM
Free Energy distance†	FE
Sigmoid Commute-Time kernel-based k-means ($\alpha = 7$)	KBKM
Sigmoid Commute-Time kernel-based k-means (optimized values of α)	KBKMopt
Louvain Method based on Newman-Girvan Modularity	LM
Relational Clustering on Common Neighbors matrix	RCCN
Relational Clustering on Sigmoid Commute-Time kernel ($\alpha = 7$)	RCSCTK

†The Free Energy distance metric will be introduced and discussed in Section 10.2.1

10.1. First research question: Louvain Method versus Relational Clustering

It makes sense to compare these two methods as they aim at returning an optimal number of clusters according to their respective criterion. First, we will discuss their relative performance on this optimal quantity and, in a second time, they will be compared on their performance when coerced into returning a partition containing the true quantity of clusters.

10.1.1. Optimal number of clusters

First, we can observe that Relational Clustering on the Common Neighbors matrix (RCCN) tends to return a large optimal quantity of clusters (see Appendix B, Table B.1) with a large proportion containing only one or two nodes. As we could observe with our datasets, even nodes inside communities can be somewhat sparsely connected to each other. This explains why working with the common neighbors leads to this outcome. The ARI is penalized by this large amount of clusters whereas, in comparison, the NMI score is larger as the method was able to extract some aspects of the community structure from the datasets [16]. Indeed, we could notice that we were able to identify, partly or totally, communities which have a sufficiently large density of internal edges. Moreover, in informal tests on graphs whose communities consist of cliques (see Section 2.1) sparsely connected to each other, the method was able to correctly detect all of them making us confident about the validity of its implementation.

We run this method only 10 times on some datasets (see Appendix B, Table B.1) because of the prohibitive execution time. This drawback might come from a large number of clusters to process and/or from our implementation of this clustering algorithm.

On the other hand, the joint use of Relational Clustering and the Sigmoid Commute-Time kernel with $\alpha = 7$ (RCSCTK) leads to a much smaller optimal amount of clusters but still greater than the true number for a majority of the datasets: For three datasets, the average optimal quantity of clusters is smaller and the true quantity was exactly reached only for one dataset (see Appendix B, Table B.2). The execution time of the algorithm is this time tractable although, at this point, this fact cannot exclude the perfectibility of the algorithm implementation.

Finally, among the three methods discussed in this section, the Louvain Method (LM) exhibits optimal quantities of communities that are the closest to the true quantities (see Appendix B, Table B.3). Furthermore, it seems that it also outperforms the other methods regarding the ARI and NMI scores (see Table 10.2). Let us now validate and summarize these findings.

Table 10.2: Average rank sums (the higher, the better according to rank coding for Friedman test) for each method and each partition quality measure.

	RCCN	RCKSCT	LM
$\bar{R}_j$(ARI)	1.4375	2.0625	2.5000
$\bar{R}_j$(NMI)	1.6250	1.8750	2.5000

We will first compute the *average absolute error* [14], that is the average absolute difference over all datasets between the optimal number of classes returned by a method and the true number of classes. It yields 136.6429 for the Relational Clustering on the Common Neighbors matrix and 11.6242 on the Sigmoid Commute-Time kernel ($\alpha = 7$). The Louvain Method has the lowest average absolute error with 2.8447.

Regarding the performance based on the ARI and NMI scores, the Friedman test ($\chi_F^2(2) = 9.125$ (p-value = 0.0104) and $\chi_F^2(2) = 6.5000$ (p-value = 0.0388), respectively) shows that at least one method has an average rank significantly different from the others. It is therefore meaningful to carry out a Nemenyi post hoc test.

From the results of the Nemenyi test based on the average ranks drawn from the ARI, we can say that the Louvain Method, which has the highest average rank, is significantly different (p-value = 0.0075) in terms of performance from the Relational Clustering based on the Common Neighbors, on average the lowest ranked method. The same conclusion holds for the NMI scores (p-value = 0.0360). However, Relational Clustering based on the Sigmoid Commute-Time kernel does not significantly differ to the Louvain Method (p-value = 0.4310) nor to Relational Clustering based on Common Neighbors (p-value = 0.1805). This means that the available data is not sufficient to draw any conclusion about this method since it would be a statistical nonsense to say that it is equivalent to the two other methods [15]. However, we can notice that RCSCTK

performs better than RCCN on 13 datasets out of 16 but is defeated by LM on 13 datasets, the same being observed for both ARI and NMI scores.

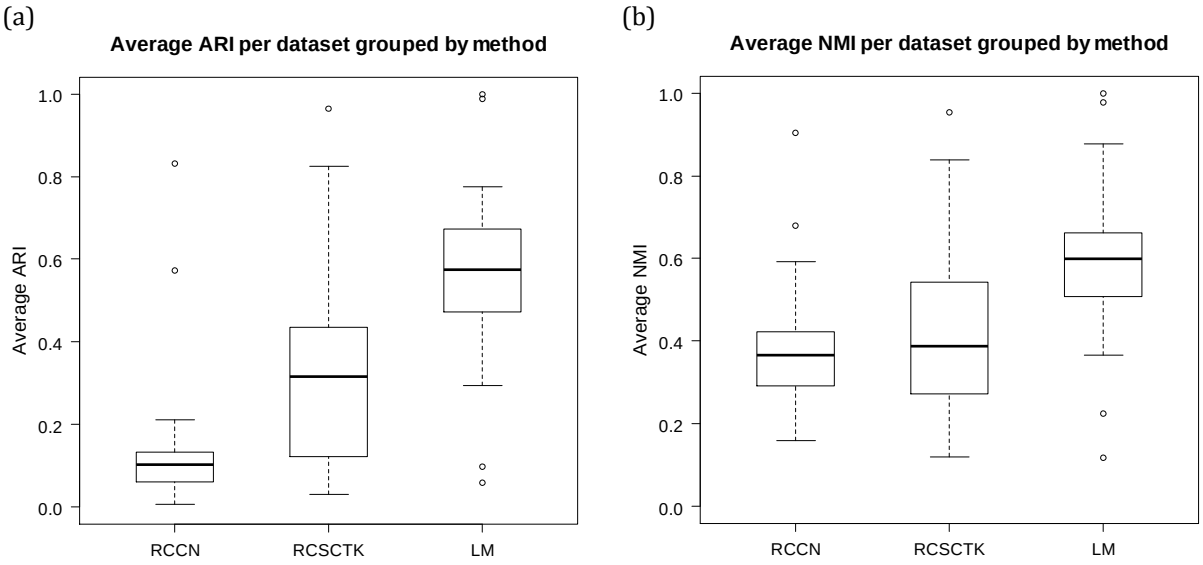

Figure 10.1: Boxplots of the average (a) ARI and (b) NMI scores per dataset when comparing the ground truth partitions and the optimal partitions as returned by the Relational Clustering (RCCN and RCSCTK) and the Louvain Method (LM).

10.1.2. Ground truth number of classes

When coercing the algorithms into returning the ground truth number of clusters, the modified Louvain Method has now the lowest average rank (see Table 10.3). Nevertheless, the Friedman test indicates no significant difference in performance between the three methods for the ARI and NMI scores at $\alpha = 0.05$ ($\chi_F^2(2) = 1.5556$, p-value = 0.4594 and $\chi_F^2(2) = 2.625$, p-value = 0.2691, respectively).

10.1.3. Conclusion

We can now discuss some findings of interest we made through this first research question. Let us first stress that in order to obtain partitions containing a number of clusters or communities equal to the true number of classes, we had to adapt the algorithms accordingly (see Sections 9.1 and 9.2.2) and our results are thus dependent on this way of proceeding.

Considering the optimal number of clusters, the Louvain Method returns a quantity that is on average the closest to the true number of classes. It also displays the best average ARI and NMI scores even if it cannot be considered as performing significantly better than the joint use of Relational Clustering and the Sigmoid Commute-Time kernel with $\alpha = 7$.

The Relational Clustering based on Common Neighbors seems to be dependent on the topology of the communities but proved to be effective in detecting, partly or totally, communities that have a relatively high internal density of edges. It exhibits the lowest average rank but we cannot

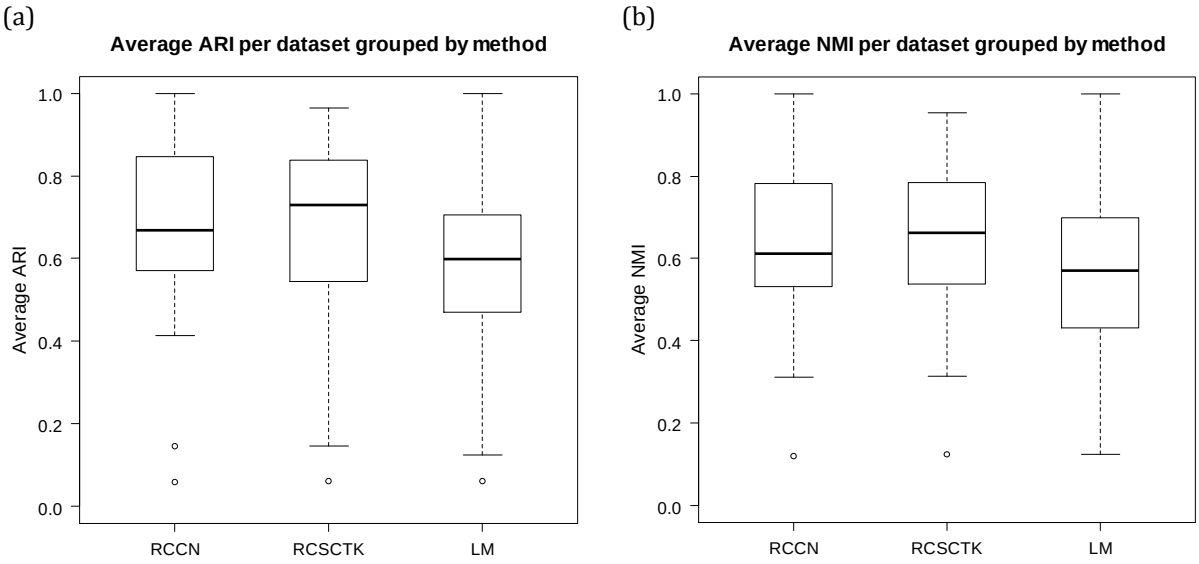

Figure 10.2: Boxplots of the average (a) ARI and (b) NMI scores per dataset when comparing the ground truth partitions and the partitions containing the true number of clusters or communities as returned by the Relational Clustering (RCCN and RCSCTK) and the Louvain Method (LM).

Table 10.3: Average rank sums (the higher, the better according to rank coding for Friedman test) for each method and each partition quality measure.

	RCCN	RCKSCT	LM
$\bar{R}_j$(ARI)	2.0000	2.2188	1.7813
$\bar{R}_j$(NMI)	1.9375	2.3125	1.7500

conclude that its performance is significantly inferior to the Relational Clustering based on the Sigmoid Commute-Time kernel with $\alpha = 7$.

Finally, if we consider the detection of the true number of clusters or communities, the Louvain Method has now the lowest average ARI and NMI scores while the Relational Clustering has the highest ones. However, according to Friedman test we cannot say that one particular method has a significantly different level of performance at $\alpha = 0.05$.

10.2. Second research question: Overall comparison of the methods when targeting the true number of classes

In this second research question, we will compare the performance of different methods when partitioning the datasets into the ground truth number of clusters. Therefore, besides the Relational Clustering and the Louvain Method previously discussed, we will also consider the distance-based and kernel-based variants of the k-means algorithm. We will first determine which distance metric performs best on the datasets when used with the distance-based k-means. We will also highlight our choice regarding the kernel to be used with the kernel-based k-means.

10.2.1. Distance-based k-means

We would like to first tackle a potential issue encountered in [77] regarding the performances of the distance-based k-means algorithm when used with the Euclidean Commute-Time and the Corrected Commute-Time distances. In [77], the recorded measures were very low for most of the datasets in comparison to the Free-Energy distance (see, e.g., [24, 77] for a definition of this distance metric) but as this problem was beyond author's scope, it was not further investigated.

From our results, we can only confirm the very low classification scores for the Euclidean Commute-Time distance as stated in [77] and postulate that this distance is not meaningful for most of the datasets without any better explanation. On the contrary, we record different levels of performance for the Euclidean Corrected Commute-Time distance. Moreover, it seems to be equivalent in terms of performance to the Free-Energy distance[15] over all selected datasets (see Figure 10.3 and Appendix D, Table D.1).

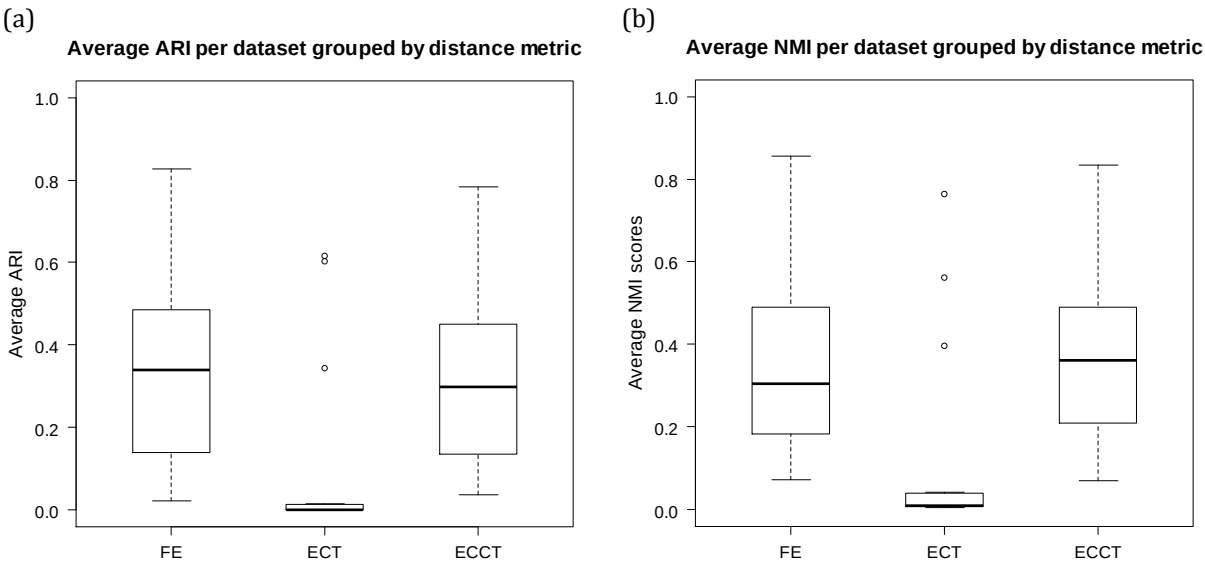

Figure 10.3: Boxplots of the average (a) ARI and (b) NMI scores per dataset for the partitions returned by the distance-based k-means algorithm with three different distance metrics: Free Energy (FE), Euclidean Commute-Time (ECT) and Euclidean Corrected Commute-Time (ECCT) distances.

Solely using visual analysis, we could assume different levels of performance from the k-means algorithm based on three different distances. Let us see if this intuition is confirmed by the Friedman test.

The Euclidean Corrected Commute-Time distance has the highest average rank for the ARI whereas the Free-Energy distance is on average ranked first according to the NMI scores. The Friedman test indicates significance, respectively $\chi_F^2(2) = 16.125$ (p-value = 0.0003) and $\chi_F^2(2) =$

[15] ARI and NMI scores for the Free-Energy distance were directly extracted from [77] and thus were not re-computed.

16.625 (p-value = 0.0002). As the p-values are smaller than 0.01 for both performance indicators, we can conclude with a high level of confidence that the methods are not equivalent in terms of performance on these datasets and it is meaningful to perform multiple comparisons tests in order to identify the differences between them.

Table 10.4: Average rank sums for each method and each partition quality measure (the higher, the better according to Friedman test coding).

	FE	ECT	ECCT
$\bar{R}_j$(ARI)	2.5000	1.1875	2.3125
$\bar{R}_j$(NMI)	2.2500	1.1875	2.5625

According to the Nemenyi post hoc test, for both ARI and NMI scores, the Free-Energy distance and the Corrected Commute-Time distance differs very significantly to the Commute-Time distance (p-values < 0.01) while there is no significant difference between them (p-values = 0.8564 and 0.6505 for ARI and NMI scores, respectively). Based on these results, we can conclude that, on our datasets, there is no significant difference in performance between the Free-Energy and the Euclidean Corrected Commute-Time when used with the distance-based k-means. As the latter two distances have the highest ranks and as there is no significant difference between them at a level of confidence α = 0.01, we decide to keep the joint use of the DBKM and Euclidean Corrected Commute-Time distance for further comparisons with the other considered methods[16].

10.2.2. Kernel-based k-means

In the previous section, we observed that the Euclidean Commute-Time distance performs poorly in comparison with both the Free Energy and Corrected Euclidean Commute-Time distances when used in the distance-based k-means. Let us now consider the Sigmoid Commute-Time kernel, a sigmoid transformation applied on the kernel derived from the Commute-Time distance, which will be jointly used with the kernel-based k-means. Let us recall (see Section 9.2.1) that for the sigmoid transformation parameter α, we will first consider a fixed value of 7 for all datasets and, in a second time, an optimized value per dataset coming from [77].

As implementing the kernel-based k-means algorithm in R was part of the assignment, we re-computed the ARI and NMI scores. Our results are consistent with those from [77]; they slightly differ as the k-means algorithm is non-deterministic. Overall, using optimized values of α leads to improvements that range from modest to substantial (see Appendix E, Table E.1).

[16] Thereafter, the acronym DBKM will only refers to the combination of the Corrected Euclidean Commute-Time distance and the distance-based k-means.

This result confirms the importance of tuning the parameter α as stated in [77]. Therefore, we will retain the Sigmoid Commute-Time kernel with optimized parameter values for further comparisons with other methods.

10.2.3. Overall results

The average ranks for each method and partition quality measure provide a first insight of the relative performance of the methods (see Table 10.5).

Table 10.5: Average rank sums for each method and each partition quality measure (the higher, the better).

	DBKM	KBKMopt	LM	RCCN	RCSCTK
$\bar{R}_j$(ARI)	1.0625	3.4375	3.1563	3.5000	3.8438
$\bar{R}_j$(NMI)	1.1875	3.8125	3.0000	3.2500	3.7500

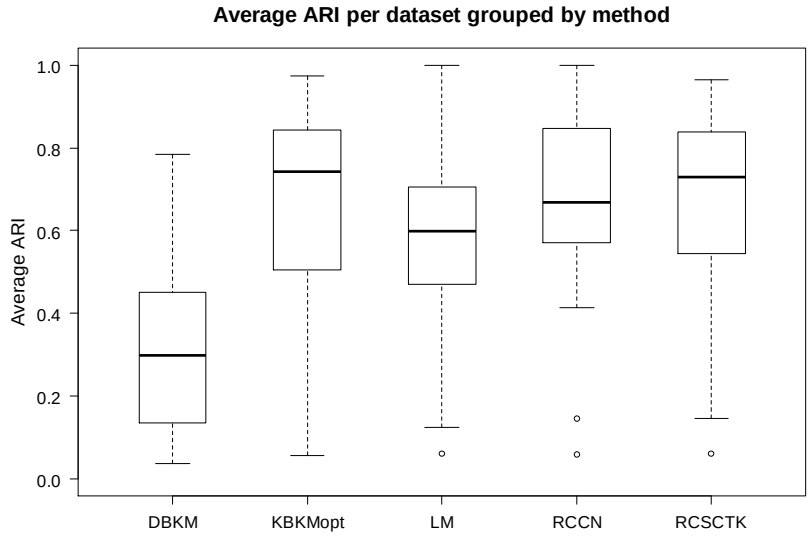

Figure 10.4: Boxplots of the average ARI per dataset for the partitions returned by the five different methods. Average NMI scores are not displayed here as they exhibit a similar pattern (see Appendix E, Figure E.1).

Friedman test indicates strong significance for both ARI and NMI scores, respectively $\chi_F^2(4)$ = 31.661 (p-value < 0.0001) and $\chi_F^2(4)$ = 29.25 (p-value < 0.0001), the null hypothesis is rejected and we then perform the Nemenyi test to detect which methods significantly differ.

As k = 5 and D = 16, the critical value is 2.728 at α = 0.05 (see Table 9.2) and from Equation 9.2, the critical difference between two average ranks is 1.2452. From Tables 10.6 and 10.7, we can make a distinction between two groups of performance. According to the p-values, the DBKM which has the lowest average rank differs very significantly to all other methods whereas the pairwise comparisons between the remaining methods lead to very high p-values. Therefore, in the first group, we find the DBKM while the second group contains the four other methods (KBKMopt, LM, RCCN, and RCSCTK). Unfortunately, the Bonferroni-Dunn test, though more

powerful than the Nemenyi test, does not allow us to better separate the relative performance of the methods. The very same conclusion can be drawn from both the ARI and NMI scores.

10.2.4. Results according to the different categories of datasets

Let us now try to further develop our findings by studying the performance of our methods on each category of datasets. Let us recall that the 16 datasets are divided into three categories (see Chapter 7): Real-world social networks datasets, artificial network datasets constructed by the LFR model and datasets derived from the Newsgroup dataset which could contain naturally overlapping communities.

Table 10.6: Lower triangle matrix containing the p-values of the pairwise comparisons according to Nemenyi test based on Friedman-test ranking matrix defined from ARI scores.

ARI	DBKM	KBKMopt	LM	RCCN
KBKMopt	0.0002			
LM	0.0017	0.9871		
RCCN	0.0001	0.9999	0.9728	
RCSCTK	< 0.0001	0.9904	0.7339	0.9728

Table 10.7: Lower triangle matrix containing the p-values of the pairwise comparisons according to Nemenyi test based on Friedman-test ranking matrix defined from NMI scores.

NMI	DBKM	KBKMopt	LM	RCCN
KBKMopt	< 0.0001			
LM	0.0104	0.5929		
RCCN	0.0021	0.8526	0.9917	
RCSCTK	< 0.0001	1.0000	0.6651	0.8991

10.2.4.1. Social networks datasets

From the average rank sums resulting from the NMI scores, the Friedman test shows no significance: $\chi_F^2(4) = 8$ (p-value = 0.0916). However, it does show significance with the ARI scores: $\chi_F^2(4) = 10.707$ (p-value = 0.0301). According to Nemenyi test (see Table 10.9), we can notice that RCCN differs very significantly to DBKM at $\alpha = 0.05$. As other contrasts are not significant at $\alpha = 0.05$, we cannot draw any conclusion based on this test about the other methods (i.e., KBKMopt, LM and RCSCTK) and carrying out the Dunn-Bonferroni test does not allow us to refine these conclusions. However, for these three unclassified methods and still according to average rank sums derived from the ARI, we can observe that:

- KBKMopt performs better than DBKM on 5 datasets out of 5, and than RCCN on one dataset;
- LM performs better than DBKM on 4 datasets, and than RCCN on one dataset;
- RCSCTK performs better than DBKM on 5 datasets, and than RCCN on 2 datasets.

Table 10.8: Average rank sums for each method and each partition quality measure (the higher, the better).

	DBKM	KBKMopt	LM	RCCN	RCSCTK
$\bar{R}_j$(ARI)	1.2000	3.2000	2.7000	4.2000	3.7000
$\bar{R}_j$(NMI)	1.6000	3.2000	2.4000	3.8000	4.0000

Table 10.9: Lower triangle matrix containing the p-values of the pairwise comparisons according to Nemenyi test based on Friedman-test ranking matrix defined from ARI scores.

ARI	DBKM	KBKMopt	LM	RCCN
KBKMopt	0.2660			
LM	0.5620	0.9870		
RCCN	0.0230	0.8550	0.5620	
RCSCTK	0.0910	0.9870	0.8550	0.9870

10.2.4.2. LFR datasets

As D = 2 et k = 5, we will consider the exact critical values of the Friedman test for more accuracy. From Equation 8.1, $\chi_F^2(4) = 6.8$ for both ARI and NMI scores and the exact critical value for D = 2 and k = 5 at a level of confidence α of 0.05 is 7.6 [47]. Although in Table 10.10 we can observe a clear hierarchy in the methods according to their respective average rank, the Friedman test does not show significance since $\chi_F^2(4) < \chi_{4;0.05}^2$.

Table 10.10: Average rank sums for each method and each partition quality measure (the higher, the better).

	DBKM	KBKMopt	LM	RCCN	RCSCTK
$\bar{R}_j$(ARI)	1.0000	3.0000	5.0000	3.5000	2.5000
$\bar{R}_j$(NMI)	1.0000	3.0000	5.0000	3.5000	2.5000

10.2.4.3. Newsgroup topics datasets

With these datasets, the Friedman test shows significance for both ARI and NMI scores: $\chi_F^2(4)$ = 21.422 (p-value = 0.0003) and 23.733 (p-value < 0.0001), respectively.

Table 10.11: Average rank sums for each method and each partition quality measure (the higher, the better).

	DBKM	KBKMopt	LM	RCCN	RCSCTK
$\bar{R}_j$(ARI)	1.0000	3.6667	3.0000	3.1111	4.2222
$\bar{R}_j$(NMI)	1.0000	4.3333	2.8889	2.8889	3.8889

From the Nemenyi test (see Tables 10.12 and 10.13), we can identify two groups of methods with a significant difference in performance at α = 0.05. From the average rank sums resulting from the NMI scores, one group only contains the DBKM while the second one includes the KBKMopt and the RCSCTK. Regarding the LM and RCCN, we cannot say to which group they are equivalent although the p-values resulting from the comparison with the DBKM are rather small. The same holds with the ARI with the exception for the RCCN which is now included in the second group.

Table 10.12: Lower triangle matrix containing the p-values of the pairwise comparisons according to Nemenyi test based on Friedman-test ranking matrix defined from ARI scores.

ARI	DBKM	KBKMopt	LM	RCCN
KBKMopt	0.0032			
LM	0.0564	0.8991		
RCCN	0.0373	0.9458	0.9999	
RCSCTK	0.0002	0.9458	0.4718	0.5685

Table 10.13: Lower triangle matrix containing the p-values of the pairwise comparisons according to Nemenyi test based on Friedman-test ranking matrix defined from NMI scores.

NMI	DBKM	KBKMopt	LM	RCCN
KBKMopt	< 0.0001			
LM	0.0830	0.2970		
RCCN	0.0830	0.2970	1.0000	
RCSCTK	0.0010	0.976	0.6650	0.6650

However, if we consider the Bonferroni-Dunn test with the KBKMopt, LM, RCCN and RCSCTK as control methods, each one at a time, the DBKM is the only method that has a significantly different performance than each control method at $\alpha = 0.05$. This conclusion holds for the average rank sums derived from both the average ARI and NMI scores.

10.2.5. Pairwise comparisons of the methods

As neither the Nemeny test nor the Bonferroni-Dunn test provides us with better discrimination between some methods, we will perform visual pairwise comparisons based on the average ARI and NMI scores per dataset [14]. Considering two methods, we generate a two-dimension graph to compare their respective average ARI or NMI scores for all datasets. We will make no comparison with respect to the Euclidean Corrected Commute-Time distance-based k-means as it was often shown to be significantly inferior in terms of performance to all other methods.

10.2.5.1. Relational Clustering on Common Neighbors versus Relational Clustering on Sigmoid Commute-Time kernel

From Figure 10.6, we can observe that RCSCTK seems to perform slightly better on Newsgroup datasets while the RCCN looks more effective on the LFR datasets. Regarding social networks datasets, it is difficult to draw any conclusion.

10.2.5.2. Relational Clustering on Sigmoid Commute-Time kernel versus Sigmoid Commute-Time kernel-based k-means with optimized parameters

Here both methods seem to be equally effective on a majority of datasets when considering the ARI while the KBKMopt seems to be a bit more effective when considering the NMI scores (see Figure 10.7).

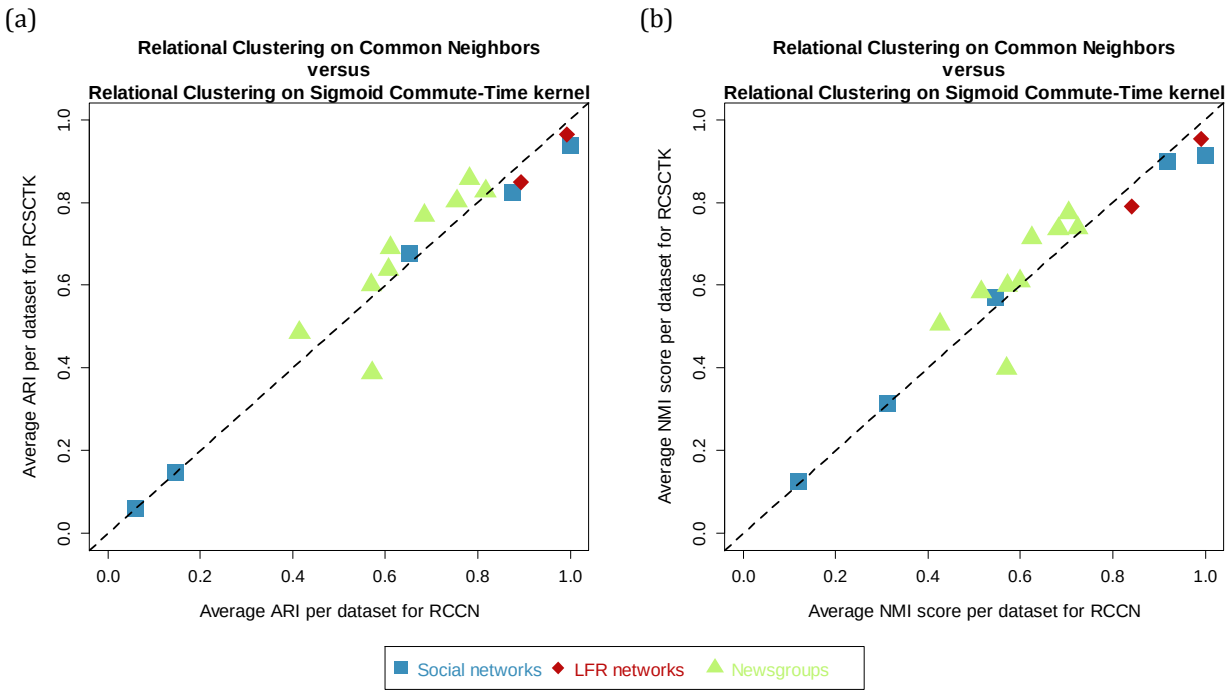

Figure 10.6: Comparison based on average (a) ARI and (b) NMI scores per dataset between Relational Clustering on Common Neighbors and on Sigmoid Commute-Time kernel.

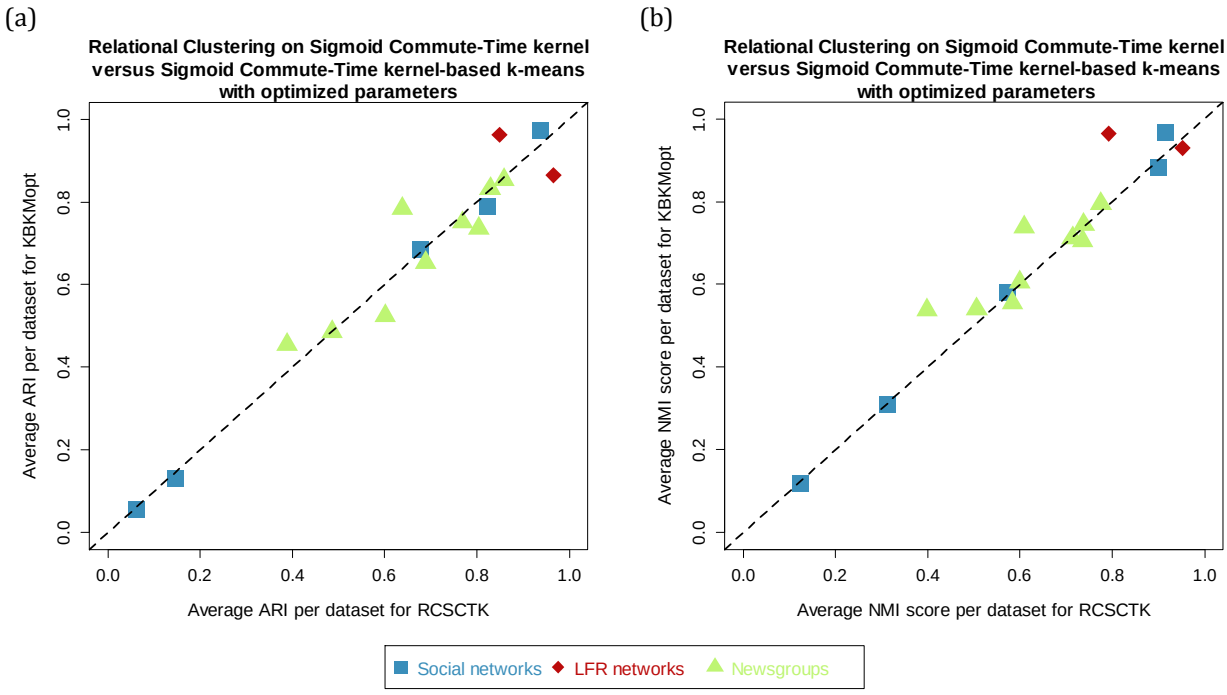

Figure 10.7: Comparison based on average (a) ARI and (b) NMI scores per dataset between Sigmoid Commute-Time kernel-based *k*-means with optimized parameters and Relational Clustering on Sigmoid Commute-Time kernel.

10.2.5.3. Louvain Method

The Louvain Method (when coerced into returning the true quantity of classes) seems to be outperformed by the three methods previously compared (KBKMopt, RCCN and RCSCTK) on a majority of datasets with the exception of the LFR datasets, the only datasets for which the Louvain Method always returns an optimal quantity of communities equal to the ground truth quantity (see Appendix B, Table B.3). The same conclusion holds for both ARI (see Figure 10.8) and NMI scores (see Appendix E, Figure E.2).

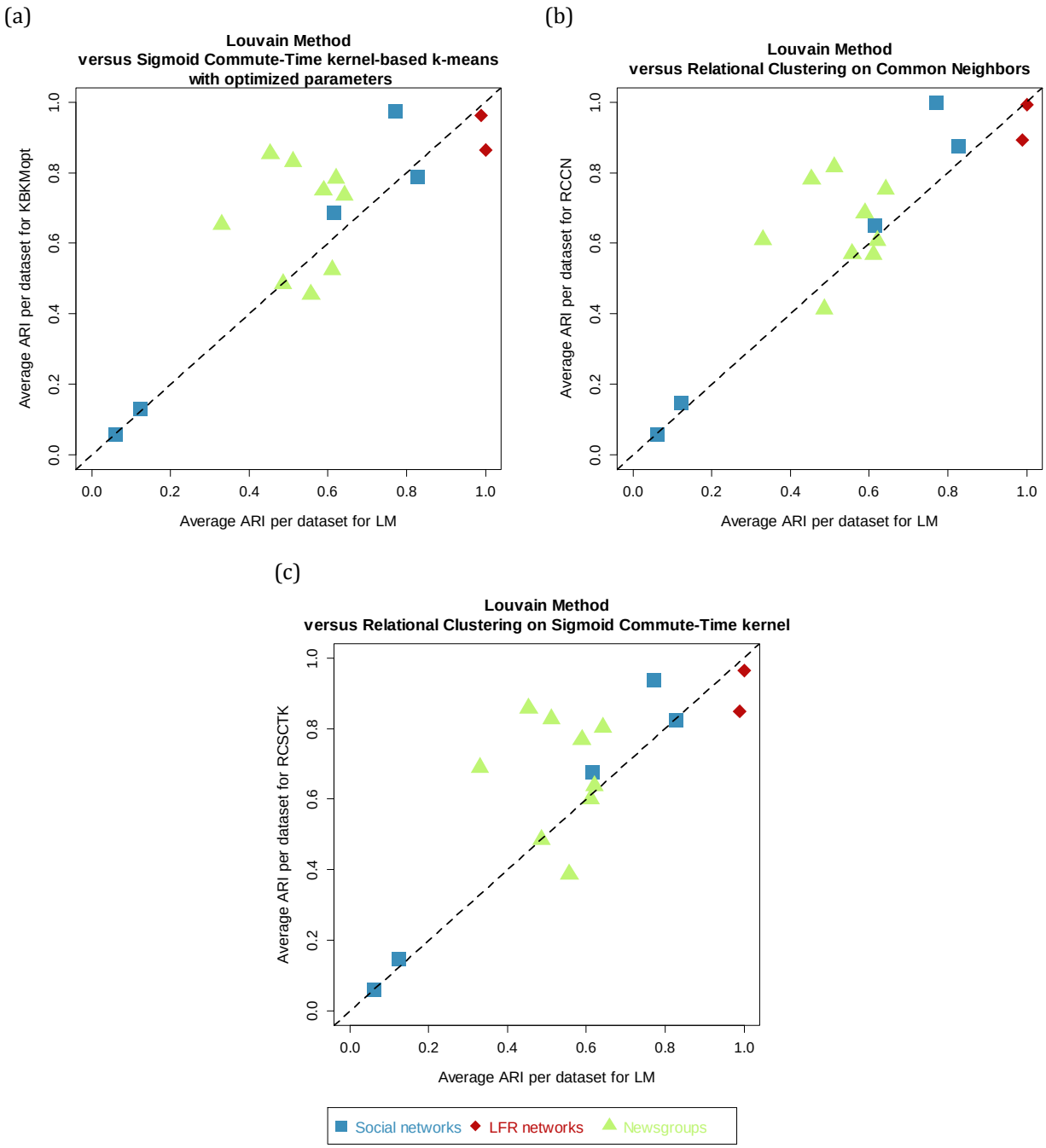

Figure 10.8: ARI-based comparison between the Louvain Method and (a) the SCTK-based *k*-means with optimized values of α, the Relational Clustering on (b) Common Neighbors and on (c) the Sigmoid Commute-Time kernel (α = 7).

10.2.5.4. Condorcet method

Finally, pairwise comparisons between methods can be displayed in a matrix where each cell contains the number of times each method in a row beats a method in a column. Based on ARI, RCSCTK is a *Condorcet winner* [46] since it beats all other methods, whereas it is KBKMopt according to NMI scores (see respectively Tables 10.14 and 10.15).

Table 10.14: Voting matrix resulting from pairwise comparisons based on ARI for 16 voters (datasets) and 4 candidates (methods).

ARI	KBKMopt	LM	RCCN	RCSCTK
KBKMopt	–	9	8	6
LM	7	–	6	6†
RCCN	8	10	–	6
RCSCTK	10	9†	10	–

† LM and RSCTK are tied on one dataset (`school_6`).

Table 10.15: Voting matrix resulting from pairwise comparisons based on NMI scores for 16 voters (datasets) and 4 candidates (methods).

NMI	KBKMopt	LM	RCCN	RCSCTK
KBKMopt	–	10	10	9
LM	6	–	6	6
RCCN	6	10	–	5
RCSCTK	7	10	11	–

10.2.6. Conclusion

At the end of this second research question, we face mixed results regarding the performance assessment of the clustering methods in the detection of the ground truth number of clusters.

- Regarding the distance-based k-means, it was shown that its combination with the Corrected Euclidean Commute-Time distance significantly outperforms the combination with the raw Euclidean Commute-Time distance. However, it was not significantly different in terms of effectiveness than the Free Energy distance-based k-means.
- That being said, it is very significantly outperformed by all other clustering methods. Then, according to the post hoc Nemenyi test, we have a group of effectiveness-equivalent methods comprising these four other methods. Eventually, the Bonferroni-Dunn test does not allow us to better discriminate between these methods.
- Concerning the Sigmoid Commute-Time kernel-based k-means, the tuning of the parameter α proved to improve both quality measures. Although improvements are sometimes small (see Appendix E, Table E.1), they are still desirable.
- The Relational Clustering on the Common Neighbors matrix and on the Sigmoid Commute-Time kernel exhibit competitive performance with respect to the kernel-based k-means.

- What concerns the modified Louvain Method, it is outperformed by all the methods (with the exception of the distance-based k-means) on a majority of datasets with the notable exception of the LFR datasets for which it always returns an optimal quantity of communities equal to the true quantity with competitive ARI and NMI scores in both cases. Nevertheless, we cannot conclude that it provides statistically worst performance than the other methods.

Furthermore, we have to mitigate somewhat all our results even if they are based on statistical evidence.

- Indeed, a first potential but central issue is that the Friedman test could in fact be too conservative [15] to effectively discriminate between our methods. We could have considered other statistical tests to compare the relative effectiveness as well as partitioning quality metrics other than the ARI and NMI.
- Moreover, when building the ranking matrix for the Friedman test, we lose a substantial amount of information such as the fact that, for some methods, the average ARI or NMI scores per dataset could be very close to each other and these small gaps in performance are eventually amplified by the use of ranks.
- In addition and because all our algorithms are non-deterministic, we cannot be sure that running each algorithm 50 times is sufficient to alleviate the effects of runs with poor initialization steps on the average ARI and NMI scores.

Finally, other considerations should also be taken into account and will come back to them in our conclusive words.

11. CONCLUSION AND FURTHER WORK

In this thesis, we investigated several methods to detect meaningful cohesive structures in undirected weighted and unweighted graphs. We considered two approaches, first targeting an optimal number of clusters and then the ground truth quantity, and selected algorithms that were appropriate in each approach. We discussed two different clustering methods, the Relational Clustering and the k-means algorithm, based on some local topological properties, the neighborhood of the nodes, and on the global structure of the graph captured by pairwise distance and similarity matrices between the nodes. We also discussed the Louvain Method considered as the state-of-the-art community detection algorithm based on the maximization of modularity. Effectiveness of the different algorithms was quantified by the Adjusted Rand Index and the Normalized Mutual Information. Finally, algorithms were compared by using the Friedman test, the post hoc Nemenyi test and the Bonferroni-Dunn test.

11.1. Key findings

In the first approach, we only considered algorithms that return an optimal quantity of clusters or communities according to their respective criterion: The Louvain Method and the Relational Clustering. In this approach and based on our findings:

- We would recommend the Louvain Method based on the modularity criterion because of its well-established efficiency and because it is the method that, on average, returns an optimal quantity of communities that is the closest to the ground truth number. It was also ranked first according to the average rank sums derived from the ARI and NMI scores.
- We would not recommend the use of the Relational Clustering on Common Neighbors because of the large execution times even on small datasets. This method could be considered for networks whose communities are cliques or near-cliques but let us recall that real-world networks are broadly sparse.
- Our results do not allow us to draw any conclusion regarding the joint use of the Relational Clustering and the Sigmoid Commute-Time kernel but the average optimal quantity of clusters is larger than the one returned by the Louvain method. Although we cannot conclude that it is statistically different in terms of performance from the Louvain Method, it is however defeated by the latter on 13 datasets out of 16 according to the average ARI and NMI scores.

In the second approach, we coerced the methods to partition the datasets into the ground truth number of communities or clusters. To this end, we considered modified versions of the Relational Clustering and Louvain method as well as the distance- and kernel-based k-means.

- While we considered three different distance metrics, the distance-based k-means is significantly outperformed by the other methods on the datasets.
- There is no statistical difference in terms of performance between the four other methods, i.e. the Sigmoid Commute-Time kernel-based k-means with optimized parameter values, the modified Louvain Method, the Relational Clustering on Common Neighbors and on the Sigmoid Commute-Time kernel.

Nevertheless, we would like to add some remarks that seem us meaningful.

- We coerced the Louvain Method and the Relational Clustering into returning the true quantity of communities despite their natural ability to find an optimal quantity of communities and our results are only valid for this way of proceeding.
- We found no statistical evidence to better discriminate between those methods even if for the modified Louvain Method, we noticed through the pairwise comparisons of methods some weaknesses in its effectiveness and therefore would not recommend this method.
- Relational Clustering showed competitive performance results in comparison with the kernel-based k-means and the modified Louvain method. Nevertheless, we have to find the value of the parameter from which we will start to get partitions containing the true number of clusters. From this perspective, Relational Clustering is more time-consuming, less practical and therefore less efficient than the other two methods.
- Eventually, in comparison with the previously mentioned methods, the Sigmoid Commute-Time kernel-based k-means (with optimized parameter values) is remarkable for both its practical and effective aspects on our datasets.

11.2. Research limitations

As any research, this thesis is by no means without limitations. From our point of view and without exhaustiveness, the most important ones are:

- Because of the scarcity of network datasets containing the nodes label, our clustering methods were tested on only 16 datasets. It would have been desirable to test them on other network datasets whose community structure is known and representing different types of systems. Nevertheless, we outlined that among our datasets, three categories of networks could be distinguished but no clustering algorithm was found to perform significantly better than the other ones on a specific category. Finally, all the datasets must be considered as of relatively small size, the largest one containing only 999 nodes.

As previously outlined, the computational complexity of an algorithm must also be taken into account when assessing the overall performance of an algorithm. Loosely speaking, the most accurate ones are often the slowest and thus do not scale well on large graphs whereas faster algorithms tend to be less accurate. For instance, the Louvain Method which was introduced nearly ten years ago is still seen as the state-of-the-art in community detection based on maximization of modularity. And it is so because even if several other methods can achieve higher levels of modularity than the Louvain Method they cannot process large graphs in a reasonable amount of time.

- We used only two external quality measures, the Average Rand Index and the Normalized Mutual Information. Although these two metrics provides us with different types of information regarding the quality of a partition returned by an algorithm, we could have considered other quality indicators that could have potentially allowed us to better discriminate between the clustering algorithms.
- The lack of significant contrast between the performance of the algorithms, from a global point of view or when considering one specific category of datasets, could also come from the fact that the Friedman test is maybe too conservative. In this case, more datasets would have been required to alleviate this tendency, especially regarding the datasets generated by the LFR model for which we only considered two samples.

11.3. Directions for further work

We implemented some of the algorithms that were required for the writing of this thesis. While we checked their effectiveness, their efficiency, that is their computational complexity and scalability on large graphs, leaves room for further work as different implementations of the same algorithm could lead to dramatically different execution times. Through the valuable experience in computer programming we gathered during the implementation task of the different algorithms and by studying several algorithms that were already implemented, we refined several times the implementation of our algorithms leading to some significant improvements in their execution time. Nevertheless, at this point we cannot be sure that it could not be further improved. Let us recall that we did not consider graphs larger than one thousand nodes.

However, we can outline two particular points that should receive attention. The first one is the computation of the pseudo inverse Laplacian matrix. Even with our small size networks, this step seems to take a relatively large amount of time. Maybe using computational approximations of this matrix could help to significantly improve the amount of time required for its computation. The second point of attention concerns the Relation Clustering algorithm. Our implementation does not seem to scale well on graphs of more than a few hundreds of nodes. As

stated in [76], a version of this clustering algorithm which was implemented by IBM in the Intelligent Miner software was able to process millions of individuals. This obviously leaves room for enhancement. Finally, using a different programming language could also help to improve the efficiency of the algorithms.

Second, and assuming the problem of scalability of the algorithms to be solved if it is actually feasible, the research process should be carried out on more and larger real-world network datasets with known community structure. This should be done because such a research is ultimately motivated by potential applications on real-world data. Moreover, gathering more performance scores per method could lead to more robust conclusions regarding the task of performance comparison. It could also help to alleviate the conservative tendency of the Friedman test if we want to still consider this test for performance comparison.

Regarding the Louvain Method, we could also consider the improvements proposed in the literature such as in [28]. This variant of the Louvain Method, called Louvain+, is shown to achieve higher levels of modularity than the original version but at the cost of an increasing computational time. It consists of a refinement of the final partition returned by the original Louvain Method. For each meta-community, the intermediate graphs (see Section 5.2.3) are successively unfolded till reaching the level of the original nodes. At each level, the local modularity optimization step is performed under the same conditions as in the original version of the algorithm. It would be interesting to test whether this refinement could lead to better partitions when the Louvain Method is coerced into returning partitions containing a quantity of clusters equal to the ground truth quantity.

Finally, we could also consider the Sigmoid Corrected Commute-Time kernel $\mathbf{K}_{\mathrm{CCT}}^{\mathrm{S}}$ (introduced in Section 5.2 but not used in this thesis) with the kernel-based *k*-means and the Relational Clustering to see if we could achieve significantly better performance scores than with the raw Sigmoid Commute-Time kernel.

11.4. Skills and knowledge acquired by the author

To conclude this thesis, we would like to give our personal views on its content and on what we had learnt through this process. We first had to acquaint ourselves with a novel and highly technical field of knowledge and research: Graph Theory. This field can be seen as the logical next step from the courses we had in Statistics, Multivariate Data Analysis, Data Mining and Pattern Recognition.

The most challenging part of this thesis was certainly to implement our community detection and node clustering methods in R programming language. Writing an algorithm, an ordered set of instructions to achieve an expected outcome, in this language was not the most difficult task.

As we were familiar with concepts of high-level programming languages, we only had to learn the syntax. Of course, this step requires rigorous and logical thinking but, as regarding foreign languages, a direct translation is always tempting but more often than not it will fail to produce efficient algorithms. Confronted with this fact, we had to considerably refine the implementation of our algorithms.

Finally, we strongly believe that the experience, knowledge and skills we gained during the writing of this thesis will be valuable for our future professional life as they show a multitasking profile and because of the broad range of concrete applications in a broad range of fields.

Bibliography

[1] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[2] H. Amini, R. Cont, and A. Minca. Resilience to contagion in financial networks. *Mathematical Finance*, 26(2):329-365, 2016.

[3] R. B. Bapat. *Graphs and matrices*. Springer, 2nd ed., 2014.

[4] A. Ben-Israel and T. Greville. *Generalized inverses: Theory and applications*. Springer, 2nd ed., 2003.

[5] H. Benhadda and F. Marcotorchino. L'analyse relationnelle pour la fouille de grandes bases de données. *Revue des Nouvelles Technologies de l'Information*, A-2:149–167, 2007.

[6] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, P10008, 2008.

[7] V. Boginski, S. Butenko, and P. M. Pardalos. Statistical analysis of financial networks. *Computational Statistics and Data Analysis*, 48(2):431–443, 2005.

[8] F. Bono and E. Gutiérrez. A network-based analysis of the impact of structural damage on urban accessibility following a disaster: The case of the seismically damaged Port Au Prince and Carrefour urban road networks. *Journal of Transport Geography*, 19(6):1443–1455, 2011.

[9] I. Borg and P. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer, 2nd ed., 2005.

[10] K. Buza, G. I. Nagy, and A. Nanopoulos. Storage-optimizing clustering algorithms for high-dimensional tick data. *Expert Systems with Applications*, 41(9):4148–4157, 2014.

[11] R. Campigotto, P. Conde Céspedes, and J.-L. Guillaume. A generalized and adaptive method for community detection. *ACM Journal of Experimental Algorithms*, V(N):Article A, 2014.

[12] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):Article 15, 2009.

[13] S. Choudhury, S. Ghosh, A. Bhattacharya, K. J. Fernandes, and M. K. Tiwari. A real time clustering and SVM based price-volatility prediction for optimal trading strategy. *Neurocomputing*, 131:419–426, 2014.

[14] A. Collette. *Comparison of some community detection methods for social network analysis*. Master's thesis, Business engineering at the Université catholique de Louvain, Belgium, 2015.

[15] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[16] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, P09008, 2005.

[17] J. G. Dias, J. K. Vermunt, and S. Ramos. Clustering financial time series: New insights from an extended hidden Markov model. *European Journal of Operation Research*, 243(3):852–864, 2015.

[18] J. G. Dias and S. B. Ramos. Dynamic clustering of energy markets: An extended hidden Markov approach. *Expert Systems with Applications*, 41(17):7722–7729, 2014.

[19] O. J. Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1962.

[20] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3–5):75–174, 2010.

[21] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences of the United States of America*, 104(1):36–41, 2007.

[22] F. Fouss, K. Francoisse, L. Yen, A. Pirotte, and M. Saerens. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Networks*, 31:53–72, 2012.

[23] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

[24] F. Fouss, M. Saerens, and M. Shimbo. *Algorithms and models for network data and link analysis*. Cambridge University Press, unpublished draft manuscript, 2016.

[25] F. Fouss, L. Yen, A. Pirotte, and M. Saerens. An experimental investigation of graph kernels on a collaborative recommendation task. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM'06)*, pages 863–868, 2006.

[26] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11(1):86–92, 1940.

[27] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[28] O. Gach and J.-K. Hao. Improving the Louvain algorithm for community detection with modularity maximization. In P. Legrand, M.-M. Corsini, J.-K. Hao, N. Monmarché, E. Lutton, and M. Schoenauer, editors, *Artificial Evolution*, volume 8752 of *Lecture Notes in Computer Sciences*, pages 145–156. Springer, 2014.

[29] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences in the United States of America*, 99(12):7821–7826, 2002.

[30] F. Göbel and A. A. Jagers. Random walks on graphs. *Stochastic Processes and their Applications*, 2(4):311–336, 1974.

[31] J. Han, M. Kamber, and J. Pei. *Data mining: Concepts and techniques.* Morgan Kaufmann, 3rd ed., 2011.

[32] X. Han, L. Wang, R. Farahbakhsh, Á. Cuevas, R. Cuevas, N. Crespi, and L. He. CSD: A multi-user similarity metric for community recommendation in online social networks. *Expert Systems with Applications*, 53:14–26, 2016.

[33] A. Heidarzade, I. Mahdavi, and N. Mahdavi-Amiri. Supplier selection using a clustering method based on a new distance interval type-2 fuzzy sets: A case study. *Applied Soft Computing*, 38:213–231, 2016.

[34] D. Hric, R. K. Darst, and S. Fortunato. Community detection in networks: Structural communities versus ground truth. *Physical Review E*, 90(6):062805, 2014.

[35] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[36] A. Jain and R. Dubes. *Algorithms for clustering data*. Prentice Hall, 1988.

[37] M. Kamalahmadi and M. M. Parast. A review of the literature on the principles of enterprise and supply chain resilience: Major findings and directions for future research. *International Journal of Production Economics*, 171(1):116–133, 2016.

[38] R. Kinney, P. Crucitti, R. Albert, and V. Latora. Modeling cascading failures in the North American power grid. *The European Physical Journal B*, 46(1):101–107, 2005.

[39] C. Kiss and M. Bichler. Identification of influencers – Measuring influence in customer networks. *Decision Support Systems*, 46(1):233–253, 2008.

[40] D. J. Klein and M. Randić. Resistance distance. *Journal of Mathematical Chemistry*, 12(1):81–95, 1993.

[41] N. Koochakzadeh, F. Keshavarz, A. Sarraf, A. Rahmani, K. Kianmehr, M. Rifaie, R. Alhajj, and J. Rokne. Stock investment decision making: A social network approach. In D. Ryżko, H. Rybiński, P. Gawrysiak, and M. Kryszkiewicz, editors, *Emerging Intelligent Technologies in Industries*, volume 369 of *Studies in Computational Intelligence*, pages 41–56. Springer, 2011.

[42] A. Lancichinetti and S. Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):1–9, 2009.

[43] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physics Review E*, 78(4):046110, 2008.

[44] F. Malliaros and M. Vazirgiannis. Clustering and community detection in directed networks: A survey. *Physics Reports*, 533(4):95–142, 2013.

[45] C. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[46] J.-F. Marcotorchino. *Optimisation en analyse ordinale des données*. Masson, 1979.

[47] L. Martin, R. Leblanc, and N. K. Toan. Tables for the Friedman rank test. *The Canadian Journal of Statistics*, 21(1):39–43, 1993.

[48] F. Marle and L.-A. Vidal. Project risk management processes: Improving coordination using a clustering approach. *Research in Engineering Design*, 22(3):189–206, 2011.

[49] R. Mesa-Arango and S. V. Ukkusuri. Demand clustering in freight logistics networks. *Transportation Research Part E*, 81:36–51, 2015.

[50] P. Michaud. Clustering techniques. *Future Generation Computer Systems*, 13(2–3):135–147, 1997.

[51] C. Musto, G. Semeraro, P. Lops, M. de Gemmis, and G. Lekkas. Personalized finance advisory through case-based recommender systems and diversification strategies. *Decision Support Systems*, 77:100–111, 2015.

[52] S. R. Nanda, B. Mahanty, and M. K. Tiwari. Clustering Indian stock market data for portfolio management. *Expert Systems with Applications*, 37(12):8793–8798, 2010.

[53] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8577–8582, 2006.

[54] M. E. J. Newman. *Networks: An introduction*. Oxford University Press, 2010.

[55] M. E. J. Newman. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46(5):323–351, 2005.

[56] M. E. J. Newman. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical Review E*, 64(1):016132, 2001.

[57] M. E. J. Newman. The physics of networks. *Physics Today*, 61(11):33–38, 2008.

[58] M. E. J. Newman and M. Girvan. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.

[59] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, 2004.

[60] J.-P. Onnela, J. Kaski, and J. Kertész. Clustering and information in correlation based financial networks. *The European Physical Journal B*, 38(2):353–362, 2004.

[61] M. Petitjean. Agrégation des similarités : une solution oubliée. *RAIRO Operations Research*, 36(1):101–108, 2002.

[62] T. Pohlert. *The pairwise comparison of mean ranks package (PMCMR)*. R Package, `http://CRAN.R-project.org/package=PMCMR`, 2014.

[63] M. A. Porter, J.-P. Onnela, and P. J. Mucha. Communities in networks. *Notices of the American Mathematical Society*, 56(9):1082–1097, 2009.

[64] A. Prat-Pérez and D. Dominguez-Sal. How community-like is the structure of synthetically generated graphs?. In *Proceedings of Workshop on Graph Data Management Experiences and Systems (GRADES'14)*, pages 1–9, 2014.

[65] R Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, `https://www.R-project.org`, 2015.

[66] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[67] A. Rebuge and D. R. Ferreira. Business process analysis in healthcare environments: A methodology based on process mining. *Information Systems*, 37(2):99–116, 2012.

[68] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. In *Proceedings of the 15th European Conference on Machine Learning (ECML'04)*, volume 3201 of *Lecture Notes in Artificial Intelligence*, pages 371–383. Springer, 2004.

[69] J. M. Santos and M. Embrechts. On the use of the adjusted rand index as a metric for evaluating supervised classification. In *Proceedings of the 19th International Conference*

on Tools with Artificial Intelligence, pages 576–584. Springer, 2009.

[70] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[71] M. Senelle, M. Saerens, and F. Fouss. The sum-over-forests clustering. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'14)*, pages 565–570, 2014.

[72] M. Song and W. M. P. van der Aalst. Towards comprehensive support for organizational mining. *Decision Support Systems*, 46(1):300–317, 2008.

[73] J. Steele and N. Iliinsky. *Beautiful visualization.* O'Reilly Media, 2010.

[74] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Regis, B. Lina, and P. Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*, 6(8):e23176, 2011.

[75] S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Academic Press, 4th ed., 2009.

[76] S. Tufféry. *Data mining and statistics for decision making*. Wiley, 2011.

[77] J. Van Damme. *Clustering of social networks, a managerial tool: Comparison of methods*. Master's thesis, Business engineering at the Université catholique de Louvain, Belgium, 2015.

[78] U. von Luxburg, A. Radl, and M. Hein. Getting lost in space: Large sample analysis of the commute distance. In *Advances in Neural Information Processing Systems 23: Proceedings of the NIPS'10 Conference*, pages 2622–2630, 2010.

[79] D. D. Wackerly, W. Mendenhall, and R. L. Scheaffer. *Mathematical statistics with applications*. Brooks/Cole, 7th ed., 2008.

[80] J. Yang, D. Zhuang, W. Xie, and G. Chen. A study of design approach of spreading schemes for viral marketing based on human dynamics. *Physica A*, 392(24):6494–6505, 2013.

[81] L. Yen, F. Fouss, C. Decaestecker, P. Francq, and M. Saerens. Graph nodes clustering based on the commute-time kernel. In *Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'07)*, volume 4426 of *Lecture Notes in Artificial Intelligence*, pages 1037–1045. Springer, 2007.

[82] L. Yen, F. Fouss, C. Decaestecker, P. Francq, and M. Saerens. Graph nodes clustering with the sigmoid commute-time kernel: A comparative study. *Data & Knowledge Engineering*, 68(3):338–361, 2009.

[83] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, and M. Saerens. Clustering using a random walk based distance measure. In *Proceedings of the 13th European Symposium on Artificial Neural Networks (ESAAN'2005)*, pages 317–324, 2005.

[84] K. Y. Yeung and W. L. Ruzzo. Details of the adjusted Rand index and clustering algorithms. *Bioinformatics*, 17(9):763–774, 2001.

[85] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

[86] W. W. Zachary. Flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.

[87] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[88] M. J. Zaki and W. Meira. *Data mining and analysis: Fundamental concepts and algorithms.* Cambridge University Press, 2014.

[89] J. Zhang and D. Maringer. A clustering application in portfolio management. In S.-I. Ao and K. Gelman, editors, *Electronic Engineering and Computing Technology*, volume 60 of *Lecture Notes in Electrical Engineering*, pages 309–321. Springer, 2010.

[90] Z. Zhu. Discovering the influential users oriented to viral marketing based on online social networks. *Physica A*, 392(16):3459–3469, 2013.

Appendices

Appendix A: Datasets description

Topic	Size	Topic	Size	Topic	Size
G-2cl-A		*G-2cl-B*		*G-2cl-C*	
Politics/general	200	Computer/graphics	200	Space/general	200
Sport/baseball	200	Motor/motorcycles	200	Politics/mideast	200
G-3cl-A		*G-3cl-B*		*G-3cl-C*	
Sport/baseball	200	Computer/windows	200	Sport/hockey	200
Space/general	200	Motor/autos	200	Religion/atheism	200
Politics/mideast	200	Religion/general	200	Medicine/general	200
G-5cl-A		*G-5cl-B*		*G-5cl-C*	
Computer/windowsx	200	Computer/graphics	200	Computer/machardware	200
Cryptography/general	200	Computer/pchardware	200	Sport/hockey	200
Politics/mideast	200	Motor/autos	200	Medicine/general	200
Politics/guns	200	Religion/atheism	200	Religion/general	200
Religion/christian	200	Politics/mideast	200	Forsale/general	200

Figure A.1: Topics included in the different subsets derived from the Newsgroup dataset. The size refers to the number of documents per topic which constitute the nodes of the graph. As one could notice, some topics of a subset could be somewhat related and some of these subsets could therefore contain overlapping natural classes.
Source: [81]

Table A.1: Main features of the network datasets.

Datasets	Subsets	Weighted/ unweighted	Number of nodes	Number of edges[17]	Number of true classes
Football	`foot`	Unweighted	115	613	12
LFR	`LFR1`	Unweighted	600	6,142	3
	`LFR3`	Unweighted	600	5,233	6
Newsgroup	`news_2cl_1`	Weighted	400	33,854	2
	`news_2cl_2`	Weighted	398	57,722	2
	`news_2cl_3`	Weighted	399	36,527	2
	`news_3cl_1`	Weighted	600	70,591	3
	`news_3cl_2`	Weighted	598	68,201	3
	`news_3cl_3`	Weighted	595	64,169	3
	`news_5cl_1`	Weighted	998	176,962	5
	`news_5cl_2`	Weighted	999	164,452	5
	`news_5cl_3`	Weighted	997	155,618	5
Political books	`polbooks`	Unweighted	105	441	3
School	`school_6`	Unweighted	236	5,899	6
	`school_11`	Unweighted	236	5,899	11
Zachary	`zachary`	Unweighted	34	78	2

[17] All graphs being undirected, reciprocal edges between pairs of nodes are recognized as one single edge.

Appendix B: First research question, optimal number of clusters

Table B.1: Average optimal number of clusters per dataset as automatically determined by the joint use of the Relational Clustering and the Common Neighbors matrix. Starred (*) datasets names indicates that only 10 trials instead of 50 were run on them because of the prohibitive execution time on these datasets.

Datasets	True number of classes	Average optimal number of clusters	Average ARI	Average NMI
foot	12	23.7667	0.8314	0.9034
LFR1*	3	334.2000	0.0065	0.2992
LFR3*	6	79.8000	0.2115	0.5918
news_2cl_1*	2	143.2000	0.0344	0.2467
news_2cl_2*	2	178.6000	0.0168	0.2149
news_2cl_3*	2	79.4000	0.1212	0.3057
news_3cl_1*	3	158.0000	0.1088	0.3728
news_3cl_2*	3	164.6000	0.0960	0.3622
news_3cl_3*	3	163.0000	0.0654	0.3415
news_5cl_1*	5	279.9000	0.1211	0.4426
news_5cl_2*	5	288.8000	0.0826	0.4038
news_5cl_3*	5	320.4000	0.0555	0.3754
polbooks	3	18.7400	0.1344	0.3694
school_6	6	7.9400	0.0831	0.1591
school_11	11	7.9800	0.1305	0.2835
zachary	2	4.9200	0.5721	0.6786

Table B.2: Average optimal number of clustering per dataset as automatically determined by the joint use of the Relational Clustering and the Sigmoid Commute-Time kernel with $\alpha = 7$.

Datasets	True number of classes	Average optimal number of clusters	Average ARI	Average NMI
foot	12	4.9200	0.4206	0.6574
LFR1	3	53.2667	0.0313	0.2556
LFR3	6	6.0000	0.9652	0.9528
news_2cl_1	2	15.4000	0.1400	0.2888
news_2cl_2	2	18.0000	0.1035	0.2319
news_2cl_3	2	13.2000	0.2284	0.3628
news_3cl_1	3	14.2000	0.3096	0.4407
news_3cl_2	3	15.4667	0.3292	0.4131
news_3cl_3	3	14.3333	0.3990	0.4142
news_5cl_1	5	16.6000	0.4495	0.4956
news_5cl_2	5	24.8000	0.2849	0.3608
news_5cl_3	5	19.8000	0.3202	0.3610
polbooks	3	2.2600	0.6652	0.5871
school_6	6	5.7800	0.0594	0.1196
school_11	11	5.7800	0.0988	0.2297
zachary	2	2.6600	0.8249	0.8392

Table B.3: Average optimal number of clustering per dataset as automatically determined by the Louvain Method.

Datasets	True number of classes	Average optimal number of communities	Average ARI	Average NMI
foot	12	9.6800	0.7761	0.8773
LFR1	3	3.0000	0.9877	0.9783
LFR3	6	6.0000	1.0000	1.0000
news_2cl_1	2	6.8800	0.4629	0.4861
news_2cl_2	2	8.7800	0.2933	0.3670
news_2cl_3	2	5.0800	0.4815	0.5292
news_3cl_1	3	5.9400	0.6978	0.6656
news_3cl_2	3	7.9600	0.5933	0.5887
news_3cl_3	3	7.0600	0.6297	0.6096
news_5cl_1	5	7.0000	0.6485	0.6596
news_5cl_2	5	7.3200	0.5559	0.6247
news_5cl_3	5	7.4400	0.5178	0.5294
polbooks	3	4.7600	0.6193	0.5376
school_6	6	5.6800	0.0592	0.1184
school_11	11	5.6200	0.0973	0.2244
zachary	2	4.0000	0.5244	0.6382

Appendix C: First research question, ground truth number of clusters

Table C.1: Average Adjusted Rand Index and Normalized Mutual Information score resulting of the comparison between the ground truth partitions and the partitions returned by the Louvain Method when coerced into returning a quantity of communities equal to the ground truth number.

Datasets	Ground truth number of communities	Average ARI	Average NMI
foot	12	0.8278	0.9013
LFR1	3	0.9894	0.9810
LFR3	6	0.9997	0.9996
news_2cl_1	2	0.4536	0.3994
news_2cl_2	2	0.3306	0.2789
news_2cl_3	2	0.5117	0.4640
news_3cl_1	3	0.6412	0.6293
news_3cl_2	3	0.5885	0.5556
news_3cl_3	3	0.6209	0.5845
news_5cl_1	5	0.6108	0.6274
news_5cl_2	5	0.5569	0.6170
news_5cl_3	5	0.4864	0.5180
polbooks	3	0.6151	0.5292
school_6	6	0.0614	0.1250
school_11	11	0.1243	0.2881
zachary	2	0.7700	0.7702

Table C.2: Average Adjusted Rand Index and Normalized Mutual Information score resulting of the comparison between the ground truth partitions and the partitions returned by the Relational Clustering based on Common Neighbors when coerced into returning a quantity of clusters equal to the ground truth number.

Datasets	Ground truth number of clusters	Average ARI	Average NMI
foot	12	0.8750	0.9196
LFR1	3	0.8926	0.8412
LFR3	6	0.9923	0.9907
news_2cl_1	2	0.7824	0.7045
news_2cl_2	2	0.6107	0.5150
news_2cl_3	2	0.8165	0.7239
news_3cl_1	3	0.7542	0.6819
news_3cl_2	3	0.6854	0.6243
news_3cl_3	3	0.6072	0.5990
news_5cl_1	5	0.5693	0.5711
news_5cl_2	5	0.5710	0.5700
news_5cl_3	5	0.4146	0.4255
polbooks	3	0.6516	0.5456
school_6	6	0.0590	0.1192
school_11	11	0.1465	0.3124
zachary	2	1.0000	1.0000

Table C.3: Average Adjusted Rand Index and Normalized Mutual Information score resulting of the comparison between the ground truth partitions and the partitions returned by the Relational Clustering based on the Sigmoid Commute-Time kernel with $\alpha = 7$ when coerced into returning a quantity of clusters equal to the ground truth number.

Datasets	Ground truth number of clusters	Average ARI	Average NMI
foot	12	0.8235	0.8991
LFR1	3	0.8483	0.7912
LFR3	6	0.9652	0.9528
news_2cl_1	2	0.8582	0.7754
news_2cl_2	2	0.6891	0.5833
news_2cl_3	2	0.8283	0.7383
news_3cl_1	3	0.8038	0.7360
news_3cl_2	3	0.7689	0.7131
news_3cl_3	3	0.6381	0.6091
news_5cl_1	5	0.6016	0.5997
news_5cl_2	5	0.3879	0.3980
news_5cl_3	5	0.4863	0.5062
polbooks	3	0.6770	0.5709
school_6	6	0.0614	0.1255
school_11	11	0.1458	0.3128
zachary	2	0.9383	0.9150

Appendix D: Second research question, comparison of the distance-based k-means with three different distance metrics

Table D.1: Average Adjusted Rand Index and Normalized Mutual Information score resulting of the comparison between the ground truth partitions and the partitions returned by the distance-based k-means with three different metrics: Free-Energy (FE), Euclidean Commute-Time (ECT) and Euclidean Corrected Commute-Time (ECCT) distances.

Datasets	Performance metric	FE distance	ECT distance	ECCT distance
zachary	ARI	0.8267	0.3435	0.7841
	NMI	0.8557	0.3963	0.8121
foot	ARI	0.5577	0.6035	0.6980
	NMI	0.7219	0.7645	0.8340
polbooks	ARI	0.4869	0.6154	0.6098
	NMI	0.4918	0.5609	0.5565
school_11	ARI	0.0801	0.0050	0.0644
	NMI	0.2307	0.0411	0.0775
school_6	ARI	0.0221	0.0000	0.0380
	NMI	0.0713	0.0395	0.0897
LFR1	ARI	0.1523	0.0041	0.0587
	NMI	0.1393	0.0111	0.0709
LFR3	ARI	0.4827	0.0109	0.4513
	NMI	0.4901	0.0392	0.6573
news_2cl_1	ARI	0.3685	0.0000	0.3399
	NMI	0.3159	0.0049	0.3251
news_2cl_2	ARI	0.3618	0.0000	0.1170
	NMI	0.3243	0.0049	0.1380
news_2cl_3	ARI	0.6855	0.0160	0.4492
	NMI	0.5561	0.0190	0.4245
news_3cl_1	ARI	0.4464	0.0000	0.3570
	NMI	0.4219	0.0070	0.4203
news_3cl_2	ARI	0.2487	0.0000	0.2412
	NMI	0.2099	0.0066	0.3014
news_3cl_3	ARI	0.3170	0.0000	0.3602
	NMI	0.2829	0.0066	0.3964
news_5cl_1	ARI	0.2753	0.0000	0.1743
	NMI	0.2959	0.0079	0.3447
news_5cl_2	ARI	0.1276	0.0000	0.2565
	NMI	0.1558	0.0078	0.3802
news_5cl_3	ARI	0.1276	0.0000	0.1534
	NMI	0.1558	0.0081	0.2804

Appendix E: Second research question, overall comparison when considering the ground truth number of classes

Table E.1: Average Adjusted Rand Index and Normalized Mutual Information score resulting of the comparison between the ground truth partitions and the partitions returned by each method: Euclidean Corrected Commute-Time distance-based k-means (DBKM), Sigmoid Commute-Time kernel-based k-means with $\alpha = 7$ (KBKM) and with optimized parameters (KBKMopt), Louvain Method (LM), Relational Clustering on Common Neighbors (RCCN) and on Sigmoid Commute-Time kernel with $\alpha = 7$ (RCSCTK).

Datasets	Performance metric	DBKM	KBKM	KBKMopt	LM	RCCN	RCSCTK
zachary	ARI	0.7841	0.9722	0.9726	0.7700	1.0000	0.9383
	NMI	0.8121	0.9635	0.9662	0.7702	1.0000	0.9150
foot	ARI	0.6980	0.8023	0.7891	0.8278	0.8750	0.8235
	NMI	0.8340	0.8807	0.8822	0.9013	0.9196	0.8991
polbooks	ARI	0.6098	0.6626	0.6858	0.6151	0.6516	0.6770
	NMI	0.5565	0.5781	0.5788	0.5292	0.5456	0.5709
school_11	ARI	0.0644	0.1270	0.1312	0.1243	0.1465	0.1458
	NMI	0.0775	0.3014	0.3089	0.2881	0.3124	0.3128
school_6	ARI	0.0380	0.0575	0.0572	0.0614	0.0590	0.0614
	NMI	0.0897	0.1168	0.1177	0.1250	0.1192	0.1255
LFR1	ARI	0.0587	0.9061	0.9626	0.9894	0.8926	0.8483
	NMI	0.0709	0.9211	0.9645	0.9810	0.8412	0.7912
LFR3	ARI	0.4513	0.8906	0.8638	0.9997	0.9923	0.9652
	NMI	0.6573	0.9409	0.9292	0.9996	0.9907	0.9528
news_2cl_1	ARI	0.3399	0.8168	0.8529	0.4536	0.7824	0.8582
	NMI	0.3251	0.7584	0.7949	0.3994	0.7045	0.7754
news_2cl_2	ARI	0.1170	0.6478	0.6540	0.3306	0.6107	0.6891
	NMI	0.1380	0.5543	0.5552	0.2789	0.5150	0.5833
news_2cl_3	ARI	0.4492	0.8214	0.8312	0.5117	0.8165	0.8283
	NMI	0.4245	0.7359	0.7448	0.4640	0.7239	0.7383
news_3cl_1	ARI	0.3570	0.7226	0.7360	0.6412	0.7542	0.8038
	NMI	0.4203	0.6955	0.7044	0.6293	0.6819	0.7360
news_3cl_2	ARI	0.2412	0.7047	0.7500	0.5885	0.6854	0.7689
	NMI	0.3014	0.6849	0.7125	0.5556	0.6243	0.7131
news_3cl_3	ARI	0.3602	0.7480	0.7838	0.6209	0.6072	0.6381
	NMI	0.3964	0.7100	0.7390	0.5845	0.5990	0.6091
news_5cl_1	ARI	0.1743	0.5076	0.5238	0.6108	0.5693	0.6016
	NMI	0.3447	0.6014	0.6053	0.6274	0.5711	0.5997
news_5cl_2	ARI	0.2565	0.4581	0.4554	0.5569	0.5710	0.3879
	NMI	0.3802	0.5420	0.5373	0.6170	0.5700	0.3980
news_5cl_3	ARI	0.1534	0.4904	0.4856	0.4864	0.4146	0.4863
	NMI	0.2804	0.5484	0.5393	0.5180	0.4255	0.5062

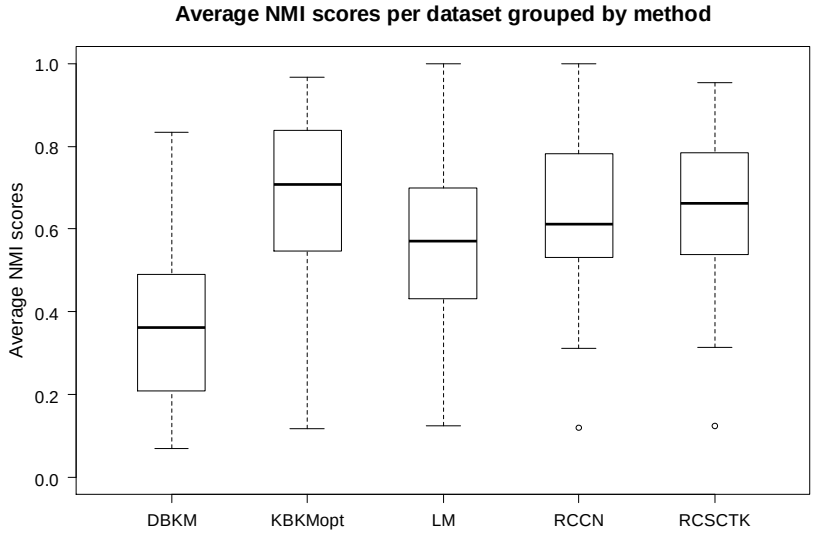

Figure E.1: Average NMI scores per dataset grouped by method: ECCT distance-based k-means (DBKM), Sigmoid Commute-Time kernel-based k-means with optimized parameters (KBKMopt), Louvain Method (LM), Relational Clustering on Common Neighbors (RCCN) and on Sigmoid Commute-Time kernel with $\alpha = 7$ (RSCTK).

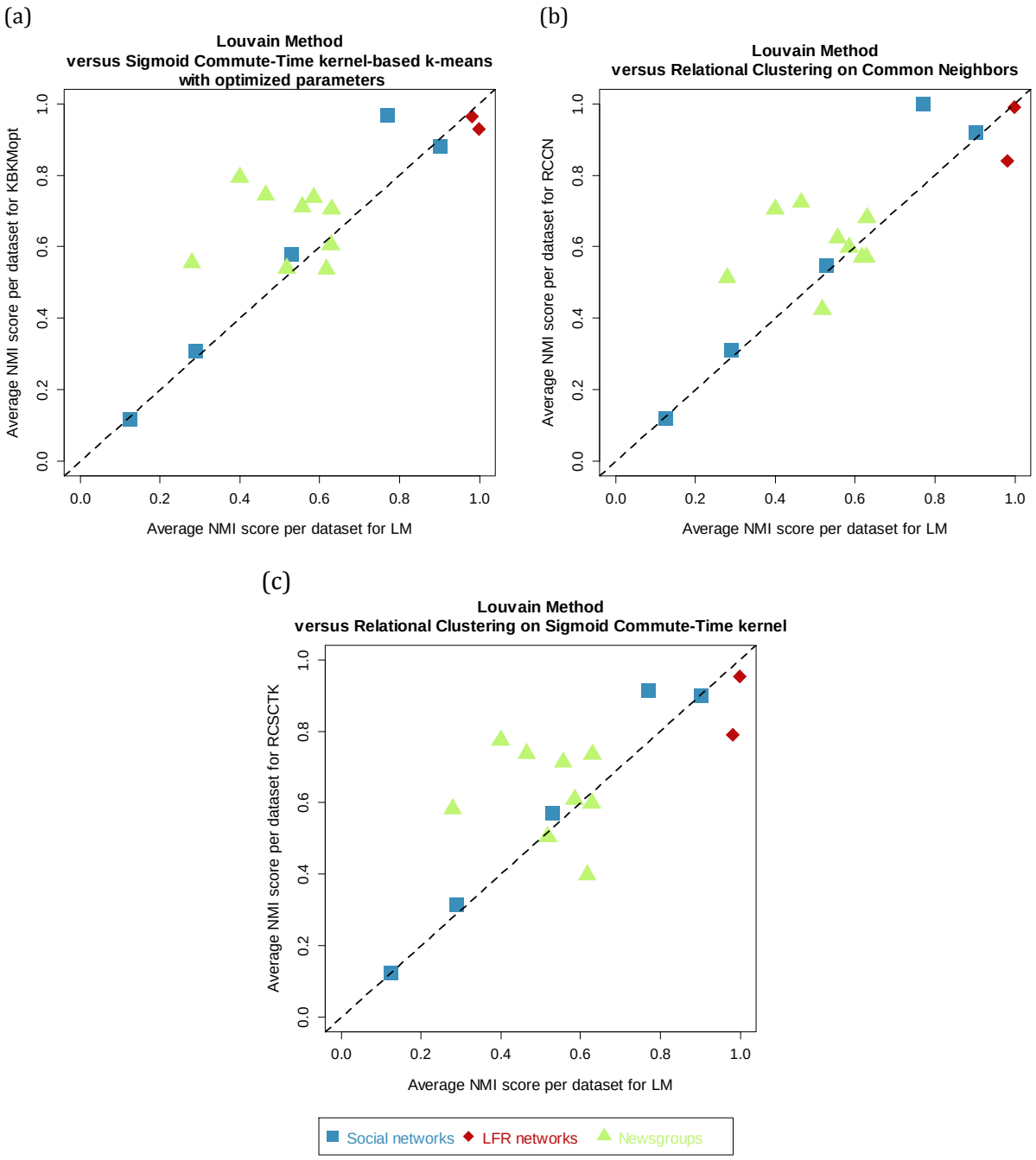

Figure E.2: NMI-based comparisons between the Louvain Method (LM) and (a) the Sigmoid Commute-Time kernel-based k-means with optimized values of α (KBKMopt), the Relational Clustering on (b) Common Neighbors (RCCN) and on (c) the Sigmoid Commute-Time kernel (α = 7) (RCSCTK).