ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program inApplied Mathematics (29^{th}cycle)

Tensor Decomposition Techniques for Analysing Time-varying Networks

By

Anna Sapienza

******

Supervisor(s):
Prof. Lamberto Rondoni, Supervisor
Dr. Ciro Cattuto, Supervisor
Dr. Laetitia Gauvin, Co-Supervisor

Doctoral Examination Committee:
Prof. Stefano Berrone,
Dott. Ciro Cattuto,
Prof. Alfio Grillo,
Prof. Yamir Moreno,
Prof. Marco Paggi,
Prof. Nicola Perra,
Prof. Lamberto Rondoni,
Prof. Lynn Schreyer

Politecnico di Torino
2017

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and this work does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Anna Sapienza
2017

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

"If you can keep your head when all about you
are losing theirs and blaming it on you,
if you can trust yourself when all men doubt you,
but make allowance for their doubting too;
if you can wait and not be tired by waiting,
or being lied about, don't deal in lies,
or being hated, don't give way to hating,
and yet don't look too good, nor talk too wise:
if you can dream and not make dreams your master;
if you can think and not make thoughts your aim;
if you can meet with Triumph and Disaster
and treat those two impostors just the same;
if you can bear to hear the truth you've spoken
twisted by knaves to make a trap for fools,
or watch the things you gave your life to, broken,
and stoop and build 'em up with worn-out tools:
if you can make one heap of all your winnings
and risk it on one turn of pitch-and-toss,
and lose, and start again at your beginnings
and never breathe a word about your loss;
if you can force your heart and nerve and sinew
to serve your turn long after they are gone,
and so hold on when there is nothing in you
except the Will which says to them: "Hold on!"
if you can talk with crowds and keep your virtue,
or walk with Kings nor lose the common touch,
if neither foes nor loving friends can hurt you,
if all men count with you, but none too much;
if you can fill the unforgiving minute
with sixty seconds' worth of distance run,
yours is the Earth and everything that's in it,
and, which is more, you'll be a Man, my son!"
Rudyard Kipling

Acknowledgements

I would like to express my gratitude to my supervisors Prof. Lamberto Rondoni and Dr. Ciro Cattuto for their help during the development of my Ph.D work. In particular, I would like to thank Dr. Ciro Cattuto for the great opportunity he gave me to work at the ISI Foundation and for his continuous support. My sincere gratitude also goes to Dr. Laetitia Gauvin, for her guidance, patience and encouragement even in the hardest moments. I could not have asked for better mentors for my Ph.D.

I acknowledge support from the Lagrange Project of the ISI Foundation funded by CRT Foundation and from the S3 project of the ISI Foundation funded by Compagnia di San Paolo.

I thank Alice, Giovanna Chiara, and all the researchers at the ISI Foundation for having rendered these three years unforgettable.

A special mention goes to Aldo, who always takes care of me and inspires me in many ways and whenever I need it. Finally, I thank my family and all my friends, that unconditionally love me and that are always by my side.

Abstract

The aim of this Ph.D thesis is the study of time-varying networks via theoretical and data-driven approaches. Networks are natural objects to represent a vast variety of systems in nature, e.g., communication networks (phone calls and e-mails), online social networks (Facebook, Twitter), infrastructural networks, etc. Considering the temporal dimension of networks helps to better understand and predict complex phenomena, by taking into account both the fact that links in the network are not continuously active over time and the potential relation between multiple dimensions, such as space and time. A fundamental challenge in this area is the definition of mathematical models and tools able to capture topological and dynamical aspects and to reproduce properties observed on the real dynamics of networks. Thus, the purpose of this thesis is threefold: 1) we will focus on the analysis of the complex mesoscale patterns, as community like structures and their evolution in time, that characterize time-varying networks; 2) we will study how these patterns impact dynamical processes that occur over the network; 3) we will sketch a generative model to study the interplay between topological and temporal patterns of time-varying networks and dynamical processes occurring over the network, e.g., disease spreading. To tackle these problems, we adopt and extend an approach at the intersection between multi-linear algebra and machine learning: the decomposition of time-varying networks represented as tensors (multi-dimensional arrays). In particular, we focus on the study of Non-negative Tensor Factorization (NTF) techniques to detect complex topological and temporal patterns in the network. We first extend the NTF framework to tackle the problem of detecting anomalies in time-varying networks. Then, we propose a technique to approximate and reconstruct time-varying networks affected by missing information, to both recover the missing values and to reproduce dynamical processes on top of the network. Finally, we focus on the analysis of the interplay between the discovered patterns and dynamical processes. To this aim, we use the NTF as an hint to devise a generative model of time-varying networks, in which we can control both the topological and temporal patterns, to identify which of them has a major impact on the dynamics.

Contents

List of Figures

List of Tables

Introduction

We always try to give a representation of our view of the world. We attempt to capture the beauty of nature in paintings and pictures, and its secrets and underlying mechanisms by describing it through synthetic representations and variables. Depending on our personal view of the world, the mechanisms at the basis of systems can be captured at different levels and angles. There is not an angle which is more relevant than another, as they simply provide a perspective from which we can observe and describe our world. As painters and photographers, we try to give a representation of what is surrounding us, in a way that allows to highlight the objects that count most in our view of natural systems. In this thesis I want to explain the perspective I used in my work, to represent the relations occurring between individuals, via two main ingredients: **time-varying networks** and **tensor decompositions**.

Time-varying networks are a natural way to represent a great variety of systems in nature [1], such as communication systems (phone calls or e-mails), online social networks (Facebook, Twitter), etc. They allow to take into account not only the structure of a system, i.e. the entities and their relations, but also how these relations vary over time.

A key research topic is then to capture meaningful **properties of time-varying networks**, such as the presence of entity similarities (i.e. communities, clusters), or of unusual changes in the network structure (i.e. anomalies, outliers). The analysis of such properties in time-varying networks provides a way to better interpret real system behaviours and their essential features: how the entities are organized and what their mutual connections mean. Moreover, the investigation of time-varying network properties helps to understand how the existing relations in the network elements **impact some real dynamics**

occurring on top of it [2], such as information diffusion, event cascades or disease spreading.

In the present thesis, we attempt to give our contribution to these main aspects regarding network organization and interaction with dynamical processes. To this aim, we decided to study the problem by taking advantage of both **theoretical and data-driven approaches**. In particular, our purpose is to improve our understanding in the following directions: in the extraction of meaningful patterns from time-varying networks; in the use of such patterns to investigate challenging questions such as anomaly detection and missing data recovery; in the analysis of the interplay between the network properties and dynamical processes.

To this aim, in the first part of the manuscript (Chapter 1) we will provide an overview of the basilar notions related to time-varying networks. We will describe the metrics used to characterize networks in time and topology, necessary to define the scope of our problem. There is a rich literature on time-varying networks [1, 3–6], however the detection of **mesoscale patterns** still lacks some robust framework for such multi-dimensional systems. Here, we propose to work at the boundary between **complex network theory and multi-linear algebra**, by using tensor decomposition techniques.

The use of such techniques on time-varying networks is facilitated by the fact that time-varying networks can be easily represented as multi-dimensional arrays, which are referred in this thesis as **tensors** [7]. Indeed, the presence of links in the snapshots of a time-varying network can be encoded in a sequence of adjacency matrices. This **multi-dimensional representation** allows to take advantage of decomposition methods, already applied successfully in several domains [8], to study time-varying networks.

In Chapter 2 we thus provide an overview of tensor decomposition techniques, that will be used as a basis to design our methods and thus extracting mesoscale structures from the network. In particular, we focus on Non-negative Tensor Factorizations (NTF) [9] for two main reasons: first the presence of non-negative constraints ensures to solve a well-posed problem and thus the existence of a solution [10]; second the non-negativity constraints ease the interpretation of the outcome of the decomposition.

As we will show in Chapter 4, the decomposition of time-varying networks enables to detect frequent and/or unexpected **complex patterns** with specific topological and temporal properties [11]. These patterns correspond to groups of links having correlated activity in time. One of our purpose is then to provide an explanation for the different types of patterns detected as well as their interpretation.

We will see that the interpretation of the uncovered patterns strongly depends on the case study, and that in some cases they can be related to anomalous behaviours. For this reason, we will extend the NTF framework to face the problem of detecting anomalies in time-varying networks. In particular, we will present our method, published in [12, 13], to capture anomalies which entangle both temporal and topological aspects.

By decomposing empirical time-varying networks, we better understand the mechanisms and properties which play a key role in the network itself. The NTF provides a synthetic representation (i.e. approximation) of time-varying networks and thus captures essential network features as topological and temporal patterns. However, we are aware of the fact that these patterns are only part of the overall network and thus we will try to give an answer to several questions opened by the study of the network approximation: does the approximation change some of the original features of the network by favouring others? Are these features important when simulating a dynamical process that occurs over the network? If they have an impact on the final outcome of the process, how can we modify the model to recover the correct dynamics?

In Chapter 5 we make an attempt to answer these questions by discussing which are the implications of discovering patterns at the mesoscale level via tensor decomposition techniques. Moreover, we will investigate how to take advantage of the presence of such patterns in the network to tackle a missing data problem, often encountered in data analysis. Indeed, while building contact networks we can be confronted to missing data due to incomplete data records. The main intuition behind our method to tackle this problem is that if we are able to reveal similarities in the activity of the links then we can use the presence of such correlation among the network elements to recover the overall behaviours of nodes whose activity is partially missing. Practically, we will show how to extend the tensor decomposition framework to handle missing

values and to merge different types of data (such as contacts and locations of individuals).

In this chapter, we will make a step further in the analysis of network properties relevant to dynamical processes. In particular, as the decomposition provides an approximation of the original network, we will devise a possible way to adjust its properties to include the key **ingredients relevant for dynamical processes**, such as spreading processes. This step will be essential to achieve an outcome of a dynamical process on the approximated network which is similar to the outcome in the original case.

Finally, we will rely on the previous observations to investigate the **interplay between patterns in time-varying networks and external dynamical processes** in a more systematic way. To this aim, we propose a procedure to find what mesoscale properties typically lead to a significant impact on the result of a dynamical process. Considering that NTF successfully detects meaningful patterns in many data, this can be used as an hint to build a **generative model of time-varying networks**, that contributes to this direction of research.

In Chapter 6, we will describe what are the key ideas behind our generative model. In particular, supported by the results obtained with the NTF, which approximates a time-varying network as a sum of sub-networks, we build a time-varying network from the sum of sub-networks, whose links have correlated activity in time. We will show how to tune this model so that it reproduces heterogeneity properties (such as burstiness) relevant for spreading processes.

To summarize the outline of the present thesis: in Chapter 1 and Chapter 2 we respectively provide the state of the art and background notions of time-varying networks and tensor decompositions; in Chapther 3 we describe the empirical social network data used throughout the work. In Chapter 4 we face the problem of discovering patterns in time-varying networks via tensor decomposition, and we modify the NTF framework to tackle the problem of detecting anomalies in time-varying networks. In Chapter 5 we exploit the correlated activity patterns of time-varying networks to handle the missing data problem with respect to spreading processes. Finally, in Chapter 6 we focus our analysis on detecting the complex patterns in networks having an impact on dynamical processes. Here, we take advantage of what we learnt from the previous chapters, to sketch a generative model for time-varying networks.

Chapter 1

Time-Varying Networks

A vast amount of systems, such as online social networks, face-to-face human interactions, and public transportation networks can be seen as sets of entities that share some relations. These collections of relations are naturally represented as networks, where each node represents an entity of the real system and each link represents a relation between two entities.

Relations in these systems often vary with time: phone calls are characterized by a duration, neurons activate and deactivate together in response to some external input, etc. In order to capture the temporal nature of such systems it is necessary to consider time-varying networks.

One important research topic related to time-varying networks, is the study of structural and temporal patterns. Moreover, another challenging question is the interplay between the topological and temporal features of networks with dynamical processes that might occur on top of them. Indeed, it is known that both topology and time affect the evolution of dynamical processes on networks. This is the case of diffusion processes, such as information diffusion, contagion dynamics, and opinion dynamics.

Following these research directions, we organized the chapter such that it gives the elements necessary to tackle the two following problems: characterizing topological and temporal patterns in networks, and studying the interplay between the extracted patterns and dynamical processes occurring over networks. In Sections 1.1 and 1.2, we introduce preliminary notions, including notations and descriptions of different time-varying networks, focusing on human interaction networks, that will be used in the applications shown in Chapter 4 and 5.

In Section 1.3 we provide the usual metrics for time-varying networks. In Section 1.4 we present some of the structural and temporal patterns encountered in networks. Finally, in Section 1.5 and 1.6, we introduce dynamical processes, giving particular attention to epidemic spreading.

1.1 Examples of Time-Varying Networks

As aforementioned, a great variety of systems in nature can be studied through time-varying networks, as they are suitable objects to represent relations between entities and their variation in time. For instance, human physical or virtual interactions can be described by time-varying contact networks [14–18], whose nodes are individuals and links describe the kind of connection between people: physical proximity, e-mails, etc. Other examples are infrastructural networks, where nodes can represent transportation means, e.g., planes, trains, etc., and links connect different locations, e.g., airports, cities, countries, etc.

The systems mentioned above are only few illustrations of the different application fields of time-varying networks. In the present work we will focus on the analysis of time-varying networks describing high-resolution data of human proximity [19–21, 14].

Depending on the way in which people communicate and share information, we can divide the related communication networks in different sub-groups. On the one hand, we have types of informations that are exchanged punctually in time as e-mail messages, text messages, instant messages, and chats. On the other hand, we have types of communication where the exchange of information between two people occurs during an interval of time. This is the case of phone calls that are characterized by a certain duration, or face-to-face interactions. In the present manuscript, we will consider networks of the second type, i.e. where the events have a duration.

1.2 Notation

A time-varying network $\mathcal{G}$ is composed by a set $\mathcal{V}$ of entities, called **nodes**, and a set $\mathcal{E}$ of relations, called **edges** or **links**, such that $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The links can

be also associated to a set $\mathcal{L}$ of **labels**, such that each link can be related to a certain property. Labels are assigned to each link, such that the resulting set $\mathcal{E}$ becomes $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{L}$.

The set of labels is specific to the considered application: in the case of human relations it could represent the type of interaction between individuals; in transportation networks it could represent the type of vehicle considered. In other contexts the set of labels $\mathcal{L}$ could be empty (i.e. labels are not included in the study).

As we are considering time-varying networks, the set of links are assumed to change and take place over a time span $\mathcal{T} \subseteq \mathbb{T}$, also known as the **lifetime** of the system. The temporal domain $\mathbb{T}$ usually coincides to $\mathbb{N}$ for discrete temporal systems and to $\mathbb{R}_+$ in the case of continuous temporal systems. Thus, the dynamics of a system can be represented through a time-varying network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}, \rho, \zeta)$, where:

- the **presence function** $\rho : \mathcal{E} \times \mathcal{T} \to \{0, 1\}$ indicates if a given link is present in the network at a specific time;
- the **latency function** $\zeta : \mathcal{E} \times \mathcal{T} \to \mathbb{T}$ provides the time needed to encounter a given link if starting at a given time.

In the present work, we always consider time-varying networks in which the set of nodes is fixed from one time to another, even if no links connect a node to the other nodes. However, it is possible to generalize the model to the case in which nodes varies in time, by adding

- the **node presence function** $\psi : \mathcal{V} \times \mathcal{T} \to \{0, 1\}$, providing the presence of a node at a certain time;
- the **node latency function** $\phi : \mathcal{V} \times \mathcal{T} \to \mathbb{T}$ providing the latency time of a node starting at a certain time.

1.3 Study of Time-varying Networks at the Microscopic and Macroscopic Scale

In the literature, a great variety of metrics and measures are defined for static networks [22, 23]. However the study of a system, whose events are temporally ordered, through a static network representation could lead to misleading results, as the temporal causality of certain events has an impact on the usual measures for static networks [24]. For instance, when we look at the reachability of nodes, or the existence of a path connecting two nodes, the temporal representation matters, as paths are necessarily constrained by the links activation, which follows a specific temporal order. Thus, some of the usual measures adopted in static networks were adapted to take into account the temporal order of events in a system.

We now introduce some basic notions summarized by Holme and Saramäki [1], Pan and Saramäki [25] and the known metrics for time-varying networks [26, 27]. Here, we report only part of the available definitions extended for time-varying networks, as our purpose is to give a global idea of how the network can be characterized. This is possible at the microscopic level through the definitions of node and link properties as well as at the macroscopic level through measures aimed at characterizing the overall network.

One important criterion to describe networks is to analyse how their nodes are interconnected. Understanding if a node is reachable from or can reach the others is fundamental to assess its importance in the network, in particular with regard to a spreading process, which we will introduce afterwards. In a time-varying network $\mathcal{G}$, the reachability of a node can be measured by adapting the existing metrics for the related static network G as functions of time. In particular, we report the concepts of journeys and temporal paths:

Definition 1. *Let us consider a time-varying network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}, \rho, \zeta)$, whose edges are $e_i \in \mathcal{E}$ and snapshots in time are $t_i \in \mathcal{T}$. A sequence of couples $\mathcal{J} = \{(e_1, t_1), (e_2, t_2), \ldots, (e_k, t_k)\}$, such that $\{e_1, e_2, \ldots, e_k\}$ is a **walk** in G, is called a **journey** or a **temporal walk** in $\mathcal{G} \iff \rho(e_i, t_i) = 1$ and $t_{i+1} \geq t_i + \zeta(e_i, t_i)\ \forall i < k$.*

Definition 2. *A journey for which each node in a time-varying network is visited at most once is called a **temporal path**.*

The starting time t_1 of a journey $\mathcal{J}$ is defined as **departure**($\mathcal{J}$) and the last time $t_k + \zeta(e_k, t_k)$ is said to be the **arrival**($\mathcal{J}$).

Journeys and paths then enable to define the **temporal connectivity** for couples of nodes in a time-varying network.

Definition 3. *A node u in $\mathcal{G}$ is said to be **temporally connected** to a node v if there exists a journey that goes from u to v.*

It is worth noting that the temporal order has to be respected. Thus this definition is not symmetric, i.e. if a node u is temporally connected to a node v in the time-varying network $\mathcal{G}$ then the node v might be not temporally connected to u. This observation refers to the concept of **causality**, which depends on the ordering of the connections between nodes. Causality is an important aspect, that, when taken into account, could lead to results which are different from the one usually obtained when studying the corresponding network aggregated in time [28].

Other measures can be derived from the concepts of journey and path, for instance we can characterized them by a **topological length**, i.e. the number of links that composes the journey/path, and a **temporal length** or **duration**, i.e. the window of time between the first and the last contact in the journey/path. The following measures help in determining which paths are important in the diffusion of a spreading process:

Definition 4. *The **topological length** of a journey $\mathcal{J}$ is the number $|\mathcal{J}| = k$ of couples (e_i, t_i).*

Definition 5. *The **temporal length** of a journey $\mathcal{J}$ corresponds to its duration, given by **arrival**($\mathcal{J}$) - **departure**($\mathcal{J}$).*

As for temporal and topological lengths, it is possible to define the concept of **distance** both in a topology and time.

Definition 6. *The **topological distance** between two nodes u and v at time t is denoted as $d_{u,t}(v)$ and it is given by*

$$\min\left\{|\mathcal{J}| : \mathcal{J} \in \mathcal{J}^*_{(u,v)}, \textbf{\textit{departure}}(J) \geq t\right\},$$

where $\mathcal{J}^*_{(u,v)}$ is the set of all possible paths starting from node u and ending at node v.

Definition 7. *The **temporal distance** between two nodes u and v at time t is denoted as $\hat{d}_{u,t}(v)$ and it is defined as*

$$\min\left\{\boldsymbol{arrival}(\mathcal{J}) : \mathcal{J} \in \mathcal{J}^{*}_{(u,v)}, \boldsymbol{departure}(\mathcal{J}) \geq t\right\} - t \ .$$

Definition 8. *The **temporal shortest path** from u to v is the temporal path that connects u to v, having the shortest temporal length.*

The definition of temporal distance can be then used to define some characteristics of time-varying networks at the macroscopic level. As an example, we can generalize the geodesic distance to time-varying networks as follows.

Definition 9. *The **characteristic temporal path length** [29] at time t of a time-varying network $\mathcal{G}$ corresponds to the temporal distance over all nodes couples in the network:*

$$L_t = \frac{1}{|\mathcal{V}|\left(|\mathcal{V}|-1\right)} \sum_{u,v} \hat{d}_{u,t}(v) \ .$$

Nodes in time-varying graphs can be temporally disconnected, so that the temporal distance goes to infinity $\hat{d}_{u,t}(v) \to \infty$ and the characteristic path length diverges. To overcome this issue we can define the **temporal global efficiency** [29], that is a generalization of the global efficiency of static networks.

Definition 10. *The **temporal global efficiency** of a time-varying network is given by*

$$E_t = \frac{1}{|\mathcal{V}|\left(|\mathcal{V}|-1\right)} \sum_{u,v} \frac{1}{\hat{d}_{u,t}(v)} \ .$$

We can also define measures with the aim of finding nodes which have a key role in the network, with regard to spreading processes. These measures can be used to define some global characteristics of the network, i.e. averaging node quantities over all the network nodes.

Definition 11. *Given a node u and two consecutive snapshots of a time-varying network $\mathcal{G}$ at time t_i and t_{i+1}. The **topological overlap** of the neighbourhood of u in the time window $[t_i, t_{i+1}]$ is*

$$C_u(t_i, t_{i+1}) = \frac{\sum_v \rho\left((u,v), t_i\right) \rho\left((u,v), t_{i+1}\right)}{\sqrt{\left[\sum_v \rho\left((u,v), t_i\right)\right]\left[\sum_v \rho\left((u,v), t_{i+1}\right)\right]}} \ .$$

Definition 12. *The **average topological overlap** of the neighbourhood of a node u is the average $C_u(t_i, t_{i+1})$ over all the possible consecutive snapshots in the network lifetime:*

$$C_u = \frac{1}{|\mathcal{T}| - 1} \sum_{i=1}^{|\mathcal{T}|-1} C_u(t_i, t_{i+1}) \ .$$

The average topological overlap is the generalization of the **local clustering coefficient** [30] in which the temporal dimension is taken into account as it measures the tendency of links to be active across different windows of time. These definitions can be then used to measure the average probability of a link in the network to be active at two consecutive times:

Definition 13. *The **temporal-correlation coefficient** is given by the average of C_u over all the possible $u \in \mathcal{V}$:*

$$C = \frac{1}{|\mathcal{V}|} \sum_u C_u \ .$$

As we previously introduced, the node properties of complex networks are particularly important to understand their dynamics. Specifically, they can highlight how and if a certain node play a role in the evolution of a dynamical process that occurs over the network. To identify such nodes, having a key role in a dynamical process, some standard measures of centrality have been extended to the case of time-varying networks. Centrality indeed is a way to assess how and how much a node in the network is connected with the others.

The most common centrality measures, based on the concept of shortest path, are the **temporal betweenness centrality** and the **temporal closeness centrality**. The betweenness centrality of a node u in a static network is defined as the fraction of shortest paths between all pairs of nodes which pass through u [2], while for time-varying networks this quantity is defined as follows [31]:

Definition 14. *the **temporal betweenness centrality** of a node u at time t is given by*

$$C_B(u, t) = \frac{1}{(|\mathcal{V}| - 1)(|\mathcal{V}| - 2)} \sum_{v \neq u} \sum_{\substack{z \neq v \\ z \neq u}} \frac{U(u, v, z, t)}{\sigma_{vz}} \ ,$$

where σ_{vz} is the number of temporal shortest paths from v to z and $U(u,v,z,t)$ is the number of temporal shortest paths from u to v in which u is traversed from the path at time t or in a precedent time $t' < t$ so that the next link of the same path will be available at $t'' > t$.

Definition 15. *The **average temporal betweenness** is defined for a node u as the average of the temporal betweenness $C_B(u,t)$ over all the snapshots in time:*

$$C_B(u) = \frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} C_B(u,t_i) \ .$$

As well as the temporal betweenness centrality, we can extend the concept of closeness centrality, that in a static network is defined as the inverse of the average distance between a certain node u and all the other nodes in the static network. Closeness centrality quantifies how close a node is to all the other network nodes.

Definition 16. *The **temporal closeness centrality** is defined as*

$$C_C(u) = \frac{|\mathcal{V}| - 1}{\sum_{\substack{v \in \mathcal{V} \\ v \neq u}} d_{u,v}} \ ,$$

where $d_{u,v}$ is the length of the temporal shortest path from u to v.

In conclusion, many concepts defined in static network theory can be adapted to measure the properties of time-varying networks at the level of nodes and links as well as at the global level. All these properties provide a way to identify the elements in networks which play a central role. These concepts are important in the perspective of studying how network properties affect dynamical processes [32]. Detecting the nodes, whose role is central in the network in terms of connectivity and causality, gives a way to understand how to influence the dynamics of a process and thus helps in devising intervention strategies aimed at modifying the process evolution. As an example, we could be interested in slowing down the process when it corresponds to the spreading of an epidemics, or we would like to speed it up in the case of information diffusion.

However, the properties related to nodes and links are only part of the overall characteristics of the network, which could be composed by even more

complex structures. This is the case of mesoscale structures which could involve several nodes or links with similar characteristics. In the present work we are particularly interested in studying the network at this level. Thus, in the next section, we provide the definition for several topological and temporal characteristics of time-varying networks.

1.4 Topological and Temporal characteristics

Real-world static networks are characterized by heterogeneous properties of the nodes, i.e. they can have different degrees and centralities. Moreover, nodes in static networks can have specific global properties as high average clustering and small average path length [22]. All these characteristics lead to networks with different connection patterns between the nodes and different structures at the mesoscale level. Mesoscale structures can include **motifs**, i.e. recurrent connectivity patterns such as triangles, and **communities**, i.e. groups of nodes/links tightly connected, which display special interconnections and node organization. These connectivity patterns can have a strong impact on the network and on a possible dynamics occurring over it. As an instance, community structures can play a key role in enforcing or inhibiting diffusion processes [33].

As motifs and communities are often observed in static networks, researchers tried to extend these concepts to the temporal case. As these notions are not unique in the literature, we report in the next sections the most common definitions related to **temporal motifs** and **temporal communities**. Moreover, heterogeneity and correlation properties can be also defined on the temporal activations of the network links. In particular, two temporal mechanisms have been found to play a key role in the network evolution: **burstiness** and **memory**. As we will see, the first one describes the heterogeneity that is present in the connection of nodes over time and how the activation of their links follows an heterogeneous **inter-event time distribution**. The second one takes into account the preference of nodes to be linked between each other from one time to another.

1.4.1 Temporal Motifs

Formally, a **motif** in a static network is defined as a class of isomorphic sub-networks.

Definition 17. *Given two static networks G_1 and G_2, their adjacency matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ are said to be **isomorphic** if there exists a permutation $\mathbf{P}$ such that*

$$\mathbf{P}^{-1}\mathbf{A}_1\mathbf{P} = \mathbf{A}_2 \, .$$

If two networks are isomorphic, then they are also **topologically equivalent**, i.e. the arrangement of their links coincides up to a re-labelling of their nodes. Thus, a **motif** is representative of a class of sub-networks sharing the same arrangement of links.

Different definitions of motifs for time-varying networks can be found in the literature: the **dynamical motifs**, and the **temporal motifs**. The first one was proposed by Bajardi et al. [34], while the second one was introduced by Kovanen et al. [35].

Definition 18. *Given a path of length l, whose list of consecutive events is*

$$\{(u_0, u_1, t_0), (u_1, u_2, t_0 + 1), \ldots, (u_{l-1}, u_l, t_0 + l - 1)\} \, ,$$

*a **dynamical** or **causal motif** corresponds to a recurrent sequence of links in the events, connected by a cause-effect relationship.*

This definition does not take into account the shape of the motifs as the links in the motifs are arranged in a linear chain, which can be viewed as recurrent time-respecting paths of the time-varying network. By contrast, the temporal motifs defined by Kovanen et al. [35] are classes of isomorphic sub-networks where the isomorphism includes also the similarity in the temporal order of the events.

Definition 19. *Two events (u_1, v_1, t_1) and (u_2, v_2, t_2) are said to be $\Delta\tau$-**adjacent** if they share at least one node and $t_2 - t_1 \leq \Delta\tau$.*

Definition 20. *Two ordered events (u_1, v_1, t_1) and (u_2, v_2, t_2) are **feasible** if $t_1 < t_2$.*

Definition 21. *Two events* $e_1 = (u_1, v_1, t_1)$ *and* $e_2 = (u_2, v_2, t_2)$ *are said to be* $\Delta\tau$***-connected*** *if there exists a sequence* $\mathcal{S} = \{e_1 = e_{n_0}, e_{n_1}, \ldots, e_{n_N} = e_2\}$ *of events such that each pair of consecutive events in* $\mathcal{S}$ *is* $\Delta\tau$*-adjacent.*

The definition of $\Delta\tau$**-connectivity** leads directly to the definition of **connected temporal sub-network**, i.e. a set of events such that all the feasible pairs of events are $\Delta\tau$-connected.

Definition 22. *A* ***temporal motif*** *is a class of isomorphic temporal sub-networks, where two sub-networks are considered isomorphic if they are topologically similar and represent the same temporal pattern, i.e. the order of the event sequence is the same.*

1.4.2 Temporal Communities

Another important topological pattern that can be defined in the study of static networks is the one of community. A community is a group of nodes, densely connected internally and sparsely connected with other groups. The study of communities and their detection has improved the understanding of network organization and the way in which a dynamics is conveyed along the network in the presence of such kind of structures.

Communities can be also seen as groups of nodes that are similar to each other in some characteristic, that could be local or global. Thus, there are a wide variety of methods in the literature that aim to detect the network communities by the computation of similarity measures, as node distance and cosine similarity [36, 37]. Traditional methods are hierarchical, partitional and spectral clustering [38].

Communities define a partition of the network. However, different communities might overlap. For this reason, the **modularity function** was proposed by Newman and Girvan [39] to quantify how much the communities in which a network is partitioned overlap. It estimates the difference between the fraction of edges among nodes belonging to the same community and the expected fraction of such edges in a null-model network with no communities.

Definition 23. *Given a static network* $G = (\mathcal{V}, \mathcal{E})$ *the* ***modularity function*** Q *of the network can be computed as*

$$Q = \frac{1}{2|\mathcal{E}|} \sum_{u,v} \left(\rho(u,v) - \mathbf{P}_{u,v}\right) \delta(c_u, c_v) \ ,$$

where $\mathbf{P}_{u,v}$ *is the expected number of edges between* u *and* v *in the null-model and* $\delta(c_u, c_v) = 1$ *if the nodes* u *and* v *belong to the same community, i.e.* $c_u = c_v$.

The usual null model used for the computation of the modularity function is the configuration model [40, 41], that is a randomized version of the original network in which the degree of the nodes remains the same and whose links are randomly formed between the nodes. In this case the modularity function becomes

$$Q = \frac{1}{2|\mathcal{E}|} \sum_{u,v} \left(\rho(u,v) - \frac{k_u k_v}{2|\mathcal{E}|}\right) \delta(c_u, c_v) \ ,$$

where k_u and k_v are respectively the degree of u and v. In the literature, different extensions of the modularity function are present, in particular for weighted networks, directed networks and networks with communities that overlap [42].

Modularity has been extended to the case of time-varying networks, and corresponds to the stability at one step of a random walk on the time-varying network:

$$Q = \frac{1}{2\mu} \sum_{u,v,s,r} \left[\left(\rho_{u,v}(s) - \gamma_s \frac{k_u(s)\, k_v(s)}{2m_s}\right)\delta_{sr} + C_{vrs}\delta_{uv}\right] \delta(g_{us}, g_{vr}) \ ,$$

where the indices r and s stand for the different network snapshots, $k_u(s)$ is the degree of node u at time s, and $m_s = \frac{1}{2}\sum_u k_u(s)$ is the total number of links at time s. This modularity function assesses the quality of a partition in a time-varying network.

Although communities have been deeply studied in static networks, tracking their evolution over time is a fundamental problem as different communities can emerge at different times, they can grow, merge or decay over time. Typical temporal community detection methods assume some level of continuity in the

structure of communities over time [38]. However these approaches might not be appropriate in the case of fast changes in the community structure.

The definition of appropriate methods for detecting communities and their evolution over time is still an open problem. We will study in Chapter 4 an alternative method to detect mesoscale structures in time-varying networks, which correspond to groups of links having a correlated activity in time. Broadly speaking, these mesoscale structures can be seen as community-like patterns, as they group together nodes which are tight by a common activity in time.

1.4.3 Burstiness

Bursty activities can be thought as significantly high activity levels that take place over limited windows of time, followed by longer periods of links inactivity. Burstiness has been observed in many complex phenomena, such as online communications [43], human interactions [44, 45], gene expression [46], and earthquakes [47]. In human social interactions, burstiness is revealed by the heavy tailed form of the inter-event time distribution [48, 49], i.e. the distribution of the length of the windows of time between two consecutive events.

A great amount of work has been devoted to develop measures that can help to determine the magnitude of the bursty patterns in complex systems. One of these measures was introduced by Goh and Barabási [47] to distinguish the effect of burstiness from the one of memory.

Definition 24. *Let $P(\tau)$ be the distribution of the inter-event time (τ) between two consecutive events and $P_P(\tau)$ be the exponential distribution*

$$P_P(\tau) \approx \exp(-\tau/\tau_0) \ ,$$

where τ_0 is the average inter-event time. Thus, the ***burstiness parameter*** *can be defined as*

$$\Delta := \frac{sign(\sigma_\tau - m_\tau)}{2} \int_0^\infty |P(\tau) - P_P(\tau)| \, d\tau \ ,$$

where m_τ and σ_τ are respectively the mean and the standard deviation of $P(\tau)$.

As an alternative we can use also the ratio between the standard deviation σ_τ and the mean m_τ to compute Δ [1]:

$$\Delta = \frac{(\sigma_\tau/m_\tau - 1)}{(\sigma_\tau/m_\tau + 1)} = \frac{(\sigma_\tau - m_\tau)}{(\sigma_\tau + m_\tau)} .$$

For finite contact sequences $\Delta \in [-1, 1]$: $\Delta = 1$ corresponds to the most bursty sequence of events, $\Delta = 0$ is a sequence with Poissonian inter-event time distribution, and $\Delta = -1$ is a periodic sequence.

1.4.4 Memory

As we have seen in the previous section, for complex systems characterized by a finite discrete event dynamics it is usual to characterize the temporal inhomogeneities by the inter-event time distribution $P(\tau)$. However, this distribution as well as the quantification of burtsy behaviours is not enough to detect the presence of correlations in the sequence of events.

The correlations taking place between consecutive bursty events can be thought as a memory process, that allows to compute the probability that a particular link can activate one more time within a certain time window Δt after other n activations. Driven by the need of detecting these correlations in the case of heterogeneous signals, Karsai et al. [50] proposed a measure to quantify memory in a sequence of events.

Definition 25. *The* ***memory function*** $p(n)$ *of a sequence of e events can be defined as:*

$$p(n) = \frac{\sum_{e=n+1}^{\infty} P(e)}{\sum_{e=n}^{\infty} P(e)} ,$$

where $P(e)$ *is the distribution of the number of events belonging to the same bursty period.*

A memory coefficient was also proposed by Goh and Barabási [47]. It is defined as the correlation coefficient of all consecutive inter-event time values in the signal over all the nodes in the network.

Definition 26. *Given a time-varying network* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ *and the pairs of consecutive inter-event times* $(\tau_{n,t}, \tau_{n,t+1})$ $\forall n \in \mathcal{V}$*, the* ***memory coefficient*** μ

is defined as:

$$\mu = \frac{1}{|\mathcal{V}|} \sum_{n=1}^{|\mathcal{V}|} \sum_{t=1}^{e_n - 1} \frac{(\tau_t - m_{n,1})(\tau_{t+1} - m_{n,2})}{\sigma_{n,1}\sigma_{n,2}} ,$$

where e_n is the number of events for the node n, $m_{n,1}$ (resp. $m_{n,2}$) and $\sigma_{n,1}$ (resp. $\sigma_{n,2}$) are the mean and the standard deviation of $\tau_{n,t}$'s (resp. $\tau_{n,t+1}$'s).

This coefficient is positive when the length of the inter-event time of consecutive events tends to be similar: when a short inter-event time is followed by a short one, or a long inter-event time is followed by a long one. Otherwise, the coefficient is negative when a short inter-event time is followed by a long one and vice versa.

Burstiness and memory have been shown to play a central role in the propagation of a dynamical process that occurs over the time-varying network. We will see how these temporal characteristics impact on dynamical processes taking place over the network in Sec. 1.6.

1.5 Dynamical processes over the network

The study of time-varying networks, the extraction of meaningful patterns that compose the systems and the availability of methods to measure some special properties that characterize the network itself is particularly important to better understand how the studied system is composed and how it behaves. In the previous sections, we have seen how to characterize some topological and temporal properties that have been found to be common in many systems, and whose interpretation can help to study real-world complex systems, such as human proximity contacts, online social interactions, etc.

The advances made in the extraction and interpretation of the network structures have contributed to the analysis of the properties of the network itself and the study of the implications that these properties have on physical and dynamical processes.

Understanding how the parts composing a network interact and if they have an impact on diffusion processes is crucial to answer questions related to different fields: how a disease spreads in a population, how computer viruses or

malwares propagates in large-scale networks, how information is diffused from one user to another and which kind of users plays a key role in the diffusion.

Indeed, many types of dynamics, such as consensus, epidemic spreading, or information diffusion, could be affected by topological and temporal heterogeneities, whose relevance was stressed in several fields, such as physics, neuroscience, and computer science.

In this work we are mainly interested in the study of dynamical processes related to the spreading of a disease, as the application that we will see will be related to human proximity interactions. However, the results achieved by the study of these particular processes could be extended to different type of processes.

In the next section we will explain some of the existing models for disease spreading, and we will apply them throughout this work having two main purposes:

- to study how well the mesoscale structures that we are able to extract from time-varying networks represent the original systems with the final aim of correctly model dynamical processes;
- to find what is the interplay between the uncovered patterns and the dynamical processes, with the aim of controlling the evolution of the process through the topological and temporal patterns of time-varying networks.

1.5.1 Spreading processes

Modelling how a dynamical process occur over a time-varying network, and in particular how the spreading of an infectious disease, an information, or influence emerges and propagates in a population, usually involves two main features:

1. a model that defines the underlying structure of the system on which the process occurs: the person-to-person interactions evolving in time.
2. a model that describes the dynamics;

On the one hand, we represent the contact patterns through time-varying networks, that describe which individuals are in contact and at which time this contact occurs. On the other hand, dependently to the process that we want to analyse there exist different types of suitable models that can be used to define the dynamics of the process. For completeness, we introduce the partial differential equations that can be used to model spreading processes. However, in our applications we simulate the spreading of a disease as a stochastic process.

Spreading processes can be modelled as **compartmental models** [1, 51, 52], in which people are divided into several classes with respect to their condition during the spreading: they can be infected, susceptible to the disease, recovered etc.

The **susceptible-infected** (SI) process is the simplest compartmental model, where the network nodes are divided into two classes: susceptible (S) and infectious (I). In this model, the overall population is considered as susceptible, and each individual in the population might successively become infectious through its contacts, under some probability λ. The process continues until saturation, i.e. no more susceptible individuals are present in the population. Once that an individual becomes infectious, he/she cannot become susceptible again. Therefore, the parameter λ that corresponds to the infection probability, also controls the speed of the spreading process, as the most individuals can be infected the fastest the end of the process is reached.

Given a fixed population of N individuals, with $N = S(t) + I(t)$ the SI model can be described by the following transition:

$$S \underset{\lambda}{\rightarrow} I \ .$$

Individuals are considered to mix homogeneously, thus the differential equations that model the process are:

$$\begin{aligned} \frac{dS(t)}{dt} &= -\lambda \frac{S(t)\, I(t)}{N} \ , \\ \frac{dI(t)}{dt} &= \lambda \frac{S(t)\, I(t)}{N} \ . \end{aligned}$$

Another well-known model for epidemic spreading is the **susceptible-infected-recovered** (SIR) process, that is an extension of the SI model in

which individuals can become immune after the infection. When immunization is taken into account, an individual can enter in a recovery state (R) with probability μ. In this state individuals cannot propagate the disease to other individuals and at the same time they cannot be infected a second time by others. Given the new state in the system R and the fixed population of $N = S(t) + I(t) + R(t)$ individuals. the state transitions are defined as:

$$S \underset{\lambda}{\rightarrow} I \underset{\mu}{\rightarrow} R \,,$$

where λ is the infection probability and μ is the probability of recovering. As before, this process can be modelled via the ordinary differential equations:

$$\begin{aligned} \frac{dS(t)}{dt} &= -\lambda \frac{S(t)\, I(t)}{N} \,, \\ \frac{dI(t)}{dt} &= \lambda \frac{S(t)\, I(t)}{N} - \mu \frac{I(t)}{N} \,, \\ \frac{dR(t)}{dt} &= \mu I(t) \,. \end{aligned}$$

Heterogeneous assumption

Assuming that all the individuals in the population are equivalent, i.e. share on average the same number of contacts between each other, does not allow to represent in a complete way many complex systems, that have been shown to have heterogeneous characteristics and contact patterns. Moreover the role of heterogeneous contacts in social interaction networks, is particularly important as this type of connectivity pattern strongly influences spreading processes.

As shown in [2], the heterogeneous topology of a population can be included in the mathematical framework of the SI and SIR models by considering a **degree block approximation**, in which the individuals having the same degree are considered equivalent. In this perspective we can write the density of infected and susceptible individuals in each block as

$$i_k = \frac{I_k}{N_k} \quad \text{and} \quad s_k = \frac{S_k}{N_k} \,,$$

where I_k and S_k are respectively the number of infected and susceptible individuals having degree k, and N_k is the total number of individuals in the block representing the k-th degree.

It is now possible to explicitly compute the deterministic infection rate equations, corresponding to the evolution of the SI and SIR process as

$$\begin{aligned} \text{SI)} \quad & \frac{di_k(t)}{dt} = \lambda\left[1 - i_k(t)\right] k\theta(t) \ , \\ \text{SIR)} \quad & \frac{di_k(t)}{dt} = \lambda\left[1 - i_k(t) - r_k(t)\right] k\theta(t) - \mu i_k(t) \ , \end{aligned}$$

where θ_k is the density of infected neighbours of the nodes with degree k and

$$\begin{aligned} \text{SI)} \quad & s_k(t) = 1 - i_k(t) \ , \\ \text{SIR)} \quad & s_k(t) = 1 - i_k(t) - r_k(t) \ . \end{aligned}$$

These models can be extended to include different state transitions or to consider other types of classes. This is the case for models as susceptible-infected-susceptible (SIS) and susceptible-exposed-infected-recovered (SEIR) models.

Analytical results for spreading processes occurring on time-varying networks are usually complex to obtain: they can be obtained only in special cases and sometimes approximations are needed. Our aim is not to find an analytical solution to the partial differential equations modelling the dynamics. Thus, to study the dynamics of such spreading processes a common way is to stochastically simulate them over networks. This is done by taking at random a node (the root), which we suppose to be the initial spreader of the infection. Then, at each time an infected node has a certain probability of infecting its neighbours and, in the case of SIR processes, a probability of recovering.

1.6 Impact of Temporal and Topological Characteristics on Dynamics

Network structure and its intrinsic dynamics are relevant factors that determine the properties and the evolution of spreading processes. Thus, much effort

has been devoted to have a better insight of the main properties governing spreading processes on networks. As a result, both topological and temporal properties, such as community structures and temporal activity patterns like burstiness, have been identified as critical aspects that could strongly influence the spreading.

Several approaches have been proposed to address the problem of finding the structures playing a significant role in a diffusion process. In particular this problem was tackled from both the analytical and computational sides [53, 54]. Other approaches include the possibility of using empirical data of time-varying networks to model SI and SIR processes upon the data. These data-driven approaches, can be used on email networks [55] as well as face-to-face proximity contacts [56], and mobile phone calls [57].

Different features may have a significant part in the system dynamics. Some features are strictly related to the topology of the network, such as the presence of different weights in the links [58], or the presence of loops. Other features are on the contrary related to the temporal nature of the system, such as the bursty activation of links [59], activity-correlated classes of links [60], and memory [61].

A major observation that arise from these researches is that the most important actor influencing the evolution of a process in many different complex systems is the level of heterogeneity that characterize the system both on the temporal and topological side [62–64].

The dynamics of complex systems involving human interactions is particularly affected by heterogeneities in the contacts and in the temporal activation of links. These inhomogeneities appear in the characteristic long-tailed distribution of the inter-event times and in the bursty activation of links. These features were observed in a wide variety of systems and are commonly used to model people interactions. At the same time, these features were highlighted to be responsible to change the outcome of dynamical processes over the network [65, 66].

These observations led to further questions on how to discriminate which feature, between the temporal and topological one, is the most impacting on a dynamics [67] and how they are correlated or compete in the spreading [68].

Apart from long-tailed distributions and inhomogeneities, links activation can be correlated: a group of links can activate at the same time, be active during a time window and stay subsequently inactive for a long period. Moreover, these activations can be also topologically correlated. This is the case of networks that display community structure. Here, links between nodes that are tightly connected activate at the same time, leading to correlated activity patterns [60] that entangle both topology and temporal activation.

Both temporal and topological patterns can have an impact on dynamical processes occurring on top of time-varying networks [69, 58]. However, the impact of the combination of these two aspects is still hard to define. Thus, we will try to investigate it in Chapter 6, by the development of a model where we can control both the temporal and structural informations of the network to influence the dynamics of a process.

Intervention Strategies

The use of mathematical models able to capture the underlying network patterns is crucial to detect those aspects playing a central role in dynamical processes. The identification of such network elements allows to both predict and control the evolution of a disease spreading [65, 70, 71].

The ability of controlling the spreading can help to design some intervention strategies with the aim of mitigating and slowing down the diffusion [72–74] in a targeted way [75].

Different strategies have been proposed to improve the targeted removal of nodes that are found to be the most important spreaders in the considered network. Typical methods target nodes by looking at their connectivity patterns, such as the number of connection, i.e. degree, and betweenness [76, 77].

The aforementioned methods take into account the evolution of the spreading process and discard the dynamical activation of the links that is typical of time-varying networks. For this reason, recent approaches were developed to investigate the effect that time-varying connectivity patterns may have on epidemics control interventions. This is the case of the work developed by Liu et al. [78], in which the authors consider a class of activity driven network models [79] to control the contagion by comparing different control strategies.

Starting from the review, provided in the next section, of some of the main results achieved in the study of the impact of network properties on dynamical processes occurring over the network, we will analyse in Chapter 6 the interplay between complex patterns and dynamical processes over networks. In particular, we will study mesoscale structures that entangle both temporal and topological structures to understand how the combination of these two factors can be used to generate synthetic time-varying networks. The main purpose is to define those characteristics having a greater impact on epidemic spreading.

1.7 Modelling time-varying networks

A common way to both study network characteristics and understand how networks react in response to some inputs or when dynamical processes are taking place, is by creating synthetic networks via generative models in which we are able to tune some desired properties in the network. Another common way is instead to take advantage of empirical time-varying networks and change their properties by randomization techniques [80].

In the case of randomized reference models, the typical procedure is to take the original sequences of events in empirical time-varying networks and reshuffle them. The reshuffling is done in a way that enables to remove specific correlations. Depending on the randomized model, there are several possible correlations which can be modified or broken with the aim of understanding what is their role in the network. These types of models are commonly used to study how dynamical processes depend on the presence of these correlations. Examples of randomized models for time-varying networks are: the randomized edges model, which is similar to the configuration model; the randomly permuted times model, where the times in which contacts occur are randomly reshuffled but the network structure and number of contacts is fixed; the randomized contacts in which the contacts are redistributed among the edges [1, 58].

Other models generate time-varying networks starting from: random walks of several lengths [81], extensions of exponential random graph models [82], from Markov processes [83–85]. In the case of generative models, the parameters, used to create synthetic networks, are inferred by taking into account real time-varying networks. Here, information about the presence of network

characteristics such as motifs, community structures or temporal activities is taken into account to build synthetic networks displaying similar properties. This is the case of models as those proposed by Stehlé et al. [44], Zhao et al. [86], which are specific for modelling social group dynamics as the links represent social interactions and are based on mechanisms such as reinforcement dynamics (similar to preferential attachment), i.e. the longer an individual has contacts with people in the same group the bigger is the probability of remaining in the group and vice-versa. Generative models, based on preferential attachment mechanisms, such as the one proposed by Fortunato et al. [87], Boguná and Pastor-Satorras [88], can be defined as connectivity driven models, as the network topology is the specific characteristic taken into account to create time-varying networks.

As an alternative to connectivity driven models, Perra et al. [79] proposed an activity driven model for time-varying networks, where the so called activity potential distribution is defined to characterize interaction patterns in the network which are heterogeneous. This framework was recently extended by Laurent et al. [89], by building a model for time-varying social networks in which additional temporal and structural correlations are taken into account, i.e. memory.

Finally, we report another class of generative models which is based on stochastic block models, successfully used to generate static networks [90]. In these models a network is assumed to be divided in sub-networks, where a probability defines how nodes in a sub-network are linked and another probability is used to define the connection between sub-networks. The general framework of stochastic block models can be changed to take into account the temporal evolution of time-varying networks. As an example, Granell et al. [91] proposed a generative model in which the temporal evolution is given by periodic oscillations in the structure of communities, with the aim of modelling communities which merge/split or grow/shrink.

1.8 Network representation

To tackle the different issues introduced in the previous sections, we need to select a representation for the networks. There exist in the literature several

ways to represent time-varying networks. The most common can be classified in two groups: the **link-stream** and the **snapshot sequence** representation.

In the **link-stream** representation, a time-varying network is defined by a sequence of contacts $c = (u, v, t)$, active during the lifetime of the network:

$$\mathcal{G} = \{c_1, c_2, \ldots, c_E\} \ ,$$

where E is the total number of active edges in the time-varying network.

In this representation a contact is instantaneously active in time and links that are continuously active over a certain time period are repeated in the list by updating the activation time. Thus, given a link (u, v) active for a period of time of length Δt the link-stream of the network will include the list of contacts $\{(u, v, t_0), (u, v, t_1), \ldots, (u, v, t_0 + \Delta t)\}$, where t_0 is the activation time of (u, v).

As an alternative it is possible to include the duration of the links by adding the ending time in the contacts, such that $c = (u, v, t_s, t_e)$, with starting time t_s and ending time t_e.

Another standard representation is the **snapshot sequence** representation, which describes a time-varying network $\mathcal{G}$ as a sequence of static networks G that correspond to the state of the network at a certain time t:

$$\mathcal{G} = \left\{G_0, G_1, \ldots, G_{|\mathcal{T}|}\right\} \ ,$$

where $|\mathcal{T}|$ is the total number of snapshots in the network, corresponding to the times in the lifetime. The snapshot representation is particularly suitable as the snapshot $G_t = (\mathcal{V}, \mathcal{E})$ at time t can be written through an **adjacency matrix** $\mathbf{Adj}_t$ of the form:

$$\mathbf{Adj}_t = \begin{cases} a_{uv} = 1 \ \text{ if } \ (u, v) \in \mathcal{E} \ , \\ a_{uv} = 0 \ \text{ otherwise} \ . \end{cases}$$

Therefore, it is now straightforward to incorporate the sequence of adjacency matrices in a three-dimensional array $\boldsymbol{\mathcal{X}}$ whose elements at time t coincide to

the elements of $\mathbf{Adj}_t$:

$$\boldsymbol{\mathcal{X}} = \begin{cases} x_{uvt} = 1 \text{ if } \mathbf{Adj}_t(u, v) = 1 , \\ x_{uvt} = 0 \text{ otherwise} . \end{cases}$$

We will take advantage of this representation to study time-varying networks through tensor decomposition techniques, as we will explain in the rest of the manuscript.

In this chapter, we introduced some key elements and questions about time-varying networks, which are necessary for the work we will describe in the following chapters. As we mentioned in the introduction we want to tackle some of the questions introduced here by using tensor decomposition techniques. This is why in the next chapter we will present the related theory.

Chapter 2

Tensor Decompositions

Tensor decomposition techniques correspond to a group of mathematical theories and methods which is a fruitful research topic in many fields, from multi-linear algebra to machine learning. The computational methods, developed to find the decomposition of tensors, turned out to be particularly adaptable to handle a broad range of applications. Thanks to their adaptability, tensor decompositions were indeed applied to tackle problems in signal processing, numerical analysis, data mining, neuroscience, computer vision, etc.

In this chapter we aim at giving an overview on the theory behind tensor decomposition techniques. The aim of this chapter is to provide the basis needed to understand how the present work is developed. We will focus on the description of the general framework for tensor decompositions and the fundamental properties of its existence and uniqueness. We will give particular attention to the Non-negative Tensor Factorization (NTF), as it is the principal technique used to develop the methods described in the following chapters.

In our work, we actively modify the computational methods and extend the NTF framework to tackle several problems related to time-varying networks. For this reason, we will give an idea of different available algorithms to compute tensor decompositions. In particular, we will explain the two main methods from which we started to develop our work: the non-linear conjugate gradient and the alternating non-negativity constraint least squares.

Finally, we will illustrate some of the used procedures and metrics to evaluate the goodness of tensor decomposition approximations, which we used to assess the quality of our results.

2.1 Tensors

Let us assume that $V_1, \ldots, V_D$ are real vector spaces, whose dimensions are $I_1, \ldots, I_D$ respectively. A tensor of order D is defined as an element of the tensor product $V_1 \circ \cdots \circ V_D$ and it is called an order-D tensor [7]. A tensor is then a mapping between two linear spaces under a change of bases. Once we choose a basis on the vector spaces, we can represent the tensor as a multi-dimensional array, which we denote as $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$. It is clear that the definition of a multi-dimensional array is not sufficient to define a tensor, as the additional definitions of spaces and bases are needed. Thus different choices of bases on the vector spaces would define different multi-dimensional array representations. Nevertheless, the use of the term tensor to mean a multi-dimensional array is widely used in data analysis fields, such as machine learning, computer vision, and neuroscience. Therefore, in the context of this work we will adopt this convention and we will always consider tensors as multi-dimensional arrays.

2.1.1 Notation

In this chapter, we provide the terminology of tensors, their operations and related definitions, which will be used throughout the overall text.

Definition 27. *(**Order**) The **order** of a tensor corresponds to the number of dimensions of the tensor. The tensor dimensions can be also called **ways** or **modes**.*

Tensors of zero-order correspond to scalar values and are represented by lower-case letters, e.g., x. Tensors of the first order correspond to vectors and are denoted by bold lower-case letters, e.g., $\mathbf{x}$ and its i-th entry is denoted as x_i. Tensors of the second order correspond to matrices and are written with bold capital letters, e.g., $\mathbf{X}$ and an entry in the position (i, j) is denoted as x_{ij}. Higher-order tensors are denoted by calligraphic letters $\boldsymbol{\mathcal{X}}$ and a value in the position $(i, j, \ldots, k)$ is denoted as $x_{ij\ldots k}$. The range of indices in an array goes from 1 to the index upper bound that is usually marked with a capital letter, e.g., $i = 1, \ldots, I$.

In this perspective a tensor is a generalization of scalars, vectors, and matrices and it can be formally defined as follows.

Definition 28. *(**Tensor**) Given the dimensions $I_1, \ldots, I_D \in \mathbb{N}$, a **tensor** $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ of order D is a D-way array, whose elements $x_{i_1 \ldots i_D}$ are indexed by $i_d \in \{i_1, \ldots, I_d\}$ for $1 \leq d \leq D$.*

We can form sub-arrays and sub-tensors by fixing a subset of the indices of the tensor. In particular we have **fibers** and **slices**.

Definition 29. *(**Fiber**) The **fiber** of a tensor is a sub-array of the tensor in which all the indices are fixed, with the exception of one.*

Thus, the column of a matrix is a mode-1 fiber and the row of a matrix is a mode-2 fiber as the first and the second index is fixed respectively.

Definition 30. *(**Slice**) The **slice** of a tensor is a sub-array of the tensor in which two indices are free and the other are fixed.*

In the case of a three-way tensor we have frontal, lateral, and horizontal slices.

2.1.2 Operations

We now need to define some of the standard operations and concepts that are used in our analysis. In particular, the Hadamard product, the Kronecker product, and the Khatri-Rao product are matrix operations needed in the current study.

The **Hadamard product** of two matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{I \times J}$, denoted by $\mathbf{A} * \mathbf{B} \in \mathbb{R}^{I \times J}$, is the element-wise product of the two matrices

$$\mathbf{A} * \mathbf{B} = \begin{pmatrix} a_{11}b_{11} & a_{12}b_{12} & \ldots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \ldots & a_{2J}b_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \ldots & a_{IJ}b_{IJ} \end{pmatrix}$$

The **Kronecker product** of two matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{K \times L}$, denoted by $\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{(IK) \times (JL)}$, is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \dots & a_{IJ}\mathbf{B} \end{pmatrix}$$

Property 1. *Let* $\mathbf{A} \in \mathbb{R}^{I \times J}$, $\mathbf{B} \in \mathbb{R}^{K \times L}$, $\mathbf{C} \in \mathbb{R}^{M \times N}$, *and* $\mathbf{D} \in \mathbb{R}^{P \times Q}$. *Then*

1. $(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T$,
2. $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$,
3. $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$,
4. $\mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C}$, *and*
5. $(\mathbf{A} \otimes \mathbf{B})^{\dagger} = \mathbf{A}^{\dagger} \otimes \mathbf{B}^{\dagger}$.

The **Khatri-Rao product** of two matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{B} \in \mathbb{R}^{J \times K}$, denoted as $\mathbf{A} \odot \mathbf{B} \in \mathbb{R}^{(IJ) \times K}$, is defined as the column-wise Kronecker product

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \dots, \mathbf{a}_K \otimes \mathbf{b}_K].$$

It is important to notice that the Khatri-Rao product of two column vectors $\mathbf{a} \in \mathbb{R}^{I \times 1}$ and $\mathbf{b} \in \mathbb{R}^{I \times 1}$ is equal to their Kronecker product

$$\mathbf{a} \odot \mathbf{b} = \mathbf{a} \otimes \mathbf{b} = vec\left(\left(ab^T\right)^T\right). \tag{2.1}$$

Property 2. *Let* $\mathbf{A} \in \mathbb{R}^{I \times L}$, $\mathbf{B} \in \mathbb{R}^{J \times L}$, *and* $\mathbf{C} \in \mathbb{R}^{K \times L}$. *Then*

1. $\mathbf{A} \odot \mathbf{B} \odot \mathbf{C} = (\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C} = \mathbf{A} \odot (\mathbf{B} \odot \mathbf{C})$,
2. $(\mathbf{A} \odot \mathbf{B})^T (\mathbf{A} \odot \mathbf{B}) = \mathbf{A}^T\mathbf{A} * \mathbf{B}^T\mathbf{B}$, *and*
3. $(\mathbf{A} \odot \mathbf{B})^{\dagger} = (\mathbf{A}^T\mathbf{A} * \mathbf{B}^T\mathbf{B})^{\dagger} (\mathbf{A} \odot \mathbf{B})^T$

The result of an **outer product** between two vectors $\mathbf{a} \in \mathbb{R}^{I \times 1}$ and $\mathbf{b} \in \mathbb{R}^{J \times 1}$ is a matrix $\mathbf{X} = \mathbf{ab}^T$ of sizes $(I \times J)$. We use the notation $\mathbf{X} = \mathbf{a} \circ \mathbf{b}$, adopted

by Kolda [92], Kolda and Bader [8], to generalize the outer product to more than two vectors and be consequently able to define higher-order tensor via this product.

Let $\mathcal{D} = \{1, \ldots, D\}$ and $\mathbf{a}^{(d)} \in \mathbb{R}^{I_d}$, $\forall d \in \mathcal{D}$, then the outer product of the considered vectors is a D-order tensor $\boldsymbol{\mathcal{X}}$ whose elements are defined as

$$x_{i_1,\ldots,i_D} = \left(\mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(D)}\right)_{i_1 \ldots i_D} = a^{(1)}_{i_1} a^{(2)}_{i_2} \ldots a^{(D)}_{i_D}, \text{ with } 1 \leq i_d \leq I_d, \forall d \in \mathcal{D}$$

The d-**mode product** between a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and a matrix $\mathbf{A} \in \mathbb{R}^{I \times I_d}$ can be written by $(\boldsymbol{\mathcal{X}} \times_d \mathbf{A}) \in \mathbb{R}^{I_1 \times \ldots I_{d-1} \times I \times I_{d+1} \times \cdots \times I_D}$ and its elements are given by

$$(\boldsymbol{\mathcal{X}} \times_d \mathbf{A})_{i_1 \ldots i_{d-1} i i_{d+1} \ldots i_D} = \sum_{i_d=1}^{I_d} x_{i_1 \ldots i_D} a_{i i_d} \, .$$

The **matricization** of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ in one mode is the indexing of the elements in the tensor to reshape it into a matrix. Formally, the matricized form of the tensor $\boldsymbol{\mathcal{X}}$, of dimension $I_{\mathcal{D}=\{1,\ldots,D\}}$, is written as $\mathbf{X}_{(\mathcal{R} \times \mathcal{C}: I_{\mathcal{D}})} \in \mathbb{R}^{J \times K}$ where

$$J = \prod_{d \in \mathcal{R}} I_d \text{ and } K = \prod_{d \in \mathcal{C}} I_d \, ,$$

with $\mathcal{R} = \{r_1, \ldots, r_L\}$ and $\mathcal{C} = \{c_1, \ldots, c_M\}$ that are partitioning of the dimensions $\mathcal{D} = 1, \ldots, D$. The indices corresponding to $\mathcal{R}$ and $\mathcal{C}$ are respectively mapped to the rows and the columns of the resulting matrix, so that

$$\left(\mathbf{X}_{(\mathcal{R} \times \mathcal{C}: I_{\mathcal{D}})}\right)_{ij} = x_{i_1 \ldots i_D}$$

with

$$j = 1 + \sum_{l=1}^{L} \left[(i_{r_l} - 1) \prod_{l'=1}^{l-1} I_{r_{l'}} \right] \text{ and } k = 1 + \sum_{m=1}^{M} \left[(i_{r_m} - 1) \prod_{m'=1}^{m-1} I_{r_{m'}} \right] .$$

A special case of the matricization operation is the d-**mode matricization** of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$:

$$\mathbf{X}_{(d)} = \mathbf{X}_{(\mathcal{R} \times \mathcal{C}: I_{\mathcal{D}})} \, ,$$

where $\mathcal{R} = \{d\}$ and $\mathcal{C} = \{1, \ldots, d-1, d+1, \ldots, D\}$. Moreover, we can convert a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ in a vector:

$$\text{vec}\,(\boldsymbol{\mathcal{X}}) := \mathbf{X}_{(\mathcal{D} \times \emptyset : I_{\mathcal{D}})} \,.$$

Finally is useful to introduce the notions of the norm and inner product of a tensor that will be necessary to define the tensor decomposition problem. The **inner product** between two tensors $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is defined as

$$\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \text{vec}\,(\boldsymbol{\mathcal{X}})^T \,\text{vec}\,(\boldsymbol{\mathcal{Y}}) = \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} x_{i_1 \ldots i_D} y_{i_1 \ldots i_D} \,.$$

The **Frobenius norm** of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ can be written in terms of the inner product between the tensor and itself, so that

$$\|\boldsymbol{\mathcal{X}}\|_F^2 = \langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{X}} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} |x|_{i_1 \ldots i_D}^2 \,. \tag{2.2}$$

Property 3. *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and $\mathcal{D} = \{1, \ldots, D\}$:*

1. *let sets $\mathcal{R}$ and $\mathcal{C}$ be a partition of $\mathcal{D}$, then $\|\boldsymbol{\mathcal{X}}\|_F = \|\mathbf{X}_{(\mathcal{R} \times \mathcal{C} : I_{\mathcal{D}})}\|_F$,*
2. *let $d \in \mathcal{D}$, then $\|\boldsymbol{\mathcal{X}}\|_F = \|\mathbf{X}_{(d)}\|_F$,*
3. *$\|\boldsymbol{\mathcal{X}}\|_F = \|vec\,(\boldsymbol{\mathcal{X}})\,\|_2$.*

Property 4. *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, then*

$$\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{Y}}\|_F^2 = \|\boldsymbol{\mathcal{X}}\|_F^2 + \|\boldsymbol{\mathcal{Y}}\|_F^2 - 2\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle \,.$$

Property 5. *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, with $\boldsymbol{\mathcal{X}} = \mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(D)}$ and $\boldsymbol{\mathcal{Y}} = \mathbf{b}^{(1)} \circ \cdots \circ \mathbf{b}^{(D)}$, then*

$$\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \prod_{d=1}^{D} \langle \mathbf{a}^{(d)}, \mathbf{b}^{(d)} \rangle \,.$$

2.2 Decompositions

The idea of decomposing a tensor into the sum of outer products of vectors is originally attributed to Hitchcock [93], that studied the problem in 1927.

Successively, many other researchers tackled the problem of multi-way factor analysis as Cattell [94] in 1944 and Tucker [95, 96, 97] in the 1960s, whose work brought to the forefront the interest of tensor decomposition techniques mainly in psychometric fields, algebraic statistics, and quantum mechanics.

Nevertheless, tensor decomposition techniques as well as their computation and implementation became really popular after the works of Harshman [98], Harshman and Lundy [99], who faced probably for the first time the related tensor approximation problem, by defining the so called Parallel Factor Analysis (PARAFAC) model. In parallel, Carroll and Chang [100] devised the Canonical Decomposition (CANDECOMP) model. Nowadays, tensor decompositions and their approximations received increasing attention and interest in a wide variety of fields [101]. Examples include signal processing, numerical linear algebra, computer vision, numerical analysis, data mining, machine learning, graph and network analysis, neuroscience, and more. These two models are the most known in the literature and usually their names are combined as CANDECOMP/PARAFAC (CP) decomposition, whose initials reminds also to the Canonical Polyadic decomposition.

Both CANDECOMP, PARAFAC and Tucker tensor decompositions extend the ideas and methods of two-ways factor analysis to analyse higher-order data. Thus, these methods can be considered as a generalization of the matrix singular value decomposition (SVD) [102] and principal component analysis (PCA) [103]. A key motivation for the use of these higher-order models is that they enable to simultaneously analyse several matrices in parallel (i.e. tensor slices), that could lead to the recovery of a unique set of factors that could not be uncover by studying the matrices separately or by studying a certain combination of them. To formally define the CP decomposition we introduce in the next section the Kruskal operator which provides a shorthand notation for the sum of the outer products of vectors that is needed for the tensor decomposition.

2.2.1 Kruskal Operator

By using the definition of operators introduced by [92], we can denote in a concise way a series of d-mode multiplication operations as the **Tucker**

operator and its special case: the **Kruskal operator** [104]. Let $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, with $\mathcal{D} = \{1, \ldots, D\}$ and suppose to have D matrices $\mathbf{A}^{(d)} \in \mathbb{R}^{J_d \times I_d}$ $\forall d \in \mathcal{D}$, then the Tucker operator is defined as

$$[\![\boldsymbol{\mathcal{G}}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] = \boldsymbol{\mathcal{G}} \times_1 \mathbf{A}^{(1)} \times_2 \cdots \times_D \mathbf{A}^{(D)} ,$$

whose result is a tensor of size $I_1 \times \cdots \times I_D$.

The **Kruskal operator** is an efficient representation for multi-dimensional multiplication of matrices that can be formally defined as follows. Let $\mathcal{D} = \{1, \ldots, D\}$ and suppose to have $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R}$ $\forall d \in \mathcal{D}$. Then the Kruskal operator is defined as:

$$[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] = [\![\boldsymbol{\mathcal{L}}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] = \boldsymbol{\mathcal{L}} \times_1 \mathbf{A}^{(1)} \times_2 \cdots \times_D \mathbf{A}^{(D)} ,$$

where $\boldsymbol{\mathcal{L}}$ is a D-order identity tensor of size $R \times \cdots \times R$, that is a tensor having ones along the super-diagonal and zeros otherwise. The result of this operation correspond to a tensor having size $I_1 \times \cdots \times I_D$. The following fundamental properties show the relation between the Kruskal operator, the matricization operation, and the Khatri-Rao product. These properties will be useful to compute the CP decomposition.

Property 6. *Let us consider the set $\mathcal{D} = \{1, \ldots, D\}$ and the matrices $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R}$ $\forall d \in \mathcal{D}$:*

1. *if we take into account the partitions $\mathcal{R} = \{r_1, \ldots, r_L\}$ and $\mathcal{C} = \{c_1, \ldots, c_M\}$ of $\mathcal{D}$, then*

$$\mathcal{X} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \iff \mathbf{X}_{(\mathcal{R} \times \mathcal{C}: I_{\mathcal{D}})} = \left(\mathbf{A}^{(r_L)} \odot \cdots \odot \mathbf{A}^{(r_1)}\right) \left(\mathbf{A}^{(c_M)} \odot \cdots \odot \mathbf{A}^{(c_1)}\right)^T .$$

   If $\mathcal{R} = \varnothing$ or $\mathcal{C} = \varnothing$ then respectively the first or the second multiplicand is replaced by a vector of ones of length R:

$$\mathbf{X}_{(\varnothing \times \mathcal{D}: I_{\mathcal{D}})} = \mathbf{1}^T \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(1)}\right)^T ,$$
$$\mathbf{X}_{(\mathcal{D} \times \varnothing: I_{\mathcal{D}})} = \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(1)}\right) \mathbf{1} .$$

2. For any $d \in \mathcal{D}$,

$$\mathcal{X} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \iff \mathbf{X}_{(d)} = \mathbf{A}^{(d)} \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right)^T .$$

Finally we introduce the norm of a Kruskal operator that has a particular form as it can be reduced to the sum of the entries of the Hadamard product of D matrices of size $R \times R$.

Property 7. *Let $\mathcal{D} = \{1, \ldots, D\}$ and $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R}$ $\forall d \in \mathcal{D}$. Then*

$$\left\| [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|^2 = \sum_{r=1}^{R} \sum_{r'=1}^{R} \left(\left(\mathbf{A}^{(1)T} \mathbf{A}^{(1)} \right) * \cdots * \left(\mathbf{A}^{(D)T} \mathbf{A}^{(D)} \right) \right)_{rr'}$$

Property 8. *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, $\mathcal{D} = \{1, \ldots, D\}$, and $\mathbf{A}^{(d)}$ $\forall d \in \mathcal{D}$, then*

1. the inner product of $\boldsymbol{\mathcal{X}}$ and the Kruskal operator leads to

$$\langle \boldsymbol{\mathcal{X}}, [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \rangle = \langle vec\boldsymbol{\mathcal{X}}, \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(1)}\right) \mathbf{1} \rangle .$$

2. The norm of the difference of a tensor $\boldsymbol{\mathcal{X}}$ and a Kruskal tensor is:

$$\left\| \boldsymbol{\mathcal{X}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|^2 = \\ = \|\boldsymbol{\mathcal{X}}\|^2 + \left\| [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|^2 - 2\langle \boldsymbol{\mathcal{X}}, [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \rangle .$$

2.2.2 CP Decomposition

In 1927 Hitchcock proposed the idea of expressing a tensor as the sum of finite number of outer products of vectors (rank-one tensors), the so called polyadic form of a tensor. Subsequently in 1944 Cattell proposed some ideas for parallel proportional analysis and the idea of multiple axes for analysis. Nevertheless, these concepts became popular after a third introduction in the 1970s in the psychometrics community, under the form of the CANDECOMP by Carroll and Chang, and the PARAFAC by Harshman.

The CP decomposition factorizes a multi-dimensional tensor as the sum of elementary pieces that we call components, having the form of rank-one tensors. Given a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, the CP decomposition aims at writing the tensor as

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \,, \tag{2.3}$$

where R is a positive integer, the rank-one tensors $\mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$ are called **components** and the matrices $\mathbf{A} \in \mathbb{R}^{I \times R}, \mathbf{B} \in \mathbb{R}^{J \times R}, \mathbf{C} \in \mathbb{R}^{K \times R}$ are called the **factor matrices** (or **loading matrices**). The factor columns of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ correspond respectively to the vectors $\mathbf{a}_r \in \mathbb{R}^{I \times 1}$, $\mathbf{b}_r \in \mathbb{R}^{J \times 1}$, and $\mathbf{c}_r \in \mathbb{R}^{K \times 1}$ $\forall r = 1, \ldots, R$, so that

$$\begin{aligned} \mathbf{A} &= [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_R] \,, \\ \mathbf{B} &= [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_R] \,, \\ \mathbf{C} &= [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_R] \,. \end{aligned}$$

The decomposition can be also written element-wise as

$$x_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \text{ with } i = 1, \ldots, I, j = 1, \ldots, J, \text{ and } k = 1, \ldots, K \,.$$

By using this definition it is possible to rewrite the problem in its matricized form, one for each mode of the tensor:

$$\begin{aligned} \mathbf{X}_1 &= \mathbf{A} \left(\mathbf{C} \odot \mathbf{B} \right)^T \,, \\ \mathbf{X}_2 &= \mathbf{B} \left(\mathbf{C} \odot \mathbf{A} \right)^T \,, \\ \mathbf{X}_3 &= \mathbf{C} \left(\mathbf{B} \odot \mathbf{A} \right)^T \,. \end{aligned}$$

It is often useful to normalize (with the Frobenius norm) the columns of the factor matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ to length one with the weights included in the core tensor $\boldsymbol{\mathcal{L}}$ that will have the weight values on the diagonal $\boldsymbol{\lambda} \in \mathbb{R}^R$, so that we can rewrite the decomposition as

$$\boldsymbol{\mathcal{X}} = [\![\boldsymbol{\mathcal{L}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \,.$$

We will focus on the decomposition of three-dimensional tensors, however it is possible to generalize the CP decomposition to higher-order tensors. Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ be a D-dimensional tensor, its CP decompositions is

$$\boldsymbol{\mathcal{X}} = [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}]\!] = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \ ,$$

where $\boldsymbol{\lambda} \in \mathbb{R}^R$ and $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R}$ for $d = 1, \ldots, D$. In this particular case, the matricized form becomes

$$\boldsymbol{\mathcal{X}}_{(d)} = \mathbf{A}^{(d)} \boldsymbol{\Lambda} \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)} \right)^T \ ,$$

where $\boldsymbol{\Lambda} = \operatorname{diag}(\boldsymbol{\lambda})$.

2.3 Rank, Uniqueness, and Existence

As we have seen in the previous sections, the CANDECOMP/PARAFAC decompositions ask for a solution to the problem of representing a tensor through the sum of rank-one tensors. However, in practice measurements are always corrupted by some noise, which makes the CP decomposition not unique in general. This is the reason why finding the solution of the CP decomposition means to compute the best rank-R approximation. Given a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, we look for an optimal rank-R approximation of $\boldsymbol{\mathcal{X}}$ of the form

$$\mathbf{App}_R \in \operatorname{argmin}_{\operatorname{rank}(\mathbf{App}) \leq R} \| \boldsymbol{\mathcal{X}} - \mathbf{App} \| \ ,$$

in other words, as we introduced in the previous section, it means to find the scalars λ_r and the unit vectors $\mathbf{a}_r^{(d)}$ with $d = 1, \ldots, D$ and $r = 1, \ldots, R$, that minimize the distance

$$\left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\| \ .$$

The norm $\|\cdot\|$ here is arbitrary and we now discuss several natural choices in the next section. As the decomposition is an approximation of an original tensor the concepts of rank, uniqueness, and existence are all fundamental. These concepts are indeed useful to analyse and assess the quality of the result given

by the approximation and enable to understand if the considered problem is well-posed or ill-posed.

2.3.1 The choice of the norm

The **Frobenius norm** or **F-norm**, defined in Eq. (2.2) is the most popular choice of norms for tensors in data analytic applications and it is the one that we will use throughout this dissertation. However, there are different norms that can be used and that can be used to better interpret the normalized values of $\boldsymbol{\mathcal{X}}$ in the case in which this tensor is non-negative. This is the case of the **E-norm** and the **G-norm**, that are defined as

$$\|\boldsymbol{\mathcal{X}}\|_E = \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} | x_{i_1 \ldots i_D} |$$
$$\|\boldsymbol{\mathcal{X}}\|_G = \max\{| x_{i_1 \ldots i_D} \,||\, i_1 = 1, \ldots, I_1; \ldots; i_D = 1, \ldots, I_D\} .$$

Note that the norms defined above are equivalent to the l^1-,l^2-, and $l^\infty-$norms of the tensor $\boldsymbol{\mathcal{X}}$ regarded as a vector of size $I_1 I_2 \ldots I_D$.

Property 9. *The E-, F-, and G-norms are multiplicative on rank-one tensors:*

$$\begin{aligned} \|\mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(D)}\|_E &= \|\mathbf{a}^{(1)}\|_1 \|\mathbf{a}^{(2)}\|_1 \ldots \|\mathbf{a}^{(D)}\|_1 , \\ \|\mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(D)}\|_F &= \|\mathbf{a}^{(1)}\|_2 \|\mathbf{a}^{(2)}\|_2 \ldots \|\mathbf{a}^{(D)}\|_2 , \\ \|\mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(D)}\|_G &= \|\mathbf{a}^{(1)}\|_\infty \|\mathbf{a}^{(2)}\|_\infty \ldots \|\mathbf{a}^{(D)}\|_\infty . \end{aligned} \tag{2.4}$$

The Frobenius norm has the advantage of being induced by an inner product on $\mathbb{R}^{I_1 \times \cdots \times I_D}$:

$$\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} x_{i_1 \ldots i_D} y_{i_1 \ldots i_D} .$$

Thus, it is possible to write the Cauchy-Schwarz inequality

$$| \langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle | \leq \|\boldsymbol{\mathcal{X}}\|_F \|\boldsymbol{\mathcal{Y}}\|_F$$

and the Hölder inequality

$$|\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle| \leq \|\boldsymbol{\mathcal{X}}\|_E \|\boldsymbol{\mathcal{Y}}\|_G .$$

2.3.2 Rank Definition and Properties

The rank is a fundamental property of matrices: given a matrix $\mathbf{A} \in \mathbb{R}^{I \times R}$, its **rank** is defined as

$$r_{\mathbf{A}} := \operatorname{rank}(\mathbf{A}) = r$$

if and only if contains at least one collection of r linearly independent columns and this fails for $r+1$ columns.

It is natural to understand how the concept of rank generalize to three-dimensional arrays or higher-order tensors. A generalization of the rank definition was first studied by Kruskal [104, 105] in the 1970s and further discussed to study the uniqueness property for multi-dimensional arrays [106].

Considering the CP decomposition of a tensor $\boldsymbol{\mathcal{X}}$, the **rank** R is the number of factors in the decomposition, and more specifically is the smallest number R such that $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ has a D-adic decomposition of rank R, with $\mathcal{D} = \{1, \ldots, D\}$:

$$\operatorname{rank}(\boldsymbol{\mathcal{X}}) := \min\{R \mid \boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)}\} . \tag{2.5}$$

This rank definition can be easily conducted to the usual definition of a matrix rank, and some of the properties of the rank of matrices can be generalized to higher order.

Property 10. *Let $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ be tensors of any order, then*

1.

$$rank(\boldsymbol{\mathcal{X}}) = rank(k\boldsymbol{\mathcal{X}}) \quad for\ any\ \ k \neq 0 ,$$

2.

$$rank(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{Y}}) \leq rank(\boldsymbol{\mathcal{X}}) + rank(\boldsymbol{\mathcal{Y}}) .$$

However, matrix and tensor ranks are quite different:

1. the rank of a real-valued tensor may actually be different over the fields $\mathbb{R}$ and $\mathbb{C}$ as we can see in Kolda [92];

2. there is no algorithm to determine the rank in (2.5) of a given tensor and this problem was shown to be NP-hard by Håstad [107];

3. let $R_{max}(I, J)$ and $R_{max}(I, J, K)$ be **maximum ranks**, i.e. the largest attainable rank, of $I \times J$ and $I \times J \times K$ arrays respectively. In the case of a matrix this is well-known as $R_{max}(I, J) = \min\{I, J\}$, however $R_{max}(I, J, K)$ is unknown or difficult to determine. The following inequalities hold:

$$\max\{I, J, K\} \leq R_{max}(I, J, K) \leq \min\{IJ, IK, JK\} ;$$

4. in the case of matrices of size $I \times J$ the **maximum rank** and the **typical rank**, i.e. the rank that occurs with probability greater than zero, coincide, whereas for tensors these two values could be different.

2.3.3 Uniqueness conditions

An important property that is needed to be analysed in the study of tensor decompositions is uniqueness. The decomposition of higher-order tensors is often unique, whereas it is not the case for matrix decompositions. The problem of tensor decomposition uniqueness was first tackled by Harshman [98] in 1970s and Kruskal [104, 105] in 1980s. Here we report some of the main results achieved and discuss the case in which the approximation problem is well-posed.

Let us consider the CP decomposition of a three-dimensional tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, whose approximation can be written by the Kruskal tensor $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. There exist some elementary operations, that we can apply on the Kruskal tensor, which do not change the final result, thus leading to indeterminacies:

1. we can permute or relabel the factors of the outer products of the decomposition, as the rank-one components can be reordered in an arbitrary

way

$$\boldsymbol{\mathcal{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathbf{AP}, \mathbf{BP}, \mathbf{CP}]\!] \text{ for any } R \times R \text{ permutation matrix } \mathbf{P} ;$$

2. we can insert some multipliers to rescale the individual vectors in the outer products such that their multiplication does not affect the final result
$$\boldsymbol{\mathcal{X}} = \sum_{r_1}^{R} (\alpha_r \mathbf{a}_r) \circ (\beta_r \mathbf{b}_r) \circ (\gamma_r \mathbf{c}_r) \text{ s.t. } \alpha_r \beta_r \gamma_r = 1 \ \forall r .$$

Two 3-ways decompositions are said to be **equivalent** if they have the same rank and one can be obtained by the other after a rescaling or a permutation. Moreover, a rank R decomposition of a tensor $\boldsymbol{\mathcal{X}}$ is **rotationally unique** (often called simply **unique**) if all the rank R decompositions of $\boldsymbol{\mathcal{X}}$ are equivalent between themselves.

The most known result on tensor decomposition uniqueness is attributed to Kruskal [104, 105], who used the concept of κ**-rank**, also known as Kruskal rank, whose term was later introduced by Harshman and Lundy in 1984. The κ**-rank** $\kappa_{\mathbf{A}}$ of a matrix $\mathbf{A} \in \mathbb{R}^{I \times R}$ is defined as the maximum value κ such that any κ columns of the matrix are linearly independent:

$$\kappa_{\mathbf{A}} := \text{the } \kappa\text{-rank of } \mathbf{A} = r$$

if and only if every r columns are linearly independent, and this fails for at least one set of $r + 1$ columns. Thus

$$\kappa_{\mathbf{A}} \leq r_{\mathbf{A}} \leq \min(I, R) \ \forall \mathbf{A} .$$

Theorem 1. *Let* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathbf{A}', \mathbf{B}', \mathbf{C}']\!]$*, whose matrices have* R *columns, and let* $\kappa_{\mathbf{A}}, \kappa_{\mathbf{B}}, \kappa_{\mathbf{C}}$ *be the* κ*-ranks of* $\mathbf{A}, \mathbf{B}, \mathbf{C}$*, such that* $\kappa_{\mathbf{A}} + \kappa_{\mathbf{B}} + \kappa_{\mathbf{C}} \geq 2R + 2$*. Then* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ *is equivalent to* $[\![\mathbf{A}', \mathbf{B}', \mathbf{C}']\!]$*.*

The most useful result on the uniqueness is given by the following corollary.

Corollary 1. *Consider a Kruskal tensor* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$*, whose factors have* R *columns each. Let* $\kappa_{\mathbf{A}}, \kappa_{\mathbf{B}}, \kappa_{\mathbf{C}}$ *be the* κ*-ranks of* $\mathbf{A}, \mathbf{B}, \mathbf{C}$*.*

If $\kappa_{\mathbf{A}} + \kappa_{\mathbf{B}} + \kappa_{\mathbf{C}} \geq 2R + 2$*, then* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ *is rotationally unique.*

Sidiropoulos and Bro [108] successively extended this result to the general case of D-way tensors (where $D > 3$), by using the definition of the κ-rank of the Khatri-Rao product of two matrices $\mathbf{A}$ and $\mathbf{B}$:

Property 11. *Let* $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_R] \in \mathbb{R}^{I \times R}$, $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_R] \in \mathbb{R}^{J \times R}$, *and*

$$\mathbf{B} \odot \mathbf{A} = [\mathbf{b}_1 \otimes \mathbf{a}_1, \ldots, \mathbf{b}_R \otimes \mathbf{a}_R] \ .$$

1. *If* $\kappa_{\mathbf{A}} \geq 1$ *and* $\kappa_{\mathbf{B}} \geq 1$, *then*

$$\kappa_{\mathbf{B} \odot \mathbf{A}} \geq \min\{\kappa_{\mathbf{A}} + \kappa_{\mathbf{B}} - 1, R\} \ ;$$

2. *if* $\kappa_{\mathbf{A}} = 0$ *or* $\kappa_{\mathbf{B}} = 0$, *then*

$$\kappa_{\mathbf{B} \odot \mathbf{A}} = 0 \ .$$

By using this result it is possible to extend the uniqueness sufficient condition to higher-order tensors (for the complete proof please refer to [108]). Thus, given a D-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ with rank R and CP decomposition equal to

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}]\!] \ , \tag{2.6}$$

a sufficient condition for uniqueness is

$$\sum_{d=1}^{D} \kappa_{\mathbf{A}^{(d)}} \geq 2R + (D - 1) \ .$$

This condition was shown to be also necessary in the cases of $R = 2, 3$ but not for $R > 3$. General necessary conditions were then considered by Liu and Sidiropoulos, who have shown that if

$$\min\{\operatorname{rank}(\mathbf{A} \odot \mathbf{B}), \operatorname{rank}(\mathbf{A} \odot \mathbf{C}), \operatorname{rank}(\mathbf{B} \odot \mathbf{C})\} = R \ ,$$

then the CP decomposition in (2.3) is unique.

This condition was shown to be valid also for higher-order tensor decompositions, so that a necessary condition of uniqueness for the CP decomposition of a D-way tensor in (2.6) is

$$\min_{d=1,\ldots,D} \left\{\operatorname{rank}\left(\mathbf{A}^{(1)} \odot \cdots \odot \mathbf{A}^{(d-1)} \odot \mathbf{A}^{(d+1)} \odot \cdots \odot \mathbf{A}^{(D)}\right)\right\} = R \ .$$

Finally, by using the fact that

$$\operatorname{rank}(\mathbf{A} \odot \mathbf{B}) \leq \operatorname{rank}(\mathbf{A} \otimes \mathbf{B}) \leq \operatorname{rank}(\mathbf{A}) \cdot \operatorname{rank}(\mathbf{B}) \ ,$$

another necessary condition for D-way tensor decompositions is

$$\min_{d=1,\ldots,D} \left\{ \prod_{\substack{d'=1 \\ d' \neq d}}^{D} \operatorname{rank}\left(\mathbf{A}^{(d')}\right) \right\} \geq R$$

2.3.4 PARAFAC degeneracy

One of the most fundamental issues, which everyone might incur, is the fact that the problem of finding the best rank-R approximation for a tensor of order 3 or higher has generally no solution. Thus, considering a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ such that its approximation is given by

$$\inf \left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\| \ ,$$

it is not attained by any choice of λ_r and $\mathbf{a}_r^{(1)}, \mathbf{a}_r^{(2)}, \ldots, \mathbf{a}_r^{(D)}$. Usually, it is also not possible to determine if a given tensor $\boldsymbol{\mathcal{X}}$ a priori will fail to have a best approximation. Moreover, this failure appears to happen in the case of different dimensions, ranks, and tensor orders, as shown by De Silva and Lim [109] and can occur also with high probability, i.e. there is the certainty that the infimum (greatest lower bound) will never be reached. This problem also extends to the case of symmetric tensors [110]. For these reason, often the problem, which we are looking for a solution, is ill-posed. This phenomenon, which is known as the **PARAFAC degeneracy**, was investigated by Bini et al. [111] and separately also by Kruskal et al. [112].

Nevertheless, it was shown by Qi et al. [113], Lim and Comon [10] that the nonexistence of a globally optimal solution for higher-order tensor decompositions can be overcome by the introduction of some constraints to the problem. In particular, they have shown that the PARAFAC degeneracy can be avoided when a non-negative PARAFAC model is fitted.

2.3.5 Non-negative Tensor Factorization

A D-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is said to be **non-negative** if all its elements are non-negative $x_{i_1,\ldots,i_D} \geq 0 \; \forall i_d = 1, \ldots, I_d$ with $d = 1, \ldots, D$. A non-negative tensor is denoted as $\boldsymbol{\mathcal{X}} \geq 0$ and the related **non-negative outer-product decomposition**, also called **Non-negative Tensor Factorization** (NTF) [9, 114] is given by

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \,, \tag{2.7}$$

where $\lambda_r \geq 0$ and $\mathbf{a}_r^{(1)}, \mathbf{a}_r^{(2)}, \ldots, \mathbf{a}_r^{(D)} \geq 0 \; \forall r$. Analogously, the problem in (2.7) can be written through the Kruskal tensor notation as

$$\boldsymbol{\mathcal{X}} = [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}]\!] \text{ s.t. } \boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)} \geq 0 \,. \tag{2.8}$$

This decomposition exists for any non-negative tensor $\boldsymbol{\mathcal{X}} \geq 0$ and the lowest R for which such a decomposition is possible it said to be its **non-negative rank**. Formally, given $\boldsymbol{\mathcal{X}} \geq 0$ its non-negative rank is defined as

$$\operatorname{rank}_+ (\boldsymbol{\mathcal{X}}) := \min \left\{ R \mid \boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)} \;\; \forall r \right\} \,.$$

Existence of a Solution for the NTF

As proved by Lim and Comon [10] non-negativity constraints are often a natural choice to avoid the PARAFAC degeneracy and thus looking for a solution to a problem that is well-posed. Their proof started from an empirical evidence in the studies carried out by Bro, who revealed that the PARAFAC degeneracy was never observed when fitting the model with non-negative valued data. Subsequently Harshman [115] conjectured that this is always the case and a part of the proof would be to demonstrate the existence of a global minimum over a non-compact feasible region. As this is not immediate, Lim and Comon started from the following consequence of the extreme value theorem, to show that all sub-level sets of the non-negative PARAFAC loss function are compact:

If a continuous real-valued function has a non-empty compact sub-level set, then it has to attain its infimum.

Let Δ^d denote a **unit d-simplex**, that is the convex hull of the standard basis vectors $\mathbf{e}_r$ in $\mathbb{R}^{d+1}$:

$$\Delta^d := \left\{ \sum_{r=1}^{d+1} \lambda_r \mathbf{e}_r \in \mathbf{R}^{d+1} \,\middle|\, \sum_{r=1}^{d+1} \lambda_r = 1, \lambda_1, \ldots, \lambda_{d+1} \geq 0 \right\} = \left\{ \mathbf{x} \in \mathbb{R}_+^{d+1} \middle|\, \|\mathbf{x}\|_1 = 1 \right\} .$$

We report below the proof of the following theorem as it is in [10]. Here, the proof uses the E-norm but the results can be extended to different norms as the Frobenius norm.

Theorem 2. *Let the tensor $\boldsymbol{\mathcal{X}}$ be non-negative, then*

$$\inf \left\{ \left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\|_E \,\middle|\, \boldsymbol{\lambda} \in \mathbb{R}_+^R, \mathbf{a}_r^{(1)} \in \Delta^{I_1 - 1}, \ldots, \mathbf{a}_r^{(D)} \in \Delta^{I_D - 1} \ \forall r \right\}$$

is attained.

Proof. Recall that $\mathbb{R}_+^d = \{\mathbf{x} \in \mathbb{R}^d | \ \mathbf{x} \geq 0\}$ and $\Delta^{d-1} = \{\mathbf{x} \in \mathbb{R}_+^d | \ \|\mathbf{x}\|_1 = 1\}$. Let us define the loss function $f : \mathbb{R}^R \times \left(\mathbb{R}^{I_1} \times \cdots \times \mathbb{R}^{I_D}\right)^R \to \mathbb{R}$ by

$$\begin{aligned} f\left(\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}\right) &:= \left\| \boldsymbol{\mathcal{X}} - [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|_E = \\ &= \left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\|_E . \end{aligned}$$

Let $\mathcal{D}$ be a subset of $\mathbb{R}^R \times \left(\mathbb{R}^{I_1} \times \cdots \times \mathbb{R}^{I_D}\right)^R = \mathbb{R}^{R(1+I_1+\cdots+I_D)}$ defined as

$$\mathcal{D} := \mathbb{R}_+^R \times \left(\Delta^{I_1 - 1} \times \cdots \times \Delta^{I_D - 1}\right)^R ,$$

closed and unbounded. Let the infimum be

$$\mu := \inf \left\{ f\left(\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}\right) \middle| \left(\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}\right) \in \mathcal{D} \right\} .$$

We want to show that the sub-level set of f restricted to $\mathcal{D}$,

$$\mathcal{E}_\alpha \left\{ \left(\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}\right) \in \mathcal{D} \middle| \, f\left(\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)}\right) \leq \alpha \right\}$$

is compact $\forall \alpha > \mu$ and thus the infimum of f on $\mathcal{D}$ must be attained. The set $\mathcal{E}_\alpha = \mathcal{D} \cap f^{-1}(-\infty, \alpha]$ is **closed** as f is continuous by the norm continuity. Now we need to show that $\mathcal{E}_\alpha$ is also bounded. Let us suppose the contrary

and denote $T = (\boldsymbol{\lambda}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(D)})$ the argument of f. Then, there exists a sequence $(T_d)_{d=1}^{\infty} \subset \mathcal{D}$ with $\|T_d\|_1 \to \infty$ but $f(T_d) \leq \alpha \ \forall d$. We know that $\|T_d\|_1 \to \infty$ implies that $\lambda_r^{(d)} \to \infty$ for at least one $r \in \{1, \ldots, R\}$. Note that

$$f(T) \geq \left| \|\boldsymbol{\mathcal{X}}\|_E - \left\| \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\|_E \right| .$$

Since all terms involved in the last term are non-negative, we have

$$\begin{aligned}
\left\| \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\|_E &= \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} \sum_{r=1}^{R} \lambda_r a_{ri_1} a_{ri_2} \ldots a_{ri_D} \\
&\geq \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} \lambda_p a_{pi_1} a_{pi_2} \ldots a_{pi_D} \\
&= \lambda_p \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} a_{pi_1} a_{pi_2} \ldots a_{pi_D} \\
&= \lambda_p \left\| \mathbf{a}_p^{(1)} \circ \mathbf{a}_p^{(2)} \circ \cdots \circ \mathbf{a}_p^{(D)} \right\|_E \\
&= \lambda_p \left\| \mathbf{a}_p^{(1)} \right\|_1 \left\| \mathbf{a}_p^{(2)} \right\|_1 \left\| \mathbf{a}_p^{(D)} \right\|_1 \\
&= \lambda_p ,
\end{aligned}$$

where at least two equalities follow from (2.4) and $\|\mathbf{a}^{(1)}\|_1 = \|\mathbf{a}^{(2)}\|_1 = \cdots = \|\mathbf{a}^{(D)}\|_1 = 1$. Hence, as $\lambda_p^{(d)} \to \infty$, then $f(T_d) \to \infty$. This contradicts the assumption $f(T_d) \leq \alpha \ \forall d$. □

It is worth noting that the proof would fail if the elements composing the sum of rank-1 terms are not unit-vectors, as they could be rescaled by non-zero positive scalars whose product gives one, as

$$\alpha \mathbf{a}^{(1)} \circ \beta \mathbf{a}^{(2)} \circ \cdots \circ \zeta \mathbf{a}^{(D)} , \ \alpha\beta \ldots \zeta = 1 .$$

We avoided this case by requiring that $\mathbf{a}^{(1)}, \mathbf{a}^{(2)}, \ldots, \mathbf{a}^{(D)}$ are unit vectors and $\boldsymbol{\lambda}$ is the vector containing their amplitude.

Finally, the following corollary states that the Theorem 2 is also valid for different norms as the Frobenius one.

Corollary 2. *Let* $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ *be non-negative and* $\|\cdot\| : \mathbb{R}^{I_1 \times \cdots \times I_D} \to [0, \infty)$ *be an arbitrary norm. Then*

$$\inf \left\{ \left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)} \right\| \;\middle|\; \boldsymbol{\lambda} \in \mathbb{R}_+^R, \mathbf{a}_r^{(1)} \in \Delta^{I_1 - 1}, \ldots, \mathbf{a}_r^{(D)} \in \Delta^{I_D - 1} \; \forall r \right\}$$

is attained.

The proof follows from the fact that all the norms on finite dimensional spaces are equivalent and so induce the same topology on $\mathbb{R}_+^{I_1 \times \cdots \times I_D}$ (further details can be found in [10]). This corollary implies that the PARAFAC degeneracy does not happen for non-negative approximations of non-negative tensors.

2.4 Computation Methods for NTF

To compute the non-negative factors matrices $\mathbf{A}^{(d)}$ (where $d = 1, \ldots, D$) of the decomposition of the tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, we need to solve an optimization problem in which we minimize a loss function. The global loss function that is usually minimized is of the form

$$f\left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}\right) = \left\| \boldsymbol{\mathcal{X}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|_F^2 + \alpha_1 \left\| \mathbf{A}^{(1)} \right\|_F^2 + \cdots + \alpha_D \left\| \mathbf{A}^{(D)} \right\|_F^2 ,$$

where $\alpha_1, \ldots, \alpha_D$ are non-negative regularization parameters.

There are many possible and used ways to compute this optimization problem with non-negative constraints. Here, we will see three of them and then we will use the most common approach with some modifications needed for our purposes. One possible approach is to use the vectorized form of f and employ the so called Non-linear Least Squares (NLS) algorithm based on Gauss-Newton methods. This method was first applied by Paatero [116] and by Bro and De Jong [117], Tomasi and Bro [118]. An alternative way to solve the problem is to optimize the cost function simultaneously with respect to all the involved variables using a Non-linear Conjugate Gradient method. This method was proposed by Acar et al. [119]. However, such a cost function is generally non convex and thus the convergence of the method is not a priori guaranteed even though the results shown are promising.

The most popular approach is the Alternating Least Squares (ALS), in which the function gradient is computed with respect to each individual factor matrix, by fixing all the others, to find a solution to the problem. The main advantage of the ALS methods are their high speed of convergence and their scalability for large-scale problems. We will see in particular the application of non-negative ALS (ANLS) update rules to solve NTF problems. The solution of these algorithms can be found by applying both the gradient function and an alternative computation proposed by Kim et al. [120], Kim and Park [121]. In particular, they illustrate the way of solving an ANLS where the non-negativity constrained least squares (NNLS) problems are solved at each iteration by the application of a Block Principal Pivoting (BPP) method. We will see the different methods for the three-way PARAFAC decomposition, in which we are interested.

2.4.1 Gauss-Newton Method

Fitting the PARAFAC model to the tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ in the least squares sense means to minimize the loss function expressed as

$$f(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \left\| \mathbf{X} - \mathbf{A} (\mathbf{C} \odot \mathbf{B})^T \right\|_F^2 , \tag{2.9}$$

where $\mathbf{X} \in \mathbb{R}^{I \times JK}$ is the matricized version of $\boldsymbol{\mathcal{X}}$ in the first mode. Minimizing (2.9) is a particularly difficult non-linear least squares problem, for which many algorithms have been proposed. These algorithms are based on the **Gauss-Newton method** [122] with some modifications to deal with the characteristics of the PARAFAC model. Solving the minimization problem corresponds to fit the PARAFAC model in the maximum likelihood sense, by providing that the residuals $\mathbf{r} = \text{vec}\mathbf{R}$ are normally distributed with zero mean and variance $\sigma^2\mathbf{I}$.

Given the vectorized form of the PARAFAC model

$$\mathbf{x} = \text{vec}\mathbf{X} = \text{vec}\left[\mathbf{A} (\mathbf{C} \odot \mathbf{B})^T\right] + \text{vec}\mathbf{R} ,$$

we can define the vector $\mathbf{p} = \text{vec}\left[\mathbf{A}^T|\mathbf{B}^T|\mathbf{C}^T\right]$ of length $N = (I + J + K)\,R$, that hold the model parameters and the minimization problem can be written as

$$\begin{aligned}\text{argmin}_{\mathbf{p}} \left\|\mathbf{r}\left(\mathbf{p}\right)\right\|_2^2 &= \text{argmin}\mathbf{r}\left(\mathbf{p}\right)^T \mathbf{r}\left(\mathbf{p}\right) \\ &= \text{argmin}\left(\mathbf{x} - \mathbf{y}\left(\mathbf{p}\right)\right)^T \left(\mathbf{x} - \mathbf{y}\left(\mathbf{p}\right)\right) \ ,\end{aligned} \tag{2.10}$$

where $\mathbf{y} = \text{vec}\left[\mathbf{A}\left(\mathbf{C} \odot \mathbf{B}\right)^T\right]$ of length $M = IJK$ and $\mathbf{r} = \left[r_1, \ldots, r_M\right]^T$.

In the Gauss-Newton method, the residuals in the neighbourhood of a point $\mathbf{p}^0$ are assumed to be approximated by a Taylor expansion truncated after the linear term in the following way:

$$\begin{aligned}r_m\left(\mathbf{p}\right) &= r_m\left(\mathbf{p}^0\right) + \sum_{n=1}^{N} \frac{\partial r_m}{\partial p_n}\left(p_n - p_n^0\right) + \mathcal{O}\left(\left\|\mathbf{p} - \mathbf{p}^0\right\|_2^2\right) \\ &\approx r_m\left(\mathbf{p}^0\right) - \sum_{n=1}^{N} \frac{\partial r_m}{\partial p_n}\left(p_n - p_n^0\right) = \tilde{r}_m\left(\mathbf{p}\right) \ \forall m \ .\end{aligned} \tag{2.11}$$

Let the matrix $\mathbf{J}\left(\mathbf{p}^0\right)$ be the Jacobian of size $M \times N$, whose elements are

$$j_{mn} = \frac{\partial r_m\left(\mathbf{p}^0\right)}{\partial p_n} = \frac{\partial y_m\left(\mathbf{p}^0\right)}{\partial p_n} \ ,$$

then, if the linear approximation in Eq. (2.10) holds, Eq. (2.11) can be expressed as a function of $\Delta\mathbf{p} = \mathbf{p} - \mathbf{p}^0$:

$$\tilde{f}\left(\Delta\mathbf{p}\right) = \tilde{\mathbf{r}}\left(\Delta\mathbf{p}\right)^T \tilde{\mathbf{r}}\left(\Delta\mathbf{p}\right) = \left\|\mathbf{r}\left(\mathbf{p}^0\right) + \mathbf{J}\left(\mathbf{p}^0\right)\Delta\mathbf{p}\right\|_2^2 \ .$$

At this stage it is possible to compute a new approximation of the parameter vector as $\mathbf{p}^{(s+1)} = \mathbf{p}^{(s)} + \Delta\mathbf{p}^{(s)}$, where $\Delta\mathbf{p}^{(s)}$ is computed as a solution to the linear problem

$$\min_{\Delta\mathbf{p}} \left\|\mathbf{r}\left(\mathbf{p}^{(s)}\right) + \mathbf{J}\left(\mathbf{p}^{(s)}\right)\Delta\mathbf{p}^{(s)}\right\|_2^2 \ , \tag{2.12}$$

by solving the system of normal equations

$$\left(\mathbf{J}^T\mathbf{J}\right)\Delta\mathbf{p}^{(s)} = -\mathbf{J}^T\mathbf{r} = \mathbf{g} \ ,$$

for $\Delta\mathbf{p}^{(s)}$. Here, $\mathbf{g}$ is the gradient of $\tilde{f}\left(\Delta\mathbf{p}\right)$. The method described above corresponds to the one proposed by Hayashi and Hayashi [123].

Damped Gauss-Newton

The algorithm exposed in the previous section is not enough to ensure a globally convergence while fitting the PARAFAC model. However, this issue can be overcome by the damped Gauss-Newton (dGN) method devised by Levenberg [124], Marquardt [125]. In dGN the update $\Delta\mathbf{p}^{(s)}$ is computed from the modified normal equation

$$\left(\mathbf{J}^T\mathbf{J} + \lambda^{(s)}\mathbf{I}_N\right)\Delta\mathbf{p}^{(s)} = -\mathbf{g}\,, \tag{2.13}$$

that is equal to solve the problem in Eq. (2.12) under the constraint that the minimum is assumed to be in a region (i.e. the **trust region**) of radius $\delta\left(\lambda^{(s)}\right)$:

$$\left\|\Delta\mathbf{p}^{(s)}\right\|_2^2 \leq \delta\left(\lambda^{(s)}\right)\,.$$

Thus, if $\lambda^{(s)}$ is large enough if compared to the singular values of $\mathbf{J}^T\mathbf{J}$, then the matrix $\left(\mathbf{J}^T\mathbf{J} + \lambda^{(s)}\mathbf{I}_N\right)$ is non-singular and the system (2.13) can be efficiently solved.

Positive Matrix Factorization 3

Several modifications to the dGN algorithm have been proposed by Paatero [116] in the so called Positive Matrix Factorization 3 (PMF3) algorithm for three-way PARAFAC decompositions. First of all the author introduced a regularization factor and a specific non-linear update. Secondly, the algorithm includes a line-search procedure that is used whenever the procedure diverges. Finally, PMF3 involves a weighted least squares loss function [126] and possible non-negativity constraints on the parameters.

The loss function including the regularization terms is written as

$$f_{PMF3}\left(\mathbf{p}\right) = \sum_{m=1}^{M} r_m^2\left(\mathbf{p}\right) + \gamma\sum_{n=1}^{N}\hat{p}_n^2 = \mathbf{r}\left(\mathbf{p}\right)^T\mathbf{r}\left(\mathbf{p}\right) + \gamma\hat{\mathbf{p}}^T\hat{\mathbf{p}}\,, \tag{2.14}$$

where $\hat{p}_n = \left(p_n - p_n^0\right)$. The loss function in Eq. (2.14) yields the following system of normal equations:

$$\left(\mathbf{J}^T\mathbf{J} + \left(\lambda^{(s)} + \gamma^{(s)}\right)\mathbf{I}_N\right)\Delta\mathbf{p}'^{(s)} = -\mathbf{J}^T\mathbf{r} + \gamma^{(s)}\hat{\mathbf{p}}^{(s)}\,,$$

which is solved for $\Delta\mathbf{p}'^{(s)}$. As $\mathbf{p}^0 = 0$, the regularization term is given by the product of the scalar γ and the norm of the vector $\mathbf{p}$, thus penalising high absolute values of the parameters. The aim of this procedure is to correct the loss function for the scaling indeterminacy.

The non-linear update $\Delta\mathbf{p}'$ can be computed by solving the following system:

$$\mathbf{J}^T\mathbf{J}\Delta\mathbf{p}'' = -\mathbf{J}\left(\mathbf{p}'\right)^T\mathbf{r}\left(\mathbf{p}'\right) + \gamma^{(s)}\hat{\mathbf{p}}^{(s)} ,$$

where $\mathbf{p}' = \mathbf{p}^{(s)} + 0.5\Delta\mathbf{p}'^{(s)}$. The final update is chosen as the one between $\Delta\mathbf{p}'^{(s)}$ and $\Delta\mathbf{p}''^{(s)}$ that provides the largest reduction in the loss function.

2.4.2 Non-linear Conjugate Gradient

As an alternative to Alternating Least Squares algorithms Acar et al. [119] proposed to solve CP decomposition by simultaneously approximate all the factor matrices through a gradient-based optimization method. Let us consider the minimization problem

$$\min f\left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}\right) = \frac{1}{2}\left\|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!]\right\|_F^2 , \tag{2.15}$$

where $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and the function f is a mapping from the cross-product of D two dimensional vector spaces to $\mathbb{R}$, that is

$$f : \mathbb{R}^{I_1 \times R} \otimes \mathbb{R}^{I_2 \times R} \otimes \cdots \otimes \mathbb{R}^{I_D \times R} \mapsto \mathbb{R} ,$$

so that the function would be of

$$P = R\sum_{d=1}^{D} I_d$$

variables. Moreover, f can be viewed as a scalar-valued function having as a parameter vector $\mathbf{x}$, that comprises the vectorized forms of the factor matrices:

$$\mathbf{x} = \begin{pmatrix} \mathbf{a}_1^{(1)} \\ \vdots \\ \mathbf{a}_R^{(1)} \\ \vdots \\ \mathbf{a}_1^{(D)} \\ \vdots \\ \mathbf{a}_R^{(D)} \end{pmatrix}$$

so that, $f : \mathbb{R}^P \mapsto \mathbb{R}$ and it is now possible to compute the gradient as a vector of size P, by calculating the partial derivatives of the function with respect to all the variables $\mathbf{a}_r^{(d)}$, with $r = 1, \ldots, R$ and $d = 1, \ldots, D$. We report in the following part the Theorem 3 in which the partial derivatives for the gradient are specified and the related proof that is taken from [119, 127].

Theorem 3. *The partial derivatives of the function f in Eq. (2.15) are given by*

$$\frac{\partial f}{\partial \mathbf{a}_r^{(d)}} = -\left(\boldsymbol{\mathcal{X}} \bigtimes_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')} \right) + \sum_{l=1}^{R} \gamma^{(d)_{rl}} \mathbf{a}_l^{(d)} \ ,$$

where $\times$ is the multiplication in multiple modes, $r = 1, \ldots, R$, $d = 1, \ldots, D$, and

$$\gamma^{(d)_{rl}} := \prod_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_l^{(d')} \ .$$

Proof. The function f can be rewritten as

$$f(\mathbf{x}) = \frac{1}{2} \underbrace{\|\boldsymbol{\mathcal{X}}\|^2}_{f_1} + \frac{1}{2} \underbrace{\left\| [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|^2}_{f_2} - \underbrace{\langle \boldsymbol{\mathcal{X}}, [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \rangle}_{f_3} \ .$$

The first summand only depends on $\boldsymbol{\mathcal{X}}$, thus as it does not involve the variables, its partial derivatives are equal to zero, i.e.

$$\frac{\partial f_1}{\partial \mathbf{a}_r^{(d)}} = \mathbf{0} \ ,$$

where $\mathbf{0}$ is a vector of length I_d. The second summand is

$$\begin{aligned}
f_2(\mathbf{x}) &= \left\| [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|^2 = \\
&= \langle \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)}, \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)} \rangle = \\
&= \sum_{k=1}^{R} \sum_{l=1}^{R} \prod_{d'=1}^{D} \mathbf{a}_k^{(d')T} \mathbf{a}_l^{(d')} = \\
&= \prod_{d'=1}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_r^{(d')} + 2 \sum_{\substack{l=1 \\ l \neq r}}^{R} \prod_{d'=1}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_l^{(d')} + \sum_{\substack{k=1 \\ k \neq r}}^{R} \sum_{\substack{l=1 \\ l \neq r}}^{R} \prod_{d'=1}^{D} \mathbf{a}_k^{(d')T} \mathbf{a}_l^{(d')} .
\end{aligned}$$

Therefore,

$$\begin{aligned}
\frac{\partial f_2}{\partial \mathbf{a}_r^{(d)}} &= 2 \left(\prod_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_r^{(d')} \right) \mathbf{a}_r^{(d)} + 2 \sum_{\substack{l=1 \\ l \neq r}}^{R} \left(\prod_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_l^{(d')} \right) \mathbf{a}_l^{(d)} = \\
&= 2 \sum_{l=1}^{R} \left(\prod_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')T} \mathbf{a}_l^{(d')} \right) \mathbf{a}_l^{(d)} .
\end{aligned}$$

The third summand is the inner product between the tensor $\boldsymbol{\mathcal{X}}$ and the Kruskal tensor $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!]$:

$$\begin{aligned}
f_3(\mathbf{x}) &= \langle \boldsymbol{\mathcal{X}}, [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \rangle = \\
&= \langle \boldsymbol{\mathcal{X}}, \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}^{(D)} \rangle = \\
&= \sum_{r=1}^{R} \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} x_{i_1 \ldots i_D} a_{i_1 r}^{(1)} \ldots a_{i_D r}^{(D)} = \\
&= \sum_{r=1}^{R} \left(\boldsymbol{\mathcal{X}} \bigtimes_{d'=1}^{D} \mathbf{a}_r^{(d')} \right) = \\
&= \sum_{r=1}^{R} \left(\boldsymbol{\mathcal{X}} \bigtimes_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')} \right) \mathbf{a}_r^{(d)} ,
\end{aligned}$$

and its partial derivative is given by

$$\frac{\partial f_3}{\partial \mathbf{a}_r^{(d)}} = \sum_{r=1}^{R} \left(\boldsymbol{\mathcal{X}} \bigtimes_{\substack{d'=1 \\ d' \neq d}}^{D} \mathbf{a}_r^{(d')} \right) .$$

The combination of the three partial derivatives computed leads to the desired result. □

It is worth noting that the scalar $\gamma_{rl}^{(d)}$ is the (r,l) entry of a matrix $\mathbf{\Gamma}^{(d)}$ defined as

$$\mathbf{\Gamma}^{(d)} = \left(\mathbf{A}^{(1)T}\mathbf{A}^{(1)}\right) * \cdots * \left(\mathbf{A}^{(d-1)T}\mathbf{A}^{(d-1)}\right) * \left(\mathbf{A}^{(d+1)T}\mathbf{A}^{(d+1)}\right) * \cdots * \left(\mathbf{A}^{(D)T}\mathbf{A}^{(D)}\right) .$$

The values can be then computed by taking the $R \times R$ matrix

$$\mathbf{\Upsilon}^{(d)} = \mathbf{A}^{(d)T}\mathbf{A}^{(d)} \ \forall d ,$$

and then

$$\Gamma^{(d)} = \Upsilon^{(1)} * \cdots * \Upsilon^{(d-1)} * \Upsilon^{(d+1)} * \cdots * \Upsilon^{(D)} .$$

Corollary 3. *The partial derivatives of f are given by*

$$\frac{\partial f}{\partial \mathbf{A}^{(d)}} = -\mathbf{X}_{(d)}\left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right) + \mathbf{A}^{(d)}\mathbf{\Gamma}^{(d)} ,$$

Regularization

The lack of a locally unique solution, due to the scaling indeterminacy can be overcome by modifying the loss function to include some regularization terms as done by Paatero [126]:

$$f_{reg}\left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}\right) = \frac{1}{2}\left\|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!]\right\|_F^2 + \frac{\lambda_d}{2}\sum_{d=1}^{D}\left\|\mathbf{A}^{(d)}\right\|_1^2 . \quad (2.16)$$

The regularization has the effect of encouraging the norms of the factor matrices to be similar.

Corollary 4. *The partial derivatives of the function f_{reg} are given by*

$$\frac{\partial f_{reg}}{\partial \mathbf{A}^{(d)}} = -\mathbf{X}_{(d)}\left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right) + \mathbf{A}^{(d)}\mathbf{\Gamma}^{(d)} + \lambda_d \mathbf{A}^{(d)} ,$$

with $d = 1, \ldots, D$.

2.4.3 Multiplicative Updating algorithm

The multiplicative updating (MU) algorithm was devised first for matrix factorizations and then was applied to solve the minimization problem for CP decompositions [128, 129]. We first have to write the approximation of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ in its matricized form for the d-th dimension, that is:

$$\mathbf{X}_{(d)} \approx \mathbf{A}^{(d)} \times \left(\mathbf{B}^{(d)}\right)^T ,$$

where

$$\mathbf{B}^{(d)} = \mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)} \in \mathbb{R}^{\left(\prod_{\substack{d'=1 \\ d' \neq d}}^{D} I_{d'}\right) \times R} .$$

Here, we report the derivation of the MU for the factor matrices $\mathbf{A}^{(d)}$ by directly using the one provided for non-negative matrix factorization algorithms [127]. Thus, the updating rule for the generic factor matrix $\mathbf{A}^{(d)}$ is

$$\mathbf{A}^{(d)} \leftarrow \mathbf{A}^{(d)} * \frac{\mathbf{X}_{(d)}\mathbf{B}^{(d)T}}{\mathbf{A}^{(d)}\mathbf{B}^{(d)}\mathbf{B}^{(d)T}} ,$$

where $\diagdown$ is the element-wise division.

2.4.4 Alternating least squares

The Alternating Least Squares (ALS) method for CP decompositions is one of the most popular and the primary workhorse algorithm, due to its speed and ease of implementation. The premise is to iteratively optimize one factor matrix at a time. Thus, at each iteration the algorithm tries to solve

$$\min_{\mathbf{A}^{(d)}} f\left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}\right) ,$$

with a particular fixed d, while holding all the other factor matrices constant. The equation can be written as

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2} \left\| \mathbf{X}_{(d)} - \mathbf{A}^{(d)} \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right)^T \right\|^2 .$$

With all but one factor fixed, the problem reduces to a linear least squares problem, having exact solution equal to

$$\mathbf{A}^{(d)} = \mathbf{X}_{(d)} \left(\left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)} \right)^T \right)^\dagger .$$

To avoid the computation of the pseudo-inverse we can simplify the equation by using the properties of the Khatri-Rao product, that lead to

$$\mathbf{A}^{(d)} = \mathbf{X}_{(d)} \left(\mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)} \right) \left(\mathbf{\Gamma}^{(d)} \right)^\dagger .$$

In this way, the method only requires to compute the pseudo-inverse of a matrix of size $R \times R$.

2.4.5 ANLS and Block Principal Pivoting

When the non-negativity constraints are included, the optimization problem to be solved is of the form

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2} \left\| \boldsymbol{\mathcal{X}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|_F^2 \text{ s.t. } \mathbf{A}^{(d)} \geq 0 \ \forall d = 1, \ldots, D . \tag{2.17}$$

The computation of such a problem is demanding not only because of the number of variables but in particular because non-negativity constraints are imposed on the factor matrices. The algorithm that we adopt as a base for the overall present work is the one proposed by Kim and Park [121], based on Alternating Non-negativity constrained Least Squares (ANLS) framework, where at each iteration the non-negativity constrained least squares sub-problems are solved. One advantage of the method proposed in [121] is the use of the so called Block Principal Pivoting (BPP) [130] to accelerate the traditional active-set method [131].

ANLS method

To solve the minimization problem in Eq. (2.17) by means of the ANLS method, we first need to rewrite the approximation model

$$\boldsymbol{\mathcal{X}} \approx [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!]$$

for any $d \in \{1, \ldots, D\}$ as

$$\mathbf{X}_{(d)} \approx \mathbf{A}^{(d)} \times \left(\mathbf{B}^{(d)}\right)^T ,$$

where

$$\mathbf{B}^{(d)} = \mathbf{A}^{(D)} \odot \cdots \odot \mathbf{A}^{(d+1)} \odot \mathbf{A}^{(d-1)} \odot \cdots \odot \mathbf{A}^{(1)} \in \mathbb{R}^{\left(\prod_{\substack{d'=1 \\ d' \neq d}}^{D} I_{d'}\right) \times R} .$$

The ANLS framework is a block-coordinate-descent method applied to Eq. (2.17). First, all the factor matrices but one ($\mathbf{A}^{(1)}$) are initialized with non-negative entries. Then, for $d = 1, \ldots, D$ the following subproblem is solved iteratively:

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2} \left\| \mathbf{B}^{(d)} \times \left(\mathbf{A}^{(d)}\right)^T - \left(\mathbf{X}_{(d)}\right)^T \right\|_F^2 \text{ s.t. } \mathbf{A}^{(d)} \geq 0 . \tag{2.18}$$

The convergence property of a block-coordinate-descent method states that if each sub-problem in the form above has a unique solution, then every limit point produced by the ANLS method is a stationary point. In particular, if matrices $\mathbf{B}^{(d)}$ are of full column rank, each sub-problem has a unique solution. An efficient algorithm to solve the problem is the BPP, described in the next section.

BPP algorithm

The BPP algorithm was first introduced by Júdice and Pires [130] for a **single right-hand side** case. Afterwards, Kim and Park [121] explained the way of accelerating the multiple right-hand side case. The adoption of the BPP to find the solution to the ANLS problem is driven by the fact that conventional active-set like methods have difficulties in finding a solution when the number of variables increases, as the number of iterations until the end strongly depends on the number of variables. By contrast, the BPP method manages to find a solution even when a high number of variables is involved, by exchanging multiple variables at a time.

Let us consider the non-negativity constrained least squares problem with single right-hand side vector:

$$\min_{\mathbf{x}\geq 0} \|\mathbf{V}\mathbf{x} - \mathbf{w}\|_2^2 \ , \tag{2.19}$$

where $\mathbf{V} \in \mathbb{R}^{P\times Q}$, $\mathbf{x} \in \mathbb{R}^{Q\times 1}$, and $\mathbf{w} \in \mathbb{R}^{P\times 1}$. The sub-problems in Eq. (2.18) are decomposed to independent instances of Eq. (2.19) with respect to each column vector of $\left(\mathbf{A}^{(d)}\right)^T$, such that $\forall i = 1, \ldots, I_d$ we have

$$\mathbf{V} = \mathbf{B}^{(d)} \ , \ \mathbf{x} = \left(\mathbf{A}^{(d)}\right)_i^T \text{ and } \mathbf{w} = \left(\mathbf{X}_{(d)}\right)_i^T \ .$$

Therefore, an algorithm for Eq. (2.19) is a basic building block of an algorithm for Eq. (2.18). The Karush-Kuhn-Tucker (KKT) optimality conditions for Eq. (2.19) are

$$\mathbf{y} = \mathbf{V}^T\mathbf{V}\mathbf{x} - \mathbf{V}^T\mathbf{w} \ , \tag{2.20a}$$

$$\mathbf{y} \geq 0 \ , \ \mathbf{x} \geq 0 \ , \tag{2.20b}$$

$$x_q y_q = 0 \ , \text{ with } q = 1, \ldots, Q \ . \tag{2.20c}$$

If $\mathbf{V}$ has full column rank, then a solution $\mathbf{x}$ that satisfies the conditions in Eqs. (2.20) is also the optimal solution for Eq. (2.19). By dividing the set of indices $\mathcal{Q} = \{1, \ldots, Q\}$ into two disjoint subsets $\mathcal{F}$ and $\mathcal{G}$, i.e.

$$\mathcal{F} \cup \mathcal{G} = \mathcal{Q} \text{ and } \mathcal{F} \cap \mathcal{G} = \varnothing \ ,$$

we can define the subgroups of variables $\mathbf{x}_\mathcal{F}, \mathbf{x}_\mathcal{G}, \mathbf{y}_\mathcal{F}$, and $\mathbf{y}_\mathcal{G}$, with the corresponding indices, and the sub-matrices of $\mathbf{V}$, $\mathbf{V}_\mathcal{F}$ and $\mathbf{V}_\mathcal{G}$, with the corresponding column indices. By initially assigning zeros to $\mathbf{x}_\mathcal{G}$ and $\mathbf{y}_\mathcal{F}$, then $\mathbf{x} = (\mathbf{x}_\mathcal{F}, \mathbf{x}_\mathcal{G})$ and $\mathbf{y} = (\mathbf{y}_\mathcal{F}, \mathbf{y}_\mathcal{G})$ always satisfy Eq. (2.20c) for any $\mathbf{x}_F$ and $\mathbf{y}_G$. At this stage, $\mathbf{x}_F$ and $\mathbf{y}_G$ can be computed by using Eq. (2.20a) and the values that satisfy Eq. (2.20b) are the one to choose. The computation is done as follows:

$$\mathbf{V}_\mathcal{F}^T\mathbf{V}_\mathcal{F}\mathbf{x}_\mathcal{F} = \mathbf{V}_\mathcal{F}^T\mathbf{w} \ , \tag{2.21a}$$

$$\mathbf{y}_\mathcal{G} = \mathbf{V}_\mathcal{G}^T\left(\mathbf{V}_\mathcal{F}\mathbf{x}_\mathcal{F} - \mathbf{w}\right) \ . \tag{2.21b}$$

We can then solve Eq. (2.21a) for $\mathbf{x}_{\mathcal{F}}$ and use it to find $\mathbf{y}_{\mathcal{G}}$ through Eq. (2.21b). The resulting pair $(\mathbf{x}_{\mathcal{F}}, \mathbf{y}_{\mathcal{G}})$ is called the **complementary basic solution**, that is said to be

$$\begin{aligned} &\textbf{feasible} \text{ if } \mathbf{x}_F \geq 0 \text{ and } \mathbf{y}_G \geq 0 \,, \\ &\textbf{infeasible} \text{ otherwise} \,. \end{aligned}$$

The optimal solution of Eq. (2.19) in the first case is $\mathbf{x} = (\mathbf{x}_F, 0)$, otherwise we need to update the sets $\mathcal{F}$ and $\mathcal{G}$ by exchanging the variables for which Eq. (2.20b) is not satisfied. That is defining a new index set

$$\mathcal{H} = \{q \in \mathcal{F} : \mathbf{x}_q < 0\} \cup \{q \in \mathcal{G} : \mathbf{y}_q < 0\}$$

and choosing a non-empty subset $\hat{\mathcal{H}} \subset \mathcal{H}$, so that $\mathcal{F}$ and $\mathcal{G}$ are updated in the following way:

$$\begin{aligned} \mathcal{F} &= \left(\mathcal{F} - \hat{\mathcal{H}}\right) \cup \left(\hat{\mathcal{H}} \cap \mathcal{G}\right) \,, \\ \mathcal{G} &= \left(\mathcal{G} - \hat{\mathcal{H}}\right) \cup \left(\hat{\mathcal{H}} \cap \mathcal{F}\right) \,. \end{aligned}$$

The method is then iterated until the number of infeasible variables goes to zero: $\left|\hat{\mathcal{H}}\right| = 0$. If the number of variables $\left|\hat{\mathcal{H}}\right|$ exchanged at each iteration is $\left|\hat{\mathcal{H}}\right| > 1$, then the algorithm is called a block principal pivoting, otherwise if $\left|\hat{\mathcal{H}}\right| = 1$ the algorithm is said to be a single principal pivoting algorithm. The **active set** method can be seen as an instance of the single principal pivoting algorithm.

To speed up the search procedure, the **full exchange rule** is usually adopted, i.e. $\left|\hat{\mathcal{H}}\right| = |\mathcal{H}|$. This rule allows to exchange all the infeasible variables at once, thus accelerating the computation by removing the number of iterations.

Let us now consider the **multiple right-hand side** case:

$$\min_{\mathbf{X} \geq 0} \|\mathbf{V}\mathbf{X} - \mathbf{W}\|_F^2 \,, \tag{2.22}$$

where $\mathbf{V} \in \mathbb{R}^{P \times Q}$, $\mathbf{X} \in \mathbb{R}^{Q \times L}$ and $\mathbf{W} \in \mathbb{R}^{P \times L}$. Eq. (2.22) can be solved by separately solving the non-negativity constrained least squares problems with each right-hand side vector. However, there are efficient ways to accelerate the

multiple right-hand side case by using the improvements provided by Kim and Park [132].

Observation 1. *The matrix* $\mathbf{V}$*, corresponding to* $\mathbf{B}^{(d)}$ *in Eq.* (2.18) *has typically one bigger dimension with respect to the other that is small. Thus, instead of computing the matrices* $\mathbf{V}_{\mathcal{F}}^T\mathbf{V}_{\mathcal{F}}, \mathbf{V}_{\mathcal{F}}^T\mathbf{w}, \mathbf{V}_{\mathcal{G}}^T\mathbf{V}_{\mathcal{F}}$*, and* $\mathbf{V}_{\mathcal{G}}^T\mathbf{w}$*, that is computationally expensive, one can compute* $\mathbf{V}^T\mathbf{V}$ *and* $\mathbf{V}^T\mathbf{W}$ *at the beginning and reuse them in the successive iterations.*

Observation 2. *By computing the BPP for multiple right-hand side vectors, at each iteration, we have to compute the variables* $\mathbf{x}_{\mathcal{F}_l}$ *and* $\mathbf{y}_{\mathcal{G}_l}$ *corresponding to the index sets* $\mathcal{F}_l$ *and* $\mathcal{G}_l$ *for the columns* $l \in \{1, \ldots, L\}$*. thus, by finding set of columns that share the same index sets, it is possible to reorder the columns in the same group and solving Eq.* (2.21a) *at once.*

Regularization

In this section we extend the ANLS framework to the case of Non-negative CP decompositions with regularization. By starting from Eq. (2.16) and adding the non-negativity constraints we can write the related ANLS sub-problems as

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2} \left\| \begin{pmatrix} \mathbf{B}^{(d)} \\ \sqrt{\lambda_d}\mathbf{I} \end{pmatrix} \times \left(\mathbf{A}^{(d)}\right)^T - \left(\mathbf{X}^{(d)}\right)^T \right\|_F^2 \text{ s.t. } \mathbf{A}^{(d)} \geq 0 \,,$$

where $\mathbf{I}$ is an identity matrix of size $R \times R$.

Observation 3. *The matrix* $\begin{pmatrix} \mathbf{B}^{(d)} \\ \sqrt{\lambda_d}\mathbf{I} \end{pmatrix}$ *is always of full column rank even if* $\mathbf{B}^{(d)}$ *may be not of full column rank, thus the regularization term can be used to ensure that each sub-problem is of full column rank. In this way each sub-problem satisfies the requirement of the convergence property of the BPP method.*

To include also sparsity on the factors $\mathbf{A}^{(d)}$ the l_1-norm can be used, so that the sub-problems for each factor are modified as

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2} \left\| \begin{pmatrix} \mathbf{B}^{(d)} \\ \sqrt{\lambda_d}\mathbf{1} \end{pmatrix} \times \left(\mathbf{A}^{(d)}\right)^T - \left(\mathbf{X}^{(d)}\right)^T \right\|_F^2 \text{ s.t. } \mathbf{A}^{(d)} \geq 0 \,,$$

where $\mathbf{1}$ is a row vector of ones of size $1 \times R$.

2.4.6 Parametrization of the factor matrices

An alternative way to impose non-negativity constraints to the factor matrices in the CP decomposition is by means of a parametrization which does not modify the cost function

$$f(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2}\left\|\boldsymbol{\mathcal{X}} - [\![\boldsymbol{\lambda};\mathbf{A},\mathbf{B},\mathbf{C}]\!]\right\|_F^2 .$$

The cost function can be rewritten in the matricized form, used to solve the problem, as

$$\begin{aligned} f(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{1}{2}\left\|\mathbf{X}_{(1)} - \mathbf{A}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{B}\right)^T\right\|_F^2 \\ &= \frac{1}{2}\left\|\mathbf{X}_{(2)} - \mathbf{B}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{A}\right)^T\right\|_F^2 \\ &= \frac{1}{2}\left\|\mathbf{X}_{(3)} - \mathbf{C}\boldsymbol{\Lambda}\left(\mathbf{B}\odot\mathbf{A}\right)^T\right\|_F^2 , \end{aligned}$$

where $\boldsymbol{\Lambda} = \operatorname{diag}(\boldsymbol{\lambda})$. Then, to consider a factor matrix $\mathbf{A}$ to be non-negative we can assume that all its entries are equal to a_{ij}^2. Therefore, we can rewrite the optimization problem as a function of the Hadamard product of the factor matrices $\mathbf{A}*\mathbf{A}$,$\mathbf{B}*\mathbf{B}$, and $\mathbf{C}*\mathbf{C}$, as:

$$\begin{aligned} h(\mathbf{A},\mathbf{B},\mathbf{C}) &= f(\mathbf{A}*\mathbf{A},\mathbf{B}*\mathbf{B},\mathbf{C}*\mathbf{C}) \\ &= \frac{1}{2}\left\|\mathbf{X}_{(1)} - (\mathbf{A}*\mathbf{A})\,\boldsymbol{\Lambda}\left((\mathbf{C}*\mathbf{C})\odot(\mathbf{B}*\mathbf{B})\right)^T\right\|_F^2 = \frac{1}{2}\left\|\boldsymbol{\delta}_{(1)}\right\|_F^2 \\ &= \frac{1}{2}\left\|\mathbf{X}_{(2)} - (\mathbf{B}*\mathbf{B})\,\boldsymbol{\Lambda}\left((\mathbf{C}*\mathbf{C})\odot(\mathbf{A}*\mathbf{A})\right)^T\right\|_F^2 = \frac{1}{2}\left\|\boldsymbol{\delta}_{(2)}\right\|_F^2 \\ &= \frac{1}{2}\left\|\mathbf{X}_{(3)} - (\mathbf{C}*\mathbf{C})\,\boldsymbol{\Lambda}\left((\mathbf{B}*\mathbf{B})\odot(\mathbf{A}*\mathbf{A})\right)^T\right\|_F^2 = \frac{1}{2}\left\|\boldsymbol{\delta}_{(3)}\right\|_F^2 . \end{aligned}$$

This approach was first used for non-negative matrix factorization problems [133] and then adapted to NTF by Royer et al. [134].

To solve the minimization problem we need to derive the differential $dh(\mathbf{A},\mathbf{B},\mathbf{C})$ for the new cost function $h(\mathbf{A},\mathbf{B},\mathbf{C})$, which is given by:

$$dh(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2}\left\langle\frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}}, d\mathbf{A}\right\rangle + \frac{1}{2}\left\langle\frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}}, d\mathbf{B}\right\rangle + \frac{1}{2}\left\langle\frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}}, d\mathbf{C}\right\rangle .$$

We know that $\langle \mathbf{A}, \mathbf{A} \rangle = \|\mathbf{A}\|_F^2 = \text{trace}\{\mathbf{A}^T\mathbf{A}\}$, so we can rewrite the cost function $h(\mathbf{A}, \mathbf{B}, \mathbf{C})$ as

$$\begin{aligned} h(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \frac{1}{2}\left\|\boldsymbol{\delta}_{(1)}\right\|_F^2 = \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(1)}^T\boldsymbol{\delta}_{(1)}\right\} \\ &= \frac{1}{2}\left\|\boldsymbol{\delta}_{(2)}\right\|_F^2 = \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(2)}^T\boldsymbol{\delta}_{(2)}\right\} \\ &= \frac{1}{2}\left\|\boldsymbol{\delta}_{(3)}\right\|_F^2 = \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(3)}^T\boldsymbol{\delta}_{(3)}\right\} . \end{aligned}$$

By using this equalities we can write the differential dh in the following way:

$$dh(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \frac{1}{2}d\text{trace}\left\{\boldsymbol{\delta}_{(1)}^T\boldsymbol{\delta}_{(1)}\right\} + \frac{1}{2}d\text{trace}\left\{\boldsymbol{\delta}_{(2)}^T\boldsymbol{\delta}_{(2)}\right\} + \frac{1}{2}d\text{trace}\left\{\boldsymbol{\delta}_{(3)}^T\boldsymbol{\delta}_{(3)}\right\} ,$$

but we know that $d(\text{trace}\{\mathbf{A}\}) = \text{trace}\{d\mathbf{A}\}$ and $\text{trace}\{\mathbf{A}\} = \text{trace}\{\mathbf{A}^T\}$, so

$$\begin{aligned} dh(\mathbf{A}, \mathbf{B}, \mathbf{C}) = &\frac{1}{2}\text{trace}\left\{d\left(\boldsymbol{\delta}_{(1)}^T\right)\boldsymbol{\delta}_{(1)}\right\} + \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(1)}^T d\boldsymbol{\delta}_{(1)}\right\} + \\ &+ \frac{1}{2}\text{trace}\left\{d\left(\boldsymbol{\delta}_{(2)}^T\right)\boldsymbol{\delta}_{(2)}\right\} + \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(2)}^T d\boldsymbol{\delta}_{(2)}\right\} + \\ &+ \frac{1}{2}\text{trace}\left\{d\left(\boldsymbol{\delta}_{(3)}^T\right)\boldsymbol{\delta}_{(3)}\right\} + \frac{1}{2}\text{trace}\left\{\boldsymbol{\delta}_{(3)}^T d\boldsymbol{\delta}_{(3)}\right\} = \\ &= \text{trace}\left\{\boldsymbol{\delta}_{(1)}^T d\boldsymbol{\delta}_{(1)}\right\} + \text{trace}\left\{\boldsymbol{\delta}_{(2)}^T d\boldsymbol{\delta}_{(2)}\right\} + \text{trace}\left\{\boldsymbol{\delta}_{(3)}^T d\boldsymbol{\delta}_{(3)}\right\} . \end{aligned}$$

As an example, we now go further in the calculation for the first term in the sum:

$$\begin{aligned} \text{trace}\left\{\boldsymbol{\delta}_{(1)}^T d\boldsymbol{\delta}_{(1)}\right\} &= 2\text{trace}\left\{-\boldsymbol{\delta}_{(1)}^T(\mathbf{A}d\mathbf{A})\boldsymbol{\Lambda}\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]\right\} = \\ &= 2\text{trace}\left\{\boldsymbol{\Lambda}\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]^T\left(-\boldsymbol{\delta}_{(1)}\right)^T(\mathbf{A}d\mathbf{A})\right\} , \end{aligned}$$

where for the last equality we used the fact that $\text{trace}\{\mathbf{AB}\} = \text{trace}\{\mathbf{BA}\}$. Finally, by applying the property of the trace for which $\text{trace}\{\mathbf{A}^T(\mathbf{B} * \mathbf{C})\} = \text{trace}\{(\mathbf{A}^T * \mathbf{B}^T)\mathbf{C}\}$, we have

$$\begin{aligned} \text{trace}\left\{\boldsymbol{\delta}_{(1)}^T d\boldsymbol{\delta}_{(1)}\right\} &= 2\text{trace}\left\{\boldsymbol{\Lambda}\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]^T\left(-\boldsymbol{\delta}_{(1)}\right)^T(\mathbf{A}d\mathbf{A})\right\} \\ &= 2\text{trace}\left\{\left[\boldsymbol{\Lambda}\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]^T\left(-\boldsymbol{\delta}_{(1)}\right)^T * \mathbf{A}^T\right]d\mathbf{A}\right\} \\ &= 2\text{trace}\left\{\left[\mathbf{A} * \left(-\boldsymbol{\delta}_{(1)}\right)\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]\boldsymbol{\Lambda}^T\right]^T d\mathbf{A}\right\} \\ &= 2\langle \mathbf{A} * \left(-\boldsymbol{\delta}_{(1)}\right)\left[(\mathbf{C} * \mathbf{C}) \odot (\mathbf{B} * \mathbf{B})\right]\boldsymbol{\Lambda}^T, d\mathbf{A}\rangle . \end{aligned}$$

By proceeding analogously for the other modes, the calculation of $dh(\mathbf{A},\mathbf{B},\mathbf{C})$ leads to

$$\begin{aligned} dh(\mathbf{A},\mathbf{B},\mathbf{C}) = {} & 2\langle \mathbf{A} * \left(-\boldsymbol{\delta}_{(1)}\right)[(\mathbf{C}*\mathbf{C}) \odot (\mathbf{B}*\mathbf{B})]\,\boldsymbol{\Lambda}^T, d\mathbf{A}\rangle \\ & + 2\langle \mathbf{B} * \left(-\boldsymbol{\delta}_{(2)}\right)[(\mathbf{C}*\mathbf{C}) \odot (\mathbf{A}*\mathbf{A})]\,\boldsymbol{\Lambda}^T, d\mathbf{B}\rangle \\ & + 2\langle \mathbf{C} * \left(-\boldsymbol{\delta}_{(3)}\right)[(\mathbf{B}*\mathbf{B}) \odot (\mathbf{A}*\mathbf{A})]\,\boldsymbol{\Lambda}^T, d\mathbf{C}\rangle\,. \end{aligned}$$

By using this result, we can then adapt different methods. We report here the resulting gradients which can be used to apply the non-negativity constraints to the NCG method:

$$\begin{aligned} \nabla_{\mathbf{A}} h(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{A}} = 2\mathbf{A} * \left(\left(-\boldsymbol{\delta}_{(1)}\right)[(\mathbf{C}*\mathbf{C}) \odot (\mathbf{B}*\mathbf{B})]\right) \\ \nabla_{\mathbf{B}} h(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{B}} = 2\mathbf{B} * \left(\left(-\boldsymbol{\delta}_{(2)}\right)[(\mathbf{C}*\mathbf{C}) \odot (\mathbf{A}*\mathbf{A})]\right) \\ \nabla_{\mathbf{C}} h(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial h(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{C}} = 2\mathbf{C} * \left(\left(-\boldsymbol{\delta}_{(3)}\right)[(\mathbf{B}*\mathbf{B}) \odot (\mathbf{A}*\mathbf{A})]\right)\,. \end{aligned}$$

In Chapter 5 we will see how to use this result to compute the gradients for the NCG in the special case the factorization of multiple tensors at the same time.

2.5 Rank Computation

Tensors used in applications are usually affected by noise, that renders the tensor rank hard to detect. A fundamental topic in finding the best CP approximation for noisy tensors is then to find the number of its components, i.e. the rank. There are in the literature several methods to compute the typical rank of a CP decomposition and we will review some of the most known. First of all we will see the Difference in Fit (DIFFIT) [135], that examines the fitting error of each candidate model and selects the number of components that corresponds to the model characterized by the maximal curvature of the error versus the number of components curve. Then we will introduce the Automatic Relevance Determination (ARD) [136] based on Bayesian frameworks, and the generalized multi-linear Minimum Description Length (N-D MDL) [137]. Finally we will introduce the method that we use to determine the number of components in our work, that is the Core Consistency Diagnostic (CORCONDIA) [138], which

looks at the deviation of the estimate core tensor from the ideal super-identity core for each model candidate. We will introduce also an approach to make the core consistency computation faster, devised by Papalexakis and Faloutsos [139].

2.5.1 Difference in Fit

The DIFFerence in FIT method (DIFFIT) was introduced by Timmerman and Kiers [135] for three-way CP decompositions as an alternative to the methods specifically devised for two-way principal component analysis (PCA). The method is based on the Tucker decomposition of a three-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$:

$$\boldsymbol{\mathcal{X}} = [\![\boldsymbol{\mathcal{G}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] + \boldsymbol{\mathcal{E}} \, ,$$

where $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{P \times Q \times R}$ is the so called core tensor, $\mathbf{A} \in \mathbb{R}^{I \times P}$, $\mathbf{B} \in \mathbb{R}^{J \times Q}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$, and $\boldsymbol{\mathcal{E}}$ is the residual tensor.

In the DIFFIT method the set of possible decompositions to be considered is reduced to the one having

$$PQ \geq R, \; PR \geq Q, \text{ and } QR \geq P \, , \tag{2.23}$$

as well as having

$$P \geq \max(I, JK) \, , \; Q \geq \max(J, IK) \, , \text{ and/or } R \geq \max(K, IJ) \; .$$

Thus the possible values P, Q, R are a priori restricted to the maximum values $P_{max}, Q_{max}, R_{max}$. The DIFFIT method can be divided in 6 steps:

1. determine the decomposition for all the combinations of (P, Q, R) components, for which the conditions in Eq. (2.23) holds;
2. compare the solutions between the same number of components ($s = P + Q + R$), and determine the best fit among the decompositions by computing the fitting/reconstruction error, given by

$$e(s) = \| \boldsymbol{\mathcal{X}} - [\![\boldsymbol{\mathcal{G}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|_F^2 \; ;$$

3. compute $dif_s = e(s) - e(s-1)$ that is the difference in fit of the best fit with s components and the one with $s-1$ components. Compute the subset of solutions for which $dif_s \geq dif_{s+n}$ holds (here, $n = 1, \ldots, S-s$ $S = P_{max} + Q_{max} + R_{max}$) and save the related number of components in the vector $t \in \mathbb{N}^{M \times 1}$;

4. calculate
$$DIFFIT_{t_m} = \frac{dif_{t_m}}{dif_{t_{m+1}}} ,$$
where $m = 1, \ldots, M$ are the indices of the subset of values selected in the previous step;

5. detect the value that corresponds to the maximal $DIFFIT_{t_m}$ among the values of t_m for which
$$dif_{t_m} \geq \frac{\|\boldsymbol{\mathcal{X}}\|_F^2}{(s_{max} - 3)} ,$$
with $s_{max} = \max(I, JK) + \max(J, IK) + \max(K, IJ)$;

6. choose the number of components associated to the best fit among all models where $P + Q + R$ corresponds to the maximal $DIFFIT_{t_m}$.

2.5.2 Automatic Relevance Determination

The Automatic Relevance Determination (ARD) method, introduced by Mørup and Hansen [136], is a Bayesian approach that was firstly derived for Tucker decompositions. As the CP decomposition is a particular case of the Tucker decomposition, the ARD approach can be applied to this model, and in this case it is a simplified version of the original method. The entries of the factor matrices $\mathbf{A}^{(d)}$ are assigned to a Gaussian prior:

$$Pr\left(\mathbf{A}^{(d)}\right) = \prod_r \left(\frac{\alpha_r^{(d)}}{2\pi}\right)^{I_d/2} \exp\left\{-\frac{\alpha_r^{(d)}}{2\pi} \left\|\mathbf{a}_r^{(d)}\right\|^2\right\} .$$

Let the entries of the factor matrices be independent across the different modes, then the **negative log posterior** is

$$\begin{aligned} -\log P = & \frac{1}{2\sigma^2} \left\| \boldsymbol{\mathcal{X}} - [\![\boldsymbol{\mathcal{L}}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(D)}]\!] \right\|_F^2 \\ & + \frac{1}{2} \sum_{d=1}^{D} \sum_{r=1}^{R} \alpha_r^{(d)} \left\| \mathbf{a}_r^{(d)} \right\|^2 \\ & - \sum_{d=1}^{D} \sum_{r=1}^{R} I_d \log \alpha_r^{(d)} \\ & + \frac{1}{2} I_1 I_2 \ldots I_D \log \sigma^2 \, , \end{aligned}$$

where $\boldsymbol{\mathcal{L}}$ is the super-identity tensor. By minimizing the above negative log posterior, it is possible to find not only the rank R of the best CP approximation but also the parameters $\{\mathbf{A}^{(d)}\}$, $\{\alpha_r^{(d)}\}$, and σ^2. This problem can be reconducted to a l_2 regularized CP decomposition of the form

$$\min_{\mathbf{A}^{(d)}} \frac{1}{2\sigma^2} \left\| \mathbf{X} - \mathbf{A}^{(d)} \mathbf{S} \right\|_F^2 + \frac{1}{2} \sum_{r=1}^{R} \alpha_r \left\| \mathbf{a}^{(d)} \right\|^2$$

with respect to the d-mode. This minimization problem has a closed-form solution, given by

$$\hat{\mathbf{A}}^{(d)} = \mathbf{X} \mathbf{S}^T \left(\mathbf{S} \mathbf{S}^T + \sigma^2 \mathrm{diag}\{\alpha\} \right)^{-1} \, ,$$

where $\alpha = [\alpha_1, \ldots, \alpha_R]$ and $\mathbf{S} = \boldsymbol{\mathcal{L}} \times_1 \mathbf{A}^{(1)} \times_2 \cdots \times_D \mathbf{A}^{(D)} \times_d \left(\mathbf{A}^{(d)} \right)^{\dagger}$. When all the factor matrices $\mathbf{A}^{(d)}$ are estimated, it is possible to find the parameters $\{\alpha_r\}$ and σ^2 through the maximum a posteriori criterion

$$\begin{aligned} \hat{\alpha}_r^{(d)} &= \frac{I_d}{\left\| \mathbf{a}_r^{(d)} \right\|^2} \; \forall d, r \, , \\ \hat{\sigma}^2 &= \frac{1}{I_1 I_2 \ldots I_D} \left\| \boldsymbol{\mathcal{X}} - [\![\boldsymbol{\mathcal{L}}; \hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(D)}]\!] \right\|_F^2 \, . \end{aligned}$$

The ARD method starts by guessing an upper bound of the number of components R_{max} and randomly initializing the factor matrices. At the end of the procedure the algorithm may not lead to factor matrices with equal number of columns $R = R_1 = \cdots = R_D$. In this case it would be necessary to reduce them in some modes so that the final rank R would be equal to $\min\{R_1, \ldots, R_D\}$.

Multi-linear Minimum Description Length

The multi-linear Minimum Description Length (N-D MDL) algorithm was introduced by da Costa et al. [140] to give an alternative to the classical models for two-dimensional rank detectors as the Minimum Description Length [141], the Akaike Information Criterion [142], or the Random Matrix Theory [143]. Recently, some improvements to the model were proposed by Liu et al. [137], who contributed to the matricization-based N-D rank detection method.

Let $\mathcal{D} = \{1, \ldots, D\}$ be the set of dimension indices for a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$. The total number of divisions of $\mathcal{D}$ into two disjoint subset of indices of the form

$$\mathcal{D}_1 = \{d_1, \ldots, d_n\} \text{ and } \mathcal{D}_2 = \mathcal{D} \setminus \mathcal{D}_1 \ ,$$

corresponds to the number of pairs of mutually-transposed matricizations of $\boldsymbol{\mathcal{X}}$ and is equal to $2^{D-1} - 1$. This number of matricized matrices are sorted in the method in descending order of their number of rows as

$$\mathbf{X}_{(1)}, \mathbf{X}_{(2)}, \ldots, \mathbf{X}_{(2^{D-1}-1)} \ .$$

Given the number of rows P_k and columns Q_k of the matrix $\mathbf{X}_{(k)}$ we have that $P_1 \geq P_2 \geq \cdots \geq P_{2^{D-1}-1}$ and $P_k \leq Q_k \ \forall k$. Let the sets of eigenvalues of the matrices $\mathbf{X}_{(k)}\mathbf{X}_{(k)}^H / Q_k$, where H is the Hermitian transposition, be

$$\begin{aligned}
&l_{1,1}, l_{2,1}, \ldots, l_{P_1,1}, \\
&l_{1,2}, l_{2,2}, \ldots, l_{P_2,2}, \\
&\vdots \\
&l_{1,2^{D-1}-1}, l_{2,2^{D-1}-1}, \ldots, l_{P_{2^{D-1}-1},2^{D-1}-1} \ .
\end{aligned}$$

The MDL method determines the number of components of the decomposition by minimizing the **penalized log-likelihood function**:

$$MDL(r) = \log\left(p_r\left(\mathbf{X}, \hat{\theta}_r\right)\right) + \frac{1}{2}\log(Q) \cdot \nu(r; P, Q) \ ,$$

where $p_r\left(\mathbf{X}, \hat{\theta}_r\right)$ is the likelihood function with maximum likelihood estimate $\hat{\theta}_r$ of the parameter θ_r for the r-model. The second term of the function is a

penalty term with ν free parameters in θ_r. In particular

$$\mu(r; P, Q) = r(2P - r)$$

and the likelihood function is

$$\log\left(p_r\left(\mathbf{X}, \hat{\theta}_r\right)\right) = -Q(P-r)\log\left(\frac{\left(\prod_{i=r+1}^{P} l_i\right)^{\frac{1}{P-r}}}{\frac{1}{P-r}\sum_{i=r+1}^{P} l_i}\right).$$

The MDL tends to underestimate the number of final components to be used. To overcome this issue, it is possible to employ the set of eigenvalues associated to the matrix $\mathbf{X}_{(2^{D-1}-1)}$, having the smallest number of rows and the largest number of columns. Finally, to improve more the performance, in [137] is proposed to build the global eigenvalues by summing all the $2^{D-1}-1$ sets of eigenvalues:

$$l_i^{(G)} = l_{i,1} + l_{i,2} + \cdots + l_{i,2^{D-1}-1}.$$

Therefore, the MDL criterion to detect the number of components becomes:

$$\hat{R}^{2^{D-1}-1} = \text{argmin}_{r\in\{0,1,\ldots,P_{2^{D-1}-1}-1\}}\text{MDL}_{N-D}(r)\ ,$$

with

$$\begin{aligned}\text{MDL}_{N-D}(r) = {} & Q_{2^{D-1}-1}\left(P_{2^{D-1}-1} - r\right) \\ & \times \log \frac{\frac{1}{P_{2^{D-1}-1}-r}\sum_{i=r+1}^{P_{2^{D-1}-1}} l_i^{(G)}}{\left(\prod_{i=r+1}^{P_{2^{D-1}-1}} l_i^{(G)}\right)^{\frac{1}{P_{2^{D-1}-1}-r}}} \\ & \frac{1}{2} r\left(2P_{2^{D-1}-1} - r\right)\log Q_{2^{D-1}-1}\ ,\end{aligned}$$

with $\mathbf{X}_{(2^{D-1}-1)}$ having $P_{2^{D-1}-1}$ rows and $Q_{2^{D-1}-1}$ columns.

2.5.3 Core Consistency Diagnostic

The Core Consistency Diagnostic (CORCONDIA) was devised by Bro and Kiers [138], and gives an evaluation of the closeness of the computed decomposition to the ideal one by comparing the core tensor of the decomposition to the ideal

one. The method is designed for three-way models but can be extended to higher-order models.

Let us consider the PARAFAC approximation of a three-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$

$$\boldsymbol{\mathcal{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] + \boldsymbol{\mathcal{E}} \,,$$

that can be written in the matricized form

$$\mathbf{X} = \mathbf{A} \left(\mathbf{C} \odot \mathbf{B}\right)^T + \mathbf{E} \,,$$

where $\mathbf{X} \in \mathbb{R}^{I \times JK}$, $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$, and $\mathbf{E} \in \mathbb{R}^{I \times JK}$ is the matrix of residuals. To compute the core consistency value we can first rewrite the problem as a Tucker3 model, by adding the super-diagonal core tensor $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{R \times R \times R}$ in its matricized form $\mathbf{L} \in \mathbb{R}^{R \times RR}$:

$$\mathbf{X} = \mathbf{A}\mathbf{L} \left(\mathbf{C} \odot \mathbf{B}\right)^T + \mathbf{E} \,.$$

Let us assume now that the PARAFAC model has been fitted with factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, we can fit the Tucker3 model to the data with the recovered factor matrices by minimizing

$$\sigma\left(\mathbf{G}\right) = \left\| \mathbf{X} - \mathbf{A}\mathbf{G} \left(\mathbf{C} \odot \mathbf{B}\right)^T \right\|_F^2 \,.$$

The optimal $\mathbf{G}$ can be computed by writing the equation in its vectorized form and applying a least squares algorithm:

$$\sigma\left(\mathbf{G}\right) = \left\| \mathrm{vec}\mathbf{X} - \left(\mathbf{C} \otimes \mathbf{B} \otimes \mathbf{A}\right) \mathrm{vec}\mathbf{G} \right\|_F^2 \,,$$

and it is then determines by

$$\mathrm{vec}\mathbf{G} = \left(\mathbf{C} \otimes \mathbf{B} \otimes \mathbf{A}\right)^{\dagger} \mathrm{vec}\mathbf{X} \,. \tag{2.24}$$

The underlying idea is to find the similarity between the implicitly imposed super-diagonal tensor $\boldsymbol{\mathcal{L}}$ and the one fitted by the least squares $\boldsymbol{\mathcal{G}}$. The way in which the similarity between the two cores is assessed is to look at the distribution of the elements in the super-diagonal and off the super-diagonal of $\boldsymbol{\mathcal{G}}$. If the elements are all close to the elements of $\boldsymbol{\mathcal{L}}$, then the model is

appropriate, otherwise a smaller number of components has to be choose. Formally, the similarity between $\boldsymbol{\mathcal{G}}$ and $\boldsymbol{\mathcal{L}}$ is given by

$$cc = 100\left(1 - \frac{\sum_{l=1}^{R}\sum_{m=1}^{R}\sum_{n=1}^{R}\left(g_{lmn} - \lambda_{lmn}\right)^2}{R}\right). \tag{2.25}$$

One of the advantages of the core consistency is that it is upper delimited, i.e. its values cannot exceed 100. High values of core consistency mean high similarity between the compared cores and then a proper model selection, while values around 50 would mean a problematic model. The core consistency can assume also negative values, meaning that the model selected is very inappropriate, and for values of core consistency near to 0 the model is invalid.

Usually, the core consistency decreases almost monotonically as the number of components increases, and as soon as the maximum number of components is exceeded, then the core consistency decreases more dramatically. This is particularly true for synthetic cases, while for real data the values could have a slow change in the slope. Due to this case, we will show different ways to choose the right number of components through the core consistency diagnostic in the following chapters.

2.5.4 Efficient Core Consistency

The core consistency is a simple way to detect the rank of a PARAFAC decomposition, but it is also hard to compute for tensors with high dimensions. This is due to the computation of the term $(\mathbf{C}\otimes\mathbf{B}\otimes\mathbf{A})^{\dagger}$, that comprises three Kronecker products and the computation of its pseudo-inverse. To avoid the products and the pseudo-inverse Papalexakis and Faloutsos [139] devised the so called **efficient CORCONDIA**, which takes advantage of the Singular Values Decomposition (SVD) to write in a computationally easier way the aforementioned term and speed up the computation of the core consistency.

Property 12. *The pseudo-inverse* $(\mathbf{C}\otimes\mathbf{B}\otimes\mathbf{A})^{\dagger}$ *can be written as*

$$\left(\mathbf{V}_a\otimes\mathbf{V}_b\otimes\mathbf{V}_c\right)\left(\boldsymbol{\Sigma}_a^{-1}\otimes\boldsymbol{\Sigma}_b^{-1}\otimes\boldsymbol{\Sigma}_c^{-1}\right)\left(\mathbf{U}_a^T\otimes\mathbf{U}_b^T\otimes\mathbf{U}_c^T\right),$$

where $\mathbf{A}=\mathbf{U}_a\boldsymbol{\Sigma}_a\mathbf{V}_a^T$, $\mathbf{B}=\mathbf{U}_b\boldsymbol{\Sigma}_b\mathbf{V}_b^T$, *and* $\mathbf{C}=\mathbf{U}_c\boldsymbol{\Sigma}_c\mathbf{V}_c^T$, *i.e. the respective SVD.*

Proof. By using the properties of the Kroneker product and the SVD of a matrix is possible to write

$$\begin{aligned}(\mathbf{A} \otimes \mathbf{B}) &= \left[\left(\mathbf{U}_a \boldsymbol{\Sigma}_a \mathbf{V}_a^T\right) \otimes \left(\mathbf{B} = \mathbf{U}_b \boldsymbol{\Sigma}_b \mathbf{V}_b^T\right)\right] \\ &= \left[\left(\mathbf{U}_a \boldsymbol{\Sigma}_a\right) \otimes \left(\mathbf{U}_b \boldsymbol{\Sigma}_b\right) \left(\mathbf{V}_a \otimes \mathbf{V}_b\right)^T\right] \\ &= \left[\left(\mathbf{U}_a \otimes \mathbf{U}_b\right) \left(\boldsymbol{\Sigma}_a \otimes \boldsymbol{\Sigma}_b\right) \left(\mathbf{V}_a \otimes \mathbf{V}_b\right)^T\right] .\end{aligned}$$

The resulting matrix $\mathbf{U}_a \otimes \mathbf{U}_b$ is orthonormal and $\boldsymbol{\Sigma}_a \otimes \boldsymbol{\Sigma}_b$ is diagonal with non-negative values. Thus, as the SVD is unique

$$\mathbf{A} \otimes \mathbf{B} = \left[\left(\mathbf{U}_a \otimes \mathbf{U}_b\right) \left(\boldsymbol{\Sigma}_a \otimes \boldsymbol{\Sigma}_b\right) \left(\mathbf{V}_a \otimes \mathbf{V}_b\right)^T\right]$$

is the SVD of $\mathbf{A} \otimes \mathbf{B}$, whose pseudo-inverse is equal to

$$\left(\mathbf{V}_a \otimes \mathbf{V}_b\right) \left(\boldsymbol{\Sigma}_a^{-1} \otimes \boldsymbol{\Sigma}_b^{-1}\right) \left(\mathbf{U}_a^T \otimes \mathbf{U}_b^T\right) .$$

□

The proof for three matrices is straightforward. The resulting equation that has to be solved in place of Eq. (2.24) is

$$\text{vec}\mathbf{G} = \left(\mathbf{V}_a \otimes \mathbf{V}_b \otimes \mathbf{V}_c\right) \left(\boldsymbol{\Sigma}_a^{-1} \otimes \boldsymbol{\Sigma}_b^{-1} \otimes \boldsymbol{\Sigma}_c^{-1}\right) \left(\mathbf{U}_a^T \otimes \mathbf{U}_b^T \otimes \mathbf{U}_c^T\right) \text{vec}\mathbf{X} .$$

In this chapter, we introduced the fundamental notions about tensor factorization problems which are needed for the work developed in the next chapters. In particular, we have illustrated the techniques and computational methods that we will use as a basis to extend the tensor decomposition framework and thus tackling the questions introduced in the Introduction.

Chapter 3

Datasets

In this chapter we introduce data on human proximity, which will be used to test our methods in Chapter 4 and 5. Recording data of human proximity is particularly important to understand what are the underlying mechanisms behind the social interactions. A standard method to record interactions among people relies on surveys, diaries and questionnaires, filled by volunteers. However, surveys provide a partial image about people interactions [144].

To have a more complete picture electronic devices are useful tools to record human proximity. For instance, Wi-Fi or Bluetooth signals, that have different spatial ranges, can be considered as a proxy of human proximity, i.e. physical co-presence.

A high temporal resolution way of recording face-to-face interactions relies on the use of radio-frequency identification devices (RFID). These devices are worn by individuals and exchange data packets in specific spatial ranges. If two devices exchange packets through the same antenna, then it effectively implies that the devices are in the same area.

Here, we consider data collected through RFID in the context of the SocioPatterns collaboration (www.sociopatterns.org). SocioPatterns is an interdisciplinary research collaboration started in 2008, involving researchers and developers from several institutions: the Institute of Scientific Interchange (ISI Foundation) of Torino, Italy; the Center of Theoretical Physics (CPT) of Marseilles, France; the Physics Laboratory of the École Normale Supérieure (ENS) of Lyon, France; and the Bitmanufaktur of Berlin, Germany.

They devised a protocol to record the interactions of people, wearing the RFID sensors in closed environments, e.g., hospitals, schools, military camps, and conferences. The protocol is defined as follows and details are provided by Cattuto et al. [145]:

1. people participating to an experiment wear small RFID wearable sensors,
2. once started, the devices exchange radio packets at close range,
3. the power of the signal can be tuned to extend the area covered by each sensor,
4. the human body acts as a shield for the radio signal, thus allowing the recording of face-to-face interactions only,
5. the information on the contacts is both stored by the sensors and sent to antennas, placed all over the environment.

This protocol allows to collect a stream of contacts that helps in the study of human dynamics. This finds its application in the development of models for the transmission of infectious diseases, such as influenza.

In the present work we will test several model based on tensor decomposition techniques on four SocioPatterns datasets, chosen for their diverse features. All the datasets are related to human face-to-face proximity measured by using the RFID sensors in closed environments. In particular, we will show the details about datasets related to two different types of environment: elementary schools, and scientific conferences.

The data collected in elementary schools are summarized in the Lyon primary school (LSCH) dataset, and in the Hong Kong primary school (HKSCH) dataset, while the other data were collected during the ACM Hypertext 2009 conference (HT09) in Italy, and the conference of the Société Française d'Hygiène Hospitalière (SFHH) in France.

In the Lyon primary school, 231 students and 10 teachers took part in the experiment as volunteers. Students and teachers were divided into 10 school classes. During the experiment, both the face-to-face interaction of people and their location in time were recoded. In particular, there were 15 locations in

the school, covered by antennas: 10 school classes, 2 stairs, 1 playground, 1 cafeteria, 1 control room. Data were collected among 2 days of observation.

In the Hong Kong primary school, the volunteers correspond to 709 students and 65 teachers divided into 30 school classes. Data where collected among 10 days of observation.

The HT09 dataset was collected during the ACM Hypertext 2009 conference, where the face-to-face proximity contacts of 113 conference attendees were recorded along 3 consecutive days.

Similarly, the SFHH dataset was collected during the aforementioned conference, where face-to-face proximity contacts of 417 volunteers were recorded during the 2 days of the conference.

Finally, data were recorded with 20 seconds of resolution, but for application purposes we aggregated the resulting network in time, with different aggregation levels, depending on the application. The resulting time-varying networks and the related aggregation levels are summarized in Tab. 3.1

Table 3.1 **Time-varying network data and aggregation levels**. In this table we reported the number of nodes and snapshots of the time-varying network created by starting from different datasets. For each dataset we provided the aggregation (in minutes) used in the applications shown throughout the thesis.

Dataset	Aggregation (min)	n. nodes	n. snapshots
LSCH	13	241	150
LSCH	15	241	131
LSCH Locations	15	241	131
HKSCH	5	774	2680
HT09	15	113	237
SFHH	15	417	129

Chapter 4

Non-negative Tensor Decomposition for Mesoscale Structure Detection in Time-varying Networks

Part of the work described in this chapter has been previously published in [12] and [13].

In this chapter, we show that tensor decomposition techniques can be applied to study time-varying networks with the aim of extracting meaningful temporal and topological patterns. We explain how to apply the NTF on time-varying networks, by focusing our analysis on time-varying social networks whose links represent the physical proximity of people in closed environments (as described in Chapter 3).

In the first part of the chapter, we provide the general procedure to carry out NTF on time-varying networks. First of all, we show how to represent the time-varying network as a tensor. Second we apply the decomposition on the network represented as a tensor to find its approximated version as a combination of sub-networks. Third, we explain the NTF results, by analysing the overall approximated network and the matrices provided as an output by the method.

We will show how that the interpretation of the components (sub-networks) strongly depends on the dataset considered. To this aim, we analyse two empirical time-varying social networks of human proximity, that even if are both related to the same environment display different characteristics in time and topology. As we will discuss throughout this chapter the components detected from the NTF can assume different meanings and they could be related to anomalous behaviours.

To illustrate this finding, we propose in the second part of the chapter a procedure based on the NTF to tackle anomaly detection problems. Here, we modify the NTF framework to iteratively decompose time-varying networks affected by anomalies that entangle both temporal and topological properties. Such anomalies correspond to groups of links sharing a correlated activity in time. The method exploits these correlated activity patterns to single out the anomalies of the network. Once detected the anomalies in the network, we finally test the performance of our method by applying it on an empirical time-varying social network.

4.1 The Decomposition of a Time-varying Network

In this section we explain the three main steps for carrying out the non-negative tensor factorization on time-varying networks. As previously outlined, the steps are:

1. the time-varying network representation as a tensor;
2. the selection of the rank and the relative best approximation;
3. the analysis of the results given by the approximation.

We start from the work developed by Gauvin et al. [11] and bring our personal contribution by comparing the decomposition on two different datasets, by explaining the effect that the NTF has on the properties of the network, and by providing an interpretation for the factor matrices resulting from the decomposition.

4.1.1 Tensor Representation

A time-varying social network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ can be represented as a three-dimensional tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|\times|\mathcal{T}|}$ having two dimensions related to nodes and the last dimension related to time.

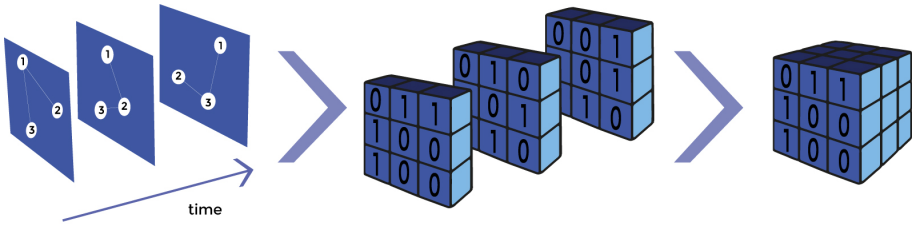


Fig. 4.1 **Tensor representation of a time-varying network**. Each snapshot of the time-varying network is represented as an adjacency matrix, having as an entry a 1 if two nodes are in contact and a 0 otherwise. The temporal succession of the resulting adjacency matrices is taken to build a tensor. The resulting tensor is binary and its slices correspond to the adjacency matrices ordered in time.

To represent a time-varying network as a tensor we adopt the snapshot sequence representation described in Section 1.8 and shown in Fig. 4.1, in which each slice of $\boldsymbol{\mathcal{X}}$ corresponds to an adjacency matrix of $\mathcal{G}$. Given the succession of these adjacency matrices the resulting tensor is binary:

$$\boldsymbol{\mathcal{X}} \in \{0,1\}^{I\times J\times K} \ ,$$

where $I = J = |\mathcal{V}|$ is the number of nodes and $K = |\mathcal{T}|$ is the total number of snapshots. Therefore, the entries of the tensor can be written in the element-wise form as:

$$\boldsymbol{\mathcal{X}} = \begin{cases} x_{ijk} = 1 \ \text{ if } \ \rho\,(i,j,k) = 1 \ , \\ x_{ijk} = 0 \ \text{ otherwise} \ , \end{cases}$$

where ρ is the presence function of the time-varying network defined in Section 1.2.

We start our analysis by applying the NTF on the LSCH and the HKSCH dataset, following the work described by Gauvin et al. [11]. The time-varying social network of the LSCH dataset comprises $I = J = 241$ individuals, whose activity was recorded for two consecutive days, with a temporal resolution of 20 seconds. However, to avoid noise given by the fine granularity of the

systems, we choose to aggregate the sensor data to have longer time intervals between contacts. In the specific case, we use 13 minutes of aggregation to avoid both noise and the loss of meaningful information given by higher levels of aggregation. However, as shown in [11] the results were found to be robust for different time aggregation levels (e.g., $5, 10, 15, 30$ and 60 minutes).

By using the aforementioned aggregation level, the resulting dataset comprises $K = 150$ snapshots in time, where the link between two nodes is present in a snapshot if it was active at least one time in the window of time that was aggregated. The resulting tensor related to the LSCH dataset is therefore

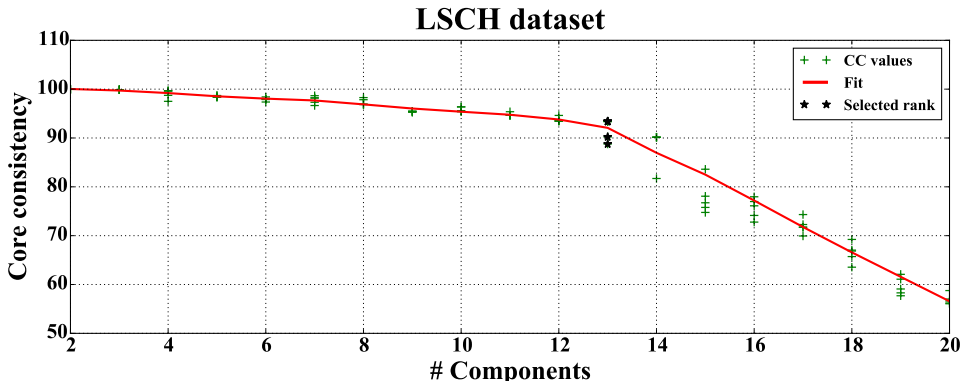
$$\boldsymbol{\mathcal{X}}_{LSCH} \in \{0,1\}^{241\times241\times150} \ .$$

Analogously, we represent the HKSCH dataset as a tensor. Here, the dataset comprises $I = J = 774$ individuals, whose activity was recorded along 10 days of observation, with 20 seconds resolution. Here we use 5 minutes of aggregation, which guarantee a good compromise between finding structures that are small enough for the purpose of the next application (see Section 4.2) and a temporal resolution in which the activity is not noisy. The temporal line is in this way divided in $K = 2680$ snapshots and the resulting tensor corresponds to

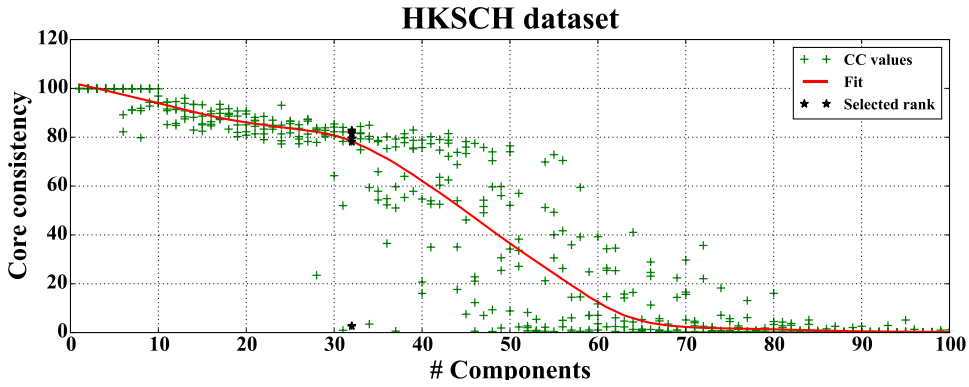
$$\boldsymbol{\mathcal{X}}_{HKSCH} \in \{0,1\}^{774\times774\times2680} \ .$$

4.1.2 The Rank

To compute the NTF on the tensors $\boldsymbol{\mathcal{X}}_{LSCH}$ and$\boldsymbol{\mathcal{X}}_{HKSCH}$ we need to find a meaningful number of components R, which describes the original dataset.



(a)



(b)

Fig. 4.2 **Core consistency values computed to find a meaningful number of components**. For each number of components we computed 5 core consistency values (green crosses) corresponding to the 5 different realizations. The red curve indicates the fit of the core consistency values, used to find the change in the slope. The selected value R for each dataset is marked with black stars. In a) we have shown the results obtained for the LSCH dataset, while in b) we have shown the results obtained for the HKSCH dataset. The image shows that the selected rank are respectively 13 and 32.

To this aim, we solve the optimization problem (2.17) with $D = 3$, i.e.

$$\min \frac{1}{2} \|\boldsymbol{\mathcal{X}} - [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_F^2 \quad \text{s.t.} \quad \boldsymbol{\lambda}, \mathbf{A}, \mathbf{B}, \mathbf{C} \geq 0 \,, \tag{4.1}$$

by varying the number of components ($r_{LSCH} \in [1, 20]$ and $r_{HKSCH} \in [1, 100]$) and by making several realizations (here 5) of the decomposition for each value of r. We compute several realizations for each rank value because the optimization problem which we solve at each time is not convex. This means

that depending on the initialization of the method (which is random) we might incur in a local minimum, thus leading to oscillating results. Thus, performing several realizations ensures to have a good estimate of the result achieved by the decomposition in correspondence to a certain number of components.

Once obtained all the decompositions we compute the corresponding core consistencies by using Equation (2.25), whose values are shown in Fig. 4.2. However, the calculation of such value is computationally costly, as it involves two Kronecker products in Eq. (2.24). Thus, we changed the implementation of this equation by rewriting the double Kronecker product through the 1-mode matricization of the tensor $\boldsymbol{\mathcal{X}}$. We know, indeed that

$$\text{vec}\left(\boldsymbol{\mathcal{X}}\right) = \text{vec}\left(\mathbf{X}_{(1)}\right) ,$$

and thus we can rewrite the product as

$$\left(\mathbf{C} \otimes \mathbf{B} \otimes \mathbf{A}\right)^{\dagger} \text{vec}\left(\mathbf{X}_{(1)}\right) = \text{vec}\left(\mathbf{A}^{\dagger}\mathbf{X}_{(1)}\left(\mathbf{C}^{T} \otimes \mathbf{B}^{T}\right)^{\dagger}\right) .$$

In this way the computation of the double product is reduced to computing just one Kronecker product.

Once the core consistency is computed, we identify the value $r = R$ where the slope of the core consistency changes, to assess the best number of components. We are looking at the change in the slope because, once identified R, for $r > R$ the model might be not appropriate as the correlated activity patterns that the method tries to uncover do not necessarily represent the overall properties in the data. For $r < R$ the core consistency values are high and indicate that the corresponding decompositions are robust. However, as different numbers of components correspond to capture different types of structures, selecting ranks smaller than R would correspond to under-fit the data and loose some important structures.

In the specific cases of $\boldsymbol{\mathcal{X}}_{LSCH}$ and $\boldsymbol{\mathcal{X}}_{HKSCH}$, by the analysis of the core consistency slope, we set the final ranks values to

$$R_{LSCH} = 13 \text{ and } R_{HKSCH} = 32 ,$$

which correspond to the highest values of rank for which the NTF is robust, according to the core consistency curve and the change in its slope.

4.1.3 The Decomposition

Once defined the number of components R for the specific datasets, we select among the 5 realizations the one with the highest core consistency value. This procedure ensures of choosing the best decomposition computed. The best decomposition is given by the approximated tensor

$$\boldsymbol{\mathcal{X}}_{app} = [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] , \tag{4.2}$$

where $\boldsymbol{\lambda}$ contains the values that normalize the different components. The vector $\boldsymbol{\lambda}$ can be seen as an indicator of the strength and importance of one component compared to the others. In general, the components found by the NTF are ranked by their λ value, i.e. the first columns of the factor matrices are characterized by higher λ values and the last columns by lower λ values. We normalize the components and save the values in the vector $\boldsymbol{\lambda}$ to compare them.

Before going into the details related to the factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and their interpretation, we analyse in the next section the characteristics of the time-varying network, approximated by the tensor in Eq. (4.2).

The Approximated Network

As introduced in Chapter 2 the NTF decomposes a tensor into the sum of rank-one tensors, that we call components. These rank-one tensors can be seen as time-varying networks in which only some of the links of the original network are active. In particular, these links belong to the same component because their activation is correlated in time.

As a result, the NTF approximates a time-varying network as a sum of sub-networks having different properties. In particular, we analyse the properties of the different components to determine which of them the NTF is able to keep in the approximation and which are different from the original network. To this aim, we first compare the original tensor $\boldsymbol{\mathcal{X}}$ and the approximated tensor $\boldsymbol{\mathcal{X}}_{app}$

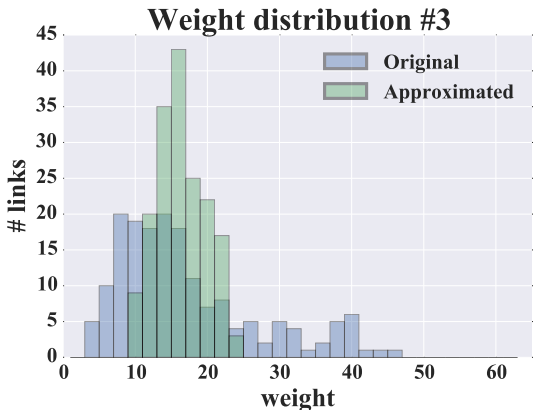
for the LSCH dataset by computing the total weight of the network. In the original case the total weight is equal to 91598, while in the approximated case the value corresponds to 10056. As the result of the NTF is an approximation of the original network, it is natural that the overall value might be a bit different. Moreover, the approximation leads to a final tensor in which the values are not binary anymore, and this is one of the reasons why the total weight is changed.

To go further in this analysis, we then compared the weight distributions and the average weights of the sub-networks corresponding to the 13 components (of the LSCH decomposition) with the same sub-networks in the original case. To compute the weight distribution in each component, we first need to find the links which are involved in each component. As we will see in the next subsections, the link memberships can be found by analysing the matrices **A** and **B**, provided by the NTF. Once identified the links belonging to each component we compute the total weight for each link, by aggregating the original time-varying network and the approximated one in time. Then, we select the weights of the links belonging to each component and we compute their average.

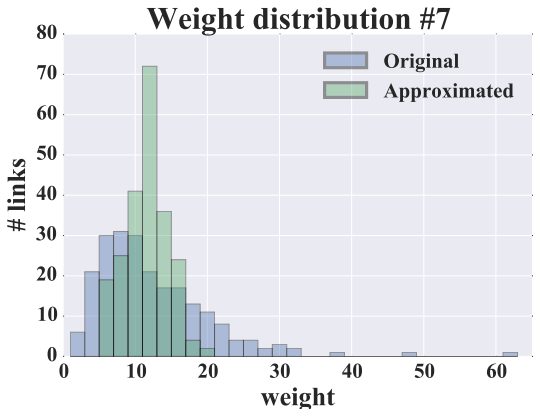
Table 4.1 **Average weights of the sub-networks corresponding to the NTF components in the original and the approximated case**. Here, the results are displayed for the decomposition of the LSCH dataset with 13 components.

Component	Original	Approximated
1	18.21	18.06
2	17.28	16.71
3	16.83	16.33
4	13.74	13.55
5	12.51	12.13
6	12.94	12.89
7	11.87	11.57
8	13.43	12.98
9	12.04	11.57
10	5.72	5.72
11	7.66	7.65
12	3.14	3.37
13	5.01	5.17

The results are summarized in Tab. 4.1 in which we can see that the average weight of the NTF components are correctly captured, as they are almost equal to the one computed in the original time-varying network. Finally we looked at the weight distribution of the sub-networks corresponding to the components and to the sub-networks in the original state.



(a) Component 3



(b) Component 7

Fig. 4.3 **Comparison of the weight distributions in the original and approximated network**. The weight distribution in the original case is broader than in the approximated one, thus indicating a redistribution of the weights in the components, in which the average weight remains fixed.

In Fig.4.3 we reported a couple of distributions (corresponding to components 3 and 7) in the original and approximated case, which are representative for the others. As we can see from the figure, the distributions in the original

cases are broader than the one in the approximated case. The weights in the approximated case are more homogeneous than in the original.

The observations on the distributions and average weights lead to the conclusion that, even if the average of weights in each of the resulting component is preserved, the contacts that occur in each component become more homogeneous than in the original case.

This observation has to be carefully taken into account when studying time-varying networks through the NTF, as heterogeneous properties are shown to be particularly important in the propagation of a process on the network. For this reason, we will propose a way to reintroduce the heterogeneity in the network after its approximation in Chapter 5.

The Factor Matrices

As a result of the approximation we obtain the factor matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ of Eq. (4.2), whose columns are the vectors of weights $\mathbf{a}_r, \mathbf{b}_r, \mathbf{c}_r$, with $r = 1, \ldots, R_{LSCH}$ and $r = 1, \ldots, R_{HKLSCH}$ for the LSCH and HKSCH dataset respectively.

Each of the components extracted is related to one particular mesoscale structure of the network. In the case of time-varying networks, as the two datasets considered, we have the first two dimensions related to nodes and the third one related to time. Therefore, on the one hand the vectors $\mathbf{a}_r$ and $\mathbf{b}_r$ of the r-th component provide the levels of membership of a node to the specific r-th component (we remind that the decomposition allows nodes to belong to more than one component, thus allowing overlapping components).

On the other hand, the vector $\mathbf{c}_r$ is related to the temporal dimension, ans provides the temporal activity of the r-th component. We have shown a profile example for the three different vectors of the decompositions and both the datasets in Fig. 4.4.

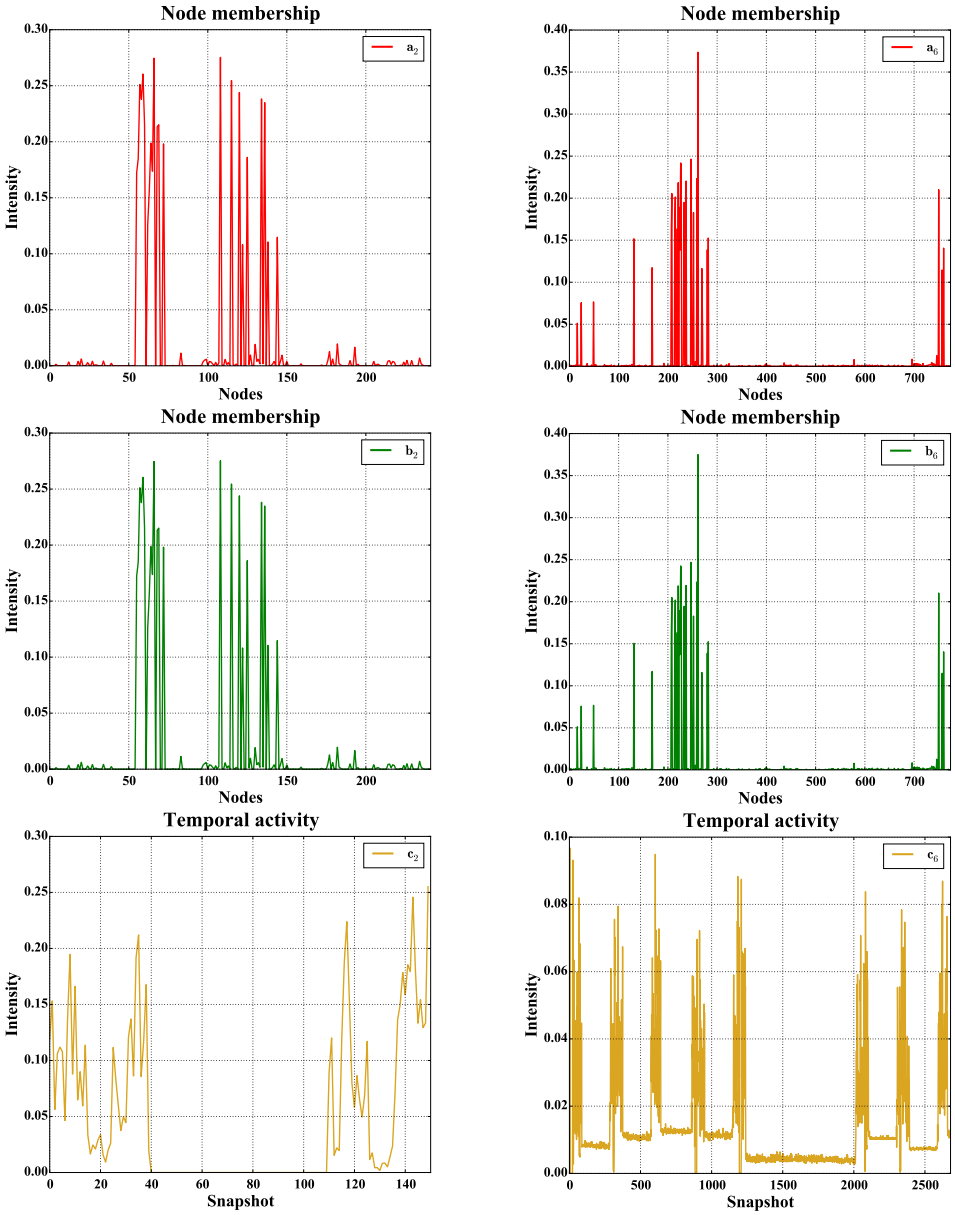


Fig. 4.4 **Node memberships a and b and temporal activity c**. The figure shows an example of **a**, **b**, and **c** related to the second component for the LSCH dataset (on the left) and the sixth component for the HKSCH dataset (on the right). The vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are normalized and their intensity corresponds to the level of membership in **a** and **b**, and to the level of activity in **c**.

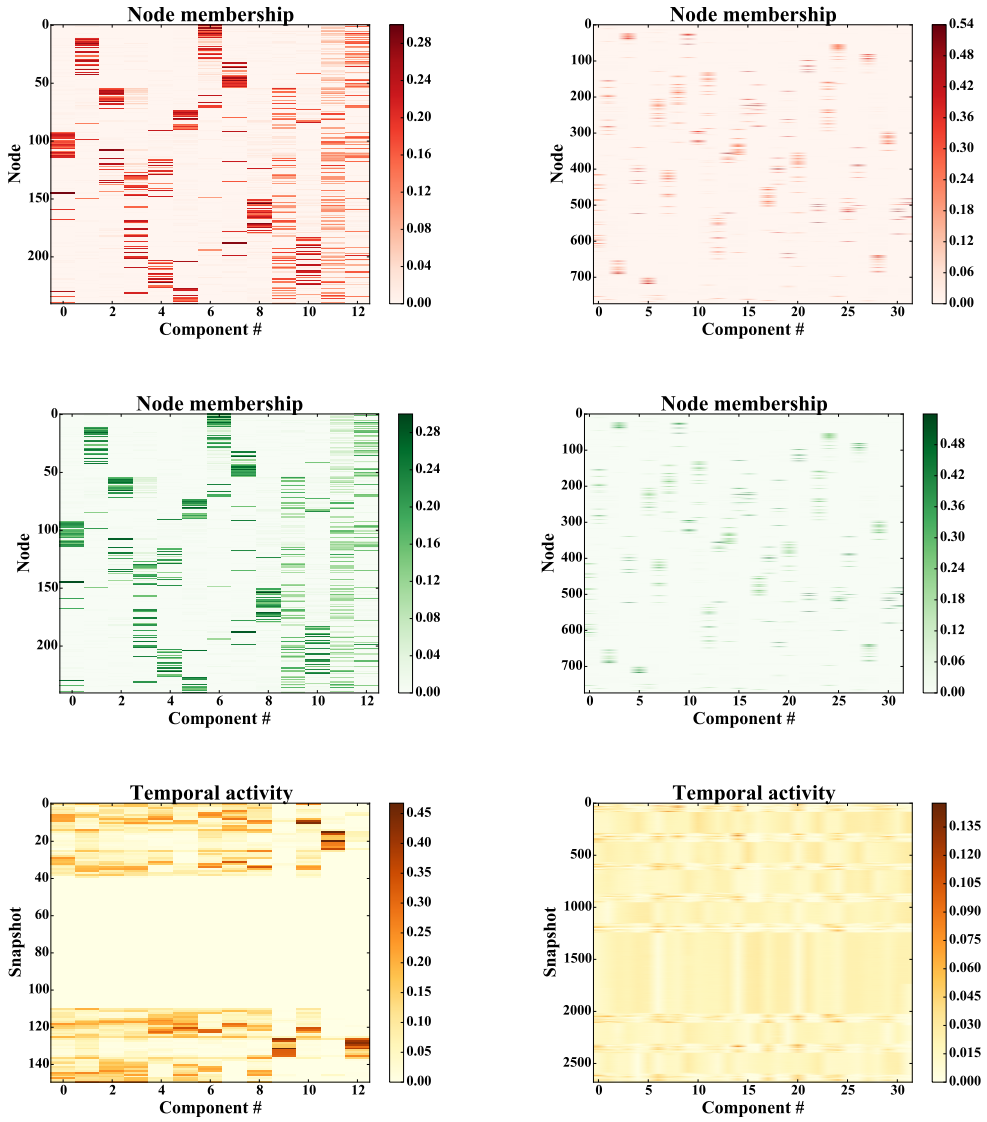
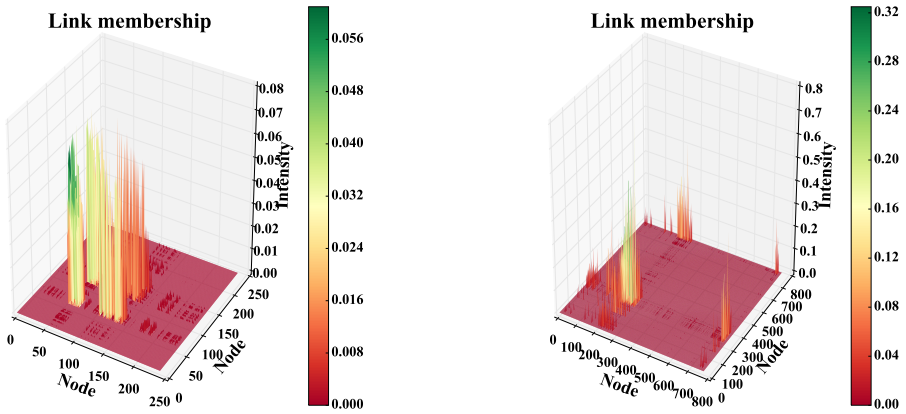


Fig. 4.5 **Factor matrices provided by the tensor decomposition**. Here, we show the factor matrices obtained by the decomposition of the LSCH dataset (on the left) and of the HKSCH dataset (on the right). The columns of the factor matrices are related to the node memberships and the temporal activity of each component. The colorbar indicates in the first two matrices (red and green) the level of membership of a node to a certain component, while in the last matrices (yellow) the intensity of the component activity at a certain time.

By observing Fig. 4.5, we can notice that the weights in the vectors vary from node to node, depending on the component. Thus, by looking at the level

of membership of each node in the components we can assign to each node the component/components which it is belonging to.

Other information can be extracted from the analysis of the combination of the matrices $\mathbf{A}$ and $\mathbf{B}$, through their Khatri-Rao product $\mathbf{A} \odot \mathbf{B}$. By recalling Eq. (2.1), we can compute the product for each component as $\mathrm{vec}\,(a^T b)$. In Fig. 4.6 we reported the result of the operation for some components that we reshaped in a matrix M of dimensions $I \times J$ for visualization purposes. The matrix provides the membership m_{ij} of each link (i, j) in the network to the component.



(a) LSCH dataset (b) HKSCH dataset

Fig. 4.6 **Link membership to the components**. Here, we displayed the values obtained by computing the product $\mathbf{A} \odot \mathbf{B}$ for a specific component. As before, we chose in a) the second component for the LSCH dataset, and in b) the sixth component for the HKSCH dataset. The values in the matrices indicate the level of membership of the links to the specific component. As we can see only part of the overall links belongs to the component. The colorbar shows the range of intensities, i.e. the level of membership, in the component.

In the LSCH case, the values among each component are partially distributed around zero values and partially distributed around a non-zero value. This bimodal distribution can be interpreted by the fact that only part of the nodes is member of a component, thus the nodes having weight around the non-zero value belong to the component and vice-versa the nodes around the zero value

do not take part in the component. The different level of membership provides a way to discern which node belongs to which component and thus to divide the nodes in different groups by using any unsupervised method, e.g., k-means with two clusters if we want to divide nodes between those belonging and not belonging to the component.

However, it is worth noting that this bimodal distribution might not exist in the case of less structured data. In the HKSCH dataset, the distribution of the weights is indeed not clearly bimodal as in the LSCH case. For this reason in the application described in Section 4.2, we define a different way to select the nodes/links that belong to a component, based on the weights accumulation.

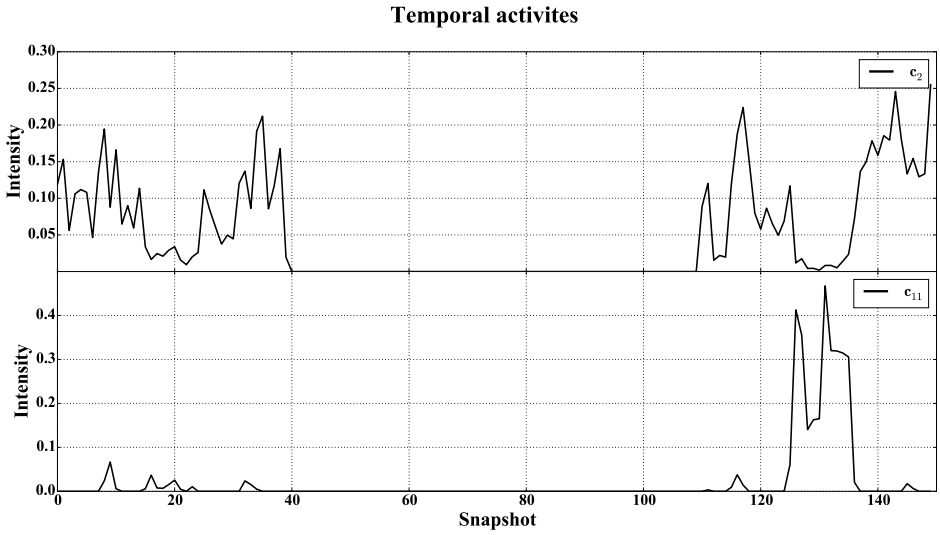


Fig. 4.7 **Example of two temporal activities of the LSCH dataset**. Here, we displayed the temporal activity related to component 2 and 11 respectively, with the aim of showing two different temporal patterns characterizing the time-varying network.

Finally, to understand the temporal structures that can be extracted from the decomposition, we analyse the activity patterns provided by $\mathbf{C}$ for each component. This analysis gives the possibility of knowing when and how each component is active. Different components can be characterized by different levels of activity (i.e. the weights of $\mathbf{c}_r$) and they are characterized by a specific activation in time. Thus, two different components might be active in different

time windows. As an example, we displayed two different activity patterns for the LSCH dataset in Fig. 4.7 and for the HKSCH dataset in Fig. 4.8.

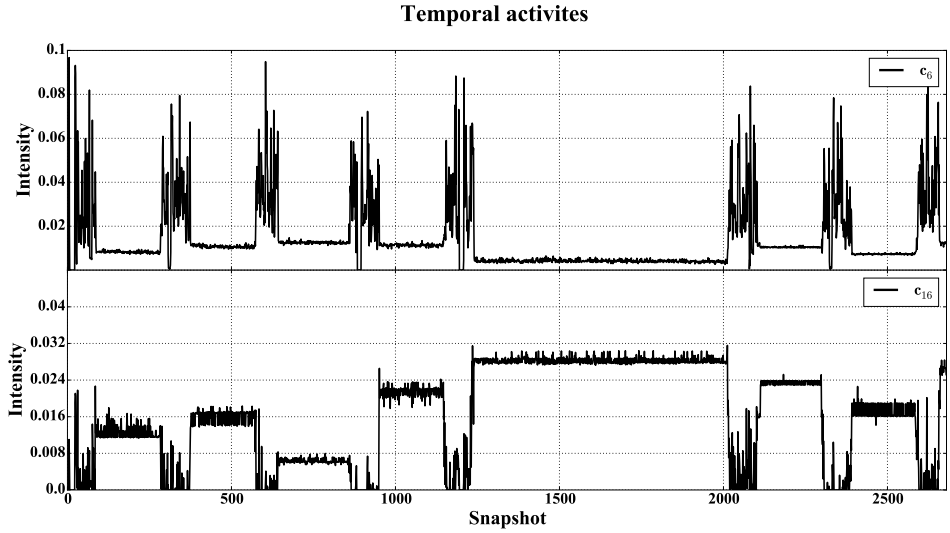


Fig. 4.8 **Example of two temporal activities of the HKSCH dataset**. Here, we displayed the temporal activity related to component 6 and 16 respectively. As before, we have shown two different temporal patterns characterizing the time-varying network.

Interpretation

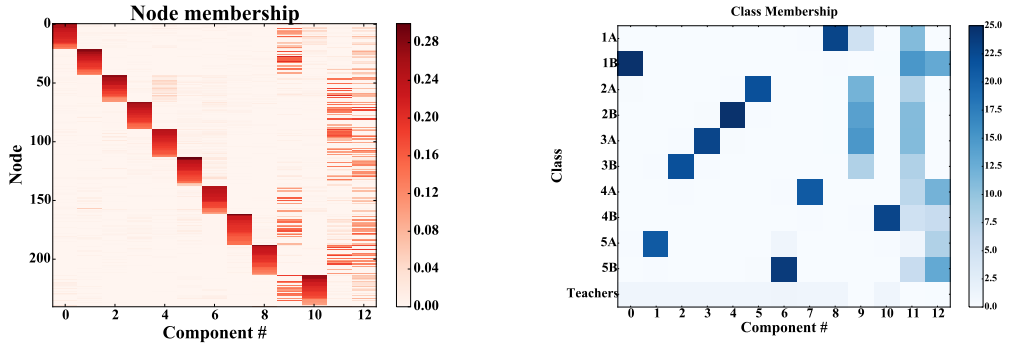
In the previous section we looked at the components extracted by the decomposition of the LSCH and the HKSCH datasets. The different columns of the factor matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ provide the information about the topological and temporal patterns that can be uncovered by applying the NTF to the time-varying networks.

We are interested in the interpretation of these topological and temporal patterns, to understand if the uncovered latent structures have an explanation that can be related in some way to the knowledge and available metadata for both the LSCH and the HKSCH datasets. The two datasets correspond to the interaction of people in two different primary schools, for which metadata related to each node are available. In particular, we know the nodes division in the school classes and the schedule of the school activities (types of classes, lunch breaks, etc.).

Consistently to the study of Gauvin et al. [11], the decomposition of the LSCH dataset gave as a result 13 different components:

- 10 components are characterized by a block structure corresponding to mutually disjoint communities in the network;
- 3 components (called **mixed**) are characterized by a significant overlap with the others and are consistently bigger than the other 10 components.

We remark that the selected number of components depend on the method that we choose to select them (here the change of slope), and that it is possible to select a similar value without changing the meaning of the decomposition. The choice of a slightly different number of components would lead to the merging/splitting of two or more components. For instance, by selecting $R = 14$ we would find 12 components equal to the $R = 13$ case, while one component would be divided in two.

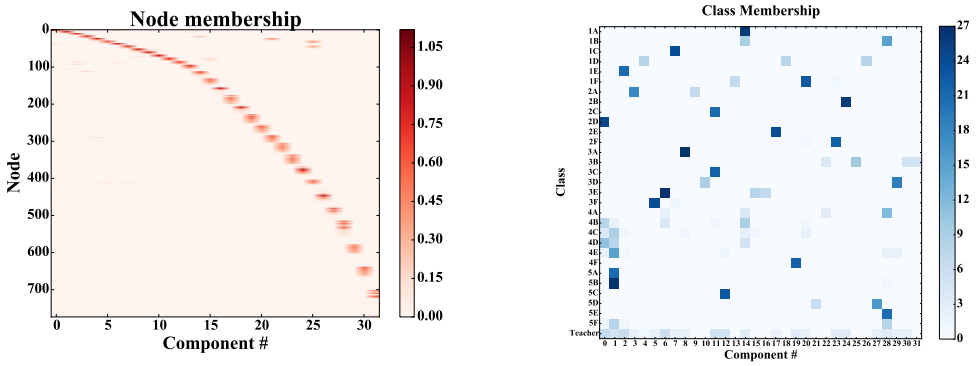


(a) Block and mixed components

(b) Class-component map

Fig. 4.9 **Result of the NTF decomposition of the LSCH dataset with $R = 13$ components**. In a) we reorganized the order of the nodes in the factor **A** for visualization purposes. The new order allows to highlight the block structure of 10 components and the mixed nature of the remaining 3 components. The colorbar indicates the level of membership of nodes to components. In b) we show the map between nodes belonging to a component and the metadata of the school class. The colorbar shows how many nodes belonging to a certain component are also belonging to a specific school class. As the figure highlights, there is a strong correspondence between the "block" components and the school classes, while people belonging to different school classes are present in the "mixed" components.

The block structure of the components is shown in Fig. 4.9(a) as well as the mixed components 9, 11 and 12, while the mapping between the nodes involved in a component and their school class can be seen in Fig. 4.9(b). For visualization purposes, in Fig. 4.9(a), we have ordered the values in the matrix **A**, by looking at the node membership to the components. To this aim, we applied a k-means with two clusters to define if a node belongs to a component and finally we ordered the nodes belonging to the same component by their membership value. As we can observe, there is a clear correspondence between the 10 components characterized by the block structure and the school classes, while the mixed components include nodes belonging to different school classes. Further analysis, on the temporal activation of the mixed components and the school schedule, made emerge that the activation of the mixed classes is related to social activities in the school such as the lunch break, thus explaining the proximity of individuals belonging to different school classes at the same time.



(a) Block and mixed components (b) Class-component map

Fig. 4.10 **Result of the NTF decomposition of the HKSCH dataset with $R = 32$ components**. In a) we reorganized the node order by their level of membership for each component. As we can see, the components are almost disjoint, i.e. they do not have nodes in common. Moreover the new order highlights the different sizes of the components. In b) we have tried to map the nodes belonging to the components to the school classes, by means of the metadata. Here the block structure is not clear anymore, as multiple components can be mapped into the same class and vice-versa. The colorbar indicated the number of nodes of a components belonging to a certain class.

We carried out the same analysis for the HKSCH dataset, whose decomposition provided 32 different components. Here, we found that all the 32

components are almost disjoint but contrarily to the LSCH case, the block sizes are strongly different from each other. Moreover, as we can see from Fig. 4.10(a) there are many nodes for which the k-means with two clusters was not able to assign a component. This could be due to the presence of nodes who do not have enough interactions to be assigned to any components. Finally, if we look at the map in Fig. 4.10(b) we can observe that the correspondence between school classes and components is not straightforward. Indeed, not only several school classes are assigned to the same component (mixed case) but also several components are assigned to one school class (i.e. the individuals of one class are divided in different components). Therefore, the components found by the decomposition of the HKSCH dataset cannot be obviously related to the school classes or to some mixed activity due to the school schedule and need to be further investigated. A possible interpretation of this observation could be the presence of different groups of correlated activity within a school class.

As we have seen in the mixed components of the LSCH dataset and for the components in HKSCH, the NTF can be used to uncover mesoscale structures that we cannot identify by simply looking at the metadata. Thus, metadata enable to understand if the results achieved by the NTF are related to some known behaviour, however they cannot be used as a pure validation of the method.

As we will see by carrying on the analysis of the HKSCH dataset, these complex mesoscale structures are characterized by temporal patterns which are active in out-schedule times, e.g. nights, weekends, etc. We will consider such correlated activity patterns as anomalous behaviours affecting the time-varying network.

To detect such anomalies, we propose an iterative method based on NTF, which we explain in the next section. The main purpose in the application of this method is to identify anomalous activity patterns, i.e. groups of links whose activation in time is correlated, and erase them from the network to clean our dataset.

4.2 Anomaly detection

Time-varying networks could be affected by anomalies [146, 147]. The detection of the latter would correspond to monitor the evolution of the network to identify some unusual events and changes in the normal trends. The definition of anomalies in time-varying networks strongly depends on the specific application of interest. In general, an anomaly is characterized by an abrupt change in the connectivity patterns between nodes in the network, e.g., the establishment of new links between nodes that normally are unrelated, or the activation of links in unexpected times.

Usually, in anomaly detection methods some models that define the "normal" characteristics of the network studied are built. The main purpose is to find outliers and anomalous behaviours in the system by observing these characteristics over time. The presence of such changes in the link activations could imply the presence of malicious behaviours in the network. Therefore, detecting anomalies that resemble normal activities but whose presence could lead to inaccurate analysis, is important.

Traditional techniques often rely on the presence of outliers and look for changes in the system that are either structural or temporal. An overview of the different anomaly detection methods and their applications was provided by Chandola et al. [148], Akoglu et al. [149]. Some of the most used anomaly detection techniques are based on classification methods [150, 151], parametric statistical modelling [152], and spectral analysis [153, 154].

However, anomalies in time-varying networks can also appear at a mesoscale level, by entangling both temporal and topological characteristics and resembling normal behaviours. The detection of such anomalies observed in time-varying networks calls for further research. Indeed, as these anomalies involve both temporal and structural characteristics they cannot be considered as simple outliers and their detection is particularly challenging [155].

In the next sections, we propose and anomaly detection method with the aim of identifying anomalous correlated activity patterns in time-varying networks. The method, published in [12, 13], is based on an iterative procedure, which includes the NTF to uncover both normal and anomalous activity patterns. The extracted patterns are labelled as anomalous and non-anomalous and a

tensor mask is used to erase the anomalous contacts from the network. We apply the method to the HKSCH dataset whose anomalies are described in the next subsection.

4.2.1 HKSCH Anomalies

During an experiment involving the RFID sensors, there exist several possible settings in which anomalies can arise. We can divide these anomalies into two main groups: the anomalies that occur because something in the system unusually or abruptly changes and that are related to real activity patterns, and the anomalies that are due to events and actions that are not related to a real activity between people. To the first group, belong all the events in which an unusual circumstance suddenly happens. This is the case of a safety drills in schools or offices, where people are forced to meet in the school yard or outside the work place. To the second group, belong all the events in which an event uncorrelated to the people interactions occur. As an example, sensors could be left in boxes in different arrangements, continuing to record their proximity with their neighbours, or people could leave their sensors somewhere or lose them for a certain period of time during the experiment.

In the case of the HKSCH dataset we recorded proximity interactions among people by following the wearing protocol for the sensors explained below:

1. we gave the sensors to participants and associated each of them to the class the participant belonged;
2. with the aim of measuring the proximity relations of the volunteers during the school activities, participants wore sensors during the school day;
3. at the end of each school day, the volunteers left the sensors in classes in different settings.

Since participants left sensors in different arrangements at the end of the school activities, we recorded a great amount of interactions between sensors even in time in which the experiment was not involving interactions between individuals. Therefore, these correlated activity patterns are not related to a real social

activity and we thus aim at detecting such anomalies in the time-varying network.

The data cleaning in presence of this type of anomalies is particularly challenging as such anomalous patterns entangle both topological and temporal features and may overlap with normal behaviours. To this aim, we devised an iterative method based on NTF and applied it on the HKSCH dataset as described in the following section.

4.2.2 Iterative Method

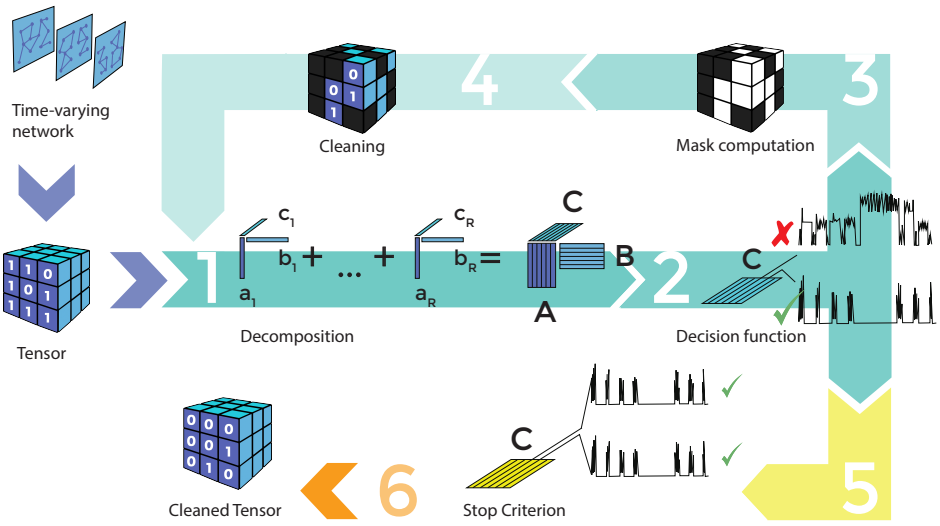


Fig. 4.11 **Iterative data cleaning procedure:** The temporal network is represented as a tensor $\boldsymbol{\mathcal{X}}$. **1.** $\boldsymbol{\mathcal{X}}$ is decomposed via non-negative tensor factorization. **2.** The temporal activity patterns $\mathbf{c}_r$ of the components are divided into anomalous and non-anomalous by a decision function. **3.** A mask $\boldsymbol{\mathcal{M}}$ is computed on the basis of the structural and temporal properties of the components. **4.** The tensor entries associated with anomalous components are zeroed out in $\boldsymbol{\mathcal{X}}$, and the new tensor $\boldsymbol{\mathcal{X}}'$ becomes the input of the successive iteration.

By starting from the tensor representation of a time-varying network as described in Section 4.1.1, we build the initial tensor $\boldsymbol{\mathcal{X}}_{HKSCH} \in \mathbb{R}^{I \times J \times K}$, from a time-varying network having $I = J = 774$ nodes and $K = 2680$ snapshots. The iterative procedure, illustrated in Fig. 4.11, is divided in 4 major steps:

1. the computation of the tensor decomposition of $\boldsymbol{\mathcal{X}}_{HKSCH}$;

2. the use of a decision function to separate the components into anomalous and non-anomalous according to their temporal activity $\mathbf{c}_r$

3. the analysis of topological and temporal patterns of the anomalous components to compute a tensor mask $\boldsymbol{\mathcal{M}}$, on the basis of the membership of the links to the components and their activation times;

4. the application of the mask to the tensor $\boldsymbol{\mathcal{X}}_{HKSCH}$ to zero out the anomalous entries.

The resulting tensor $\boldsymbol{\mathcal{X}}'_{HKSCH}$ after the first iteration is used as an input for the successive iteration, and the process is repeated until no more anomalous components are selected by the decision function.

By following the steps in the procedure, first of all we decomposed $\boldsymbol{\mathcal{X}}_{HKSCH}$ by solving the optimization problem (4.1). In order to select the final number of components R we follow the rank analysis described in Section 4.1.2. In particular, we computed the core consistency for several ranks ($r = 1, \ldots, 100$) with 5 realizations for each rank value. We then compute the curve corresponding to the core consistency over the interval $[1, ..., R_{max}]$ and we select the final rank R with the following procedure.

We compute the standard deviation between the 5 realizations for each number of components. We then study the standard deviations between the realizations to detect the rank in which the different core consistency values start to oscillate. To this aim, we apply the Otsu method [156] which assumes that the function given as an input contains values following a bimodal distribution. The method computes the optimal threshold which can be used to separate the two groups of values. The threshold is defined by minimizing the intra-class variance, or vice-versa by maximizing their inter-class variance. By using the threshold provided by the Otsu method, we can identify the first value of the standard deviation that exceeds the threshold and the corresponding rank R_{ex}. As a result, the selected rank is $R = R_{ex} - 1$.

This procedure is based on the observation that once the final rank R is reached the results of the NTF for higher ranks give core consistency values that start to oscillate, while in the case of lower ranks the results are stable. In the iterative approach, we preferred this rank selection method to the change of slope detection described in Section 4.1.2 as it is more restrictive: the selected

ranks will be lower than the one provided by the other method. In this way, we ensure to find only meaningful component at each iteration. Moreover, even if some meaningful components are not included in the factors, by selecting a lower rank value, we will be able to successively uncover them as the method is iterative and at each iteration anomalous components are erased from the tensor. The technique is thus robust as the rank variation in the selection procedure does not affect the final outcome of the process.

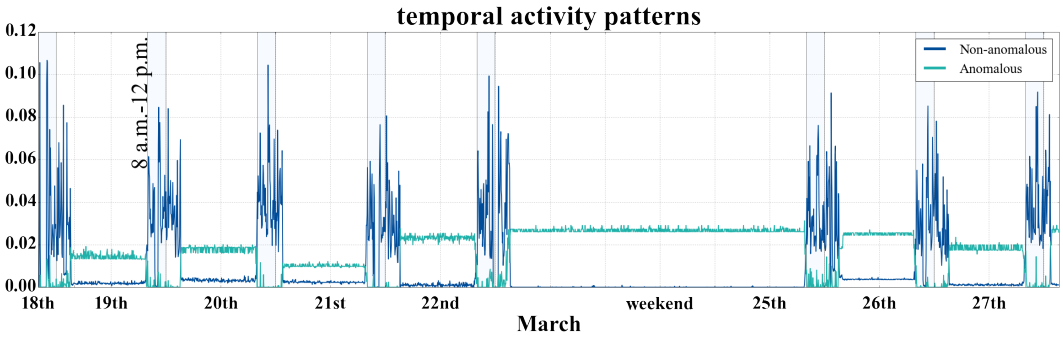


Fig. 4.12 **Example of two normalized activity patterns obtained after an iteration of the iterative method.** The components are representative of a non-anomalous (blue) and an anomalous (green) behaviour related to the case study. The non-anomalous pattern is mostly active on or around the time windows $[8a.m., 12p.m.]$, marked with a shaded area.

Once selected the best decomposition over the five runs corresponding to the selected R, the procedure divides the components into anomalous and non-anomalous. In particular we use the temporal activity patterns of the decomposition as a basis to identify the anomalous components and devise a decision function to divide them from the others. Fig. 4.12 illustrates an example of two activity patterns from two components, a non-anomalous and an anomalous one. Considering the entire time series, it is possible to define a suitable daily window (here from 8 a.m. to 12 p.m.) in which non-anomalous components are mostly active, whereas anomalous components exhibit low activity. Hence, the decision function adopts the times at which these strong activities appear as a guideline to single the anomalous components out. In particular, for each temporal activity pattern $\mathbf{c}_r$, we selected the values that exceed the mean plus the standard deviation of the vector $\mathbf{c}_r$ and we considered them as the strongest intensity of the component. Successively, we compute the time in which this activity was taking place and we select those times that

fall in the daily time windows. Finally we count the number of the values located in the time windows. If this number increases after a random shuffling of the time series, it means that the most of the activity is located outside the daily window and the temporal activity pattern $\mathbf{c}_r$ is consequently labelled as anomalous. Contrarily, if the number decreases after the shuffling then most of the activity is located in the window and the corresponding temporal activity pattern is labelled as non-anomalous.

The next step in the procedure is the computation of a tensor mask to clean the original tensor. To build the mask, we take into account both the temporal and topological informations by:

1. identifying the times at which the anomalous interactions occur;
2. selecting the links involved in the anomalous component.

The temporal activity patterns found by the NTF are bursty, i.e. alternate between periods of activity and periods of negligible activity, as can be observed by the examples in Fig. 4.12. Moreover, the fluctuations of the anomalous temporal patterns in the activity periods are small, as the sensors continuously recorded the proximity with the same neighbours. However, from one activation period to another the level of intensity may change, due to the different settings from one day to another. Hence, to select the activity periods of the anomalous components, we divide the related temporal activity patterns $\mathbf{c}_r$ classified as anomalous into 10 intervals (from 12 a.m. to 12 a.m.), excluding the weekend. In each interval, the times at which anomalous interactions occur can be selected by the application of a step detection algorithm to the time series. Here, we adopt the Otsu algorithm [156] to compute the threshold value significant to the step detection. The application of this method on each of the intervals allows to recognise the presence of jumps for each time series and thus identify the starting and ending times at which anomalies occur. As a result, we collect a set of times in which the anomalous interactions occur for each anomalous component.

We successively identify the links involved in each anomalous component r by studying the topological patterns $\mathbf{a}_r$ and $\mathbf{b}_r$, which provide the nodes membership to the components. As explained in Section 4.1.3, the identification of the links membership to a component r can be done by taking into account

the outer product ab^T. Given a link (i, j), its membership to the component r is defined by:

1. calculating the product $a_{ir}b_{jr}$;
2. computing the square of the values $(a_{ir}b_{jr})^2$;
3. sorting the values $(a_{ir}b_{jr})^2$ in descending order;
4. summing the values until the sum exceeds the 95% of the norm $\|\mathbf{a}_r \cdot \mathbf{b}_r^T\|_F^2$;
5. selecting the links included in the sum.

This procedure applied to the anomalous components gives as a result the links belonging to the component. Finally, we can compute the tensor mask by adopting both the activation times k and the links (i, j) previously detected. Hence, the tensor mask is defined as

$$\boldsymbol{\mathcal{M}} = \begin{cases} m_{ijk} = 0 \text{ if } (i, j, k) \text{ is anomalous}, \\ m_{ijk} = 1 \text{ otherwise}. \end{cases}$$

The mask can be applied to the original tensor $\boldsymbol{\mathcal{X}}_{HKSCH}$ by computing their Hadamard product to zero out the interactions associated with anomalies. The resulting tensor

$$\boldsymbol{\mathcal{X}}'_{HKSCH} = \boldsymbol{\mathcal{M}} * \boldsymbol{\mathcal{X}}_{HKSCH},$$

becomes the input of the successive iteration in the procedure. The described iterative method ends when it reaches a state in which no more anomalous patterns are detected.

4.2.3 Results and validation

In this section we report the results obtained by applying the iterative method on the HKSCH dataset, as explained earlier. We report the results, obtained after 39 iterations of the procedure, in Fig. 4.13. Here, we display the total number of contacts at each time of the experiment. For visualization purposes, we divided the number of contacts by school class via the available metadata of

the class membership of each sensor, even though we did not use any metadata in the iterative procedure.

The right hand side of Fig. 4.13 shows the total amount of contacts in the original tensor $\boldsymbol{\mathcal{X}}_{HKSCH}$: here, we found a comparable amount of contacts during the school closing and opening time. On the contrary, the left hand side of Fig. 4.13 shows the total amount of contacts of the resulting tensor $\boldsymbol{\mathcal{X}}'_{HKSCH}$ after the iterative procedure: here, we can observe that a great amount of contacts are identified as spurious by the method and thus erased from the original dataset.

We are now interested in validating the method, to see if the contacts that were detected and erased by the procedure were effectively related to anomalies.

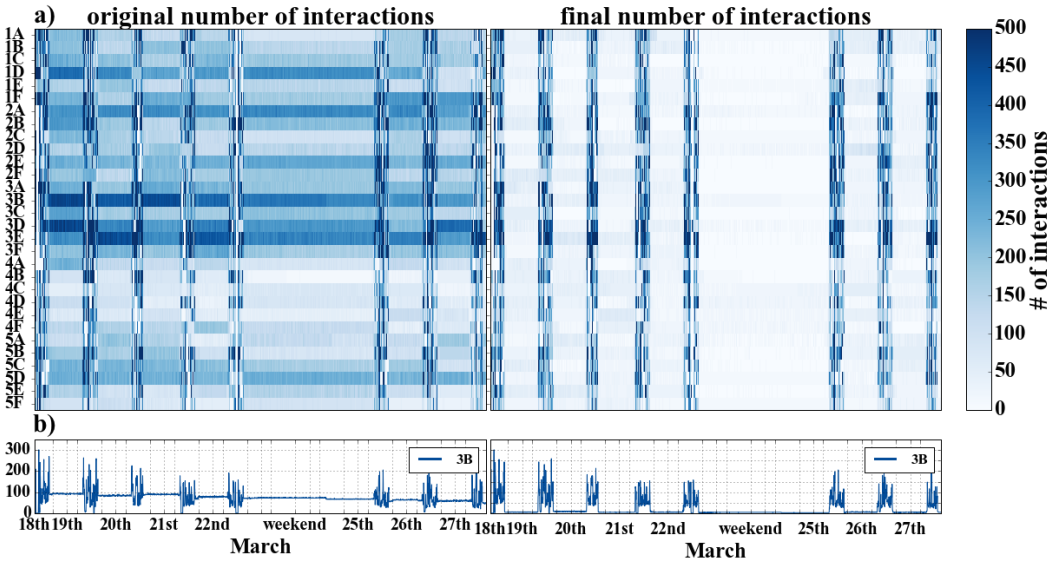


Fig. 4.13 **Evolution of the number of interactions with time measured on the original and cleaned tensor for each school class. a)** The colour of the pixels encodes the number of interactions recorded at the corresponding time. The class labels were used to group the interaction of nodes belonging to the same class. While interactions in the original state are distributed along the entire timeline, the cleaning procedure managed to identify and remove most of the anomalies. **b)** Time series representing the evolution of the number of interactions among people belonging to one selected school class, measured both on the original and cleaned tensor. On the left, it is possible to observe the great amount of interaction events recorded by sensors during the entire timeline. On the right, the corresponding time series after the application of the iterative method is shown.

To this aim, we built a reference tensor $\boldsymbol{\mathcal{X}}_{ref}$ by using the metadata of the school classes and the school schedule to clean the original tensor in the following way:

1. we considered the off-schedule interactions as anomalous;
2. we collected the starting and ending times of the scheduled activity for each class;
3. for each class we selected the nodes corresponding to the sensors belonging to the class and we computed the total number of contacts as a function of time;
4. we detected the steps in these time series to find the intervals delimiting the class activities;
5. we masked away the contacts, whose activations occur in intervals that fall outside the class schedule.

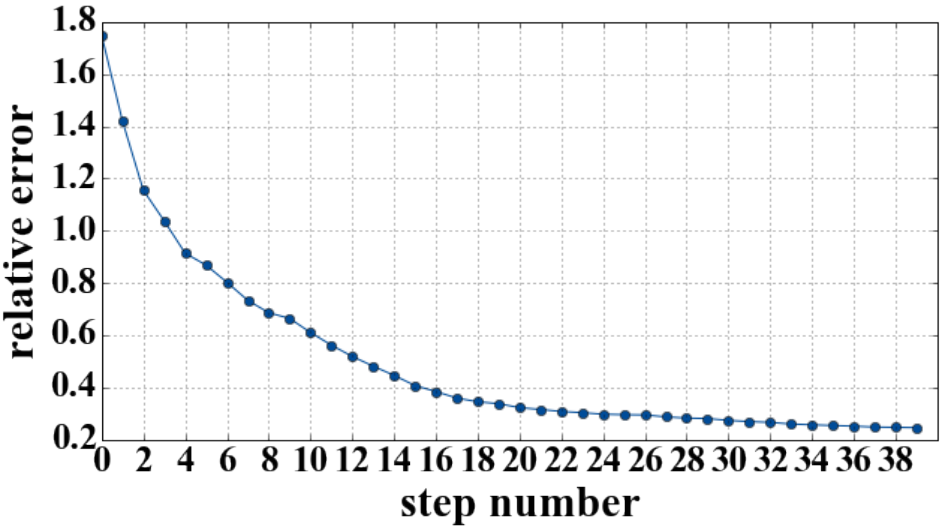


Fig. 4.14 **Relative error** between the cleaned and the reference tensor, computed at each iteration of the method by using the L_1-norm.

We adopted the resulting tensor $\boldsymbol{\mathcal{X}}_{ref}$ to carry out the evaluation of the iterative method at two levels: the global level and the school class level. At the

global level, we compared the ground truth with the cleaned tensor $\mathcal{X}'_{HKSCH}$ at each iteration, by computing the relative error

$$err = \frac{\|\boldsymbol{\mathcal{X}}'_{HKSCH} - \boldsymbol{\mathcal{X}}_{ref}\|_{l_1}}{\|\boldsymbol{\mathcal{X}}_{ref}\|_{l_1}} .$$

This quantity, shown in Fig. 4.14, monotonically decreases as the number of iterations increases and successively stabilizes around the low value of 0.2.

Table 4.2 **Scores of the tensor entry classification**. The table report the Precision Recall and F1-score obtained by comparing the reference tensor to the one obtained as a result of the iterative procedure.

	Precision	**Recall**	**F1-score**
Anomalous	0.99	0.87	0.92
Non-anomalous	0.81	0.99	0.89

Furthermore, we used the reference tensor $\boldsymbol{\mathcal{X}}_{ref}$ to label the entries of the original tensor $\boldsymbol{\mathcal{X}}_{HKSCH}$ as anomalous and non-anomalous. We compared this labelling with the one of the cleaned tensor $\boldsymbol{\mathcal{X}}'_{HKSCH}$ to compute the overall accuracy of the method that we found to be 0.91 and the recall and precision values at each entry level, which we reported in Tab. 4.2. In particular, the method recall provides how many relevant values are selected as it is defined as the fraction of true positives over the relevant elements, and the precision of the method quantifies how many selected elements are relevant, as it is defined by the fraction of true positives obtained over the sum of true and false positives. We also computed the $F1-$score, which is a weighted average of the precision and recall:

$$F1 - \text{score} = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} .$$

As shown in Tab. 4.2 these measures highlight the high performance achieved by the iterative procedure.

At the school class level, we tested the efficiency of the iterative method by comparing the temporal activities of the classes in $\boldsymbol{\mathcal{X}}_{ref}$ and in $\boldsymbol{\mathcal{X}}'_{HKSCH}$. The temporal activities are computed by summing, separately for each school class, the total number of contacts recorded at each time. We then compare the time series corresponding to the same school class in the two tensors

through the similarity measures of the Pearson coefficient and the dynamic time warping [157].

As a result, we found that all the Pearson correlation coefficients between the cleaned and the ground truth time series fall in $[0.89, 0.99]$. These results are statistically significant, as it is shown by the $p-values$, which are lower than 10^{-3}.

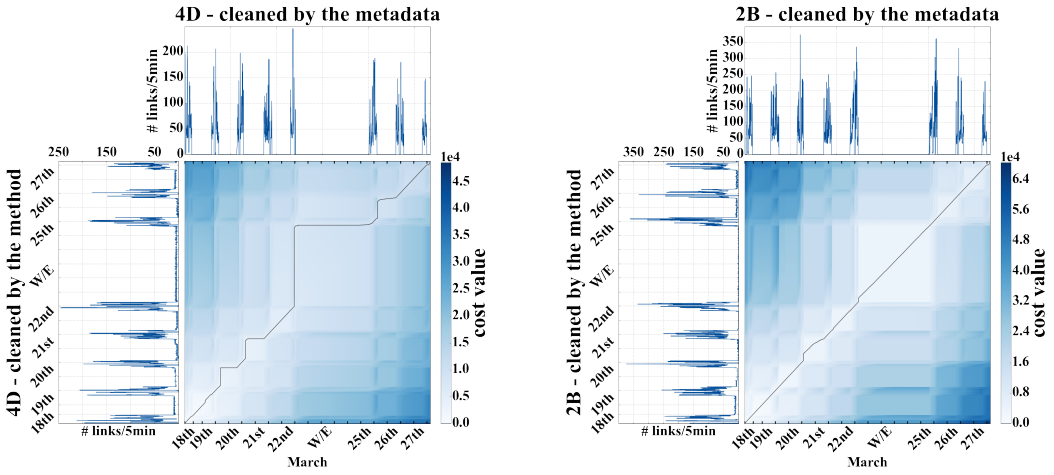


Fig. 4.15 **Example of two dynamic time warping cost matrices**, in which we compare the time series measured on the cleaned tensor and the reference tensor. The series are computed at the level of the classes and correspond to the evolution of the number of interactions in time. Here, we show classes $4D$ and $2B$, whose Pearson coefficients are respectively 0.89 and 0.97. We compute the cost matrix for the alignment of the time series to evaluate their similarity. The colour intensity indicates the cost value in each matrix position. Here, the block shapes are due to windows of time corresponding to out of school schedule periods, in which values are mostly homogeneous. The lines shown in the matrix represent the optimal warping path, which is near the matrix diagonal.

Finally, we computed the dynamic time warping distance between each couple of time series (original and cleaned). The dynamic time warping algorithm provides a similarity metric by looking for an optimal alignment between two time series. The result of this operation is summarized by a cost matrix, whose entries are the alignment costs of each pair of the time series values. In particular, the cost is given by computing the Euclidean distance.

By means of the cost matrix the dynamic time warping finds an optimal warping path, that is close to the matrix diagonal if the considered time series are similar. Here, we compute the cost matrix of each school class by comparing

the time series of the number of interactions per class of the cleaned and the reference tensor. We reported an example of two cost matrices in Fig. 4.15. The depicted costs matrices are related to the school classes $4D$ and $2B$, whose Pearson coefficients are respectively the lowest (0.89) and the median in the range of coefficient values (which is equal to 0.97) among all the classes. We obtained analogous results in terms of cost matrices and Pearson coefficients in the case of the other classes.

We highlight the fact that the validations achieved by computing the dynamic time warping are consistent with those obtained by the Pearson coefficients and indicate that the compared time series are highly similar. This high similarity is an indicator of the efficiency of the method to clean the data, although it only processes the structural and temporal properties of the network.

4.3 Conclusion

In the present chapter, we have discussed the use of non-negative tensor factorization to reveal mesoscale structures in time-varying networks. We found that a time-varying network can be split in several sub-networks through the NTF. These sub-networks are characterized by links having a correlated activity in time.

We have seen that, when metadata are available, some of these patterns can be linked to expected groups (e.g., groups of nodes belonging to the same school classes). However, some of the uncovered structures might be not linked to something already known and thus need further investigation on their nature. Indeed, we have shown, through the application on the HKSCH dataset, that some of these correlated activity patterns could be related to anomalous behaviours.

Hence, we devised a method based on the NTF to systematically identify and remove anomalous connectivity patterns from time-varying networks. Our method operates on the tensor representing the time-varying network studied and iterates a procedure consisting in its decomposition, the labelling of the components as anomalous and non-anomalous, the construction of a tensor

mask including all the anomalies detected, and the application of the mask on the original tensor to erase the anomalous contacts from the tensor. Our method thus enables a semi-supervised cleaning of time-varying networks.

Furthermore, we investigated the performances of the method by building a reference tensor that we used as a ground truth. We relied on the metadata to clean the original tensor and we compared the resulting tensor to the output of the method. We have shown by means of different validation levels that the method achieves high accuracy in the identification of the anomalies in the HKSCH dataset, proving that the application of the method is possible on real data. The procedure and results of the present chapter are published in [12, 13].

In conclusion, the use of the NTF to find meaningful patterns in time-varying networks can be extended to different applications. In the applications we presented, we have shown that the NTF is able to approximate a time-varying network by keeping some of the properties of the original network while changing others. We also observed that the NTF introduces some level of homogeneity in the approximated network that is not present in the original case. In particular, the number of contacts in the time-varying network tends to be homogeneously distributed.

In the next chapter, we will see how NTF reveals to be useful in the recovery of missing information in time-varying networks. The rationale behind the fact that we can infer some lacking information derives from the evidences found in this chapter about the correlated activity patterns. Indeed, since with the NTF we are able to detect groups of links having similar activity, having partial information about a link could be enough to recover the unknown part by assigning to the link the properties of its group. To this purpose, we extend the model based on the NTF and we also consider the case in which additional data sources are available. This is particularly advantageous not only to recover missing information but also to detect correlated patterns involving diverse dimensions, e.g., space.

Chapter 5

Interplay between Time-varying Networks and Dynamical Processes: impact of the mesoscale structures

As mentioned in Chapter 1 temporal and structural properties of time-varying networks have a strong impact on the outcome of dynamical processes. In Chapter 4 we presented a technique to highlight temporal and topological characteristics of networks at the mesoscale level. We want now to investigate the interplay between the mesoscale structures detected and dynamical processes. This is useful to understand how the dynamics changes in response to the presence or absence of certain network characteristics.

Recalling what we explained in the previous chapter, tensor decomposition techniques and NTF in particular are effective tools to extract complex mesoscale structures that characterize the network in time and topology. These structures involve different link activations at different times, resulting in correlated activity patterns which we know that have an impact on dynamical processes.

By taking into account these observations, we will tackle the problem of missing data in the network. In particular, we aim at recovering the unknown information at the mesoscale level, exploiting the correlated activity patterns of the network. Second, since missing data in networks strongly bias the outcome of a dynamical process, we focus on recovering the properties of the original

network needed for the dynamics.

To solve a missing data recovery problem we focus on two main steps. In the first step, we try to recover the properties of time-varying networks by the raw information provided by different tensor decomposition techniques. Here, we adopt the NTF framework to recover meaningful information from the partial network, and we also present an extension of this framework, in which we can take advantage of external data sources when available. This last procedure is suitable both to recover the missing values through additional information and to extract meaningful properties from multiple tensors at a time.

As explained in the previous chapter, the approximation of time-varying networks through tensor decomposition techniques are characterised by less broad distributions in the number of contacts. We also know that heterogeneity properties of networks strongly influence the outcome of dynamical processes. Thus, we divide our procedure to retrieve the lost information in two parts: the approximation of the network and its reconstruction. In the first part, we use the relevant patterns provided by the decomposition to recover the missing information at the mesoscale level. In the second part, we reconstruct the network by using some of the properties that are important to find the correct outcome of dynamical processes.

Finally, we test our method by simulating spreading processes on top of the original network, the partial network and the reconstructed network. As we will see, our procedure manages to achieve a good agreement between the outcome of the spreading process on the reconstructed and the original network.

5.1 Missing data recovery

The advances made in data collection, due to the accessibility to internet and electronic devices, led to the availability of high resolution data. As a consequence, the use of such data allowed to better study complex systems as time-varying networks.

However, as we have seen in the previous chapter, the data collection process could influence the resulting time-varying network, which can exhibits several types of issues, such as anomalies. Another issue that might arise, despite the

effort made in data collection, consists in the presence of missing entries, i.e. missing contacts or missing node activities [158].

The lack of information can arise for several reasons: lack of participation during surveys, incomplete records (diary-based, device-based), and technical issues occurring during the data collection process [159]. The presence of partial information can affect the properties of temporal networks, by impacting on the structure of the network itself [160]. This modification in the network structure negatively reflects on the evolution of dynamical processes, that differ from the original evolution and inevitably lead to inaccurate or misleading results.

A great amount of work was carried out to cope with missing data and several methods were developed for the recovery of missing entries [161]. The most common methods are the analysis of the network by ignoring the missing links or the replacement of the missing entries by some plausible value, e.g., the mean. Other methods include: distributional models, which estimate the likelihood of the presence of a link on the basis of the observed links and nodes attributes [162]; hierarchical structure methods [163], stochastic block models [164], and expectation maximization methods [165], that try to extract the connectivity patterns in the available part of the network to infer and complete the unknown part. More recently re-sampling procedures were used to better estimate the outcome of dynamical processes over temporal networks [166]. However, these methods are focused on the recovery of the singular links in networks, or require some external knowledge about the network properties, such as the communities structure, inter-event time distribution or contact duration.

Here, we propose an approach based on the NTF that allows to extract temporal and topological patterns from the network, as shown in Chapter 4. Our approach does not rely on the use of a-priori knowledge such as metadata, and allows to both detect fundamental properties of the studied temporal network and to use the presence of correlated activity patterns of links in the time-varying network to recover information at the mesoscale level in the network.

We adopt the NTF to approximate the time-varying network and we then reconstruct the approximated network with the aim of recovering a similar version of the original network. This network version allows to predict the

evolution and characteristics of dynamical processes occurring over the network, which we have seen to be particularly important in fields as health care, where the process to be predicted corresponds to a disease spreading and the related outcome can help to design ad hoc control strategies [167].

In the following sections we provide a methodology that is twofold: the approximation of a time varying network, to recover missing information; and the network reconstruction to be able of reproducing dynamical processes that occur over the network. We will analyse the method performance in recovering the activity of nodes having partial information as well as how it achieves to reproduce the dynamics of an epidemic modelled by an SIR process.

5.1.1 Network approximation

To approximate a time-varying network affected by missing entries, we adopt a version of the NTF, as in [168], which takes into account the presence of missing values in the tensor to be factorized. Given a tensor $\boldsymbol{\mathcal{X}}$, representing a time-varying network (as in Sec. 4.1.1) with some missing entries, the NTF framework shown in Eq. (4.1) is modified by including a binary tensor mask $\boldsymbol{\mathcal{W}}$ of the same size of $\boldsymbol{\mathcal{X}}$ whose entries are defined as

$$w_{ijk} = \begin{cases} 1 \text{ if } x_{ijk} \text{ is known}, \\ 0 \text{ otherwise}. \end{cases}$$

To build the tensor mask $\boldsymbol{\mathcal{W}}$, we assume to be aware of the nodes that have missing information in the network. More precisely, we assume that all the information relative to the nodes that are not missing and all the non-zero entries related to a missing node are meaningful. We apply the adapted version of the NTF by minimizing an optimization function of the form:

$$f_w(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \| \boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{W}} - \boldsymbol{\mathcal{W}} * [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|_F^2 , \tag{5.1}$$

To this aim, we generate a first set of factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, we factorize a tensor $\tilde{\boldsymbol{\mathcal{X}}}$ whose values are given by the combination of values of $\boldsymbol{\mathcal{X}}$ and of $[\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. In particular, the tensor $\tilde{\boldsymbol{\mathcal{X}}}$ has the same size and the same values of $\boldsymbol{\mathcal{X}}$ in the positions that are known, and unknown values are replaced by

the approximation in $[\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. Thus, the tensor that we approximate is updated at each iteration by:

$$\tilde{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{W}} + (1 - \boldsymbol{\mathcal{W}}) [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] . \tag{5.2}$$

Joint Non-negative Tensor Factorization

When studying time-varying networks, we might have access to some external information, characterizing the nodes or links of the network. Thus, instead of just using the correlated activity patterns in a time-varying network to recover its missing values, an alternative is to use the external information. This is possible if the new data source is correlated in some dimension with the time-varying network studied, to reconstruct the missing values.

To integrate and analyse multiple data sources at a time, each represented as a tensor, we can use a generalized NTF framework that allows to decompose multiple tensor at once. This technique is called Joint Non-negative Tensor Factorization (JNTF) [169], and allows to study the correlated activity patterns in multiple dimension by coupling different tensors.

Formally, given S different data sources, each represented as a tensor $\boldsymbol{\mathcal{X}}_s$ with $s = 1, \ldots, S$, we can adapt the optimization problem in Eq. (4.1) as:

$$\begin{aligned} \min & \frac{1}{2} \sum_{s=1}^{S} \| \boldsymbol{\mathcal{X}}_s - [\![\boldsymbol{\lambda}_s; \mathbf{A}_s, \mathbf{B}_s, \mathbf{C}_s]\!] \|_F^2 + \frac{1}{2} \sum_{s=1}^{S} \alpha_s \| \boldsymbol{\lambda}_s \|_2 \\ & \text{s.t. } \boldsymbol{\lambda}_s, \mathbf{A}_s, \mathbf{B}_s, \mathbf{C}_s \geq 0 . \end{aligned} \tag{5.3}$$

This generalization of the NTF problem allows to integrate data from several sources by approximating together the tensors $\boldsymbol{\mathcal{X}}_s$. Moreover, we can envisage different couplings between tensors by imposing that some of the factor matrices $\mathbf{A}_s, \mathbf{B}_s, \mathbf{C}_s$ will be common for certain s values, and we can add some regularization terms, which ensure sparsity.

The use of the JNTF can be illustrated by the following case study. Let us consider the time-varying social network of the LSCH dataset, and suppose to have access to the position of the nodes of the network during the same time period (see Chapter 3 for details). We can represent the contact network and the location network, providing the position of the nodes at each time, by the

tensors $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$, whose dimensions will be respectively node-node-time and node-location-time. As a consequence, the tensors $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ are coupled in two dimensions, as they share the nodes and the snapshots in time, thus the Eq. (5.3) becomes:

$$\begin{aligned} \min & \frac{1}{2}\|\boldsymbol{\mathcal{X}}_1 - [\![\boldsymbol{\lambda}_1; \mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1]\!]\|_F^2 + \frac{\alpha_1}{2}\|\boldsymbol{\lambda}_1\|_2 \\ & + \frac{1}{2}\|\boldsymbol{\mathcal{X}}_2 - [\![\boldsymbol{\lambda}_2; \mathbf{A}_2, \mathbf{B}_2, \mathbf{C}_2]\!]\|_F^2 + \frac{\alpha_2}{2}\|\boldsymbol{\lambda}_2\|_2 \\ \text{s.t. } & \boldsymbol{\lambda}_{1,2}, \mathbf{A}_{1,2}, \mathbf{B}_{1,2}, \mathbf{C}_{1,2} \geq 0\,, \end{aligned} \tag{5.4}$$

where $\mathbf{A}_1 = \mathbf{A}_2 = \mathbf{A}$ and $\mathbf{C}_1 = \mathbf{C}_2 = \mathbf{C}$. Here, the two matrices $\mathbf{A}$ and $\mathbf{B}_1$ provide the membership of the nodes to the components, the matrix $\mathbf{C}$ gives their temporal activity and the matrix $\mathbf{B}_2$ represents the mapping between the components and the locations. It is worth noting that in this case we can have different $\boldsymbol{\lambda}$ vectors. Thus, this framework has the advantage of being less constrained than solving the NTF on a tensor having 4 dimensions (node-node-time-location).

The adaptation to the missing data problem in the JNTF is now straightforward, as we can substitute to the term ,corresponding to the tensor affected by the missing information, the function in Eq. (5.1). Following the aforementioned example, let us now suppose that some elements in $\boldsymbol{\mathcal{X}}_1$ are missing. We want to recover the missing entries by using the external information provided in $\boldsymbol{\mathcal{X}}_2$, and consequently we need to solve the following problem:

$$\begin{aligned} \min & \frac{1}{2}\|\boldsymbol{\mathcal{X}}_1 * \boldsymbol{\mathcal{W}} - (1 - \boldsymbol{\mathcal{W}}) * [\![\boldsymbol{\lambda}_1; \mathbf{A}, \mathbf{B}_1, \mathbf{C}]\!]\|_F^2 + \frac{\alpha_1}{2}\|\boldsymbol{\lambda}_1\|_2 \\ & + \frac{1}{2}\|\boldsymbol{\mathcal{X}}_2 - [\![\boldsymbol{\lambda}_2; \mathbf{A}, \mathbf{B}_2, \mathbf{C}]\!]\|_F^2 + \frac{\alpha_2}{2}\|\boldsymbol{\lambda}_2\|_2 \\ \text{s.t. } & \boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C} \geq 0\,. \end{aligned}$$

To solve this minimization problem, we modified both the NCG algorithm and the KKT conditions in the ANLS method. We provide the details of the gradient computation as well as the modification of the conditions for the ANLS method in the following sections. By solving the problem, we can use the results of the raw approximation to see what characteristics of the original network we can recover.

NCG adaptation

The computation with the NCG for both the Joint factorization and the case of incomplete data was provided by Acar et al. [169], whose results are found by using the Poblano Toolbox [170]. However, the provided package does not include the computation of the joint factorization with the non-negativity constraints. Here, we provide the computation of the gradient, which is necessary to solve the JNTF problem. To this aim, we include the non-negativity constraints through the Hadamard product of the factor matrices, as shown in Section 2.4.6.

Let us consider the cost function for the JNTF in Eq. (5.4), where the tensors $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ are coupled in the first and third dimension. To incorporate the non-negativity constraints we use the Hadamard product of the factors, such that the cost function, rewritten in matricized form, becomes

$$
\begin{aligned}
h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) &= f_{JNTF}\left(\boldsymbol{\lambda}_{1,2} * \boldsymbol{\lambda}_{1,2}, \mathbf{A} * \mathbf{A}, \mathbf{B}_{1,2} * \mathbf{B}_{1,2}, \mathbf{C} * \mathbf{C}\right) \\
&= \frac{1}{2}\left\|\mathbf{X}_{1,(1)} - \left(\mathbf{A} * \mathbf{A}\right)\left(\boldsymbol{\Lambda}_1 * \boldsymbol{\Lambda}_1\right)\left(\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{B}_1 * \mathbf{B}_1\right)\right)^T\right\|_F^2 + \\
&+ \frac{1}{2}\left\|\mathbf{X}_{2,(1)} - \left(\mathbf{A} * \mathbf{A}\right)\left(\boldsymbol{\Lambda}_2 * \boldsymbol{\Lambda}_2\right)\left(\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{B}_2 * \mathbf{B}_2\right)\right)^T\right\|_F^2 = \\
&= \frac{1}{2}\left\|\delta_{1,(1)}\right\|_F^2 + \frac{1}{2}\left\|\delta_{2,(1)}\right\|_F^2 \qquad (5.5) \\
&= \frac{1}{2}\left\|\mathbf{X}_{1,(2)} - \left(\mathbf{B}_1 * \mathbf{B}_1\right)\left(\boldsymbol{\Lambda}_1 * \boldsymbol{\Lambda}_1\right)\left(\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right)^T\right\|_F^2 + \\
&+ \frac{1}{2}\left\|\mathbf{X}_{2,(2)} - \left(\mathbf{B}_2 * \mathbf{B}_2\right)\left(\boldsymbol{\Lambda}_1 * \boldsymbol{\Lambda}_1\right)\left(\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right)^T\right\|_F^2 = \\
&= \frac{1}{2}\left\|\delta_{1,(2)}\right\|_F^2 + \frac{1}{2}\left\|\delta_{2,(2)}\right\|_F^2 \qquad (5.6) \\
&= \frac{1}{2}\left\|\mathbf{X}_{1,(3)} - \left(\mathbf{C} * \mathbf{C}\right)\left(\boldsymbol{\Lambda}_1 * \boldsymbol{\Lambda}_1\right)\left(\left(\mathbf{B}_1 * \mathbf{B}_1\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right)^T\right\|_F^2 + \\
&+ \frac{1}{2}\left\|\mathbf{X}_{2,(3)} - \left(\mathbf{C} * \mathbf{C}\right)\left(\boldsymbol{\Lambda}_2 * \boldsymbol{\Lambda}_2\right)\left(\left(\mathbf{B}_2 * \mathbf{B}_2\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right)^T\right\|_F^2 = \\
&= \frac{1}{2}\left\|\delta_{1,(3)}\right\|_F^2 + \frac{1}{2}\left\|\delta_{2,(3)}\right\|_F^2 \ . \qquad (5.7)
\end{aligned}
$$

To find a solution for the minimization of this cost function, we need to derive the differential $dh\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)$ with respect to all variables $\boldsymbol{\lambda}_{1,2}$, $\mathbf{A}$,$\mathbf{B}_{1,2}$,

$\mathbf{C}$, which is given by:

$$
\begin{aligned}
&dh\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) = \\
&= \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \mathbf{A}}, d\mathbf{A}\rangle + \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \mathbf{B}_1}, d\mathbf{B}_1\rangle + \\
&+ \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \mathbf{B}_2}, d\mathbf{B}_2\rangle + \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \mathbf{C}}, d\mathbf{C}\rangle + \\
&+ \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \boldsymbol{\lambda}_1}, d\boldsymbol{\lambda}_1\rangle + \frac{1}{2}\langle \frac{\partial h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right)}{\partial \boldsymbol{\lambda}_2}, d\boldsymbol{\lambda}_2\rangle \ .
\end{aligned}
$$

First, the derivatives with respect to the factors $\mathbf{B}_1$ and $\mathbf{B}_2$ can be found by following the steps in Section 2.4.6. Thus,

$$
\begin{aligned}
\nabla_{\mathbf{B}_1} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) &= \frac{\partial h}{\partial \mathbf{B}_1} = 2\mathbf{B}_1 * \left(\left(-\delta_{1,(2)}\right)\left[\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right]\right) \ , \\
\nabla_{\mathbf{B}_2} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) &= \frac{\partial h}{\partial \mathbf{B}_2} = 2\mathbf{B}_2 * \left(\left(-\delta_{2,(2)}\right)\left[\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right]\right) \ .
\end{aligned}
$$

Second, the derivatives with respect to the factors $\mathbf{A}$ and $\mathbf{C}$ are given by the sum of the derivatives of terms in Eq. (5.5) and Eq. (5.7) respectively. Thus

$$
\begin{aligned}
\nabla_{\mathbf{A}} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) = \frac{\partial h}{\partial \mathbf{A}} &= 2\mathbf{A} * \left(\left(-\delta_{1,(1)}\right)\left[\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{B}_1 * \mathbf{B}_1\right)\right]\right) + \\
&+ 2\mathbf{A} * \left(\left(-\delta_{2,(1)}\right)\left[\left(\mathbf{C} * \mathbf{C}\right) \odot \left(\mathbf{B}_2 * \mathbf{B}_2\right)\right]\right) \\
\nabla_{\mathbf{C}} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) = \frac{\partial h}{\partial \mathbf{C}} &= 2\mathbf{C} * \left(\left(-\delta_{1,(3)}\right)\left[\left(\mathbf{B}_1 * \mathbf{B}_1\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right]\right) + \\
&+ 2\mathbf{C} * \left(\left(-\delta_{2,(3)}\right)\left[\left(\mathbf{B}_2 * \mathbf{B}_2\right) \odot \left(\mathbf{A} * \mathbf{A}\right)\right]\right) \ .
\end{aligned}
$$

Finally, we have to compute the derivatives with respect to $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$. Let us consider f_{JNTF} and rewrite it as follows:

$$
\begin{aligned}
f_{JNTF}\left(\boldsymbol{\lambda}_{1,2} * \boldsymbol{\lambda}_{1,2}, \mathbf{A} * \mathbf{A}, \mathbf{B}_{1,2} * \mathbf{B}_{1,2}, \mathbf{C} * \mathbf{C}\right) = \\
= \frac{1}{2} \underbrace{\|\boldsymbol{\mathcal{X}}_1\|_F^2}_{f_1} + \frac{1}{2} \underbrace{\|[\![\boldsymbol{\lambda}_1 * \boldsymbol{\lambda}_1; \mathbf{A} * \mathbf{A}, \mathbf{B}_1 * \mathbf{B}_1, \mathbf{C} * \mathbf{C}]\!]\|_F^2}_{f_2} \\
- \underbrace{\langle \boldsymbol{\mathcal{X}}_1, [\![\boldsymbol{\lambda}_1 * \boldsymbol{\lambda}_1; \mathbf{A} * \mathbf{A}, \mathbf{B}_1 * \mathbf{B}_1, \mathbf{C} * \mathbf{C}]\!] \rangle}_{f_3} + \underbrace{\frac{\alpha_1}{2} \sqrt{\left(\boldsymbol{\lambda}_1 * \boldsymbol{\lambda}_1\right)^2 + \epsilon}}_{f_4} \\
+ \frac{1}{2} \underbrace{\|\boldsymbol{\mathcal{X}}_2\|_F^2}_{f_5} + \frac{1}{2} \underbrace{\|[\![\boldsymbol{\lambda}_2 * \boldsymbol{\lambda}_2; \mathbf{A} * \mathbf{A}, \mathbf{B}_2 * \mathbf{B}_2, \mathbf{C} * \mathbf{C}]\!]\|_F^2}_{f_6} \\
- \underbrace{\langle \boldsymbol{\mathcal{X}}_2, [\![\boldsymbol{\lambda}_2 * \boldsymbol{\lambda}_2; \mathbf{A} * \mathbf{A}, \mathbf{B}_2 * \mathbf{B}_2, \mathbf{C} * \mathbf{C}]\!] \rangle}_{f_7} + \underbrace{\frac{\alpha_2}{2} \sqrt{\left(\boldsymbol{\lambda}_2 * \boldsymbol{\lambda}_2\right)^2 + \epsilon}}_{f_8}.
\end{aligned}
$$

We compute the element-wise derivatives for all the addends in the equation above, with respect to $\lambda_{1,r'}$ (similarly for $\lambda_{2,r'}$). The terms f_1, f_5, f_6, f_7, and f_8 do not depend on $\boldsymbol{\lambda}_1$, thus:

$$
\frac{\partial f_1}{\partial \lambda_{1,r'}} = \frac{\partial f_5}{\partial \lambda_{1,r'}} = \frac{\partial f_6}{\partial \lambda_{1,r'}} = \frac{\partial f_7}{\partial \lambda_{1,r'}} = \frac{\partial f_8}{\partial \lambda_{1,r'}} = \mathbf{0} \quad \forall r' \,.
$$

The remaining derivatives can be computed as follows:

$$
\begin{aligned}
\frac{\partial f_2}{\partial \lambda_{1,r'}} &= \frac{\partial}{\partial \lambda_{1,r'}} \left[\langle \sum_{r=1}^{R} \lambda_{1,r}^2 \mathbf{a}_r^2 \circ \mathbf{b}_{1,r}^2 \circ \mathbf{c}_r^2, \sum_{r=1}^{R} \lambda_{1,r}^2 \mathbf{a}_r^2 \circ \mathbf{b}_{1,r}^2 \circ \mathbf{c}_r^2 \rangle \right] \\
&= \frac{\partial}{\partial \lambda_{1,r'}} \left[\sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \left(\sum_{r=1}^{R} \lambda_{1,r}^2 a_{ir}^2 b_{1,jr}^2 c_{kr}^2 \right)^2 \right] \\
&= 4 \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \left(\sum_{r=1}^{R} \lambda_{1,r}^2 a_{ir}^2 b_{1,jr}^2 c_{kr}^2 \right) \cdot \lambda_{1,r'} a_{ir'}^2 b_{1,jr'}^2 c_{kr'}^2 \\
&= 4 \lambda_{1,r'} [\![\boldsymbol{\lambda}_1 * \boldsymbol{\lambda}_1; \mathbf{A} * \mathbf{A}, \mathbf{B}_1 * \mathbf{B}_1, \mathbf{C} * \mathbf{C}]\!] \times_1 \mathbf{a}_{r'}^2 \times_2 \mathbf{b}_{1,r'}^2 \times_3 \mathbf{c}_{r'}^2 \,;
\end{aligned}
$$

$$\begin{aligned}
\frac{\partial f_3}{\partial \lambda_{1,r'}} &= \frac{\partial}{\partial \lambda_{1,r'}}\left[\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(\sum_{r=1}^{R} x_{1,ijk}\lambda_{1,r}^2 a_{ir}^2 b_{1,jr}^2 c_{kr}^2\right)\right] \\
&= 2\lambda_{1,r'}\left(\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{1,ijk} a_{ir'}^2 b_{1,jr'}^2 c_{kr'}^2\right) \\
&= 2\lambda_{1,r'}\boldsymbol{\mathcal{X}}_1 \times_1 \mathbf{a}_{r'}^2 \times_2 \mathbf{b}_{1,r'}^2 \times_3 \mathbf{c}_{r'}^2 \; ; \\
\frac{\partial f_4}{\partial \lambda_{1,r'}} &= \alpha_1 \frac{\lambda_{1,r'}^3}{\sqrt{\left(\lambda_{1,r'}^2\right)^2 + \epsilon}} \; .
\end{aligned}$$

The resulting derivatives for $\boldsymbol{\lambda}_1$ and analogously for $\boldsymbol{\lambda}_2$ are then

$$\begin{aligned}
&\nabla_{\boldsymbol{\lambda}_{1,r'}} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) = \\
&\quad = 2\boldsymbol{\lambda}_{1,r'} * \left(\left[\!\left[\boldsymbol{\lambda}_1 * \boldsymbol{\lambda}_1; \mathbf{A} * \mathbf{A}, \mathbf{B}_1 * \mathbf{B}_1, \mathbf{C} * \mathbf{C}\right]\!\right] - \boldsymbol{\mathcal{X}}_1\right) \times_1 \mathbf{a}_{r'}^2 \times_2 \mathbf{b}_{1,r'}^2 \times_3 \mathbf{c}_{r'}^2 + \\
&\quad + \alpha_1 \frac{\lambda_{1,r'}^3}{\sqrt{\left(\lambda_{1,r'}^2\right)^2 + \epsilon}} \\
&\nabla_{\boldsymbol{\lambda}_{2,r'}} h\left(\boldsymbol{\lambda}_{1,2}, \mathbf{A}, \mathbf{B}_{1,2}, \mathbf{C}\right) = \\
&\quad = 2\boldsymbol{\lambda}_{2,r'} * \left(\left[\!\left[\boldsymbol{\lambda}_2 * \boldsymbol{\lambda}_2; \mathbf{A} * \mathbf{A}, \mathbf{B}_2 * \mathbf{B}_2, \mathbf{C} * \mathbf{C}\right]\!\right] - \boldsymbol{\mathcal{X}}_2\right) \times_1 \mathbf{a}_{r'}^2 \times_2 \mathbf{b}_{2,r'}^2 \times_3 \mathbf{c}_{r'}^2 + \\
&\quad + \alpha_2 \frac{\lambda_{2,r'}^3}{\sqrt{\left(\lambda_{2,r'}^2\right)^2 + \epsilon}} \; .
\end{aligned}$$

In the following section, we provide the adaptation of the KKT conditions for the computation of the JNTF. This adaptation can be used as an alternative to the functions provided in the Poblano Toolbox, to solve JNTF problems through the ANLS and BPP algorithms.

KKT adaptation

In this section, we show how to adapt the ANLS framework and in particular how to compute the new KKT optimality conditions to perform the joint factorization with non-negativity constraints. To this aim, we have to solve the minimization problem in Eq. (5.3). Here, we take as regularization terms the square of the l_2-norm, which, as we will show below, will be useful to adapt the KKT conditions. We show the procedure to solve Eq.(5.4), in which the two data sources are coupled in two dimensions, i.e. $S = 2$, $\mathbf{A} = \mathbf{A}_1 = \mathbf{A}_2$, and

$\mathbf{C} = \mathbf{C}_1 = \mathbf{C}_2$. The solution to this problem will be useful to find a solution for the application described in Section 5.1.3.

To solve the problem through the BPP algorithm we have to rewrite the minimization problem in the form of Eq. 2.22, and solve it for each of the factor matrices (here we include $\boldsymbol{\lambda}_{1,2}$ in the factor matrices). To this aim, we start by solving the problem for the factor matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ which are respectively present in the first and third term of Eq. 5.4. With respect to these factor matrices we can rewrite the equation by using the 2-mode matricization of $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$, which leads to the following approximations:

$$\mathbf{X}_{1,(2)} \approx \mathbf{B}_1 \boldsymbol{\Lambda}_1 \left(\mathbf{C} \odot \mathbf{A}\right)^T \text{ and } \mathbf{X}_{2,(2)} \approx \mathbf{B}_2 \boldsymbol{\Lambda}_2 \left(\mathbf{C} \odot \mathbf{A}\right)^T ,$$

where $\boldsymbol{\Lambda}_1 = \operatorname{diag}\left(\lambda_{1,1}, \ldots, \lambda_{1,R}\right)$ and $\boldsymbol{\Lambda}_2 = \operatorname{diag}\left(\lambda_{2,1}, \ldots, \lambda_{2,R}\right)$. We can rewrite the approximations as

$$\mathbf{X}_{1,(2)}^T \approx \left(\mathbf{C} \odot \mathbf{A}\right) \boldsymbol{\Lambda}_1 \mathbf{B}_1^T \text{ and } \mathbf{X}_{2,(2)}^T \approx \left(\mathbf{C} \odot \mathbf{A}\right) \boldsymbol{\Lambda}_2 \mathbf{B}_2^T ,$$

where $\boldsymbol{\Lambda}_{1,2}^T = \boldsymbol{\Lambda}_{1,2}$, and thus

$$\mathbf{B}_1^T \approx \boldsymbol{\Lambda}_1^{-1} \left(\mathbf{C} \odot \mathbf{A}\right)^\dagger \mathbf{X}_{1,(2)}^T \text{ and } \mathbf{B}_2^T \approx \boldsymbol{\Lambda}_2^{-1} \left(\mathbf{C} \odot \mathbf{A}\right)^\dagger \mathbf{X}_{2,(2)}^T .$$

By using the third property of the Khatri-Rao product in Prop. 2, we can write the approximation as

$$\begin{aligned} \boldsymbol{\Lambda}_1 \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \mathbf{B}_1^T &\approx \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{1,(2)}^T , \\ \boldsymbol{\Lambda}_2 \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \mathbf{B}_2^T &\approx \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{2,(2)}^T . \end{aligned}$$

The related subproblems of Eq.(5.4) for $\mathbf{B}_1$ and $\mathbf{B}_2$ are then reduced to:

$$\begin{aligned} \min_{\mathbf{B}_1} \frac{1}{2} \left\| \boldsymbol{\Lambda}_1 \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \mathbf{B}_1^T - \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{1,(2)}^T \right\|_F^2 , \\ \min_{\mathbf{B}_2} \frac{1}{2} \left\| \boldsymbol{\Lambda}_2 \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \mathbf{B}_2^T - \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{2,(2)}^T \right\|_F^2 . \end{aligned}$$

Finally, the subproblems can be respectively written in the form of Eq. 2.22 by assigning

$$\mathbf{V} = \mathbf{\Lambda}_1 \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \ , \ \ \mathbf{X} = \mathbf{B}_1^T \ \text{ and } \ \mathbf{W} = \mathbf{\Lambda}_2 \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{1,(2)}^T \ ,$$
$$\mathbf{V} = \left(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A}\right) \ , \ \ \mathbf{X} = \mathbf{B}_2^T \ \text{ and } \ \mathbf{W} = \left(\mathbf{C} \odot \mathbf{A}\right)^T \mathbf{X}_{2,(2)}^T \ .$$

The solution to the subproblems is now straightforward as illustrated in Section 2.4.5.

We now need to rewrite the subproblems for the factors $\mathbf{A}$ and $\mathbf{C}$ which are present in both the first and third term of Eq. (5.4). We show the procedure for $\mathbf{A}$ (which is analogous for $\mathbf{C}$). First, we write the minimization problem by using the 1-mode matricization of $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ (3-mode for $\mathbf{C}$):

$$\min_{\mathbf{A}} \frac{1}{2} \left\| \left(\mathbf{C} \odot \mathbf{B}_1\right) \mathbf{\Lambda}_1 \mathbf{A}^T - \mathbf{X}_{1,(1)}^T \right\|_F^2 + \frac{1}{2} \left\| \left(\mathbf{C} \odot \mathbf{B}_2\right) \mathbf{\Lambda}_2 \mathbf{A}^T - \mathbf{X}_{2,(1)}^T \right\|_F^2 \ .$$

By following the procedure shown above, we can write the problem in the following way:

$$\begin{aligned} \min_{\mathbf{A}} \frac{1}{2} \left\| \mathbf{\Lambda}_1 \left(\mathbf{C}^T\mathbf{C} * \mathbf{B}_1^T\mathbf{B}_1\right) \mathbf{A}^T - \left(\mathbf{C} \odot \mathbf{B}_1\right)^T \mathbf{X}_{1,(1)}^T \right\|_F^2 + \\ + \frac{1}{2} \left\| \mathbf{\Lambda}_2 \left(\mathbf{C}^T\mathbf{C} * \mathbf{B}_2^T\mathbf{B}_2\right) \mathbf{A}^T - \left(\mathbf{C} \odot \mathbf{B}_2\right)^T \mathbf{X}_{2,(1)}^T \right\|_F^2 \ , \end{aligned}$$

in which we can assign

$$\mathbf{V}_1 = \mathbf{\Lambda}_1\mathbf{C}^T\mathbf{C} + \mathbf{B}_1^T\mathbf{B}_1 \ , \ \ \mathbf{X} = \mathbf{A}^T \ \text{ and } \ \mathbf{W}_1 = \left(\mathbf{C} \odot \mathbf{B}_1\right)^T \mathbf{X}_{1,(1)}^T \ ,$$
$$\mathbf{V}_2 = \mathbf{\Lambda}_2\mathbf{C}^T\mathbf{C} + \mathbf{B}_2^T\mathbf{B}_2 \ \text{ and } \ \mathbf{W}_2 = \left(\mathbf{C} \odot \mathbf{B}_2\right)^T \mathbf{X}_{2,(1)}^T \ ,$$

leading to

$$f\left(\mathbf{X}\right) = \min_{\mathbf{X}} \frac{1}{2} \left\| \mathbf{V}_1\mathbf{X} - \mathbf{W}_1 \right\|_F^2 + \frac{1}{2} \left\| \mathbf{V}_2\mathbf{X} - \mathbf{W}_2 \right\|_F^2 \ . \tag{5.8}$$

To solve the problem in Eq. (5.8) we need to adapt the KKT conditions, which result to be

$$\nabla f\left(\mathbf{X}\right) = \underbrace{\left(\mathbf{V}_1^T\mathbf{V}_1 + \mathbf{V}_2^T\mathbf{V}_2\right)}_{\mathbf{V}_{1,2}} \mathbf{X} - \underbrace{\left(\mathbf{V}_1^T\mathbf{W}_1 + \mathbf{V}_2^T\mathbf{W}_2\right)}_{\mathbf{W}_{1,2}}$$
$$\nabla f\left(\mathbf{X}\right) \geq 0 \ , \ \nabla f\left(\mathbf{X}\right)^T \mathbf{X} = 0 \ , \ \mathbf{X} \geq 0 \ ,$$

whose solution is given by solving

$$\mathbf{X}^T\mathbf{V}_{1,2}^T - \mathbf{W}_{1,2}^T = 0\ .$$

Since in Eq. (5.4) are present the regularization terms, we have to adapt the KKT conditions for the minimization problem with respect to $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$. We show the procedure for $\boldsymbol{\lambda}_1$, which is analogous for $\boldsymbol{\lambda}_2$. We consider the cost function built from the terms in which $\boldsymbol{\lambda}_1$ is involved (the first and second in Eq.(5.4)):

$$f_{\boldsymbol{\lambda}_1} = \frac{1}{2}\left\|\boldsymbol{\mathcal{X}}_1 - [\![\boldsymbol{\lambda}_1; \mathbf{A}, \mathbf{B}_1, \mathbf{C}]\!]\right\|_F^2 + \frac{\alpha_1}{2}\left\|\boldsymbol{\lambda}_1\right\|_2^2\ ,$$

and we rewrite it through the vectorization:

$$f_{\boldsymbol{\lambda}_1} = \frac{1}{2}\left\|\mathrm{vec}\left(\mathbf{C}\odot\mathbf{B}\odot\mathbf{A}\right)\boldsymbol{\lambda}_1 - \mathrm{vec}\left(\boldsymbol{\mathcal{X}}\right)\right\|_F^2 + \frac{\alpha_1}{2}\left\|\boldsymbol{\lambda}_1\right\|_2^2\ .$$

Minimizing the cost function $f_{\boldsymbol{\lambda}_1}$ is equivalent to minimize $f'_{\boldsymbol{\lambda}_1}$, obtained by incorporating the regularization term as follows:

$$f'_{\boldsymbol{\lambda}_1} = \frac{1}{2}\left\|\underbrace{\begin{pmatrix}\mathrm{vec}\left(\mathbf{C}\odot\mathbf{B}_1\odot\mathbf{A}\right)\\ \sqrt{\alpha_1}\end{pmatrix}}_{\mathbf{V}}\underbrace{\boldsymbol{\lambda}_1}_{\mathbf{x}} - \underbrace{\begin{pmatrix}\mathrm{vec}\left(\boldsymbol{\mathcal{X}}\right)\\ 0\end{pmatrix}}_{\mathbf{w}}\right\|_F^2\ .$$

The solution to the minimization of $f'_{\boldsymbol{\lambda}_1}$ follows from Eq. (2.19), as shown in Section 2.4.5.

5.1.2 Network reconstruction

Once the network is approximated by using the NTF or the JNTF, we use the resulting factor matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ as a guide to rebuild the network, by starting from the original information contained in $\boldsymbol{\mathcal{X}}$. This is done with the aim of both preserving the heterogeneity in the nodes contacts and extracting general behaviours characterizing the network. This is particularly important in the perspective of using the recovered information to study a dynamic process occurring over the network. In particular we suppose to know which of the nodes have only partial activity and we discard the information about the times that were erased. This is based on the observation that while collecting data

we are usually aware of the nodes that displayed some issues (e.g., people that did not compiled surveys, sensors that did not work correctly) and not about the time in which these issues occur.

To reconstruct the time-varying network from the components we rely on the membership of links (involving a node with partial information) to the components and infer each link activity by looking at the activation time of its temporal component. As shown in Section 4.2.2, the membership of the links can be recovered by computing the product $a_{ir}b_{jr}$ for each link (i, j) in the component r, and by accumulating the sum of their square values in decreasing order until it exceeds the 95% of the norm $\|\mathbf{a}_r \cdot \mathbf{b}_r^T\|_F^2$.

The next step is to detect the times in which these links are active. For this purpose we adopt the Otsu threshold, already used in Section 4.2.2, in the following way:

1. we apply on each temporal activity $\mathbf{c}_r$ the Otsu method, which provides a threshold τ;
2. we use the threshold τ to transform $\mathbf{c}_r$ in its binary version (having a 1 when the activity is above τ and a 0 otherwise);
3. we use the binary vector to find the activation times which are not present in the original network $\boldsymbol{\mathcal{X}}$.

Once computed both the links membership and their activation we reconstruct the tensor by adding to the original the contacts between the links involving at least a missing node at the detected times.

As we mentioned in the previous chapter, the distribution of contacts of a time-varying network after its approximation via the NTF tends to be more homogeneous than in the original state. In particular, even though the average weight of the components are perfectly kept by the decomposition, the same does not hold for the weight of the links, that tends to be similar in the same component.

Hence, to reconstruct the time-varying network we recalibrate the weight distribution by taking as a guide the distribution in the original state. This procedure is particularly important because we are here interested in recovering

a time-varying network that has the characteristics of the original one that matters most to reproduce the outcome of a dynamical process.

The reconstructed time-varying network is then obtained by the following procedure. First of all we compute the original weight distribution by taking into account only the links, whose activity is fully known. Then we can operate in two distinct ways:

1. we compute the weight of links involving at least a node having partial information and we reassign their weight by the one picked uniformly at random from the distribution;

2. we compute the weight of links involving at least a node having partial information and we reassign their weights by ranking the values, so that the links having higher weight in the partial information available will have higher weight after the assignment and vice-versa.

As the links in the approximation are modulated in time by the temporal activity of their component, their total weight might be often greater than the one to be reassigned. Thus, to reassign the weight we erase part of the activity in the times that we recovered through the Otsu method until the new weight match the one picked from the distribution.

5.1.3 Application

We applied the method, to recover the mesoscale properties of missing data in time-varying networks and the relative procedure to recover the right evolution of dynamical processes, to three different time-varying social networks collected by the SocioPatterns collaboration. The datasets, whose details are described in Chapter 3 are the HT09 and SFHH related to people proximity interactions during two conferences and the LSCH dataset, also described in Chapter 4

To apply the procedure, we simulated on the three datasets a loss of data determined by a percentage $p_{nodes} \in [0.1, 0.2, 0.4]$ of nodes picked uniformly at random, whose a percentage $p_{times} = 0.5$ of their activity was removed consecutively in time. To erase the values we proceeded as follows. We represented each dataset as a tensor $\boldsymbol{\mathcal{X}}$, from which we erase the contacts of the selected nodes for the first half of the time, to create a new tensor $\boldsymbol{\mathcal{M}}$.

We build the tensor mask $\boldsymbol{\mathcal{W}}$, used to solve Eq. (5.1). To this aim, we only assume to know the nodes for which part of the activity is missing. The tensor mask $\boldsymbol{\mathcal{W}}$ is then a tensor of the same size of $\boldsymbol{\mathcal{X}}$ with entries equal to 1 if they are related to a node which is not in the set of those with partial information. Then, for each node having partial information we consider as possibly unknown (i.e. 0) all the zero-entries related to that node if no contacts were present at that time in $\boldsymbol{\mathcal{M}}$.

In the next section we report the results achieved by applying the procedure to recover a reconstructed version of the time-varying networks in the case of the three datasets with partial information, represented as the tensors $\boldsymbol{\mathcal{M}}_{HT09}$, $\boldsymbol{\mathcal{M}}_{SFHH}$, and $\boldsymbol{\mathcal{M}}_{LSCH}$.

Approximation results

As described in Section 5.1.1, the first step is the approximation of the tensors via non-negative tensor factorization techniques. We start our analysis by solving the NTF problem of Eq (5.1), which provides as a result the components needed to build an approximated version of the network.

Table 5.1 **Selected rank and core consistency values**. The table reports the selected ranks R for each of the LSCH, HT09, and SFHH datasets, as well as the core consistency values CC, corresponding to the best realization out of 20 decompositions. These values are displayed for the $10\%, 20\%$ and 40% of missing nodes in the datasets and half the time span.

Dataset	% nodes	% times	R	CC
LSCH	10	50	12	91.7
	20	50	12	92
	40	50	11	94.3
HT09	10	50	9	94.6
	20	50	9	91.3
	40	50	7	89
SFHH	10	50	12	92.5
	20	50	12	91.2
	40	50	11	85

First of all we select the number of components to decompose each tensor. The assessment of the final number of components R is given by the computation of the core consistency between $\boldsymbol{\mathcal{M}}$ and its approximation in the tensor entries that are not related to nodes with partial information. We performed 5 different decompositions for each number of components $r \in [2, \ldots, R_{max}]$. Here, R_{max} varies depending on the dataset. In particular, $R_{max} = R_{final} - 1$, where R_{final} corresponds to the first number of components in which the core consistency at each realization is lower than 85%. We decided to set a high threshold for the core consistency instead of looking at the change of slope as done in Section 4.1.2, to have approximations as faithful as possible to the original network. We highlight that the final rank could vary depending on the number of nodes with activity partially missing taken into account, because the greatest is the percentage of missing nodes the most the correlations in the activity of the links are broken. The final values selected for each datasets and each percentage of nodes are reported in Table 5.1.

Once selected the best rank R, we run for that value 20 decompositions for each tensor and pick the result corresponding to the highest core consistency values. Table 5.1 also reports the core consistency values obtained, which correspond to the best realization for each case.

Table 5.2 **Weight comparison**. Comparison of the total sum of the weights in the original $\boldsymbol{\mathcal{X}}$, missing $\boldsymbol{\mathcal{M}}$, and approximated $\boldsymbol{\mathcal{X}}_{app}$ tensor of each dataset and different percentages of missing values.

	LSCH	**HT09**	**SFHH**
Original	89298	11550	36976
10%-50%			
Missing	80956	10692	32266
Approximated	95619	13330	38297
20%-50%			
Missing	73166	9396	29096
Approximated	94863	13002	36833
40%-50%			
Missing	61572	7940	22048
Approximated	91878	16159	35798

As discussed in the previous sections, the NTF is able to keep in the approximation some of the characteristics of the original time-varying network, while changes others. For this reason we computed the total sum of the weights of the links in the original time-varying network $\boldsymbol{\mathcal{X}}$, the one with the missing values $\boldsymbol{\mathcal{M}}$, and the approximated one $\boldsymbol{\mathcal{X}}_{app}$, summarized in Tab. 5.2.

Consistently to what we have shown in the previous chapter, even in presence of missing values, the NTF manages to keep almost fixed the total sum of the approximated tensor. The little increasing in the total tensor weight is due to the fact that the approximation tends to create more links between nodes which are found to be in the same components. This phenomenon is reinforced in the presence of nodes having partial activity as the information of the contacts between them is missing and thus these nodes tend to be connected to all the nodes in the same components. However, increasing the amount of nodes with partial information decreases the connection in the network and thus erases the correlations between the activities of the nodes.

Table 5.3 **Average weights** computed in each sub-networks corresponding to the links belonging to the NTF components and their activity in time. Here, the original and $20\% - 50\%$ approximated cases are compared. The results show that even with partial information the NTF is able to keep the average weights of the original network.

	LSCH		**HT09**		**SFHH**	
Components	**Original**	**Approximated**	**Original**	**Approximated**	**Original**	**Approximated**
1	17.32	17.34	6.91	8.10	23.54	16.06
2	7.90	8.53	18.18	18.18	10.84	11.48
3	17.07	15.70	23.33	17.56	2.57	2.45
4	11.69	11.75	4.69	5.08	0.44	0.46
5	12.59	12.38	9.53	9.00	5.86	5.60
6	11.86	11.34	2.59	2.40	0.46	0.55
7	13.05	11.84	2.21	2.40	3.07	4.06
8	12.03	11.66	13.00	11.00	1.02	1.19
9	5.44	5.29	3.72	4.69	1.21	1.26
10	7.66	7.65			2.37	2.07
11	4.86	5.01			0.86	0.91
12	2.81	3.09			20.83	9.64

As we discussed in Section 4.1.3, this mechanism makes the approximated network more homogeneous. Thus, to be able of correctly reproducing dynamical processes on top of the network, we need to reinsert the heterogeneity

properties. We will see how the reconstruction step is effective in solving this problem in the next section.

These observations as well as those introduced in Chapter 4 stress the importance of reintroducing the heterogeneity in the number of contacts, as the outcome of a diffusion process might be different from the approximated network provided by the NTF to the original network. However, the NTF is able to keep some of the original network characteristics, even in the presence of missing values. This is the case of the average weight in the sub-networks corresponding to the components. We show as a representative example the average weights in the three datasets corresponding to the 20% of missing nodes and 50% of the timeline in Table 5.3.

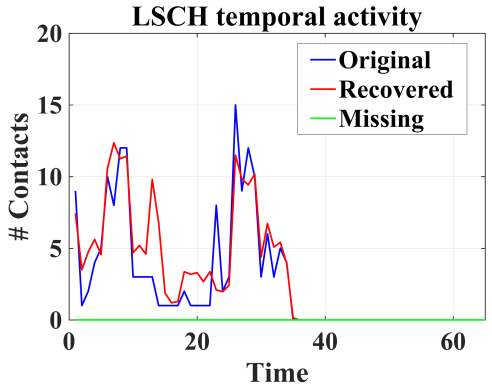
Table 5.4 **Pearson's correlation coefficient** between temporal activities of the nodes, whose activity was partially missing, in the original and in the approximated network. The coefficient are computed by comparing the total number of contacts of each of these nodes on half the temporal line in the original and approximated cases. The table shows the set in which the coefficients vary and the related median value. The results corresponds to the statistically significant values achieved, i.e. having p-value $< 10^{-3}$.

	Pearson's coeff.	**median value**	**p-value**
LSCH			
10%-50%	[0.70, 0.93]	0.85	$< 10^{-3}$
20%-50%	[0.52, 0.92]	0.85	$< 10^{-3}$
40%-50%	[0.54, 0.94]	0.84	$< 10^{-3}$
HT09			
10%-50%	[0.29, 0.81]	0.65	$< 10^{-3}$
20%-50%	[0.23, 0.83]	0.59	$< 10^{-3}$
40%-50%	[0.22, 0.78]	0.52	$< 10^{-3}$
SFHH			
10%-50%	[0.29, 0.90]	0.60	$< 10^{-3}$
20%-50%	[0.31, 0.88]	0.56	$< 10^{-3}$
40%-50%	[0.29, 0.92]	0.58	$< 10^{-3}$

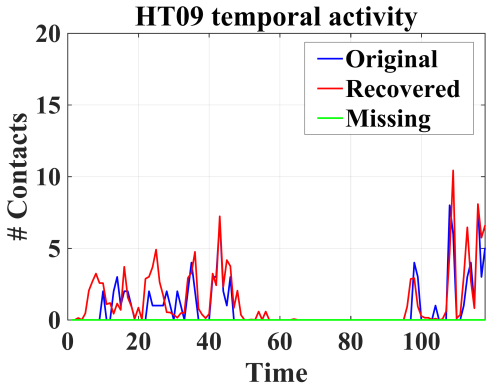
Moreover, when approximating a network in which strong correlated activity patterns are present, the overall activity in time of the nodes with partial information is well approximated. The results achieved in recovering the

total number of contacts with time for nodes, whose activity was partially missing, are shown by Table 5.4 in which we reported the Pearson's coefficients (range, median values) for statistically significant cases (p-values $< 10^{-3}$). As we can observe by the coefficients values the NTF achieved good results in approximating the overall activity of the nodes (i.e. the total number of contacts of a node in time).

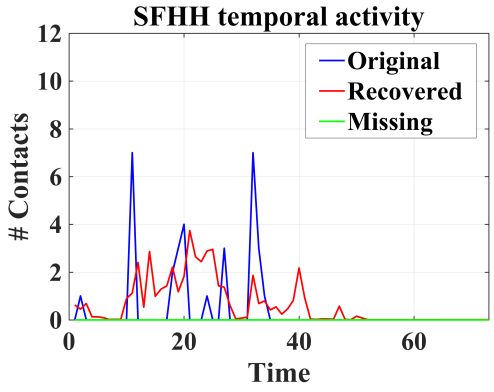
To better understand this result, we displayed in Fig. 5.1 the profiles of one missing node activity for each dataset in the approximation $\boldsymbol{\mathcal{X}}_{app}$ and compare it with the original case $\boldsymbol{\mathcal{X}}$ and the one with partial information $\boldsymbol{\mathcal{M}}$.



(a) Activity of one node with partial information in the LSCH dataset



(b) Activity of one node with partial information in the HT09 dataset



(c) Activity of one node with partial information in the SFHH dataset

Fig. 5.1 **Comparison of one activity of a nodes with partial information in the original, missing and recovered case**. The comparison is shown for one representative node in each dataset. Here, the total amount of contacts in the original case is displayed in blue, the missing in green, and recovered by the NTF in red.

Reconstruction results

Once obtained the best approximation with NTF, we can recover some mesoscale properties for the nodes whose part of the activity was missing by studying both the topological and temporal patterns. In particular, we find the links involving nodes with partial activity and their activation times as explained in Section 5.1.2. To obtain the reconstructed network $\boldsymbol{\mathcal{X}}_{rec}$ we inserted heterogeneity characteristics in the network by reassigning the weight for each of the added link. This operation yield a time-varying network in which the missing information is recovered and its heterogeneity properties are kept.

As a result we reconstructed the time-varying networks of the LSCH, HT09, and SFHH datasets. With the aim of evaluating how much our procedure enables to recover the outcome of dynamical processes, we simulated an epidemic spreading over the three different datasets. Then, we compared the results of the simulations in the original network $\boldsymbol{\mathcal{X}}$, in the one with the partial information $\boldsymbol{\mathcal{M}}$ and in the recovered one $\boldsymbol{\mathcal{X}}_{rec}$.

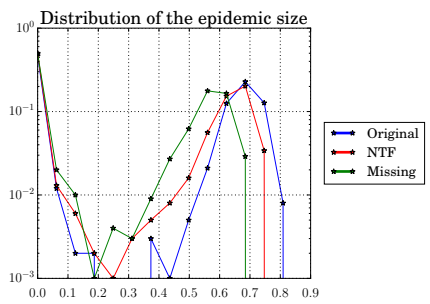
For this purpose, we computed an SIR process with different probabilities of infection λ and recovery μ. In particular we computed the process for all the couples (λ, μ) with $\lambda, \mu = 10^{x-1}$, where we took 20 values of $x \in [-2, 1]$. Then, we run several realizations (here 1000) for each couple of probabilities. Each realization starts from a node (the root) which we take among the nodes having all the information. We selected for the study the simulations that satisfy the following criterion:

- the disease spreading has to be finished in the time span of the dataset ;
- the number of infected/recovered individuals has to be greater than the 20% and lower than the 80% of the entire population.

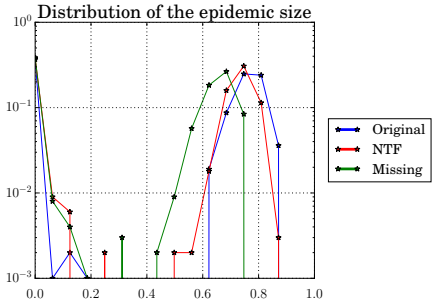
These requirements allow to take into account in the study only those simulations that really started and finished in the entire duration.

To compare the different results obtained by using $\boldsymbol{\mathcal{X}}$, $\boldsymbol{\mathcal{M}}$, and $\boldsymbol{\mathcal{X}}_{rec}$ we computed the distribution of the epidemic size, in the original, missing, and recovered cases. We reported the results for the LSCH dataset in Fig. 5.2. For each percentage of p_{nodes} we displayed two representative couples of the selected probabilities. In particular, we selected $\lambda = 10^{-0.8}, \mu = 10^{-0.6}$ and

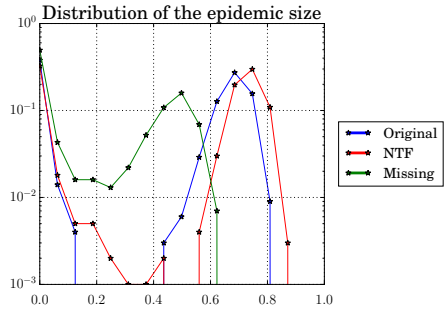
$\lambda = 10^{-0.6}, \mu = 10^{-0.6}$ with the 10% and 20% of nodes, while as the number of missing nodes reaches the 40% we selected the probabilities $\lambda = 10^{-1.4}, \mu = 10^{-1.3}$ and $\lambda = 10^{-1.3}, \mu = 10^{-0.95}$.



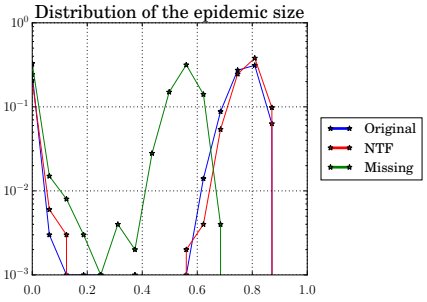
(a) LSCH 10% – 50%



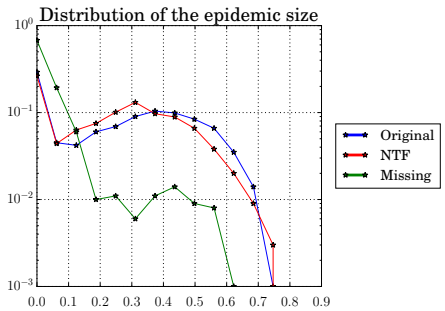
(b) LSCH 10% – 50%



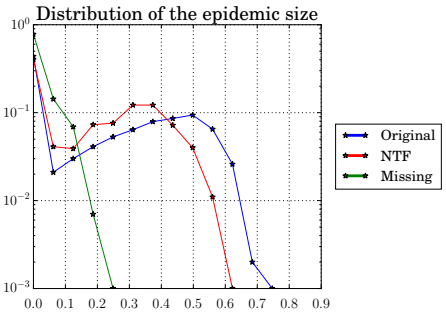
(c) LSCH 20% – 50%



(d) LSCH 20% – 50%



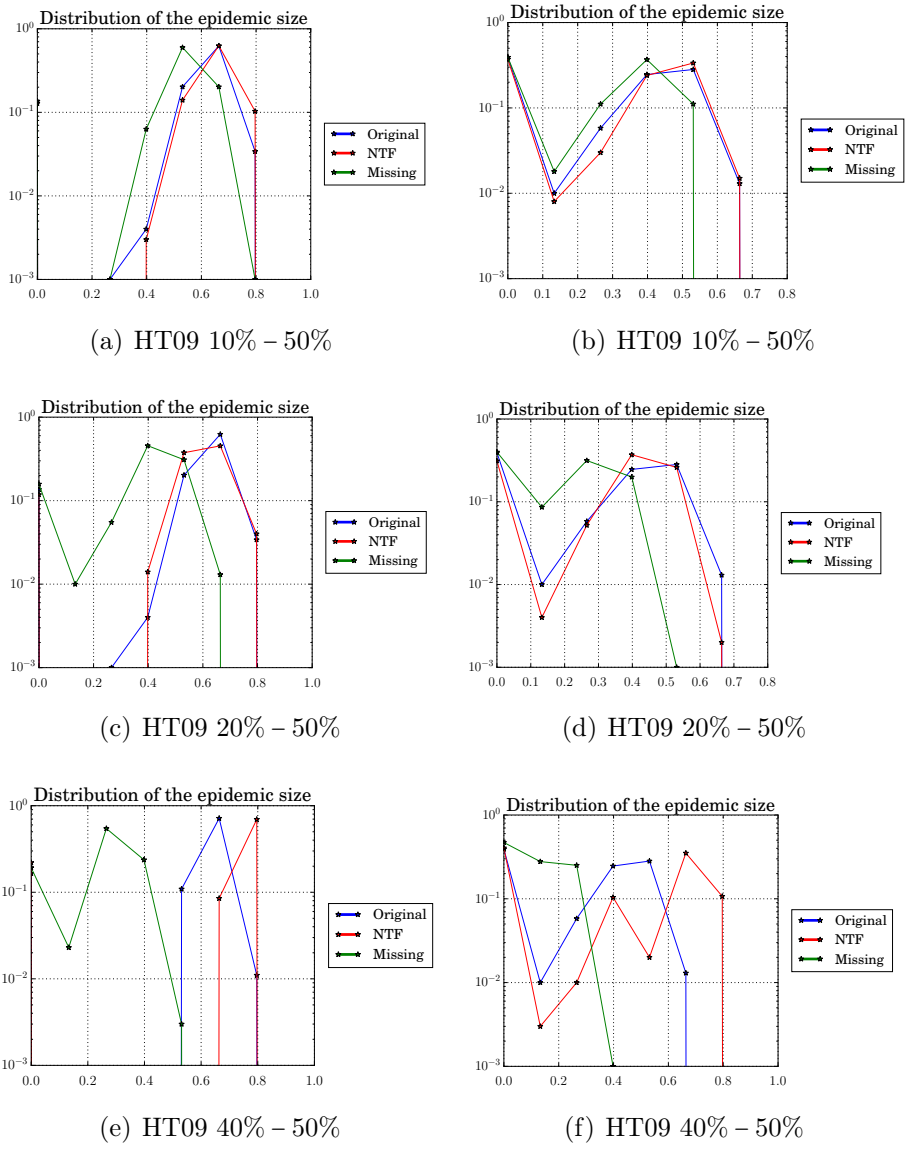
(e) LSCH 40% – 50%



(f) LSCH 40% – 50%

Fig. 5.2 **Distribution of the epidemic size** in the original, missing and recovered cases. We compare the results obtained **in the LSCH dataset** for 10%, 20% and 40% of nodes in which we erased the activity in the 50% of the times. The results are shown for two significant couples of probabilities: $\left(\lambda = 10^{-0.8}, \mu = 10^{-0.6}\right)$ and $\left(\lambda = 10^{-0.6}, \mu = 10^{-0.6}\right)$ for 10%, 20%, and $\left(\lambda = 10^{-1.4}, \mu = 10^{-1.3}\right)$ and $\left(\lambda = 10^{-1.3}, \mu = 10^{-0.95}\right)$ for 40%.

The difference in the couples of probabilities for the case with $p_{nodes} = 40\%$ is due to the selection procedure. To compare the distribution of the epidemic size we simulate SIR processes which start from the same root, which is a node chosen outside the missing node set.



(a) HT09 10% – 50%

(b) HT09 10% – 50%

(c) HT09 20% – 50%

(d) HT09 20% – 50%

(e) HT09 40% – 50%

(f) HT09 40% – 50%

Fig. 5.3 **Distribution of the epidemic size** in the original, missing and recovered cases. We compare the results obtained **in the HT09 dataset** for $10\%, 20\%$ and 40% of nodes in which we erased the activity in the 50% of the times. The results are shown for two significant couples of probabilities: $\left(\lambda = 10^{-0.6}, \mu = 10^{-1.3}\right)$ and $\left(\lambda = 10^{-0.5}, \mu = 10^{-0.8}\right)$ for $10\%, 20\%$, and $\left(\lambda = 10^{-0.5}, \mu = 10^{-1.1}\right)$ and $\left(\lambda = 10^{-0.5}, \mu = 10^{-0.8}\right)$ for 40%.

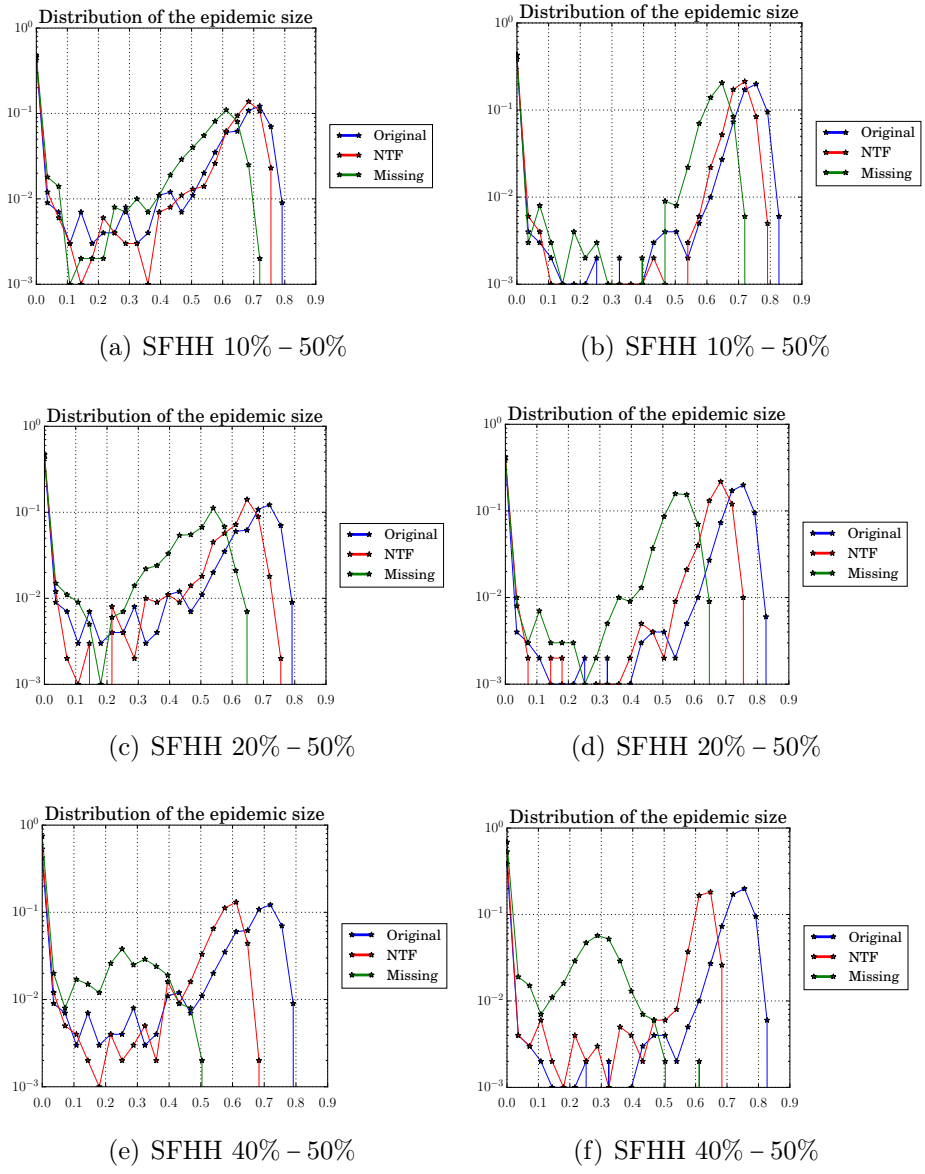
However, as the set of missing nodes changes, the selected root could be different from one case to the other, thus giving as a result different selected probabilities.

The results obtained for LSCH highlight the performance of the procedure, as it achieved a good match with the original case. It is worth noting that the results of the method are closer to the original case than those obtained by simply simulating the epidemic spreading on the network affected by the missing values.

In Fig. 5.3 we displayed the results achieved on the HT09 dataset. Here, we chose as couples of probabilities $\lambda = 10^{-0.6}, \mu = 10^{-1.3}$ and $\lambda = 10^{-0.5}, \mu = 10^{-0.8}$ for the cases with 10% and 20% of missing nodes, while for 40% of missing nodes we chose $\lambda = 10^{-0.5}, \mu = 10^{-1.1}$ and $\lambda = 10^{-0.5}, \mu = 10^{-0.8}$. The results achieved in the case with 10% and 20% of missing nodes are consistent to what was previously found for LSCH. The epidemic size is correctly kept in contrast to the results obtained from $\boldsymbol{\mathcal{M}}$. Even if the distributions in the case of 40% of missing nodes seem less precise if compared with the other cases, they are reasonable: as the number of groups of correlated activities in the network is small the deletion of a great amount of entries leads to the disruption of the interconnection in the groups. Therefore, the recovering of the missing values is more difficult and the results deviate a bit from the original. In any case the information provided by these results is realistic and closer to the original behaviour than the one obtained by the study of the network with partial information.

Finally we presented the results obtained by applying the procedure on the SFHH dataset. The results are shown in Fig. 5.4, in which we displayed the couples of probabilities $\lambda = 10^{-0.8}, \mu = 10^{-1.3}$ and $\lambda = 10^{-0.6}, \mu = 10^{-1.1}$ in all the three cases. As the results shown in the previous cases, the application of the procedure on the SFHH network approximated by the NTF allowed to get closer to the epidemic sizes of the original process. In the case of 40% of nodes with partial information the epidemic size is underestimated. However, it is a normal mechanism that can occur when the amount of missing values is drastically increasing. This result is due to the fact that to recover the missing information the NTF takes advantage of the correlated activity patterns in the time-varying network, and when the amount of lacking information is higher

the correlation between the activities of the links in the network decreases as many links are lost. To overcome this limitation, we propose the use of the JNTF to approximate the network by exploiting some external information.



(a) SFHH 10% – 50%

(b) SFHH 10% – 50%

(c) SFHH 20% – 50%

(d) SFHH 20% – 50%

(e) SFHH 40% – 50%

(f) SFHH 40% – 50%

Fig. 5.4 **Distribution of the epidemic size** in the original, missing and recovered cases. We compare the results obtained **in the SFHH dataset** for 10%, 20% and 40% of nodes in which we erased the activity in the 50% of the times. The results are shown for two significant couples of probabilities: $\left(\lambda = 10^{-0.8}, \mu = 10^{-1.3}\right)$ and $\left(\lambda = 10^{-0.6}, \mu = 10^{-1.1}\right)$.

JNTF results

As we have seen in the previous section, when the amount of missing values affecting the time-varying network drastically increases, the NTF has difficulty in recovering the missing information. This difficulty arises from the fact that when we erase contacts from the time-varying network, we are also destroying the correlated activity patterns between the links. Thus, as the NTF looks for correlation in different dimensions, the stronger this correlation is affected the worse is the final approximation.

To overcome this issue, we adopted the JNTF, which looks for correlations in the data from multiple data sources at a time. In this way, we are able to add the information lacking in the principal dataset by using some external knowledge about it. Among the three datasets used to test our procedure, the LSCH dataset has as an additional information the locations (rooms in the school) in which individuals were at each time. As explained in Chapter 3, in 15 locations of the school (e.g., classes, playground, etc.) there were antennas. The antennas registered the presence of individuals in their proximity, given as a result a time-varying location network, describing where individuals were at each time during the data collection process. The correlation between the two networks is then simple: if two individuals were in contact then they were in the same location; if two individuals were in the same location they could have been in contact.

To compare the JNTF results with the one obtained by the NTF we simulated a loss of data corresponding to $p_{nodes} = 0.2$ and a p_{times} of the time span, erased consecutively from its beginning, varying in the range $[0.1, 0.2, 0.3, \ldots, 1]$. We look at these different percentages to establish at which amount of missing data the NTF is not anymore able to recover the missing information and see how the JNTF can be used to fix this issue.

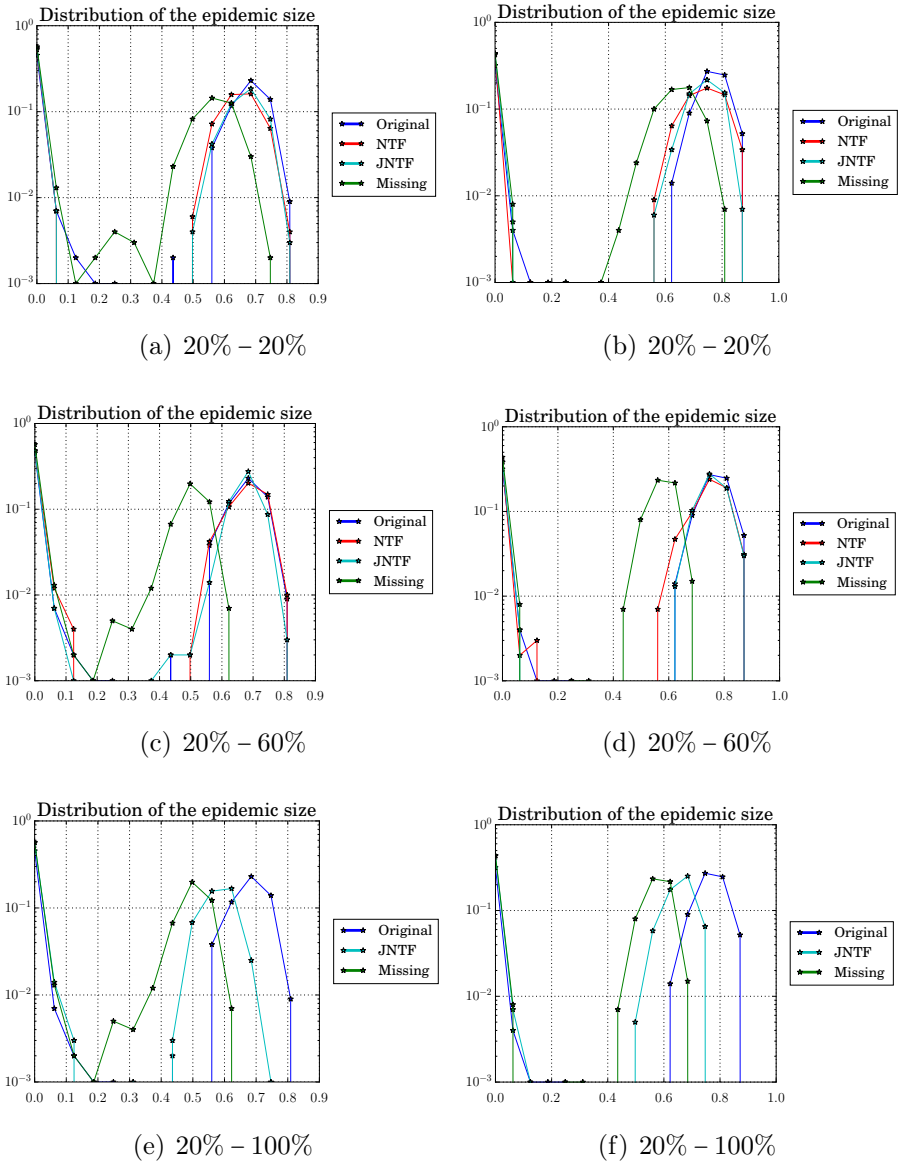
To assess the number of components for the JNTF, as this method computes the sum of several approximations at a time we use as an hint the value obtained by adopting the NTF on the tensor with the missing information, as it corresponds to the data that we want to approximate. As shown in the Table 5.5, the core consistency values corresponding to the best realizations over 20 runs of the JNTF are lower than the one achieved by the simple

NTF. However, this does not necessarily correspond to a worse approximation but finds an explanation in the fact that we are decomposing two tensors at the same time. Indeed, the involvement of external information that even if correlated is different from the tensor that we want to recover leads to lower core consistency values.

Table 5.5 **Core consistency values of the NTF and the JNTF best realizations** for each percentage of the time span erased and 20% of missing nodes.

% time	CC NTF	CC JNTF
10	89.9	77.7
20	89.7	75.3
30	89.7	72.2
40	92.3	75.7
50	89.5	73.7
60	89.8	72
70	89.6	75.7
80	90	70.4
90	90.2	59
100	90.4	70.5

To evaluate the results obtained on the simulation of the SIR processes and compare the NTF with the JNTF we reported in Fig. 5.5 the distribution of the epidemic size in the original network, the one with partial information and the two reconstructed by the approximation obtained with the NTF and the JNTF. In particular, we displayed as representative cases those corresponding to the $20\%, 60\%$ and 100% of time line erased. As the percentages is selected consecutively on the time span of the network instead of on the overall activity of the nodes, the cases between 30% and 60% coincide as in the LSCH datasets in that window of time no activity was recorded.



(a) 20% – 20% (b) 20% – 20%

(c) 20% – 60% (d) 20% – 60%

(e) 20% – 100% (f) 20% – 100%

Fig. 5.5 **Distribution of the epidemic size to compare the results achieved by using the NTF and the JNTF**. Here, we show as significant amount of data erased the couples 20% – 20%, 20% – 60%, and 20% – 100%. In the last case the NTF, which is displayed in the other figures with the red line, coincide to the green line which is the one obtained by simulating the SIR process on $\mathcal{M}$

For the comparison we show two couples of selected probabilities: $\lambda = 10^{-0.8}, \mu = 10^{-0.6}$ and $\lambda = 10^{-0.6}, \mu = 10^{-0.6}$. As we observe in Fig 5.5, the results obtained by the NTF and the JNTF are comparable in the case of 20% of the time span erased. The results in both cases well reproduce the outcome

of the SIR process, in opposition to those obtained on the partial network. The same observation can be done in the case of 40% of the time span erased, even if in this case the JNTF is more precise than the NTF to reproduce the distribution of the epidemic size. This is particularly clear in the right panel of the 20% – 60% case. Finally, we have shown the case in which the overall activity of the missing nodes is erased. In this case, the NTF is not able to recover any value as it cannot find any correlation in the data. For this reason we did not display the red curve, as the results for the NTF coincide to the one obtained on the partial network (green curve). It is worth noting that, despite the overall activity of the nodes was erased, the JNTF is able to capture, by using the external information provided by the co-location of the nodes, their groups of correlated activity. Thus even in this extreme case the method allows to get closer to the original outcome of the process.

5.2 Conclusion

In this chapter we have shown how to use tensor factorization techniques to uncover mesoscale properties for nodes for which we have only partial information. In other terms, we extracted meaningful groups of links from the partial network and we assigned groups and their corresponding properties (temporal activity) to nodes having partial information. This enabled us to build an approximation of the original network. In the second part we have shown how to adjust this approximation, to reproduce the evolution and characteristics of dynamical processes that occur over the networks.

To approximate the network, we adapted both the NTF and a generalized version of the NTF: the JNTF. This method allows to integrate the information provided by different types of time-varying networks. In the NTF case, the missing information can be recovered at the mesoscale level by analysing the correlation of the links having partial information with the other links in the network. This step allows to find the membership of the links having partial information and their temporal activity. In the JNTF case, the missing information is not only recovered by the correlated activity patterns established in the time-varying network of contacts, but also by using some external source of information to approximate the network. This is possible by assuming that

the external data are coupled with the source with partial information in some dimension.

As the approximation changes some of the relevant properties of the original network, we then described how to adjust these properties on the approximated network. To this aim, we reassigned the heterogeneity in the number of contacts per link and in their temporal activation, characteristics which were found to be essential for dynamical processes.

In conclusion, we managed to devise a methodology that successfully recover mesoscale properties of nodes from a partial network. We provided an alternative methodology based on the JNTF. This methodology helps in the case in which the percentage of missing values is so high, that it renders the NTF incapable of correctly recovering the missing information. The use of such a technique is possible when external data sources are available, as it gives a natural way to merge the information from different data sources.

Finally, we provided a procedure to reconstruct time-varying networks such that it enables to reproduce the outcome of an epidemics, even in case of missing information. We obtained a good agreement between the distributions of the epidemic size in the original and recovered networks. The achieved results highlight the impact that different characteristics of the network have in modelling a diffusion process.

The analysis carried out in the present chapter confirmed that heterogeneous properties, such as a broad distribution of weights and burstiness, as well as correlated activity patterns are principal ingredients found in empirical social network data and key aspects regarding spreading processes. Thus, we decided to use these ingredients to devise a generative model for time-varying networks. In the next chapter, we will see the main idea behind our generative model, which is based on the work we developed up to this point.

Chapter 6

Generative Model for Time-varying Networks

The studies which we carried out in the previous chapters helped us to understand and learn some meaningful properties about time-varying networks. In particular, we have shown that correlated activity patterns, uncovered by tensor decompositions, are an important part in the description of time-varying networks. However, tensor decompositions tend to approximate time-varying networks, by homogenizing some of their characteristics, e.g., the weight distribution. And we know, from both the literature and the work developed in the previous chapter, that heterogeneity properties are essential when studying dynamical processes occurring over networks.

In this chapter, we sketch a generative model for time-varying networks in which we take into account all the ingredients which we found to be important in our previous works. Our intention is to use this model to study the interplay between network properties and dynamical processes in a systematic way. We will show the results achieved, which, although preliminary, are promising and represent a basis for the development of our future work, which we delineate at the end of the chapter.

6.1 Topological and temporal effects

How can we affect dynamical processes occurring over a network? What are the elements that play a striking role in the evolution of processes such as a disease spreading? Is it possible to predict which are the properties of a network having a greater impact on the dynamics?

With the aim of answering these questions, a great amount of work have been devoted to investigate the underlying structure of networks and how its elements can affect dynamical processes. This is of particular relevance because some factors can strongly affect the evolution of dynamics such as spreading processes and can be used to plan targeted mitigation interventions.

In the recent years, several studies have shown that both temporal and topological characteristics in networks contribute to slow down dynamical processes [58, 78], by comparing the dynamics with randomized models. An alternative way to study the importance of such characteristics is by devising generative models for time-varying networks, as we mentioned in Section 1.7. To devise such models, empirical time-varying networks properties, e.g., inter-event time distribution, degree distribution, etc., are analysed. The purpose is to extract meaningful properties to generate synthetic networks in which realistic topological and temporal correlations can be tuned [68].

As introduced in Section 1.7, an example of generative model model for time-varying networks was proposed by Granell et al. [91]. The model, based on stochastic block models, generate a synthetic network from the combination of sub-networks in which the nodes are tied by a certain probability. Here, the number of links in the network is drawn from a binomial distribution, and the temporal activation of the links is determined by a periodic function, i.e. triangular waveform.

However, the aforementioned models either rely on randomization techniques or take into account the global properties of the network to assign them to the single nodes. In this work, we are interested in devising a generative model of time-varying networks, which is not based on a randomization technique and in which we can create the network by means of mesoscale structures. This model can be used to generate synthetic networks, built from the sum of several sub-networks, in which we assign different correlated activity patterns with

certain properties in both time and topology. Our aim is to study the interplay between the assigned correlated activity patterns and dynamical processes occurring over the network.

6.2 Generative model

One major challenge is the study of the effect of coupled temporal and topological properties of networks on dynamical processes. As we mentioned before, a lot of work has been devoted to answer questions on how the dynamical processes are affected by the network properties. However, usually either the temporal or the topological aspects are investigated.

Our aim is then to devise a generative model of time-varying networks in which we are able to assign specific characteristics in both time and topology. Having the control on such properties and having the possibility of modulating them, give us a way of assessing their impact on dynamical processes.

To devise such a generative model we started from one key idea: a time-varying network can be seen as composed by a sum of several sub-networks. This observation comes from the previous studies on time-varying networks, which we carried out through the use of tensor factorization techniques. As we have seen in the previous chapters, a time-varying network can be represented as a tensor that the NTF is able to approximate through the sum of rank-one tensors. Generally speaking, these rank-one tensors can be interpreted as sub-networks, in which only part of the original links are active. Each rank-one tensor in fact will be the representation of one group of links and their correlated activity patterns in the original network. This idea has also a natural interpretation when considering time-varying social networks: people are often engaged in several activities, which might involve different people at different times.

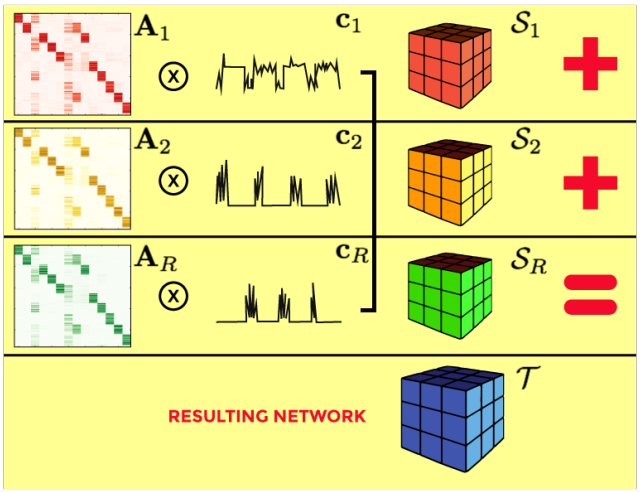


Fig. 6.1 **Generative model for time-varying networks**. A synthetic time-varying network is created by summing several sub-networks. Each sub-network $\mathcal{S}_r$ is characterized by a topological structure $\mathbf{A}_r$ which is modulated in time by a specific temporal pattern $\mathbf{c}_r$. Each sub-network is characterized by different properties in topology and time.

Starting from this idea, we use as an hint the factor matrices of the NTF to generate time-varying networks. The procedure is shown in Fig. 6.1 and for each sub-network $\mathcal{S}_r$ can be divided as follows:

1. we create an adjacency matrix $\mathbf{A}_r$ on the basis of observations made on empirical networks (see details below);
2. we modulate the matrix $\mathbf{A}_r$ by computing its outer product with a temporal activity.

Finally, we sum all the sub-networks to have the final synthetic time-varying network $\boldsymbol{\mathcal{T}}$.

To generate the topological structures we started from the observations derived by Gauvin et al. [167], whose aim was to determine and rank the mesoscale structures identified by the NTF by studying which has an impact on a spreading process. In the paper, the mesoscale structures having a major

impact on the dynamics are identified in those composed by mixed clusters (mixed school classes). Such sub-networks are then characterized by a lower weighted clustering coefficient, defined as [171]:

$$C_i = \frac{\sum_{j,k} w_{ij} w_{jk} w_{ik}}{\sum_{j \neq k} w_{ij} w_{ik}},$$

where w_{ij} is the weight of the link (i, j). Thus we decided to generate the topological structures, i.e. the matrices with the links of a certain sub-network, in a way that ensure of having different clustering coefficients in different structures. To this aim, we relied on observations made on empirical time-varying social networks: we require that the ratio of the largest to the lowest node membership in a component is bounded, and that a node is usually involved in a limited number of components.

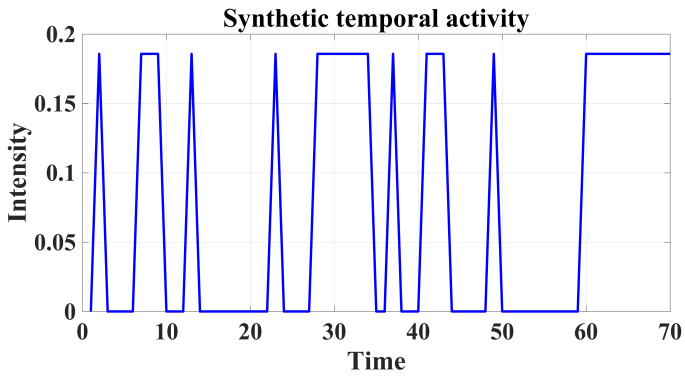


Fig. 6.2 **Synthetic temporal activity** used to modulate a topological structure in time and creating a sub-network. The activity is built on the basis of a temporal activity of the components in the decomposition of the LSCH dataset. The activity is binary after the application of the Otsu threshold and normalized afterwards

Then we adapt the temporal activities of the components in the LSCH decomposition, which we use to modulate the topological part, as follows. We apply the Otsu threshold on the temporal activities, to detect when they were active or inactive. We use the resulting threshold on the temporal activities to make them binary and finally we normalize them. An example of the resulting temporal activities $\mathbf{c}_r$ is given in Fig. 6.2. With this procedure we are able to generate sub-networks characterized by different clustering coefficient and different temporal activities.

6.3 Results

One of the strengths of this model is that we are able to combine and control different topological and temporal structures to create a synthetic time-varying network. Moreover, as the model is additive, we are not only able to create synthetic networks by summing several sub-networks, but we can also remove one sub-network at a time, as shown in Fig. 6.3. This procedure allows to assess the impact of each sub-network on the evolution of a dynamical process.

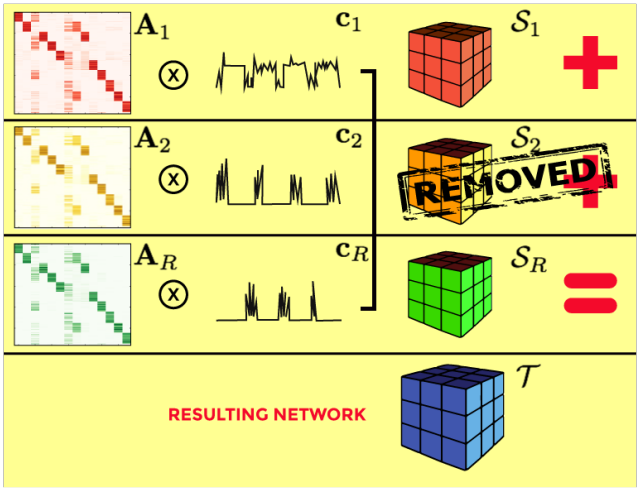


Fig. 6.3 **Procedure to determine the impact of a mesoscale structure in time-varying networks**. We start from the synthetic network $\mathcal{T}$ which we generate by the sum of several sub-networks $\mathcal{S}_r$. We remove one $\mathcal{S}_r$ at a time to compare the original network to the one in which one mesoscale structure is not present anymore.

To study the impact of each sub-network, we remove one of them at a time from the overall synthetic network. Then, we simulate an SI process over the original network and over the one generated by the removal of one sub-network. Finally we compute the delay ratio [172, 167], which is defined as

$$\tau_r = \left\langle \frac{t_j^r - t_j}{t_j} \right\rangle$$

where t_j is the time in which half of the population is infected in the original network with the infection starting from node j, t_j^r is the half infection time in the network with the removal.

As we assigned on purpose different clustering coefficients to the different sub-networks, we first of all investigated if there exists a correlation between the delay ratio and the clustering coefficient. Here we present the case in which a synthetic network is created by $N = 12$ sub-networks. Each sub-network is built by following the procedure explained in the previous section.

After the removal procedure for all the sub-networks and the related simulation of the SI processes, we computed the delay-ratio values for the 12 cases. We have shown the results in Fig. 6.4

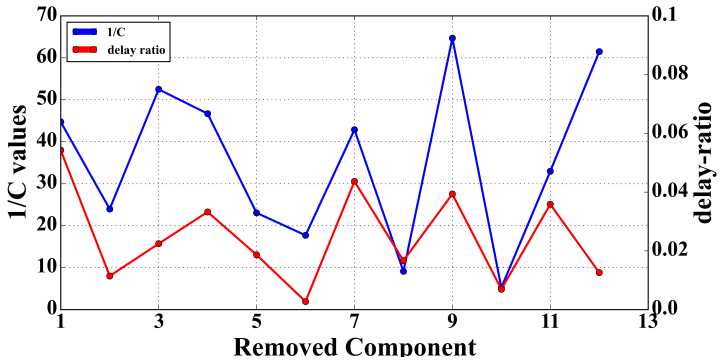


Fig. 6.4 **Delay-ratio values and the inverse of the clustering coefficient** are shown for each of the 12 sub-network removals. The x-axis indicates the removed sub-network.

As we can observe in the figure, there is a clear correlation between the delay-ratio and the inverse of the clustering coefficient values: the lower is the clustering coefficient the higher is the delay-ratio. The only case that does not follow the observed trend corresponds to the removal of the 12-th sub-network. However, by further investigating the properties of this sub-network we noticed that even if its topological structure is characterized by a low clustering coefficient, the activation of the links occur late in time, as shown by the temporal activity in Fig 6.5.

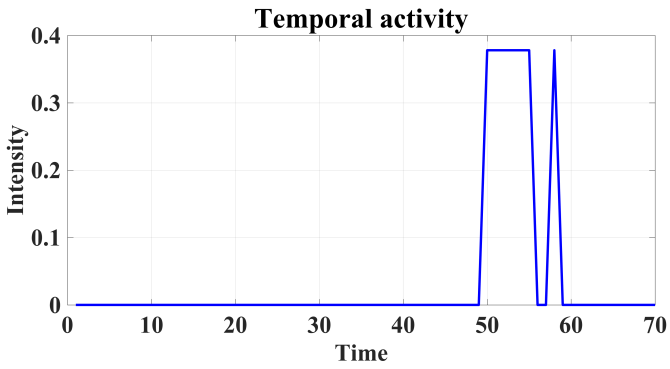


Fig. 6.5 **Temporal activity** of the 12-th sub-network used to build the final synthetic time-varying network. The total activation of the time-series is concentrated on the second half of the time line.

Therefore, as the group of links of the 12-th sub-network activates late in time, by construction, the removal of this sub-network cannot have a significant impact on delaying the time in which half of the population is infected.

6.4 Future work

As we have shown in the previous sections, our generative model allows to create synthetic time-varying networks. Our method is simple as it is based on an additive representation of time-varying networks. This representation enables to control the temporal and topological properties of the sub-networks and we have seen that the clustering coefficient is an important factor to determine the impact of a sub-network on dynamical processes.

Encouraged by these promising results, we are now decided to modify the generative model to be able of creating time-varying networks in which the sub-networks are characterized by heterogeneous properties, such as burstiness. To this purpose, we further studied the characteristics of the temporal and topological properties of the components in empirical networks to be able to assign similar characteristics and create new synthetic structures.

In particular, we would like to adopt the **negative binomial distribution** as a candidate to assign the structural and temporal properties of the network. We use this distribution to assign the weight of links belonging to the same sub-network. In this way not only we act on the number of contacts per link but we

also erase partially the correlated activity of the nodes in the same component, thus rendering their temporal activity patterns bursty. This is important because we have seen that the heterogeneous properties are fundamental when studying the interplay between time-varying networks and dynamical properties.

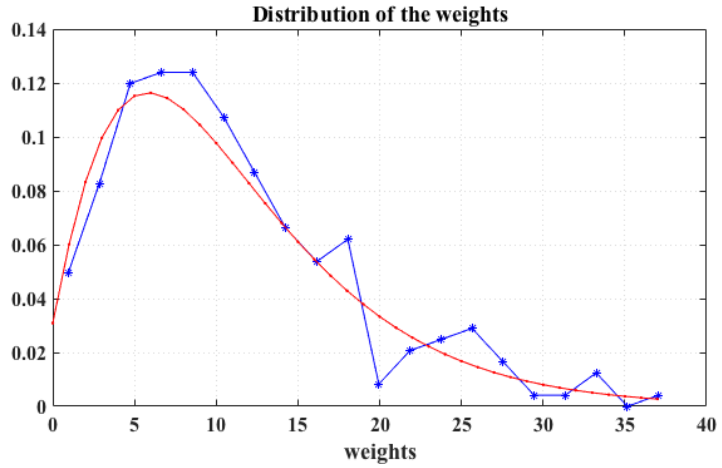


Fig. 6.6 **Fit of the weight distribution of a sub-network by means of the negative binomial distribution**. The sub-network is given by the decomposition of the LSCH dataset.

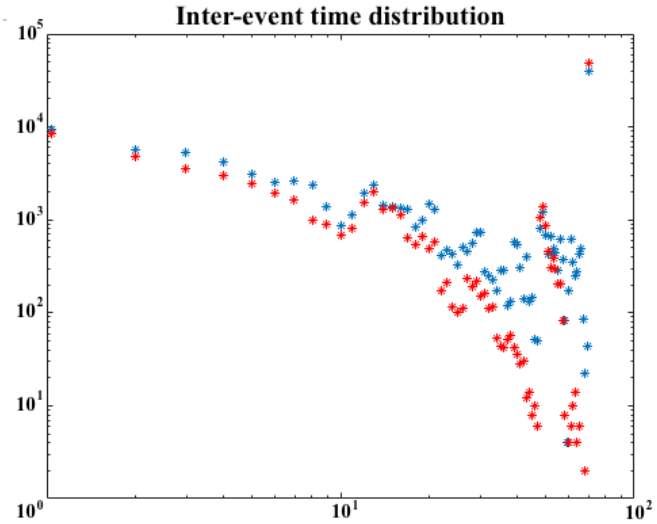
We propose the use of the negative binomial distribution

$$D(x) = \sum_{n=0}^{x} \binom{n+r-1}{r-1} p^r (1-p)^n$$

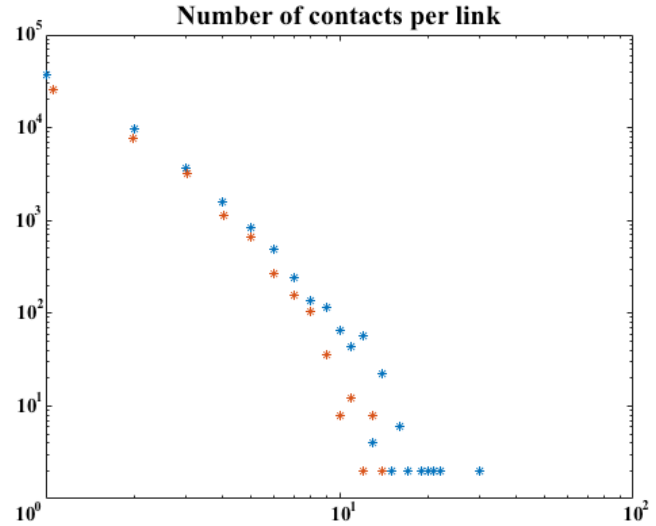
as we observed that it is particularly suitable to fit the weight distribution of the sub-networks (corresponding to the components of the NTF). We provide in Fig. 6.6 an example of fit of the weight distribution of the links in one of the aforementioned sub-networks with the negative binomial distribution.

To test the impact of this distribution in the model, we first take into account the LSCH dataset as a starting point. We decompose the network and use the binomial distribution to generate new topological and temporal structures in which we can assign the weight to the links of the synthetic network. To assign the new weights to the links we use a similar approach to the one presented in Section 5.1.2: we select a weight from the negative binomial distribution; we look at the temporal activity of the link; we erase link activation randomly in time, such that the final weight equals the one to

be assigned. In this way we both keep partial correlated activity between links belonging to the same sub-network, and we introduce heterogeneity properties in time and topology.



(a) Inter-event time distribution



(b) Number of contacts per links distribution

Fig. 6.7 **Inter-event time distributions and number of contacts per links distributions** in the original and synthetic network.

As shown in Fig. 6.7, through this procedure, we are able to find a number of contacts per link distribution and an inter-event time distribution which are in good agreement with what we observed in real contact networks. Therefore, the resulting synthetic network is characterized by real network structures and properties. It has been shown that in some cases, generating a network with these two distributions taken from the original network is enough to reproduce the outcome of a spreading process on the original network [60].

6.5 Conclusion

Devising a generative model for time-varying networks is a major challenge that helps in the investigation of the interplay between the network properties and dynamical processes which occur over the network.

Here we sketched a generative model in which we can control both the temporal and topological characteristics and combining them. Our model is the result of the conclusions of the previous works which we carried out to study time-varying networks through tensor decomposition. Indeed, we used the NTF as a basis to create a generative model in which a synthetic time-varying network is created by summing multiple sub-networks at a time. This assumption is supported by our previous studies in which we decomposed time-varying networks into the sum of elementary pieces, corresponding to sub-networks.

This additive construction could remind the block structure of stochastic block models, where blocks correspond to sub-networks. However, our sub-networks are built from a static network modulated in time. In this way, each sub-network is characterized by specific topological and temporal structure at the mesoscale level, thus corresponding to a particular correlated activity pattern, which is not present in stochastic block models.

The simulations of SI processes on the original synthetic network and on the altered networks, where one sub-network was removed at a time, brought to the identification of the clustering coefficient value as a decisive factor to predict the impact of a certain sub-network on the spreading process. It is worth noting that our model architecture, although preliminary, is such that

we can act separately on temporal activities and structural properties. This construction allows to extend the study to richer temporal patterns and to test the robustness of our observations.

We are now going further in the definition of the generative model by refining the way we are creating the synthetic temporal and topological patterns. As we have learnt from the literature and our previous work, heterogeneous properties, such as the number of contacts per link and burstiness, are fundamental to create realistic time-varying networks. We proposed to include such heterogeneities while building the topological and temporal patterns in the generative model. A natural candidate, which we observed to be suitable for such purpose, is the negative binomial distribution, which we used to assign the weight distribution to each sub-network in the model. This way of using the negative binomial distribution helps in the generation of networks displaying inter-event time distributions and number of contacts per link distributions which resemble the one encountered in real-world networks. And, these distributions are known to be crucial in the outcome of a spreading process.

In conclusion, we proposed a model to generate synthetic time-varying networks, with partial correlated activity patterns. We know that these are important characteristics found in empirical networks. Nevertheless, we want also to explore new directions for generating time-varying networks. To this purpose, we are now collaborating with Tiago Peixoto on reproducing the outcome of SI processes with a generative model for time-varying networks based on variable-order hidden Markov chain model [85]. This model differs from the one we proposed in this chapter as it is based on maximum likelihood estimation techniques, which provide in an unsupervised fashion the set of parameters, such as the number of components, required to generate the network.

Conclusion

The study of time-varying networks is a wide research topic, which has been tackled in many fields from several points of view. This is a direct consequence of the fact that time-varying networks are appropriate objects to represent a wide range of complex natural systems, in a concise manner.

In the development of my Ph.D thesis I have looked at the problem of analysing time-varying networks with a slightly different point of view from the one usually adopted in complex systems fields. Indeed, all the methods devised in the present work rise from the intersection between multiple disciplines: multi-linear algebra, machine learning and complex network theory. We took advantage of the knowledge developed in these disciplines to provide an answer to specific questions related to time-varying networks.

In the first part of this work, I have focused on the analysis of the specific properties of time-varying networks, by trying to give an answer to several questions about the network itself: how are the elements in time-varying networks organized? What are their relations in topology and time? How can we detect such relations and their evolution? Which are the meanings of the detected relations?

To answer to these questions I decided to study time-varying networks by taking advantage of one special group of mathematical methods: tensor decomposition techniques. As the literature suggests, tensor decompositions can be applied to handle a great variety of applications. However, what I found to be the most powerful advantage of such techniques is that, once you learn how these techniques are built and how to apply them to your specific problem, it is possible to modify their framework to face several issues. Therefore, the first main result of this thesis corresponds to the development of several methods and

procedures in which we **extended the mathematical framework** behind tensor decompositions.

The second main ingredient in the present work consists in the fact that by applying tensor decomposition techniques on time-varying networks, we have studied their element organization and their relations at one specific level: **the mesoscale level**. As a result, tensor decompositions provide an **additive representation** of time-varying networks, in which the uncovered structures (even from one application to another) have all a common characteristic: they are **correlated activity patterns**. Thus, our understanding of time-varying networks and the related properties is based on the fact that time-varying networks often display an organization of their elements which corresponds to similarities in both topology and time.

The correlated activity patterns which can be uncovered have an interpretation which strongly depends on the application. We have seen that such patterns can be related in time-varying social networks to people sharing different activities with others at different times. We have also seen that these activity patterns can be linked to anomalous behaviours caused by data collection processes. For this reason, we provided an iterative methodology to face the problem of detecting anomalies in time varying networks, where the anomalies are characterized both in topology and time.

The ability of uncovering correlated activity patterns does not end in itself. Indeed, it can be used to face other issues. This is the case of recovering properties of nodes in the network, whose activity is partially known, which is a fundamental problem as time-varying networks are often affected by missing information. To face such a problem, we exploited the existing correlations in link activations to find the groups of correlated activity from the partial information we know and use their relations to recover some properties about nodes at the mesoscale level. This problem was tackled by approximating time-varying networks in two different ways. First, we used the correlated activity patterns in the network itself to infer the missing information at the mesoscale level. Second, we have shown how to infer some properties of the missing information by the coupling of external data sources.

While I was tackling different problems as anomaly detection or missing data recovery, I was also deepen my understanding in the **implications** of applying

tensor decomposition techniques on time-varying networks. As we have seen, the resulting network provided by the decomposition is an approximation of the original network. Thus, some of the properties of the network are kept while some others are modified. The awareness of this fact led to the identification of one another essential ingredient in my study: tensor decompositions modify the **heterogeneity properties** of time-varying networks. The approximated network has indeed a less broader weight distribution which reflects also in the mitigation of bursty activations of the links in the network.

These observations put together opened new questions about the study of time-varying networks in relation with dynamical processes: is the approximated network enough to recover the outcome of dynamical processes? If not, which are the essential properties to recover the dynamics? How can we insert back these properties in the approximated network? What is the impact of such properties on the dynamics?

In the second part of the present thesis, we then focused on the analysis of the interplay of the network properties and dynamical processes. As a first step, we identified the main characteristics needed to recover the outcome of diffusion processes, such as epidemic spreading. We have seen how to adjust the network characteristics after the approximation given by the tensor decomposition to recover the outcome of the dynamics. The results of this work confirmed what is also pointed out in the literature: heterogeneity properties, such as the **number of contacts per link** and **burstiness**, are essential when studying dynamical processes on time-varying networks.

At this stage of my investigation I have collected a certain number of clues which turned out to be essential to study the impact of network characteristics on dynamical processes. To summarize, we found that the presence of two main ingredients is essential in time-varying networks: correlated activity patterns, and heterogeneity properties. These observations led to the intention of studying the impact of such features in a systematic way. To this aim, we detached from tensor decomposition techniques and took all the elements discovered to combine them into a generative model of time-varying networks.

The idea behind this generative model is indeed given by the main results achieved in my research: the generation of time-varying networks in which we take advantage of an additive representation, like tensor decompositions; we

build each piece of the network as a correlated activity pattern in which we can control both the temporal and topological characteristics; finally we introduce heterogeneity properties which allow to recover realistic distributions of the number of contacts per link and inter-event time.

In conclusion, the overall work I presented in this thesis can be divided in two main parts: in the first part I zoomed in the analysis of time-varying networks to decompose them in meaningful pieces of informations, which I collected and combined in the second part to generate time-varying networks and investigate their interplay with dynamical processes.

References

[1] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[2] Alain Barrat, Marc Barthelemy, and Alessandro Vespignani. *Dynamical processes on complex networks*. Cambridge University Press, 2008.

[3] Márton Karsai, Nicola Perra, and Alessandro Vespignani. Time varying networks and the weakness of strong ties. *arXiv preprint arXiv:1303.5966*, 2013.

[4] Michele Starnini, Andrea Baronchelli, Alain Barrat, and Romualdo Pastor-Satorras. Random walks on temporal networks. *Physical Review E*, 85 (5):056115, 2012.

[5] Nicola Perra, Andrea Baronchelli, Delia Mocanu, Bruno Gonçalves, Romualdo Pastor-Satorras, and Alessandro Vespignani. Random walks and search in time-varying networks. *Physical review letters*, 109(23):238701, 2012.

[6] Bruno Ribeiro, Nicola Perra, and Andrea Baronchelli. Quantifying the effect of temporal resolution on time-varying networks. *arXiv preprint arXiv:1211.7052*, 2012.

[7] Pierre Comon. Tensors: a brief introduction. *IEEE Signal Processing Magazine*, 31(3):44–53, 2014. intro to tensor decompositions: tensors formal definitions and problem statement + approximate decompositions.

[8] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. Overview of higher-order tensors and their decompositions. properties, decomposition models, softwares.

[9] Ning Hao, Lior Horesh, and ME Kilmer. Nonnegative tensor decomposition. In *Compressed Sensing & Sparse Filtering*, pages 123–148. Springer, 2014.

[10] Lek-Heng Lim and Pierre Comon. Nonnegative approximations of nonnegative tensors. *Journal of chemometrics*, 23(7-8):432–441, 2009. PARAFAC degeneracy and the existence of a solution for the nonnegative case.

[11] Laetitia Gauvin, André Panisson, and Ciro Cattuto. Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach. *PloS one*, 9(1):e86028, 2014.

[12] Anna Sapienza, André Panisson, Joseph Wu, Laetitia Gauvin, and Ciro Cattuto. Anomaly detection in temporal graph data: An iterative tensor decomposition and masking approach. *Proceedings of AALTD. Portugal*, page 117, 2015.

[13] Anna Sapienza, Joseph Wu, Laetitia Gauvin, Ciro Cattuto, et al. Detecting anomalies in time-varying networks using tensor decomposition. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 516–523. IEEE, 2015.

[14] Alain Barrat, C Cattuto, V Colizza, F Gesualdo, L Isella, E Pandolfi, J-F Pinton, L Ravà, C Rizzo, M Romano, et al. Empirical temporal networks of face-to-face human interactions. *The European Physical Journal Special Topics*, 222(6):1295–1309, 2013.

[15] Michele Starnini, Andrea Baronchelli, and Romualdo Pastor-Satorras. Modeling human dynamics of face-to-face interaction networks. *Physical review letters*, 110(16):168701, 2013.

[16] Michele Starnini, Andrea Baronchelli, and Romualdo Pastor-Satorras. Model reproduces individual, group and collective dynamics of human contact networks. *arXiv preprint arXiv:1409.0507*, 2014.

[17] Carey E Priebe, John M Conroy, David J Marchette, and Youngser Park. Scan statistics on enron graphs. *Computational & Mathematical Organization Theory*, 11(3):229–247, 2005.

[18] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.

[19] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, et al. High-resolution measurements of face-to-face contact patterns in a primary school. *PloS one*, 6(8):e23176, 2011.

[20] Marcel Salathé, Maria Kazandjieva, Jung Woo Lee, Philip Levis, Marcus W Feldman, and James H Jones. A high-resolution human contact network for infectious disease transmission. *Proceedings of the National Academy of Sciences*, 107(51):22020–22025, 2010.

[21] André Panisson, Alain Barrat, Ciro Cattuto, Wouter Van Den Broeck, Giancarlo Ruffo, and Rossano Schifanella. On the dynamics of human

proximity for data diffusion in ad-hoc networks. *Ad Hoc Networks*, 10(8): 1532–1543, 2012.

[22] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308, 2006.

[23] L da F Costa, Francisco A Rodrigues, Gonzalo Travieso, and Paulino Ribeiro Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in physics*, 56(1):167–242, 2007. Survey on the characteristics of complex networks (no temporal dimension considered).

[24] John Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 31–36. ACM, 2009.

[25] Raj Kumar Pan and Jari Saramäki. Path lengths, correlations, and centrality in temporal networks. *Physical Review E*, 84(1):016105, 2011.

[26] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012. Collection of properties and formalisms related to TVG to unify the notation and concepts: definitions, classes, static properties to dynamic properties.

[27] Vincenzo Nicosia, John Tang, Cecilia Mascolo, Mirco Musolesi, Giovanni Russo, and Vito Latora. Graph metrics for temporal networks. In *Temporal Networks*, pages 15–40. Springer, 2013. definitions of walks, paths, connectedness and connected components for temporal graphs.

[28] René Pfitzner, Ingo Scholtes, Antonios Garas, Claudio J Tessone, and Frank Schweitzer. Betweenness preference: Quantifying correlations in the topological dynamics of temporal networks. *Physical review letters*, 110(19):198701, 2013.

[29] John Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Characterising temporal distance and reachability in mobile and online social networks. *ACM SIGCOMM Computer Communication Review*, 40(1): 118–124, 2010.

[30] John Tang, Salvatore Scellato, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Small-world behavior in time-varying graphs. *Physical Review E*, 81(5):055101, 2010.

[31] John Tang, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Vincenzo Nicosia. Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, page 3. ACM, 2010.

[32] Guilherme Ferraz de Arruda, André Luiz Barbieri, Pablo Martín Rodriguez, Yamir Moreno, Luciano da Fontoura Costa, and Francisco Aparecido Rodrigues. The role of centrality for the identification of influential spreaders in complex networks. *arXiv preprint arXiv:1404.4528*, 2014.

[33] Clara Stegehuis, Remco van der Hofstad, and Johan SH van Leeuwaarden. Epidemic spreading on complex networks with community structures. *Scientific Reports*, 6, 2016.

[34] Paolo Bajardi, Alain Barrat, Fabrizio Natale, Lara Savini, and Vittoria Colizza. Dynamical patterns of cattle trade movements. *PloS one*, 6(5): e19869, 2011.

[35] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005, 2011.

[36] Nick Koudas, Sunita Sarawagi, and Divesh Srivastava. Record linkage: similarity measures and algorithms. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 802–803. ACM, 2006.

[37] Elizabeth A Leicht, Petter Holme, and Mark EJ Newman. Vertex similarity in networks. *Physical Review E*, 73(2):026120, 2006.

[38] Santo Fortunato. Community detection in graphs. *Physics reports*, 486 (3):75–174, 2010.

[39] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[40] Michael Molloy and Bruce Reed. A critical point for random graphs with a given degree sequence. *Random structures & algorithms*, 6(2-3): 161–180, 1995.

[41] Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. Random graphs with arbitrary degree distributions and their applications. *Physical review E*, 64(2):026118, 2001.

[42] Vincenzo Nicosia, Giuseppe Mangioni, Vincenza Carchiolo, and Michele Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03024, 2009.

[43] Hang-Hyun Jo, Márton Karsai, János Kertész, and Kimmo Kaski. Circadian pattern and burstiness in mobile phone communication. *New Journal of Physics*, 14(1):013055, 2012.

[44] Juliette Stehlé, Alain Barrat, and Ginestra Bianconi. Dynamical and bursty interactions in social networks. *Physical review E*, 81(3):035101, 2010.

[45] Enrico Ubaldi, Alessandro Vezzani, Marton Karsai, Nicola Perra, and Raffaella Burioni. Burstiness and tie reinforcement in time varying social networks. *arXiv preprint arXiv:1607.08910*, 2016.

[46] Alvaro Sanchez and Ido Golding. Genetic determinants and cellular constraints in noisy gene expression. *Science*, 342(6163):1188–1193, 2013.

[47] K-I Goh and A-L Barabási. Burstiness and memory in complex systems. *EPL (Europhysics Letters)*, 81(4):48002, 2008.

[48] R Dean Malmgren, Daniel B Stouffer, Adilson E Motter, and Luís AN Amaral. A poissonian explanation for heavy tails in e-mail communication. *Proceedings of the National Academy of Sciences*, 105(47):18153–18158, 2008.

[49] Antoine Moinet, Michele Starnini, and Romualdo Pastor-Satorras. Burstiness and aging in social temporal networks. *Physical review letters*, 114 (10):108701, 2015.

[50] Márton Karsai, Kimmo Kaski, Albert-László Barabási, and János Kertész. Universal features of correlated bursty behaviour. *Scientific reports*, 2, 2012.

[51] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925, 2015.

[52] Petter Holme. Representations of human contact patterns and outbreak diversity in sir epidemics. *IFAC-PapersOnLine*, 48(18):127–131, 2015.

[53] Hang-Hyun Jo, Juan I Perotti, Kimmo Kaski, and János Kertész. Analytically solvable model of spreading dynamics with non-poissonian processes. *Physical Review X*, 4(1):011041, 2014.

[54] Alexey N Medvedev and Janos Kertesz. Empirical study of the role of the topology in spreading on communication networks. *arXiv preprint arXiv:1607.01484*, 2016.

[55] Mark EJ Newman, Stephanie Forrest, and Justin Balthrop. Email networks and the spread of computer viruses. *Physical Review E*, 66(3): 035101, 2002.

[56] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Vittoria Colizza, Lorenzo Isella, Corinne Régis, Jean-François Pinton, Nagham Khanafer, Wouter Van den Broeck, et al. Simulation of an seir infectious disease model on the dynamic contact network of conference attendees. *BMC medicine*, 9(1):1, 2011.

[57] Giovanna Miritello, Esteban Moro, and Rubén Lara. Dynamical strength of social ties in information spreading. *Physical Review E*, 83(4):045102, 2011.

[58] Márton Karsai, Mikko Kivelä, Raj Kumar Pan, Kimmo Kaski, János Kertész, A-L Barabási, and Jari Saramäki. Small but slow world: How network topology and burstiness slow down spreading. *Physical Review E*, 83(2):025102, 2011.

[59] Dávid X Horváth and János Kertész. Spreading dynamics on networks: the role of burstiness, topology and non-stationarity. *New Journal of Physics*, 16(7):073037, 2014.

[60] Laetitia Gauvin, André Panisson, Ciro Cattuto, and Alain Barrat. Activity clocks: spreading dynamics on temporal networks of human contact. *arXiv preprint arXiv:1306.4626*, 2013.

[61] Christian L Vestergaard, Mathieu Génois, and Alain Barrat. How memory generates heterogeneous dynamics in temporal networks. *Physical Review E*, 90(4):042805, 2014.

[62] Alexei Vazquez, Balazs Racz, Andras Lukacs, and Albert-Laszlo Barabasi. Impact of non-poissonian activity patterns on spreading processes. *Physical review letters*, 98(15):158702, 2007.

[63] Sungmin Lee, Luis EC Rocha, Fredrik Liljeros, and Petter Holme. Exploiting temporal network structures of human interaction to effectively immunize populations. *PloS one*, 7(5):e36439, 2012.

[64] José Luis Iribarren and Esteban Moro. Impact of human activity patterns on the dynamics of information diffusion. *Physical review letters*, 103(3): 038702, 2009.

[65] Anna Machens, Francesco Gesualdo, Caterina Rizzo, Alberto E Tozzi, Alain Barrat, and Ciro Cattuto. An infectious disease model on empirical networks of human contact: bridging the gap between dynamic network data and contact matrices. *BMC infectious diseases*, 13(1):1, 2013.

[66] Juan Fernández-Gracia, Víctor M Eguíluz, and Maxi San Miguel. Update rules and interevent time distributions: Slow ordering versus no ordering in the voter model. *Physical Review E*, 84(1):015103, 2011.

[67] Jean-Charles Delvenne, Renaud Lambiotte, and Luis EC Rocha. Diffusion on networked systems is a question of time or structure. *Nature communications*, 6, 2015.

[68] Oriol Artime, José J Ramasco, and Maxi San Miguel. Dynamics on networks: competition of temporal and topological correlations. *arXiv preprint arXiv:1604.04155*, 2016.

[69] J-P Onnela, Jari Saramäki, Jorkki Hyvönen, György Szabó, David Lazer, Kimmo Kaski, János Kertész, and A-L Barabási. Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences*, 104(18):7332–7336, 2007.

[70] Vitaly Belik, Florian Fiebig, and Philipp Hövel. Controling contagious processes on temporal networks via adaptive rewiring. *arXiv preprint arXiv:1509.04054*, 2015.

[71] Petter Holme. Temporal network structures controlling disease spreading. *arXiv preprint arXiv:1605.00915*, 2016.

[72] Yang Liu, Yong Deng, Marko Jusup, and Zhen Wang. A biologically inspired immunization strategy for network epidemiology. *Journal of theoretical biology*, 400:92–102, 2016.

[73] Vitaly Belik, Philipp Hövel, and Rafael Mikolajczyk. Control of epidemics on hospital networks. In *Control of Self-Organizing Nonlinear Systems*, pages 431–440. Springer, 2016.

[74] Enys Mones, Arkadiusz Stopczynski, Alex Pentland, Nathaniel Hupert, and Sune Lehmann. Vaccination and complex social dynamics. *arXiv preprint arXiv:1603.00910*, 2016.

[75] Valerio Gemmetto, Alain Barrat, and Ciro Cattuto. Mitigation of infectious disease at school: targeted class closure vs school closure. *BMC infectious diseases*, 14(1):1, 2014.

[76] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic dynamics and endemic states in complex networks. *Physical Review E*, 63(6): 066117, 2001.

[77] Maksim Kitsak, Lazaros K Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H Eugene Stanley, and Hernán A Makse. Identification of influential spreaders in complex networks. *Nature physics*, 6(11):888–893, 2010.

[78] Suyu Liu, Nicola Perra, Marton Karsai, and Alessandro Vespignani. Controlling contagion processes in time-varying networks. *arXiv preprint arXiv:1309.7031*, 2013.

[79] Nicola Perra, Bruno Gonçalves, Romualdo Pastor-Satorras, and Alessandro Vespignani. Activity driven modeling of time varying networks. *arXiv preprint arXiv:1203.5351*, 2012.

[80] Garry Robins and Philippa Pattison. Random graph models for temporal processes in social networks*. *Journal of Mathematical Sociology*, 25(1): 5–41, 2001.

[81] Alain Barrat, Bastien Fernandez, Kevin K Lin, and Lai-Sang Young. Modeling temporal networks using random itineraries. *Physical review letters*, 110(15):158702, 2013.

[82] Steve Hanneke, Wenjie Fu, Eric P Xing, et al. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

[83] Andrea EF Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time in edge-markovian dynamic graphs. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*, pages 213–222. ACM, 2008.

[84] Tiago P Peixoto. Model selection and hypothesis testing for large-scale network models with overlapping groups. *Physical Review X*, 5(1):011033, 2015.

[85] Tiago P Peixoto and Martin Rosvall. Modeling sequences and temporal networks with dynamic community structures. *arXiv preprint arXiv:1509.04740*, 2015.

[86] Kun Zhao, Juliette Stehlé, Ginestra Bianconi, and Alain Barrat. Social network dynamics of face-to-face interactions. *Physical review E*, 83(5): 056109, 2011.

[87] Santo Fortunato, Alessandro Flammini, and Filippo Menczer. Scale-free network growth by ranking. *Physical review letters*, 96(21):218701, 2006.

[88] Marián Boguná and Romualdo Pastor-Satorras. Class of correlated random networks with hidden variables. *Physical Review E*, 68(3):036112, 2003.

[89] Guillaume Laurent, Jari Saramäki, and Márton Karsai. From calls to communities: a model for time-varying social networks. *The European Physical Journal B*, 88(11):1–10, 2015.

[90] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[91] Clara Granell, Richard K Darst, Alex Arenas, Santo Fortunato, and Sergio Gómez. Benchmark model to assess community structure in evolving networks. *Physical Review E*, 92(1):012805, 2015.

[92] Tamara Gibson Kolda. *Multilinear operators for higher-order decompositions*. United States. Department of Energy, 2006. Definition of Kruskal operator + its properties it's used to define the PARAFAC decomposition.

[93] Frank L Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1):164–189, 1927.

[94] Raymond B Cattell. The three basic factor-analytic research designs—their interrelations and derivatives. *Psychological bulletin*, 49 (5):499, 1952.

[95] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 122137, 1963.

[96] Ledyard R Tucker. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology*, 110119, 1964.

[97] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[98] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis. 1970.

[99] Richard A Harshman and Margaret E Lundy. Parafac: Parallel factor analysis. *Computational Statistics & Data Analysis*, 18(1):39–72, 1994.

[100] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[101] Rasmus Bro. Parafac. tutorial and applications. *Chemometrics and intelligent laboratory systems*, 38(2):149–171, 1997. General notion of parafac + application in chemometrics.

[102] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[103] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[104] Joseph B Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, 18(2):95–138, 1977.

[105] Joseph B Kruskal. Rank, decomposition, and uniqueness for 3-way and n-way arrays. In *Multiway data analysis*, pages 7–18. North-Holland Publishing Co., 1989.

[106] Heather Michele Clyburn Bush. Khatri-rao products and conditions for the uniqueness of parafac solutions for ixjxk arrays. 2006.

[107] Johan Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4): 644–654, 1990.

[108] Nicholas D Sidiropoulos and Rasmus Bro. On the uniqueness of multilinear decomposition of n-way arrays. *Journal of chemometrics*, 14(3):229–239, 2000. Generalization of Kruskal results on decomposition uniqueness from three-linear decomp to multilinear one.

[109] Vin De Silva and Lek-Heng Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

[110] Pierre Comon, Gene Golub, Lek-Heng Lim, and Bernard Mourrain. Symmetric tensors and symmetric tensor rank. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1254–1279, 2008.

[111] Dario Bini, Grazia Lotti, and Francesco Romani. Approximate solutions for the bilinear form computational problem. *SIAM Journal on Computing*, 9(4):692–697, 1980.

[112] JB Kruskal, RA Harshman, and ME Lundy. How 3-mfa data can cause degenerate parafac solutions, among other relationships. *Multiway data analysis*, pages 115–121, 1989.

[113] Yang Qi, Pierre Comon, and Lek-Heng Lim. Uniqueness of nonnegative tensor approximations. *arXiv preprint arXiv:1410.8129*, 2014.

[114] J Douglas Carroll, Geert De Soete, and Sandra Pruzansky. Fitting of the latent class model via iteratively reweighted least squares candecomp with nonnegativity constraints. In *Multiway data analysis*, pages 463–472. North-Holland Publishing Co., 1989.

[115] Lek-Heng Lim. Optimal solutions to non-negative parafac/multilinear nmf always exist. In *Workshop on Tensor Decompositions and Applications, Centre International de rencontres Mathématiques, Luminy, France*, 2005.

[116] Pentti Paatero. Least squares formulation of robust non-negative factor analysis. *Chemometrics and intelligent laboratory systems*, 37(1):23–35, 1997.

[117] Rasmus Bro and Sijmen De Jong. A fast non-negativity-constrained least squares algorithm. *Journal of chemometrics*, 11(5):393–401, 1997. Modification of DeFacto algorithm to compute in efficient way the NNLS for PARAFAC decomposition by considering also the nonnegative constraints.

[118] Giorgio Tomasi and Rasmus Bro. A comparison of algorithms for fitting the parafac model. *Computational Statistics & Data Analysis*, 50(7): 1700–1734, 2006.

[119] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. An optimization approach for fitting canonical tensor decompositions. *Sandia National Laboratories, Tech. Rep. SAND2009-0857*, 2009.

[120] Hyunsoo Kim, Haesun Park, and Lars Eldén. Non-negative tensor factorization based on alternating large-scale non-negativity-constrained least squares. In *2007 IEEE 7th International Symposium on BioInformatics and BioEngineering*, pages 1147–1151. IEEE, 2007. NTF algorithm based on ANLS + regularization.

[121] Jingu Kim and Haesun Park. Fast nonnegative tensor factorization with an active-set-like method. In *High-Performance Scientific Computing*, pages 311–326. Springer, 2012. NTF computed by ANLS and BPP algorithms.

[122] Ake Björck. *Numerical methods for least squares problems*. Siam, 1996.

[123] Chikio Hayashi and Fumi Hayashi. A new algorithm to solve parafac-model. *Behaviormetrika*, 9(11):49–60, 1982.

[124] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944.

[125] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[126] Pentti Paatero. A weighted non-negative least squares algorithm for three-way 'parafac'factor analysis. *Chemometrics and Intelligent Laboratory Systems*, 38(2):223–242, 1997.

[127] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799. ACM, 2005.

[128] Max Welling and Markus Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.

[129] Yong-Deok Kim and Seungjin Choi. Nonnegative tucker decomposition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[130] Joaquim J Júdice and Fernanda M Pires. A block principal pivoting algorithm for large-scale strictly monotone linear complementarity problems. *Computers & operations research*, 21(5):587–596, 1994.

[131] Charles L Lawson and Richard J Hanson. *Solving least squares problems*, volume 15. SIAM, 1995.

[132] Jingu Kim and Haesun Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In *2008 Eighth IEEE International Conference on Data Mining*, pages 353–362. IEEE, 2008.

[133] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[134] Jean-Philip Royer, Nadege Thirion-Moreau, and Pierre Comon. Computing the polyadic decomposition of nonnegative third order tensors. *Signal Processing*, 91(9):2159–2171, 2011. Preconditionated nonlinear conjugate gradient algorithms. Add nonnegative constraints to parafac model by using adamard product of the components.

[135] Marieke E Timmerman and Henk AL Kiers. Three-mode principal components analysis: Choosing the numbers of components and sensitivity to local optima. *British journal of mathematical and statistical psychology*, 53(1):1–16, 2000.

[136] Morten Mørup and Lars Kai Hansen. Automatic relevance determination for multi-way models. *Journal of Chemometrics*, 23(7-8):352–363, 2009.

[137] Kefei Liu, João Paulo CL da Costa, Hing Cheung So, Lei Huang, and Jieping Ye. Detection of number of components in candecomp/parafac models via minimum description length. *Digital Signal Processing*, 51: 110–123, 2016.

[138] Rasmus Bro and Henk AL Kiers. A new efficient method for determining the number of components in parafac models. *Journal of chemometrics*, 17(5):274–286, 2003. Definition of the core consistency diagnostic.

[139] Evangelos E Papalexakis and Christos Faloutsos. Fast efficient and scalable core consistency diagnostic for the parafac decomposition for big sparse tensors. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5441–5445. IEEE, 2015. Core consistency algorithm to make the computation faster and allocating less memory.

[140] Joao Paulo CL da Costa, Martin Haardt, Florian Romer, and Giovanni Del Galdo. Enhanced model order estimation using higher-order arrays. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, pages 412–416. IEEE, 2007.

[141] Mati Wax and Thomas Kailath. Detection of signals by information theoretic criteria. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(2):387–392, 1985.

[142] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[143] Angela Quinlan, Jean-Pierre Barbot, Pascal Larzabal, and Martin Haardt. Model order selection for short data: An exponential fitting test (eft). *EURASIP Journal on Applied Signal Processing*, 2007(1):201–201, 2007.

[144] Timo Smieszek, Stefanie Castell, Alain Barrat, Ciro Cattuto, Peter J White, and Gérard Krause. Contact diaries versus wearable proximity sensors in measuring contact patterns at a conference: method comparison and participants' attitudes. *BMC infectious diseases*, 16(1):341, 2016.

[145] Ciro Cattuto, Wouter Van den Broeck, Alain Barrat, Vittoria Colizza, Jean-François Pinton, and Alessandro Vespignani. Dynamics of person-to-person interactions from distributed rfid sensor networks. *PloS one*, 5 (7):e11596, 2010.

[146] William Eberle and Lawrence Holder. Discovering structural anomalies in graph-based data. In *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pages 393–398. IEEE, 2007.

[147] Misael Mongiovi, Petko Bogdanov, Razvan Ranca, Ambuj K Singh, Evangelos E Papalexakis, and Christos Faloutsos. Netspot: Spotting significant anomalous regions on dynamic networks. In *Proceedings of the 13th SIAM international conference on data mining (SDM), Texas-Austin, TX*. SIAM, 2013.

[148] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[149] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[150] Dharanipragada Janakiram, VA Reddy, and AVU Phani Kumar. Outlier detection in wireless sensor networks using bayesian belief networks. In *2006 1st International Conference on Communication Systems Software & Middleware*, pages 1–6. IEEE, 2006.

[151] Joel W Branch, Chris Giannella, Boleslaw Szymanski, Ran Wolff, and Hillol Kargupta. In-network outlier detection in wireless sensor networks. *Knowledge and information systems*, 34(1):23–54, 2013.

[152] Jie-Fang Liu and Ning Zhou. Localization anomaly detection for wireless sensor networks. In *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, volume 2, pages 644–648. IEEE, 2010.

[153] Tsuyoshi Idé and Hisashi Kashima. Eigenspace-based anomaly detection in computer systems. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 440–449. ACM, 2004.

[154] Jimeng Sun, Yinglian Xie, Hui Zhang, and Christos Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *SDM*, pages 366–377. SIAM, 2007.

[155] Hadi Fanaee Tork, Márcia Oliveira, João Gama, Simon Malinowski, and Ricardo Morla. Event and anomaly detection using tucker3 decomposition. In *Workshop on Ubiquitous Data Mining*, page 8, 2012.

[156] Mei Fang, GuangXue Yue, and QingCang Yu. The study on an application of otsu method in canny operator. In *International Symposium on Information Processing (ISIP)*, pages 109–112. Citeseer, 2009.

[157] David Sankoff and Joseph B Kruskal. Time warps, string edits, and macromolecules: the theory and practice of sequence comparison. *Reading: Addison-Wesley Publication, 1983, edited by Sankoff, David; Kruskal, Joseph B.*, 1, 1983.

[158] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[159] Rossana Mastrandrea and Alain Barrat. How to estimate epidemic risk from incomplete contact diaries data? *PLoS Comput Biol*, 12(6):e1005002, 2016.

[160] Gueorgi Kossinets. Effects of missing data in social networks. *Social networks*, 28(3):247–268, 2006.

[161] Mark Huisman. Imputation of missing network data: some simple procedures. *Journal of Social Structure*, 10(1):1–29, 2009.

[162] Roger Guimerà and Marta Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52):22073–22078, 2009.

[163] Aaron Clauset, Cristopher Moore, and Mark EJ Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453 (7191):98–101, 2008.

[164] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008.

[165] Myunghwan Kim and Jure Leskovec. The network completion problem: Inferring missing nodes and edges in networks. In *SDM*, volume 11, pages 47–58. SIAM, 2011.

[166] Mathieu Génois, Christian L Vestergaard, Ciro Cattuto, and Alain Barrat. Compensating for population sampling in simulations of epidemic spread on temporal contact networks. *Nature communications*, 6, 2015.

[167] Laetitia Gauvin, André Panisson, Alain Barrat, and Ciro Cattuto. Revealing latent factors of temporal networks for mesoscale intervention in epidemic spread. *arXiv preprint arXiv:1501.02758*, 2015.

[168] Jean-Philip Royer, Nadege Thirion-Moreau, and Pierre Comon. Nonnegative 3-way tensor factorzation taking into account possible missing data. In *20th European Signal Processing Conference (EUSIPCO-2012)*, pages 1–5. Elsevier, 2012.

[169] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*, 2011.

[170] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Poblano v1. 0: A matlab toolbox for gradient-based optimization. *Sandia National Laboratories, Albuquerque, NM and Livermore, CA, Tech. Rep. SAND2010-1422*, 2010.

[171] Jari Saramäki, Mikko Kivelä, Jukka-Pekka Onnela, Kimmo Kaski, and Janos Kertesz. Generalizations of the clustering coefficient to weighted complex networks. *Physical Review E*, 75(2):027105, 2007. Alternative definitions for weighted clustering coefficients.

[172] Michele Starnini, Anna Machens, Ciro Cattuto, Alain Barrat, and Romualdo Pastor-Satorras. Immunization strategies for epidemic processes in time-varying contact networks. *Journal of theoretical biology*, 337: 89–100, 2013.