Imperial College London
Faculty of Natural Sciences
Department of Mathematics

Latent factor representations of dynamic networks with applications in cyber-security

Francesco Sanna Passino

October, 2020

Supervised by Professor Nicholas A. Heard

Submitted in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in Statistics of Imperial College London
and the Diploma of Imperial College London

Statement of originality

I certify that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

Francesco Sanna Passino
October, 2020

Copyright

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Nick Heard. His knowledge, expertise, and enthusiasm for the subject have been inspirational, and his valuable advice, guidance, and support have been crucial to complete this project. Over the years, Nick became not only an academic I admire, but also a role model I look up to in life, and I am profoundly grateful for that.

I am also thankful for the financial support received from the Engineering and Physical Sciences Research Council (EPSRC) and the Department of Mathematics at Imperial College London. Working on this PhD project would have not been possible without their funding.

I would like to thank my internal examiners for the PhD progression milestones, Professor Almut Veraart and Dr Sarah Filippi, whose suggestions greatly improved the quality of this work.

I am also extremely grateful to Dr Melissa Turcotte, Dr Anna Bertiger and Dr Joshua Neil, who supervised me during summer internships at the Los Alamos National Laboratory and Microsoft Corporation. Their expertise and guidance helped me understand subject-specific details and crucial aspects of the data and application. I would also like to thank Dr Patrick Rubin-Delanchy for the helpful discussions on some parts of this project.

A special thanks goes the many friends that I met over the years at Imperial. They carried me through the ups and downs of research, helping me more than they can imagine.

Finally, I would like to thank my family and closest friends for their love and encouragement.

$\mathscr{F}$.

Abstract

Dynamic networks frequently arise in nature, representing, for example, neuronal connectivity in the brain, internet connections between computers, and human interactions within social networks. Motivated by specific characteristics of computer networks, this thesis gives novel contributions to three important branches of statistical analysis of dynamic graphs: modelling of connectivity events, graph clustering, and link prediction. Nodes and edges within the network will be assumed to have latent, unobserved characteristics, estimated using statistical latent factor models.

The first part of this work proposes edge-based models for individual connection events. A model for dynamic network evolution based on the Pitman-Yor process is proposed, which is used for network-wide anomaly detection. Furthermore, a model for classification of periodic arrivals in event time data is proposed, to separate human and automated activity within individual edges. Correct classification of these two types of activity is fundamental for effective intrusion detection.

In its second part, this work proposes novel methodologies for graph clustering. Finding nodes with similar connectivity patterns is important for reliable network monitoring. A model for simultaneous estimation of the latent dimension and number of communities in spectral clustering is proposed, under a generalised random dot product graph interpretation of the stochastic blockmodel. The model is then extended to allow for heterogeneous within-community degree distributions, proposing a novel spectral clustering algorithm under the degree corrected stochastic blockmodel.

Finally, in the third part of this thesis, link prediction methods are discussed. The ability to correctly associate anomaly scores with the connections in a network is crucial for the cyber-defence of an organisation. In this work, it is demonstrated that random dot product graphs are a powerful and scalable tool for link prediction in large graph. Furthermore, an extension of the popular Poisson matrix factorisation model is proposed, which correctly models binary matrices and allows the effects of nodal covariates to be estimated. Scalable inferential techniques are also discussed.

Overall, this work gives contributions towards a unified model for cyber-security applications, in a Bayesian framework, with the ultimate aim to dynamically predict the connectivity of IP addresses, or users and hosts, in a computer network.

Table of Contents

List of Algorithms, Figures and Tables

Algorithms

Figures

Tables

1 Introduction

The role of statistical and data science methods in cyber-security and cyber-defence applications has become increasingly important in recent years. Intrusion attempts on computer networks occur daily via a number of different attack methods, such as distributed denial-of-service (DDoS), fast flux, man-in-the-middle, ransomware, phishing, or worms. In computer networks, a number of high-volume and high-frequency data sources are available, within which it might be possible to detect the presence of malicious activity in the network. To improve the security of computer networks in government, industry, and academia, there is a requirement for developing probabilistic model-based techniques for identifying the most subtle intrusion attempts using these data sources. The advantage of statistical approaches is their ability to learn, from historical data, complex patterns of normal computer and network behaviour. Statistical methods can be used to complement more traditional *signature-based* methods, which are only able to detect attacks for which a signature has previously been created, in order to identify more subtle attacks. In particular, statistical techniques represent the main tool for *anomaly-based detection*, which looks for deviations from a model of the normal behaviour of the network (see, for example, Chandola, Banerjee, and Kumar, 2009). Anomaly detection has a relevant advantage over signature-based methods: previously unobserved attacks, or *zero-day exploits*, can still potentially be identified. Existing techniques for anomaly-based intrusion detection are surveyed in Lazarevic et al. (2003), Patcha and Park (2007), Liao et al. (2013), Buczak and Guven (2016), Jeske et al. (2018) and Perusquía, Griffin, and Villa (2020). Detailed accounts of statistical approaches to network security problems can be also found in many academic books, for example Marchette (2001); Adams and Heard (2014); Adams and Heard (2016); Heard et al. (2018). More generally, some of the most popular anomaly detection methods on networks are spectral methods (Idé and Kashima, 2004), scan statistics (Priebe et al., 2005), and random walks (Tong, Faloutsos, and Pan, 2006). Comprehensive overviews of anomaly detection methods specifically targeted towards network data are given by Akoglu, Tong, and

Koutra (2015), Ranshous et al. (2015), and Ahmed, Naser Mahmood, and Hu (2016).

Motivated by challenges posed by cyber-security problems, this thesis proposes novel statistical methodologies for modelling the normal behaviour of a computer network. The network will be mathematically interpreted as a collection of nodes, connected by edges. The underlying approach of all the modelling choices in this thesis will be to assume latent unobserved characteristics of nodes, edges or individual events, which are generically denoted as *latent factors*. Consequently, the concept of latent variable modelling will be extensively used across all chapters in this thesis. A number of different inferential techniques will be used, but particular attention will be given to the Bayesian approach, often overlooked in statistical cyber-security because of scalability concerns.

For cyber-security applications, given the large scale of the problems, it is often required to combine evidence from multiple data sources and performing analyses of the same data at multiple resolutions. There are essentially three possible types of models that can be built in a computer network: *global* models, aimed at describing the entire network, and *node-based* and *edge-based* models, constructed at the node and edge level.

For node-based or edge-based approaches, it is possible to construct sophisticated models, which can then be applied to the entire network using distributed computing and big data platforms, like Hadoop MapReduce. Examples are the two-state Markov modulated process proposed in Scott and Smyth (2003), the adaptive threshold method for network counts of Lambert and Liu (2006), and the two-stage Bayesian anomaly detection method of Heard et al. (2010). Within cyber-security applications, Neil et al. (2013) use scan statistics to detect anomalous subgraphs, and Turcotte, Heard, and Neil (2014) propose a two-stage approach where edges are given individual anomaly scores and then grouped into anomalous subgraphs. Latent variable models offer again a natural framework to model the activity on a single edge or node: it is assumed that the observed connections are related to values attributed to unobserved, or latent, components. In this thesis, two novel models for individual connection events will be proposed in Part I.

Node-based or edge-based models typically treat all unobserved edges equally, and this motivates the need for a network-wide understanding of node similarity and likely new edges. For such global models, it is necessary to use tools that are sophisticated enough to capture the complexity of the problem, but also realistically computable. Scalable techniques that could be used in this context are spectral clustering (von Luxburg, 2007), singular value decompositions (Dunlavy, Kolda, and Acar, 2011), or Poisson matrix factorisation (Gopalan, Hofman, and Blei, 2015). Latent position models (Hoff, Raftery, and Handcock, 2002) have also been successfully used on anomaly detection applications related to cyber-security (Lee et al., 2019). The idea behind these methods is to represent the probability of a connection between two entities as a function of their node-specific feature vectors. In this thesis, global models will be used for two fundamental tasks for understanding computer networks: *graph clustering* (also known as *community detection*), and *link prediction*. Community detection is the task of finding meaningful groups of nodes within

the network (Girvan and Newman, 2002; Fortunato, 2010), and will be discussed in Part II of this thesis. Community detection methods have been extensively used for anomaly detection in large networks (see, for example, Gao et al., 2010; Ji, Yang, and Gao, 2013). On the other hand, link prediction is aimed at predicting the presence of an edge between two nodes in a network. Predicting probabilities of links is valuable for detection of anomalous connections (Huang and Zeng, 2006; Turcotte et al., 2016). In particular, it is necessary to correctly score *new links* (Metelli and Heard, 2019), representing previously unobserved connections. The task is particularly important since it is common to observe malicious activity associated with new links, for example lateral movement (Neil et al., 2013), and it is therefore crucial to understand the normal process of link formation in order to detect a cyber-attack. Methods for link prediction in computer networks will be proposed in Part III of this work.

In this thesis, the proposed methodologies will be applied on high quality data from real world computer networks, collected at Imperial College London (ICL) and Los Alamos National Laboratory (LANL). The availability of such data sources is particularly significant, since the lack of appropriate datasets has been identified as one of the main limitations to widespread applications of data science methods to cyber-security problems (Kumar, Wicker, and Swann, 2017; Anderson et al., 2018; Amit et al., 2019). In particular, the LANL user-authentication data (Kent, 2016; Turcotte, Kent, and Hash, 2018) are used to test models for normal behaviour of users and hosts within the network. Understanding user-authentication patterns within an enterprise network is particularly important for network security, since over 80% of network breaches in 2019 involved brute force or the use of stolen credentials (Verizon, 2020). Similarly, Imperial College London collects network flow data, which summarise connections between internet protocol (IP) addresses. Models on network flows have been successfully used to detect botnets, denial-of-service attacks, network scans, and worms (Sperotto et al., 2010). Particular characteristics of user-authentication and network flow data will drive and motivate many of the modelling approaches proposed in this thesis. The next section briefly establishes the mathematical framework used in this work for statistical analysis of computer network data.

1.1. Mathematical representation of networks

A *network*, or *graph*, is a collection of *vertices*, or *nodes*, connected by *edges*, or *links* (Kolaczyk, 2009; Newman, 2010), written $\mathbb{G} = (V, E)$, where V is the node set and E is the edge set. An edge between two vertices i and j is usually denoted as $(i, j) \in E$. For *undirected* graphs, $(i, j) \in E \iff (j, i) \in E$, sometimes written $i \sim j$ in the literature, and i and j are said to be *neighbours* or *adjacent*. If the graph is *directed*, the previous statement does not necessarily hold, and each edge has a specific *direction* or *arrow*, pointing from one node to another, for example $i \to j$, meaning $(i, j) \in E$ but not necessarily that $(j, i) \in E$. An *adjacency matrix*

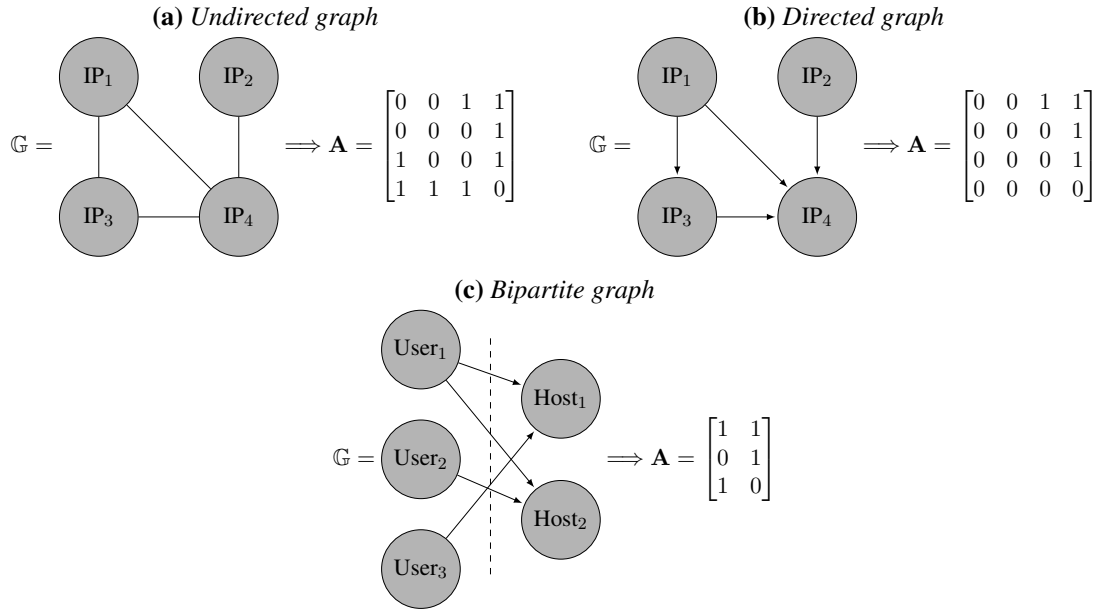


Figure 1.1.: *Toy examples of computer network graphs $\mathbb{G}$ and corresponding adjacency matrix $\mathbf{A}$. For (a) and (b), $V = \{IP_1, IP_2, IP_3, IP_4\}$, whereas for (c) $V_1 = \{User_1, User_2, User_3\}$ and $V_2 = \{Host_1, Host_2\}$.*

$\mathbf{A} = \{A_{ij}\} \in \{0,1\}^{|V|\times|V|}$ can be associated with any graph, taking $A_{ij} = \mathbb{1}_E\{(i,j)\}$, where $|\cdot|$ denotes the cardinality of the set and $\mathbb{1}.\{\cdot\}$ is the indicator function. It will be assumed that a node cannot link to itself, implying $\mathbf{A}$ is a *hollow* matrix. Sometimes the node set of a network can be divided into two subsets, say $V = V_1 \cup V_2$, $V_1 \cap V_2 = \varnothing$, such that edges are only of the type $(i,j) \in E$, with $i \in V_1$ and $j \in V_2$; then, the graph is said to be *bipartite*. The sets V_1 and V_2 are called *parts* of the graph and the graph itself is usually denoted as $\mathbb{G} = \{(V_1, V_2), E\}$. Bipartite graphs allow for a convenient formulation of the adjacency matrix, which can be written in *rectangular* form as a $|V_1| \times |V_2|$ matrix, with rows corresponding to the nodes in V_1 and columns to the nodes in V_2. In this case, $A_{ij} = 1$ if $i \in V_1$ connects to $j \in V_2$, and $A_{ij} = 0$ otherwise. An extremely simple representation of a computer network is a graph where, for example, the set V is made up of IP addresses, or, in the bipartite graph case, V_1 corresponds to users, and V_2 to hosts. Three toy examples are given in Figure 1.1.

The row and column sums of the adjacency matrix $\mathbf{A}$ of a directed graph are respectively denoted as *out-degree* and *in-degree sequences*:

$$d_i^{\text{out}} = \sum_{j=1}^{|V|} A_{ij},\ i = 1, \ldots, |V|, \qquad d_j^{\text{in}} = \sum_{i=1}^{|V|} A_{ij},\ j = 1, \ldots, |V|. \tag{1.1}$$

For an undirected graph with adjacency matrix $\mathbf{A}$, the row and column sums are identical, and

simply known as the *degree sequence*:

$$d_i = \sum_{i=1}^{|V|} A_{ij} = \sum_{j=1}^{|V|} A_{ij},\ i = 1, \ldots, |V|.$$

The degree sequence of an undirected graph is usually arranged in a $|V| \times |V|$ diagonal matrix known as the *degree matrix*:

$$\mathbf{D} = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_{|V|} \end{bmatrix}.$$

The degree matrix is used in the literature to define the *Laplacian matrix* $\mathbf{D} - \mathbf{A}$. Many *normalised* versions of the Laplacian matrix have been defined. In this thesis, the *normalised symmetric Laplacian matrix* (von Luxburg, 2007) is used:

$$\mathbf{L} = \mathbf{I}_{|V|} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2},$$

where $\mathbf{I}_{\cdot}$ is the $(|V| \times |V|)$-dimensional identity matrix. For the application to computer networks, $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is usually preferred to $\mathbf{I}_{|V|} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, since the eigenvalues of the former lie in $[-1, 1]$, providing a convenient interpretation for disassortative networks (Khor, 2010; Rubin-Delanchy, Adams, and Heard, 2016). In fact, computer networks tend to exhibit a disassortative structure: similar nodes (for example, printers) do not communicate with similar entities (other printers), but with dissimilar nodes.

The definition of graph can be further extended to networks that are not static over time. For example, in computer networks, a stochastic process is observed on each edge, with multiple connection events observed at different time points. In this context, the graph is said to be *dynamic*. Following Metelli and Heard (2019), the connections and corresponding arrival times on a dynamic network can be interpreted as a *marked point process* $(\mathcal{T}, \mathcal{X}, \mathcal{Y}) = \{(T_i, X_i, Y_i)_{i \geq 1}\}$, where T_i is a random variable on $\mathbb{R}_+$, corresponding to the *arrival time*, and (X_i, Y_i) is a *mark* in $V \times V$ (or $V_1 \times V_2$ for a bipartite graph) corresponding to a network edge. The *realised* sequences of event times and corresponding marks are denoted with $0 \leq t_1 \leq t_2 \leq \cdots$ and $(x_1, y_1), (x_2, y_2), \ldots$ respectively. From $(\mathcal{T}, \mathcal{X}, \mathcal{Y})$, a *dynamic graph* $\mathbb{G}_t = (V, E_t)$ is defined from the right-continuous stochastic process $\{E_t : t \geq 0\}$ of observed network edges. More formally:

$$E_t = \{(x, y) : (x, y) \in V \times V,\ N_{xy}(t) > 0\},$$

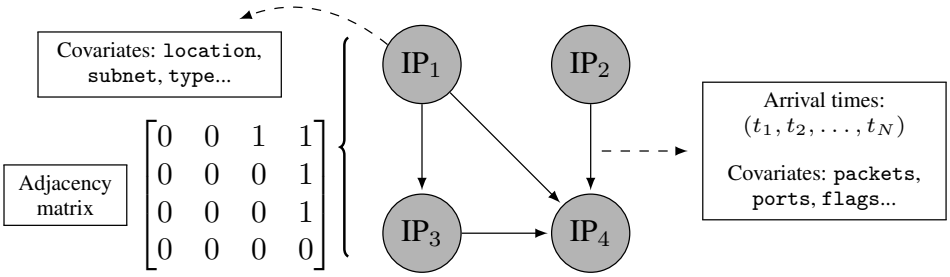


Figure 1.2.: *Toy example of a computer network graph formed by 4 IP addresses.*

where $N_{xy}(\cdot)$ is the *counting process* observed on each network edge, defined as

$$N_{xy}(t) = \sum_{i \geq 1} \mathbb{1}_{[0,t]}(t_i)\mathbb{1}_x(x_i)\mathbb{1}_y(y_i). \tag{1.2}$$

The corresponding adjacency matrix $\mathbf{A}_t$ at time t is obtained by setting $A_{ijt} = \mathbb{1}_{E_t}\{(i,j)\}$ for all $i, j \in V$. Computer networks can be efficiently summarised as dynamic networks, but there are also additional characteristics that make them more complex. For example, each connection event has additional associated information. In network flow data, described in Section 2.2, additional features are represented by the number of bytes and packets exchanged, protocol, ports and duration. Also, each node might have a set of associated covariates. For example, for IP addresses, those could be represented by the location, subnet or type of machine. A toy example of the structure of a computer network is given in Figure 1.2, which extends the example in Figure 1.1b.

1.2. Structure of the thesis

This section briefly describes the structure of this thesis, and the contributions and content of each chapter. More detailed descriptions about the methodological contributions in this work are given in the conclusion of each chapter.

Chapter 2 gives a description the computer network data used in this thesis: the Los Alamos National Laboratory user-authentication data, and the Imperial College London network flow data. Specific features of those large scale networks motivate many of the methodologies proposed in this work.

After the introduction to computer network data, the thesis is divided into three main parts, representing three different tasks in computer network modelling: *scoring individual events*, *graph clustering*, and *link prediction*. Each part is divided into two chapters, and each chapter will be concluded with a discussion on the strengths and limitations of the proposed methodologies. In Part I, two novel models for scoring individual events are proposed, tackling two important problems in computer networks: modelling network evolution, and separation of human and

automated activity.

Chapter 3 describes a novel Bayesian nonparametric model for the evolution of dynamic networks based on the Pitman-Yor process. This model explicitly admits power-laws in the number of connections on each edge, often present in real world networks, and, for careful choices of the parameters, power-laws for the degree distribution of the nodes. An empirical method for the estimation of the hyperparameters of the Pitman-Yor process is proposed, and some necessary corrections for uniform discrete base distributions are carefully addressed. The methodology is tested in an anomaly detection study on the LANL enterprise computer network, and successfully detects connections from a red-team penetration test.

Chapter 4 examines sequences of arrival times which contain automated events, with the objective to separate polling and non-periodic activity. This is first achieved using a simple mixture model on the unit circle based on the angular positions of each event time on the p-clock, where p represents the main periodicity associated with the automated activity; this model is then extended by combining a second source of information, the time of day of each event. Efficient implementations exploiting conjugate Bayesian models are discussed, and performance is assessed on the network flow data collected at Imperial College London.

The second part of this thesis, Part II, focuses on a second task in network analysis: clustering of nodes, also known as community detection. In this thesis, spectral clustering methods are analysed, and novel methodologies to estimate the number of communities and the dimension of the latent space are proposed.

Chapter 5 proposes a novel Bayesian model for simultaneous and automatic selection of the appropriate dimension of the latent space and the number of blocks in spectral clustering. The model is based on a novel conjecture on the asymptotic distribution of estimates of the latent features obtained using spectral methods. Extensions to directed and bipartite graphs are also discussed. The model is tested on simulated and real world network data, showing promising performance for recovering latent community structure.

Chapter 6 extends the methodology proposed in Chapter 5, allowing for heterogeneous within-community degree distributions. The proposed method is based on a suitable transformation to spherical coordinates of latent features obtained using spectral methods, and on a novel modelling assumption in the transformed space. The model is then incorporated into the model selection framework for the number of communities and latent dimension proposed in Chapter 5. Results show improved performance over competing methods on simulated and real world computer network datasets.

The third part of this work, Part III, describes methods for link prediction in computer networks, building up from the modelling framework introduced in Part II.

Chapter 7 discusses statistical techniques for link prediction based on the popular random dot product graph. The link probabilities of links are obtained by fusing information at two stages:

spectral methods provide estimates of latent positions for each node, and time series models are used to capture temporal dynamics. In this way, traditional link prediction methods, usually based on decompositions of the entire network adjacency matrix, are extended using temporal information. The methods presented in the chapter are applied to a number of simulated and real world computer network graphs, showing promising results. Overall, the chapter provides valuable guidelines for practitioners on the performance for link prediction of different random dot product graph models.

Chapter 8 extends the popular Poisson matrix factorisation (PMF) model to include scenarios that are commonly encountered in cyber-security applications. Specifically, an extension is proposed to explicitly handle binary adjacency matrices and include known covariates associated with the graph nodes. A seasonal PMF model is also presented to handle seasonal networks. To allow the methods to scale to large graphs, variational methods are discussed for performing fast inference. The results show an improved performance over the standard PMF model and other statistical network models.

The thesis is concluded with a discussion in **Chapter 9**, summarising the main contributions of this work.

1.3. List of publications based on this work

Part of the work presented in this thesis has been accepted for publication in academic journals, in the form of research papers:

Sanna Passino, F. and Heard, N. A. (2019). "Modelling dynamic network evolution as a Pitman-Yor process". In: *Foundations of Data Science* 1(3), pp. 293–306.

Sanna Passino, F. and Heard, N. A. (2020a). "Bayesian estimation of the latent dimension and communities in stochastic blockmodels". In: *Statistics and Computing* 30(5), pp. 1291–1307.

Sanna Passino, F. and Heard, N. A. (2020b). "Classification of periodic arrivals in event time data for filtering computer network traffic". In: *Statistics and Computing* 30(5), pp. 1241–1254.

In particular, Chapter 3 is based on Sanna Passino and Heard (2019)[1], Chapter 4 on Sanna Passino and Heard (2020b)[2], and Chapter 5 on Sanna Passino and Heard (2020a)[3].

Three other articles have been submitted for publication and are currently under review:

[1] Reproduction of the work is permitted by the American Institute of Mathematical Sciences publishing agreement, clause 6: *"The Work may be reproduced by any means for educational and scientific purposes by the Author(s) without fee or permission, with the exception that reproduction by services that collect fees for delivery of documents may be licensed only by the Publisher."*

[2] The article is distributed in Open Access under the terms of the Creative Commons CC BY license, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

[3] The article is distributed in Open Access under the terms of the Creative Commons CC BY license, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Sanna Passino, F., Heard, N. A., and Rubin-Delanchy, P. (2020). "Spectral clustering on spherical coordinates under the degree-corrected stochastic blockmodel". In: *arXiv e-prints*. arXiv: 2011.04558.

Sanna Passino, F., Turcotte, M. J. M., and Heard, N. A. (2020). "Graph link prediction in computer networks using Poisson matrix factorisation". In: *arXiv e-prints*. arXiv: 2001.09456.

Sanna Passino, F., Bertiger, A. S., Neil, J. C., and Heard, N. A. (2020). "Link prediction in dynamic networks using random dot product graphs". In: *arXiv e-prints*. arXiv: 1912.10419.

In particular, Chapter 6 is based on Sanna Passino, Heard, and Rubin-Delanchy (2020), Chapter 7 on Sanna Passino et al. (2020), and Chapter 8 on Sanna Passino, Turcotte, and Heard (2020).

Additionally, most chapters have associated *Github* repositories with code to reproduce the results presented in this thesis:

i. `fraspass/pitman_yor` for Chapter 3;
ii. `fraspass/human_activity` for Chapter 4;
iii. `fraspass/sbm` for Chapter 5;
iv. `fraspass/dcsbm` for Chapter 6.

2

Computer network data

The methodologies proposed in this thesis will be mostly demonstrated on three datasets related to cyber-security. The two most recent versions of the user-authentication data released by the Los Alamos National Laboratory[1] will be used: LANL 2015 (Kent, 2016) and LANL 2017 (Turcotte, Kent, and Hash, 2018). Also, the network flow data collected at Imperial College London will be extensively analysed. In order to demonstrate the wider applicability of the proposed methodologies, some of the techniques presented in this thesis will be also applied on two additional publicly available datasets, related to transportation and communication networks:

i. the Santander cycles graph, a bike-sharing network in central London, part of Transport for London (TfL)[2], used in Chapters 5 and 7;
ii. the Enron graph, a network of email communications between employees of Enron Corporation[3], used in Chapter 5.

Since the main focus of this thesis is the application to cyber-security data, only the LANL user-authentication data and the Imperial College London NetFlows will be described in the next two sections. The discussion on the networks constructed from the Santander cycles and Enron graph is deferred to Sections 5.6.2 and 5.6.3.

The remainder of this chapter describes the three cyber-security data sources that motivate the methodologies proposed in this thesis. In particular, details will be given on how to obtain dynamic network structures from such data sources.

[1] The data are publicly available online at `https://csr.lanl.gov/data/` (last accessed on 5th October, 2020).
[2] The data are publicly available at `https://cycling.data.tfl.gov.uk/`, powered by TfL Open Data (last accessed on 5th October, 2020).
[3] The data are publicly available at `https://www.cs.cmu.edu/~./enron/` (last accessed on 5th October, 2020).

2.1. Los Alamos National Laboratory user-authentication data

2.1.1. LANL 2015: Comprehensive, Multi-Source Cyber-Security Events

The user-authentication dataset contains Windows-based authentication events from both individual computers and centralised active directory domain controller servers (Kent, 2016). An example list of 5 comma separated entries is reported in Example 2.1.

Example 2.1: *Example of user-authentication events from the LANL 2015 data*

```
1,C580$@DOM1,SYSTEM@C580,C580,C580,Negotiate,Service,LogOn,Success
1,C599$@DOM1,C599$@DOM1,C553,C553,?,Network,LogOff,Success
1,U1782@DOM1,U1782@DOM1,C2800,C2800,Negotiate,Batch,LogOn,Fail
1,C608$@DOM1,C608$@DOM1,C608,C467,?,?,TGS,Success
2,U6@DOM1,U6@DOM1,C606,C1065,Kerberos,Network,LogOn,Success
```

In this work, the entries of interest for constructing a graph are the following:

i. Entry 1: the time of the authentication event, recorded in seconds;
ii. Entry 4: the source computer originating the authentication event;
iii. Entry 5: the destination computer where the authentication event is terminating.

The graph is simply denoted as $\mathbb{G} = (V, E)$, where V is the set of *computers*, and $E \subseteq V \times V$ is the set of observed edges, such that $(i, j) \in E$ if a connection from computer i to computer j is observed. The adjacency matrix is obtained setting $A_{ij} = \mathbb{1}_E\{(i, j)\}$, $i, j \in V$. In total, the data contain $1{,}051{,}430{,}459$ events, involving $16{,}230$ source computers and $15{,}895$ destination computers, for a total of $17{,}684$ unique machines and $419{,}744$ unique observed edges. Interestingly, the data also contain $48{,}079$ records labelled as red-team events, resulting from a simulated intrusion on the nodes `C17693`, `C18025`, `C19932` and `C22409`. The dataset will be used on a network-wide anomaly detection study in Chapter 3, aimed at identifying the compromised source nodes.

2.1.2. LANL 2017: Unified Host and Network Data Set

The data contain authentication logs collected over a 90-day period from computers in the Los Alamos National Laboratory (LANL) enterprise network running a Microsoft Windows operating system (Turcotte, Kent, and Hash, 2018). Two example records in *JSON* format are displayed in Example 2.2.

Example 2.2: *Examples of user-authentication event from the LANL 2017 data*

```
{"UserName": "User865586", "EventID": 4624, "LogHost": "Comp256596", "LogonID": "0x5aa8bd4",
 "DomainName": "Domain001", "LogonTypeDescription": "Network", "Source": "Comp782342",
 "AuthenticationPackage": "Kerberos", "Time": 87264, "LogonType": 3}
{"UserName": "User044801", "EventID": 4768, "LogHost": "ActiveDirectory",
"DomainName": "Domain001", "Status": "0x0", "Source": "Comp445617", "Time": 86587}
```

Table 2.1.: *Host log event IDs considered in this thesis (based on Table 2 in Turcotte, Kent, and Hash, 2018).*

(a) *Authentication and process event IDs*

Event ID	Description
4624	An account was successfully logged on, see Logon types **(b)**
4625	An account failed to logon, see Logon types **(b)**
4688	Process start
4768	Kerberos authentication ticket was requested
4776	The domain controller attempted to validate credentials
4800	The workstation was locked
4801	The workstation was unlocked
4802	The screensaver was invoked
4803	The screensaver was dismissed

(b) *Logon types (events 4624, 4625)*

Logon type	Description
2	Interactive
3	Network
4	Batch
5	Service
7	Unlock
9	NewCredentials
10	RemoteInteractive
11	CachedInteractive

Table 2.2.: *Covariates and corresponding number of factor levels.*

(a) *Users*

Covariate code	Number of factor levels	Covariate code	Number of factor levels
0	300	3	2
1	417	4	3
2	334	5	8

(b) *Hosts*

Covariate code	Number of factor levels
0	308
1	19
2	408

From each authentication record, the following fields are extracted for analysis:

i. the time of the event (`Time`),
ii. the user credential that initiated the event (`UserName`),
iii. the computer where the authentication originated (`Source`),
iv. the computer the credential was authenticating to (most often `LogHost`).

Only a subset of event IDs (`EventID`), listed in Table 2.1, are considered. Two bipartite graphs are then generated: first, the network users and the computers from which they authenticate, which will be denoted *User – Source*; second, the same users and the computers or servers they are connecting to, denoted *User – Destination*. The two graphs will be analysed in Chapters 7 and 8.

As generic notation, let $\mathbb{G} = \{(V_1, V_2), E\}$ represent one of these bipartite graphs, where V_1 is the set of *users* and V_2 a set of *computers* (sometimes referred to as *hosts*). The set $E \subseteq V_1 \times V_2$ represents the observed edges, such that $(i, j) \in E$ if user $i \in V_1$ connected to host $j \in V_2$ in a given time interval. The set of edges E can be represented by the rectangular $|V_1| \times |V_2|$ binary adjacency matrix $\mathbf{A}$, where $A_{ij} = \mathbb{1}_E\{(i, j)\}$.

For each user and computer, a list of categorical covariates is also available: 6 covariates are available for the users, and 3 for the computers. In total, there are $K = 1{,}064$ factor levels available for the user credentials and $H = 735$ factor levels for the computers. The number of factor levels for each covariate is summarised in Table 2.2. Due to de-identification of the true underlying covariates, it is not possible to explore the physical meaning of such nodal features.

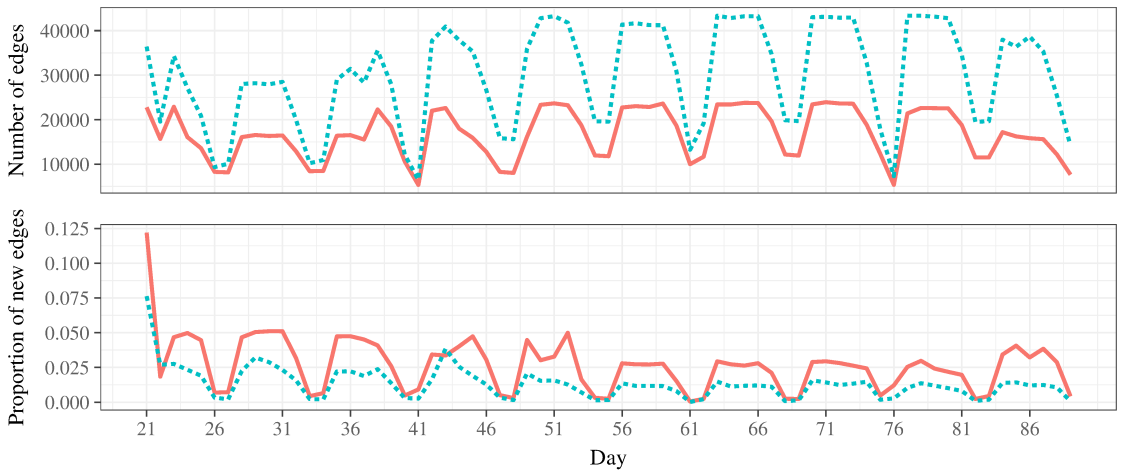


Figure 2.1.: *Number of links per day (top), and proportion of those that are new (bottom), after* 20 *days of observation of the LANL computer network.* ***Solid red*** *curve: User – Source.* ***Dashed blue*** *curve: User – Destination.*

As mentioned in Chapter 1, for cyber-security applications it would be valuable to accurately predict and assess the significance of new links. Importantly, many new links are formed each day as part of normal operating behaviour of a computer network; to demonstrate this, Figure 2.1 shows the the total number of edges formed each day and the proportion of those that are new for the two graphs, *User – Source* and *User – Destination*. Even though the relative proportion is small, this would still provide many more alerts than could be practically acted upon each day.

During the days 57 through 82, LANL conducted a red-team exercise, where the security team tested the robustness of the network by attempting to compromise other network hosts. All the red-team events were generated from the compromised node `Comp215429`, resulting in a total of 84 compromised edges in *User – Source*, and 132 in *User – Destination*.

2.2. Imperial College London NetFlow data

In this thesis, the network flow (NetFlow) data collected by the ICT services of Imperial College London will be extensively used, in particular in the chapters from 4 to 7. *NetFlow* is a network protocol originally developed by Cisco for collecting and monitoring data traffic *flows*. A flow is a set of packets passing through an observation point in the network, during a given time window (Claise, 2004). All the packets in the flow must share a set of common properties (Hofstede et al., 2014). Usually, the common characteristics are the protocol, the source and destination IP address, and the source and destination ports. The packets are aggregated into flows by a *NetFlow exporter*, for example a router or firewall, which periodically communicates with a *NetFlow collector*, where

the records are processed, parsed and stored. In short, *NetFlow records* essentially represent summaries of connections between IP addresses within a computer network, collected in bulk at internet routers. A detailed description about flow monitoring using NetFlow is given in Hofstede et al. (2014).

As an example, 5 randomly selected raw NetFlow records from the Imperial College London computer network are presented in Example 2.3. The true IP addresses have been anonymised, using randomly generated IPs.

Example 2.3: *Example of NetFlow records from the Imperial College data*

```
2|1581017227|264|1581034506|752|6|0|0|0|4087188370|57278|0|0|0|502274866|22|64580|64580|860|570|24|16|1|52
2|1581033920|512|1581033923|584|6|0|0|0|1337257956|56402|0|0|0|970825878|21|64580|64587|710|570|2|0|3|180
2|1581033996|288|1581033996|288|17|0|0|0|2340918866|63054|0|0|0|1564121766|53|64580|64583|856|540|0|0|1|62
2|1581034024|192|1581034054|400|6|0|0|0|1643182485|57589|0|0|0|2185303490|80|64580|786|710|540|27|0|51|2918
2|1581034055|168|1581034055|168|6|0|0|0|2508404747|49802|0|0|0|303496792|443|64580|64580|710|540|27|0|16|7405
```

For the purposes of this thesis, the most relevant entries are:

i. Entries from 2 to 5, encoding the time at which data packets is first and last seen in Unix time (number of milliseconds elapsed since 1st January, 1970);
ii. Entry 6, representing the protocol of the connection. For example, the value of 6 corresponds to the Transmission Control Protocol (TCP), and 17 to the User Datagram Protocol (UDP);
iii. Entries from 7 to 10 and from 12 to 15, representing the source and destination IP addresses in decimal form;
iv. Entries 11 and 16, corresponding to the source and destination ports.

In computer networks, it is normal to observe TCP bidirectional connections between a *client* and a *server* (for details about the TCP protocol, see, for example, Comer, 2014). The client requests permissions to access services from the server, and data flows are exchanged between the two devices in both directions. Those connections would appear as two separate NetFlow records. Therefore, for each pair of source and destination IPs, it is necessary to determine the correct direction of the connection. In this thesis, the direction is approximately determined from the source and destination ports in each NetFlow record: if one of the ports belongs to a list of well-known common service ports, then the corresponding machine is assigned the role of server. This is usually the case for more than 99% of the flows. The services associated with the ports in the list include, for example, secure shell (SSH, port 22) and web browsing (HTTP, port 80, and HTTPS, port 443). If both, or none of the ports belong to the list, then the server is chosen to be the IP address corresponding to the lowest port number.

Imperial College London has a range of 345,098 IP addresses (Metelli, 2018), but the recorded flows also include external connections, where one of the endpoints is not necessarily in the college network, but could be anywhere in the internet. This implies that the number of unique IP addresses observed in the data is much larger than the range of IP addresses assigned to the college. On

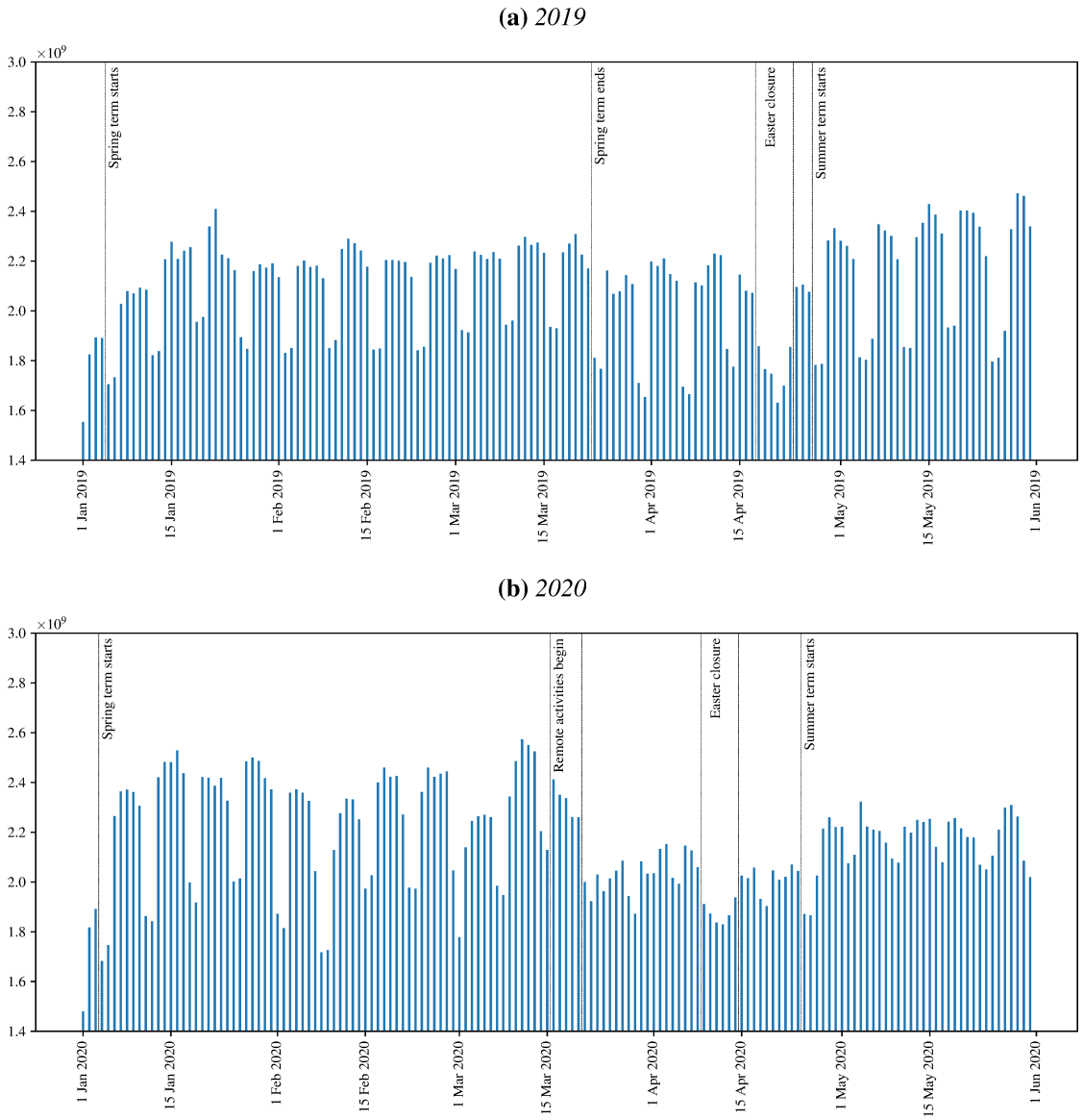


Figure 2.2.: *Number of flows per day in the Imperial College NetFlow data, between 1st January and 31st May in 2019 and 2020.*

average, approximately 2.4 billion flows are observed per day on a normal working day during term-time, and only 40% of those have both source and destination IPs belonging to the college IP ranges. Figure 2.2 shows the time series of the total number of flows observed daily between 1st January and 31st May, 2019 (Figure 2.2a), and in the same period in 2020 (Figure 2.2b). Clearly, both figures show seasonal effects of working days, term time, and Easter breaks. In particular, Figure 2.2b also shows an abrupt reduction of the number of flows during working days after the end of the spring term, corresponding to the transition to remote running of the university because

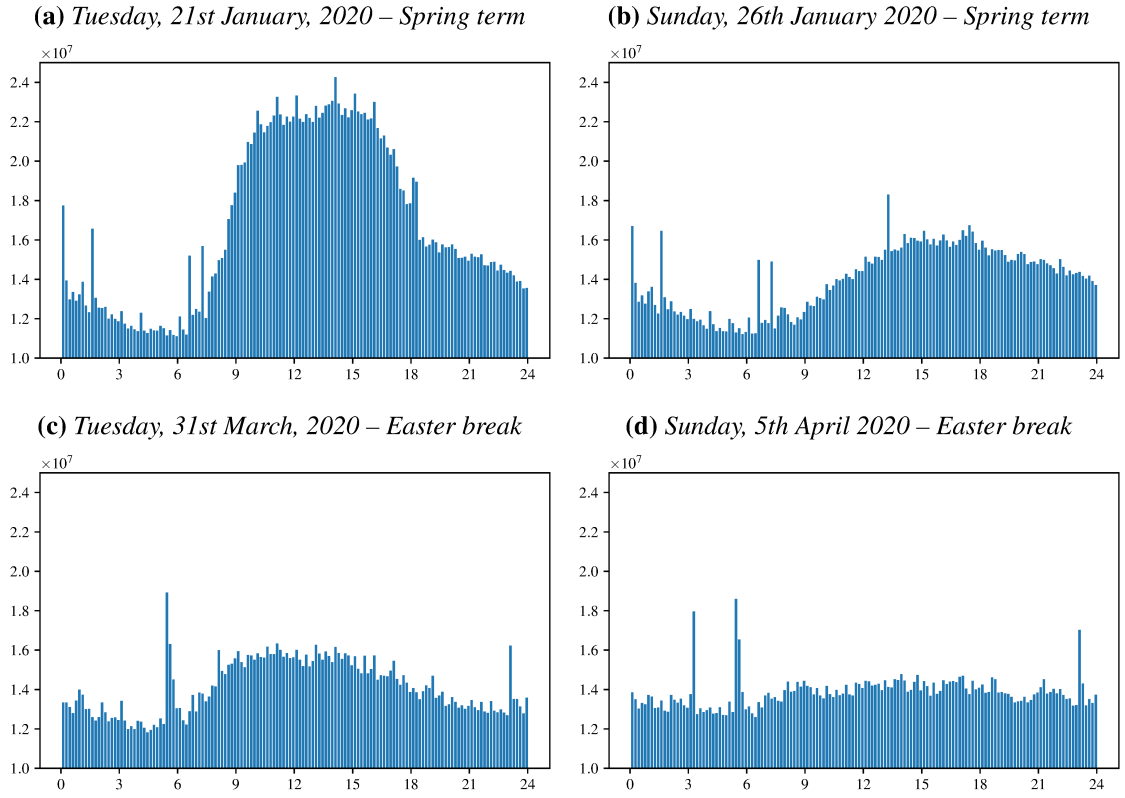


Figure 2.3.: *Daily histograms of the number of flows on 4 selected dates, mixed between working and non-working days, during term or break time. Bins have duration of 10 minutes each.*

of the COVID-19 pandemic. The activity on campus was minimal after approximately 18th March, 2020, but the number of daily flows still exceeded 2 billion on average until before the Easter closure, demonstrating that only a small part of the network activity is due to the presence of students and staff on campus. Dynamic models for network evolution need to be simple enough to scale to such a large number of daily observations. This is one of the motivations underlying the Pitman-Yor network-wide model proposed in Chapter 3.

It is also informative to analyse the daily distribution of the traffic. Figure 2.3 reports daily histograms for four selected dates. Figures 2.3a and 2.3c correspond to working days, whereas Figures 2.3b and 2.3d to weekends. Similarly, Figures 2.3a and 2.3b refer to term-time dates, whereas Figures 2.3c and 2.3d to the Easter break. Comparing Figures 2.3a and 2.3c confirms that the activity during working hours significantly decreased when college students and staff were required to work from home. Despite evident effects of working hours on the volume of traffic, the network clearly exhibits a significant proportion of automated and periodic activity, with frequent spikes and bursts. This motivates Chapter 4, which proposes a novel model for separation of human and periodic activity on network edges.

Part I.

Models for individual events

In this part of the thesis, two models will be proposed for scoring individual connection events within a dynamic network. Chapter 3 presents a Bayesian nonparametric network-wide model for the sequences of observed connections, which can be used to detect anomalous nodes. Chapter 4 deals with the problem of detecting periodicity and separating human and automated activity within network edges, proposing a novel Bayesian model for classification of periodic arrival times.

3

Modelling dynamic network evolution using the Pitman-Yor process

As discussed in Chapter 1, dynamic networks can be interpreted as a marked stochastic process $(\mathcal{T}, \mathcal{X}, \mathcal{Y})$. Under this representation, principled modelling of the observed sequences of marks $\{(x_i, y_i)\}_{i\geq 1}$, representing source and destination computers, is important for the purposes of network monitoring and identification of malicious attacks. This is a particularly challenging task, especially because of the large data volumes involved (see Chapter 2), leading to a tradeoff between mathematical complexity and computational feasibility. Bayesian nonparametric models, in particular the Dirichlet process, have been successfully used in Heard and Rubin-Delanchy (2016) for network-wide anomaly detection on computer network data. In Dirichlet processes, the distribution of the frequencies of the elements in the sequence presents an exponential decay, which is often unrealistic in many real world applications. For example, in natural language, the distribution of word frequencies tends to be fat-tailed, and it often exhibits a power law behaviour (Barabási, 2005). A random variable Z has a power law distribution, $Z \sim \text{PL}(\gamma)$, for some $\gamma > 1$, if $p(z) \propto z^{-\gamma},\ z \geq 1$; typically, $2 \leq \gamma \leq 3$ in real world applications (Newman, 2005). Power laws are also frequently observed in the degree distributions of computer networks, or on the frequencies of connections from or to a given node. For example, an illustration is given in Figure 3.1 for the LANL 2015 network (data described in Section 2.1.1). The curves in Figures 3.1a and 3.1b are better approximated by a power law than an exponential distribution.

This chapter examines a joint network-wide generative model for the sequence of observations $(x_1, y_1), (x_2, y_2), \ldots \in V \times V$, based on the Bayesian nonparametric Pitman-Yor process (Pitman and Yor, 1997; Ishwaran and James, 2001). The Pitman-Yor process, also known as the two-parameter Poisson-Dirichlet process, provides a natural extension of the Dirichlet process (Ferguson, 1973) where the probability of observing infrequently observed events can be further

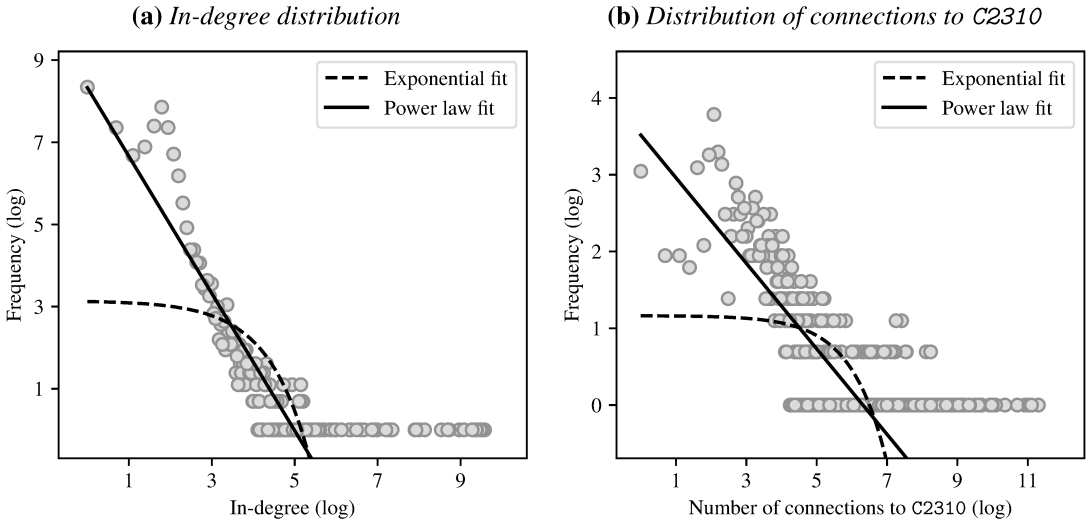


Figure 3.1.: *LANL 2015 computer network: in-degree distribution of the destination nodes, and frequency distribution of the sources connecting to the destination* `C2310`*, with exponential and power law fit obtained by weighted least squares regression on the log-log scale.*

discounted. The Pitman-Yor process has been successfully used in natural language processing and computational linguistics applications (Goldwater, Griffiths, and Johnson, 2005; Teh, 2006) to appropriately model the frequency of words in languages. The model admits power law structures for the frequencies of clients connecting to a given server, and for the network in-degree and out-degree distributions. The power law is explicitly modelled by the interplay between a strength parameter α and a discount parameter $\beta \in [0, 1)$.

In this chapter, the work of Heard and Rubin-Delanchy (2016) is extended in multiple directions. First, a more flexible model for network evolution, the Pitman-Yor process, is used to model the sequence of observed connections. Second, a reliable parameter estimation procedure for the parameters is proposed. Third, corrections for discrete uniform base distributions are discussed. Finally, the performance of the model on a network-wide anomaly detection study is analysed, using a number of different p-value combiners.

The model proposed in this chapter could be classified as a model for sequences of categorical observations, embedded within a dynamic network framework. Statistical modelling of categorical sequences is a well established branch of statistics and natural language processing, with relevant applications in speech recognition. Common methods mainly include n-gram models, based on n-order Markov assumptions (see the survey of Jurafsky et al., 2014), neural models, in particular recurrent neural networks (Bengio et al., 2003; Mikolov et al., 2010), maximum entropy or exponential models (Rosenfeld, 1996) and positional models (Lv and Zhai, 2009). Models for arrival times on dynamic networks are also extensively studied in the literature (see, for example, Perry and Wolfe, 2013; Matias and Miele, 2017). The model presented in this chapter is much

simpler, and only uses the arrival times to appropriately order the sequences, but it is well-suited for an efficient network-wide implementation on networks with large numbers of nodes and high frequency activity, and allows to directly obtain a predictive probability for a link.

The remainder of the chapter is organised as follows: Section 3.1 describes the Pitman-Yor process and a novel modelling framework for network evolution. Section 3.2 proposes a parameter estimation procedure, followed by a discussion on corrections for uniform discrete base measures in Section 3.3. Section 3.4 briefly analyses the types of degree-distributions arising from the proposed model, and Section 3.5 describes a network-wide anomaly detection framework. Finally, Section 3.6 presents the results on simulated data and on a real world computer network.

3.1. Modelling sequences using the Pitman-Yor process

Consider an infinitely exchangeable sequence of nodes $x_1, x_2, \ldots \in V$. After observing n events $\boldsymbol{x}_n = (x_1, \ldots, x_n)$, let $(x_1^\star, \ldots, x_{K_n}^\star)$ be the K_n unique observations in $\boldsymbol{x}_n$. Let F_0 be a *base* probability distribution on V. Then the distribution of observed nodes is a Pitman-Yor process $\text{PY}(\alpha, \beta, F_0)$, if the predictive distribution for the next observed node is

$$p(x_{n+1}|\boldsymbol{x}_n) = \frac{\alpha + \beta K_n}{\alpha + n} F_0(x_{n+1}) + \sum_{j=1}^{K_n} \frac{N_{jn} - \beta}{\alpha + n} \delta_{x_j^\star}(x_{n+1}), \tag{3.1}$$

where $\beta \in [0, 1)$ is a *discount* parameter, $\alpha > -\beta$ is a *strength* parameter, $\delta_v(u) = 1$ if $u = v$, 0 otherwise, and $N_{jn} = \sum_{i=1}^{n} \delta_{x_i}(x_j^\star)$ is the number of occurrences of the value $x_j^\star$ in $\boldsymbol{x}_n$. Note that $\beta = 0$ corresponds to the more familiar Dirichlet process (Ferguson, 1973), $\text{PY}(\alpha, 0, F_0) \equiv \text{DP}(\alpha, F_0)$.

The dynamics of the process can be most easily understood using the Chinese Restaurant metaphor (Aldous, 1985), which will be extensively used in this chapter. Starting from an empty restaurant with a potentially infinite number of tables, the first customer sits at the first table and then for $n = 1, 2, \ldots$, the $(n+1)$-th customer either sits at a new table or chooses an already occupied table with the following probabilities:

$$\mathbb{P}(\text{new table}) = \frac{\alpha + \beta K_n}{\alpha + n}, \qquad \mathbb{P}(j\text{-th table}) = \frac{N_{jn} - \beta}{\alpha + n},$$

for $j = 1, \ldots, K_n$. In this metaphor, N_{jn} is the number of customers sitting at the j-th table when n customers are in the restaurant. If each table is associated with a random dish drawn from a non-atomic base distribution F_0, the sequence $x_1, x_2, \ldots$ corresponding to the dishes eaten by successive customers is an exchangeable stochastic process with De Finetti measure (Ishwaran and James, 2001) $\text{PY}(\alpha, \beta, F_0)$. Figure 3.2 presents a cartoon representation of this metaphor for the Pitman-Yor process.

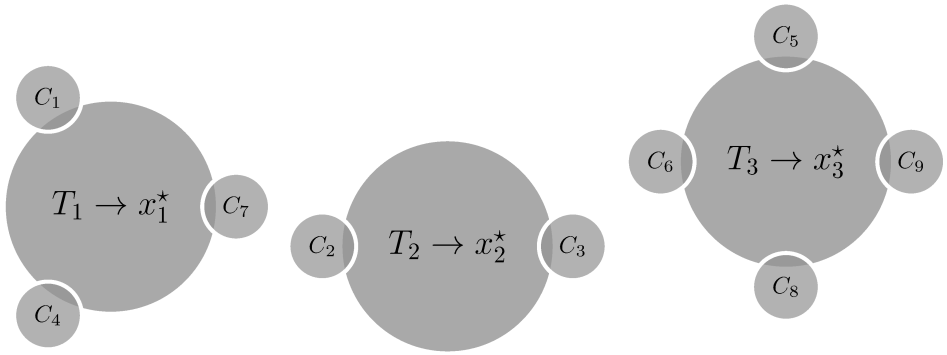


Figure 3.2.: *Cartoon example of the Chinese Restaurant metaphor for the Pitman-Yor process, where C_j represents the jth customer and $x_j^\star$ is the unique dish served at table T_j. The vector $\boldsymbol{x}_n = (x_1, \ldots, x_n)$ denotes the observed sequence of dishes eaten by each customer. Customer C_i being seated at table T_j is denoted $C_i \to T_j$. Then, for example, the tenth customer will sit at T_1 with probability $(3 - \beta)/(\alpha + 9)$.*

Denoting the sorted sequence of table occupancies $N_{(1n)}, \ldots, N_{(K_n n)}$, where $N_{(jn)} \geq N_{(j'n)}$ if $j > j'$, the Pitman-Yor process has the following property for $\beta \neq 0$ (Proposition 17, Pitman and Yor, 1997):

$$\mathbb{E}\left(\lim_{n\to\infty} \frac{N_{(jn)}}{n}\right) \approx \frac{\Gamma((1+\alpha)/\beta + 1)\Gamma(1-\beta)^{-1/\beta}}{(1+\alpha)\Gamma(\alpha/\beta + 1)} j^{-1/\beta}, \tag{3.2}$$

for j large. The expectation in (3.2) defines a power law distribution for the table sizes, which is an attractive property in many real world applications. On the other hand, if $\beta = 0$, corresponding to a Dirichlet process with concentration parameter α, the expectation in (3.2) is approximately $\exp(-j/\alpha)/\alpha$, obtained using the stick-breaking representation of the Dirichlet process (Sethuraman, 1994). The exponential decay, as demonstrated in Figure 3.1, might not be entirely suitable for the application to computer networks.

In this chapter, for an observed sequence of links $(x_1, y_1), \ldots, (x_N, y_N) \in V \times V$, it is assumed that the joint distribution has the following hierarchical structure:

$$\begin{aligned} x_i | y_i &\sim F_{x|y_i},\ i = 1, 2 \ldots, N, \\ y_i &\overset{iid}{\sim} G,\ i = 1, 2 \ldots, N, \\ F_{x|y} &\sim \text{PY}(\alpha_y, \beta_y, F_0),\ y \in V, \\ G &\sim \text{PY}(\alpha_0, \beta_0, G_0), \end{aligned} \tag{3.3}$$

where $\{F_{x|y}\}$ and G are unknown probability mass functions on the node set V. The probability of obtaining a link (x, y) is decomposed in two parts: $p(x, y) = p(x|y)p(y)$. The sequence of destination nodes $y_1, \ldots, y_N \in V$ is assumed to be exchangeable with a Pitman-Yor hierarchical distribution. Similarly, conditional on the destination node $y \in V$, it is assumed that the sequence of

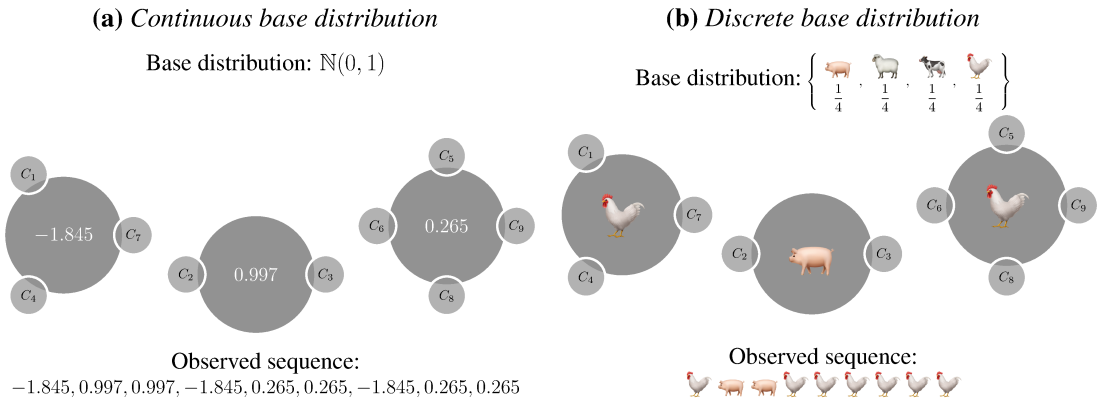


Figure 3.3.: *Toy example of observed sequences obtained using the Chinese Restaurant metaphor for continuous and discrete base distributions. For the customer allocations in Figure 3.2, the dishes $x_j^\star$ served at each table are sampled from a continuous and discrete distribution.*

source nodes $x_1, x_2, \ldots \in V$ connecting to y is again exchangeable with a Pitman-Yor hierarchical distribution, with parameters depending on the specific value of y. For the application to computer networks, the decomposition $p(x, y) = p(x|y)p(y)$ is preferred over $p(x, y) = p(y|x)p(x)$: for identifying red-team behaviour, anomalous client computers must be selected, hence each event is first given a score measuring how surprising the server found the connection from the corresponding client, and then all such measures are combined for a given client.

Note that care is needed when the base distribution of the process is atomic. In this case, using the Chinese Restaurant metaphor, it is possible that two or more tables serve the same dish, which means that the same draw from the base distribution is associated with multiple tables. Hence, the predictive probability (3.1) cannot be identified, since the underlying K_n, and consequently N_{jn}, are not determinable from $\boldsymbol{x}_n$. In contrast, with continuous base distributions, the draws are distinct with probability 1. The two scenarios are pictorially described in Figure 3.3: in Figure 3.3a, the draws are from a standard normal distribution and therefore distinct, whereas Figure 3.3b shows a restaurant that only serves 4 types of meat, which causes identifiability issues in the observed sequence.

Attention in the literature is mostly devoted to non-atomic base distributions. For the case of atomic F_0, Buntine and Hutter (2010) suggest introducing latent *multiplicities* and *table indicators*, corresponding to the number of tables contributing to the total number of times a value $x_j^\star$ is observed. An efficient sampler for this representation is derived in Chen, Du, and Buntine (2011). In this chapter, the node set will be assumed to be countable, and simple adaptation techniques will be used in the estimation of the parameters to take this problem into account.

3.2. Empirical estimation of the hyperparameters

Consider a Pitman-Yor process with non-atomic base distribution and $0 \leq \beta < 1$, and let K_n be the number of occupied tables after n customers have entered the restaurant.

Proposition 3.1 (Pitman (2002))**.** For a Pitman-Yor process $\text{PY}(\alpha, \beta, F_0)$, where F_0 is non-atomic, the expected number $\mathbb{E}(K_n)$ of occupied tables after n customers have entered the restaurant is:

$$\mathbb{E}(K_n) = \frac{\Gamma(n+\alpha+\beta)\Gamma(1+\alpha)}{\beta\Gamma(n+\alpha)\Gamma(\beta+\alpha)} - \frac{\alpha}{\beta}. \tag{3.4}$$

Proof. The result is obtained using the predictive distribution in (3.1), which gives a recursive expression for $\mathbb{E}(K_n)$:

$$\mathbb{E}(K_{n+1}) = \frac{\alpha + \beta\mathbb{E}(K_n)}{\alpha + n} + \mathbb{E}(K_n). \tag{3.5}$$

From (3.5), the equation for $\mathbb{E}(K_{n+1})$ can be rewritten as a first-order difference equation with variable coefficients $\mathbb{E}(K_{n+1}) = a(n)\mathbb{E}(K_n) + b(n)$, where $a(n) = (n+\alpha+\beta)/(n+\alpha)$ and $b(n) = \alpha/(n+\alpha)$. Given the initial condition $\mathbb{E}(K_0) = 0$, the equation has solution

$$\mathbb{E}(K_n) = \sum_{k=0}^{n-1} \frac{\alpha}{\alpha+k} \prod_{j=k+1}^{n-1} \frac{j+\alpha+\beta}{j+\alpha} = \frac{\alpha\Gamma(n+\alpha+\beta)}{\Gamma(n+\alpha)} \sum_{k=0}^{n-1} \frac{\Gamma(\alpha+k)}{\Gamma(\alpha+\beta+k+1)}, \tag{3.6}$$

which is obtained using the properties of the gamma function. The summation in (3.6) is simplified using the beta function $\text{B}(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$ and the properties of the geometric series:

$$\begin{aligned}\sum_{k=0}^{n-1} \frac{\Gamma(\alpha+k)}{\Gamma(\alpha+\beta+k+1)} &= \sum_{k=0}^{n-1} \frac{\text{B}(\alpha+k, \beta+1)}{\Gamma(\beta+1)} = \frac{1}{\Gamma(\beta+1)} \int_0^1 \left(\sum_{k=0}^{n-1} u^{\alpha+k-1} \right) (1-u)^{\beta} \text{d}u \\ &= \frac{1}{\Gamma(\beta+1)} \int_0^1 (u^{\alpha-1} - u^{\alpha+n-1})(1-u)^{\beta-1} \text{d}u = \frac{\text{B}(\alpha,\beta) - \text{B}(\alpha+n,\beta)}{\Gamma(\beta+1)}.\end{aligned}$$

Substituting the above expression in (3.6) gives the result in (3.4). □

For large n, Stirling's approximation could be used to obtain $\Gamma(n+\alpha+\beta)/\Gamma(n+\alpha) \approx n^{\beta}$ and approximate the expectation (3.4). Furthermore, for $\beta = 0$, Korwar and Hollander (1973) prove that $\lim_{n\to\infty} \mathbb{E}(K_n)/\log(n) = \alpha$. Combining the two cases, the expected value of K_n can be approximated for n large as:

$$\mathbb{E}(K_n) \approx \begin{cases} \alpha \log(n) & \text{if } \beta = 0, \\ \dfrac{\Gamma(1+\alpha)n^{\beta}}{\beta\Gamma(\beta+\alpha)} - \dfrac{\alpha}{\beta} & \text{if } \beta > 0. \end{cases} \tag{3.7}$$

Similarly, the expectation of the number of tables of size m after observing n customers enter,

H_{mn}, can be approximated for large n (Wallach et al., 2010) as:

$$\mathbb{E}(H_{mn}) \approx \frac{\Gamma(1+\alpha)n^{\beta}}{\Gamma(\beta+\alpha)m!}\prod_{j=1}^{m-1}(j-\beta). \tag{3.8}$$

Therefore, after N observations, an estimate $(\hat{\alpha}, \hat{\beta})$ of the hyperparameters α and β is obtained by solving the following non-linear system of equations:

$$H_{1N} = \frac{\Gamma(1+\hat{\alpha})N^{\hat{\beta}}}{\Gamma(\hat{\beta}+\hat{\alpha})}, \qquad K_N = \frac{H_{1N}-\hat{\alpha}}{\hat{\beta}}. \tag{3.9}$$

The estimation procedure in (3.9) is inspired by the method of moments (see, for example, Casella and Berger, 2002), which matches expectations to their sample counterparts.

A further estimate could be based on an alternative approximation of $\mathbb{E}(K_n)$, often used in the literature (Pitman, 2002; Wallach et al., 2010), obtained by noting that $\lim_{n\to\infty} n^{\beta} = \infty$ when $\beta > 0$. Therefore, when N is large, the value of $\Gamma(1+\alpha)n^{\beta}/\beta\Gamma(\beta+\alpha)$ will dominate α/β in (3.7), and hence

$$\hat{\beta} \approx \frac{H_{1N}}{K_N}. \tag{3.10}$$

The estimate (3.10) is particularly illuminating: asymptotically, β is the long term ratio between the tables with just one customer and the total number of tables. Under this approximation, from (3.9), $\hat{\alpha}$ can be obtained numerically as a solution to the equation

$$K_N = \frac{\Gamma(1+\hat{\alpha})N^{\hat{\beta}}}{\hat{\beta}\Gamma(\hat{\beta}+\hat{\alpha})}. \tag{3.11}$$

Otherwise, if $\hat{\beta} = 0$ then the process can be approximated by a Dirichlet process, and from (3.7),

$$\hat{\alpha} = \frac{K_N}{\log(N)}. \tag{3.12}$$

3.3. A correction for discrete uniform base distributions

Under the Pitman-Yor generative process, only the sequence of ordered dishes $\boldsymbol{x}_n = (x_1, \ldots, x_n)$ is observed. For discrete base distributions, the number of unique dishes, denoted here $\tilde{K}_n$, observed in $\boldsymbol{x}_n$ provides only a lower bound for the number of occupied tables K_n, since the same dish may be drawn from the base distribution multiple times. For example, in Figure 3.3b, $\tilde{K}_n = 2$, but $K_n = 3$. Similarly, the number of of dishes eaten by only one customer, denoted $\tilde{H}_{1n}$, provides only a lower bound for the number of single-customer tables, H_{1n}.

Assuming a uniform discrete base distribution on a sufficiently large set of nodes V, the approxi-

mations in (3.7) and (3.8) might be acceptable, but in this particular scenario it is also possible to use a simple correction. Conditioning on the true but unobserved number of draws from the base distribution (equivalently, the number of tables) K_n, a uniform base distribution implies that the expectation of the number of unique draws $\tilde{K}_n$ can be obtained as a generalisation of the *birthday problem* with $|V|$ days in a year:

$$\mathbb{E}(\tilde{K}_n|K_n) = |V| \left\{ 1 - \left(\frac{|V|-1}{|V|} \right)^{K_n} \right\}.$$

The quantity of interest is the expected number of distinct birthdays $\tilde{K}_n$, among K_n people, using a modified calendar with $|V|$ days. Substituting $\mathbb{E}(\tilde{K}_n|K_n)$ with $\tilde{K}_n$ and solving for K_n yields the approximation

$$\hat{K}_n = \log\left(1 - \frac{\tilde{K}_n}{|V|}\right) \log^{-1}\left(\frac{|V|-1}{|V|}\right). \tag{3.13}$$

Similarly,

$$\mathbb{E}(\tilde{H}_{1n}|H_{1n}, K_n) = H_{1n}\left(\frac{|V|-1}{|V|}\right)^{K_n-1}, \tag{3.14}$$

and hence the approximation

$$\hat{H}_{1n} = \tilde{H}_{1n}\left(\frac{|V|}{|V|-1}\right)^{\hat{K}_n-1}. \tag{3.15}$$

The expectation (3.14) is easily computed considering the following experiment: imagine K_n balls, H_{1n} of which are red, are placed at random into $|V|$ boxes. Then $\tilde{H}_{1n}$ is the the random variable corresponding to the number of boxes containing exactly one red ball, and no other balls.

The performance of (3.13) and (3.15) as rival estimates to $\tilde{K}_n$ and $\tilde{H}_{1n}$ will be assessed via simulations. Also, Figure 3.4 shows $\mathbb{E}\{\tilde{K}_n \mid \mathbb{E}(K_n)\}$ and $\mathbb{E}\{\tilde{H}_{1n} \mid \mathbb{E}(H_{1n}), \mathbb{E}(K_n)\}$ as functions of n for different values of the number of atoms $|V|$ in the discrete uniform base distribution. The plots demonstrate that, in expectation, $\tilde{K}_n$ and $\tilde{H}_{1n}$ largely deviate from $\mathbb{E}(K_n)$ and $\mathbb{E}(H_{1n})$ even for fairly large values of $|V|$, for example $|V| = 1{,}000$.

3.4. Power laws and Pitman-Yor dynamic graphs

In this section, some properties of the graph generated from distinct Pitman-Yor processes on each destination node are analysed. After an observation period $[0, T)$, it is possible to construct an adjacency matrix $\mathbf{A} \in \{0,1\}^{|V|\times|V|}$, where $A_{ij} = 1$ if the source node x_i connected to the destination node y_j at least once, and 0 otherwise. The row and column sums of $\mathbf{A}$ correspond to the out-degree and in-degree sequences $d_1^{\text{out}}, \ldots, d_{|V|}^{\text{out}}$ and $d_1^{\text{in}}, \ldots, d_{|V|}^{\text{in}}$, see (1.1). Real world graphs are commonly characterised by power law degree distributions. The in-degree and out-degree

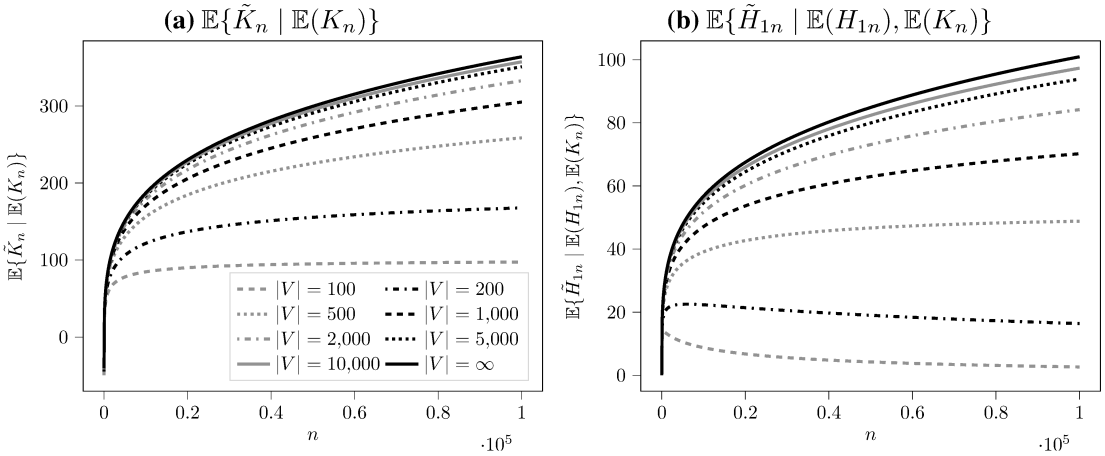


Figure 3.4.: *Plots of* $\mathbb{E}\{\tilde{K}_n \mid \mathbb{E}(K_n)\}$ *and* $\mathbb{E}\{\tilde{H}_{1n} \mid \mathbb{E}(H_{1n}), \mathbb{E}(K_n)\}$ *as functions of* n*, up to* $n = 100{,}000$*, for* $\alpha = 10$*,* $\beta = 0.25$ *and discrete uniform base distributions with different numbers of atoms.*

distributions in a graph generated from a Pitman-Yor process for each destination node can be controlled by the discount and strength parameters and the base distribution.

First, note that $d_j^{\text{in}} = \tilde{K}_{N_j}$, where N_j is the total number of connections to the destination node y_j. Furthermore, for a discrete uniform base distribution F_0 with a sufficiently large number of atoms $|V|$, $\tilde{K}_{N_j} \approx K_{N_j}$ holds. Hence, conditional on the parameters α and β of the process, $d_j^{\text{in}} \propto N_j^{\beta}$ in expectation from (3.7). Therefore, if $N_j \sim \text{PL}(\gamma)$, which holds for a suitable choice of α_0 and β_0 in (3.3), see (3.2), then approximately $d_j^{\text{in}} \sim \text{PL}\{(\gamma - 1 + \beta)/\beta\}$.

The out-degree distribution is slightly more difficult to control: it is highly dependent on the choice of the base distribution F_0. For a discrete uniform base distribution over V:

$$\mathbb{E}(d_i^{\text{out}} | K_{N_j}) = \sum_{j=1}^{|V|} \left\{ 1 - \left(\frac{|V| - 1}{|V|} \right)^{K_{N_j}} \right\},$$

which does not give a power law. In order to generate a distribution which resembles a power law, one can set a non-uniform discrete base distribution $F_0(x) = \pi_x \mathbb{1}_V(x)$, with $\sum_{x \in V} \pi_x = 1$ and $\pi_x \geq 0 \; \forall \; x \in V$, where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{|V|}) \sim \text{Dirichlet}(\boldsymbol{\theta})$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{|V|})$, and $\theta_x \overset{iid}{\sim} \text{PL}(\gamma), \; x \in V$. In this case:

Proposition 3.2. Under the Pitman-Yor network-wide process in (3.3), with non-uniform discrete base distribution $F_0(x) = \pi_x \mathbb{1}_V(x)$, with $\sum_{x \in V} \pi_x = 1$ and $\pi_x \geq 0 \; \forall \; x \in V$, where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{|V|}) \sim \text{Dirichlet}(\boldsymbol{\theta})$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{|V|})$:

$$\mathbb{E}(d_i^{\text{out}} | K_{N_j}, \boldsymbol{\theta}) = \sum_{j=1}^{|V|} \left\{ 1 - \frac{\Gamma(\sum_{i'} \theta_{i'}) \Gamma(K_{N_j} + \sum_{i' \neq i} \theta_{i'})}{\Gamma(\sum_{i' \neq i} \theta_{i'}) \Gamma(K_{N_j} + \sum_{i'} \theta_{i'})} \right\},$$

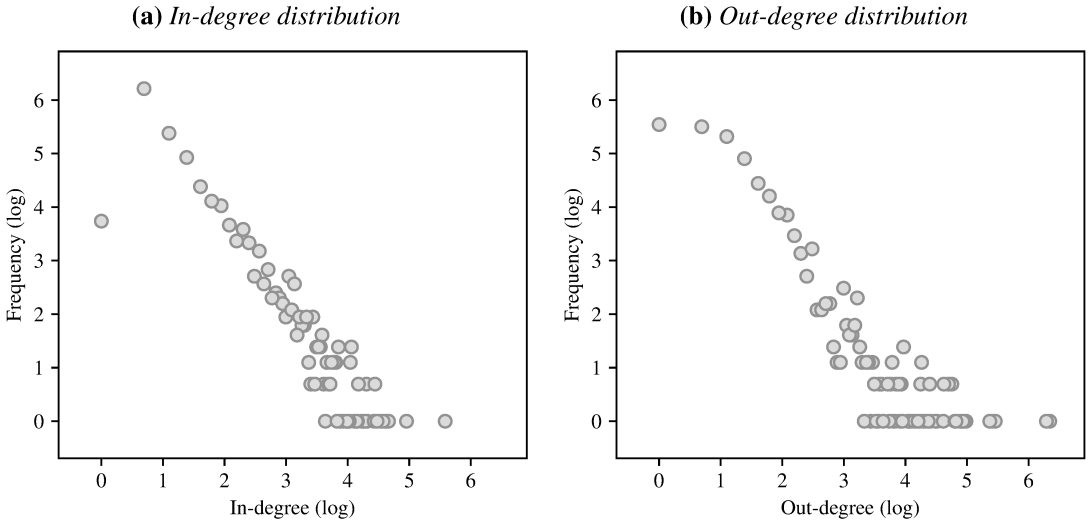


Figure 3.5.: *In-degree and out-degree distributions on the log-log scale, obtained from a single simulation of the Pitman-Yor network process with* $|V| = 1{,}500$*, simulating* $N_y \sim PL(1.5),\ y \in V$*,* $\theta_x \sim PL(2),\ x \in V$*, and fixing* $\alpha_y = 10$ *and* $\beta_y = 0.25\ \forall\ y \in V$*.*

A similar formula can be also derived for $\mathbb{E}(d_j^{\text{in}}|K_{N_j}, \boldsymbol{\theta})$.

Proof. The result is obtained as follows:

$$\begin{aligned}\mathbb{E}(d_i^{\text{out}}|K_{N_j}, \boldsymbol{\theta}) &= \sum_{j=1}^{|V|} \mathbb{E}(A_{ij}|K_{N_j}, \boldsymbol{\theta}) = \sum_{j=1}^{|V|} [1 - \mathbb{P}(A_{ij} = 0|K_{N_j}, \boldsymbol{\theta})] \\ &= \sum_{j=1}^{|V|} \left(1 - \int_0^1 \mathbb{P}(A_{ij} = 0|K_{N_j}, \boldsymbol{\theta}, \pi_i) p(\pi_i|\boldsymbol{\theta}) \mathrm{d}\pi_i\right), \end{aligned} \tag{3.16}$$

where $A_{ij} = 0$ denotes the event of x_i never connecting to y_j. Since $\boldsymbol{\pi} \sim \text{Dirichlet}(\boldsymbol{\theta})$, then $\pi_i \sim \text{Beta}(\theta_i, \sum_{j \neq i} \theta_j)$. Hence, the integral in (3.16) is:

$$\int_0^1 \mathbb{P}(A_{ij} = 0|K_{N_j}, \boldsymbol{\theta}, \pi_i) p(\pi_i|\boldsymbol{\theta}) \mathrm{d}\pi_i = \frac{\Gamma(\sum_j \theta_j)}{\Gamma(\theta_i)\Gamma(\sum_{j \neq i} \theta_j)} \int_0^1 \pi_i^{\theta_i - 1} (1 - \pi_i)^{K_{N_j} + \sum_{j \neq i} \theta_j - 1} \mathrm{d}\pi_i.$$

Evaluating the Euler integral in the equation above immediately gives the result. □

Hence, the distribution of d_i^{out} strongly depends on the interplay between N_j, which contributes to K_{N_j}, and $\boldsymbol{\theta}$, which means that it is controlled by α_0, β_0 and G_0 in (3.3). Figure 3.5 gives an example of the degree distributions generated by the Pitman-Yor network model under this framework, which strongly resemble power law distributions.

3.5. Calculating p-values for anomaly detection

For any given destination node $y \in V$, a p-value can be computed for each observed source node x_{n+1}, given the history $\boldsymbol{x}_n$. From the posterior predictive (3.1), the p-value p_{n+1} associated with the $(n+1)$-th connection to the destination node y is

$$p_{n+1} = \sum_{x \in V} p(x|\boldsymbol{x}_n) \mathbb{1}_{[0, p(x_{n+1}|\boldsymbol{x}_n)]}\{p(x|\boldsymbol{x}_n)\}. \tag{3.17}$$

The p-values (3.17) are discrete and stochastically larger than standard uniform random variables. For this reason, it can be preferable to consider *mid p-values* (Lancaster, 1952). Defining the stochastically smaller quantity

$$p^*_{n+1} = \sum_{x \in V} p(x|\boldsymbol{x}_n) \mathbb{1}_{[0, p(x_{n+1}|\boldsymbol{x}_n))}\{p(x|\boldsymbol{x}_n)\},$$

where the indicator function instead acts on the half-open interval $[0, p(x_{n+1}|\boldsymbol{x}_n))$ and thus sums all possibilities strictly less probable than the event observed, the mid p-value is given by

$$q_{n+1} = \frac{p_{n+1} + p^*_{n+1}}{2}. \tag{3.18}$$

In many problems, mid-p-values have been shown to outperform standard p-values, in particular in anomaly detection procedures in computer networks (Rubin-Delanchy, Heard, and Lawson, 2018).

Similarly, the p-value for the joint event (x_{n+1}, y_{n+1}), conditional on $(\boldsymbol{x}_n, \boldsymbol{y}_n)$, is

$$\sum_{x \in V, y \in V} p(x, y|\boldsymbol{x}_n, \boldsymbol{y}_n) \mathbb{1}_{[0, p(x_{n+1}, y_{n+1}|\boldsymbol{x}_n, \boldsymbol{y}_n)]}\{p(x, y|\boldsymbol{x}_n, \boldsymbol{y}_n)\}, \tag{3.19}$$

where $p(x_{n+1}, y_{n+1}|\boldsymbol{x}_n, \boldsymbol{y}_n) = p(x_{n+1}|y_{n+1}, \boldsymbol{x}_n) p(y_{n+1}|\boldsymbol{y}_n)$ by conditional independence assumptions. Note that $p(x_{n+1}|y_{n+1}, \boldsymbol{x}_n)$ only depends on the values of the source nodes x_i in the sequence $\boldsymbol{x}_n$ such that $y_i = y_{n+1}$. The calculation of (3.19) is intractable, but the p-value could be suitably approximated using a p-value combiner. Given a sequence of p-values $p_1, p_2, \ldots, p_\ell$, common choices for p-value combiners are (Heard and Rubin-Delanchy, 2018):

i. Fisher's method (Fisher, 1934): $S_F = -2 \sum_{i=1}^{\ell} \log(p_i) \sim \chi^2_{2\ell}$,
ii. Pearson's method (Pearson, 1933): $S_P = -2 \sum_{i=1}^{\ell} \log(1 - p_i) \sim \chi^2_{2\ell}$,
iii. Tippett's method (Tippett, 1931): $S_T = \min\{p_i\} \sim \text{Beta}(1, \ell)$,
iv. Stouffer's method (Stouffer et al., 1949): $S_S = \sum_{i=1}^{\ell} \Phi^{-1}(p_i) \sim \mathbb{N}(0, \ell)$, where $\Phi^{-1}(\cdot)$ is the inverse cumulative distribution function of a standard normal distribution.

For approximation of the p-value in (3.19), $\ell = 2$, with p_1 and p_2 the corresponding p-values for y_{n+1} and $x_{n+1}|y_{n+1}$.

3.6. Applications and results

The proposed model and estimation procedure have been validated and tested on synthetic networks. Furthermore, the goodness-of-fit of the Pitman-Yor process has been assessed on the LANL 2015 computer network (Section 2.1.1) released by the Los Alamos National Laboratory (Kent, 2016).

3.6.1. Estimation of the Pitman-Yor hyperparameters

The different techniques proposed for the estimation of the Pitman-Yor hyperparameters have been extensively compared using a simulation study, constructed as follows: $S = 10{,}000$ sequences of table allocations, each of length $N = 100{,}000$, have been simulated using the Chinese Restaurant metaphor of the Pitman-Yor process, setting $\alpha = 7$ and $\beta = 0.25$. For each of the S sequences, the tables have been assigned a label drawn at random, with replacement, from a discrete uniform distribution with $|V| = 1{,}000$ and $|V| = 16{,}230$ atoms (corresponding to the number of source machines in the LANL 2015 network). Matching the sequence of table allocations with the table labels gives a sample sequence from a Pitman-Yor process with uniform discrete base distribution F_0 with $|V|$ atoms. From the resulting sequence, the values of $\tilde{K}_N$ and $\tilde{H}_{1N}$ are calculated, and their corrected counterparts in (3.13) and (3.15), $\hat{K}_N$ and $\hat{H}_{1N}$. The values are compared with the true K_N and H_{1N}, available from the simulated table allocation. The parameter estimates $\hat{\alpha}$ and $\hat{\beta}$ obtained using the pair (3.10) and (3.11) are referred to as *Method 1* and (3.9) as *Method 2*; both are computed using either the uncorrected estimates $(\tilde{H}_{1N}, \tilde{K}_N)$ or the corrected pair $(\hat{H}_{1N}, \hat{K}_N)$, producing four different estimates of the parameters. Kernel density estimates across the simulations for each parameter and each estimation method are plotted in Figures 3.6 and 3.7. The kernel bandwidth choice is based on Silverman's rule of thumb (Silverman, 1986).

In Figure 3.6 and 3.7, the plots *(a)* and *(b)* show that parameter estimation based on the most popular approximation for $\mathbb{E}(K_n)$ in the literature, Method 1, gives biased results. On the other hand, when Method 2 is used to estimate the pair (α, d), and the corrections (3.13) and (3.15) are used, the results are excellent, and the proposed estimation procedure on average correctly recovers the exact values of the parameters used to simulate the data. When the correction is not applied, the performance of Method 2 is still fairly reliable for α, but performs poorly for estimating β. Also, the estimates obtained from the uncorrected values K_N and H_{1N}, in plots *(c)* and *(d)* can deviate significantly from the true values. The plots *(c)* and *(d)* in Figure 3.6 and 3.7 show again the importance of the correction for a small number of atoms in the base distribution. For $|V| = 16{,}230$, the observed value $\tilde{K}_N$ is on average close to the true K_N. This does not happen for $|V| = 1{,}000$, and the approximation becomes crucial.

Overall, the simulation confirms the reliability of the parameter estimation procedure based on (3.9) for a Pitman-Yor process. The performance of the proposed procedure is far superior to the method based on the estimates (3.10) and (3.11), which are obtained from the most common

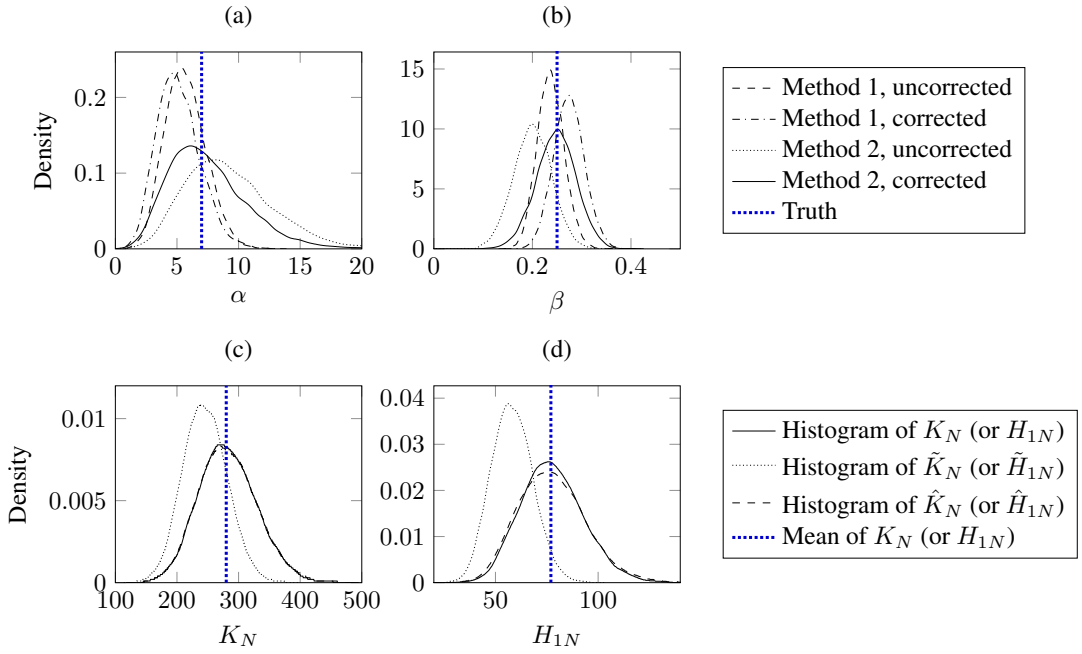


Figure 3.6.: *Kernel density estimates of the parameter estimates from 10,000 simulations,* $|V| = 1{,}000$.

approximation of $\mathbb{E}(K_n)$ used in the literature. Furthermore, for discrete uniform base distributions, the corrections proposed in equations (3.13) and (3.15) are shown to be highly beneficial for parameter estimation, especially when the number of atoms in the base distribution $|V|$ is not large.

3.6.2. Empirical assessment of the goodness-of-fit

In this section, the empirical estimation procedure of the hyperparameters and goodness-of-fit of the Pitman-Yor process is evaluated on selected nodes in the LANL 2015 network. As described in Section 2.1.1, the LANL 2015 dynamic network contains $1{,}051{,}430{,}459$ events, involving $16{,}230$ source computers and $15{,}895$ destination computers, for a total of $17{,}684$ unique machines and $419{,}744$ unique observed edges. The predictive p-values (3.17) and mid-p-values (3.18) were calculated for all the events. In order to assess the model fit, it is possible to examine the Q-Q plot of the p-values, which are approximately uniformly distributed in $(0, 1)$ under a correct model specification. Examples of the Q-Q plots observed for six destination nodes are plotted in Figure 3.8, obtained using the estimate (3.9) and the corrections (3.13) and (3.15). Let $V_S \subseteq V$ be the subset of nodes which are ever observed to initiate a connection. The base distribution F_0 of the Pitman-Yor process was chosen to be uniform on the set of source computers V_S: $F_0(x) = 1/16{,}230 \times \mathbb{1}_{V_S}(x)$. The estimated values of the parameters of the Pitman-Yor process, and additional summary statistics, are reported in Table 3.1. The Q-Q plots show, for some of the nodes, an excellent fit on real data under the strong assumptions made in this modelling framework (see plots for `C5716`, `U7`, `C2525`,

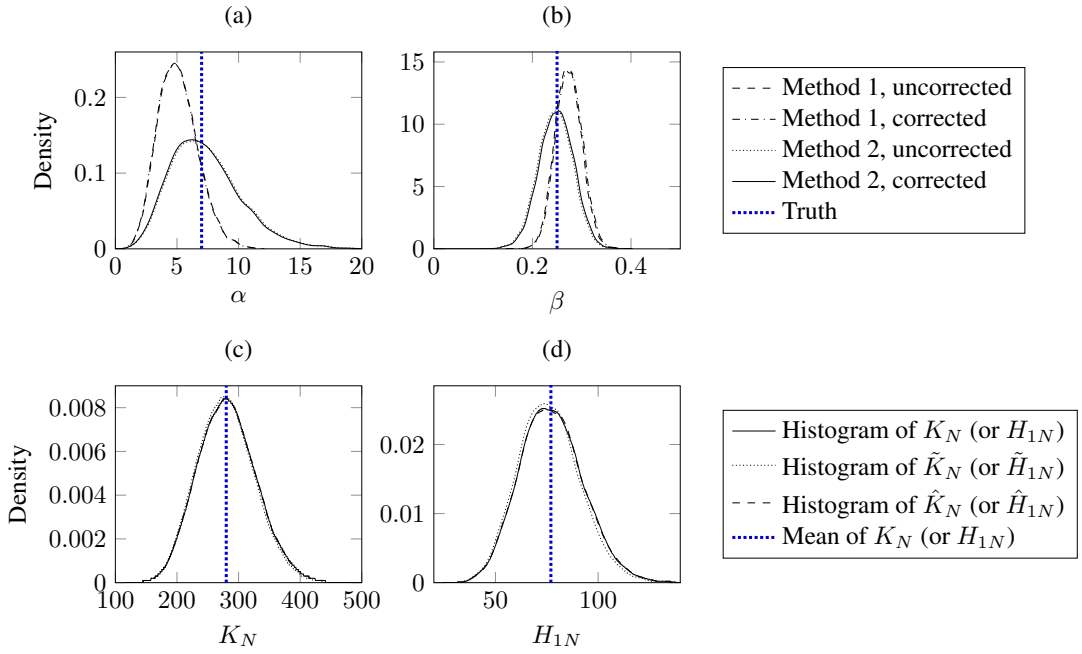


Figure 3.7.: *Kernel density estimates of the parameter estimates from 10,000 simulations,* $|V| = 16{,}230$.

Table 3.1.: *Estimated Pitman-Yor parameters for 6 destination nodes.*

Destination	N	$\tilde{K}_N$	$\hat{K}_N$	$\tilde{H}_{1N}$	$\hat{H}_{1N}$	$\hat{\alpha}$	$\hat{\beta}$
C5716	113,987	3,401	3,816.418	144	182.164	651.519	0.047
U7	138,286	2,700	2,952.989	350	419.819	302.093	0.142
C2525	204,532	1,555	1,634.571	97	107.272	183.319	0.066
C1877	342,766	5,095	6,114.758	226	329.390	866.520	0.054
C395	518,058	5,957	7,422.437	442	698.259	841.357	0.094
C423	2,426,512	2,705	2,958.988	166	199.188	230.040	0.067

and C1877). On the other hand, in some other cases (see plots for C395 and C423) the distribution of the p-values is not uniform, but follows the theoretical distribution only on the left tail.

It is also possible to compare the performance of the method for different choices of α and β. A destination node, C1438, is used as an example. The computer C1438 appears as destination in $N = 136{,}743$ connections, with $K_N = 394$ unique edges and $H_{1N} = 72$ source computers that connected only once, resulting in $\hat{\alpha} = 27.434$ and $\hat{\beta} = 0.113$ after solving (3.9) with corrections (3.13) and (3.15). The three plots in Figure 3.9 present the sequential values of $\tilde{K}_n$ and $\tilde{H}_{1n}$ and their ratio $\tilde{H}_{1n}/\tilde{K}_n$, corresponding corrected estimates, and average sample paths obtained from Pitman-Yor processes with a number of different values of the parameters.

It is immediately clear from the plots that the only sample path which correctly captures the behaviour of $\tilde{K}_n$, $\tilde{H}_{1n}$ and $\tilde{H}_{1n}/\tilde{K}_n$ simultaneously is the Pitman-Yor process with estimates $\hat{\alpha}$

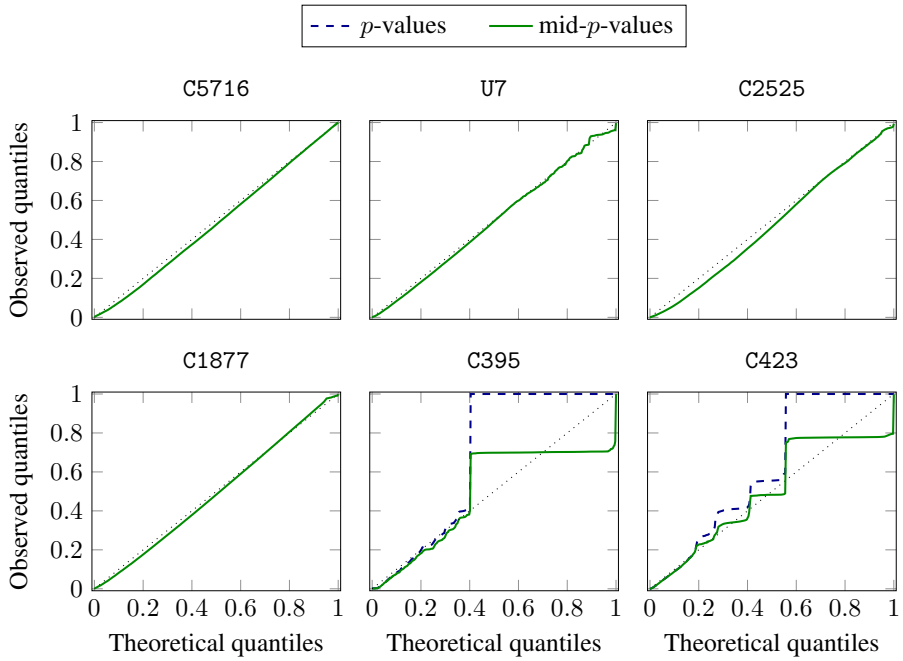


Figure 3.8.: *Uniform Q-Q plot for the Pitman-Yor process fitted to six destination nodes.*

and $\hat{\beta}$ obtained using Method 2, with or without correction for discrete F_0. The fit is excellent, and the observed trajectories almost coincides with the average estimates obtained from multiple simulations of the process. The Dirichlet process fails to track $\tilde{K}_n$ and $\tilde{H}_{1n}$ individually, and gives a slightly better performance when modelling the ratio $\tilde{H}_{1n}/\tilde{K}_n$. The Pitman-Yor process with estimates obtained from Method 1 only marginally tracks the data, reaching approximately the observed values $\tilde{K}_N$ and $\tilde{H}_{1N}$ only at the end of the process ($N = 136{,}743$). On the other hand, the ratio is not modelled in a satisfactory way.

Overall, the Pitman-Yor fit is far superior to the Dirichlet process, even without an optimal choice of the parameters, showing that in this case adding the discount parameter β is beneficial. The simulation empirically confirms that the Pitman-Yor process seems to be a suitable choice, but the parameters should be estimated carefully.

3.6.3. Network-wide anomaly detection

Let us suppose that $p_1, \dots, p_N$ are the p-values computed for the N events observed on a given edge $(x, y) \in E$, and $q_1, \dots, q_N$ are the corresponding mid-p-values. Those p-values can be obtained from the sequence $y_{n+1}|\boldsymbol{y}_n$, or from $x_{n+1}|y_{n+1}, \boldsymbol{x}_n$, or from a combination of the two p-values at the event level, as described in earlier sections. For this analysis, uniform base distributions F_0 and G_0 were used: $F_0(x) = 1/16{,}230 \times \mathbb{1}_{V_S}(x)$, and $G_0(x) = 1/17{,}684 \times \mathbb{1}_V(x)$.

Given the sequence of p-values, it is possible to combine the scores into a single distribution

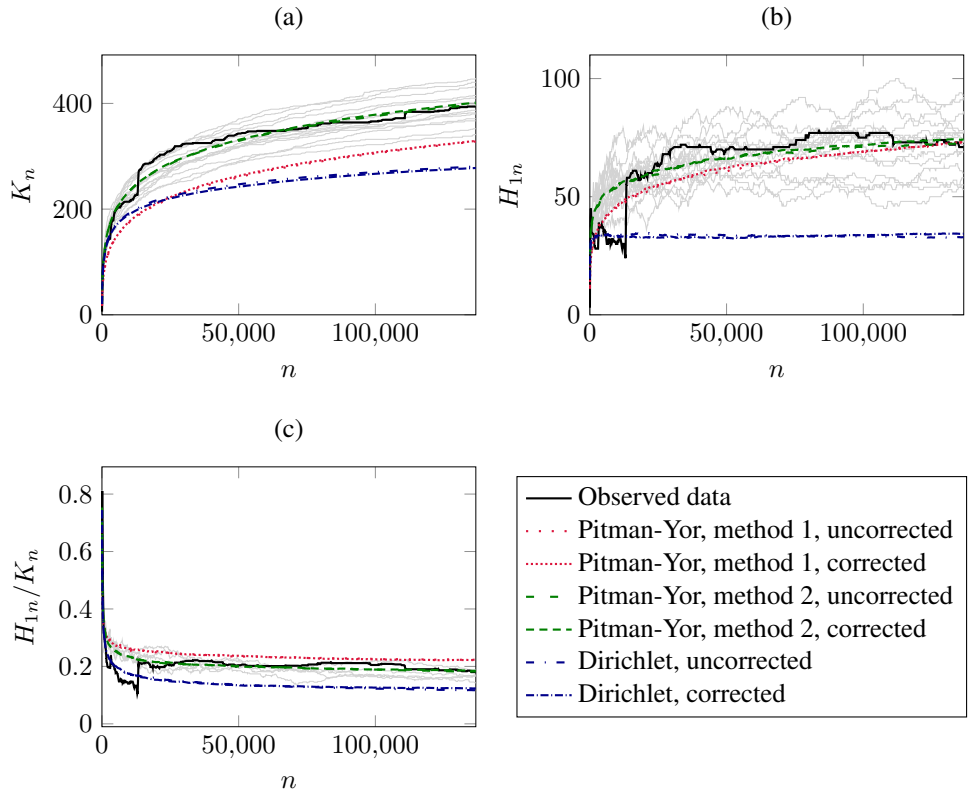


Figure 3.9.: *Plot of the corrected K_n (a), H_{1n} (b) and their ratio H_{1n}/K_n (c) as a function of n for the connections to the destination computer* `C1438`*, and averaged sample paths from* 100 *samples from a Pitman-Yor process, obtained using different estimates of the parameters. The grey lines correspond to 10 realised trajectories of simulated Pitman-Yor processes, obtained using the corrected estimate* (3.9)*.*

for each edge. Heard and Rubin-Delanchy (2016) suggest to use the minimum p-value method, or Tippett's method. Note that in this framework the distributional result is only approximate, because of the discreteness of the p-values. Then, the lower tail area of the Beta$(1, N)$ distribution evaluated at $\min\{p_1, \ldots, p_N\}$ gives the p-value p_{xy} associated with the edge (x, y).

Following Heard and Rubin-Delanchy (2016), $\tilde{E}_x$ is defined as the set of edges in the network graph with source node x on which connections have been observed: $\tilde{E}_x = \{(x, y) : y \in V \cap (x, y) \in E\}$. The p-values p_{xy} on each edge can be combined into a single score s_x for each source node using one of the combiners previously presented. For example, using the Pearson's combiner, under a normal behaviour of the network, the theoretical distribution of s_x is $s_x \sim \chi^2_{2|\tilde{E}_x|}$. Therefore, a p-value p_x for the source computer x is given by the left tail probability of the $\chi^2_{2|\tilde{E}_x|}$ distribution, given the observed s_x. From the list $\{p_x,\ x \in V_S\}$, where V_S is the set of source nodes, it is then possible to identify the most anomalous source computers by ranking the p-values. A similar procedure can be carried out sequentially using the mid-p-values $q_1, \ldots, q_N$, or different combiners at the event, edge or node level. The procedure is fully parallelisable and particularly

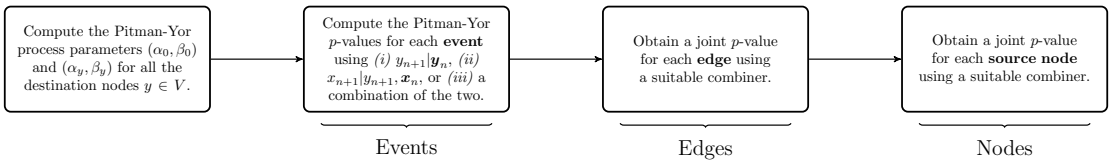


Figure 3.10.: *Flowchart describing the network-wide anomaly detection procedure.*

suitable for implementation on standard platforms for big data analysis like Hadoop MapReduce, and has been applied to the LANL 2015 data. A flowchart summarising the procedure is reported in Figure 3.10.

For each destination computer $y \in V$, α_y and β_y are estimated using (3.9) and the corrections (3.13) and (3.15), and similarly for α_0 and β_0. Table 3.2 reports the anomaly rankings of the compromised red-team computers (see Section 2.1.1), obtained using the minimum p-value method at the edge level and a number of different combiners at the event and node level.

Two of the compromised computers are consistently ranked among the top-1% most anomalous nodes. In particular, `C17693` is sometimes ranked as most anomalous machine overall. For the two remaining nodes, associated with more subtle activity within the network, improvements are achieved when using mid-p-values, but the malicious activity on these nodes is still difficult to detect. Overall, the results confirm the conclusions in Rubin-Delanchy, Heard, and Lawson (2018) about the improved detection performance when using mid-p-values. This is particularly evident in the case of the two least anomalous compromised machines: using mid-p-values, the ranking improves significantly. The activity associated with `C17693` is anomalous even when only the p-values associated with the sequence of destination nodes $y_1, y_2, \ldots$ are considered. For the other three machines, the rankings significantly improve when the edge activity from the p-values for $x_{n+1}|y_{n+1}$ is used for computing the anomaly scores. Using the combined scores at the event level does not significantly improve the results, showing that most of the malicious activity might be due to edge-related anomalies.

3.7. Conclusion and discussion

In this chapter, a model for events in dynamic networks was presented. The sequence of edges was modelled by using the Pitman-Yor process, a two-parameter extension of the Dirichlet process. Empirical methods for choosing the hyperparameters, based on large sample results, have been proposed and tested on the Los Alamos National Laboratory authentication data, showing excellent fit on the activity of real destination computers. Furthermore, corrections for discrete base distributions were carefully addressed. The Pitman-Yor process allows for more flexible and accurate modelling of the tails of the predictive distribution, which also implies more accurate modelling of the new links in the network. The model has been tested on synthetic data and applied

Table 3.2.: *Anomaly rankings for the four red-team source computers.*

		Event level score: Events $x_{n+1}\|y_{n+1}, \boldsymbol{x}_n$ only Standard p-values p_{n+1}			Event level score: Events $x_{n+1}\|y_{n+1}, \boldsymbol{x}_n$ only Mid-p-values q_{n+1}		
Edge level combiner: Tippett		Node level combiner: Fisher	Pearson	Stouffer	Node level combiner: Fisher	Pearson	Stouffer
Source computer	C17693	5	2	4	2	1	5
	C18025	138	75	78	151	74	105
	C19932	3831	8870	8877	3571	2754	3151
	C22409	3767	15773	15764	3450	6984	3756
		Events $y_{n+1}\|\boldsymbol{y}_n$ only Mid p-values q_{n+1}			Event level combiner: Tippett Mid-p-values q_{n+1}		
Edge level combiner: Tippett		Node level combiner: Fisher	Pearson	Stouffer	Node level combiner: Fisher	Pearson	Stouffer
Source computer	C17693	6	5	5	6	5	5
	C18025	2806	1536	1674	142	96	107
	C19932	5407	8882	8914	3813	2264	3232
	C22409	12126	15808	15878	3803	6516	4196
		Event level combiner: Fisher Standard p-values p_{n+1}			Event level combiner: Fisher Mid-p-values q_{n+1}		
Edge level combiner: Tippett		Node level combiner: Fisher	Pearson	Stouffer	Node level combiner: Fisher	Pearson	Stouffer
Source computer	C17693	3	5	5	5	2	5
	C18025	151	88	101	155	90	106
	C19932	6339	3818	4879	4937	3017	3996
	C22409	6120	14799	5379	4451	6695	5236

to the LANL 2015 authentication dataset, showing good performance in a network-wide anomaly detection study.

The Pitman-Yor model might be used as a building block for more complex models. For example, in computer networks, the arrival time of each event is also available, and suitable models for arrival times on each edge have been proposed in the literature (Price-Williams and Heard, 2019). The overall performance of the method is remarkable, since only two pieces of information are used in this chapter: the source and destination node. The methodology can be potentially extended to consider any additional information which is usually available for each event: for example, in computer network data, authentication type and logon type.

3.7.1. Limitations and future work

The equation for the predicted probability (3.1) gives equal weight to *all* previous observations, which could be a relevant drawback if the model is fitted over very long periods of time, when the network structure could have dynamically evolved. A more realistic modelling approach might

Table 3.3.: *Estimated Pitman-Yor parameters for 6 destination nodes after removing the periodic edges.*

Destination	N	$\tilde{K}_N$	$\hat{K}_N$	$\tilde{H}_{1N}$	$\hat{H}_{1N}$	$\hat{\alpha}$	$\hat{\beta}$
C5716	50,564	2,793	3,064.929	144	173.920	610.480	0.056
U7	19,387	2,279	2,455.688	350	407.150	481.421	0.165
C2525	8,766	829	850.895	97	102.214	167.921	0.120
C1877	91,557	3,368	3,774.724	226	285.162	622.574	0.075
C395	14,939	5,911	7,349.927	442	695.146	764.683	0.094
C423	15,769	1,619	1,705.504	166	184.382	367.639	0.108

give more weight to recent observations. The main advantage of the methodology proposed in this chapter is the simple parameter estimation procedure, which is particularly suitable for scaling the model to the entire network. Alternative strategies based on assigning uneven weights to different observations might increase the computational burden for parameter estimation and the calculation of predictive probabilities.

For the Dirichlet process, (3.12) has been used in this chapter for estimating α, using the approximation $\mathbb{E}(K_n) \approx \alpha \log(n)$ traditionally used in the literature (for example, Pitman, 2002; Wallach et al., 2010). The correct value of expectation is $\mathbb{E}(K_n) = \sum_{i=1}^{n} \alpha/(\alpha + i - 1)$, which follows from (3.5), setting $\beta = 0$. Using a Riemann sum argument, $\mathbb{E}(K_n)$ is approximated as:

$$\mathbb{E}(K_n) = \sum_{i=1}^{n-1} \frac{\alpha}{\alpha + i - 1} \approx \alpha \int_0^{n-1} \frac{1}{\alpha + u} \mathrm{d}u = \alpha \log\left(1 + \frac{n-1}{\alpha}\right) \approx \alpha[\log(n) - \log(\alpha)].$$

The equation shows that the approximation $\mathbb{E}(K_n) \approx \alpha \log(n)$, traditionally used in the literature, ignores the term $\alpha \log(\alpha)$, which could be relevant for parameter estimation. A similar scenario occurs for the Pitman-Yor process, where the estimates (3.10) and (3.11), obtained ignoring the term α/β in (3.7), are largely outperformed by (3.9).

Furthermore, a large majority of the traffic observed on computer networks can be ascribed to automated polling activity. This clearly violates the assumption of exchangeability underlying the model presented in this chapter. A more realistic modelling framework should distinguish automated and human behaviour. Table 3.3 reports the Pitman-Yor parameter estimates for the same destination nodes used in Table 3.1, after removing the periodic edges using the procedure of Heard, Rubin-Delanchy, and Lawson (2014). Clearly, a large proportion of the activity is automated, as demonstrated by the comparison between the values of N in the two tables 3.1 and 3.3. Also, parameter estimates significantly change across the two tables. This problem motivates the next chapter, which proposes an edge-based model for classification of periodic arrival times.

4

Classification of periodic arrivals in event time data

As discussed in Section 3.7.1, a large proportion of computer network data tends to exhibit periodic behaviour (Heard, Rubin-Delanchy, and Lawson, 2014; Price-Williams, Heard, and Turcotte, 2017). Table 3.3 demonstrates that, for the Pitman-Yor process model proposed in the previous chapter, polling activity has a considerable effect on the parameter estimates. Periodic event time data are also frequently observed in many other real life applications, for example astrophysics (Cicuttin et al., 1998), bioinformatics (Kocak et al., 2013), and object tracking (Li et al., 2010). The periodic arrival times can often be mixed with non-periodic events. Therefore, to model the generating process appropriately, it is necessary to correctly distinguish the event types. This chapter proposes a statistical method for classification of periodic arrivals within a sequence of event times.

In this chapter, the focus will be on network flow data, described in Section 2.2. Commonly, a large proportion of the connections from a network host can be ascribed to legitimate, automated polling to various services. It is therefore an important step in the model-building process to be able to correctly identify which connections are due to the presence of a human at the machine, and which others are purely automated. Making this distinction is crucial for network monitoring and statistical intrusion detection: anomalies related to the presence of an intruder within the network will be significantly easier to detect when the polling connections are filtered out from the analysis. Realistic modelling strategies seek to treat the two components separately: Price-Williams and Heard (2019) show that a nonparametric Wold process with step-function excitation is a suitable choice for modelling human events in computer network traffic data. That model can only be applied when periodic connections are not present: this chapter provides a novel statistical framework for filtering automated traffic when human events are mixed with polling connections.

A useful network-wide filtering approach for polling behaviour, based on Fisher's g-test for

periodicities, was proposed in Heard, Rubin-Delanchy, and Lawson (2014). For each pair of network nodes, the method looks for strong peaks in the periodogram of the event series of connections along that edge. The methodology is specifically developed in the context of computer network data, but it can be applied to any sequence of arrival times. A limitation of this approach is that all connections from an edge are deemed to be automated if the maximal periodicity for that edge is found to be significant, whereas activity on some network edges can contain a mixture of both automated and human activity. For example, connections to an email server are continuously refreshed with a fixed periodicity, but the user might also manually ask if new messages have been received. It is therefore potentially valuable to further understand which of the events on such edges are actually associated with the presence of a user. This chapter aims to complement the existing methodologies and provide a data filtering algorithm for network connection records, where each connection on an edge will be classified as periodic or non-periodic through a mixture probability model. Note that the aim of the proposed model is not necessarily to discern malicious automated activities, such as those generated by botnets, from human activities, but to provide a statistical technique for separating purely automated, polling activity, either malicious or legitimate, from non-periodic connections, which also include human activity.

The problem of periodicity detection in computer network traffic has been extensively studied in the computer science literature. Common approaches include spectral analysis (Barbosa, Sadre, and Pras, 2012; AsSadhan and Moura, 2014; Heard, Rubin-Delanchy, and Lawson, 2014; Price-Williams, Heard, and Turcotte, 2017), which are often combined with thresholding methods (Bartlett, Heidemann, and Papadopoulos, 2011; Huynh et al., 2016; Chen et al., 2016). Alternatives include modelling of inter-arrival times (Bilge et al., 2012; Qiao et al., 2012; Hubballi and Goyal, 2013), where distributional assumptions are imposed and the behaviour is tested under the null of no periodicities (He et al., 2009; McPherson and Ortega, 2011). Finally, some authors identify signals of periodicities in the autocorrelation function (Gu, Zhang, and Lee, 2008; Qiao et al., 2013), using changepoint methods (Price-Williams, Heard, and Turcotte, 2017) or summary statistics computed sequentially in time windows (Eslahi et al., 2015). Price-Williams, Heard, and Turcotte (2017) also use wrapped distribution for detecting automated subsequences of events, and their methodology is able to handle changes in the periodicity and parameters in the model, but the human activity within a periodic subsequence is not captured. Most models proposed in the literature are aimed at classifying the entire edge as purely periodic or non-periodic. The model proposed in this chapter further analyses the edges with dominant periodicities, with the objective of recovering the human connections, when present; each observation is separately classified as periodic or non-periodic. The models described in this chapter could make a direct contribution to real-world network analysis, providing an efficient method for separating human and polling connections on the same edge, allowing deployment of existing methodologies (Price-Williams and Heard, 2019) for analysis of the filtered events.

The remainder of the chapter is organised as follows: Section 4.1 summarises the use of Fisher's g-test for identifying the dominant periodicity in event time data. Using that periodicity, Section 4.2 introduces two transformations of event times which will be used to classify individual events as periodic or non-periodic. Models for these two quantities are presented in Sections 4.3 and 4.4 respectively. Applications on real and synthetic data are discussed in Section 4.5.

4.1. Fisher's g-test for detecting periodicities in event time data

Let $t_1 < t_2 < \cdots < t_N$ be a sequence of arrival times, and $N(\cdot)$ be a counting process recording the number of events over time. In the computer network application, $N(\cdot)$ counts connections over time from the client to the server, for any particular client and server pair, see (1.2). It is most practical to treat $N(\cdot)$ as a discrete-time process, with connection counts aggregated within bins of fixed width δ. Thus $N(t)$ will denote the number of events after $t\delta$ seconds. The increments of the process are the corresponding bin counts $\mathrm{d}N(t) = N(t) - N(t-1)$.

After T time units of observation, the discrete Fourier transform for the zero-mean corrected process yields the periodogram

$$\hat{S}(f) = \frac{1}{T}\left|\sum_{t=1}^{T}\left(\mathrm{d}N(t) - \frac{N(T)}{T}\right)e^{-2\pi\imath ft}\right|^2.$$

The Fast Fourier Transform (FFT) allows efficient computation of $\hat{S}(f_k)$ at the discrete time Fourier frequencies $f_k = k/T$, $k = 1, \ldots, m$ where $m = \lfloor T/2 \rfloor$, in $\mathcal{O}(T \log T)$ operations. Peaks in the periodogram values might correspond to periodic signals in the sequence of arrival times. Fisher (1929) proposed an exact test for the null hypothesis of no periodicities using the *g-statistic*,

$$g(\hat{S}) = \frac{\max_{1\leq k\leq m}\hat{S}(f_k)}{\sum_{1\leq j\leq m}\hat{S}(f_j)}. \tag{4.1}$$

The test arises in the theory of harmonic time series analysis, and is the uniformly most powerful symmetric invariant procedure (Anderson, 1971) against the alternative hypothesis of a periodicity existing at a single Fourier frequency, for a null hypothesis of a white noise spectrum (for further details, see Percival and Walden, 1993). Under such null hypothesis, Fisher (1929) derived an exact p-value for a realised value g of $g(\hat{S})$, for which there also exists a convenient asymptotic approximation (Jenkins and Priestley, 1957):

$$\mathbb{P}\{g(\hat{S}) > g\} = \sum_{j=1}^{\min\{\lfloor 1/g\rfloor, m\}}(-1)^{j-1}\binom{m}{j}(1-jg)^{m-1} \approx 1 - \{1-\exp(-mg)\}^m. \tag{4.2}$$

If an sequence of arrival times is found to be periodic at a given significance level, the corre-

sponding *period* is

$$p = \delta \left(\underset{f_k : 1 \leq k \leq m}{\operatorname{argmax}} \hat{S}(f_k) \right)^{-1}. \tag{4.3}$$

In a Bayesian setting, methods to detect periodicities have been developed in astrophysics and astrostatistics (Jaynes, 1987), or in biostatistics and bioinformatics (de Lichtenberg et al., 2005; Kocak et al., 2013). None of these methods are fully scalable and as easy to interpret as the g-test; therefore, for the purposes of this chapter, the periodicities will be obtained using (4.1) and the corresponding p-value (4.2).

In computer network traffic, if the p-value is below a pre-specified small significance level, then the *entire* edge is deemed to be periodic. Otherwise, if an edge is found to be not significantly periodic, then it is assumed that the majority of the activity on that edge can be ascribed to non-periodic events, possibly related to the presence of a human at the machine. If an edge is classified as periodic using the g-test, it is also possible that the observed connections contain a mixture of both polling and human activity. The objective of this chapter is to further refine the classification performance for such mixed-type edges, classifying not only the entire edge activity as periodic or non-periodic, but each observed event on the edge.

The performance of the g-test on mixtures of periodic and non-periodic event times can be investigated via simulation. A sequence of 1000 events repeating every $p = 10$ seconds is generated, and mixed with events generated from a Poisson process on the same time frame, with different rates λ. For each value of λ, the simulation is repeated 100 times to estimate the expected p-value (4.2) from the g-test and the results are reported in Figure 4.1. For interpretability, the mean proportion of periodic events, which is monotonically decreasing in λ, is plotted on the horizontal axis. It is clear from Figure 4.1 that the expected p-value decreases when the proportion of periodic events increases, but the p-value is sufficiently small even when the proportion of automated events is small. For example, for a 2% percentage of polling arrival times, the resulting expected p-value of the g-test is ≈ 0.0001.

4.2. Circular statistics for classifying event times

Let $t_1 < t_2 < \cdots < t_N$ be a sequence of arrival times, and let $\boldsymbol{z} = (z_1, \ldots, z_N)$ be a vector of binary indicator variables, such that $z_i = 1$ if the i-th event was periodic, and $z_i = 0$ if it was non-periodic, or human-generated. For each event time t_i, the following circular transformation can be defined

$$x_i = \frac{2\pi}{p}(t_i \bmod p). \tag{4.4}$$

This transformation is particularly suitable for sequences presenting *fixed phase polling* (Price-Williams, Heard, and Turcotte, 2017): the event times are expected to occur every p seconds, with a

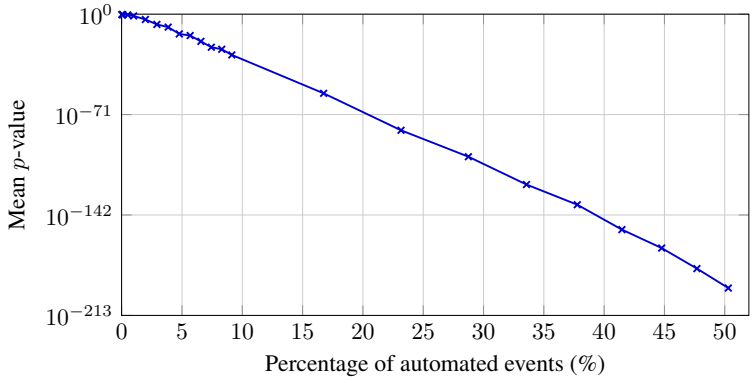


Figure 4.1.: *Expected p-value for the g-test against the percentage of periodic events.*

zero-mean random error. Wrapping the sequence to $[0, 2\pi)$ also makes the methodology robust to *dropouts* in the observations. If the events occur p seconds after the preceding arrival time, plus error, then the sequence exhibits *fixed duration polling*, and a more appropriate transformation might be:

$$x_i = \frac{2\pi}{p}\{(t_i - t_{i-1}) \bmod p\},$$

with $x_1 = 0$. This chapter mostly concerns with fixed phase polling, but the methodology could be adapted to the case of fixed duration polling.

The aim of the proposed methodology is to use the observed vector $\boldsymbol{x} = (x_1, \ldots, x_N)$ to estimate $\boldsymbol{z}$. For the i-th event, the first measurement, x_i, will reveal whether it was synchronous with the polling found to occur on those arrival times.

In some applications, a second *known* periodic effect will be present, such as a daily or annual seasonality. Denote this second periodicity p', where typically $p \ll p'$. A second circular transformation can then be defined:

$$y_i = \frac{2\pi}{p'}(t_i \bmod p'). \tag{4.5}$$

Within the application in computer networks, it could be assumed $p' = 86{,}400s$. Such measurement will show the time of day (which is 86,400 seconds long when there are no clock changes) at which the event occurred, and can be compared against an inferred diurnal model corresponding to human activity. More generally, one could be interested in the estimation of the density of the non-periodic events on the entire observation period, which yields the generic transformation $\tilde{t}_i = 2\pi t_i/T$.

In the next section, a mixture model for $\boldsymbol{x}$ is proposed, which can be used to classify events purely on their synchronicity with the polling signal. Then in Section 4.4 the model is extended to incorporate $\boldsymbol{y}$, to see how much extra discriminative information can be extracted from the time of day. The measurements (4.4) and (4.5) have both been scaled to lie on the unit circle with domain

$[0, 2\pi)$. This consistency in scaling will be convenient for specifying the full probability model (4.23) for event times in Section 4.4, since this makes simultaneous use of both quantities.

4.3. A wrapped normal-uniform mixture model

If a sequence of arrival times is classified periodic with period p (4.3), then a majority of the wrapped values $\boldsymbol{x}$ from (4.4) will be concentrated around a peak. A wrapped normal distribution $\mathbb{WN}_{[0,2\pi)}(\mu, \sigma^2)$ model is therefore proposed for those events, where $\sigma > 0$ quantifies the variability of event times around the peak location $\mu \in [0, 2\pi)$. The density of $\mathbb{WN}_{[0,2\pi)}(\mu, \sigma^2)$ is

$$\phi_{\text{WN}}^{[0,2\pi)}(x; \mu, \sigma^2) = \sum_{k=-\infty}^{\infty} \phi(x + 2\pi k; \mu, \sigma^2) \mathbb{1}_{[0,2\pi)}(x), \tag{4.6}$$

where $\phi(\cdot; \mu, \sigma^2)$ and later $\Phi\{\cdot; \mu, \sigma^2\}$ will represent, respectively, the density and distribution functions of the Gaussian distribution $\mathbb{N}(\mu, \sigma^2)$.

In practical applications, p will usually be relatively small, hence it is reasonable to assume that the density of the non-periodic events is smooth and therefore locally well-approximated by a uniform distribution on the unit circle. Together, these components imply a density for x_i conditional on the latent variable z_i,

$$f(x_i|z_i) = \phi_{\text{WN}}^{[0,2\pi)}(x_i; \mu, \sigma^2)^{z_i} (2\pi)^{z_i - 1} \mathbb{1}_{[0,2\pi)}(x_i). \tag{4.7}$$

Let $\theta \in [0, 1]$ be the unknown proportion of events which are generated automatically and periodically, such that $\mathbb{P}(z_i = 1) = \theta$. Finally, let

$$\boldsymbol{\psi} = (\mu, \sigma^2, \theta)$$

be the three model parameters which have been introduced. Then, assuming the individual values of $\boldsymbol{x}$ are drawn independently of one another, the likelihood function of the three model parameters is

$$L(\boldsymbol{\psi}|\boldsymbol{x}) = \prod_{i=1}^{N} \left\{ \theta \phi_{\text{WN}}^{[0,2\pi)}(x_i; \mu, \sigma^2) + \frac{1 - \theta}{2\pi} \right\} \mathbb{1}_{[0,2\pi)}(x_i). \tag{4.8}$$

It is not analytically possible to optimise the likelihood in (4.8) directly; instead, an Expectation-Maximisation (EM) algorithm (Dempster, Laird, and Rubin, 1977), common for mixture models, is proposed in the next section. Figure 4.2 shows the density function of the uniform and wrapped normal distribution, and a mixture of the two.

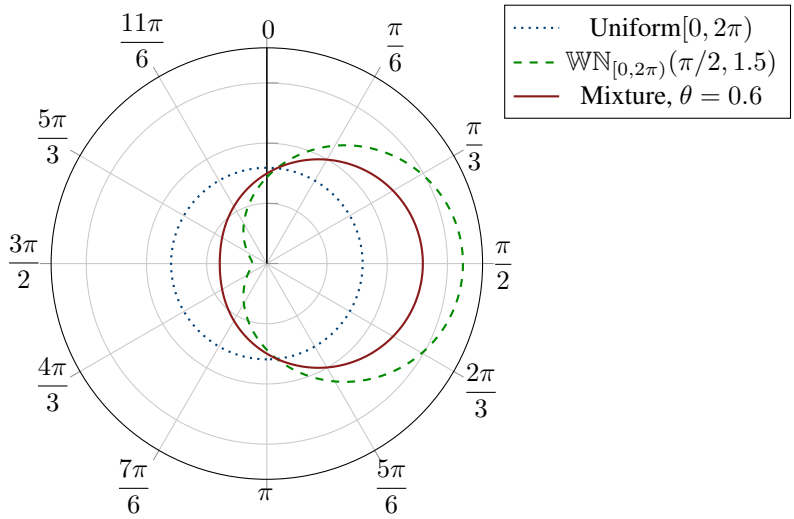


Figure 4.2.: *Examples of uniform density, wrapped normal density with* $\mu = \pi/2$ *and* $\sigma^2 = 1.5$*, and the mixture density* (4.8) *with* $\theta = 0.6$.

4.3.1. An EM algorithm for parameter estimation

The EM algorithm is an iterative method for likelihood optimisation, commonly used in missing data problems. The EM algorithm iterates between the following two steps until convergence: first, the calculation of the expectation of the log-likelihood function with respect to the conditional distribution the latent variables, given the observed data and a guess of the parameters (the E-step); second, the maximisation of the resulting conditional expectation with respect to the model parameters (the M-step). More details about the algorithm are given in Dempster, Laird, and Rubin (1977) and McLachlan and Krishnan (2007).

In the uniform-wrapped normal model (4.7) presented in the previous section, the allocations $\boldsymbol{z}$ could be considered as missing observations. In order to develop an EM algorithm for estimating $\boldsymbol{\psi}$, it is necessary to introduce additional latent variables $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_N)$ for the mixture components in the wrapped normal model (4.6). For $1 \leq i \leq N$, if $z_i = 0$ then let $\kappa_i = 0$ with probability 1; for $z_i = 1$ and $k \in \mathbb{Z}$, let

$$\mathbb{P}(\kappa_i = k | z_i = 1, \mu, \sigma^2) = \Phi\{2\pi(k+1); \mu, \sigma^2\} - \Phi\{2\pi k; \mu, \sigma^2\}. \tag{4.9}$$

Further, let

$$x_i | z_i = 1, \kappa_i = k, \mu, \sigma^2 \sim \bar{\mathbb{N}}_{[0,2\pi)}(\mu - 2\pi k, \sigma^2),$$

denoting a normal distribution with mean $\mu - 2\pi k$ and variance σ^2, truncated to $[0, 2\pi)$. Then the conditional density for x_i given z_i is again (4.7). The role of the latent variable κ_i is depicted in Figure 4.3.

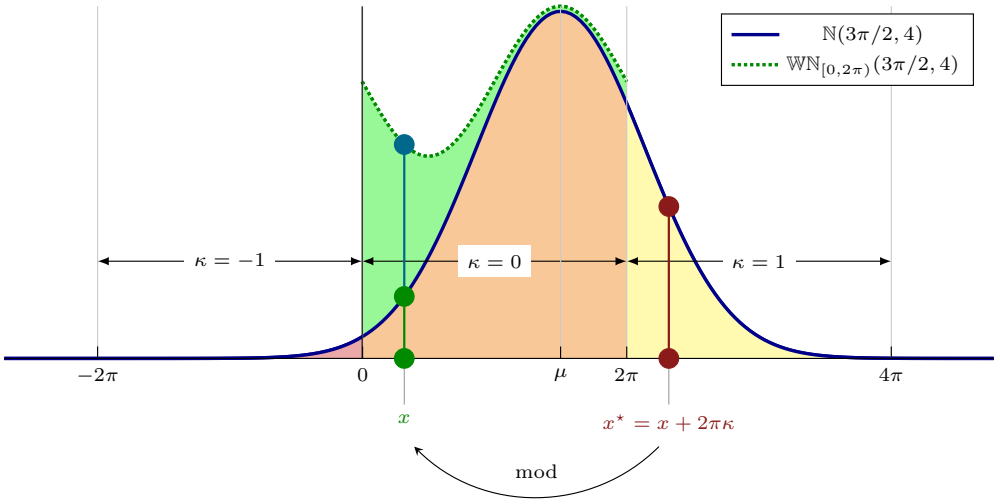


Figure 4.3.: *Interpretation of the latent variable κ. Suppose $x^\star \sim \mathbb{N}(\mu, \sigma^2)$, $x = x^\star \bmod 2\pi$ and $\kappa = (x^\star - x)/(2\pi)$. Then $x \sim \mathbb{WN}_{[0,2\pi)}(\mu, \sigma^2)$ and $\kappa = k$ with probability given by* (4.9).

Using the latent assignments $\boldsymbol{z}$ and $\boldsymbol{\kappa}$, the revised likelihood function is

$$L(\boldsymbol{\psi}|\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\kappa}) \propto \prod_{i=1}^{N} \left(\frac{1-\theta}{2\pi}\right)^{1-z_i} \left\{\theta\phi(x_i + 2\pi\kappa_i; \mu, \sigma^2)\right\}^{z_i}. \tag{4.10}$$

At iteration m of the EM algorithm, given an estimate $\boldsymbol{\psi}_{(m)}$ of $\boldsymbol{\psi}$, the E-step computes

$$Q(\boldsymbol{\psi}|\boldsymbol{\psi}_{(m)}) = \mathbb{E}_{\boldsymbol{z},\boldsymbol{\kappa}|\boldsymbol{x},\boldsymbol{\psi}_{(m)}} \left\{\log L\left(\boldsymbol{\psi}|\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\kappa}\right)\right\}, \tag{4.11}$$

usually known as Q-function, where the expectation is taken with respect to the conditional distribution of $\boldsymbol{z}$ and $\boldsymbol{\kappa}$, given $\boldsymbol{x}$ and $\boldsymbol{\psi}_{(m)}$. This amounts to evaluating the so called *responsibilities*,

$$\zeta_{i(j,k)} = \mathbb{E}_{\boldsymbol{z},\boldsymbol{\kappa}|\boldsymbol{x},\boldsymbol{\psi}_{(m)}} \left\{\mathbb{1}_{(j,k)}(z_i, \kappa_i)|x_i, \boldsymbol{\psi}_{(m)}\right\} = \mathbb{P}\left(z_i = j, \kappa_i = k|x_i, \boldsymbol{\psi}_{(m)}\right), \tag{4.12}$$

since, using (4.10), the Q-function (4.11) then simplifies to

$$\sum_{i=1}^{N} \left\{\zeta_{i(0,0)} \log\left(\frac{1-\theta_{(m)}}{2\pi}\right) + \sum_{k=-\infty}^{\infty} \zeta_{i(1,k)} \log\left(\theta\phi(x_i; \mu_{(m)} - 2\pi k, \sigma^2_{(m)})\right)\right\}. \tag{4.13}$$

The responsibilities in (4.12) can be calculated using Bayes theorem, giving

$$\zeta_{i(j,k)} \propto \left\{\theta_{(m)}\phi(x_i; \mu_{(m)} - 2\pi k, \sigma^2_{(m)})\right\}^{j} \left(\frac{1-\theta_{(m)}}{2\pi}\right)^{1-j}, \tag{4.14}$$

where the normalising constant is $\theta_{(m)} \sum_{k'=-\infty}^{\infty} \phi(x_i; \mu_{(m)} - 2\pi k', \sigma^2_{(m)}) + (1-\theta_{(m)})/2\pi$. Finally,

maximising (4.13) with respect to $\boldsymbol{\psi}$ as the M-step gives:

$$\begin{aligned}
\tilde{\mu}_{(m+1)} &= \frac{\sum_{i=1}^{N}\sum_{k=-\infty}^{\infty}(x_i + 2\pi k)\zeta_{i(1,k)}}{\sum_{i=1}^{N}\sum_{k=-\infty}^{\infty}\zeta_{i(1,k)}}, \\
\mu_{(m+1)} &= \tilde{\mu}_{(m+1)} \bmod 2\pi, \\
\sigma^2_{(m+1)} &= \frac{\sum_{i=1}^{N}\sum_{k=-\infty}^{\infty}(x_i + 2\pi k - \tilde{\mu}_{(m+1)})^2\zeta_{i(1,k)}}{\sum_{i=1}^{N}\sum_{k=-\infty}^{\infty}\zeta_{i(1,k)}}, \\
\theta_{(m+1)} &= \frac{1}{N}\sum_{i=1}^{N}\sum_{k=-\infty}^{\infty}\zeta_{i(1,k)} = 1 - \frac{1}{N}\sum_{i=1}^{N}\zeta_{i(0,0)}.
\end{aligned} \tag{4.15}$$

In practical computations, the infinite sums must be truncated to a suitable level.

The EM algorithm for the uniform-wrapped normal model is summarised in Algorithm 4.1.

Algorithm 4.1: EM algorithm for the uniform-wrapped normal mixture model.

Result: maximum likelihood estimates $\hat{\boldsymbol{\psi}} = (\hat{\mu}, \hat{\sigma}^2, \hat{\theta})$ for the model parameters.

initialise the estimates: $\boldsymbol{\psi}_{(1)} = (\mu_{(1)}, \sigma^2_{(1)}, \theta_{(1)})$;

repeat

  E-step: calculate the responsibilities $\zeta_{i(j,k)}$ using (4.14);

  M-step: update the estimates $\boldsymbol{\psi}_{(m+1)} = (\mu_{(m+1)}, \sigma^2_{(m+1)}, \theta_{(m+1)})$ using (4.15);

until *convergence*;

4.3.2. A Bayesian formulation

Data augmentation (Higdon, 1998) can be used to construct an analogue of the EM algorithm in a Bayesian setting. In this case, a prior distribution to the model parameters $\boldsymbol{\psi}$ is required. Inference is then carried out by examining the joint posterior distribution of the model parameters and latent variables $\boldsymbol{z}$ and $\boldsymbol{\kappa}$. For the model proposed in this section, and in general for all the models in this thesis, the joint posterior is analytically intractable, and it is therefore necessary to use Markov Chain Monte Carlo methods (MCMC) to obtain samples. MCMC methods are based on constructing Markov chains with equilibrium distribution corresponding to the target probability distribution of interest, in such a way that states from the chain asymptotically correspond to samples from the desired distribution. In particular, in this section a Gibbs sampler (Geman and Geman, 1984) is used. Gibbs sampling generates a Markov chain by iteratively sampling from the conditional distribution of each parameter given the current sampled values of the remaining parameters and the data. For Gibbs sampling to be feasible, such distributions are usually of tractable form. More details about Gibbs sampling, and MCMC algorithms in general, are given in any modern text on computational or Bayesian statistics (see, for example, Givens and Hoeting, 2012; Gelman et al., 2013).

A convenient choice of prior distribution assumes a factorisation $p(\boldsymbol{\psi}) = p(\mu, \sigma^2)p(\theta)$, where $\theta \sim \text{Beta}(\gamma_0, \delta_0)$ and $(\mu, \sigma^2) \sim \text{NIG}(\mu_0, \lambda_0, \alpha_0, \beta_0)$ and NIG denotes the normal-inverse-gamma distribution. The chosen prior distributions are conjugate for the likelihood, implying that the posterior distributions are in the same probability distribution family as the priors (for more details, see Bernardo and Smith, 1994). This allows the inferential process to be analytically tractable. The prior and posterior probabilities for the latent assignments, conditional on $\boldsymbol{\psi}$, are the same as (4.9) and (4.14) respectively. Conditional on $\boldsymbol{z}$, the posterior distribution for the mixing proportion is

$$\theta|\boldsymbol{z} \sim \text{Beta}(\gamma_0 + N_1, \delta_0 + N_0), \tag{4.16}$$

where $N_1 = \sum_{i=1}^{N} z_i$ is the number of automated events and $N_0 = N - N_1$ is the number of human and non-periodic automated events. The conditional posterior distribution of μ and σ^2 is $\text{NIG}(\mu_{N_1}, \lambda_{N_1}, \alpha_{N_1}, \beta_{N_1})$, where

$$\begin{aligned}
\tilde{x} &= \sum_{i:z_i=1} (x_i + 2\pi\kappa_i)/N_1, \\
\mu_{N_1} &= \frac{\lambda_0\mu_0 + N_1\tilde{x}}{\lambda_0 + N_1}, && (4.17)\\
\lambda_{N_1} &= \lambda_0 + N_1, && (4.18)\\
\alpha_{N_1} &= \alpha_0 + N_1/2, \\
\beta_{N_1} &= \beta_0 + \frac{1}{2}\left\{ \sum_{i:z_i=1} (x_i + 2\pi\kappa_i - \tilde{x})^2 + \frac{\lambda_0 N_1}{\lambda_{N_1}}(\tilde{x} - \mu_0)^2 \right\}.
\end{aligned}$$

Similarly to the case of (4.15), samples $\mu = \tilde{\mu} \bmod 2\pi$ from the posterior for (μ, σ^2) should be used, where $\tilde{\mu}$ is sampled from $\text{NIG}(\mu_{N_1}, \lambda_{N_1}, \alpha_{N_1}, \beta_{N_1})$. Note that, conditional on the pairs (z_i, κ_i), the calculations of the conditional posterior distribution of μ and σ^2 follow the standard updates for the posterior distribution of a Gaussian distribution with unknown mean and variance (see, for example, Bernardo and Smith, 1994; Murphy, 2012).

The Gibbs sampler for the uniform-wrapped normal model is summarised in Algorithm 4.2.

Algorithm 4.2: Gibbs sampler for the uniform-wrapped normal mixture model.

Result: samples from the posterior $p(\boldsymbol{\psi}, \boldsymbol{z}, \boldsymbol{\kappa}|\boldsymbol{x})$.

intialise the parameters: $(\mu_{(0)}, \sigma^2_{(0)})$ and $\theta_{(0)}$;

for $j = 1, 2, \ldots$ **do**

  sample $(\boldsymbol{z}_{(j)}, \boldsymbol{\kappa}_{(j)})$ from $p(z_i, \kappa_i|x_i, \boldsymbol{\psi}_{(j-1)}),\ i = 1, \ldots, N$, using (4.12);

  sample the mixing proportion $\theta_{(j)}$ from $p(\theta|\boldsymbol{z}_{(j)})$ using (4.16);

  sample $(\mu_{(j)}, \sigma^2_{(j)})$ from $\text{NIG}(\mu_{N_1}, \lambda_{N_1}, \alpha_{N_1}, \beta_{N_1})$, see (4.15), where μ_{N_1}, λ_{N_1}, α_{N_1} and β_{N_1} are calculated conditional on $(\boldsymbol{z}_{(j)}, \boldsymbol{\kappa}_{(j)})$;

end

4.4. Incorporating time of day

The model presented in Section 4.3 only made use of $\boldsymbol{x}$, the arrival times once wrapped onto the unit circle according to the estimated periodicity p (4.4); recall that these values reveal the synchronicity of each event with the automated polling signal. However, further information might potentially be obtained from $\boldsymbol{y}$ (4.5), the times of day at which each event occurred. In computer networks, this is a reasonable assumption, since any human generated events should be subject to some level of diurnality. This section introduces a model for the daily distribution of human connections to help extract this extra information. Following Turcotte, Heard, and Neil (2014), a flexible model for the distribution of arrivals of human events through a typical day will be obtained by assuming the density to be a step-function with $\ell \geq 1$ segments, written

$$s(y; \ell, \boldsymbol{\tau}, \boldsymbol{h}) = \frac{\mathbb{1}_{[0,\tau_1)\cup[\tau_\ell,2\pi)}(y)\; h_\ell}{2\pi - \tau_\ell + \tau_1} + \sum_{j=1}^{\ell-1} \frac{\mathbb{1}_{[\tau_j,\tau_{j+1})}(y)\; h_j}{\tau_{j+1} - \tau_j}. \tag{4.19}$$

The segment probabilities $\boldsymbol{h} = (h_1, \ldots, h_\ell) \in [0,1]^\ell$ satisfy $\sum_{j=1}^{\ell} h_j = 1$ and the circular changepoints $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_\ell)$, $0 \leq \tau_1 < \ldots < \tau_\ell < 2\pi$ determine the step positions.

The number of segments ℓ is treated as unknown and assigned a geometric prior with parameter $\nu \in (0,1)$ and mass function $\nu(1-\nu)^{\ell-1}$. The natural prior for $\boldsymbol{h}|\boldsymbol{\tau}, \ell$ (Bernardo and Smith, 1994) is

$$\text{Dirichlet}\left(\eta\Lambda\{(\tau_1,\tau_2)\}, \ldots, \eta\Lambda\{(\tau_{\ell-1},\tau_\ell)\}, \eta\Lambda\{(\tau_\ell, 2\pi)\cup(0,\tau_1)\}\right), \tag{4.20}$$

where $\eta > 0$ is a concentration parameter and $\Lambda\{\cdot\}$ is here taken to be the Lebesgue measure. The hierarchical specification of the model is completed with an uninformative prior on the segment locations, assumed to be the order statistics of ℓ draws from the uniform distribution on $[0, 2\pi)$: $p(\boldsymbol{\tau}|\ell) = \ell!(2\pi)^{-\ell}$. Bayes theorem gives the posterior distribution of the parameters up to a proportionality constant:

$$\begin{aligned} p(\boldsymbol{h}, \boldsymbol{\tau}, \ell|\boldsymbol{y}) \propto\; & p(\boldsymbol{y}|\boldsymbol{h}, \boldsymbol{\tau}, \ell) p(\boldsymbol{h}|\boldsymbol{\tau}, \ell) p(\boldsymbol{\tau}|\ell) p(\ell) = \nu(1-\nu)^{\ell-1}\Gamma(2\pi\eta)\ell!(2\pi)^{-\ell} \\ & \times \left(\frac{h_\ell}{2\pi - \tau_\ell + \tau_1}\right)^{N'_\ell} \frac{h_\ell^{\eta(2\pi-\tau_\ell+\tau_1)-1}}{\Gamma\{\eta(2\pi-\tau_\ell+\tau_1)\}} \prod_{j=1}^{\ell-1} \left(\frac{h_j}{\tau_{j+1}-\tau_j}\right)^{N'_j} \frac{h_j^{\eta(\tau_{j+1}-\tau_j)-1}}{\Gamma[\eta(\tau_{j+1}-\tau_j)]}, \end{aligned}$$

where $p(\boldsymbol{y}|\boldsymbol{h}, \boldsymbol{\tau}, \ell) \propto \prod_{i=1}^{N} s(y_i; \boldsymbol{h}, \boldsymbol{\tau}, \ell)$, and $N'_j = \sum_{i=1}^{N} \mathbb{1}_{[\tau_j,\tau_{j+1})}(y_i)$ is the number of observations in the j-th segment, $1 \leq j \leq \ell - 1$, $N'_\ell = N - \sum_{j=1}^{\ell-1} N'_j$.

Given ℓ segments defined by changepoints $\boldsymbol{\tau}$, the Dirichlet probabilities $\boldsymbol{h}$ can be also integrated out to yield the marginal density function $p(\boldsymbol{y}|\boldsymbol{\tau}, \ell)$ of observing daily arrival times $\boldsymbol{y}$, which is

given by:

$$p(\boldsymbol{y}|\boldsymbol{\tau},\ell) = \int_{S_\ell} p(\boldsymbol{y},\boldsymbol{h}|\boldsymbol{\tau},\ell)\mathrm{d}\boldsymbol{h} = \int_{S_\ell} p(\boldsymbol{y}|\boldsymbol{h},\boldsymbol{\tau},\ell)p(\boldsymbol{h}|\boldsymbol{\tau},\ell)\mathrm{d}\boldsymbol{h}$$
$$= \frac{\Gamma(2\pi\eta)\Gamma\{N'_\ell + \eta(2\pi - \tau_\ell + \tau_1)\}}{\Gamma(2\pi\eta + N)\Gamma\{\eta(2\pi - \tau_\ell + \tau_1)\}(2\pi - \tau_\ell + \tau_1)^{N'_\ell}} \prod_{j=1}^{\ell-1} \frac{\Gamma\{N'_j + \eta(\tau_{j+1} - \tau_j)\}}{\Gamma\{\eta(\tau_{j+1} - \tau_j)\}(\tau_{j+1} - \tau_j)^{N'_j}}, \tag{4.21}$$

where S_ℓ is the probability ℓ-simplex.

In contrast to the human events, automated periodic events are generated regularly by the underlying polling mechanism, which is likely to be irrespective of the time of day. Recall from Section 4.1 that the binary indicator variable z_i is defined to be equal to 1 if the i-th event was periodic, and 0 otherwise. The approach which will now be adopted is to model the conditional density for the unwrapped event time t_i, depending on the value of z_i.

For simplicity of presentation, it will be assumed that the length of the observation period, T, will be both a whole number of days and an integer multiple of p. Under this assumption

$$f(t_i|z_i) = \frac{2\pi}{T} f(x_i|z_i = 1)^{z_i} f(y_i|z_i = 0)^{1-z_i} = \frac{2\pi}{T} \phi_{\text{WN}}^{[0,2\pi)}(x_i;\mu,\sigma^2)^{z_i} s(y_i;\ell,\boldsymbol{\tau},\boldsymbol{h})^{1-z_i}, \tag{4.22}$$

implying the marginal distribution mixture density

$$f(t_i) = \frac{2\pi}{T}\left\{\theta\phi_{\text{WN}}^{[0,2\pi)}(x_i;\mu,\sigma^2) + (1-\theta)s(y_i;\ell,\boldsymbol{\tau},\boldsymbol{h})\right\}. \tag{4.23}$$

If T is not a whole number of days or a multiple of p, standard rules for transformations of random variables give the marginal density:

$$f(t_i) = \theta\frac{2\pi/p \times \phi_{\text{WN}}^{[0,2\pi)}(x_i;\mu,\sigma^2)}{\lfloor T/p\rfloor + \int_0^{\upsilon_1} \phi_{\text{WN}}^{[0,2\pi)}(x_i;\mu,\sigma^2)\mathrm{d}x_i} + (1-\theta)\frac{2\pi/p' \times s(y_i;\boldsymbol{\tau},\boldsymbol{h},\ell)}{\lfloor T/p'\rfloor + \int_0^{\upsilon_2} s(y_i;\boldsymbol{\tau},\boldsymbol{h},\ell)\mathrm{d}y_i},$$

where $\upsilon_1 = (T \bmod p)/p \times 2\pi$ and $\upsilon_2 = (T \bmod p')/p' \times 2\pi$, supposing $\delta = 1$. The normalising constants can be also written integrating with respect to t_i as $\lfloor T/p\rfloor + 2\pi/p\int_0^{T \bmod p} \phi_{\text{WN}}^{[0,2\pi)}(t_i \bmod p \times 2\pi/p;\mu,\sigma^2)\mathrm{d}t_i$ and $\lfloor T/p'\rfloor + 2\pi/p'\int_0^{T \bmod p'} s(t_i \bmod p' \times 2\pi/p';\boldsymbol{\tau},\boldsymbol{h},\ell)\mathrm{d}t_i$. When $T \bmod p \approx 0$ and $T \bmod p' = 0$, then $\upsilon_1 \approx 0$, $\upsilon_2 = 0$, $\lfloor T/p\rfloor \approx T/p$ and $\lfloor T/p'\rfloor = T/p'$, which allows the density to be approximated as (4.23).

Figure 4.4 provides a graphical summary of the full model (4.23), and Figure 4.5 shows an illustrative example of the mixture density. Note that relaxing the assumptions of divisibility of T by p or p' simply requires straightforward calculation of corresponding normalising constants in (4.22), and this adjustment will be negligible when $\lfloor T\delta/p\rfloor$ is large.

Since most of the prior distributions have been chosen to be conjugate, it is possible to explicitly

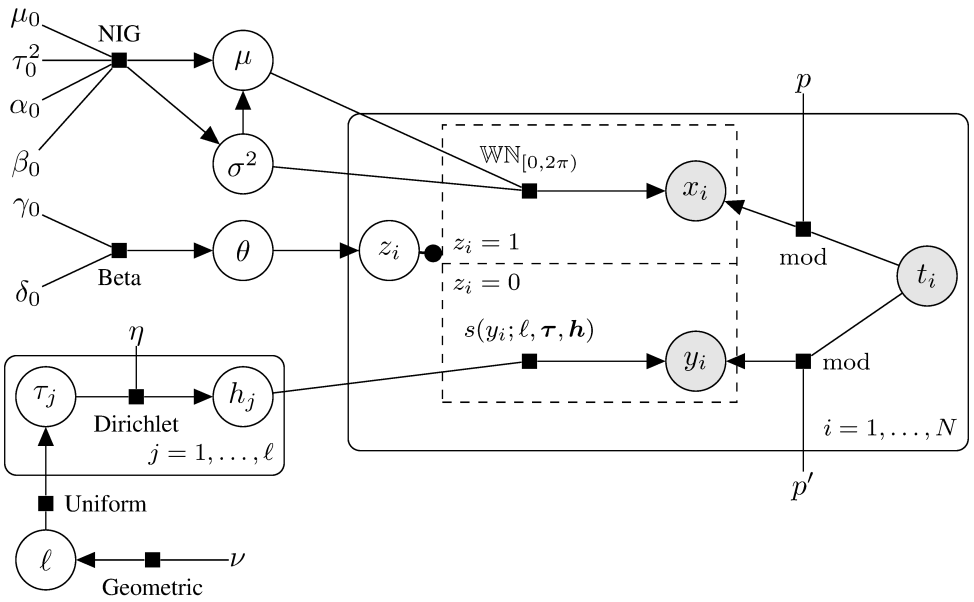


Figure 4.4.: *Graphical representation of the mixture model for classification of periodic activity.*

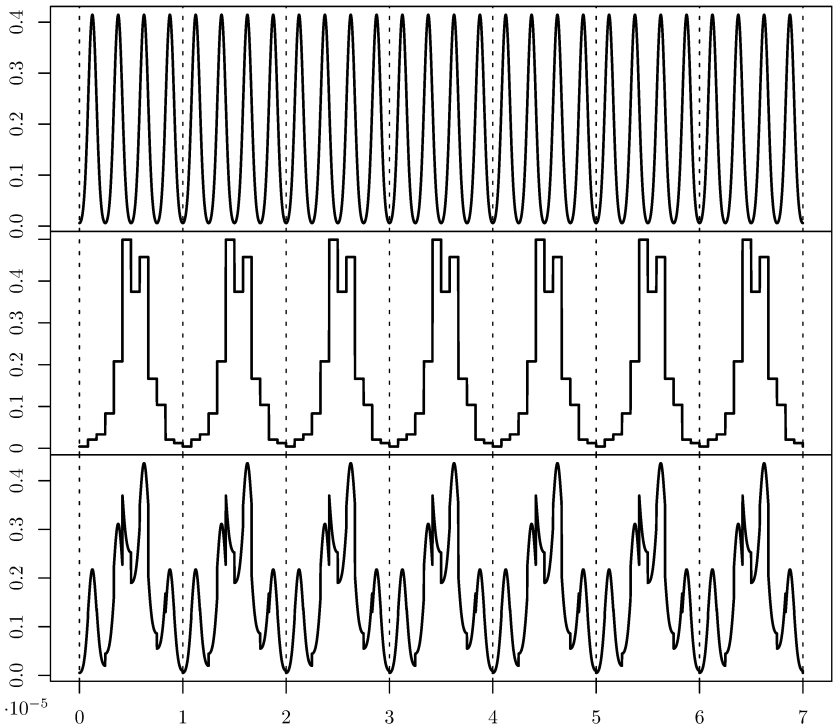


Figure 4.5.: *Example of the component densities in* (4.23)*, with* $T = 7 \times 86{,}400$ *(7 days),* $p = 21{,}600$ *(6 hours),* $\mu = \pi$*,* $\sigma^2 = 1$*,* $\theta = 0.5$*,* $\ell = 12$*, equally spaced segment locations, and step function heights* $\boldsymbol{h}$ *chosen to resemble a human daily distribution of arrival times. Upper panel: density of automated events (4 peaks per day are recorded since* $p = 6$ *hours). Middle: density of human events (daily distribution repeated each day). Lower: the resulting mixture density* (4.23)*.*

integrate out the segment heights $\boldsymbol{h}$, see (4.21), and the mixing proportion θ, leading to a collapsed Metropolis-within-Gibbs sampler (Liu, 1994) for inference. This is advantageous, since it reduces the simulation effort to sampling the latent variables $\boldsymbol{z}$ and $\boldsymbol{\kappa}$, the parameters μ and σ^2 for the wrapped normal component (4.6), and the number of circular changepoints ℓ and their locations $\boldsymbol{\tau}$ in the human event density (4.19). The algorithm is described in details in Section 4.4.1.

4.4.1. Bayesian inference

This section describes the collapsed Metropolis-within-Gibbs sampler used for Bayesian inference for the model described in the previous section. As opposed to the Gibbs sampler in Section 4.3.2, the conditional distributions are not all analytically tractable, and therefore Metropolis-Hastings (Hastings, 1970) steps are required within the algorithm. In this section, since the segment heights and mixing proportions are integrated out, the proposed algorithm is also referred to as *collapsed sampling* (Liu, 1994). Metropolis-Hastings is one of the most common MCMC algorithms for obtaining random samples from probability distributions, and it is described in most modern computational and Bayesian statistics textbooks (Givens and Hoeting, 2012; Gelman et al., 2013). At each iteration, the Metropolis-Hastings algorithm selects a candidate sample from a proposal distribution dependent on the current state of the Markov chain, which is either accepted or rejected.

As before, the conditional distribution for each parameter are required. Note that the full conditional distribution for the latent variable pair (z_i, κ_i) for the i-th event factorises as

$$\begin{aligned}\mathbb{P}\{(z_i,\kappa_i)=(z,k)|\boldsymbol{z}_{-i},\boldsymbol{\kappa}_{-i},\boldsymbol{t},\mu,\sigma^2,\ell,\boldsymbol{\tau}\}=\\ =\mathbb{P}(\kappa_i=k|z_i=z,t_i,\mu,\sigma^2)\mathbb{P}(z_i=z|\boldsymbol{z}_{-i},\boldsymbol{t},\mu,\sigma^2,\ell,\boldsymbol{\tau}),\end{aligned} \quad (4.24)$$

where the subscript $-i$ on a vector denotes the same vector with the i-th element removed. The first term on the right hand side of (4.24) is

$$\mathbb{P}(\kappa_i=k|z_i,t_i,\mu,\sigma^2)\propto\{\phi(x_i+2\pi k;\mu,\sigma^2)\}^{z_i}\{\mathbb{1}_0(k)\}^{1-z_i}.$$

For the second term, it is easily seen that

$$\mathbb{P}(z_i=z|\boldsymbol{z}_{-i},\boldsymbol{t},\mu,\sigma^2,\ell,\boldsymbol{\tau})\propto\mathbb{P}(z_i=z|\boldsymbol{z}_{-i})f(t_i|\boldsymbol{t}_{-i},\boldsymbol{z},\mu,\sigma^2,\ell,\boldsymbol{\tau}). \quad (4.25)$$

The first term on the right hand side of (4.25) can be rewritten as the marginalised prior probability ratio $\mathbb{P}(\boldsymbol{z})/\mathbb{P}(\boldsymbol{z}_{-i})$. Note that:

$$\mathbb{P}(\boldsymbol{z})=\int_0^1\mathbb{P}(\boldsymbol{z}|\theta)p(\theta)\mathrm{d}\theta=\frac{\Gamma(N_1+\gamma_0)\Gamma(N_0+\delta_0)}{\Gamma(N+\gamma_0+\delta_0)}\frac{\Gamma(\gamma_0+\delta_0)}{\Gamma(\gamma_0)\Gamma(\delta_0)}.$$

Hence, letting $N_1^{-i}=\sum_{i'\neq i}z_{i'}$ be the number of classified periodic events excluding the i-th

event,

$$\mathbb{P}(z_i = 1|\boldsymbol{z}_{-i}) = \frac{N_1^{-i} + \gamma_0}{N - 1 + \gamma_0 + \delta_0},$$

and $\mathbb{P}(z_i = 0|\boldsymbol{z}_{-i}) = 1 - \mathbb{P}(z_i = 1|\boldsymbol{z}_{-i})$. For the second term of (4.25), for $z_i = 1$, $f(t_i|z_i = 1, \mu, \sigma^2) \propto \phi_{\text{WN}}^{[0,2\pi)}(x_i; \mu, \sigma^2)$. For $z_i = 0$, from (4.21) it follows that

$$\begin{aligned} f(t_i|\boldsymbol{t}_{-i}, z_i = 0, \boldsymbol{z}_{-i}, \ell, \boldsymbol{\tau}) \propto f(y_i|\boldsymbol{y}_{-i}, z_i = 0, \boldsymbol{z}_{-i}, \ell, \boldsymbol{\tau}) &= \frac{f(\boldsymbol{y}|z_i = 0, \boldsymbol{z}_{-i}, \ell, \boldsymbol{\tau})}{f(\boldsymbol{y}_{-i}|\boldsymbol{z}_{-i}, \ell, \boldsymbol{\tau})} \\ &= \frac{\sum_{i' \neq i} \mathbb{1}_0(z_{i'}) \mathbb{1}_{[\tau_{j^\star}, \tau_{j^\star+1})}(y_{i'}) + \eta(\tau_{j^\star+1} - \tau_{j^\star})}{(N_0^{-i} + 2\pi\eta)(\tau_{j^\star+1} - \tau_{j^\star})}, \end{aligned} \quad (4.26)$$

where $j^\star \in \{1, \ldots, \ell - 1\}$ is the segment $[\tau_{j^\star}, \tau_{j^\star+1})$ containing y_i. If $j^\star = \ell$, then $\tau_{j^\star+1} - \tau_{j^\star}$ in (4.26) is substituted by $\Lambda\{(0, \tau_1) \cup (\tau_\ell, 2\pi)\} = 2\pi - \tau_\ell + \tau_1$.

For inference on μ and σ^2 conditional on the samples for which $z_i = 1$, the results in (4.17) and (4.18) still apply. Furthermore, inference for the number and location of circular changepoints is possible using Reversible Jump Markov Chain Monte Carlo (RJMCMC, Green, 1995), conditioning on the samples for which $z_i = 0$. Three scenarios are possible, corresponding to three Metropolis-Hastings moves: shift of one of the changepoints, insert a changepoint (*birth*), or delete a changepoint (*death*). The proposal distribution $q(\cdot|\cdot)$ for the Metropolis-Hastings algorithm is chosen to select a birth, death or shift move with equal probabilities. Conditional on the number of changepoints ℓ, the proposed changepoints are then obtained as follows (Green, 1995):

(a) *Shift* – an index j is picked at random from $\{1, \ldots, \ell\}$, and the proposed new changepoint $\tau^\star$ is sampled uniformly from (τ_{j-1}, τ_{j+1}),

(b) *Birth* – a new changepoint $\tau^\star$ is sampled at random from the uniform distribution in $[0, 2\pi)$,

(c) *Death* – a changepoint is removed at random from $\boldsymbol{\tau}$.

Shift one of the changepoints. Suppose that a shift move is selected, and the changepoint τ_j is shifted to $\tau^\star$. In this case, ℓ does not change. Define $N_1^\star = \sum_{i=1}^N \mathbb{1}_0(z_i)\mathbb{1}_{[\tau_{j-1}, \tau^\star)}(y_i)$ and $N_2^\star = \sum_{i=1}^N \mathbb{1}_0(z_i)\mathbb{1}_{[\tau^\star, \tau_{j+1})}(y_i)$. The acceptance probability for the move is calculated as $\alpha = \min\{1, \rho\}$, where:

$$\begin{aligned} \rho = &\frac{\Gamma\{N_1^\star + \eta(\tau_\star - \tau_{j-1})\}\Gamma\{N_2^\star + \eta(\tau_{j+1} - \tau_\star)\}}{\Gamma\{N'_{j-1} + \eta(\tau_j - \tau_{j-1})\}\Gamma\{N'_j + \eta(\tau_{j+1} - \tau_j)\}} \frac{\Gamma\{\eta(\tau_j - \tau_{j-1})\}\Gamma\{\eta(\tau_{j+1} - \tau_j)\}}{\Gamma\{\eta(\tau_\star - \tau_{j-1})\}\Gamma\{\eta(\tau_{j+1} - \tau_\star)\}} \\ &\times \frac{(\tau_{j+1} - \tau_j)^{N'_j}(\tau_j - \tau_{j-1})^{N'_{j-1}}}{(\tau_{j+1} - \tau_\star)^{N_2^\star}(\tau_\star - \tau_{j-1})^{N_1^\star}}. \end{aligned}$$

Additional adjustments in the expression above are required if τ_1 or τ_ℓ are shifted. Note that the above form for ρ is obtained from the conditional distribution $p(\boldsymbol{\tau}, \ell|\boldsymbol{y})$ using the decomposition $p(\boldsymbol{\tau}, \ell|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{\tau}, \ell)p(\boldsymbol{\tau}|\ell)p(\ell)$, and the prior and proposal ratios cancel out.

Insert one changepoint (*birth*). Suppose that a birth move is selected. Assume that the new proposed changepoint $\tau^\star$ falls between τ_j and τ_{j+1}. In this case ℓ is augmented by one. As before, define $N_1^\star = \sum_{i=1}^N \mathbb{1}_0(z_i)\mathbb{1}_{[\tau_j,\tau^\star)}(y_i)$ and $N_2^\star = \sum_{i=1}^N \mathbb{1}_0(z_i)\mathbb{1}_{[\tau^\star,\tau_{j+1})}(y_i)$. The acceptance probability for the move is again $\alpha = \min\{1, \rho\}$, where:

$$\rho = \frac{\Gamma\{N_1^\star + \eta(\tau_\star - \tau_j)\}\Gamma\{N_2^\star + \eta(\tau_{j+1} - \tau_\star)\}\Gamma\{\eta(\tau_{j+1} - \tau_{j-1})\}(\tau_{j+1} - \tau_j)^{N_j'}(1-\nu)}{\Gamma\{N_j' + \eta(\tau_{j+1} - \tau_j)\}\Gamma\{\eta(\tau_\star - \tau_j)\}\Gamma\{\eta(\tau_{j+1} - \tau_\star)\}(\tau_\star - \tau_j)^{N_1^\star}(\tau_{j+1} - \tau_\star)^{N_2^\star}}. \tag{4.27}$$

Additional adjustments in the expression above are required if $\tau_\star < \tau_1$ or $\tau_\star > \tau_\ell$. The product between the prior and proposal ratios simply reduces to $1 - \nu$.

Delete one changepoint (*death*). If a death move is selected, one of the changepoints is deleted at random from $\boldsymbol{\tau}$. The acceptance probability is again $\alpha = \min\{1, \rho\}$, with ρ obtained inverting the ratio (4.27) (Green, 1995).

The Bayesian inferential procedure for the model is summarised in Algorithm 4.3.

Algorithm 4.3: Collapsed Metropolis-within-Gibbs sampler for inference on the model for classification of periodic arrival times.

Result: samples from the posterior $p(\mu, \sigma^2, \boldsymbol{\tau}, \ell, \boldsymbol{z}, \boldsymbol{\kappa}|\boldsymbol{t})$.
intialise the parameters: $(\mu_{(0)}, \sigma^2_{(0)})$ and $(\boldsymbol{\tau}_{(0)}, \ell_{(0)})$;
for $j = 1, 2, \ldots$ **do**
  for $i = 1, \ldots, N$ **do**
    sample $(z_i, \kappa_i)_{(j)}$ from $p(z_i, \kappa_i|\boldsymbol{t}, \mu_{(j-1)}, \sigma^2_{(j-1)}, \boldsymbol{\tau}_{(j-1)}, \ell_{(j-1)}, \boldsymbol{z}^\star_{-i}, \boldsymbol{\kappa}^\star_{-i})$, using (4.24), where $\boldsymbol{z}^\star_{-i}$ and $\boldsymbol{\kappa}^\star_{-i}$ contain the most recent samples for $\boldsymbol{z}_{-i}$ and $\boldsymbol{\kappa}_{-i}$;
  end
  sample $(\boldsymbol{\tau}_{(j)}, \ell_{(j)})$ from $p(\boldsymbol{\tau}, \ell|\boldsymbol{y}, \boldsymbol{z}_{(j)})$ using Metropolis-Hastings, proposing a birth, death, or shift of one of the changepoints;
  sample $(\mu_{(j)}, \sigma^2_{(j)})$ from NIG$(\mu_{N_1}, \lambda_{N_1}, \alpha_{N_1}, \beta_{N_1})$, see (4.15), where μ_{N_1}, λ_{N_1}, α_{N_1} and β_{N_1} are calculated conditional on $(\boldsymbol{z}_{(j)}, \boldsymbol{\kappa}_{(j)})$;
end

4.5. Applications

The algorithms described in the previous sections have been applied on computer network flow data collected at Imperial College London, for a single client IP address X, setting $p' = 86{,}400$. In order to show the efficacy of the methods for filtering polling traffic, examples are presented using simulated data, a synthetically fused mixture and some raw Network Flow data.

4.5.1. Simulated data

The performance of the Gibbs sampler in recovering the correct densities for the model in Section 4.4 is first assessed on simulated data. Non-periodic events were simulated from a range of densities of increasing complexity, inspired by the test signals in Donoho and Johnstone (1994), rescaled and shifted to represent probability distributions on $[0, 2\pi)$. Three distributions are used:

i. a step function density with 10 segments, where the changepoints and segment probabilities were sampled from a Uniform$[0, 2\pi)$ and Dirichlet$(1, \ldots, 1)$ respectively,
ii. a heavisine function on $[0, 2\pi)$, $f(y) \propto 6+4\sin(2y)-\text{sign}(y/2\pi-0.3)-\text{sign}(0.72-y/2\pi)$,
iii. a function $f(y) \propto \sum_{j=1}^{11} u_j(1+|(y/2\pi - v_j)/w_j|)^{-4}$ with 11 bumps, with the same choices of Donoho and Johnstone (1994) for the parameters u_j, v_j and w_j, scaled to $[0, 2\pi)$.

3,000 events are simulated from the chosen distributions, and then assigned to a random day of the week, implying $p' = 86{,}400$. Those events are mixed with 2,000 periodic events generated from a wrapped normal distribution with mean $\mu = 5$ and variance $\sigma^2 = 1$ on $[0, 2\pi)$, rescaled and assigned at random to windows of $p = 10$ seconds over one week. Note that the variance of the periodic signal is chosen to be relatively large to make the inferential procedure more complicated. In practical applications, the value of σ^2 is expected to be much smaller.

The results of the Gibbs sampling procedure for estimation of the density of non-polling events, using the model in Section 4.4, are reported in Figure 4.6. The algorithm is able to recover the density with good confidence, even in case of departures from the step function assumption. The densities used in the simulation are within the highest posterior density regions (HDPR) in Figure 4.6. Note that it is not possible to expect the fit of the estimated density to correspond perfectly to the density used to simulate the data, since the simulation is repeated only once, for a sample of size 3,000, and the variability for the wrapped normal component was chosen to be large.

The estimates for the remaining parameters in the simulation using the step function density resulted in $(\hat{\mu}, \hat{\sigma}^2, \hat{\theta}) = (5.0162, 0.9890, 0.4022)$. The performance of the classification algorithm can be assessed using the area under the receiver operating characteristic (ROC) curve, commonly denoted as AUC. For the step function density, the resulting AUC score is 0.8161. For the heavisine function, the estimates of the parameters are $(5.0268, 0.9868, 0.3882)$, and AUC $= 0.8007$. Finally, for the function with bumps, the estimates are $(5.0162, 0.9890, 0.4022)$, and AUC $= 0.9337$. The parameter estimates correspond to the values used in the simulation, and the AUC values are acceptable considering the complexity of the simulation and the fact that $\sigma^2 = 1$.

The computational efficiency of the collapsed Gibbs sampler for the model in Section 4.4 have been evaluated via simulation. Table 4.1 reports the elapsed time for 1000 sweeps of the sampler for different values of N, the number of observed arrival times. The experiments were performed running *python* code on a MacBook Pro 2017 with a 2.3 GHz Intel Core i5 dual-core processor, and the events were generated using the same simulation described in this section, with a step function density with 10 changepoints for the non-polling events.

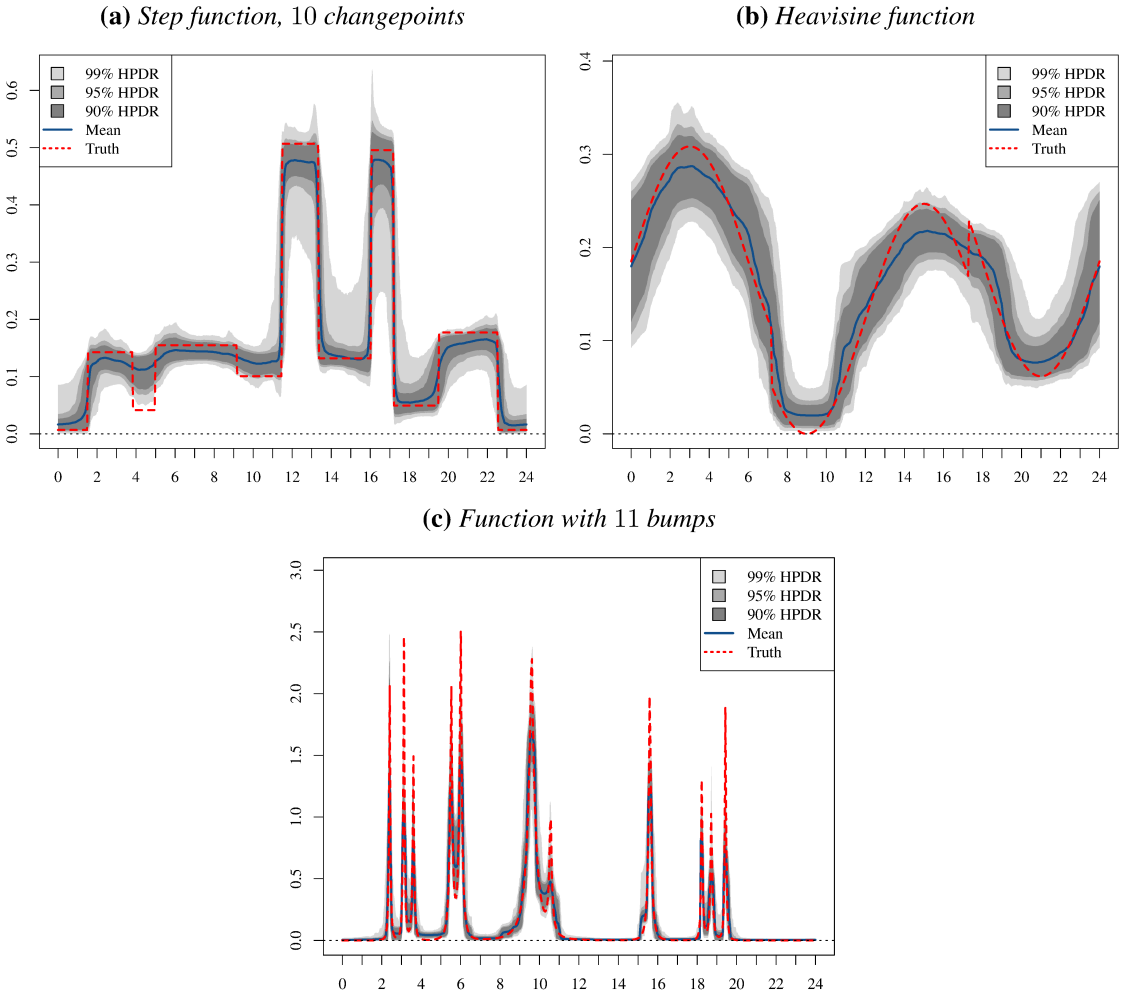


Figure 4.6.: *Estimated daily density of non-polling events for the three simulated examples in Section 4.5.1.*

Table 4.1.: *Elapsed time (in seconds) for* 1000 *sweeps of the collapsed Gibbs sampler for the model in Section 4.4, as a function of the number of observations* N.

N	100	1,000	5,000	10,000	25,000	50,000
Time	1.82	10.68	52.01	122.58	292.88	729.04

4.5.2. Synthetically labeled data: a mixture of automated and human connections

A fusion of two different network edges is considered: first, the activity between the client X and the Dropbox server 108.160.162.98, found to be strongly periodic at period $p \approx 55.66s$, with associated p-value < 0.0001; and second, the activity between the client X and the Midasplayer server addresses 217.212.243.163 and 217.212.243.186, which exhibit activity exclusively during the day, relating to a human user playing the popular online game Candy Crush. Seven days of data

starting from the first observation time on each edge were used in the present analysis, resulting in $32{,}865$ Dropbox events and $4{,}779$ Candy Crush connections. The histograms of daily activity for the two edges are presented in Figure 4.7. Notice that Dropbox is slightly more active at night than during the day, which makes the analysis more difficult. This is not an uncommon behaviour for automated edges, which tend to 'stand down' during the day when a human sits at the machine. On the other hand, Candy Crush events only happen during working hours.

The uniform-wrapped normal mixture model (*cf.* Section 4.3) fitted to the fused data using the EM algorithm quickly converges to the parameter estimates $(\hat{\mu}, \hat{\sigma}^2, \hat{\theta}) = (4.3376, 0.4059, 0.8585)$. The same results are obtained using different initialisation points and comparing different convergence criteria. Given the output of the EM algorithm, it is possible to filter the connections, keeping those such that $\zeta_{i(0,0)} > \sum_{k=-\infty}^{\infty} \zeta_{i(1,k)}$ at the final iteration, where the infinite sum is in practice truncated to a suitable level. These filtered events are those which would be assigned to the uniform (non-periodic) component of the mixture in (4.7). In total, $2{,}818$ wrapped times were classified as non-periodic, and $2{,}386$ of these are connections to Candy Crush servers, resulting in a false positive rate FPR $= 0.013$ and false negative rate FNR $= 0.501$. Note that it is not surprising that approximately 50% of the Candy Crush edges are missed, because these fall into the high density area of the wrapped normal by chance, being approximately uniform on the p-clock.

The results from the EM algorithm were then compared to the inferences obtained from the posterior distribution of the parameters $\boldsymbol{\psi}$ using the Bayesian algorithm of Section 4.3.2. The prior parameters were set to the uninformative values $\mu_0 = \pi, \lambda_0 = 1, \alpha_0 = \beta_0 = \gamma_0 = \delta_0 = 1$, although given the large quantity of data available, the choice of the prior is in practice not influential on the results of the procedure. The resulting mean of the posterior distribution for $\boldsymbol{\psi}$ is $\hat{\boldsymbol{\psi}} = (\hat{\mu}, \hat{\sigma}^2, \hat{\theta}) = (4.3375, 0.4064, 0.8583)$, almost identical to the result obtained using the EM algorithm. This is expected, since the two methods represent two different inferential approaches for the same model. Very similar results are also obtained when filtering the data. For event i, let $\hat{z}_i$ be the Monte Carlo estimate of z_i; then classifying events as non-periodic if $\hat{z}_i < 0.5$ yields $2{,}810$ events, and $2{,}377$ of those are Candy Crush connections, corresponding to FPR $= 0.013$ and FNR $= 0.502$. In practice, for the uniform-wrapped normal model, it is recommended to use the EM algorithm, which converges faster than the Bayesian Markov Chain Monte Carlo procedure, providing, as expected, equivalent results.

Finally, it is of interest to see whether the classification performance can be improved using the extended model presented in Section 4.4. The algorithm was initialised from the output of the EM algorithm, and the additional parameters were set to the uninformative values $\nu = 0.1$ and $\eta = 1$, although again the algorithm is robust to different starting points. The resulting posterior mean estimates of wrapped normal distribution parameters are $(\hat{\mu}, \hat{\sigma}^2) = (4.362, 0.376)$ and $\hat{\theta} = 0.8506$ for the mixing proportion, which are slightly different from the previous analysis; in particular, the variance is lower. The estimated daily distribution of the non-periodic arrival times is plotted

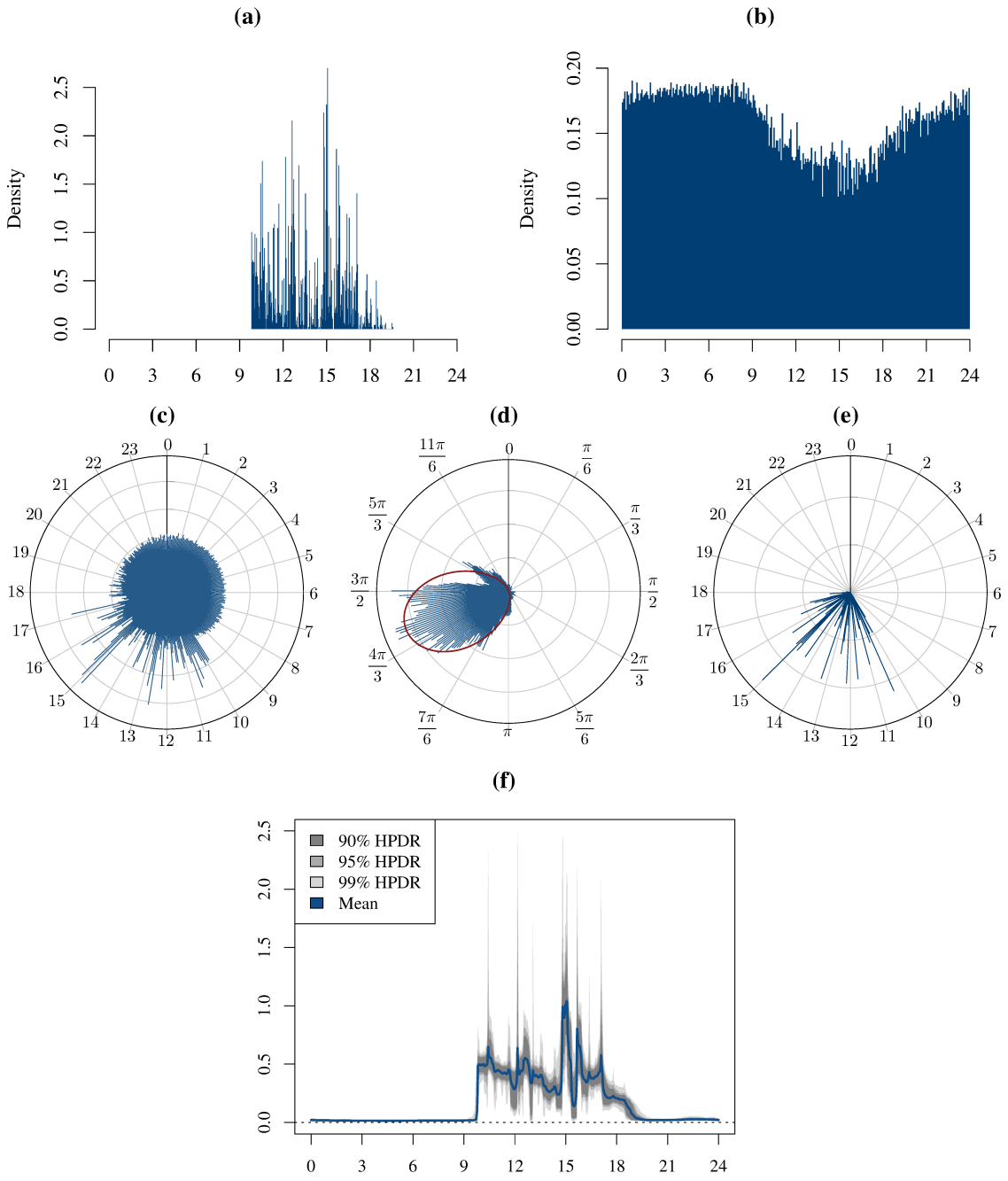


Figure 4.7.: *Analysis on the synthetic Dropbox-Candy Crush data set. Top panel: histogram of the daily arrival times y_i of the Candy Crush (left) and Dropbox (right) events (bin size: 5 minutes). Middle panel: polar histogram of the daily arrival times for the mixed data (left), polar histogram of the wrapped arrival times x_i with period $p = 55.66$ for the filtered periodic events and estimated wrapped normal density (middle), and histogram of the daily arrival times for the filtered non-periodic events (right). Bottom: estimated daily density of non-periodic events.*

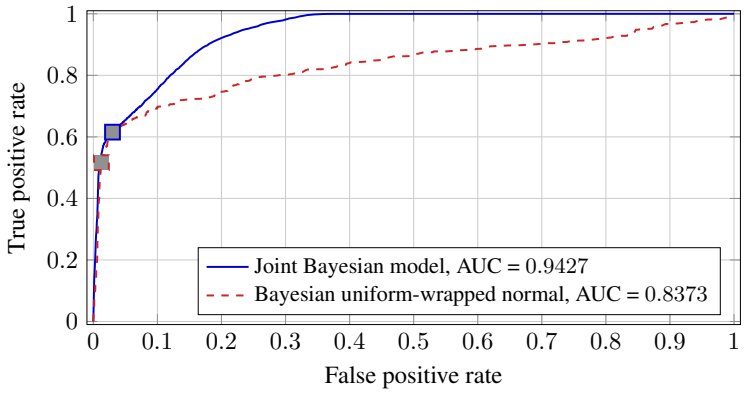


Figure 4.8.: *ROC curves and AUC values evaluating the performance of two methods for classification of human events: Bayesian uniform-wrapped normal mixture (Section 4.3.2) and joint Bayesian model (Section 4.4). The grey squares correspond to the threshold* 0.5.

in Figure 4.7. Note that its mean almost perfectly reproduces the histogram of the daily arrival times of the Candy Crush events. The estimated density has been obtained by sampling from the posterior distribution $h|\boldsymbol{\tau}, \ell, \boldsymbol{y}, \boldsymbol{z}$ for each iteration of the Gibbs sampler, which has known form under the conjugate prior (4.20), and then averaging the density across the iterations. In this case, $3{,}947$ filtered events are labelled as non-periodic ($\hat{z}_i < 0.5$), with $2{,}948$ true positives, corresponding to FPR $= 0.030$ and FNR $= 0.383$. The resulting histogram of the filtered data is plotted in Figure 4.7. The posterior distribution for the number of changepoints in the human density is approximately normally distributed around the value $\ell = 28$, which roughly corresponds to one changepoint per hour of the day.

The algorithms proposed in the chapter can be more efficiently compared for classification purposes using a ROC curve for different values of the threshold for $\hat{z}_i$. The plot for this example is reported in Figure 8.4, and it clearly shows that the proposed methodologies correctly classify a significant proportion of the events very well, with low false positive rates for the threshold 0.5. Furthermore, including the daily arrival times $\boldsymbol{y}$ in the model is clearly beneficial. For practical applications, it is recommended to choose a threshold that guarantees low false positive rates for detection of human events: in this example, 0.5 seems an appropriate choice.

4.5.3. Real data: Imperial College NetFlow

In this example, the activity between a client Y and the server IP 13.107.42.11, used by the software *Outlook*, is analysed. The arrival times refer to a time period between August, 2017 and November, 2017, and 7 days of activity after the first observation were considered. The daily distribution of the activity on the edge is reported in Figure 4.9. A total number of $7{,}583$ connections were recorded. It can be observed from the histogram that the activity on the edge is almost entirely automatic, but the number of connections slightly increases during working hours compared to the

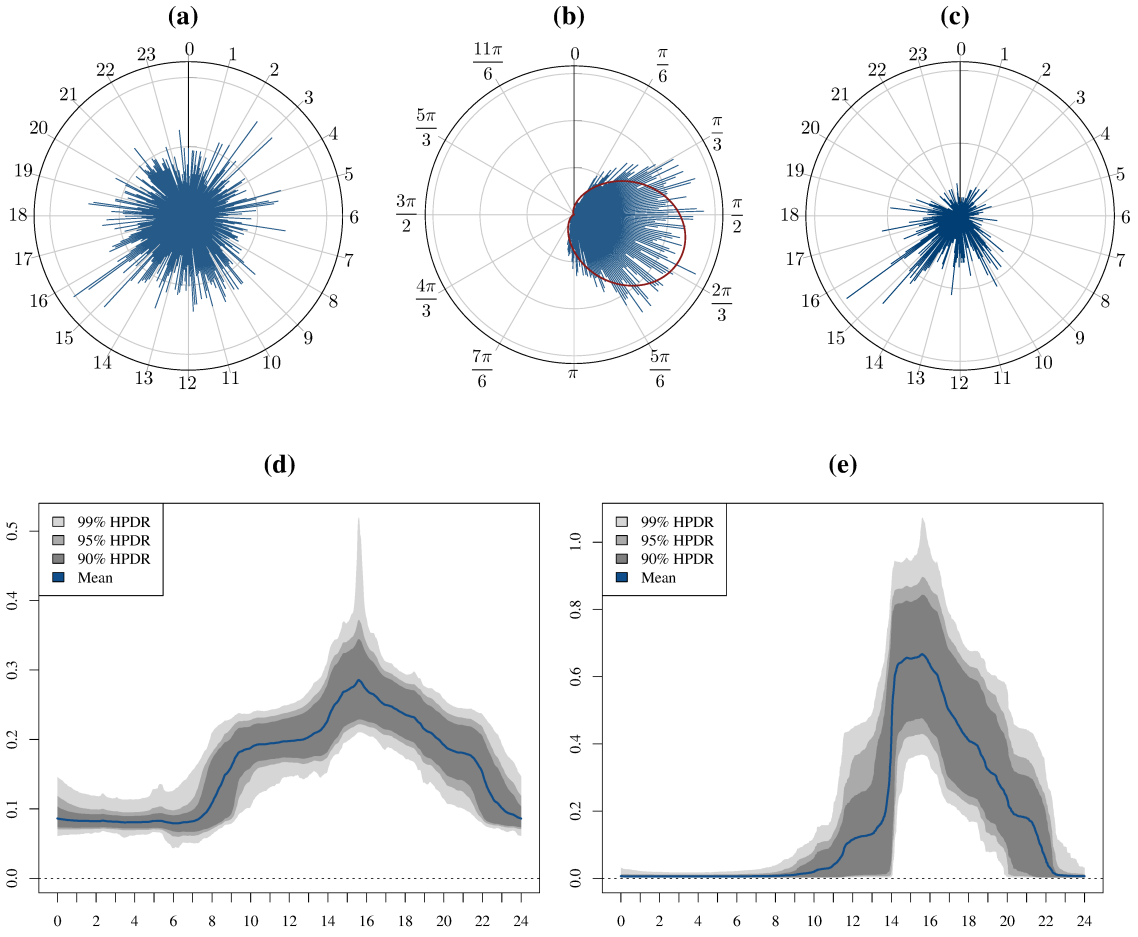


Figure 4.9.: *Analysis on the edge $Y \rightarrow 13.107.42.11$ (Outlook). Top: polar histograms of the daily arrival times y_i of the events (left), and of the wrapped arrival times x_i for periodicity $p \approx 8s$ with fitted wrapped normal density (middle) and finally of the filtered non-periodic events (right). Bottom: resulting estimated daily density of non-periodic events (left) from applying the algorithm once, and then the estimated daily density of human events (right) obtained from re-applying the algorithm with the same periodicity on the filtered events from plot (c).*

night (compared with the dip observed in other automated services like Dropbox). This suggests a mixture between human activity and polling behaviour on this edge, which is further supported by the nature of the software. The arrival times on the edge have been found to be strongly periodic at period $p \approx 8s$, with an associated g-test p-value $< 10^{-7}$.

The uniform-wrapped normal mixture model (*cf.* Section 4.3), with period $8s$, fitted using the EM algorithm converges to the parameter estimate $(\hat{\mu}, \hat{\sigma}^2, \hat{\theta}) = (1.872, 0.670, 0.714)$. In this case study, 1,246 of the 7,583 events were assigned to the uniform (non-periodic) category using the criterion $\zeta_{i(0,0)} > 0.5$. Identical parameter estimates are obtained using the Bayesian mixture model, and classification of the connections as periodic or non-periodic are again almost the same,

with $1{,}232$ connections classified as non-periodic from the model. Furthermore, most of the activity in the filtered events is concentrated in working hours, even though this is not explicitly encouraged by this model. This is promising since the algorithm has been able to recover human-like activity from an edge that apparently seems almost entirely automated.

Next, the Gibbs sampler was used to infer the parameters of the joint Bayesian model (*cf.* Section 4.4). The same prior parameter values as the previous section were used. The convergence of the sampler to the correct target is again almost immediate. The number of non-periodic connections was estimated as $1{,}430$. The resulting posterior mean for the parameters of the wrapped normal distribution for the polling component is $(\hat{\mu}, \hat{\sigma}^2) = (1.885, 0.6249)$ and $\hat{\theta} = 0.6935$ for the mixing proportion. The daily distribution of the non-periodic connections is reported in Figure 4.9 and displays a strong diurnal pattern, suggesting human behaviour has been classified well.

However, it is also evident that, in this example, the algorithm classifies as human a proportion of connections occurring during the night. Potential issues that can arise are multiple periodicities or phase shifts within the same data stream. A possible solution would be to iteratively repeat the analysis on the filtered non-periodic events until no significant short term periodicities are obtained using the g-test. In this example, repeating the analysis with period $8s$ allows the residual automated activity to be filtered out, thereby obtaining an estimated daily distribution which is entirely consistent with human-like behaviour, shown in Figure 4.9e. After this last stage of the analysis, only 181 events are retained as human-generated, corresponding to $\approx 2.5\%$ of the initial $7{,}583$ events. This proportion is consistent with results obtained in previous studies on computer network data (Price-Williams, Heard, and Turcotte, 2017).

The performance of the algorithm for filtering polling activity can also be assessed by comparing the model fit to both the filtered and unfiltered event streams when applying the nonparametric Wold process model of Price-Williams and Heard (2019), which has been shown to be suitable for human-like events in computer network traffic. There, a counting process of human-generated events is modelled with a conditional intensity represented as a step function with an inferred number of changepoints. If $y_1, y_2, \ldots$ are the event times of such a counting process $Y(t)$, the conditional intensity has the form

$$\lambda_Y(t) = \lambda + \sum_{j=1}^{\ell} \lambda_j \mathbb{I}_{[\tau_{j-1}, \tau_j)}(t - y_{Y(t)}), \tag{4.28}$$

where $0 \equiv \tau_0 < \tau_1 < \ldots < \tau_\ell$ are a finite sequence of changepoints and $\lambda_1 > \ldots > \lambda_\ell$ are a decreasing sequence of corresponding step heights, representing the fall in intensity experienced as the waiting time increases between the current time t and the most recent event $y_{Y(t)}$. In contrast, periodic network events are not self-exciting; their conditional intensity would decrease immediately after an event, and only increase when the next periodic signal is anticipated.

Price-Williams and Heard (2019) used predictive p-values to assess model fit of the intensity

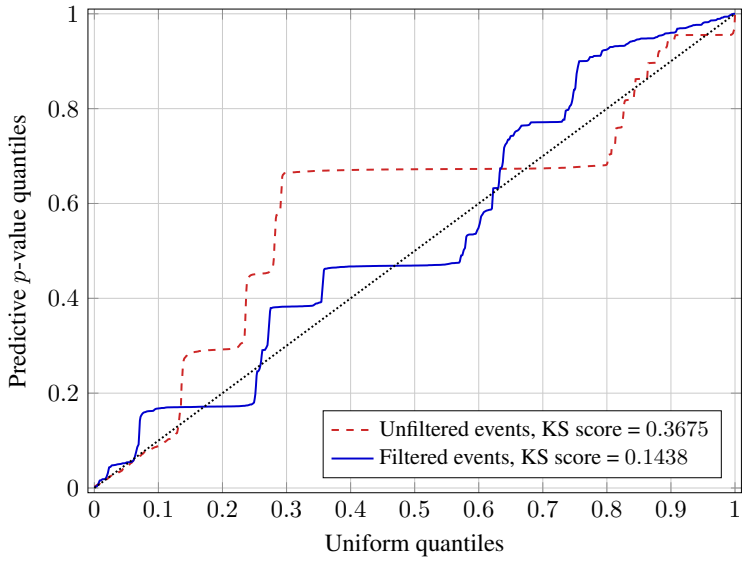


Figure 4.10.: *Uniform Q-Q plots of predictive p-values on the unfiltered and filtered non-periodic events, obtained using the nonparametric Wold process model of Price-Williams and Heard (2019), and corresponding Kolmogorov-Smirnov scores.*

model (4.28). Defining $y_0 \equiv 0$ and the compensator function $\Lambda(t) = \int_0^t \lambda_Y(s)\mathrm{d}s$, a lower-tail p-value of the i-th waiting time is

$$p_i = 1 - \exp\left[-\{\Lambda(y_i) - \Lambda(y_{i-1})\}\right]. \tag{4.29}$$

Figure 4.10 reports the Q-Q plot of the distribution of predictive p-values (4.29) obtained using the first 4 days of observations as training data and the remaining days as test data, for both the unfiltered and filtered non-periodic events from Figure 4.9c. The distribution of the p-values clearly improves when the filtered non-periodic events are used. The Kolmogorov-Smirnov (KS) score, based on the maximum absolute difference between the empirical and theoretical CDFs, significantly decreases for the filtered events, reaching a value which is consistent with the results obtained by Price-Williams and Heard (2019) on Imperial College NetFlow data.

This example strikingly shows characteristics of real computer network traffic data: the activity on automatic edges only slightly increases during the day due to the presence of a human at the machine. Despite these difficulties, the algorithm was successfully able to derive a reasonable distribution for the human events.

4.6. Conclusion and discussion

In this chapter, a statistical framework for classification of arrival times in event time data has been proposed. The methodology was motivated by application to computer network modelling for

cyber-security. In particular, the filtering methodology developed in Heard, Rubin-Delanchy, and Lawson (2014) has been extended to network edges that present a mixture of human and automated polling activity, in order to prevent the loss of information caused by totally removing a seemingly automated edge from the analysis. This has initially been achieved using a simple mixture model based on a uniform distribution and a wrapped normal distribution on the unit circle. Frequentist and Bayesian algorithms for the estimation of the parameters have been presented. The model has then been extended to include available information on the daily arrival times of the events, demonstrating significant performance improvements on synthetic data sets with known labels. Bayesian inference is straightforward since simple conjugate distributions are used, and therefore minimal adaptation is required from the user. Synthetically fused and real data examples show that the model is able to recover a significant amount of the non-periodic activity and its distribution.

After fitting the model, the estimated values of the parameters can be used for instantaneous estimation of $z_{i'}$ for classification of future arrival times $t_{i'}$. Depending on the application, it might be necessary to update the parameter estimates from time to time as more data become available. The Bayesian framework naturally allows for prior-posterior updates, where the estimated posterior parameters can be used as prior hyperparameters when new data are available (Bernardo and Smith, 1994). In that case, it would be necessary to perform the inferential procedure again, including the newly observed arrival times, and possibly removing a subset of the old observations to both fix the overall computational cost of the inferential procedure, which otherwise grows in N as shown in Table 4.1, and allow for any adaptation in the model.

The methodology proposed in this chapter generically fits within the literature on Bayesian model based clustering (for example, Lau and Green, 2007), where MCMC methods are commonly used for inference on the latent allocations and model parameters (for example, West, Müller, and Escobar, 1994; Richardson and Green, 1997). The proposed model complements and extends this literature, providing a Bayesian framework for classification of event time data, when a mixture of periodic and non-periodic events is observed.

The proposed methodology is also flexible, and the parametric distributions used in this chapter could be changed according to the application. For example, the parametrisation of the wrapped component could be changed to a double exponential, also known as Laplace distribution, allowing for thicker tails. In this case, the EM algorithm for a double exponential-uniform mixture would also have closed form solution. Also, the value of p' could be modified to obtain densities at different resolutions. For example, setting $p' = 7 \times 86{,}400$ would give a weekly density for the non-periodic events, more flexible than the daily density used in the experiments in this chapter.

4.6.1. Limitations and future work

Further possible extensions of the model could allow explicit accounting for phase shifts, using mixtures of wrapped normals with shared variances for the automated component, or allowing for

changepoints in the mean μ of the wrapped normal distribution, accounting for the arrival order of each x_i. Furthermore, the case of multiple periodicities could be considered, using tests for multiple polling frequencies, for example Siegel (1980), yielding periodicities $p_1, \ldots, p_K$, and obtaining a mixture with multiple transformations $x_{ik} = (t_i \bmod p_k) \times 2\pi/p_k,\ k = 1, \ldots, K$. The model could also be adapted to allow for fixed duration polling, and alternative distributions could also be considered for the automated component. In principle, the non-periodic part of the model could also be further extended. An even more general framework for density estimation in a Bayesian setting is the Dirichlet process mixture (Escobar and West, 1995). Inference in this case is cumbersome, but algorithms exist for relatively fast implementation (Neal, 2000).

Within the application of computer network security, improvements might be achieved by including host specific information; unified data sets of this type have recently become available (Turcotte, Kent, and Hash, 2018). Finally, the algorithm can be applied independently on multiple computer network edges, or, in principle, the same human density could be fitted for all edges emanating from the same source node, but allowing for different periodicities for traffic on each edge. Unfortunately, the Bayesian MCMC procedure does not scale well in N, as demonstrated by Table 4.1, which makes the network-wide application of such a model difficult. Nevertheless, the proposed methodology provides a valuable contribution for fine-grained analysis of individual edges.

Part II.

Graph clustering

The models for individual events proposed in Part I treated nodes (Chapter 3) and edges (Chapter 4) independently. As discussed in Chapter 1, relationships between different nodes or edges could instead be captured using network-wide models. In this part, attention will be devoted to the task of understanding node similarity. In particular, novel models for graph clustering, also known as community detection, will be proposed. Chapter 5 discusses a novel methodology for simultaneous estimation of the latent dimension and number of communities in stochastic blockmodels, interpreted as random dot product graphs. Chapter 6 extends this methodology to the degree-corrected stochastic blockmodel, proposing a novel framework for spectral clustering on spherical coordinates.

5

Estimation of the latent dimension and communities in stochastic blockmodels

The models proposed in Part I were aimed at scoring individual events observed on the network. In Chapter 3, the node-based Pitman-Yor process represented a very simple, but flexible, modelling assumption for capturing known characteristics of real world computer networks. In Chapter 4, an edge-based methodology to separate periodic and non-periodic events was presented, and successfully applied to NetFlow data. One of the possible drawbacks of both methodologies is that nodes and edges are treated independently, and similarities between entities are not explicitly taken into account within the models. In Chapter 1, it has been discussed that *global* models for network adjacency matrices are well suited for revealing relationships between different nodes or edges. The remainder of this thesis will therefore focus on proposing global models for two important tasks: graph clustering and link prediction. In particular, the next the next two chapters will tackle the problem of clustering nodes based on a graph adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, where $n = |V|$. This problem is also known as *community detection* (Girvan and Newman, 2002).

Statistical models for network adjacency matrices are well studied and understood (see, for example, the surveys Goldenberg et al., 2009; Fienberg, 2012). The remaining chapters of this thesis will mostly focus on *latent space models*. Latent space models (Hoff, Raftery, and Handcock, 2002) represent a flexible and broad approach to statistical analysis of networks: each node i is assigned a latent position $\boldsymbol{x}_i$ in a d-dimensional latent space $\mathbb{X}$, and edges between pairs of nodes are typically generated independently, with the probability of observing a link between nodes i and j obtained through a *kernel* function $\kappa : \mathbb{X} \times \mathbb{X} \to [0,1]$ of the respective latent positions:

$$\mathbb{P}(A_{ij} = 1) = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{5.1}$$

Different ideas and techniques for embedding observed graphs into low dimensional spaces are

explored in the literature (for a survey, see, for example, Cai, Zheng, and Chang, 2018). *Random dot product graphs* (RDPGs, Young and Scheinerman, 2007) are a popular class of latent position models, where $\mathbb{X} \subseteq \mathbb{R}^d$, and the function $\kappa(\cdot)$ is an inner product $\langle \cdot, \cdot \rangle$ on $\mathbb{X} \times \mathbb{X}$. RDPGs are analytically tractable and have therefore been extensively studied; a survey of the existing statistical inference techniques is presented in Athreya et al. (2018).

The *stochastic blockmodel* (SBM) (Holland, Laskey, and Leinhardt, 1983) is the classical statistical model for clustering graphs (Snijders and Nowicki, 1997; Nowicki and Snijders, 2001). Assuming K clusters, or communities, each node is assigned a community membership $z_i \in \{1, \dots, K\}$ with probabilities $\boldsymbol{\theta}$, from the $K-1$ probability simplex. The probability of a link only depends on the community allocations z_i and z_j of the two nodes. Given a symmetric matrix $\mathbf{B} \in [0,1]^{K \times K}$ of inter-community probabilities, then independently

$$\mathbb{P}(A_{ij} = 1) = B_{z_i z_j}. \tag{5.2}$$

The likelihood for an observed symmetric adjacency matrix $\mathbf{A}$ is therefore

$$L(\mathbf{A}|\boldsymbol{z}, \mathbf{B}) = \prod_{1 \leq i < j \leq n} B_{z_i z_j}^{A_{ij}} (1 - B_{z_i z_j})^{1 - A_{ij}}. \tag{5.3}$$

Stochastic blockmodels have appealing statistical properties, and can well approximate any independent-edge network model if the number of communities is sufficiently large (Bickel and Chen, 2009; Wolfe and Olhede, 2013).

Limits and bounds for community detection in stochastic blockmodels, from an information-theoretic approach, have been extensively studied and explored in the literature (Abbe, Bandeira, and Hall, 2016; Abbe, 2018). SBMs have also been successfully extended in different directions: allowing for multiple overlapping communities (Airoldi et al., 2008), incorporating degree corrections (Karrer and Newman, 2011), capturing node popularity (Sengupta and Chen, 2018), Bayesian models (Peixoto, 2018; van der Pas and van der Vaart, 2018) and dynamically evolving models (Xu and Hero III, 2013; Matias and Miele, 2017; Ludkin, Eckley, and Neal, 2018; Pensky and Zhang, 2019). Stochastic blockmodels can also be easily represented as random dot product graphs: each community is assigned a latent position $\boldsymbol{\mu}_k \in \mathbb{R}^d$, $k = 1, \dots, K$, which is common to all the nodes belonging to the cluster, and $\mathbf{B}$ is obtained from the inner products of those positions. Hence, in this framework, $d = \operatorname{rank}(\mathbf{B}) \leq K$.

Spectral clustering techniques (von Luxburg, 2007) use of the spectrum of the adjacency or Laplacian matrix to estimate the latent position, and cluster the nodes in the estimated lower dimensional space, usually via k-means (Ng, Jordan, and Weiss, 2001). The embeddings based on the eigendecomposition of the adjacency and Laplacian matrices are usually denoted as adjacency and Laplacian spectral embeddings (ASE and LSE), and will be described in details in Section 5.1. Spectral clustering provides a consistent statistical estimation procedure for the latent positions and

communities in SBMs (Rohe, Chatterjee, and Yu, 2011; Sussman et al., 2012; Lei and Rinaldo, 2015) and more generally in random dot product graphs (Tang, Sussman, and Priebe, 2013; Sussman, Tang, and Priebe, 2014). Rubin-Delanchy et al. (2017) directly links adjacency and Laplacian spectral embedding to the generalised random dot product graph (GRDPG), an extension of the RDPG, and advocates for Gaussian mixture modelling (GMM) of the latent positions estimated using spectral methods. More generally, clustering structure in latent space models is also modelled via mixtures of normals (Handcock, Raftery, and Tantrum, 2007). Alternatives to spectral clustering include variational methods (Celisse, Daudin, and Pierre, 2012) and pseudo-likelihood approaches (Amini et al., 2013). SBMs have also been extended to the directed case (Wang and Wong, 1987; Rohe, Qin, and Yu, 2016), and appropriate embeddings for co-clustering, in most cases based on the singular value decomposition (SVD), have been derived in the literature (Dhillon, 2001; Malliaros and Vazirgiannis, 2013; Zheng and Skillicorn, 2015).

One of the practical issues of spectral embedding, and in general most graph embedding algorithms, is that it requires a suitable pre-specified latent dimensionality d (usually $d \ll |V|$) as input, and, subsequently, a suitable number of clusters K, conditionally, crucially, on the previous choice of d. For a practical example of this procedure on a real world network, see Priebe et al. (2019). In general, in spectral clustering, similarly to what practitioners do in principal component analysis (PCA), the investigator examines the scree-plot of the eigenvalues and chooses the dimension based on the location of *elbows* in the plot (Jolliffe, 2002), or uses the *eigengap* heuristic (see, for example, von Luxburg, 2007). Automatic methods for thresholding have also been suggested (Zhu and Ghodsi, 2006; Chatterjee, 2015).

A relevant body of literature is also devoted to methods for the selection of the number of communities in stochastic blockmodels (see, for example, Zhao, Levina, and Zhu, 2011; Newman and Reinert, 2016). Several likelihood-based information criteria have been proposed (for example, Latouche, Birmelé, and Ambroise, 2012; Saldaña, Yu, and Feng, 2017; Hu et al., 2019). Alternative methods include network cross validation (Li, Levina, and Zhu, 2020; Chen and Lei, 2018), tests on the distribution of the eigenvalues of a transformation of the adjacency matrix (Bickel and Sarkar, 2016; Lei, 2016), or combinations of spectral methods and binary segmentation (Ma, Su, and Zhang, 2018). Often, practitioners simply set $d = K$, for some d, assuming that $\mathbf{B}$ has full rank in the stochastic blockmodel framework. Under the full rank assumption, one may estimate $d = K$ as the number of eigenvalues of $\mathbf{A}$ which are larger than $\sqrt{n}$ (Chatterjee, 2015; Lei, 2016). In this chapter, the problem of selecting d is approached from the perspective of variable selection in model based clustering, which is widely studied in the literature (Fowlkes, Gnanadesikan, and Kettenring, 1988; Law, Figueiredo, and Jain, 2004; Tadesse, Sha, and Vannucci, 2005; Raftery and Dean, 2006; Maugis, Celeux, and Martin-Magniette, 2009; Witten and Tibshirani, 2010).

Similarly, the problem of correctly selecting the number of clusters is also common in K-means or GMMs, since it is usually required to specify a number of components in the mixture. Usually

the parameter is chosen by minimising information criteria (for example, AIC or BIC). A widely used selection criterion is the Integrated Classification Likelihood (ICL) of Biernacki, Celeux, and Govaert (2000). Exact versions of the ICL based on the adoption of prior conjugated distributions of the model parameters have been obtained in the literature (for example, Côme and Latouche, 2015; Wyse, Friel, and Latouche, 2017). Numerous other techniques have also been proposed for GMMs with unknown number of components (Mengersen and Robert, 1996; Richardson and Green, 1997; Stephens, 2000; Nobile, 2004; Dellaportas and Papageorgiou, 2006; Miller and Harrison, 2018).

The sequential approach in estimating d and K is suboptimal, since the estimate of K would depend on the estimate of d. Therefore, it is desirable to jointly estimate the two parameters, a problem which is not explored in the literature. This chapter addresses the problem in a Bayesian framework, proposing a novel methodology to automatically select d and K, simultaneously. Techniques for selection of K in GMMs will be incorporated within the spectral embedding framework, allowing for K and d, the number of communities and latent dimension of the latent positions, to be random and learned from the data. A structured Bayesian model for simultaneously inferring the dimension of the latent space, the number of communities, and the community allocations is proposed. The model is based on asymptotic results (Athreya et al., 2016; Rubin-Delanchy et al., 2017; Tang and Priebe, 2018) on the leading components of spectral embeddings, obtained for d fixed and known. The asymptotic theory is combined with realistic assumptions about the remaining components of the embedding, empirically tested and justified on simulated data. Furthermore, extensions to the directed and bipartite case will be discussed.

The proposed model has multiple advantages: the latent dimension d and number of communities K are modelled separately, and the Bayesian framework allows for automatic selection of the two parameters. The model also allows estimation of d even when $d < K$, and gives insights on the goodness-of-fit of the stochastic blockmodel on observed network data, based on the embedding. The method is tested on simulated data and applied to real world computer and transportation networks. It should be noted that Yang et al. (2020) have simultaneously and independently proposed a similar inferential procedure within a frequentist framework.

The chapter is organised as follows: Section 5.1 introduces adjacency spectral and Laplacian embeddings and the GRDPG. The novel Bayesian model for selection of the appropriate dimension of the latent space is discussed in Section 5.2, followed by careful illustration of posterior inference procedures in Section 5.3. Section 5.4 discusses the effects of the curse of dimensionality on the model, and suggests a remedy. Extensions of the model are presented in Section 5.5, and results and applications are finally discussed in Section 5.6.

5.1. GRDPG and spectral embeddings

In this thesis, the stochastic blockmodel will be interpreted as a specific case of a generalised random dot product graph (GRDPG, Rubin-Delanchy et al., 2017), defined below.

Definition 1 (Generalised random dot product graph, GRDPG)**.** *For $d > 0$, let d_+, d_- be non-negative integers such that $d_+ + d_- = d$. Let $\mathbb{X} \subseteq \mathbb{R}^d$ such that $\forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{X}$, $0 \leq \boldsymbol{x}^\top \mathbf{I}(d_+, d_-)\boldsymbol{y} \leq 1$, where*

$$\mathbf{I}(p, q) = \mathit{diag}(1, \ldots, 1, -1, \ldots, -1)$$

with p ones and q minus ones. Let $\mathcal{F}$ be a probability measure on $\mathbb{X}$, $\mathbf{A} \in \{0,1\}^{n\times n}$ be a symmetric matrix and $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top \in \mathbb{X}^n$. Then $(\mathbf{A}, \mathbf{X}) \sim \mathit{GRDPG}_{d_+,d_-}(\mathcal{F})$ if $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \overset{iid}{\sim} \mathcal{F}$ and for $i < j$, independently,

$$\mathbb{P}(A_{ij} = 1 | \boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \mathbf{I}(d_+, d_-)\boldsymbol{x}_j. \tag{5.4}$$

To represent the K-community stochastic blockmodel with inter-community probabilities $\mathbf{B}$ as a GRDPG, $\mathcal{F}$ can be chosen to have mass concentrated on atoms $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^d$ such that $\boldsymbol{\mu}_i^\top \mathbf{I}(d_+, d_-)\boldsymbol{\mu}_j = B_{ij}\ \forall\ i, j \in \{1, \ldots, K\}$. For estimation of the latent positions in SBMs, adjacency spectral embedding (ASE) or Laplacian spectral embedding (LSE) are commonly used (see, for example, Athreya et al., 2018). ASE and LSE are two common techniques to embed the adjacency matrix of an undirected graph into a latent space of dimension d. Suppose $\mathbf{A} \in \{0,1\}^{n\times n}$ is a symmetric adjacency matrix of an undirected graph with n nodes.

Definition 2 (Adjacency spectral embedding – ASE)**.** *For $d \in \{1, \ldots, n\}$, consider the spectral decomposition $\mathbf{A} = \hat{\mathbf{\Gamma}}\hat{\mathbf{\Lambda}}\hat{\mathbf{\Gamma}}^\top + \hat{\mathbf{\Gamma}}_\perp\hat{\mathbf{\Lambda}}_\perp\hat{\mathbf{\Gamma}}_\perp^\top$, where $\hat{\mathbf{\Lambda}}$ is a $d \times d$ diagonal matrix containing the top d eigenvalues in magnitude, in decreasing order, $\hat{\mathbf{\Gamma}}$ is a $n \times d$ matrix containing the corresponding orthonormal eigenvectors, and the matrices $\hat{\mathbf{\Lambda}}_\perp$ and $\hat{\mathbf{\Gamma}}_\perp$ contain the remaining $n - d$ eigenvalues and eigenvectors. The adjacency spectral embedding $\hat{\mathbf{X}} = [\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_n]^\top$ of $\mathbf{A}$ in $\mathbb{R}^d$ is*

$$\hat{\mathbf{X}} = \hat{\mathbf{\Gamma}}|\hat{\mathbf{\Lambda}}|^{1/2} \in \mathbb{R}^{n\times d},$$

where the operator $|\cdot|$ applied to a matrix returns the absolute value of its entries.

Definition 3 (Laplacian spectral embedding – LSE)**.** *For $d \in \{1, \ldots, n\}$, consider the Laplacian matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, $\mathbf{D} = \mathit{diag}(\sum_{j=1}^n A_{ij})$, and its spectral decomposition $\mathbf{L} = \check{\mathbf{\Gamma}}\check{\mathbf{\Lambda}}\check{\mathbf{\Gamma}}^\top + \check{\mathbf{\Gamma}}_\perp\check{\mathbf{\Lambda}}_\perp\check{\mathbf{\Gamma}}_\perp^\top$. The Laplacian spectral embedding $\check{\mathbf{X}} = [\check{\boldsymbol{x}}_1, \ldots, \check{\boldsymbol{x}}_n]^\top$ of $\mathbf{A}$ in $\mathbb{R}^d$ is $\check{\mathbf{X}} = \check{\mathbf{\Gamma}}|\check{\mathbf{\Lambda}}|^{1/2}$.*

The modified Laplacian $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ (Rohe, Chatterjee, and Yu, 2011) is preferred to $\mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ since the eigenvalues of the former lie in $[-1, 1]$, providing a convenient interpretation for disassortative networks (Rubin-Delanchy, Adams, and Heard, 2016).

For consistent estimation of the latent positions in a SBM, interpreted as a GRDPG, Rubin-Delanchy et al. (2017) suggest to fit a Gaussian mixture model with K components to the d-dimensional adjacency or Laplacian spectral embedding.

Algorithm 5.1: Spectral estimation of the SBM (spectral clustering)

Input: adjacency matrix $\mathbf{A}$ (or Laplacian $\mathbf{L}$), latent dimension d, number of communities $K \geq d$.

1 compute spectral embeddings $\hat{\mathbf{X}} = [\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_n]^\top$ (or $\check{\mathbf{X}} = [\check{\boldsymbol{x}}_1, \ldots, \check{\boldsymbol{x}}_n]^\top$) into $\mathbb{R}^d$,

2 fit a Gaussian mixture model to $\hat{\mathbf{X}}$ (or $\check{\mathbf{X}}$) with K components.

Result: return cluster centres $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^d$ and cluster allocations $z_1, \ldots, z_n$.

Intuitively, the estimation procedure proposed by Rubin-Delanchy et al. (2017) holds because, taking a graph with N nodes with d known, and restricting the attention to the first n nodes, with $n < N$, the following central limit theorem holds for the adjacency spectral embedding:

$$N^{1/2}(\mathbf{Q}_N \hat{\boldsymbol{x}}_i - \boldsymbol{x}_i) \longrightarrow \mathbb{N}_d\{\mathbf{0}, \mathcal{S}(\boldsymbol{x}_i)\} \tag{5.5}$$

in distribution as $N \to \infty$, independently for $i = 1, \ldots, n$, where $\mathbf{Q}_N$ is a matrix from the indefinite orthogonal group $\mathbb{O}(d_+, d_-)$, $\mathbb{N}_d(\cdot)$ is the d-dimensional multivariate normal distribution, and $\mathcal{S}(\boldsymbol{x}_i)$ can be analytically computed (Theorem 7, Rubin-Delanchy et al., 2017). A similar result also holds for the Laplacian spectral embedding. For a stochastic blockmodel, interpreted as a GRDPG, (5.5) implies that asymptotically

$$\mathbf{Q}\hat{\boldsymbol{x}}_i \approx \mathbb{N}_d\left(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right) \tag{5.6}$$

for some joint linear transformation given by $\mathbf{Q} \in \mathbb{O}(d_+, d_-)$, where $\boldsymbol{\Sigma}_{z_i}$ is a community-specific covariance matrix. More details about the distortion effect of $\mathbf{Q}$ are given in Sections 2.2 and 2.3 of Rubin-Delanchy et al. (2017).

The result in (5.5) holds for d fixed and known, but in this thesis it is of interest to treat d as a random, unknown parameter. If a m-dimensional embedding is considered, with $m > d$, then asymptotic theory implies an approximate normal distribution with non-zero means and a full covariance within each cluster for the top-d components of the embedding, but *no theoretical results* have been obtained for the remaining $m - d$ columns. It is therefore necessary to propose a model for the remaining part of the embedding, which will be carefully described in Section 5.2, and empirically justified and assessed in Section 5.6.1.

It should be noted that it is only possible to estimate the vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ up to an orthogonal transformation; specifically, for any matrix $\mathbf{Q} \in \mathbb{O}(d_+, d_-)$, $(\mathbf{Q}\boldsymbol{\mu}_{z_i})^\top \mathbf{I}(d_+, d_-)(\mathbf{Q}\boldsymbol{\mu}_{z_j}) = \boldsymbol{\mu}_{z_i}^\top \mathbf{I}(d_+, d_-)\boldsymbol{\mu}_{z_j}$, which implies that the likelihood (5.3) is invariant to any such transformation. This is inconsequential for much the same reason, however, as knowledge of the true transformation

would not change any inferences about the network.

A second source of non-identifiability in the GRDPG interpretation of the SBM is the *uniqueness up to artificial dimension blow-up* (Cape, Tang, and Priebe, 2019): for $(\mathbf{A}, \mathbf{X}) \sim \text{GRDPG}_{d_+,d_-}(\mathcal{F})$, there exists $\tilde{\mathcal{F}}$ on $\mathbb{R}^{\tilde{d}}$, with $\tilde{d} > d$, such that $(\mathbf{A}, \mathbf{X}) = (\tilde{\mathbf{A}}, \tilde{\mathbf{X}})$ in distribution, with $(\tilde{\mathbf{A}}, \tilde{\mathbf{X}}) \sim \text{GRDPG}_{\tilde{d}_+,\tilde{d}_-}(\tilde{\mathcal{F}})$. In the stochastic blockmodel setting, this essentially means that any matrix $\mathbf{B} \in [0,1]^{K\times K}$ with rank d can be obtained as an inner product between latent positions on arbitrarily large dimensions. Therefore, in this chapter, the latent dimension d is strictly interpreted as $d = \text{rank}(\mathbf{B})$.

5.2. A Bayesian model for SBM embeddings

For simplicity, the estimated embeddings will be generically denoted as $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times m}$, $\boldsymbol{x}_i \in \mathbb{R}^m$ for some m, $d \leq m \leq n$. In this chapter, m is always assumed to be fixed and obtained from a preprocessing step. Choosing an appropriate value of m is arguably much easier than choosing the correct d, and, in the proposed model, the correct d can be recovered for any choice of m, as long as $d \leq m$. Let $\mathbf{X}_{:d}$ denote the first d columns of $\mathbf{X}$, and $\mathbf{X}_{d:}$ the $m-d$ remaining columns. The notation $\boldsymbol{x}_{i,:d}$ denotes the first d elements $(x_1, \ldots, x_d)$ of the vector $\boldsymbol{x}_i$, and similarly $\boldsymbol{x}_{i,d:}$ denotes the last $m-d$ elements $(x_{d+1}, \ldots, x_m)$. Suppose a latent dimension d, K communities, and latent community assignments $\boldsymbol{z} = (z_1, \ldots, z_n)$. The latent positions of nodes within each community are assumed to be generated from an m-dimensional community specific Gaussian distribution:

$$\boldsymbol{x}_i | d, z_i, \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}, \boldsymbol{\sigma}^2_{z_i} \sim \mathbb{N}_m \left(\begin{bmatrix} \boldsymbol{\mu}_{z_i} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{z_i} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}^2_{z_i} \mathbf{I}_{m-d} \end{bmatrix} \right). \tag{5.7}$$

Following the results in Section 5.1, the initial components $\boldsymbol{x}_{i,:d}$ are assumed to have unconstrained mean vector $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and positive definite $d \times d$ covariance matrix $\boldsymbol{\Sigma}_k$. In contrast, for $\boldsymbol{x}_{i,d:}$, two constraints are imposed: the mean is an $(m-d)$-dimensional vector of zeros, and the covariance is a diagonal matrix $\boldsymbol{\sigma}^2_k \mathbf{I}_{m-d}$ with positive entries $\boldsymbol{\sigma}^2_k = (\sigma^2_{k,d+1}, \ldots, \sigma^2_{k,m})$. The validation of the model assumptions will be discussed in details in Section 5.6.1. Assuming group-specific covariances $\boldsymbol{\sigma}^2_k$ for $\mathbf{X}_{d:}$ adds extra complexity to the model, and implies that d cannot take the simple interpretation of being the number of dimensions relevant for clustering (see, for example, Raftery and Dean, 2006). However, empirical evidence, discussed in Sections 5.4 and 5.6.1, suggests that the last $m-d$ components of the embedding contain a cluster structure which reflects the communities in the top-d dimensions. Following the discussion in the previous sections, the parameter d actually represents the dimension of the latent positions that generate the network, equivalent to the rank of the unobserved matrices $\mathbb{E}(\mathbf{A})$ and $\mathbf{B} \in [0,1]^{K\times K}$.

In order to complete the model specification, conjugate priors can be placed on the parameters as

follows:

$$\begin{aligned}
(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)|d &\overset{iid}{\sim} \text{NIW}_d(\mathbf{0}, \kappa_0, \nu_0 + d - 1, \boldsymbol{\Delta}_d), \\
\sigma^2_{k,j} &\overset{iid}{\sim} \text{Inv-}\chi^2(\lambda_0, \sigma^2_0),\ j = d+1, \ldots, m, \\
d|\boldsymbol{z} &\sim \text{Uniform}\{1, \ldots, K_\varnothing\}, \\
z_i|\boldsymbol{\theta} &\overset{iid}{\sim} \text{Multinoulli}(\boldsymbol{\theta}),\ i = 1, \ldots, n, \\
\boldsymbol{\theta}|K &\sim \text{Dirichlet}\left(\alpha/K, \ldots, \alpha/K\right), \\
K &\sim \text{Geometric}(\omega),
\end{aligned} \tag{5.8}$$

where, if $n_k = \sum_{i=1}^n \mathbb{1}_k\{z_i\}$ is the size of community k, $K_\varnothing = \sum_{k=1}^K \mathbb{1}_{\mathbb{N}_+}\{n_k\}$ is the number of non-empty communities. In (5.7), NIW_d denotes the d-dimensional normal-inverse-Wishart distribution, and Inv-χ^2 is a scaled inverse chi-square distribution with λ_0 degrees of freedom and scaling parameter σ^2_0. The hyperparameters $\kappa_0, \nu_0, \lambda_0, \sigma^2_0$ and α are assumed to be non-negative, whereas $\omega \in (0, 1)$, and $\boldsymbol{\Delta}_d \in \mathbb{R}^{d\times d}$ is a positive definite matrix. Details about the parametrisation of the distributions used in (5.8) and in this chapter are given in Appendix A.

Yang et al. (2020) have simultaneously and independently proposed a model similar to (5.7) in a frequentist framework, reaching similar assumptions. The conjecture on the distribution of $\mathbf{X}_{d:}$ is essentially the same, except for the choice of the diagonal elements of the cluster-specific covariance matrix: Yang et al. (2020) use a common variance parameter σ^2_k for the last $m - d$ columns of the embedding, whereas a $(m - d)$-dimensional vector of variances $\boldsymbol{\sigma}^2_k$ is used in this chapter. Additionally, as a second difference from Yang et al. (2020), the full model proposed here will also incorporate a second-level community cluster structure on these vectors of variances, which will be introduced in Section 5.4.

A cartoon representation of the model is given in Figure 5.1. Note that the condition $d \leq K$ is explicitly enforced in (5.7). More specifically, $d \leq K_\varnothing$, which avoids an artificial matching between d and K using empty clusters, which are given non-zero probability mass under the Dirichlet-Multinoulli prior on $(\boldsymbol{\theta}, \boldsymbol{z})$. One can also model d and K separately in an analogous way, changing the prior $p(d)$ to, for example,

$$d \sim \text{Geometric}(\delta), \tag{5.9}$$

independently of K and $\boldsymbol{z}$, for $\delta \in (0, 1)$; this will later be referred to as the unconstrained model. The alternative prior (5.9) is particularly useful in practical applications and provides a useful interpretation of d: when $d \leq K$, then $d = \text{rank}(\mathbf{B})$, but when $d > K$, this implies that the observed data might deviate from the stochastic blockmodel assumption, and provides a useful diagnostic for model validation and goodness-of-fit.

The likelihood associated with the spectral embedding $\mathbf{X} \in \mathbb{R}^{n\times m}$ obtained from a stochastic

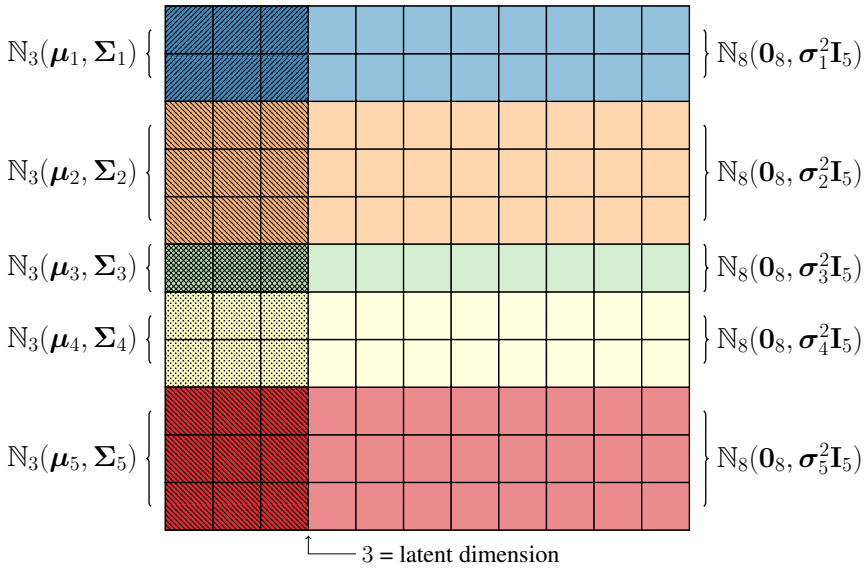


Figure 5.1.: *Cartoon example for the generating process of the embedding of an 11-node stochastic blockmodel GRDPG with $K = 5$ communities and latent dimension $d = 3$.*

blockmodel can be expressed as:

$$L(\mathbf{X}) = \prod_{i=1}^{n} \left\{ \phi(\boldsymbol{x}_{i,:d}; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}) \prod_{j=d+1}^{m} \phi(x_{i,j}; 0, \sigma^2_{z_i,j}) \right\},$$

where $\phi(\cdot)$ denotes the (possibly multivariate) Gaussian density function. Hence, the posterior distribution, up to a normalising constant, has form

$$p(\{\boldsymbol{\mu}_k\}, \{\boldsymbol{\Sigma}_k\}, \{\boldsymbol{\sigma}_k^2\}, \boldsymbol{z}, \boldsymbol{\theta}, K, d|\mathbf{X}) \propto L(\mathbf{X}) \times \prod_{k=1}^{K} \left\{ p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|d) \prod_{j=d+1}^{m} p(\sigma^2_{k,j}|d) \right\} \prod_{i=1}^{n} p(z_i|\boldsymbol{\theta}) p(K) p(d).$$

The $\text{NIW}_d(\mathbf{0}, \kappa_0, \nu_0 + d - 1, \boldsymbol{\Delta}_d)$ prior on the pair $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is conjugate and yields a conditional posterior $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)|\mathbf{X}, \boldsymbol{z}, d \sim \text{NIW}_d(\boldsymbol{m}^{(k)}_{:d}, \kappa_{n_k}, \nu_{n_k} + d - 1, \mathbf{D}^{(k)}_{:d})$, where:

$$\begin{aligned}
\boldsymbol{m}^{(k)}_{:d} &= \textstyle\sum_{i:z_i=k} \boldsymbol{x}_{i,:d} / \kappa_{n_k}, \\
\kappa_{n_k} &= \kappa_0 + n_k, \\
\nu_{n_k} &= \nu_0 + n_k, \\
\mathbf{D}^{(k)}_{:d} &= \boldsymbol{\Delta}_d + \textstyle\sum_{i:z_i=k} \boldsymbol{x}_{i,:d} \boldsymbol{x}^\top_{i,:d} - \kappa_{n_k} \boldsymbol{m}^{(k)}_{:d} \boldsymbol{m}^{(k)\top}_{:d}.
\end{aligned} \tag{5.10}$$

This is a well-known result in Bayesian analysis of the multivariate normal distribution (for a proof, see, for example, Murphy, 2012), and represents a multivariate extension of the univariate normal-inverse gamma model presented in Section 4.3.2. By standard methods for inference in a multivariate Gaussian mixture model with NIW prior, the covariance matrix $\boldsymbol{\Sigma}_k$ can be explicitly integrated out from the posterior to obtain the marginal

$$p(\boldsymbol{\mu}_k|\mathbf{X}, \boldsymbol{z}, d) = t_{\nu_{n_k}}\left(\boldsymbol{\mu}_k \middle| \boldsymbol{m}_{:d}^{(k)}, \frac{1}{\kappa_{n_k}\nu_{n_k}}\mathbf{D}_{:d}^{(k)}\right),$$

where $t_\nu(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the density function of a (possibly multivariate) Student's t distribution with ν degrees of freedom, location $\boldsymbol{\mu}$, and shape matrix $\boldsymbol{\Sigma}$. Henceforth, $\boldsymbol{\mu}_k$ can easily be resampled in a simple Gibbs sampling step, conditional on the actual value of d and on the community allocations $\boldsymbol{z}$. In this chapter, the location vectors $\boldsymbol{\mu}_k$ are collapsed out too, but the distribution is instructive to present other distributional results below, and could be also used if the objective of the analysis is to also recover the explicit form of the latent positions.

In a multivariate Gaussian model with conjugate NIW prior, it is also possible to analytically express the marginal likelihood of the observed data, conditioning on a community specific Gaussian, on the assignments $\boldsymbol{z}$ and on the dimension of the latent space d:

$$p\left(\mathbf{X}_{:d}^{(k)} \middle| d, \boldsymbol{z}\right) = \pi^{-n_k d/2}\frac{\kappa_0^{d/2}|\boldsymbol{\Delta}_d|^{(\nu_0+d-1)/2}}{\kappa_{n_k}^{d/2}|\mathbf{D}_{:d}^{(k)}|^{(\nu_{n_k}+d-1)/2}}\prod_{i=1}^{d}\frac{\Gamma\{(\nu_{n_k}+d-i)/2\}}{\Gamma\{(\nu_0+d-i)/2\}}, \tag{5.11}$$

where $\mathbf{X}_{:d}^{(k)}$ is the subset of rows of $\mathbf{X}_{:d}$ such that $z_i = k$.

Given the Inv-$\chi^2(\lambda_0, \sigma_0^2)$ prior, the posterior for $\sigma_{j,k}^2$ is Inv-$\chi^2(\lambda_{n_k}, s_j^{(k)})$, where $\lambda_{n_k} = \lambda_0 + n_k$, and $s_j^{(k)} = (\lambda_0\sigma_0^2 + \sum_{i:z_i=k} x_{i,j}^2)/\lambda_{n_k}$ (Murphy, 2012). Similar calculations give the full marginal likelihood for the remaining portion of the embedding $\mathbf{X}_{d:}^{(k)}$:

$$p\left(\mathbf{X}_{d:}^{(k)} \middle| d, \boldsymbol{z}\right) = \pi^{-n_k(m-d)/2}\left\{\frac{\Gamma(\lambda_{n_k}/2)}{\Gamma(\lambda_0/2)}\right\}^{m-d}\prod_{j=d+1}^{m}\frac{(\lambda_0\sigma_0^2)^{\lambda_0/2}}{\left(\lambda_{n_k}s_j^{(k)}\right)^{\lambda_{n_k}/2}}. \tag{5.12}$$

Also, note that the probabilities $\boldsymbol{\theta}$ associated to the community assignment can be easily integrated out, resulting in the following marginal likelihood, conditional on K:

$$p(\boldsymbol{z}|K) = \frac{\Gamma(\alpha)\prod_{k=1}^{K}\Gamma(n_k+\alpha/K)}{\Gamma(\alpha/K)^K\Gamma(n+\alpha)}. \tag{5.13}$$

The distributional results presented in (5.10), (5.11), (5.12) (for a proof, see, for example, Murphy, 2012), and (5.13), are the building blocks for the MCMC sampler which is used to make Bayesian inference on the model parameters of interest.

5.3. Inference via MCMC

Since the full posterior is not analytically tractable, inference is performed using MCMC sampling with trans-dimensional moves (Green, 1995). The main objective of the analysis is to cluster the nodes, and therefore the locations $\boldsymbol{\mu}_k$, the variance parameters $\boldsymbol{\Sigma}_k$ and $\boldsymbol{\sigma}_k^2$ and the community probabilities $\boldsymbol{\theta}$ are considered as nuisance parameters and integrated out, similarly to the algorithm presented in Section 4.4.1. Therefore, the remaining parameters for inference are the latent space dimension d, the number of communities K, and the community allocations $\boldsymbol{z}$. Essentially, in the proposed collapsed Metropolis-within-Gibbs sampler (Liu, 1994), four moves are available (Richardson and Green, 1997):

i. Propose a change in the community allocations $\boldsymbol{z}$,
ii. Propose to split (or merge) two communities (changing K),
iii. Propose to create (or remove) an empty community (changing K),
iv. Propose a change in the latent dimension d.

Each move is described in the subsequent four subsections, and the corresponding Metropolis-Hastings acceptance probabilities are calculated. A similar sampler is also used by Wyse and Friel (2012) to estimate the number of clusters in stochastic blockmodels. Note that the latent dimension d could also be treated as a nuisance parameter and marginalised out. This might not always be computationally feasible, especially in cases when m or K are large.

5.3.1. Change in the community allocations

A fully collapsed Gibbs update for each community assignment is available:

$$p(z_i = k|\boldsymbol{z}_{-i}, \mathbf{X}, d, K) \propto p(z_i = k|\boldsymbol{z}_{-i}, d, K)p(\boldsymbol{x}_i|\{\boldsymbol{x}_j\}_{j\neq i:z_j=k}, d). \tag{5.14}$$

In the special case where $d = K_\varnothing$ and $n_{z_i} = 1$, the full conditional distribution for z_i assigns probability one to retaining the same value since the model does not permit $d > K_\varnothing$. Otherwise, from (5.13):

$$p(z_i = k|\boldsymbol{z}_{-i}, d, K) = \frac{n_k^{-i} + \alpha/K}{n - 1 + \alpha}. \tag{5.15}$$

where $n_k^{-i} = n_k - \mathbb{1}_k(z_i)$. Similarly, the remaining term in (5.14), $p(\boldsymbol{x}_i|\{\boldsymbol{x}_j\}_{j\neq i:z_j=k}, d)$, can be obtained as the ratio of marginal likelihoods

$$p(\boldsymbol{x}_i|\{\boldsymbol{x}_j\}_{j\neq i:z_j=k}, d) = \frac{p(\boldsymbol{x}_i, \{\boldsymbol{x}_j\}_{j\neq i:z_j=k}|d)}{p(\{\boldsymbol{x}_j\}_{j\neq i:z_j=k}|d)}. \tag{5.16}$$

The ratio (5.16) can be decomposed as the product of two ratios of marginal likelihoods, separating the top d and last $m - d$ components of the estimated latent positions. Using (5.11), the first ratio,

corresponding to the top d components, is equivalent to the following multivariate Student's t distribution (Murphy, 2012):

$$p(\boldsymbol{x}_{i,:d}|\{\boldsymbol{x}_{j,:d}\}_{j\neq i:z_j=k}, d) = t_{\nu_{n_k^{-i}}}\left(\boldsymbol{x}_{i,:d}\middle|\boldsymbol{m}_{:d}^{(k)}, \frac{\kappa_{n_k^{-i}}+1}{\kappa_{n_k^{-i}}\nu_{n_k^{-i}}}\mathbf{D}_{:d}^{(k),-i}\right), \tag{5.17}$$

where the additional superscript $-i$ denotes a cluster quantity that is computed excluding the allocation z_i of the i-th node. The second ratio, which accounts for the last $m-d$ dimensions, has the form

$$p(\boldsymbol{x}_{i,d:}|\{\boldsymbol{x}_{j,d:}\}_{j\neq i:z_j=k}, d) = \prod_{j=d+1}^{m} t_{\lambda_{n_k^{-i}}}\left(x_{i,j}\middle|0, s_j^{(k),-i}\right). \tag{5.18}$$

5.3.2. Split or merge two communities

To vary the number of communities, move proposals inspired by Sequentially-Allocated Merge-Split sampling (Dahl, 2003; Jain and Neal, 2004) are used. Two indices i and j are sampled at random from the n nodes, and without loss of generality assume $z_i \leq z_j$. If $z_i = z_j$, then the single cluster is split, whereas if $z_i < z_j$ the two clusters are merged. In both move types, node i will remain in the same cluster, denoted $z_i^\star = z_i$. In the merge move, all elements of cluster z_j are reassigned to cluster z_i (with any higher indexed clusters subsequently decremented). For the split move, node j is first reassigned to cluster $K^\star = K+1$ with new allocation $z_j^\star = K^\star$; then, in random order, the remaining nodes currently allocated to cluster z_i are randomly reassigned to clusters z_i or $K^\star$ with probability proportional to their predictive distribution from the generative model (5.16). Denoting the resulting product of renormalised predictive densities from these reallocations by $q(K^\star, \boldsymbol{z}^\star|K, \boldsymbol{z})$, corresponding to the proposal distribution, the acceptance probability for a split move, for example, is

$$\alpha(K^\star, \boldsymbol{z}^\star|K, \boldsymbol{z}) = \min\left\{1, \frac{p(\mathbf{X}|d, K^\star, \boldsymbol{z}^\star)p(d|\boldsymbol{z}^\star, K^\star)p(\boldsymbol{z}^\star, K^\star)}{p(\mathbf{X}|d, K, \boldsymbol{z})p(d|\boldsymbol{z}, K)p(\boldsymbol{z}, K)q(K^\star, \boldsymbol{z}^\star|K, \boldsymbol{z})}\right\}. \tag{5.19}$$

The ratio of densities for $\mathbf{X}$ in (5.19) will depend only upon the rows of the matrix corresponding to the cluster being split (or similarly, merged), and these expressions will decompose as a products of terms for the first d and remaning $m-d$ components (*cf.* (5.17), (5.18)).

5.3.3. Create or remove an empty community

Adding or removing empty communities whilst fixing $\boldsymbol{z}$ corresponds to proposing $K^\star = K+1$ or $K^\star = K-1$ respectively, although the latter proposal is not possible if $K = K_\varnothing$, meaning there

are currently no empty communities. The acceptance probability is simply

$$\alpha(K^\star|K) = \min\left\{1, \frac{p(\boldsymbol{z}|K^\star)p(K^\star)q_\varnothing}{p(\boldsymbol{z}|K)p(K)}\right\},$$

where the proposal ratio $q_\varnothing = 2$ if $K^\star = K_\varnothing$, $q_\varnothing = 0.5$ if $K = K_\varnothing$ and $q_\varnothing = 1$ otherwise.

5.3.4. Change in the latent dimension

Given a current value d, a new value $d^\star$ is proposed from a density $q(d^\star|d) \propto \xi^{|d^\star - d|}\mathbb{1}_{\mathcal{D}}(d^\star)$ on a neighbourhood $\mathcal{D} = \{\max\{1, d-l\}, \ldots, d-1, d+1, \ldots, \min\{d+l, m\}\}$, typically with $l \leq 5$ and $\xi \in (0,1)$. The acceptance ratio reduces to

$$\alpha(d^\star|d) = \min\left\{1, \frac{p(\mathbf{X}|d^\star, K, \boldsymbol{z})p(d^\star|\boldsymbol{z})}{p(\mathbf{X}|d, K, \boldsymbol{z})p(d|\boldsymbol{z})}\frac{q(d|d^\star)}{q(d^\star|d)}\right\}.$$

Notably, if $d^\star > d$, the ratio $p(\mathbf{X}|d^\star, K, \boldsymbol{z})/p(\mathbf{X}|d, K, \boldsymbol{z})$ only depends on the first $d^\star$ components of the embedding, since the last $m - d^\star$ components remain independent by (5.7).

5.3.5. Inferring communities

Markov Chain Monte Carlo samplers for mixture models with varying number of clusters are well known to be affected by *label switching* (Jasra, Holmes, and Stephens, 2005), since the likelihood is invariant to permutations of the cluster labels. However, the estimated posterior similarity between nodes i and j, $\hat{\pi}_{ij} = \hat{\mathbb{P}}(z_i = z_j|\mathbf{X}) = \sum_{s=1}^{M} \mathbb{1}_{z_i^{(s)}}\{z_j^{(s)}\}/M$ is invariant to label switching. Communities can be estimated from the MCMC chains using the posterior similarity matrix $\{\hat{\pi}_{ij}\}$ and the PEAR method (maximisation of the posterior expected adjusted Rand index, Fritsch and Ickstadt, 2009). The estimated communities $\hat{\boldsymbol{z}}$ are calculated as:

$$\hat{\boldsymbol{z}} = \underset{\boldsymbol{z}^\star}{\operatorname{argmax}}\left\{\frac{\sum_{i<j}\mathbb{1}_{z_i^\star}\{z_j^\star\}\hat{\pi}_{ij} - \sum_{i<j}\mathbb{1}_{z_i^\star}\{z_j^\star\}\sum_{i<j}\hat{\pi}_{ij}/\binom{n}{2}}{\frac{1}{2}[\sum_{i<j}\mathbb{1}_{z_i^\star}\{z_j^\star\} + \sum_{i<j}\hat{\pi}_{ij}] - \sum_{i<j}\mathbb{1}_{z_i^\star}\{z_j^\star\}\sum_{i<j}\hat{\pi}_{ij}/\binom{n}{2}}\right\}.$$

Alternatively, if a configuration with a fixed number of clusters K is required, the clusters can been estimated using hierarchical clustering with average linkage, using $1 - \hat{\pi}_{ij}$ as distance measure (Medvedovic, Yeung, and Bumgarner, 2004). The n nodes are initially allocated to n different clusters, each of size 1, which are then iteratively merged until only K clusters are remaining. Clusters are merged according to their average posterior dissimilarity.

5.4. Second-level clustering of community variances

Empirical analyses of simulations from the stochastic blockmodel show that identifying and clearly separating the K clusters in $\mathbf{X}_{d:}$ is particularly difficult for the sampler in settings when $d \ll m$.

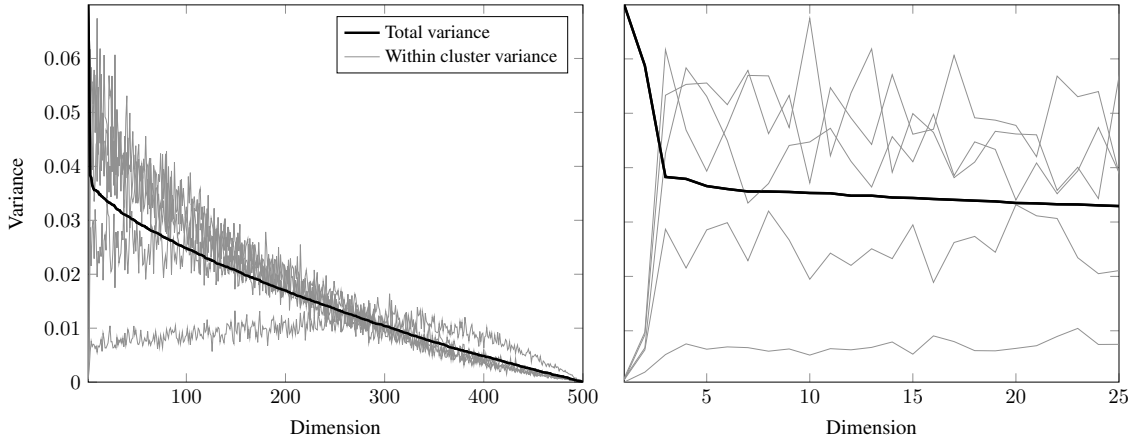


Figure 5.2.: *Empirical within-block variance and total variance for the* 500*-dimensional adjacency embedding obtained from a simulated* 5*-block SBM with* $d = 2$*,* $n = 500$*, means* $\boldsymbol{\mu}_1 = [0.7, 0.4], \boldsymbol{\mu}_2 = [0.1, 0.1], \boldsymbol{\mu}_3 = [0.4, 0.8], \boldsymbol{\mu}_4 = [-0.1, 0.5]$ *and* $\boldsymbol{\mu}_5 = [0.3, 0.5]$*, and* $n_k = 100$ *for* $k = 1, \ldots, K$*. The right panel is the left panel plot zoomed in to the first 25 dimensions.*

The problem is particularly evident when $m = n$ and d is small. In this case, it has been assessed empirically that the within-cluster variance of the true communities in simulated datasets seems to converge to similar values, such that $\sigma^2_{k,j} \approx \sigma^2_{\ell,j}$ for $j \gg d$ and $k \neq \ell$. Therefore, when m is large enough, the selected model tends to be under-specified: the correct dimension d is identified, but the true number of communities K is underestimated. This is also one of the main reasons why it is not advisable to directly fit a Gaussian mixture model on $\mathbf{X} \in \mathbb{R}^{n \times m}$ and allow K to be random, ignoring the role of d.

The problem is illustrated in Figure 5.2, which shows the within-cluster variance for each dimension, obtained from performing ASE with $m = 500$ from a simulation of a stochastic blockmodel with $n = 500$ nodes, containing $K = 5$ communities with well-separated mean locations, with true latent dimension $d = 2$. Each community was assigned the same number of nodes. In Figure 5.2, the clustering structure $\boldsymbol{z}$ is assumed to be known. More details about simulations of SBMs are given in Section 5.6.1. In the plot, the within-cluster variances of three of the five communities of the simulated graph fluctuate around the same values on each dimension larger than d. For a dimension larger than approximately 150, four of the five clusters have approximately the same variance on the subsequent dimensions. Therefore, when $m = n$, the MCMC sampler selects the MAP estimate $\hat{K} = 2$ for parsimony, and increases the variance of the Gaussian distributions on the first two dimensions, on which the clusters are well separated.

The solution proposed here is to assume shared variance parameters between some of the clusters for dimensions larger than d. Specifically, each community $k \in \{1, \ldots, K\}$ is assigned a second-level cluster allocation $v_k \in \{1, \ldots, H\}$, with $H \leq K$. If $v_k = v_\ell$, then for $j > d$, $\sigma^2_{k,j} = \sigma^2_{\ell,j}$.

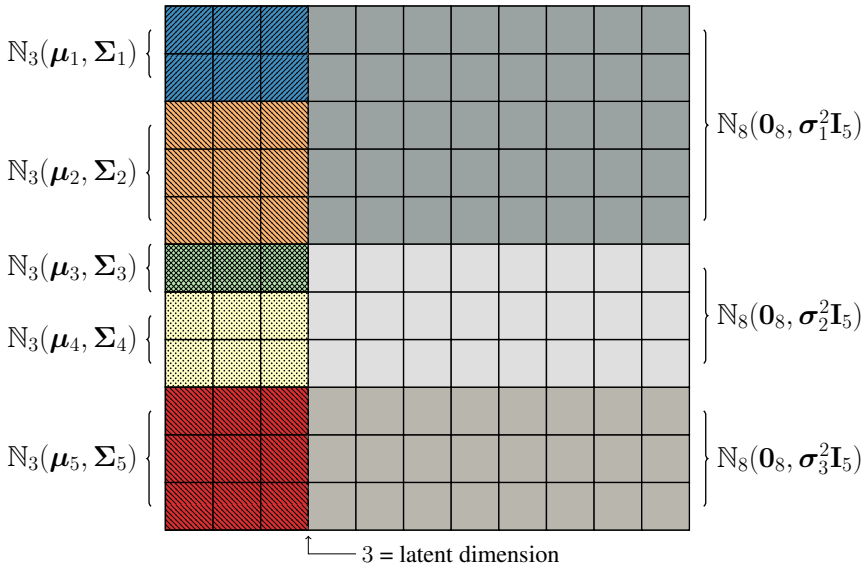


Figure 5.3.: *Cartoon for the generating process of the embedding of an 11-node stochastic blockmodel GRDPG with $K = 5$ communities and latent dimension $d = 3$, with common variance for communities 1-2 and 3-4 on the right hand side of the matrix.*

Formally,

$$\begin{aligned}
\boldsymbol{x}_i | d, K, z_i, v_{z_i} &\sim \mathbb{N}_m \left(\begin{bmatrix} \boldsymbol{\mu}_{z_i} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{z_i} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}^2_{v_{z_i}} \mathbf{I}_{m-d} \end{bmatrix} \right), \\
v_k | K, H &\sim \text{Multinoulli}(\boldsymbol{\phi}),\ k = 1, \ldots, K, \\
\boldsymbol{\phi} | H &\sim \text{Dirichlet}\left(\beta/H, \ldots, \beta/H\right), \\
H | K &\sim \text{Uniform}\{1, \ldots, K\}.
\end{aligned}$$

Essentially, the vector $\boldsymbol{v} = (v_1, \ldots, v_K)$ defines a *clustering of communities*. Figure 5.3 depicts a cartoon example of this extended model. Note that if $H = 1$, all the communities are assigned to the same second-level cluster, and the problem of selecting d essentially reduces to an *ordinal* version of the feature selection problem in clustering (Raftery and Dean, 2006). Also, if $H = 1$, then there is no information about the clusters in the last $m - d$ components of the embedding.

Under this extended model, the posterior distribution for $\sigma^2_{j,k}$ changes due to the aggregation of communities in the second level. Under the Inv-$\chi^2(\lambda_0, \sigma_0^2)$ prior, the posterior is Inv-$\chi^2(\lambda_{n_{\bullet k}}, s_j^{(\bullet k)})$, where $n_{\bullet k} = \sum_{\ell : v_\ell = k} n_\ell$ and

$$s_j^{(\bullet k)} = \frac{1}{\lambda_{n_{\bullet k}}} \left(\lambda_0 \sigma_0^2 + \sum\nolimits_{i : z_i = k} x_{ij}^2 \right).$$

Calculations similar to (5.12) give the correct form of the marginal likelihood for the right hand side of the matrix, restricted to a given value of v_k. Clearly, $\boldsymbol{\phi}$ can be again marginalised out, yielding the marginal likelihood

$$p(\boldsymbol{v}|H) = \frac{\Gamma(\beta)\prod_{h=1}^{H}\Gamma(\sum_{k=1}^{K}\mathbb{1}_h\{v_k\}+\beta/H)}{\Gamma(\beta/H)^H\Gamma(K+\beta)}.$$

The MCMC sampler described in Section 5.3 must be slightly adapted. For the Gibbs sampling move in Section 5.3.1, the product of univariate Student's t densities in (5.18) is modified using the appropriate $(\lambda_{n_{\bullet k}}, s_j^{(\bullet k)})$ pair. For the change in dimension, $p(\mathbf{X}|d, K, \boldsymbol{z}, \boldsymbol{v})$ should be computed using the shared variances and the allocations $\boldsymbol{v}$. When an empty community is proposed, as in Section 5.3.3, the ratio $p(\boldsymbol{v}^\star|K)/p(\boldsymbol{v}|K)$ must be added, limited to the second level allocation of the additional community. The value v_k for the proposed empty cluster can be simply chosen at random from $\{1,\ldots,H\}$. Finally, for the split-merge move in Section 5.3.2, if $z_i = z_j$ for the two selected nodes, then $v_{z_i} = v_{z_j}$ after the split move. Alternatively, if $z_i \neq z_j$, then the new value of v_k is sampled at random from v_{z_i} and v_{z_j}.

Finally, three additional moves are required: resampling the second-level cluster allocations $\boldsymbol{v}$ using a Gibbs sampling step; proposing a second-level split-merge move; and adding or removing an empty second-level cluster. When $\boldsymbol{\phi}$ and the parameters of the Gaussian distributions are marginalised out, the second-level allocations are resampled according to the following equation:

$$p(v_k = h|\boldsymbol{v}_{-k}, \mathbf{X}, \boldsymbol{z}, d, K) \propto p(v_k = h|\boldsymbol{v}_{-k}, K)p\left(\mathbf{X}_{d:}^{(k)}\middle|\left\{\mathbf{X}_{d:}^{(\ell)}\right\}_{\ell\neq k:v_\ell=h}, v_k = h, d, K\right), \tag{5.20}$$

where the independence assumption between $\mathbf{X}_{:d}^{(k)}$ and $\mathbf{X}_{d:}^{(k)}$ is used. Similarly to (5.15):

$$p(v_k = h|\boldsymbol{v}_{-k}, K) = \frac{\sum_{\ell\neq k}\mathbb{1}_h\{v_\ell\}+\beta/H}{K-1+\beta}.$$

The calculations for the second term in (5.20) are similar to (5.16):

$$p\left(\mathbf{X}_{d:}^{(k)}\middle|\left\{\mathbf{X}_{d:}^{(\ell)}\right\}_{\ell\neq k:v_\ell=h}, v_k = h, d, K\right) = \frac{p\left(\mathbf{X}_{d:}^{(k)}, \left\{\mathbf{X}_{d:}^{(\ell)}\right\}_{\ell\neq k:v_\ell=h}\middle| v_k = h, d, K\right)}{p\left(\left\{\mathbf{X}_{d:}^{(\ell)}\right\}_{\ell\neq k:v_\ell=h}\middle| d, K\right)},$$

which can be computed using (5.12). The second-level split-merge move and the proposal of an empty cluster follows the same guidelines in Sections 5.3.2 and 5.3.3.

Potentially, the model could be extended further using the same reasoning: from the plot in Figure 5.2, it is clear that the different clusters begin to share the same variance at different points in the plot. Empirically, all the variances approximately converge to the same values at

large dimensions, and it is therefore possible to identify a $(K-1)$-vector of discrete points in $\{d, d+1, \ldots, m\}$ at which different community variances coalesce. For the plot in Figure 5.2, such vector could be $(d, d, d, 150, n)$, with $d = 2$ and $n = m = 500$.

5.5. Extension to directed and bipartite graphs

A directed graph $\mathbb{G} = (V, E)$ has the property that $(i,j) \in E \implies (j,i) \in E$, meaning the corresponding adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$ is not, in general, symmetric. In a random dot product graph context, it can be assumed that each node has two underlying latent positions $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, characterising, respectively, its behaviour as a source and as a destination of the connection. Therefore, $\mathbb{P}(A_{ij} = 1) = \boldsymbol{x}_i^\top \boldsymbol{y}_j$. For a stochastic blockmodel $\mathbb{P}(A_{ij} = 1) = B_{z_i z_j} = \boldsymbol{\mu}_{z_i}^\top \boldsymbol{\mu}'_{z_j}$ for latent positions $\boldsymbol{\mu}_{z_i}, \boldsymbol{\mu}'_{z_j} \in \mathbb{R}^d$, where the matrix $\mathbf{B} \in [0,1]^{K \times K}$ is in this case asymmetric. The adjacency matrix of a directed graph is traditionally embedded into a lower dimensional latent space using the singular value decomposition (SVD).

Definition 4 (Adjacency embedding of the directed graph – DASE)**.** *Given a directed graph with adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, and an integer $d \in \{1, \ldots, n\}$, consider the singular value decomposition*

$$\mathbf{A} = \begin{bmatrix} \mathbf{U} & \mathbf{U}_\perp \end{bmatrix} \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_\perp \end{bmatrix} \begin{bmatrix} \mathbf{V}^\top \\ \mathbf{V}_\perp^\top \end{bmatrix} = \mathbf{U}\mathbf{S}\mathbf{V}^\top + \mathbf{U}_\perp \mathbf{S}_\perp \mathbf{V}_\perp^\top,$$

where $\mathbf{S} \in \mathbb{R}_+^{d \times d}$ is diagonal matrix containing the top d singular values in decreasing order, $\mathbf{U} \in \mathbb{R}^{n \times d}$ and $\mathbf{V} \in \mathbb{R}^{n \times d}$ contain the corresponding left and right singular vectors, and the matrices $\mathbf{S}_\perp$, $\mathbf{U}_\perp$, and $\mathbf{V}_\perp$ contain the remaining $n - d$ singular values and vectors. The d-dimensional directed adjacency embedding of $\mathbf{A}$ in $\mathbb{R}^d$, is defined as the pair

$$\hat{\mathbf{X}} = \mathbf{U}\mathbf{S}^{1/2}, \quad \hat{\mathbf{Y}} = \mathbf{V}\mathbf{S}^{1/2}.$$

Writing $\mathbf{X} = \mathbf{U}\mathbf{D}^{1/2}$ and $\mathbf{Y} = \mathbf{V}\mathbf{D}^{1/2}$, removing the *hat* notation as in Section 5.2, the rows of $\mathbf{X}$ characterise the activities of each node as a source, and the rows of $\mathbf{Y}$ characterise the same nodes as destinations. The model in (5.7) can be easily adapted to directed graphs. Treating the embeddings $\mathbf{X}$ and $\mathbf{Y}$ as independent, each could be modelled separately using the same Gaussian structure and prior distributions (5.7). Alternatively, the two embeddings could be modelled jointly, assuming three parameters common to both embeddings: the latent dimension d, the number of communities K and the vector of node assignments to those communities, $\boldsymbol{z}$.

In some contexts it will be more relevant to consider different community membership structures for the same set of nodes when considering them as source nodes or destination nodes. In this case, let K denote the number of source communities and K' denote the number of destination

communities; similarly let $\boldsymbol{z}$ denote the assignments of nodes to source communities, and $\boldsymbol{z}'$ the allocations to destination communities. The problem of jointly learning $\boldsymbol{z}$ and $\boldsymbol{z}'$ (as well as d) is commonly known as *co-clustering*, and the corresponding network model is known as the stochastic co-blockmodel (ScBM, Rohe, Qin, and Yu, 2016), or Latent Block Model (LBM, Govaert and Nadif, 2010). Given an asymmetric matrix $\mathbf{B} \in [0,1]^{K \times K'}$, then $\mathbb{P}(A_{ij} = 1) = B_{z_i z'_j}$. From a random dot product graph perspective, it is assumed that $B_{z_i z'_j} = \boldsymbol{\mu}_{z_i}^\top \boldsymbol{\mu}'_{z'_i}$, for some latent positions $\boldsymbol{\mu}_{z_i}, \boldsymbol{\mu}'_{z'_i} \in \mathbb{R}^d$ and $d = \operatorname{rank}(\mathbf{B}) \leq \min(K, K')$.

The Bayesian model for ScBMs can be easily represented as a separate model for $\mathbf{X}$ and $\mathbf{Y}$, of the form given in (5.7), with the latent dimension of the embedding d now the only common parameter. Inference via MCMC can be performed in an equivalent way to the method described in Section 5.3; the only difference is in the expression of the acceptance ratio for a change in the shared latent dimension d, but the procedure can exploit the results obtained in Section 5.3.4, using the fact that

$$p(\mathbf{X}, \mathbf{Y} | d, K, K', \boldsymbol{z}, \boldsymbol{z}') = \prod_{k=1}^{K} p\left(\mathbf{X}_{:d}^{(k)} \middle| d\right) p\left(\mathbf{X}_{d:}^{(k)} \middle| d\right) \prod_{k'=1}^{K'} p\left(\mathbf{Y}_{:d}^{(k')} \middle| d\right) p\left(\mathbf{Y}_{d:}^{(k')} \middle| d\right),$$

where all the marginal likelihoods can be equivalently obtained from (5.11). Furthermore, the model can be appropriately modified when $d \ll m$ to include the second-level cluster allocations proposed in Section 5.4.

Finally, in bipartite graphs, the embeddings could be estimated using DASE (Definition 4) on the rectangular adjacency matrix $\mathbf{A}$ (*cf.* Section 1.1). Note that the ScBM extends trivially to the bipartite graph case, which is essentially a special case of a directed graph, with the cluster configurations for source and destination nodes now inescapably unrelated and each node possessing only one latent representation in $\mathbb{R}^d$.

5.6. Applications and results

The Bayesian latent feature network models described in this chapter have been applied to both simulated and real world network data from undirected, directed and bipartite graphs. The real network data analysed are from networks obtained from the Santander bikes hires in London, the Enron Email Dataset, and the Imperial College London computer network; details are given in the corresponding sections.

The model and MCMC sampler have been tested using different combinations of the hyperparameters, showing robustness to the prior choice. In absence of strong prior information about the community structure, it is advisable to use the usual *uniformative* values $\kappa_0 = \nu_0 = \lambda_0 = \alpha = \beta = 1$, and $\omega = \delta = 0.1$. For the proposal of change in dimension (*cf.* Section 5.3.4), $\xi = 0.8$. Those values have been used as default settings for the MCMC sampler in the next sections. Inferential

performance is sensitive to extreme values of the variance parameters, relative to the prior mean, but otherwise robust. So in practice, the expectation of the prior for the variance parameters should be chosen to be on the same scale as the observed data. The cluster configuration could be suitably initialised using K-means for some pre-specified K, usually chosen according to the scree-plot criterion. The second-level clusters have been initialised setting $H = K$. In order to set the prior covariances to a realistic value, the correlations in the $\boldsymbol{\Delta}_d$ matrices could be set to zero, and the elements on the diagonal of $\boldsymbol{\Delta}_d$ to the average within-cluster variance based on the K-means cluster configuration. Similarly, the prior σ_{0j}^2 values could be set to the total variance on the corresponding column of the embedding.

In Sections 5.6.2 and 5.6.3, the algorithms were initialised using the above guidelines, and run for a total of $M = 500{,}000$ samples with burn-in $25{,}000$, for a number of different chains to check for convergence.

5.6.1. Synthetic data and model validation

In order to validate the model assumptions in (5.7), stochastic blockmodels have been simulated and the fit of the proposed model has been evaluated on the estimated latent positions. A stochastic blockmodel can be simulated starting from a matrix $\mathbf{B} \in [0,1]^{K \times K'}$ containing the probabilities of connection between communities, and a vector $\boldsymbol{\theta}$ of community allocation probabilities. For an undirected graph, $K = K'$ and the constraint $B_{k\ell} = B_{\ell k}$ is imposed; similarly, for directed graphs with a shared cluster configuration (*cf.* Section 5.5), $K = K'$. Each element $B_{k\ell}$ of $\mathbf{B}$ could be generated, for example, from independent beta draws.

The latent dimension d corresponds to the rank of the matrix $\mathbf{B}$, and a random matrix $\mathbf{B}$ generated from independent beta draws has full rank with probability 1. Therefore, to simulate $d < K$ a low-rank approximation of $\mathbf{B}$ must be used to generate the embedding. For undirected graphs, a truncated spectral decomposition with d eigenvectors can be used: $\tilde{\mathbf{B}} = \hat{\boldsymbol{\Gamma}}\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{\Gamma}}^\top$ (recall Definition 2). Similarly, for the directed and bipartite graphs, the truncated SVD is an appropriate approximation: $\tilde{\mathbf{B}} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ (see Definition 4), for d singular vectors. Note that under this low-rank approximation, it must be checked that each element $\tilde{B}_{k\ell} \in [0,1]$. The procedure is summarised in Algorithm 5.2.

Algorithm 5.2: Simulation of an undirected stochastic blockmodel.

1 for $i = 1, \ldots, n$, sample $z_i \sim$ Multinoulli$(\boldsymbol{\theta})$,
2 simulate $\mathbf{B} = \{B_{k\ell}\} \in [0,1]^{K \times K}$, $B_{k\ell} = B_{\ell k}$, where, for $k \leq \ell$, $B_{k\ell} \sim$ Beta(a, b) for $a, b > 0$,
3 obtain a rank-d truncation of $\mathbf{B}$ using $\tilde{\mathbf{B}} = \hat{\boldsymbol{\Gamma}}\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{\Gamma}}^\top$, ensuring that $\tilde{B}_{k\ell} \in [0,1]\ \forall k, \ell$,
4 obtain the adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, $A_{ij} = A_{ji}$, where $A_{ij} \sim$ Bernoulli$(\tilde{B}_{z_i z_j})$.

The algorithm can be also extended to directed and bipartite graphs. In this section, each element $B_{k\ell}$ of $\mathbf{B}$ was generated from a Beta$(1.2, 1.2)$ distribution, which produces communities with a

moderate level of separation. For the given choice of K, the community allocations probabilities in the simulations were chosen to be $\boldsymbol{\theta} = (1/K, \ldots, 1/K)$, providing balanced clusters.

If a stochastic blockmodel is simulated using Algorithm 5.2, all the parameters are known, and the fit of model (5.7) can be evaluated using the true underlying cluster allocations $\boldsymbol{z}$.

5.6.1.1. Emprical analysis of spectral embeddings

Figure 5.4 illustrates results for a synthetic undirected stochastic blockmodel with $d = 2$ and $K = 5$. Figure 5.4a shows the scatterplot of the first two columns of the adjacency embedding $\mathbf{X}$, coloured using the true underlying communities. The plot shows well-separated clusters, which can be suitably modelled using a Gaussian mixture. Figure 5.4b shows the scatterplot of the next two dimensions. Clearly, the community mean locations are significantly different from zero in just the first two dimensions. This is further illustrated in Figure 5.4c, which plots the empirical within-cluster means for each dimension, obtained using the known clustering $\boldsymbol{z}$. Figure 5.4c gives empirical evidence that the form $[\boldsymbol{\mu}_{z_i}^\top, \mathbf{0}^\top]^\top$ for the mean of $\boldsymbol{x}_i$ in (5.7) suitably describes what is observed in SBMs. Similarly, Figure 5.4d shows the empirical within-cluster variances for each dimension, again obtained using the known values of $\boldsymbol{z}$ from the simulation. In Figure 5.4d, the difference between the within-cluster and overall variance is evident only in the first two dimensions, after which the quantities are of the same order of magnitude. The plot also shows that the within-cluster variances differ across communities, suggesting that it is appropriate to have cluster-specific values of $\sigma_{j,k}^2$ for $j > d$; this phenomenon can also be witnessed in Figure 5.4b, and Figure 5.2 in Section 5.4. Nevertheless, if the MCMC sampler were run on the simulated data in Figure 5.4, it could also be appropriate to use a second-level clustering with $H = 3$, since the variances of three of the five communities are approximately the same for dimensions larger than $d = 2$. Furthermore, for small m and fairly large n, it seems from Figure 5.4d that the vector $\boldsymbol{\sigma}_k^2$ could be approximated by a constant σ_k^2 as in Yang et al. (2020). However, for small values of n and large m, as in Figure 5.2, parameter vectors $\boldsymbol{\sigma}_k^2$ are clearly required. If a constant σ_k^2 is used, the inferential procedure is essentially identical, but (5.12) should be modified accordingly.

Figure 5.4e shows the histogram of the sample correlation coefficients $r_{ij}^{(k)},\ i, j = 1, \ldots, m,\ i < j$, for each community, obtained from the known $\boldsymbol{z}$. The cluster-specific correlation coefficients $r_{12}^{(k)}$ between $\mathbf{X}_1$ and $\mathbf{X}_2$ are represented by bullet points in the plot, suggesting dependence for at least one of the clusters, confirming the result of Rubin-Delanchy et al. (2017), and providing further evidence that cluster-specific full covariance matrices $\boldsymbol{\Sigma}_k$ should be used for the covariance of $\boldsymbol{x}_i$ in (5.7). On the other hand, the empirical within-cluster correlations for $\mathbf{X}_{d:}$ tend to be small and centred around 0, suggesting that the assumption of independence is appropriate in that part of the model in (5.7). Finally, Figure 5.4f plots the marginal log-likelihood as a function of d, using the known cluster configuration $\boldsymbol{z}$. The marginal likelihood strongly favours the true value $d = 2$, resulting in a posterior distribution essentially consisting of a point mass at the true value.

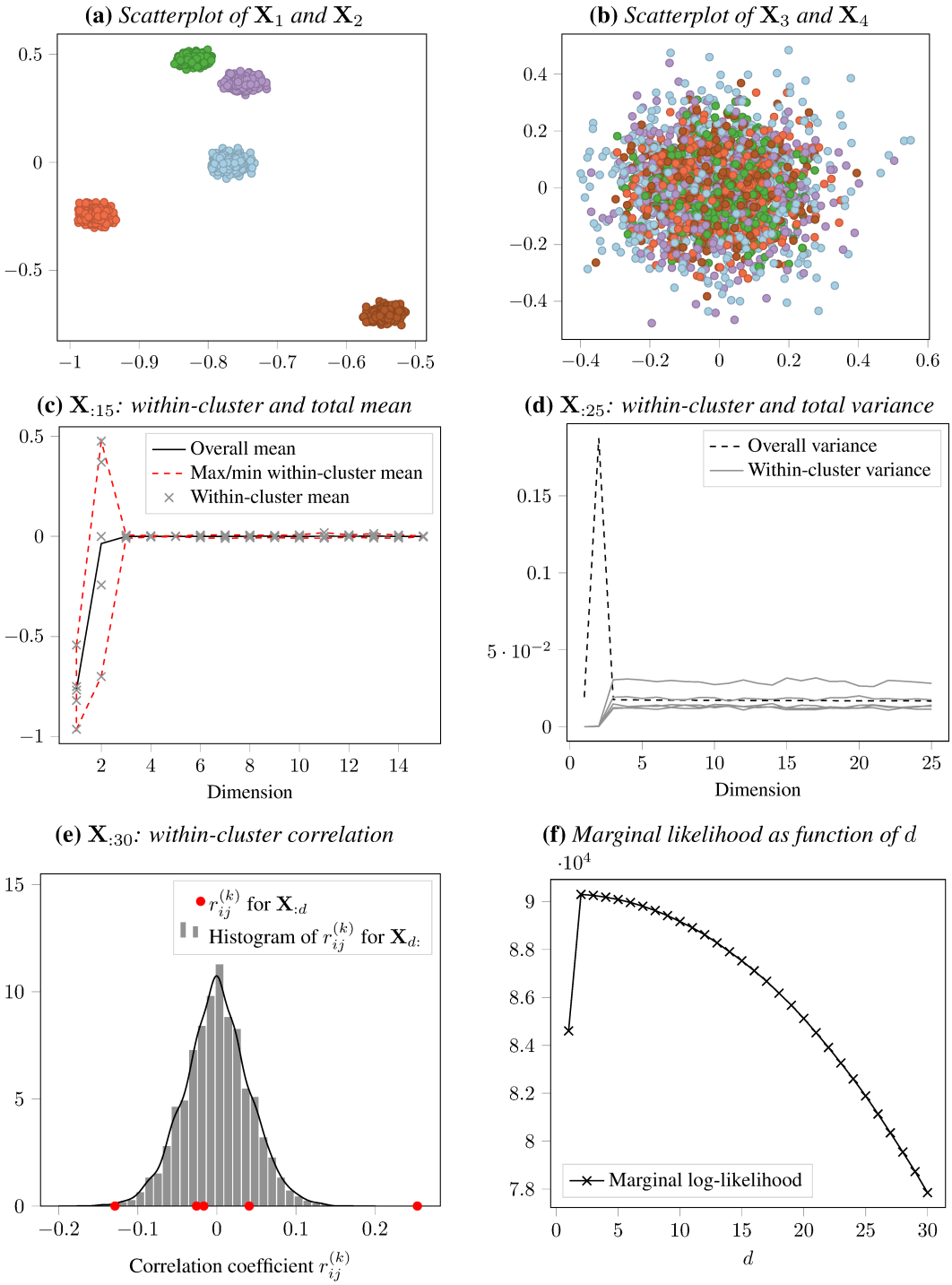


Figure 5.4.: *Adjacency embedding for an undirected graph with $n = 2{,}500$ nodes, $K = 5$, obtained from a symmetric $\mathbf{B} \in [0,1]^{K\times K}$ with $B_{k\ell} \sim$ Beta$(1.2, 1.2)$, $\boldsymbol{\theta} = (1/K, \ldots, 1/K)$ and $d = 2$. For (e) and (f), $m = 50$ is used.*

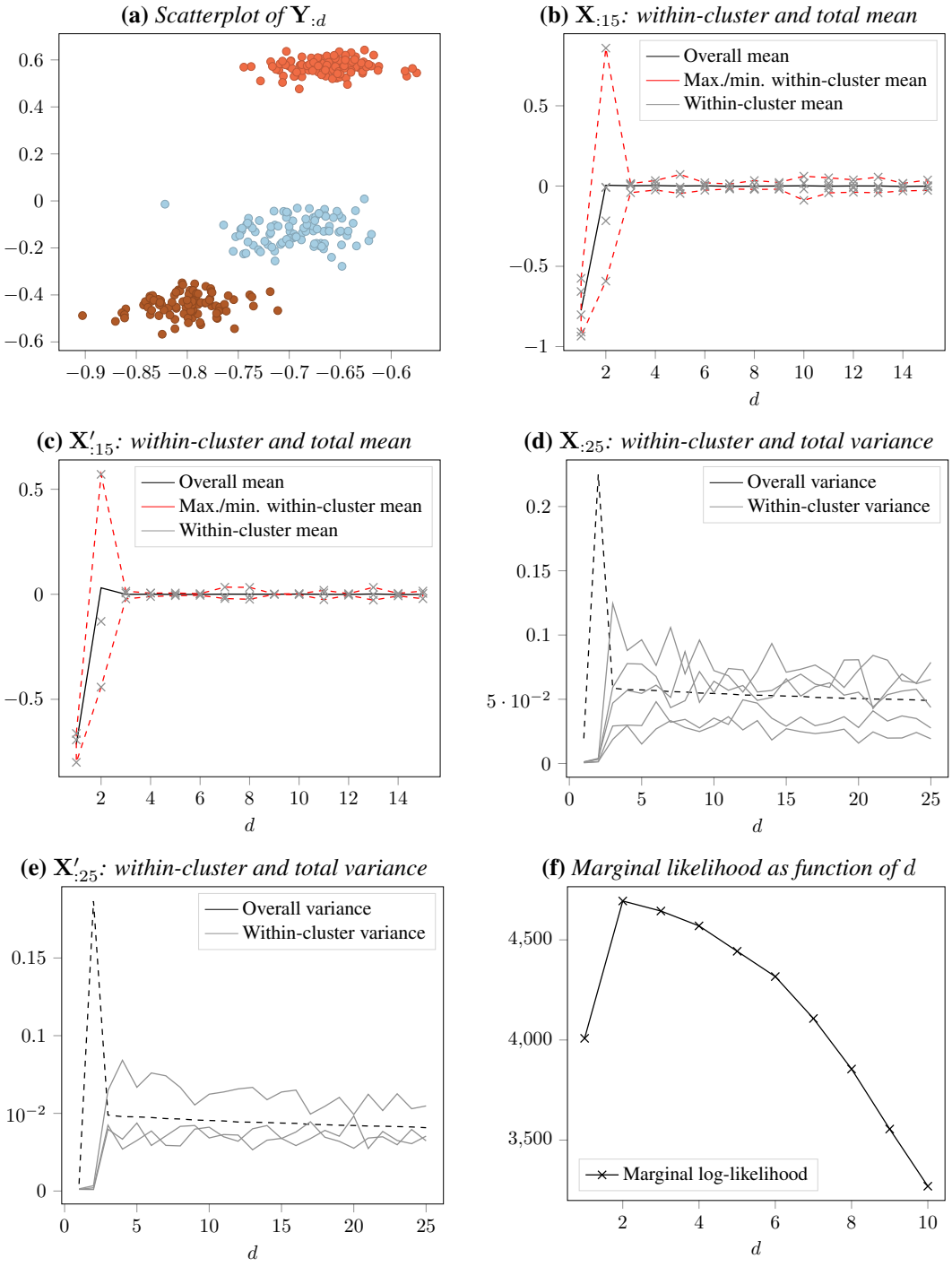


Figure 5.5.: *Simulated adjacency embedding for a bipartite* 250×300 *graph with* $K = 5$ *and* $K' = 3$*, obtained from* $\mathbf{B} \in [0,1]^{K \times K'}$ *with* $B_{k\ell} \sim$ *Beta*$(1.2, 1.2)$*,* $\boldsymbol{\theta} = (1/K, \ldots, 1/K)$*,* $\boldsymbol{\theta}' = (1/K', \ldots, 1/K')$*, and* $d = 2$*. For (f),* $m = 50$*.*

Figure 5.5 shows similar results for a simulated bipartite graph with separate community structures for nodes as sources and destinations, with $d = 2$, $K = 5$ and $K' = 3$. Again, the scatterplot for $\mathbf{X}_1'$ and $\mathbf{X}_2'$ in Figure 5.5a are well-separated and can be easily estimated using the Gaussian mixture model. Figure 5.5b and 5.5c show the empirical within-cluster means for each dimension, obtained using $\boldsymbol{z}$ and $\boldsymbol{z}'$. The zero-mean assumption for the columns with index larger than d seems to hold even for a relatively small number of nodes per community. Figure 5.5d and 5.5e show the empirical within-cluster variances for each dimension. Again, it is confirmed that the variances are different for each community, even on the last $m - d$ components. The within-cluster variance on the last $m - d$ also seem to be decreasing, showing that for small graphs the parameter vector $\boldsymbol{\sigma}_k^2$ could be preferable to a constant σ_k^2 as in Yang et al. (2020). In Figure 5.5f, the marginal likelihood strongly favours the true value $d = 2$, which again results in a point mass posterior centred at the true value. Similar considerations hold for the correlations, with results which are similar to the plots in Figure 5.4e for the undirected graph.

5.6.1.2. Parameter estimation

Table 5.1 shows the results from inference for model (5.7) when applied to SBMs simulated with different values of d and K, for all the possible combinations of the models considered in this chapter. For each (d, K) pair, an undirected SBM with $n = 1{,}000$ nodes is generated, and the MCMC sampler is run three times, each with $M = 10{,}000$ samples after a burn-in of $2{,}500$. The samplers are initialised using the guidelines discussed in the introduction to Section 7.3. The table reports the averaged values of d, $K_\varnothing$ and $H_\varnothing$ obtained from the MCMC chains. The posterior mean values of d and K which are obtained are extremely close to the true values, implying that the results support the proposed model.

From Table 5.1, the performance of adjacency and Laplacian spectral embedding for recovering d, K and $\boldsymbol{z}$ seems to be equivalent. Similar results for $\mathbf{A}$ and $\mathbf{L}$ are also obtained for the application on real data in Section 5.6.2. Note that, for fixed values of K, Priebe et al. (2019) demonstrate that, in practical applications, LSE tends to better capture the affine structure in stochastic blockmodels, whereas ASE better identifies the core-periphery structure. Also, Table 5.1 shows that the constrained and unconstrained models seem to give an equivalent performance. The difference between the constrained model (5.8) and unconstrained model (5.9) will be more evident in Sections 5.6.2 and 5.6.3, where the data might deviate from the stochastic blockmodel assumption. Overall, for synthetic data, the model seems robust and able to detect the correct d and K in a variety of different settings.

5.6.1.3. Second-level clustering

It is also possible to evaluate the effect of the second-order clustering for estimation of the community structure $\boldsymbol{z}$, and the parameters d and K. For the same simulated graph, the MCMC sampler

Table 5.1.: *Results of the inferential procedure for undirected SBMs simulated using different* (d, K) *pairs.*

(d, K)	Model	$m = 25$		
		$\bar{d}$	$\bar{K}_\varnothing$	$\bar{H}_\varnothing$
$(2, 2)$	constrained, ASE	2.00	2.00	1.99
	unconstrained, ASE	2.00	2.00	1.99
	constrained, LSE	2.01	2.03	1.99
	unconstrained, LSE	2.02	2.02	1.99
$(2, 5)$	constrained, ASE	2.00	5.05	1.77
	unconstrained, ASE	2.00	5.07	1.80
	constrained, LSE	2.05	5.10	3.11
	unconstrained, LSE	2.07	5.11	3.10
$(6, 7)$	constrained, ASE	6.00	7.04	2.10
	unconstrained, ASE	6.00	7.05	2.20
	constrained, LSE	6.00	7.10	2.47
	unconstrained, LSE	6.00	7.07	2.39

(d, K)	Model	$m = 25$		
		$\bar{d}$	$\bar{K}_\varnothing$	$\bar{H}_\varnothing$
$(9, 9)$	constrained, ASE	8.97	9.01	2.08
	unconstrained, ASE	9.00	9.01	1.98
	constrained, LSE	9.00	9.02	2.12
	unconstrained, LSE	9.00	9.04	2.11
$(9, 12)$	constrained, ASE	9.00	12.02	1.96
	unconstrained, ASE	9.00	12.01	1.90
	constrained, LSE	9.00	12.03	2.60
	unconstrained, LSE	9.00	12.02	2.53
$(10, 15)$	constrained, ASE	10.00	14.78	1.25
	unconstrained, ASE	10.00	14.11	1.27
	constrained, LSE	10.00	14.81	1.81
	unconstrained, LSE	10.00	15.01	1.87

has been run with and without the second order clustering, using the same settings as the simulation in Table 5.1. Note that the absence of second order clustering corresponds to the case $H = K$. The procedure is repeated for different values of m, for n fixed, to study the effect of second-order clustering when m is increased. The results of the simulations are summarised in Table 5.2 for two different simulated graphs. The table reports the *maximum a posteriori* (MAP) values of d and $K_\varnothing$, and the adjusted Rand index (ARI) (Hubert and Arabie, 1985) for the estimated clustering of the nodes, obtained setting $K_\varnothing$ to its MAP estimate. For simplicity, only the results for the unconstrained model using ASE are reported. For the case when H is unknown, the table reports the posterior mean of $H_\varnothing$.

In all the simulation settings, the model was able to recover the correct values of d, but when m is large, only the second-order clustering model allows K to be estimated correctly. When the second-order clustering is not used and m is very large relative to d and K, the MAP estimate $\hat{K}_\varnothing$ tends to be underestimated, and the inferred clustering structure is therefore also negatively affected. Also, the table shows that as $m \to n$, the estimated value of $H_\varnothing$ tends to decrease towards 1. Hence, to reduce the computational burden for large m, it would be possible to set $H = 1$, corresponding to the scenario $\boldsymbol{\sigma}_k^2 = \boldsymbol{\sigma}^2 \; \forall k$, studied in Raftery and Dean (2006) in the context of variable selection in GMMs. Table 5.2 also suggests that the model with second-order clustering is robust to the choice of m. This is one of the main advantages of the proposed model: the correct d can be recovered for any choice of m, provided $d \leq m$. Choosing an upper bound m is easier than choosing the correct d, especially because of the robustness of the model. On the other hand, choosing large values of m makes the MCMC sampler computationally more expensive. A suitable choice would be to set m based on common criteria to obtain d (for example, the profile likelihood criterion of Zhu and Ghodsi, 2006). For a given value $d^\star$ obtained using such criterion, it would be appropriate to choose $m > d^\star$, and then correct the initial estimate of d using the proposed model.

Table 5.2.: *Results for the MCMC sampler on simulated undirected SBMs for different values of m, with and without second order clustering.*

(d, K)	m	H random				$H = K$		
		$\hat{d}$	$\hat{K}_\varnothing$	$\bar{H}_\varnothing$	ARI	$\hat{d}$	$\hat{K}_\varnothing$	ARI
$(3, 5)$	15	3	5	1.669	1.000	3	5	1.000
	50	3	5	1.577	1.000	3	4	0.768
	150	3	5	1.467	1.000	3	4	0.768
	500	3	5	1.006	1.000	3	4	0.768
$(9, 12)$	15	9	12	1.979	1.000	9	12	1.000
	50	9	12	1.912	1.000	9	12	1.000
	150	9	12	1.875	1.000	9	11	0.942
	500	9	12	1.388	1.000	9	5	0.517

5.6.2. Undirected graphs: Santander bikes

The *Santander Cycle hire scheme* is a bike sharing system implemented in central London. Transport for London periodically releases data on the bike hires[1]. Considering this as a network, the nodes correspond to bike sharing stations, and an undirected edge between stations i and j is drawn if at least one ride between the two stations is completed within the time period considered. In this example, one week of data were considered, from 5 September until 11 September, 2018. The total number of stations used is $n = 783$, and the total number of undirected edges is $|E| = 96{,}060$, implying the adjacency matrix is fairly dense. Note that it is possible to collect and return the bike to the same docking station. Those edges, amounting to less than 1% of $|E|$, have been removed, since our modelling framework is specifically developed for hollow binary matrices.

The results of the Bayesian inferential procedure, using the unconstrained prior (5.9) for d, applied to the adjacency and Laplacian embeddings for the Santander bike network are presented in Figure 5.6. The initial value of K was set to 10, with $m = 25$, but similar estimates were obtained using different starting points for K and different values of m. It is interesting to note the different shapes of the posterior barplots of $K_\varnothing$ and $H_\varnothing$, Figures 5.6a and 5.6b, showing that the second-level clustering is crucial to obtain a more accurate model fit when the adjacency embedding is used. On the other hand, for the Laplacian embedding, the posteriors for $K_\varnothing$ and $H_\varnothing$ are extremely similar, suggesting that the second-level clustering is not required for $m = 25$. The MAP value $d = 11$ for the adjacency and Laplacian embeddings correspond to the elbow in the scree-plots (see Figure 5.6c for $\mathbf{A}$ and Figure 5.6d for $\mathbf{L}$).

Note that, especially in the case of the adjacency embedding, d and K have similar values, showing that the two graphs might be well described by a stochastic blockmodel. Similarly, the constrained model with $d \sim \text{Uniform}\{1, \dots, K_\varnothing\}$ (5.7) returns the same MAP estimates for d, but

[1] The data are publicly available at `https://cycling.data.tfl.gov.uk/`, powered by TfL Open Data (last accessed on 5th October, 2020).

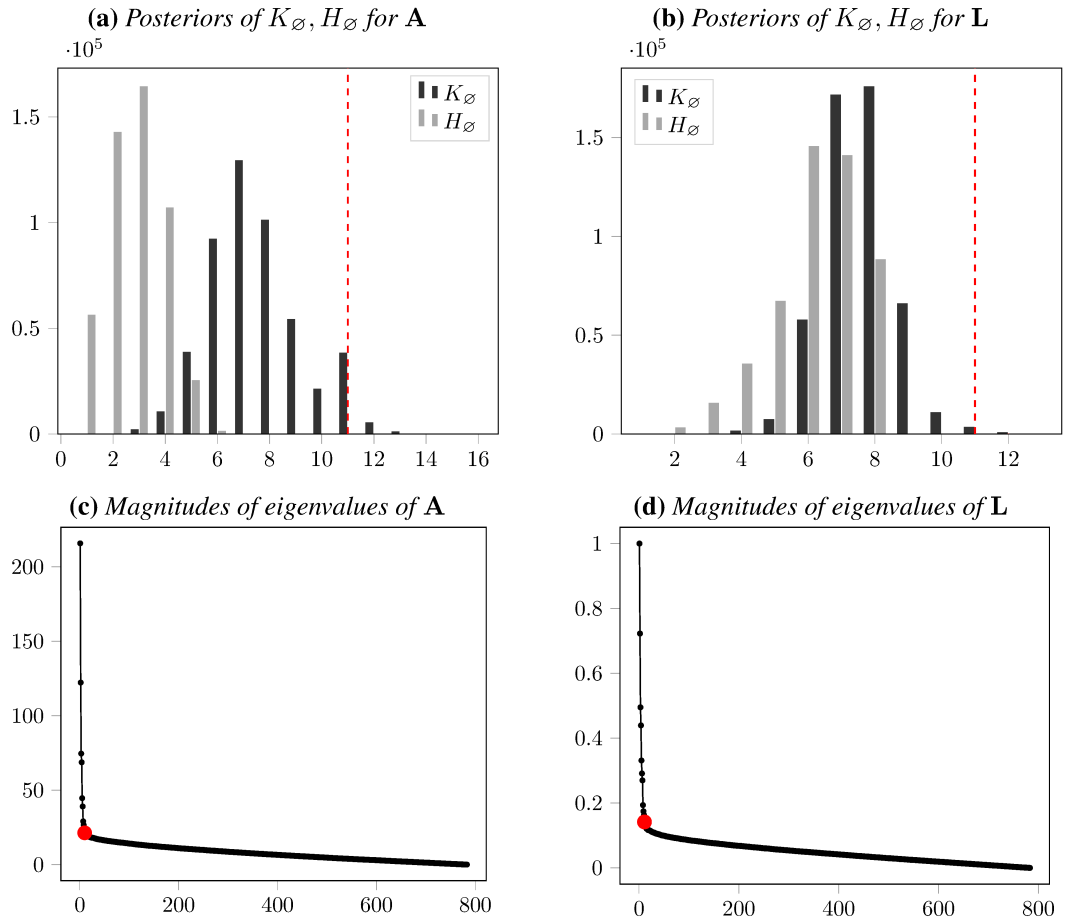


Figure 5.6.: *Posterior distributions and scree-plots for the Santander bike network data using adjacency and Laplacian embeddings, for the unconstrained model* (5.9)*. The dashed line in (a) and (b), and the large bullet point in (c) and (d) represent MAP estimates of* d*.*

the constraint $d \leq K_\varnothing$ results in a larger number of small clusters; the posterior of $K_\varnothing$ essentially reduces to the rescaled probability mass function obtained from the unrestricted model, constrained such that $d \leq K_\varnothing$, since the posterior for d is approximately a point mass.

The resulting estimated clustering for the unconstrained model (5.9) based on the adjacency embedding and $K = 11$ (the MAP for d), plotted in Figure 5.7, shows a clear structure: the largest communities have approximately the same extension, with a few exceptions. This is expected, since the bikes are free for the first 30 minutes and have limited speed, and are therefore used for small distance journeys. Two clusters are significantly smaller than the others, and correspond to touristic areas around Westminster, Covent Garden and Buckingham Palace. On the other hand, two clusters have a large geographical extension, and cover the East and West London areas. For the adjacency embedding, the MAP clustering obtained from the restricted model is almost identical. The PEAR method (Fritsch and Ickstadt, 2009) suggests $K = 7$ communities instead. Similarly, if

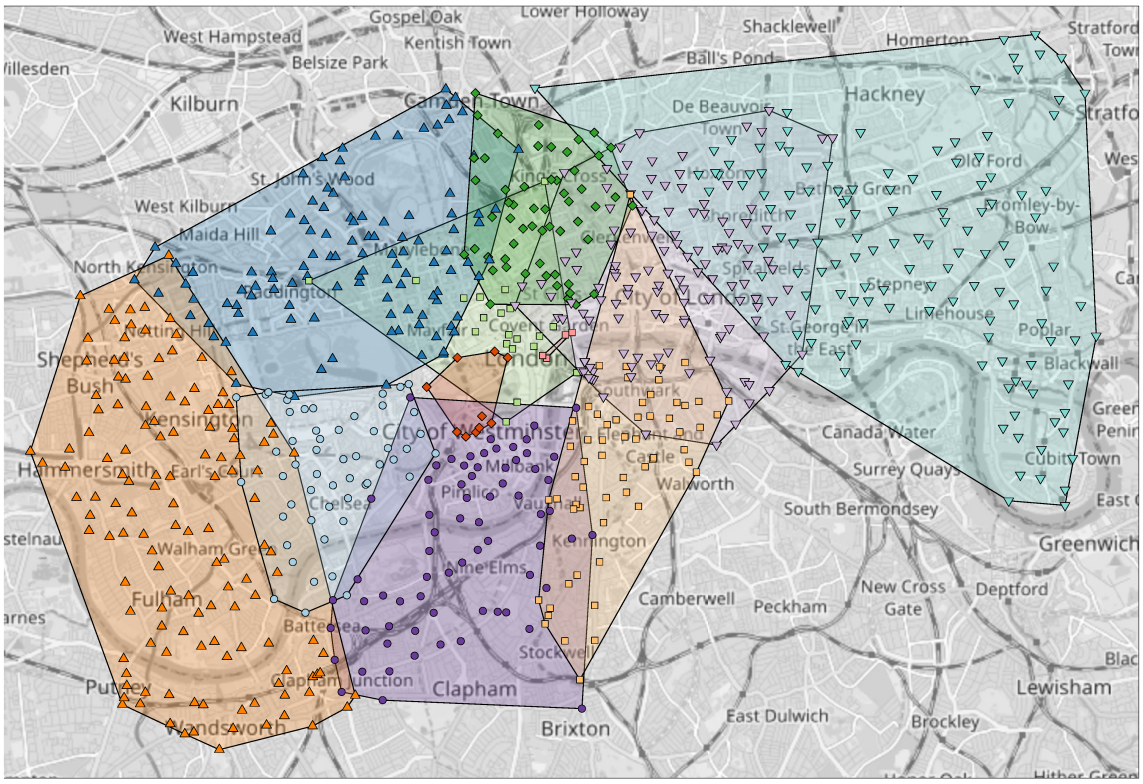


Figure 5.7.: *Santander bike sharing stations in London and maximum a posteriori estimates of the cluster allocations of the stations, obtained using hierarchical clustering with distance* $1-\hat{\pi}_{ij}$*,* $K = 11$*. Stations in the same convex hull share the same cluster.*

the Laplacian embedding is used, the MAP clustering structure suggested by PEAR has $K = 7$ communities for the unconstrained model (5.9) and $K = 12$ for the constrained model (5.7).

5.6.3. Directed graphs: Enron Email Dataset

Next, the algorithm is applied to a directed network: the Enron Email Dataset[2]. The Enron database is a corpus of emails sent by the employees of the Enron corporation. The version of the Enron data which has been analysed here is described in Priebe et al. (2005), and consists of $n = 184$ nodes and 3,010 directed edges. A directed edge $i \to j$ is drawn if the employee i sent an email to the employee j.

The results of analysing this network are presented in Figure 5.8. The initial value of K was set to 10, with $m = 25$, but again similar results were obtained using different starting points for K and different values of m. The plots in Figures 5.8b and 5.8c report the estimated posterior distributions of $K_\varnothing$ and $H_\varnothing$ for the constrained (5.7) and unconstrained (5.9) models. Interestingly, the MAP estimate for d coincides with the MAP estimate for K in the unconstrained model, which

[2]The data are publicly available at `https://www.cs.cmu.edu/~./enron/` (last accessed on 5th October, 2020).

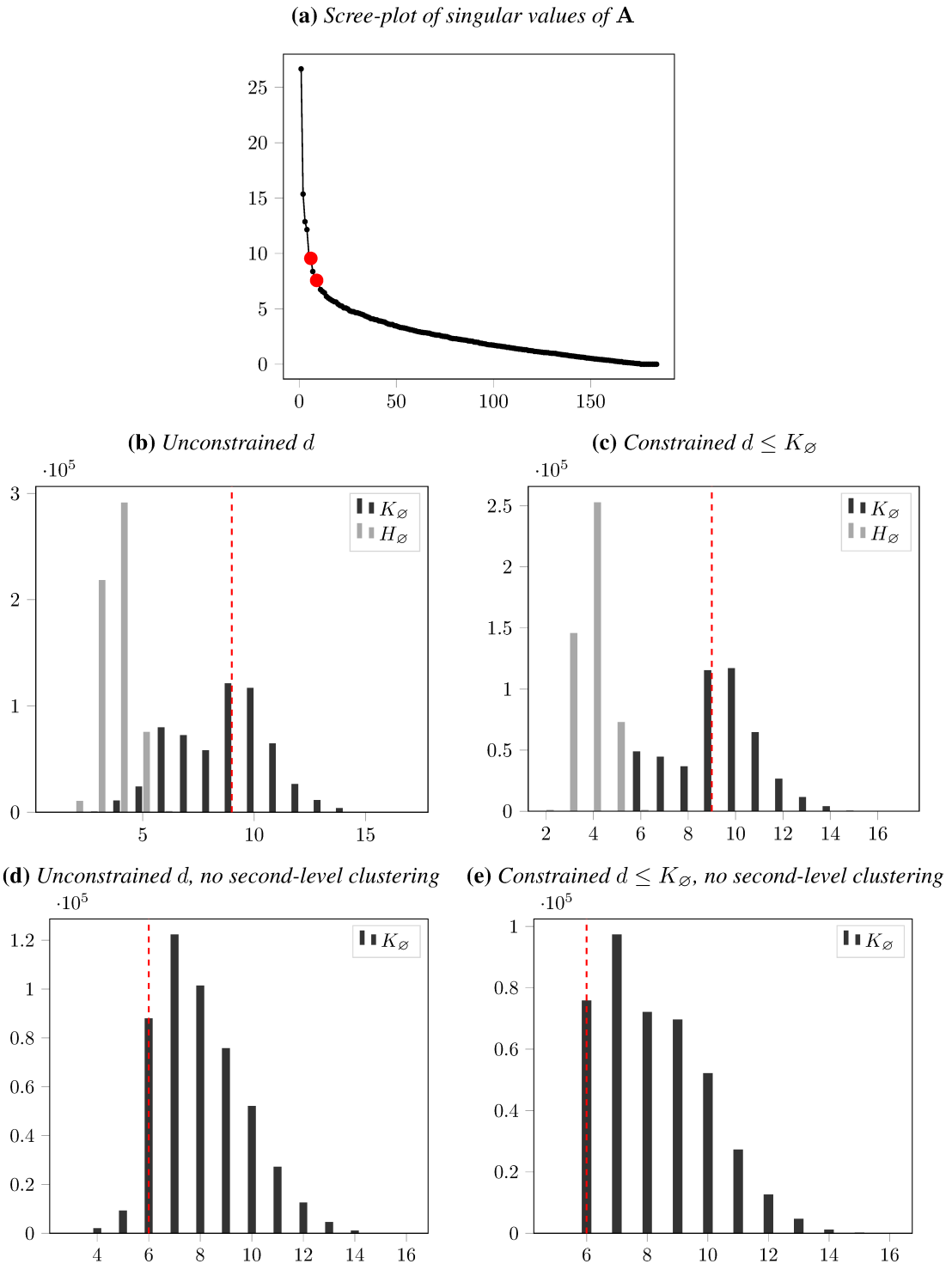


Figure 5.8.: *Scree-plot of singular values of* **A** *and posterior distributions of* $K_\varnothing$, $H_\varnothing$ *for the Enron data. The large bullet points in (a), and the dashed lines in (b)–(e) represent MAP estimates of* d.

is promising. For the constrained model, the MAP for K exceeds the MAP for d by 1, allowing for $\text{rank}(\mathbf{B}) < K$. Overall, d and K have similar values, showing that the graph might be well described by a directed stochastic blockmodel.

The posterior distributions for $K_\varnothing$ and $H_\varnothing$ in Figures 5.8b and 5.8c are fairly different, showing that the second-level clustering might be relevant for this model. Inference on the model without second-level clustering confirms this impression: the posteriors for $K_\varnothing$, presented in Figures 5.8d and 5.8e have a more symmetric shape, and the MAP latent dimension is $d = 6$. As before, the MAP for K is $d + 1 = 7$, providing some evidence for the possibility $\text{rank}(\mathbf{B}) < K$.

From Figure 5.8a, the selected MAP values $d = 6$ and $d = 9$ for the models with and without second-level clustering seem to be a tradeoff between the two most popular criteria for selection of the appropriate latent dimension: the *eigengap* heuristic suggests $d = 5$ if the second largest difference is considered, and the elbow in the scree-plot is approximately located around $d \approx 15$.

5.6.4. Bipartite graphs: Imperial College NetFlows

The proposed model has been shown to work well on a transportation network (Section 5.6.2) and on an email network (Section 5.6.3), but so far no indication has been given on its performance on the computer network data central to this thesis. This section demonstrates that the stochastic blockmodel might be an oversimplifying assumption in many networks related to cyber-security applications. This will motivate the need for refinements to the model which has been proposed in this chapter, in order to accommodate some of the characteristics of computer network graphs.

For example, for the Los Alamos National Laboratory computer network data (Turcotte, Kent, and Hash, 2018), the adjacency spectral embedding does not exhibit Gaussian clusters (see, for example, the embeddings plotted in Marchette, 2018), clearly a violation of the assumptions in (5.7). This is also observed in the Imperial College NetFlow data, as demonstrated in the example in this section.

The following bipartite graph has been constructed from the network flow data collected at Imperial College London (see Section 2.2): the edges consist of all the HTTP (port 80) and HTTPS (port 443) connections observed from 439 machines hosted in computer laboratories within the Departments of Chemistry, Civil & Environmental Engineering, Mathematics, and the School of Medicine. The observation period ranges between 1st January, 2020, and 31st January, 2020. In total, $717{,}912$ unique connections to $60{,}635$ internet servers were observed. The periodic and automated activity has been filtered by considering only edges such that the proportion of arrival times between 7am and 12am is larger than 0.95, corresponding to the opening hours of most Imperial College buildings. In the adjacency matrix, $A_{ij} = 1$ if the host i connected to the internet server j at least once during the observation period, and $A_{ij} = 0$ otherwise. The DASE embedding (Definition 4) was calculated from the adjacency matrix. Since the nodes are labelled by department, it could be assumed that the true underlying number of clusters is $K = 4$.

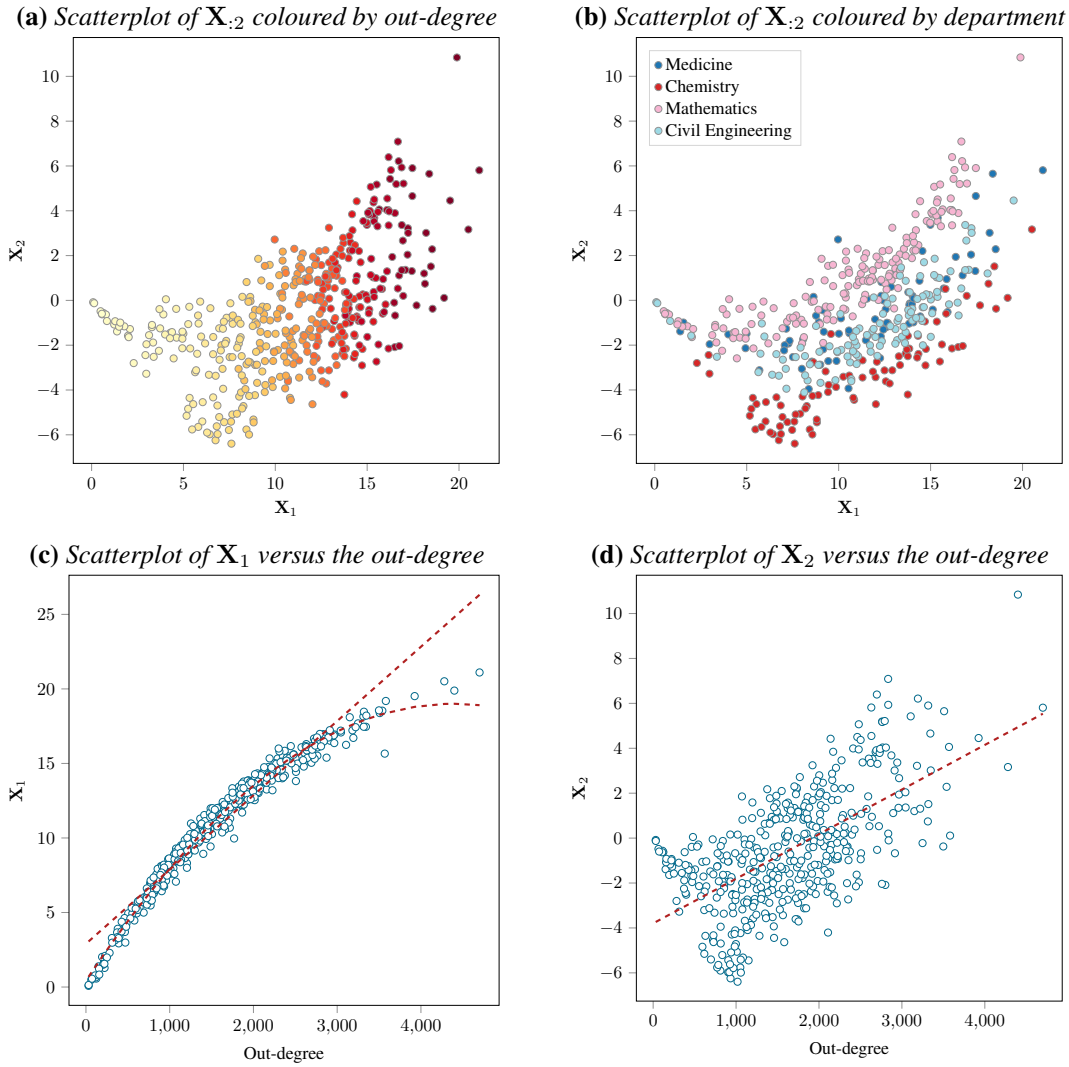


Figure 5.9.: *Scatterplots of the top-2 dimensions of the source embeddings of the bipartite graph, coloured according to (a) out-degree percentile and (b) department. Plots (c) and (d) show the scatterplots of $\mathbf{X}_{:2}$ versus the out-degree, with linear and quadratic regression lines.*

Figure 5.9 shows the scatterplot of the first two dimensions of the source embeddings $\mathbf{X}$, obtained from the singular value decomposition of the adjacency matrix $\mathbf{A}$ of the bipartite graph. The points in Figure 5.9a are coloured according to the percentile corresponding to the out-degree of each node. Similarly, the colours in Figure 5.9b correspond to the department, which is a known label of each source node.

Figure 5.9a clearly shows that the top dimensions of the embedding tend to separate the nodes by degree. This is confirmed by the two plots in Figure 5.9c and 5.9d, showing the scatterplots of $\mathbf{X}_1$ and $\mathbf{X}_2$ versus the out-degree of each node. It is clear that strong relationships, respectively quadratic and linear, exist between the first two dimensions of the embedding and the out-degree of

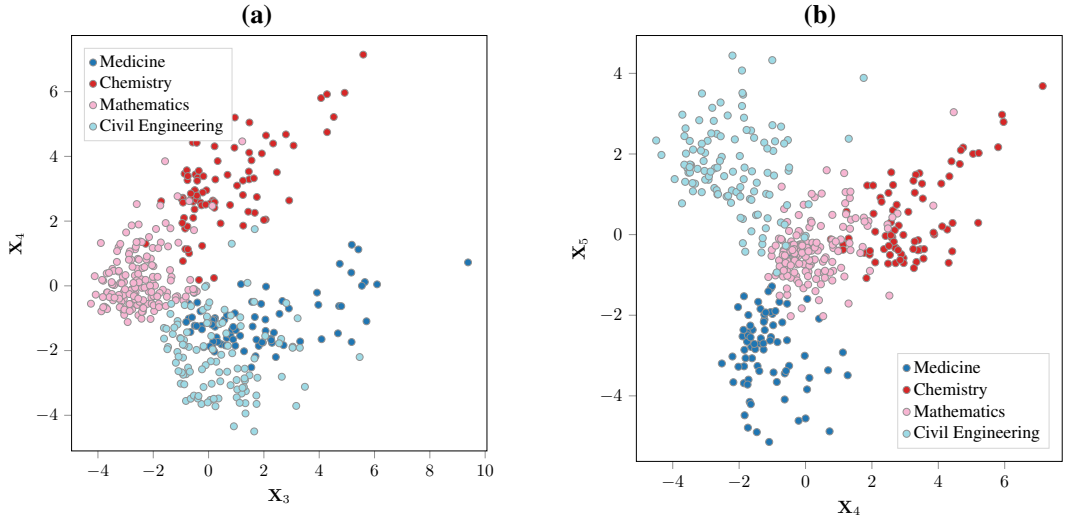


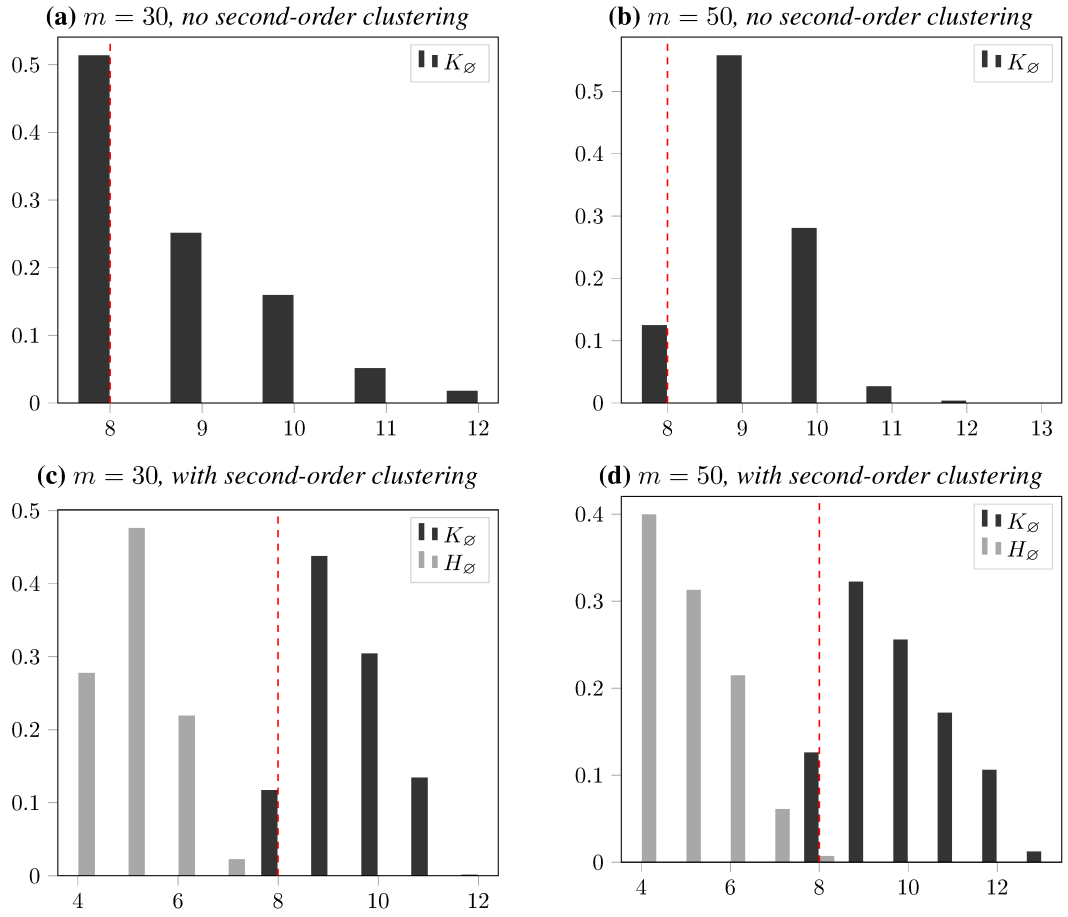
Figure 5.10.: *Scatterplots of the dimensions 3, 4 and 5 of the source embeddings of the bipartite graph, coloured according to department.*

the nodes. This might cause undesirable results when clustering enterprise computer networks.

Comparing Figures 5.9a and Figure 5.9b also shows that nodes have similar within-department out-degree distributions. This is reasonable, since all the machines are used from students in different departments: the hosts present similar within-department activity levels, but different between-department activity types. This implies that machines in different departments tend to connect to different computer servers, because of the different nature of the department specific activities carried out by the students in computer laboratories. Therefore, clustering using features correlated with out-degree might not be representative of the underlying block structure of the network.

In Figure 5.9b, some separation between the $K = 4$ departments seems to be present in $\mathbf{X}_2$, but most of the variability on that dimension is due to the out-degree. The division by department is evident in subsequent dimensions, as Figure 5.10 shows. In particular, $\mathbf{X}_3$, $\mathbf{X}_4$ and $\mathbf{X}_5$ show more clearly the separation between the true departmental clusters. In Figure 5.10, Gaussian clusters seem to be present also on dimensions exceeding K, suggesting that the stochastic blockmodel assumption with $K = 4$ might not be appropriate for this graph.

The Bayesian model for simultaneous selection of d and K has been fitted on the DASE source embedding $\mathbf{X}$, for $m = 30$ and $m = 50$, independently of the embedding $\mathbf{Y}$ for the destination nodes. The results are presented in Figure 5.11, and are based on chains of length $50{,}000$, obtained after a burn-in period of length $5{,}000$. From the plots, the choice of d seems consistent for different values of m. Similarly, the number of clusters K fluctuates around similar values across different model configurations. For this graph, the MAP estimate for the dimension is $d = 8$. For the number of clusters, $K = 9$ is MAP for three model configurations.



(e) *Scatterplot of* $\mathbf{X}_1$ *and* $\mathbf{X}_2$*, marked by department, coloured by estimated clusters* ($K = 9$, $m = 50$)

(f) *Scatterplot of* $\mathbf{X}_4$ *and* $\mathbf{X}_5$*, marked by department, coloured by estimated clusters* ($K = 9$, $m = 50$)

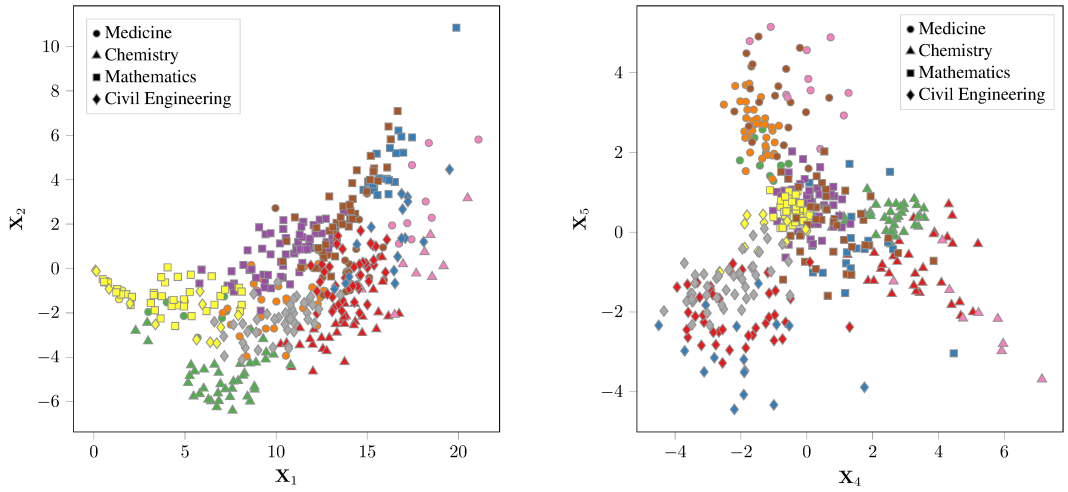


Figure 5.11.: *Posterior distributions of* $K_\varnothing$ *and* $H_\varnothing$ *for the constrained model, MAP estimates of* d*, and estimated clustering for* $m = 50$ *with* $K = 9$ *(MAP), without second-order clustering.*

Figures 5.11e and 5.11f show the estimated clustering structure for $K = 9$, based on the model with $m = 50$. As expected, the clusters tend to separate the nodes according to two characteristics: degree and department. Roughly, it seems that the degree distribution is partitioned in four classes: small, moderate, high and very high degree. This is particularly evident in Figure 5.11e, where the four groups are approximately separated by the thresholds 8, 12 and 17 on $\mathbf{X}_1$. On the other hand, the scatterplot for $\mathbf{X}_4$ and $\mathbf{X}_5$ in Figure 5.11f, when compared with Figure 5.10b, shows that the clustering procedure approximately separates the nodes according to their department.

Overall, this example clearly shows that the stochastic blockmodel might not be an appropriate choice for this graph, since it does not account for heterogeneous degree distributions within each true cluster. The problem will be addressed in the next chapter, where a spectral clustering algorithm on a transformation of the embedding to spherical coordinates is proposed. It must be remarked that, despite the suboptimal performance of the proposed model on computer network data, the modelling framework discussed in this chapter has already been shown to be extremely useful in practice when applied to networks having a stochastic blockmodel structure.

5.7. Conclusion and discussion

In this chapter, a novel Bayesian model has been proposed for automatic and simultaneous estimation of the number of communities and latent dimension of stochastic blockmodels, interpreted as special cases of generalised random dot product graphs. The Bayesian framework allows the number of communities K and latent dimension d to be treated as random variables, with associated posterior distributions. The postulated model is based on asymptotic results in the theory of network embeddings and random dot product graphs, and has been validated on synthetic datasets, showing good performance at recovering the latent parameters and communities. The model has been extended to directed and bipartite graphs, using SVD embeddings and allowing for co-clustering.

Overall, the main advantage of the proposed methodology is to allow for an arbitrarily large value of m, the number of columns (dimension) of the embedding at the first stage of the analysis, and then to treat d and K separately, allowing for the case $d = \text{rank}(\mathbf{B}) < K$, which is often overlooked in the literature. Problems arising from overspecification of m are tackled using a second-level clustering procedure. Also, the model provides an automated procedure and criterion to select the dimension of the embedding and an appropriate number of communities. If d is not constrained to be less than or equal to K, the model also provides empirical evidence of the goodness-of-fit of a stochastic blockmodel for the observed data. Results on real world network data sets show encouraging results in recovering the correct d, when compared to commonly used heuristic methods, and the community structure.

5.7.1. Limitations and future work

The methodology proposed in this chapter provides a valuable contribution to the literature on stochastic blockmodels, allowing for simultaneous selection of the embedding dimension and the number of communities in spectral clustering, but advancements are still required to extend the algorithm to graphs with a large number of nodes. In particular, the MCMC inferential procedure described in Section 5.3 is computationally demanding, which makes the algorithm difficult to scale to large networks.

Furthermore, as demonstrated in Section 5.6.4, the stochastic blockmodel is an overly simplifying assumption in many real world networks, including some cyber-security applications. Section 5.6.4 gives an example on the Imperial College NetFlow data, where the model is unable to recover known labels related to the nodes. This issue motivates the model explored in the next chapter, which represents an extension of the methodology from this chapter to graphs presenting heterogeneous within-community degree distributions.

6

Spectral clustering under the degree-corrected stochastic blockmodel

As demonstrated in Section 5.6.4 with an example of a computer network, the stochastic blockmodel has a significant practical weakness when applied to real world graphs: it does not account for heterogeneity in the within-community degree-distribution. Degree-corrected stochastic blockmodels (DCSBM, Karrer and Newman, 2011) overcome this problem by introducing node-specific parameters $\rho_i \in [0, 1]$, and modifying the edge probability (5.2) as follows:

$$A_{ij} \sim \text{Bernoulli}(\rho_i \rho_j B_{z_i z_j}),\ i < j,\ A_{ij} = A_{ji}. \tag{6.1}$$

Similarly to the SBM, the DCSBM can also be expressed as a special case of RDPG, setting latent positions $\boldsymbol{x}_i = \rho_i \boldsymbol{\mu}_{z_i}$ in (5.1), for community-specific latent vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$. A model selection framework for choosing the most appropriate model between SBM and DCSBM is proposed in Yan et al. (2014), and a Bayesian method to quantify the amount of degree-correction is proposed in Herlau, Schmidt, and Mørup (2014).

This chapter mainly concerns with community detection in DCSBMs under a random dot product graph interpretation, adapting the model presented in Chapter 5. In particular, this chapter describes a novel transformation of the adjacency spectral embedding under the degree-corrected stochastic blockmodel, which is then used within a joint estimation framework for the community structure, the number of communities K and the latent dimension d of the latent positions.

Community detection in blockmodels, including the degree-corrected stochastic blockmodel, is an active field of research, as described in Chapter 5. Zhao, Levina, and Zhu (2012) present a general theory for assessing consistency under the degree-corrected stochastic blockmodel. Amini et al. (2013) use a pseudo-likelihood approach, providing consistency results for the estimators. Peng

and Carvalho (2016) frame the DCSBM in a Bayesian setting, using a logistic regression formulation with node correction terms. Chen, Li, and Xu (2018) propose a modularity maximisation approach. Gao et al. (2018) obtain a minimax risk result for community detection in DCBMs and propose a two-step clustering algorithm based on k-medians.

Similarly to the case of stochastic blockmodels, spectral clustering methods (von Luxburg, 2007) provide arguably one of the most popular approaches for the community detection task under the DCSBM (Chaudhuri, Chung, and Tsiatas, 2012; Qin and Rohe, 2013; Lei and Rinaldo, 2015; Gulikers, Lelarge, and Massoulié, 2017). A common technique for spectral clustering under the degree-corrected stochastic blockmodels has been proposed in Qin and Rohe (2013), and uses k-means on the normalised rows of the embedding based on the spectral decomposition of a regularised Laplacian matrix (Chaudhuri, Chung, and Tsiatas, 2012). The row-normalisation of the embedding is a well-established approach for spectral clustering (for example, Ng, Jordan, and Weiss, 2001), but the rows live in a $d-1$ dimensional manifold, so it does not seem fully appropriate to fit a model for d-dimensional clusters to such an embedding. Alternative methods include the SCORE algorithm of Jin (2015), which proposes to use k-means on an embedding scaled by the leading eigenvector, showing that the effect of degree heterogeneity can be largely removed by applying this scaling-invariant mapping to each row.

Most estimation methods commonly require the number of communities K to be known. Different methodologies have been proposed in the literature for selecting of the number of communities in stochastic blockmodels and their degree-corrected version (see the introduction to Chapter 5 and references therein). Furthermore, spectral clustering methods require the specification of the embedding dimension, and clustering is usually carried out *after* selecting an estimate of d. As discussed in the previous chapter, this sequential approach is suboptimal, since the estimated clustering configuration and the estimated number of communities K would ideally be estimated jointly with d.

This chapter extends the work in Chapter 5, providing two main contributions. First, a novel estimation method, suited to degree-corrected stochastic blockmodels, is proposed, based on a transformation of the spectral embedding for known d and K. The d-dimensional spectral embedding is reduced to a set of $d-1$ *directions*, or *angles*, changing the coordinates from a Cartesian coordinate system to a spherical system. Second, the proposed clustering estimation method is incorporated within a simultaneous model selection framework for d and K akin to (5.7), providing a joint estimation framework for the two parameters and the community allocations.

The chapter is organised as follows: Section 6.1 discusses the properties of spectral embedding in degree-corrected stochastic blockmodels. Next, Section 6.2 proposes a novel modelling framework for spectral clustering under the DCSBM, which is validated via simulations in Section 6.3. A method for selecting d and K is discussed in Section 6.4, and finally results and applications are discussed in Section 6.5.

6.1. Spectral embedding of degree-corrected stochastic blockmodels

The two embedding methods defined in the previous chapter, ASE and LSE (Definitions 2 and 3), are also commonly used for spectral clustering on degree-corrected stochastic blockmodels. In the context of DCSBMs, regularised versions of the LSE are also studied, for example $(\mathbf{D} + \tau\mathbf{I}_n)^{-1/2}\mathbf{A}(\mathbf{D} + \tau\mathbf{I}_n)^{-1/2}$, $\tau \geq 0$, proposed in Chaudhuri, Chung, and Tsiatas (2012). In this chapter, the main focus will be on the adjacency spectral embedding, denoted $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times m}$, where $\boldsymbol{x}_i \in \mathbb{R}^m$, for some m chosen such that $d \leq m \leq n$. As discussed in the introduction to the present chapter, the *row-normalised embedding* is also commonly used (Ng, Jordan, and Weiss, 2001). The row-normalised embedding will be denoted as $\tilde{\mathbf{X}} = [\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n]^\top$, where $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i/\|\boldsymbol{x}_i\|$. The notation $\tilde{\mathbf{X}}_{:d}$ denotes the first d columns of $\tilde{\mathbf{X}}$, whereas $\tilde{\mathbf{X}}_{d:}$ represents the $m - d$ remaining columns. Importantly, the parameter m will again be assumed to be fixed.

For d fixed and known, the RDPG Gaussian central limit theorems (Athreya et al., 2016; Rubin-Delanchy et al., 2017; Tang and Priebe, 2018) described in Section 5.1 clearly hold also for the DCSBM, setting $\boldsymbol{x}_i = \rho_i\boldsymbol{\mu}_{z_i}$, as discussed in the introduction to this chapter. In particular, the central limit theorem (5.5) predicts that the degree distortion causes each community to be represented as a *ray* through the origin in the adjacency spectral embedding:

$$\mathbf{Q}\hat{\boldsymbol{x}}_i \approx \mathbb{N}_d\left\{\rho_i\boldsymbol{\mu}_{z_i}, \mathcal{S}(\rho_i\boldsymbol{\mu}_{z_i})\right\}, \tag{6.2}$$

for some orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$ which is jointly applied on all the rows of $\hat{\mathbf{X}}$. The asymptotic approximation in (6.2) can be contrasted with (5.6), which provides instead strong justification for estimation of stochastic blockmodels via Gaussian mixture modelling on the ASE.

An example of a simulated embedding for a stochastic blockmodel is given in Figure 6.1a. In the simulation, the SBM has $n = 1{,}000$ nodes, $K = 4$, $d = 4$, and the entries of the block connectivity matrix $\mathbf{B}$ are sampled from a Uniform$(0, 1)$ distribution; the nodes are deterministically allocated to equal-sized communities. Clearly, Figure 6.1a intuitively shows that Gaussian mixture modelling on the rows of the ASE is clearly a suitable procedure, and it is easy to distinguish and separate the $K = 4$ communities. On the other hand, Figure 6.1b shows the embeddings for a simulated degree-corrected stochastic blockmodel obtained using the same parameters, community allocations and block connectivity matrix as Figure 6.1a, adding degree corrections ρ_i sampled from a Beta$(2, 1)$ distribution. The $K = 4$ communities are visually still easy separate, but the assumption of normality clearly does not hold, and therefore Gaussian mixture models or k-means are not suitable methods.

Figure 6.1b intuitively motivates the novel modelling choice in this chapter: modelling the *spherical coordinates*, or *angles* in the embedding arising from DCSBMs, rather than the ASE itself. This is achieved via a transformation of the embedding, which is described in the following section.

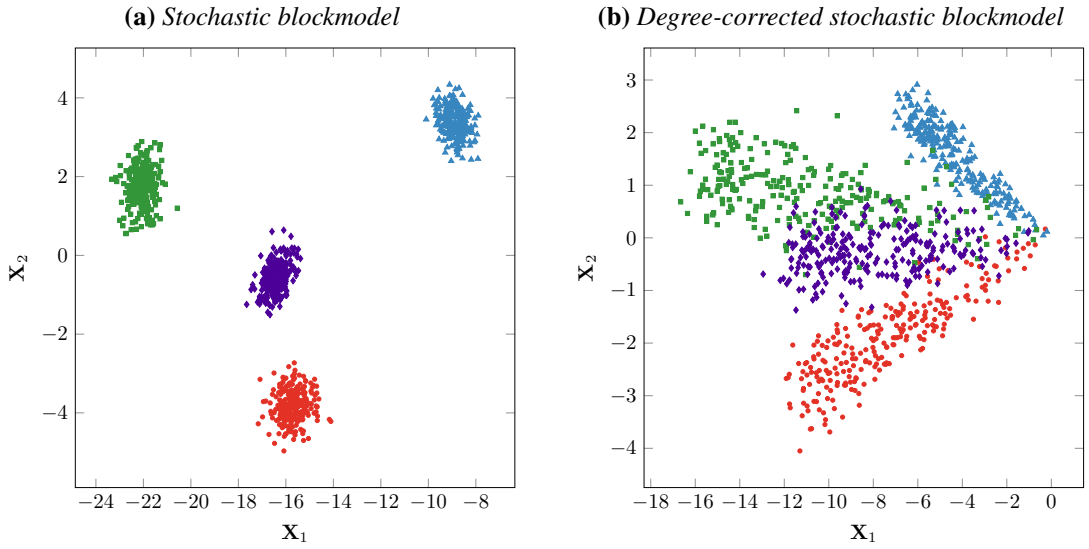


Figure 6.1.: *Scatterplot of the top two dimensions of the ASE for a simulated stochastic blockmodel with* $n = 1{,}000$ *nodes,* $K = 4$ *and* $d = 4$*, with an equal number of nodes allocated to each group. The same block connectivity matrix* $\mathbf{B}$*, sampled from a Uniform*$(0, 1)$ *distribution, has been used for simulating the adjacency matrices. The DCSBM is corrected by parameters* ρ_i *sampled from a Beta*$(2, 1)$ *distribution. The scatterplots are labelled by community memberships.*

6.2. Modelling a transformation of DCSBM embeddings

Consider a m-dimensional vector $\boldsymbol{x} \in \mathbb{R}^m$. The m Cartesian coordinates $\boldsymbol{x} = (x_1, \ldots, x_m)$ can be converted in $m - 1$ spherical coordinates $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{m-1})$ on the unit m-sphere using a mapping $f_m : \mathbb{R}^m \to [0, 2\pi)^{m-1}$ such that $f_m : \boldsymbol{x} \mapsto \boldsymbol{\theta}$, where:

$$\theta_1 = \begin{cases} \arccos(x_2/\|\boldsymbol{x}_{:2}\|) & x_1 \geq 0, \\ 2\pi - \arccos(x_2/\|\boldsymbol{x}_{:2}\|) & x_1 < 0, \end{cases}$$
$$\theta_j = 2\arccos(x_{j+1}/\|\boldsymbol{x}_{:j+1}\|), \; j = 2, \ldots, m-1. \tag{6.3}$$

For simplicity, $\theta_2, \ldots, \theta_{m-1}$ are rescaled to $[0, 2\pi)$, using a slight modification of the classical transformation from Cartesian to spherical coordinates (see, for example, Blumenson, 1960), which otherwise constrains the last $m - 2$ angles to $[0, \pi)$.

Consider an $(m + 1)$-dimensional adjacency embedding $\mathbf{X} \in \mathbb{R}^{n\times(m+1)}$ and define its transformation $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n]^\top \in [0, 2\pi)^{n\times m}$, where the transformation (6.3) is applied row-wise, such that $\boldsymbol{\theta}_i = f_{m+1}(\boldsymbol{x}_i), \; i = 1, \ldots, n$. The symbols $\boldsymbol{\Theta}_{:d}$ and $\boldsymbol{\theta}_{i,:d}$ will denote respectively the first d columns of the matrix and d elements of the vector, and $\boldsymbol{\Theta}_{d:}$ and $\boldsymbol{\theta}_{i,d:}$ will represent the remaining $m - d$ components.

In this chapter, a model is proposed for the transformed embeddings $\boldsymbol{\Theta}$. Suppose a latent

space dimension d, K communities, and latent community assignments $\boldsymbol{z} = (z_1, \ldots, z_n)$. The transformed coordinates $\boldsymbol{\Theta}$ are assumed to be generated independently from community-specific m-dimensional multivariate normal distributions:

$$\boldsymbol{\theta}_i | d, z_i, \boldsymbol{\vartheta}_{z_i}, \boldsymbol{\Sigma}_{z_i}, \boldsymbol{\sigma}^2_{z_i} \sim \mathbb{N}_m \left(\begin{bmatrix} \boldsymbol{\vartheta}_{z_i} \\ \pi \mathbf{1}_{m-d} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{z_i} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}^2_{z_i} \mathbf{I}_{m-d} \end{bmatrix} \right), \tag{6.4}$$

where $\boldsymbol{\vartheta}_{z_i} \in [0, 2\pi)^d$ represents a community-specific mean angle, $\mathbf{1}_m$ is a m-dimensional vector of ones, $\boldsymbol{\Sigma}_{z_i}$ is a $d \times d$ full covariance matrix, and $\boldsymbol{\sigma}^2_k = (\sigma^2_{k,d+1}, \ldots, \sigma^2_{k,m})$ is a vector of positive variances. The model in (6.4) could be also completed in a Bayesian framework using the same conjugate prior distributions from (5.8). The likelihood associated with the transformed embedding $\boldsymbol{\Theta} \in \mathbb{R}^{n \times m}$ obtained from a degree-corrected stochastic blockmodel can be expressed as:

$$L(\boldsymbol{\Theta}) = \prod_{i=1}^{n} \left\{ \phi(\boldsymbol{\theta}_{i,:d}; \boldsymbol{\vartheta}_{z_i}, \boldsymbol{\Sigma}_{z_i}) \prod_{j=d+1}^{m} \phi(\theta_{i,j}; \pi, \sigma^2_{z_i,j}) \right\}. \tag{6.5}$$

A discussion on the appropriateness of the normal distribution to model data on $[0, 2\pi)^m$ is warranted. Normally, a wrapped normal distribution would be preferred for such data, as adopted in Chapter 4. In the context of DCSBMs, it is known that $\boldsymbol{\Theta}$ arises from a transformation of the embedding $\mathbf{X}$, and the form of the transformation can be used to inform the modelling decisions. The arccosine function is monotonically decreasing, and communities will tend to have similar values in $x_{i,j}/\|\boldsymbol{x}_{i,:j}\|$ for $j = 1, \ldots, m-1$, see (6.3). If two points in $\boldsymbol{\Theta}$ reach the extremes 0 and 2π, then they are unlikely to belong to the same community. Therefore, the idea of building clusters using a wrapped distribution does not apply to this context. The only case that could cause concern is $\theta_{i,1}$, where the transformation (6.3) is not monotonically increasing or decreasing, but has a discontinuity at 0. On the other hand, $\mathbf{X}_1$, the first column of the adjacency spectral embedding, corresponds to the scaled leading eigenvector of the adjacency matrix, and therefore its elements are *all* non-negative by the Perron-Frobenius theorem for non-negative matrices (see, for example, Meyer, 2000). The theorem essentially makes the transformation $\boldsymbol{x}_{:2} \mapsto \theta_1$ in (6.3) monotonic, since one of the two conditions in the equation for θ_1 is satisfied by all the values in $\mathbf{X}_1$.

The rationale behind the model assumptions in (6.4) is to utilise the method of normalisation to the unit circle (Qin and Rohe, 2013) but assume normality on the spherical coordinates, not on their Cartesian counterparts. The transformed initial components $\boldsymbol{\theta}_{i,:d}$ are assumed to have unconstrained mean vector $\boldsymbol{\vartheta}_k \in [0, 2\pi)^d$. The $d \times d$ covariance matrix $\boldsymbol{\Sigma}_k$ for the spherical coordinates is assumed to be positive definite.

Further remarks about the assumption of Gaussian mixture modelling are in order: in spectral clustering under the stochastic blockmodel, the Gaussian mixture model on the informative part of the embedding matches the theoretical asymptotic distribution of the spectral embedding estimators

(Athreya et al., 2016; Rubin-Delanchy et al., 2017; Tang and Priebe, 2018), and it is preferable to k-means clustering, which is commonly used in the literature (Rohe, Chatterjee, and Yu, 2011). Similarly, in the context of degree-corrected stochastic blockmodels, the k-means algorithm on the normalised rows $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i/\|\boldsymbol{x}_i\|$ of $\mathbf{X}_{:d}$ is proposed in Qin and Rohe (2013) for known d and K, but such an algorithm might not closely capture some of the covariance structure observed in the embeddings, and the constraint of unit norm might cause distortions on some of the dimensions. This will be further explored via simulations in Section 6.3.

In contrast to the structured model for $\boldsymbol{\Theta}_{:d}$, the remaining $m-d$ dimension of the embedding are modelled as noise, using similar constraints to those imposed in (5.7): the mean of the distribution is a $(m-d)$-dimensional vector centred at $2\arccos(0) = \pi$, and the covariance is a diagonal matrix $\boldsymbol{\sigma}_k^2\mathbf{I}_{m-d}$ with positive diagonal entries. The mean value of π reflects the assumption in (5.7) of clusters of $\mathbf{X}_{:d}$ centred at zero: for $j > d$, $x_{i,j}$ is expected to be near 0, further reduced when dividing by the norm $\|\boldsymbol{x}_{i,:j}\|$, which makes the transformed coordinate centre fluctuate around π. Importantly, the assumption of cluster-specific variances on $\boldsymbol{\Theta}_{d:}$ implies that d does *not* have the simple interpretation of being the number of dimensions relevant for clustering. This fundamentally differentiates the proposed modelling framework from traditional variable selection methods within clustering (see, for example, Raftery and Dean, 2006). The parameter $d+1$ in this model represents the dimension of the latent positions that generate the network, which corresponds to the rank of the block connectivity matrix $\mathbf{B} \in [0,1]^{K\times K}$.

6.3. Empirical validation of the model

In order to validate the proposed modelling approach in (6.4), an extensive simulation study has been carried out.

6.3.1. Gaussian mixture modelling of DCSBM embeddings

First, a simple simulation is used to show that k-means or Gaussian mixtures might not be appropriate for modelling an embedding $\mathbf{X}$ arising from a degree-corrected stochastic blockmodel, even when the normalised embedding $\tilde{\mathbf{X}}$ is used. Spectral embeddings have been obtained from a simulated degree-corrected stochastic blockmodel with $n = 1{,}000$ nodes, $K = 2$ and $d = 2$, with an equal number of nodes allocated to each group, and $B_{11} = 0.1$, $B_{12} = B_{21} = 0.05$ and $B_{22} = 0.15$, corrected by parameters ρ_i sampled from a Beta$(2,1)$ distribution. Since the community allocations $\boldsymbol{z}$ are known *a priori* in the simulation, it is possible to evaluate the community-specific distributions. The results are plotted in Figure 6.2, which shows the scatterplot and histograms of the marginal distributions for the 2-dimensional adjacency spectral embedding $\mathbf{X}$ and its normalised version $\tilde{\mathbf{X}}$, and for its transformation to spherical coordinates $\boldsymbol{\Theta}$, all labelled by community. From Figure 6.2a, it is clear that applying k-means or Gaussian mixture modelling on $\mathbf{X}$ seems suboptimal, since the

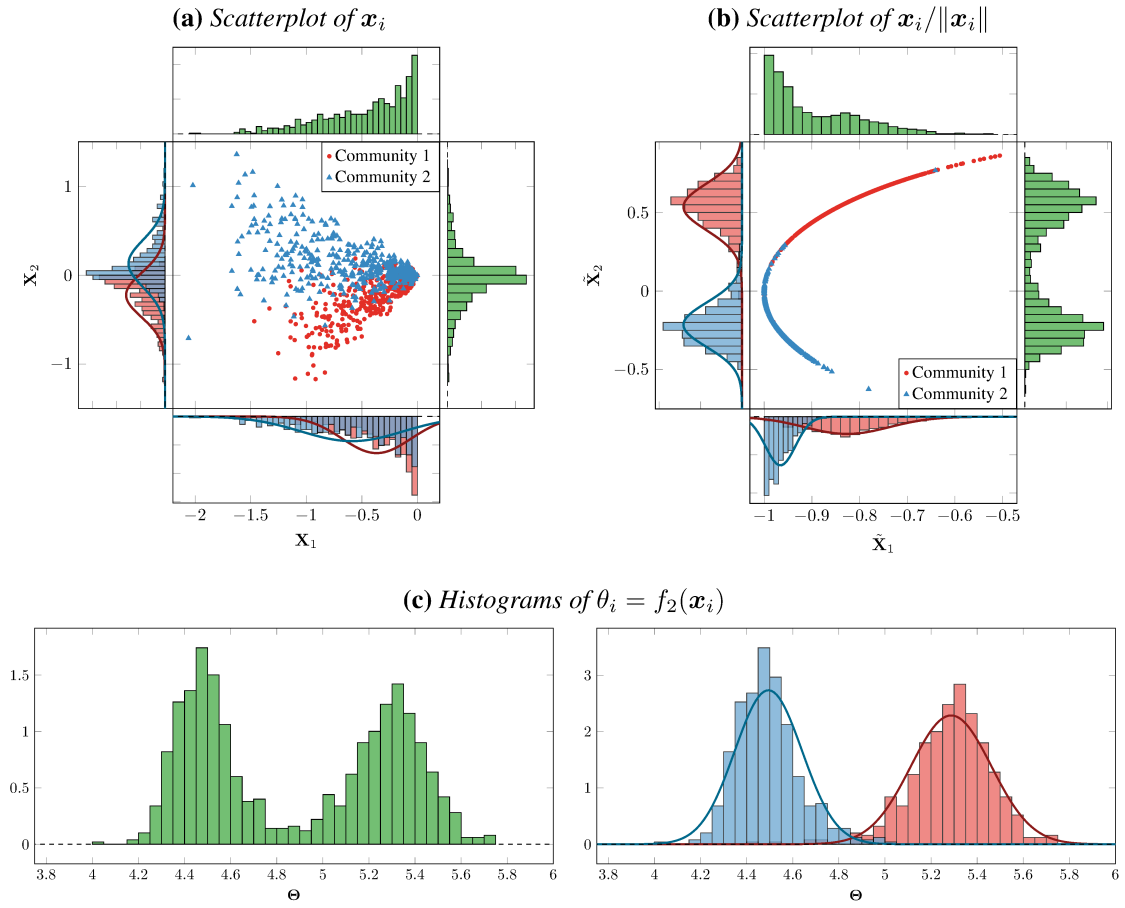


Figure 6.2.: *Plots of $\boldsymbol{x}_i$, $\boldsymbol{x}_i/\|\boldsymbol{x}_i\|$ and $\theta_i = f_2(\boldsymbol{x}_i)$ for a simulated stochastic blockmodel with $n = 1{,}000$ nodes, $K = 2$ and $d = 2$, with an equal number of nodes allocated to each group, and $B_{11} = 0.1, B_{12} = B_{21} = 0.05$ and $B_{22} = 0.15$, corrected by parameters $\rho_i \sim Beta(2, 1)$. Joint and community-specific marginal distributions with an MLE Gaussian fit are also shown.*

joint distribution or cluster-specific marginal distributions do not appear to be normally distributed. Figure 6.2b shows that row normalisation is beneficial, since the marginal distributions for $\tilde{\mathbf{X}}_2$ are normally distributed within each community, but it is clearly not appropriate for modelling the joint distribution and for at least one of the two marginal distributions for $\tilde{\mathbf{X}}_1$. On the other hand, the transformation $\boldsymbol{\Theta}$ in Figure 6.2c visually meets the assumption of normality; despite some skewness, there is clear separation between the two groups.

6.3.2. Undirected graphs

In order to validate the conjecture on the model likelihood proposed in (6.4), a simulation study has been carried out. $N = 1{,}000$ degree-corrected stochastic blockmodels with $n = 1{,}500$ nodes have

been simulated, fixing $K = 3$ and pre-allocating the nodes to equal-sized clusters. The following block community probability matrix, sampled from Uniform$(0, 1)^{K\times K}$, is used in the simulation:

$$\mathbf{B} = \begin{bmatrix} 0.66 & 0.16 & 0.59 \\ 0.16 & 0.27 & 0.07 \\ 0.59 & 0.07 & 0.81 \end{bmatrix}. \tag{6.6}$$

The matrix is positive definite and has full rank 3, implying that $d = 2$ in the embedding $\mathbf{\Theta}$. For each of the N simulations, the link probabilities are corrected using the degree-correction parameters ρ_i sampled from a Beta$(2, 1)$ distribution, and the adjacency matrices $\mathbf{A}$ are obtained using (6.1). For each of the simulated graphs ASE is calculated for a large value of m. The results are summarised in Figure 6.3.

The true underlying cluster allocations are known in the simulation, and can be used to validate the model assumptions. Figure 6.3a shows the boxplots of the community-specific means for the first two components of $\mathbf{\Theta}$. The mean values show minimal variation across the different simulations, and are clearly different from 0 and differ between clusters. On the other hand, Figure 6.3b shows the boxplots for the community-specific means in $\mathbf{\Theta}_{d:}$, which seem to be all centred at π, with small variability. Hence, the assumption on the mean structure in (6.4) seems to be justified in the simulation. In Figure 6.3c, boxplots for the community-specific variances in $\mathbf{\Theta}_{:d}$ are plotted. Again, it seems that having cluster-specific variances is sensible. Figure 6.3d shows the correlation r_{12} between $\mathbf{\Theta}_1$ and $\mathbf{\Theta}_2$ for the $K = 3$ three communities, and it is clear that the variables are highly dependent, justifying the non-diagonal covariance matrix.

A potentially more controversial modelling choice is the community-specific variance for the last $m - d$ components of the embedding. Figure 6.3e shows the variances for each community on different dimensions exceeding 2. It is clear that the variance is different across different communities. This consideration is reinforced by Figure 6.3f, which shows the boxplots of Kolmogorov-Smirnov (KS) scores obtained from fitting community-specific Gaussian distributions on the dimensions exceeding d, compared to the KS score for a Gaussian fit on all the estimated latent positions for the corresponding dimension. Clearly, a correct modelling approach must use community-specific variances. This implies that some residual cluster information is present also in the last $m - d$ dimensions of the embedding. Finally, Figure 6.3g plots the boxplots of correlations between $\mathbf{\Theta}_1$, $\mathbf{\Theta}_3$ and $\mathbf{\Theta}_4$. Consistent with the model in (6.4), the correlations are scattered around 0, and the assumption of independence between those components seems reasonable.

6.3.3. Normality of the transformed embedding

The most important comparison is to establish whether the embedding $\mathbf{\Theta}$ is better suited to Gaussian mixture modelling than the row-normalised embedding $\tilde{\mathbf{X}}$ traditionally used in the literature. To make this comparison, the p-values for the two Mardia tests for multivariate normality (Mardia,

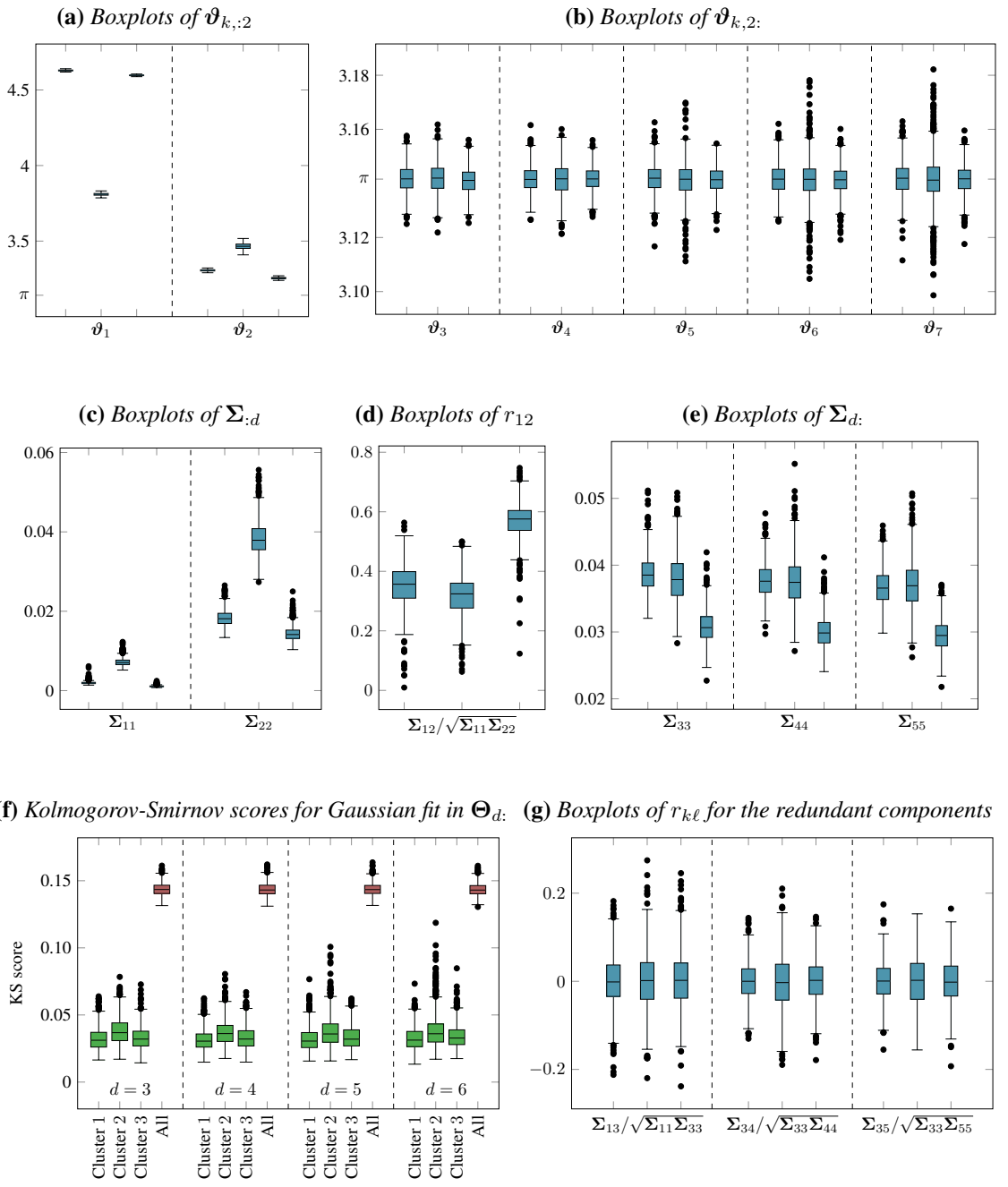


Figure 6.3.: *Boxplots for* $N = 1{,}000$ *simulations of a degree-corrected stochastic blockmodel with* $n = 1{,}500$ *nodes,* $K = 3$*, equal number of nodes allocated to each group, and* $\mathbf{B}$ *described in* (6.6)*, corrected by parameters* ρ_i *sampled from a Beta*$(2, 1)$ *distribution.*

1970) have been calculated for each community-specific distribution, for each simulated graph. The Mardia tests are based on multivariate extensions of skewness and kurtosis: assume a sequence of random vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell \in \mathbb{R}^d$, and define the sample mean and sample covariance as $\bar{\boldsymbol{x}} = \sum_{i=1}^{\ell} \boldsymbol{x}_i/\ell$ and $\mathbf{S} = \sum_{i=1}^{\ell}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top/\ell$ respectively. Mardia (1970) defines two test statistics for multivariate skewness and kurtosis:

$$T_S = \frac{1}{6\ell}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\left[(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top \mathbf{S}^{-1}(\boldsymbol{x}_j - \bar{\boldsymbol{x}})\right]^3,$$

$$T_K = \sqrt{\frac{\ell}{8d(d+2)}}\left\{\frac{1}{\ell}\sum_{i=1}^{\ell}\left[(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top \mathbf{S}^{-1}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})\right]^2 - \frac{d(d+2)(\ell-1)}{\ell+1}\right\}.$$

Under the null hypothesis of multivariate normality, $T_S \to \chi^2\{d(d+1)(d+2)/6\}$ and $T_K \to \mathbb{N}(0,1)$ in distribution for $\ell \to \infty$ (Mardia, 1970). Given an observed value of T_S and T_K, p-values p_S and p_K can be calculated from the asymptotic distribution. For the row-normalised embedding, such p-values are denoted $p_S(\tilde{\mathbf{X}})$ and $p_K(\tilde{\mathbf{X}})$. Similarly, $p_S(\boldsymbol{\Theta})$ and $p_K(\boldsymbol{\Theta})$ correspond to the p-values obtained from the transformed embedding $\boldsymbol{\Theta}$.

Figure 6.4 reports the histograms of the differences of the p-values for the two tests applied for each community on the spherical embedding $\boldsymbol{\Theta}_{:2}$ and the row-normalised embedding $\tilde{\mathbf{X}}_{:3}$, obtained from the simulation in Section 6.3.2, assuming the true value of d is known. Clearly, both Figure 6.4a and Figure 6.4b show that the community-specific distributions in $\tilde{\mathbf{X}}_{:3}$ have more extreme values of skewness and kurtosis when compared to the corresponding distribution in $\boldsymbol{\Theta}_{:2}$, confirming the impression in Figure 6.2 that the transformation (6.3) to $\boldsymbol{\Theta}$ tends to *Gaussianise* the embeddings $\mathbf{X}$ or $\tilde{\mathbf{X}}$. Despite this improvement, some skewness and kurtosis are still present in $\boldsymbol{\Theta}$, as visually confirmed by Figure 6.2c.

6.3.4. Bipartite graphs

An analogous simulation has been repeated for a bipartite stochastic co-blockmodel (Rohe, Qin, and Yu, 2016) with degree-correction, with $|V_1| = 1{,}500$ and $|V_2| = 2{,}500$ nodes, $K = 3$, $K' = 5$, equal number of nodes allocated to each community, and $\mathbf{B} \sim \text{Uniform}(0,1)^{K \times K'}$. The resulting matrix $\mathbf{B}$ has the following form:

$$\mathbf{B} = \begin{bmatrix} 0.266 & 0.433 & 0.698 & 0.834 & 0.847 \\ 0.620 & 0.470 & 0.400 & 0.632 & 0.541 \\ 0.884 & 0.940 & 0.976 & 0.914 & 0.547 \end{bmatrix}. \tag{6.7}$$

As before, the degree correction parameters $\rho_i,\ i = 1, \ldots, 1{,}500$, and $\rho'_j,\ j = 1, \ldots, 2{,}500$, were sampled from a Beta$(2,1)$ distribution for each of the N graphs, and the adjacency matrices were obtained setting $A_{ij} \sim \text{Bernoulli}(\rho_i \rho'_j B_{z_i z'_j})$ (*cf.* Section 5.5). The results are reported in Fig-

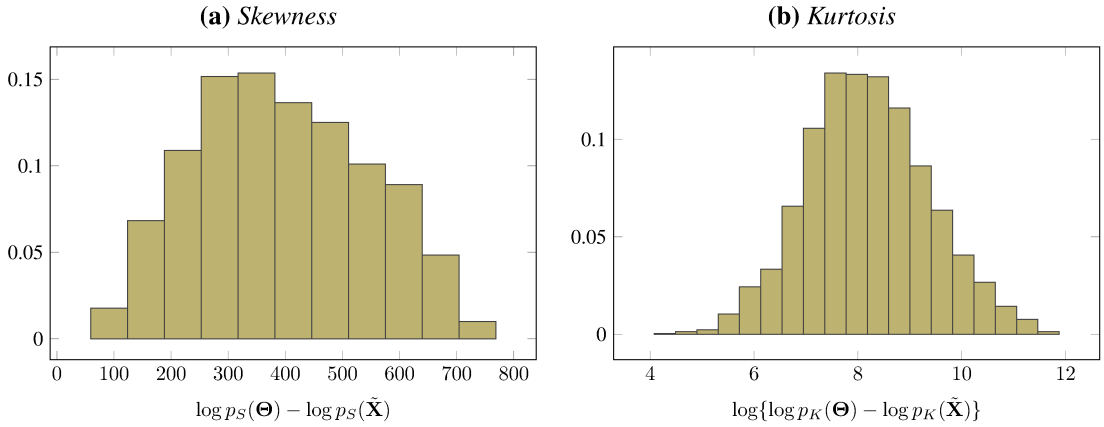


Figure 6.4.: *Histograms of differences of the p-values for the two Mardia multivariate normality tests for skewness and kurtosis on $\tilde{\mathbf{X}}_{:3}$ and $\boldsymbol{\Theta}_{:2}$.*

ure 6.5. For one of the simulations, the scatterplot of the DASE embedding $\mathbf{Y}$, the row-normalised embedding $\tilde{\mathbf{Y}}$, and the corresponding spherical coordinates $\boldsymbol{\Theta}'$, are plotted in Figure 6.5a, 6.5b and 6.5c. The scatterplots again show that the transformation to spherical coordinates makes the embedding more suited to Gaussian mixture modelling, and the curvature due to the degree-correction or row-normalisation is reduced. Figure 6.5d shows the boxplots of the community-specific means on $\boldsymbol{\Theta}_{:2}$ and $\boldsymbol{\Theta}'_{:2}$. In this case, there should be $d = \min\{K, K'\} - 1$ dimensions of $\boldsymbol{\Theta}$ for which the cluster means are different from π, and this seems to be confirmed when comparing Figure 6.5d and Figure 6.5e, where the latter shows the community-specific boxplots for the means on $\boldsymbol{\Theta}_{2:}$ and $\boldsymbol{\Theta}'_{2:}$. Similarly, Figures 6.5f and 6.5g show similar plots for the within-community variances for $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}'$, confirming that the community-specific variability is different on $\boldsymbol{\Theta}_{2:}$ and $\boldsymbol{\Theta}'_{2:}$ across the different communities.

6.4. Model selection and parameter estimation

A full Bayesian treatment of the model (6.4) would require assigning prior distributions to the parameters and sampling from the analytically intractable posterior distribution. For this model, an extension of the MCMC procedure from in Section 5.3 would be required. Unfortunately, the MCMC algorithm is computationally expensive and does not scale well in the number of nodes n and in the number of dimensions m. Therefore, a classical approach of model comparison via Bayes factors (Kass and Raftery, 1995), already used by Yang et al. (2020) in the context of simultaneous model selection in SBMs, is adopted. A Bayes factor is obtained as the ratio of marginal likelihoods for two competing models. The *marginal likelihood*, also known as *integrated likelihood*, or *evidence*, is $p(\boldsymbol{\Theta}|d, K)$, where all the model parameters except (d, K) are

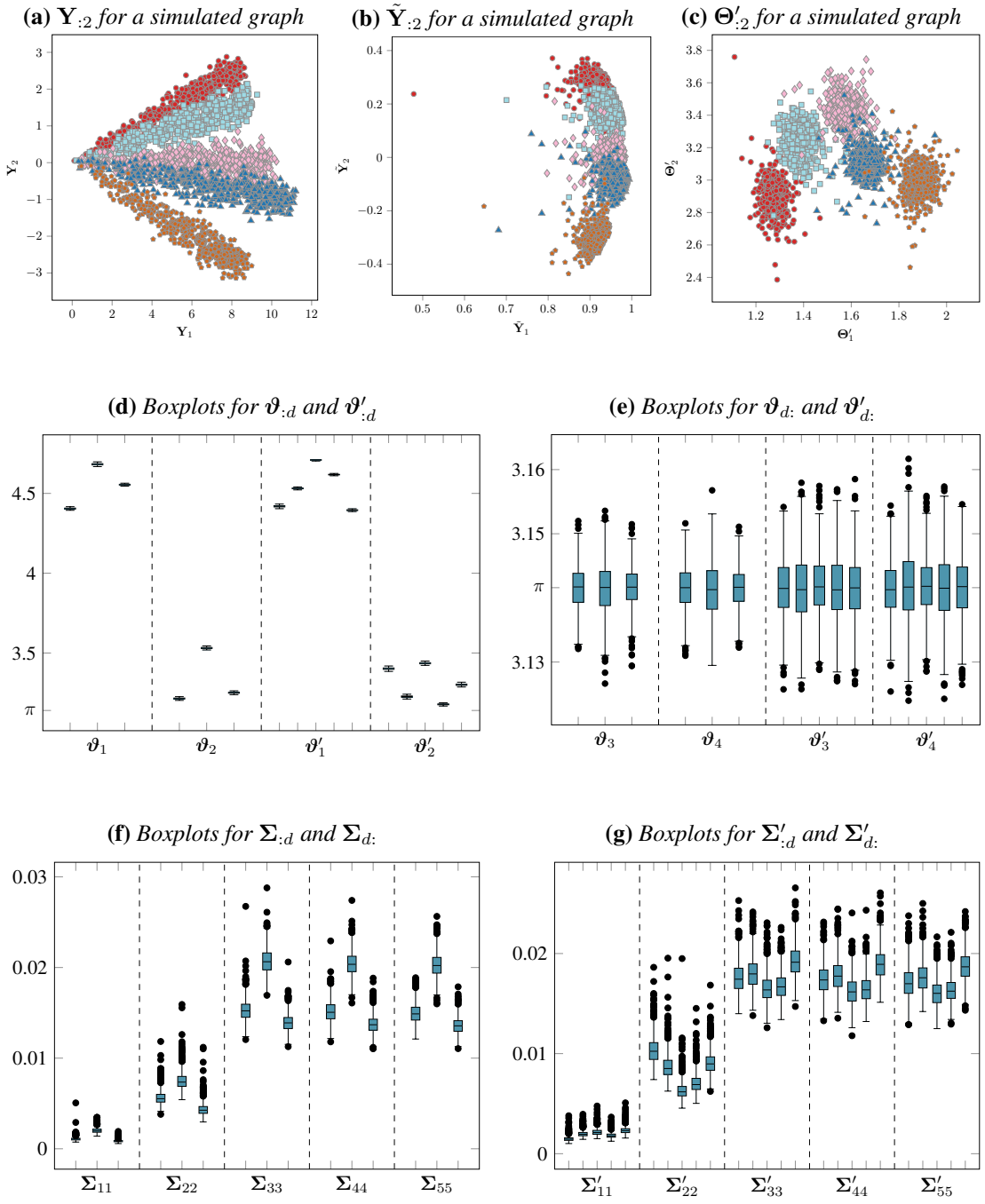


Figure 6.5.: *Boxplots and scatterplots for* $N = 1{,}000$ *simulations of a degree-corrected stochastic co-blockmodel with* $|V_1| = 1{,}500$ *and* $|V_2| = 2{,}500$ *nodes,* $K = 3$*,* $K' = 5$*, equal number of nodes allocated to each group, and* $\mathbf{B}$ *described in* (6.7)*, corrected by parameters* ρ_i *sampled from a Beta*$(2, 1)$ *distribution.*

marginalised out. For the model in (6.4), the marginal likelihood is analytically not tractable. In those cases, a Laplace approximation of $2p(\boldsymbol{\Theta}|d,K)$, known as the Bayesian information criterion (BIC, Schwarz, 1978), is commonly used. Assume that the maximum likelihood estimate (MLE) of the model parameters are $\{\hat{\boldsymbol{\vartheta}}_k, \hat{\boldsymbol{\Sigma}}_k, \hat{\boldsymbol{\sigma}}_k^2, \hat{\psi}_k\}_{k=1,\dots,K}$, where, similarly to Section 4.3, (4.8), $\psi_1,\dots,\psi_K$ represent the mixing proportions, such that $\mathbb{P}(z_i=k|K)=\psi_k,\ k=1,\dots,K$, where $\psi_k \geq 0\ \forall k$ and $\sum_{k=1}^K \psi_k = 1$. Then, $-2p(\boldsymbol{\Theta}|d,K) \approx \text{BIC}(d,K)$, where the BIC is defined as:

$$\text{BIC}(d,K) = -2\sum_{i=1}^{n}\log\left\{\sum_{k=1}^{K}\hat{\psi}_j\phi(\boldsymbol{\theta}_{i,:d};\hat{\boldsymbol{\vartheta}}_k,\hat{\boldsymbol{\Sigma}}_k)\prod_{j=d+1}^{m}\phi(\theta_{i,j};\pi,\hat{\sigma}^2_{k,j})\right\} + K\log(n)(d^2/2+d/2+m+1). \quad (6.8)$$

The first part of the BIC (6.8) is known as *incomplete data negative log-likelihood*, where $\boldsymbol{z}$ is marginalised out, whereas the second part is a penalty for large numbers of parameters. The estimated values of d and K are obtained using grid search: $(\hat{d},\hat{K}) = \text{argmin}_{(d,K)}\,\text{BIC}(d,K)$. The latent dimension d has range $\{1,\dots,m\}$, whereas $K \in \{1,\dots,n\}$. In practice, it is convenient to fix a maximum number of clusters $K^\star$ in the grid search procedure, such that $K \in \{1,\dots,K^\star\}$.

From (6.8), it follows that maximum likelihood estimates of the Gaussian model parameters are required for each pair (d,K). The EM algorithm (Dempster, Laird, and Rubin, 1977), already used in Section 4.3.1, is typically used for problems involving likelihood maximisation in model based clustering (for example, Fraley and Raftery, 2002). Finding maximum likelihood estimates for the model parameters in (6.4) only requires a simple modification of the standard algorithm for Gaussian mixture models (see, for example, McLachlan and Krishnan, 2007), adding constraints on the means and covariances of the last $m-d$ components. Given the maximum likelihood estimates, the community allocations are estimated from the incomplete data likelihood as:

$$\hat{z}_i = \underset{j\in\{1,\dots,K\}}{\text{argmax}}\left\{\hat{\psi}_j\phi_d\left(\boldsymbol{\theta}_{i,:d};\hat{\boldsymbol{\vartheta}}_j,\hat{\boldsymbol{\Sigma}}_j\right)\right\},\ i=1,\dots,n, \quad (6.9)$$

where the second Gaussian term in the likelihood (6.5), accounting for the last $m-d$ components of the embedding, is removed to reduce the bias-variance tradeoff (Yang et al., 2020). The estimates (6.9) directly follow the procedure of Yang et al. (2020), which shows via simulations on stochastic blockmodels that, after estimating *all* the parameters for the m-dimensional model (5.7), estimates of the community memberships tend to be superior when only the initial d dimensions of the embedding are considered, as in (6.9).

For a given pair (d,K), fast estimation and convergence can be achieved by initialising the EM algorithm with the maximum likelihood estimates of a Gaussian mixture model fitted on the initial d components of the m-dimensional embedding. Most programming languages have statistical libraries for fast estimation of Gaussian mixture models; for example *mclust* in *R*, or *sklearn* in

python. This approach for initialisation will be followed in Section 6.5.2. The full procedure is summarised in Algorithm 6.1.

Algorithm 6.1: Estimation of the latent dimension d, number of communities K, and community allocations $\boldsymbol{z}$.

Input: transformed embedding $\boldsymbol{\Theta} \in \mathbb{R}^{n \times m}$, maximum number of communities $K^\star$.
Result: estimated community allocations $\hat{\boldsymbol{z}}$ and estimated latent dimensionality $\hat{d}$.

```
for d = 1, 2, ..., m do
    for K = 1, ..., K* do
        calculate maximum likelihood estimates {ψ̂_j, ϑ̂_j, Σ̂_j, σ̂²_j}_{k=1,...,K} using the EM
          algorithm,
        calculate BIC(d, K),
    end
end
(d̂, K̂) ← argmin_{(d,K)} BIC(d, K),
estimate the community allocations using (6.9) and (d̂, K̂).
```

An extension of the inferential and model selection procedure described above to a fully Bayesian setting could also be considered, using the collapsed Metropolis-within-Gibbs algorithm from Section 5.3. An advantage of adopting a Bayesian approach would be that d and K are considered as parameters of the posterior distribution, and a grid search over (d, K) pairs would not be required. On the other hand, the main drawback is the computational burden and the lack of scalability of such an approach.

6.5. Applications and results

In this section, the model selection procedure is assessed on simulated degree-corrected stochastic blockmodels and real-world bipartite graphs obtained from the network flow data collected at Imperial College London (Section 2.2). The quality of the clustering is evaluated using the adjusted Rand index (ARI, Hubert and Arabie, 1985). Suppose underlying true cluster allocations $\boldsymbol{z}$, with K groups, and corresponding estimates $\hat{\boldsymbol{z}}$ and $\hat{K}$ estimated communities. The ARI is defined as:

$$\mathrm{ARI}(\boldsymbol{z}, \hat{\boldsymbol{z}}) = \frac{\sum_{k=1}^{K}\sum_{\ell=1}^{\hat{K}}\binom{q_{k\ell}}{2} - \left[\sum_{k=1}^{K}\binom{q_{k\bullet}}{2}\sum_{\ell=1}^{\hat{K}}\binom{\hat{q}_{\ell\bullet}}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_{k=1}^{K}\binom{q_{k\bullet}}{2} + \sum_{\ell=1}^{\hat{K}}\binom{\hat{q}_{\ell\bullet}}{2}\right] - \left[\sum_{k=1}^{K}\binom{q_{k\bullet}}{2}\sum_{\ell=1}^{\hat{K}}\binom{\hat{q}_{\ell\bullet}}{2}\right] \Big/ \binom{n}{2}},$$

where $q_{k\ell} = \sum_{i=1}^{n} \mathbb{1}_k\{z_i\}\mathbb{1}_\ell\{\hat{z}_i\}$, $q_{k\bullet} = \sum_{i=1}^{n} \mathbb{1}_k\{z_i\}$, and $\hat{q}_{\ell\bullet} = \sum_{i=1}^{n} \mathbb{1}_\ell\{\hat{z}_i\}$. Higher values of the ARI correspond to better clustering performance, reaching a maximum of 1 for perfect agreement between the estimated clustering and the true labels. In this section, the maximum number of clusters for the grid search procedure is set to $K^\star = 30$.

Table 6.1.: *Estimated performance from $N = 250$ simulated degree-corrected stochastic blockmodels with $n = 1{,}000$, $\mathbf{B}$ sampled from Uniform$(0, 1)$ and degree corrections sampled from Beta$(2, 1)$, and equally sized communities, for $K = 2$ and $K = 3$.*

	$K = 2$			$K = 3$		
	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\boldsymbol{\Theta}$	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\boldsymbol{\Theta}$
Proportion of correct estimates of d	0.780	0.740	0.764	0.720	0.668	0.776
Proportion of correct estimates of K	0.012	0.052	0.465	0.004	0.012	0.364
Average Adjusted Rand Index	0.344	0.462	0.724	0.573	0.686	0.822

6.5.1. Synthetic networks

The performance of the double model selection procedure described in Section 6.4 is evaluated on simulated degree-corrected stochastic blockmodels. $N = 250$ undirected graphs with $n = 1{,}000$ nodes and $K \in \{2, 3\}$ communities were simulated, randomly selecting the entries of $\mathbf{B}$ from Uniform$(0, 1)$ and sampling the degree correction parameters from Beta$(2, 1)$. The nodes were allocated to communities of equal size. For each of the graphs, the model (5.7) is applied to the ASE $\mathbf{X}$ and its row-normalised version $\tilde{\mathbf{X}}$ for $m = 10$, selecting the estimates of d and K using BIC. Also, the model (6.4) is fitted to $\boldsymbol{\Theta}$, estimating d and K using the selection procedure described in Section 6.4. The results are presented in Table 6.1.

The table shows that the three different methodologies have similar performance when estimating the correct value of the latent dimension d, but differ significantly in the ability to estimate the number of communities K. In particular, the Gaussian mixture model is not well suited to either the standard embedding $\mathbf{X}$ nor the row-normalised $\tilde{\mathbf{X}}$, and the distortion caused by the degree-corrections and row-normalisation does not allow correct estimation of K. This problem is alleviated when the spherical coordinates are used. The improvement is reflected in a significant difference in the clustering performance, demonstrated by the average ARI scores.

A similar simulation has been repeated for degree-corrected bipartite stochastic co-blockmodels. Again, $N = 250$ graphs were generated, setting $K = 2$, $K' = 3$, and communities of equal size. The entries of the co-block connectivity matrix $\mathbf{B} \in [0, 1]^{K \times K'}$ were sampled from Uniform$(0, 1)$, and the degree corrections from Beta$(2, 1)$. The adjacency matrix was obtained by simulating $A_{ij} \sim$ Bernoulli$(\rho_i \rho'_j B_{z_i z'_j})$, and the DASE $\mathbf{X}$ and $\mathbf{Y}$ with $m = 10$ were calculated. As before, the model (5.7) was applied independently on $\mathbf{X}$ and $\mathbf{Y}$, and on the row-normalised embeddings $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$. Also, (6.4) was fitted to $\boldsymbol{\Theta} = f_{10}(\mathbf{X})$ and $\boldsymbol{\Theta}' = f_{10}(\mathbf{Y})$. The results are reported in Table 6.2.

The results match those obtained for the undirected graph model (Table 6.1). In particular, it is not recommended to use the DASE embeddings $\mathbf{X}$ and $\mathbf{Y}$ to estimate the number of communities K, and consequently the clustering structure, whereas $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}'$ consistently provide higher values

Table 6.2.: *Estimated performance from $N = 250$ simulated bipartite degree-corrected stochastic block-models with $n = 1{,}000$, $n' = 1{,}500$, equally-sized communities with $K = 2$, $K' = 3$, $\mathbf{B}$ entries sampled from Uniform$(0, 1)$ and degree corrections sampled from Beta$(2, 1)$.*

	$K = 2$			$K' = 3$		
	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\boldsymbol{\Theta}$	$\mathbf{Y}$	$\tilde{\mathbf{Y}}$	$\boldsymbol{\Theta}'$
Proportion of correct estimates of d	0.756	0.596	0.584	0.792	0.612	0.600
Proportion of correct estimates of K	0.004	0.020	0.328	0.012	0.020	0.400
Average Adjusted Rand Index	0.416	0.534	0.794	0.407	0.507	0.763

of the ARI. Table 6.2 also shows that the estimates of d based on the model (5.7) on $\mathbf{X}$ and $\mathbf{Y}$ seem to be more accurate than the competing methodologies. It might be therefore tempting to construct a hybrid model that uses $\boldsymbol{\Theta}_{:d}$ for the top-d embeddings and $\mathbf{X}_{d:}$ for the remaining components. Unfortunately, model comparison via BIC is not possible in this hybrid setting.

6.5.2. Imperial College network flow data

The performance of the model has been evaluated on three bipartite networks obtained from the network flow data collected at Imperial College London. Similarly to Section 5.6.4, the edges relate to all the HTTP (port 80) and HTTPS (port 443) connections observed between 1st January, 2020 and 31st January, 2020 from machines hosted within the Imperial College London campuses. The source nodes V_1 are computers hosted in college laboratories, and the destination nodes V_2 are internet servers. The two node sets are non-overlapping. Only edges for which the percentage of arrival times between 7am and 12am is larger than 95% were considered. The three bipartite networks analysed in this chapter are:

i. *ICL1*: a $628 \times 54{,}111$ graph that includes 628 hosts in the departments of Electrical & Electronic Engineering, Earth Science & Engineering, and Physics, forming a total of $668{,}155$ links,
ii. *ICL2*: a $439 \times 60{,}635$ graph that includes 439 hosts in the departments of Chemistry, Civil & Environmental Engineering, Mathematics, and the School of Medicine, forming a total of $717{,}912$ links. This graph was previously analysed in Section 5.6.4,
iii. *ICL3*: a $1{,}011 \times 84{,}664$ graph that includes $1{,}011$ hosts in the departments of Aeronautical Engineering, Civil & Environmental Engineering, Electrical & Electronic Engineering, Mathematics, Mechanical Engineering, and Physics, forming a total of 1,470,074 links.

For the source nodes forming those networks, an underlying community structure is given by the department or building to which each machine belongs. The labels for building and department are identical for ICL1 and ICL2, but slightly differ for ICL3, since the departments of Aeronautical Engineering and Mechanical Engineering share the same building. Students use the computer

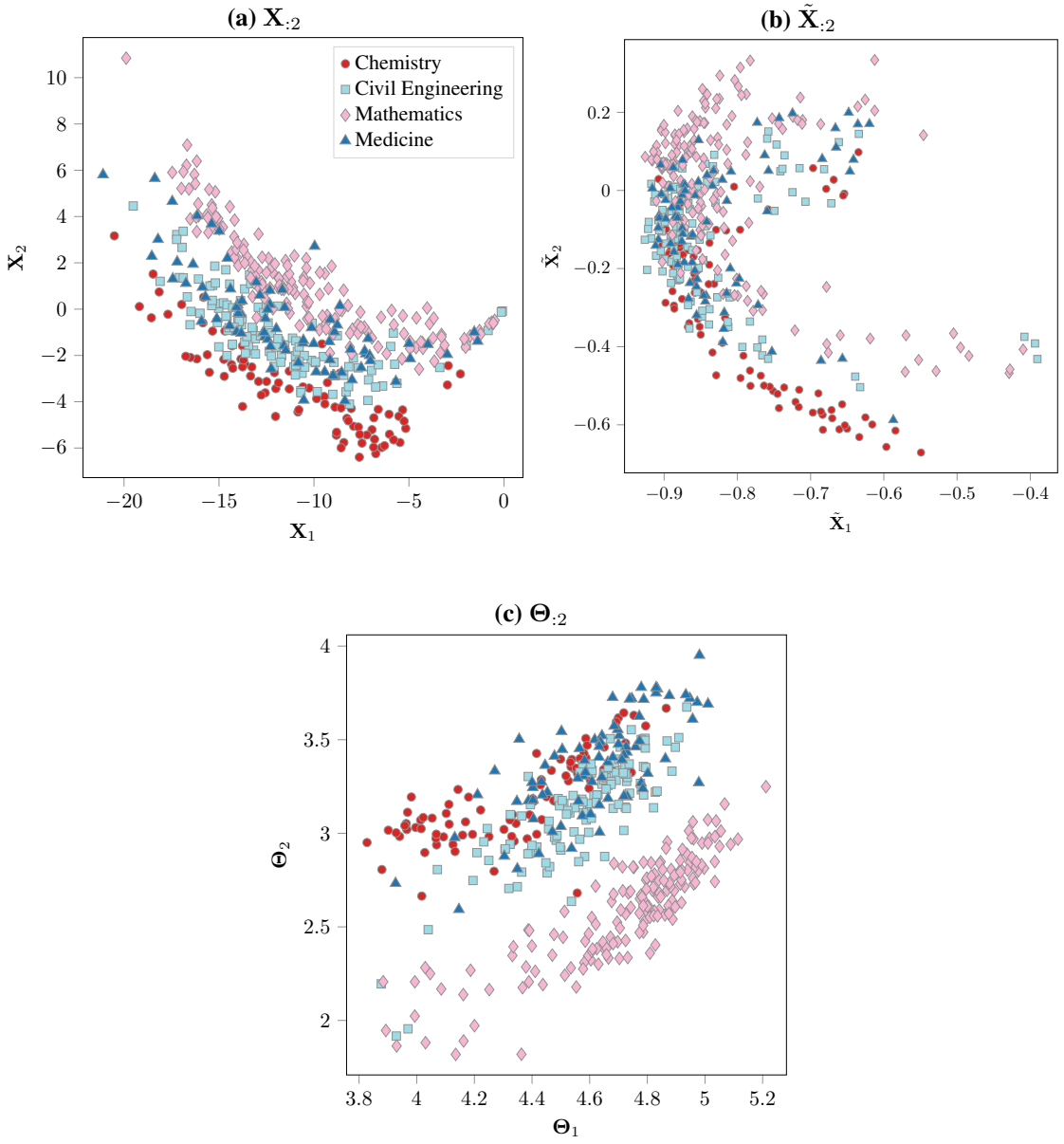


Figure 6.6.: *ICL2: scatterplot of the two leading dimensions of* $\mathbf{X}$, $\tilde{\mathbf{X}}$ *and* $\mathbf{\Theta}$.

laboratories for tutorials and classes, and therefore some variation is expected in the activities of different machines across different departments and buildings. Figure 6.6 shows the scatterplots of the leading 2 dimensions of the m-dimensional embeddings $\mathbf{X}$, $\tilde{\mathbf{X}}$ and $\mathbf{\Theta}$ from ICL2 for $m = 30$, showing that the clustering task is particularly tricky in such datasets, and there is not much separation between the communities.

For each of the three network graphs i-iii, the DASE has been calculated for $m = 30$ and $m = 50$, and the model in (6.4) has been fitted on the source embeddings $\mathbf{X}$, the row-normalised version $\tilde{\mathbf{X}}$

Table 6.3.: *Estimates of* (d, K) *for the embeddings* $\mathbf{X}$, $\tilde{\mathbf{X}}$ *and* $\mathbf{\Theta}$ *for* $m = 30$ *and* $m = 50$.

(a) $m = 30$

	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\mathbf{\Theta}$
ICL1	$(26, 6)$	$(11, 6)$	$(11, 4)$
ICL2	$(28, 5)$	$(8, 7)$	$(15, 4)$
ICL3	$(24, 10)$	$(17, 5)$	$(15, 5)$

(b) $m = 50$

	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\mathbf{\Theta}$
ICL1	$(22, 5)$	$(11, 6)$	$(11, 4)$
ICL2	$(29, 4)$	$(8, 7)$	$(15, 4)$
ICL3	$(25, 6)$	$(13, 6)$	$(16, 5)$

Table 6.4.: *ARI for clustering on* $\mathbf{X}$, $\tilde{\mathbf{X}}$ *and* $\mathbf{\Theta}$, *based on the estimates of* (d, K) *in Table 6.3.*

(a) $m = 30$

	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\mathbf{\Theta}$
ICL1	0.259	0.324	0.418
ICL2	0.441	0.736	0.938
ICL3 (building)	0.246	0.342	0.409
ICL3 (department)	0.273	0.341	0.356

(b) $m = 50$

	$\mathbf{X}$	$\tilde{\mathbf{X}}$	$\mathbf{\Theta}$
ICL1	0.262	0.317	0.418
ICL2	0.359	0.743	0.938
ICL3 (building)	0.269	0.265	0.364
ICL3 (department)	0.272	0.296	0.358

and the transformation $\mathbf{\Theta}$. The resulting estimated values of d and K, obtained using the minimum BIC, are reported in Table 6.3. In order to reduce the sensitivity to initialisation, the model was fitted 10 times for each pair (d, K), and only the estimates corresponding to the smallest BIC were retained.

Comparing Table 6.3a and 6.3b, it seems that the estimates of d and K based on the embedding $\mathbf{\Theta}$ are more stable for the two different values of m when compared to the estimates based on $\mathbf{X}$ or $\tilde{\mathbf{X}}$. Also, the estimates of K based on $\mathbf{\Theta}$ seem to be closer to the underlying true number of communities in the three networks. In particular $K = 3$ for ICL1, $K = 4$ for ICL2, and $K = 5$ or $K = 6$ for ICL3, based on the number of buildings and departments respectively.

Based on the estimates in Table 6.3, the estimated community allocations z_i were obtained using (6.9), and the adjusted Rand scores were calculated using the department and building as labels. The results are reported in Table 6.4. Clearly, clustering on the embedding $\mathbf{\Theta}$ seems to outperform the alternatives $\mathbf{X}$ and $\tilde{\mathbf{X}}$ in all the three networks. In some cases, the improvement is substantial, for example in ICL2, where the proposed method reaches the score 0.938, corresponding to only 9 misclassified nodes out of 439, particularly remarkable considering the lack of separation of the clusters in Figure 6.6.

6.6. Conclusion and discussion

In this chapter, a novel method for spectral clustering under the degree-corrected stochastic block-model has been proposed. The model is based on a transformation to spherical coordinates of

the commonly used adjacency spectral embedding. Such a transformation seems more suited to Gaussian mixture modelling than the alternative row-normalised embedding, which is widely used in the literature for spectral clustering. The proposed methodology is then incorporated within a simultaneous model selection scheme that allows the model dimension d and the number of communities K to be determined. The optimal values of d and K are chosen using the popular Bayesian information criterion (BIC). The framework also extends simply to include directed and bipartite graphs. Results on synthetic data and real-world computer networks show superior performance over competing methods.

6.6.1. Limitations and future work

Exact modelling of the embeddings under the degree corrected stochastic blockmodel is a complex task, and the Gaussian model for $\boldsymbol{\Theta}$ is only an approximation of the theoretical distribution of the spherical coordinates. An alternative choice might be the general projected normal distribution (Hernandez-Stumpfhauser, Breidt, and van der Woerd, 2017). The m-dimensional random vector $\mathbf{U} = \mathbf{X}/\|\mathbf{X}\|$ has a projected normal distribution $\text{PN}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ if $\mathbf{X} \sim \mathbb{N}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The density of the projected normal is usually expressed in the spherical coordinate system after a suitable transformation of $\mathbf{U}$. Importantly, the distribution of $\mathbf{U}$ does not change after scaling by any positive constant ρ, hence $\text{PN}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \text{PN}_m(\rho\boldsymbol{\mu}, \rho^2\boldsymbol{\Sigma})$. Unfortunately, in general $\mathcal{S}(\rho_i \boldsymbol{\mu}_{z_i}) \neq \rho_i^2 \mathcal{S}(\boldsymbol{\mu}_{z_i})$ in (6.2) (see Rubin-Delanchy et al., 2017), which implies that the community-specific distribution on the sphere is approximately a mixture of projected normal distributions with node-specific covariances. Therefore, the projected normal model would also be an approximated model under the degree-corrected stochastic blockmodel, similarly to the Gaussian model on spherical coordinates proposed in this chapter. Overall, this chapter demonstrated that the Gaussian mixture model on the angular coordinates is a suitable approximation in a variety of different scenarios, and it significantly improved the clustering performance.

Figure 6.6a also shows that the communities in the embedding $\mathbf{X}$ are not exactly *rays* as the theory predicts, but there is additional structure and curvature which the DCSBM does not account for, and which is consequently reflected in estimates of d larger than K. Obtaining values of d that largely exceed K might represent evidence that the data generating process does not follow exactly a degree-corrected stochastic blockmodel, since the rank of $\mathbf{B}$ has maximum value $\min\{K, K'\}$. Therefore, future research on the Imperial College NetFlow data should investigate models that impose geometric requirements on the latent space, for example the latent structure model (LSM, Athreya et al., 2020). Alternatively, different choices in the latent position model kernel function in (5.1), other than the inner product used in random dot product graphs, could be considered, allowing for curvature in the embeddings.

Part III.

Link prediction

Part II introduced random dot product graphs as powerful tools for spectral clustering of networks. In the third and final part of the thesis, the focus shifts to a fundamental task for computer network security: *link prediction*. First, Chapter 7 demonstrates that random dot product graphs could also be efficiently used for dynamic link prediction on large networks. Second, Chapter 8 describes a latent position model for link prediction based on Poisson matrix factorisation, which has suitable properties for application to computer networks.

7

Dynamic link prediction using random dot product graphs

Link prediction is defined as the task of predicting the presence of an edge between two nodes in a network, based on latent characteristics of the graph (Liben-Nowell and Kleinberg, 2007). The ability to correctly predict and associate anomaly scores with the connections in a network is valuable for the cyber-defence of enterprises. Adversaries attacking a computer network often affect relationships between nodes within these networks, such as users authenticating to computers, or clients connecting to servers. Therefore, predicting links in order to identify significant deviations in expected behaviour could lead to the detection of an otherwise extremely damaging network breach. Random dot product graphs, introduced in Chapters 5 and 6, represent a flexible modelling approach, and are easily estimated from spectral and singular value decompositions, which can be efficiently calculated even in large networks. This chapter discusses how random dot product graphs can be used in practice for link prediction.

Away from cyber applications, the link prediction problem has been an active field of research (see, for example, Dunlavy, Kolda, and Acar, 2011; Lü and Zhou, 2011; Menon and Elkan, 2011), being similar, especially in its *static* formulation, to recommender systems (Adomavicius and Tuzhilin, 2005). Static link prediction (for example, Clauset, Moore, and Newman, 2008) aims at filling in missing entries in a single incomplete graph adjacency matrix, as opposed to *temporal* link prediction (for example, Dunlavy, Kolda, and Acar, 2011), which aims at predicting future snapshots of the graph given one or more fully observed snapshots. Static link prediction problems have been successfully tackled using probabilistic matrix factorisation methods, especially classical Gaussian matrix factorisation (Salakhutdinov and Mnih, 2007), and are currently widely used in the technology industry (see, for example, Agarwal, Zhang, and Mazumder, 2011; Khanna et al., 2013; Paquet and Koenigstein, 2013). The problem is also actively studied in other disciplines,

for example physics (Lü and Zhou, 2011), biology (Navlakha, Gitter, and Bar-Joseph, 2012), or computer science, where graph neural networks (Zhang and Chen, 2018; Wu et al., 2020) have recently gained popularity.

This chapter discusses methods for *temporal link prediction* (Dunlavy, Kolda, and Acar, 2011): given a sequence of adjacency matrices $\mathbf{A}_1, \dots, \mathbf{A}_T$ observed over time, the main objective is to reliably predict $\mathbf{A}_{T+1}$. It is assumed that snapshots of a dynamic network are observed at discrete time points $t = 1, \dots, T$, obtaining a sequence of graph snapshots $\mathbb{G}_t = (V, E_t)$. The set V, representing the set of nodes, is invariant over time, whereas the set E_t is a time dependent edge set, where $(i, j) \in E_t$ for $i, j \in V$, if i connected to j at least once during the time period $(t-1, t]$. Each snapshot of the graph can be characterised by the adjacency matrix $\mathbf{A}_t \in \{0, 1\}^{n \times n}$, where $n = |V|$ and for $1 \leq i, j \leq n$, $A_{ijt} = \mathbb{1}_{E_t}\{(i, j)\}$, such that $A_{ijt} = 1$ if a link between the nodes i and j exists in $(t-1, t]$, and $A_{ijt} = 0$ otherwise.

In this chapter, temporal link prediction techniques based on random dot product graphs are discussed and compared. The main contribution is to adapt existing methods for multiple RDPG graph inference for temporal link prediction. Overall, this chapter provides insights into the predictive capability of random dot product graphs, and gives useful guidelines for practitioners on the optimal choice of the embedding for temporal link prediction. Furthermore, this chapter proposes methods to combine the information obtained via spectral methods with time series models, to capture the temporal dynamics of the observed graphs. The proposed methodologies will be extensively compared on real world and simulated networks in Section 7.3. It will be shown that this approach significantly improves the predictive performance of multiple RDPG models, especially when the network presents a seasonal or temporal evolution. Importantly, the strategies for combination of individual embeddings, and their time series extensions, can in principle be applied to *any* embedding method for static graphs, despite the main focus on RDPGs of this chapter. This chapter mainly focus on RDPGs because of the wide variety of embedding techniques which have been suggested in the literature under this model, but that so far have not been compared for link prediction purposes.

RDPGs have not been formally extended to a dynamic setting, but models for multiple heterogeneous graphs on the same node set have recently been proposed in the literature. Early examples discuss methods for clustering and community detection with multiple graphs (Tang, Lu, and Dhillon, 2009; Shiga and Mamitsuka, 2012; Dong et al., 2014). More recently, the focus has been on testing for differences in brain connectivity networks (Arroyo-Relión et al., 2019; Ginestet et al., 2017; Durante and Dunson, 2018; Kim and Levina, 2019). Levin et al. (2017) propose an *omnibus* embedding in which the different graphs are jointly embedded into a common latent space, providing distinct representations for each graph and for each node. Wang et al. (2019) propose the *multiple random eigen graph* (MREG) model, where a common set of d-dimensional latent features $\mathbf{X}$ is shared between the graphs, and the inner product between the latent positions is weighted

differently across the networks, obtaining $\mathbb{E}(\mathbf{A}_t) = \mathbf{X}\mathbf{R}_t\mathbf{X}^\top$, where $\mathbf{R}_t$ is a $d \times d$ diagonal matrix. Nielsen and Witten (2018) propose the *multiple random dot product graph* (multi-RDPG), which more naturally extends the RDPG to the multi-graph setting. Their formulation is similar to the MREG of Wang et al. (2019), but $\mathbf{X}$ is modelled as an orthogonal matrix, and $\mathbf{R}_t$ is constrained to be positive semi-definite. The model is further extended in *common subspace independent edge* (COSIE) graphs (Arroyo-Relión et al., 2020), in which $\mathbf{R}_t$ does not need to be a diagonal matrix. In this chapter, omnibus embeddings (Levin et al., 2017) and COSIE graphs (Arroyo-Relión et al., 2020) will be analysed for link prediction purposes, and compared to standard spectral embedding techniques.

Many other models other than RDPGs have been proposed in the literature for link prediction. Traditionally, the temporal link prediction task is tackled using tensor decompositions (Dunlavy, Kolda, and Acar, 2011). Dynamic models have also been proposed in the literature of Poisson matrix factorisation and recommender systems (Charlin et al., 2015; Hosseini et al., 2020), and extended to Bayesian tensor decompositions (Schein et al., 2015). In general, including time has been shown to significantly improve the predictive performance in a variety of model settings, for example stochastic blockmodels (Ishiguro et al., 2010; Xu and Hero III, 2014; Xing, Fu, and Song, 2010). Durante, Dunson, and Vogelstein (2017) also propose a Bayesian nonparametric framework, interpreting the network snapshots as realisations from a common network-valued random variable, modelled through an unknown probability mass function, characterised using a mixture of low-rank factorisations. Unfortunately, the Bayesian nonparametric scheme cannot be scaled to large networks. Latent feature models for dynamic networks, which extend static latent position models (Hoff, Raftery, and Handcock, 2002), have also been extensively discussed in the literature (Sarkar and Moore, 2006; Krivitsky and Handcock, 2014; Durante and Dunson, 2014; Sewell and Chen, 2015; Durante and Dunson, 2016). Latent features could be obtained via matrix factorisation, considering constant and time-varying components within the decomposition (Deng et al., 2016; Yu, Aggarwal, and Wang, 2017; Yu et al., 2017). Usually, a Markov assumption is placed on the evolution of the latent positions (Zhu et al., 2016; Chen and Li, 2018; Sewell and Chen, 2017). Gao, Denoyer, and Gallinari (2011) propose to combine node features into a temporal link prediction framework based on matrix factorisation. Nonparametric (Sarkar, Chakrabarti, and Jordan, 2014; Durante and Dunson, 2014) and deep learning (Li et al., 2014) approaches have also been considered. Other examples of commonly used embedding methods are the widely used DeepWalk (Perozzi, Al-Rfou, and Skiena, 2014), node2vec (Grover and Leskovec, 2016), LINE (Tang et al., 2015), and GraphSAGE (Hamilton, Ying, and Leskovec, 2017) algorithms, popular in the machine learning literature.

The chapter is organised as follows: Section 5.1 introduces the generalised random dot product graph (GRDPG) and adjacency spectral embeddings, the main statistical tools used in this chapter. Methods for link prediction based on random dot product graphs are discussed in Section 7.1.

Section 7.2 presents techniques to improve the predictive performance of the RDPG models, based on time series models information. Results and applications on simulated and real world networks are finally discussed in Section 7.3.

7.1. Dynamic link prediction in random dot product graphs

Given a time series of network adjacency matrices $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_T$, the objective is to correctly predict $\mathbf{A}_{T+1}$. The most common approach in the literature (Sharan and Neville, 2008; Scheinerman and Tucker, 2010; Dunlavy, Kolda, and Acar, 2011) is to analyse a collapsed version $\tilde{\mathbf{A}}$ of the adjacency matrices:

$$\tilde{\mathbf{A}} = \sum_{t=1}^{T} \psi_{T-t+1} \mathbf{A}_t, \tag{7.1}$$

where $\psi_1, \ldots, \psi_T \in \mathbb{R}$ is a sequence of weights. Scheinerman and Tucker (2010) propose to consider an average adjacency matrix, setting $\psi_t = 1/T \; \forall \; t = 1, \ldots, T$, which provides the maximum likelihood estimate of $\mathbb{E}(\mathbf{A}_t)$ if $\mathbf{A}_1, \ldots, \mathbf{A}_T$ are sampled independently from the same Bernoulli($\mathbf{X}\mathbf{X}^\top$) distribution. The main limitation of such a model is that it is assumed that the graphs do not display any temporal evolution. Furthermore, if (7.1) is used, it is assumed that all the possible edges of the adjacency matrix follow the same dynamics. Obtaining the ASE $\hat{\mathbf{X}} = [\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_n]$ of $\tilde{\mathbf{A}}$ leads to an estimate the scores:

$$\mathbf{S} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top. \tag{7.2}$$

For simplicity, the inner product (7.2) is not weighted by the matrix $\mathbf{I}(d_+, d_-)$, implicitly assuming $d_+ = d$ and $d_- = 0$ for the remainder of this chapter. The estimation approaches in (7.1) and (7.2) will be used as baselines for comparison with alternative methods for temporal link prediction techniques using RDPGs, which will be proposed and discussed in the remainder of this section. The proposed methods will be classified in the following two categories: averages of inner products of embeddings (AIP), and inner products of an average of embeddings (IPA).

First, it is possible to consider the individual ASE for each adjacency matrix $\mathbf{A}_t$, and calculate an AIP score:

$$\mathbf{S}_{\text{AIP}} = \frac{1}{T} \sum_{t=1}^{T} \hat{\mathbf{X}}_t \hat{\mathbf{X}}_t^\top. \tag{7.3}$$

A second option is to obtain an averaged embedding $\bar{\mathbf{X}}$ from $\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \ldots, \hat{\mathbf{X}}_T$, and use this for predicting the link probabilities. This procedure is slightly more complex than (7.3), since the embeddings are invariant to orthogonal transformations: given an orthogonal matrix $\boldsymbol{\Omega}_t \in \mathbb{R}^{d \times d}$, $\mathbb{E}(\mathbf{A}_t) = \mathbf{X}_t\mathbf{X}_t^\top = (\mathbf{X}_t\boldsymbol{\Omega}_t)(\mathbf{X}_t\boldsymbol{\Omega}_t)^\top$. Therefore, the embeddings $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_T$ only provide estimates of the corresponding latent positions up to an orthogonal transformation, which could

vary for different values of t. Consequently, the embeddings must be suitably *aligned* or *registered*, before a direct comparison can be carried out. Discussion of a technique to align the embeddings is deferred to Appendix B. Assuming that an averaged embedding $\bar{\mathbf{X}}$ is obtained, the matrix of IPA scores for prediction of $\mathbf{A}_{T+1}$ is:

$$\mathbf{S}_{\text{IPA}} = \bar{\mathbf{X}}\bar{\mathbf{X}}^\top. \tag{7.4}$$

Similar scoring mechanisms can be derived for the techniques of multiple graph inference described in the introduction to this chapter, such as the omnibus embedding (Levin et al., 2017), based on the the omnibus matrix:

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_1 & \frac{\mathbf{A}_1+\mathbf{A}_2}{2} & \cdots & \frac{\mathbf{A}_1+\mathbf{A}_T}{2} \\ \frac{\mathbf{A}_2+\mathbf{A}_1}{2} & \mathbf{A}_2 & \cdots & \frac{\mathbf{A}_2+\mathbf{A}_T}{2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathbf{A}_T+\mathbf{A}_1}{2} & \frac{\mathbf{A}_T+\mathbf{A}_2}{2} & \cdots & \mathbf{A}_T \end{bmatrix}. \tag{7.5}$$

The ASE $\hat{\mathbf{X}}$ of $\tilde{\mathbf{A}}$ gives T latent positions for each node. The individual estimates $\hat{\mathbf{X}}_t = [\hat{\boldsymbol{x}}_{1t}, \ldots, \hat{\boldsymbol{x}}_{nt}]$ of the latent positions for the t-th adjacency matrix are represented by the sub-matrix formed by the estimates between the $((t-1)n+1)$-th and tn-th row of $\hat{\mathbf{X}}$. Then, from the time series $\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \ldots, \hat{\mathbf{X}}_T$ of omnibus embeddings, a matrix of scores can be obtained using either AIP (7.3) or IPA (7.4). In this case, the individual embeddings are directly comparable and an alignment step is not required. On the other hand, the omnibus embedding cannot easily be updated when new graphs $\mathbf{A}_{T+1}, \mathbf{A}_{T+2}, \ldots$ become available, since $\tilde{\mathbf{A}}$ and the embedding must be recomputed for each new snapshot. The idea of an omnibus embedding can be also easily extended to directed and bipartite graphs, constructing the matrix $\tilde{\mathbf{A}}$ analogously and then calculating the DASE.

Embeddings generated using the more parsimonious COSIE model (Arroyo-Relión et al., 2020) are also considered. In COSIE networks, the latent positions are assumed to be common across the T snapshots of the graph, but the link probabilities are scaled by a time-varying matrix $\mathbf{R}_t \in \mathbb{R}^{d\times d}$: $\mathbb{E}(\mathbf{A}_t) = \mathbf{X}\mathbf{R}_t\mathbf{X}^\top$. The common latent positions $\mathbf{X}$ and the time series of weighting matrices $\mathbf{R}_1, \ldots, \mathbf{R}_T$ can be estimated via multiple adjacency spectral embedding (MASE, Arroyo-Relión et al., 2020).

Definition 5 (Multiple adjacency spectral embedding (Arroyo-Relión et al., 2020))**.** *Given a sequence of network adjacency matrices $\mathbf{A}_1, \ldots, \mathbf{A}_T$, and an integer $d \in \{1, \ldots, n\}$, obtain the individual ASEs $\hat{\mathbf{X}}_t = \hat{\mathbf{\Gamma}}_t|\hat{\mathbf{\Lambda}}_t|^{1/2} \in \mathbb{R}^{n\times d}$. Then, construct the $n \times Td$ matrix $\tilde{\mathbf{\Gamma}} = [\hat{\mathbf{\Gamma}}_1, \ldots, \hat{\mathbf{\Gamma}}_T] \in \mathbb{R}^{n\times Td}$, and consider its singular value decomposition $\tilde{\mathbf{\Gamma}} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^\top + \tilde{\mathbf{U}}_\perp\tilde{\mathbf{S}}_\perp\tilde{\mathbf{V}}_\perp^\top$, where $\tilde{\mathbf{S}} \in \mathbb{R}_+^{d\times d}$ is a diagonal matrix containing the top d singular values in decreasing order, $\tilde{\mathbf{U}} \in \mathbb{R}^{n\times d}$ and $\tilde{\mathbf{V}} \in \mathbb{R}^{Td\times d}$ contain the corresponding left and right singular vectors, and the matrices $\tilde{\mathbf{S}}_\perp$, $\tilde{\mathbf{U}}_\perp$,*

and $\tilde{\mathbf{V}}_\perp$ *contain the remaining singular values and vectors. The* d*-dimensional multiple adjacency embedding of* $\mathbf{A}_1, \ldots, \mathbf{A}_T$ *in* $\mathbb{R}^d$ *is given by* $\hat{\mathbf{X}} = \tilde{\mathbf{U}}$*, which provides an estimate of* $\mathbf{X}$*, and the sequence* $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_T$*, where*

$$\hat{\mathbf{R}}_t = \hat{\mathbf{X}}^\top \mathbf{A}_t \hat{\mathbf{X}}.$$

For prediction, the matrix of AIP scores could be obtained from the time series of estimated link probabilities $\hat{\mathbf{X}}\hat{\mathbf{R}}_1\hat{\mathbf{X}}^\top, \ldots, \hat{\mathbf{X}}\hat{\mathbf{R}}_T\hat{\mathbf{X}}^\top$:

$$\mathbf{S}_{\text{AIP}} = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{X}}\hat{\mathbf{R}}_t\hat{\mathbf{X}}^\top. \tag{7.6}$$

Alternatively, an averaged $\bar{\mathbf{R}}$ could be equivalently obtained from the time series of estimates $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_T$. Combining $\bar{\mathbf{R}}$ with the estimate of the latent positions $\hat{\mathbf{X}}$ yields the IPA scores:

$$\mathbf{S}_{\text{IPA}} = \hat{\mathbf{X}}\bar{\mathbf{R}}\hat{\mathbf{X}}^\top. \tag{7.7}$$

The COSIE model can also be extended to directed and bipartite graphs assuming $\mathbb{E}(\mathbf{A}_t) = \mathbf{X}\mathbf{R}_t\mathbf{Y}^\top$. This construction leads to estimates $\hat{\mathbf{R}}_t = \tilde{\mathbf{U}}_{\text{X}}^\top \mathbf{A}_t \tilde{\mathbf{U}}_{\text{Y}}$, where $\tilde{\mathbf{U}}_{\text{X}}$ and $\tilde{\mathbf{U}}_{\text{Y}}$ are estimates of $\mathbf{X}$ and $\mathbf{Y}$ obtained from MASE on the DASE embeddings $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_T$ and $\hat{\mathbf{Y}}_1, \ldots, \hat{\mathbf{Y}}_T$, based on two matrices $\tilde{\mathbf{\Gamma}}$ constructed from the left and right singular vectors (*cf.* Definition 5).

In summary, several link prediction schemes based on random dot product graph models have been proposed, corresponding to three different types of spectral embedding: scores based on individual embeddings, (7.3) and (7.4); omnibus scores, (7.3) and (7.4), based on the matrix representation in (7.5); COSIE scores, (7.6) and (7.7). Two scores, denoted AIP and IPA, are calculated for each embedding type. All methods will be compared to the popular collapsed adjacency matrix method in (7.2).

The methods described in this section are all based on truncated eigendecompositions of some form of the adjacency matrix. Graph adjacency matrices are binary and normally highly sparse. In this setting, efficient algorithms based on the power method calculate the required decomposition of a matrix $\mathbf{A}$ in $\mathcal{O}\{\text{nnz}(\mathbf{A})P(1/\varepsilon)\}$ (Ghashami, Liberty, and Phillips, 2016), where $\text{nnz}(\cdot)$ denotes the number of non-zero entries of the matrix, $P(\cdot)$ is a polynomial function and $\varepsilon \in (0, 1)$ is an approximation error parameter. This represents a significant improvement over the cubic cost $\mathcal{O}(n^3)$ for the full eigendecomposition of a dense matrix, and allows the methodologies in this section to be scalable to large networks.

7.2. Improving prediction using time series models

The collapsed matrix used in (7.1) assumes that the underlying dynamics of each link are the same across the entire graph. This assumption is particularly limiting in real world applications, where

different behaviours might be associated with different nodes or links. Instead, edge specific matrix parameters $\mathbf{\Psi}_1, \ldots, \mathbf{\Psi}_T \in \mathbb{R}^{n \times n}$ might be able to more reliably capture the behaviour of each edge. A modification of the collapsed matrix $\tilde{\mathbf{A}}$ in (7.1) is therefore proposed:

$$\tilde{\mathbf{A}} = \sum_{t=1}^{T} \left(\mathbf{\Psi}_{T-t+1} \odot \mathbf{A}_t \right), \tag{7.8}$$

where $\mathbf{\Psi}_1, \ldots, \mathbf{\Psi}_T \in \mathbb{R}^{n \times n}$ is a sequence of weighting matrices, and $\odot$ is the Hadamard element-wise product.

The idea could be easily extended to all the other prediction settings proposed in Section 7.1, replacing the average link probability or average embedding with an autoregressive combination. For example, from the sequence of standard embeddings $\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \ldots, \hat{\mathbf{X}}_T$, it could be possible to obtain the scores as:

$$\mathbf{S}_{\text{PIP}} = \sum_{t=1}^{T} \left(\mathbf{\Psi}_{T-t+1} \odot \hat{\mathbf{X}}_t \hat{\mathbf{X}}_t^\top \right). \tag{7.9}$$

Alternatively, it could be possible to use a similar technique to obtain an estimate $\tilde{\mathbf{X}}_{T+1} = \sum_{t=1}^{T} (\mathbf{\Psi}_{T-t+1} \odot \hat{\mathbf{X}}_t)$ of the embedding $\mathbf{X}_{T+1}$, where in this case $\mathbf{\Psi}_t \in \mathbb{R}^{n \times d}$. The scores are then obtained as:

$$\mathbf{S}_{\text{IPP}} = \tilde{\mathbf{X}}_{T+1} \tilde{\mathbf{X}}_{T+1}^\top. \tag{7.10}$$

Similarly, for COSIE, the scores could be based on a linear combination of $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_T$ to estimate $\mathbf{R}_{T+1}$. The equations (7.9) and (7.10) define two different methodologies to extend multiple RDPG inference models to a dynamic setting. Following the same nomenclature used in Section 7.1, the scores based on time series models have been respectively denoted as PIP for the predicted inner product (7.9), and IPP for the inner product of the prediction (7.10). The PIP scores have a particularly interesting property: the node-specific embeddings are combined with time series models at the edge levels, fusing information at two different network resolutions.

Note that, for COSIE scores, $\mathbf{S}_{\text{AIP}} = \mathbf{S}_{\text{IPA}}$ from (7.6) and (7.7), but $\mathbf{S}_{\text{PIP}} \neq \mathbf{S}_{\text{IPP}}$. This is because for $\mathbf{S}_{\text{PIP}}$, the scores are obtained directly from a prediction based on the estimated link probabilities, whereas for $\mathbf{S}_{\text{IPA}}$, the prediction is based on an estimate of $\mathbf{R}_{T+1}$ from $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_T$.

For estimation of the weighting matrices $\mathbf{\Psi}_1, \ldots, \mathbf{\Psi}_T$, the time series of link probabilities or node embeddings will be modelled independently. Seasonal autoregressive integrated (SARI) processes represent a flexible modelling assumption. A univariate time series $Z_1, \ldots, Z_T \in \mathbb{R}$, is a SARI$(p, b)(P, B)_s$ with period s if the series is a causal autoregressive process defined by the equation

$$\phi(L)\Phi(L^s)(1-L)^b(1-L^s)^B Z_t = \varepsilon_t,\ \varepsilon_t \overset{iid}{\sim} \mathbb{N}(0, \sigma^2), \tag{7.11}$$

where $\phi(v) = 1 - \phi_1 v - \cdots - \phi_p v^p$, $\Phi(v) = 1 - \Phi_1 v - \cdots - \Phi_P v^P$, and L is the *lag* operator $L^k Z_t = Z_{t-k}$ (Brockwell and Davis, 1987). For example, consider a process SARI$(1, 0)(0, 1)_s$.

The model equation (7.11) becomes $\tilde{Z}_t = \phi_1 \tilde{Z}_{t-1} + \varepsilon_t$ with $\tilde{Z}_t = Z_t - Z_{t-s}$. The process is causal if and only if $\phi(v) \neq 0$ and $\Phi(v) \neq 0$ for $|v| \leq 1$. The parameters of the process p, b, P, B and s are all *integer-valued*, and can be interpreted as follows: s is the seasonal periodicity; b corresponds to the differencing required to make the process stationary in mean, variance, and autocorrelations, and B refers to the seasonal differencing; p is to the number of autoregressive terms appearing in the equation, and similarly P refers to the number of seasonal autoregressive terms. More details about SARI models and their interpretation are given in Brockwell and Davis (1987).

The value of s usually depends on the application domain. For example, in computer networks with daily network snapshots, it is reasonable to assume $s = 7$, which represents a periodicity of one week. The remaining parameters, p, b, P and B, could be estimated using information criteria. For small values of T, the corrected Akaike information criterion (AICc) is preferred. The corresponding coefficients of the polynomials $\phi(v)$ and $\Phi(v)$, and the variance σ^2, can be estimated via maximum likelihood (Brockwell and Davis, 1987). For a discussion on automatic selection of the parameters in SARI models, see Hyndman and Khandakar (2008).

For prediction of future values Z_{t+1} conditional on $Z_1, \ldots, Z_t$, the general forecasting equation is obtained from (7.11), setting ε_{t+1} to its expected value $\mathbb{E}(\varepsilon_{t+1}) = 0$, and obtaining an estimate $\hat{Z}_{t+1}$ solving from the known terms of the equation. Analogously, k-steps ahead forecasts for Z_{t+k} can also be obtained.

In this chapter, the univariate time series $Z_1, \ldots, Z_T$ modelled using SARI are of four different types:

i. Time series of estimated link probabilities: for example $\hat{\boldsymbol{x}}_{i1}^\top \hat{\boldsymbol{x}}_{j1}, \ldots, \hat{\boldsymbol{x}}_{iT}^\top \hat{\boldsymbol{x}}_{jT}$ for the standard ASE, representing the sequence of estimated scores for the edge (i, j). This type of time series is used to obtain PIP scores, see (7.9);

ii. Time series of node embeddings on a given embedding dimension: for example $\hat{x}_{ir1}, \ldots, \hat{x}_{irT}$, obtained considering only the i-th node embedding on the r-th dimension from $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_T$. Such values are used for the IPP scores, see (7.10);

iii. Time series of entries of the COSIE matrix, used to obtain IPP scores, see (7.10). For example: $\hat{R}_{kh1}, \ldots, \hat{R}_{khT}$, corresponding to the entries (k, h) in $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_T$;

iv. Times series of binary entries of the network adjacency matrices: for example $A_{ij1}, \ldots, A_{ijT}$ for the edge (i, j), used for (7.8).

The coefficients for each entry of the weighting matrices $\boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_T$ are obtained by matching (7.8), (7.9) and (7.10) with the model equation (7.11).

In this chapter, the binary time series $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_T$ is also modelled using indipendent SARI models for estimation of (7.8). Such a modelling approach might not be entirely technically suited for binary-valued time series, but this choice has relevant practical advantages: most programming languages have packages for automatic estimation of the parameters in SARI models, whereas the choice of initial values and estimation of the parameters in most generalised process for binary

time series (MacDonald and Zucchini, 1997; Kauppi and Saikkonen, 2008; Benjamin, Rigby, and Stasinopoulos, 2003) is notoriously difficult, which is not desirable when the estimation task should be performed automatically and in parallel over a large set of time series.

The time series modelling extensions presented in this section are computationally expensive, since up to n^2 or nd time series model are fitted, each with complexity $\mathcal{O}(Tk)$, where k is the number of models compared for the purpose of model selection using AICc. In the case of the COSIE score (7.7), the cost for prediction of future values of $\mathbf{R}_t$ is only $\mathcal{O}(d^2Tk)$. Therefore, with the exception of the IPP COSIE scores, the methodologies proposed in this section do not scale well to large networks, unlike the techniques proposed in Section 7.1.

7.3. Results

The proposed methods were tested on synthetic data and on real world dynamic networks from two different application domains: transportation systems and cyber-security. For all examples, the parameter d was selected using the profile likelihood elbow criterion of Zhu and Ghodsi (2006).

7.3.1. Simulated data

The performance of the rival link prediction techniques discussed in this chapter is initially compared on simulated data from a *seasonal* stochastic blockmodel, following up from the discussion in Chapters 5 and 6. To simulate a stochastic blockmodel, the within-community probability matrix $\mathbf{B} = \{B_{ij}\} \in [0,1]^{K\times K}$ was generated, where $B_{ij} \sim \text{Beta}(1.2, 1.2)$ is the probability of a link between two nodes in communities i and j, and K is the number of communities. The matrix has full rank with probability 1, hence $K = d$. In the simulation, $T = 100$ graph snapshots with $n = 100$ and $K = 5$ were generated. The community allocations were chosen to be time dependent, assuming a seasonality of one week. For each node, community allocations $z_{i,u}, u = 0, \ldots, s-1$, with $s = 7$, were sampled uniformly from $\{1, \ldots, K\}$. Then, the adjacency matrices were obtained as:

$$\mathbb{P}(A_{ijt} = 1) = B_{z_{i,t \bmod s}, z_{j,t \bmod s}},\ t = 1, \ldots, T.$$

Therefore, the link probabilities change over time, with a periodicity of 7 days. The models presented in Section 7.1 were fitted using the first $T' = 80$ snapshots of the graph, with the objective of predicting the remaining $T - T'$ adjacency matrices. The methods that are initially compared are: ASE (7.2) of the averaged adjacency matrix (7.1); AIP score (7.3) and IPA score (7.4); AIP and IPA scores calculated from the omnibus embedding (7.5); AIP and IPA scores (7.6)–(7.7) calculated from the COSIE embedding.

The link prediction problem can be framed as a binary classification task. Hence, the performances of the methods presented in this chapter are evaluated using the area under the receiver operating characteristic (ROC) curve, commonly known as AUC scores. The results are plotted in

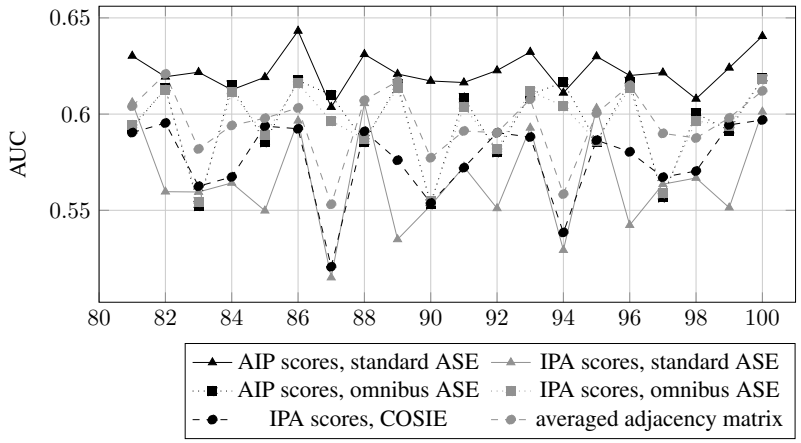


Figure 7.1.: *Results of the link prediction procedure on the simulated seasonal SBM.*

Figure 7.1. The best performance is achieved by the AIP score based on the standard ASE, which outperforms the commonly used method of the collapsed adjacency matrix (7.1).

It is anticipated that the predictions should be improved upon by the PIP and IPP extensions presented in Section 7.2, since the simulated network has clear dynamics which are not explicitly taken into account using the techniques from Section 7.1. In particular, four methods are discussed:

i. ASE of the *collapsed adjacency matrix* in (7.8), with weights obtained from independent seasonal SARI$(p, b)(P, B)_7$ processes fitted on each binary sequence $A_{ij1}, A_{ij2}, \ldots, A_{ijT'}$ such that at least one $A_{ijt} = 1$,

ii. PIP scores (7.9) based on the edge time series $\hat{\boldsymbol{x}}_{i1}^\top \hat{\boldsymbol{x}}_{j1}, \hat{\boldsymbol{x}}_{i2}^\top \hat{\boldsymbol{x}}_{j2}, \ldots, \hat{\boldsymbol{x}}_{iT'}^\top \hat{\boldsymbol{x}}_{jT'}$ obtained from the individual ASEs on each $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{T'}$,

iii. IPP scores (7.10) based on prediction of the subsequent embeddings $\tilde{\mathbf{X}}_{T'+1}, \tilde{\mathbf{X}}_{T'+2}, \ldots$ from the time series of *aligned*[1] individual ASEs $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_{T'}$,

iv. IPP scores based on prediction of COSIE correction matrices $\tilde{\mathbf{R}}_{T'+1}, \tilde{\mathbf{R}}_{T'+2}, \ldots$ from the time series $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_{T'}$, where independent models are fitted to the $d \times d$ time series corresponding to each entry.

The time series models were fit using the function *auto_arima* in the statistical *python* library *pmdarima*, using the corrected AIC criterion to estimate the number of parameters. The results are presented in Figure 7.2.

The AIP method (7.3) which had the best performance in Figure 7.1, is significantly improved by the PIP score (7.9) using time series modelling, and it is overall the only method that reaches values of the AUC well above 0.8. Remarkably, the performance of the extended collapsed adjacency matrix method in (7.8) outperforms the results based on most of the other methods, despite the

[1]The indefinite Procrustes alignment, described in Appendix B, has been implemented in *python* using *rpy2* and the R codebase developed by Joshua Agterberg, available online at `https://github.com/jagterberg/indefinite_procrustes` and `https://jagterberg.github.io/assets/procrustes_simulation.html` (last accessed on 17th September, 2020).

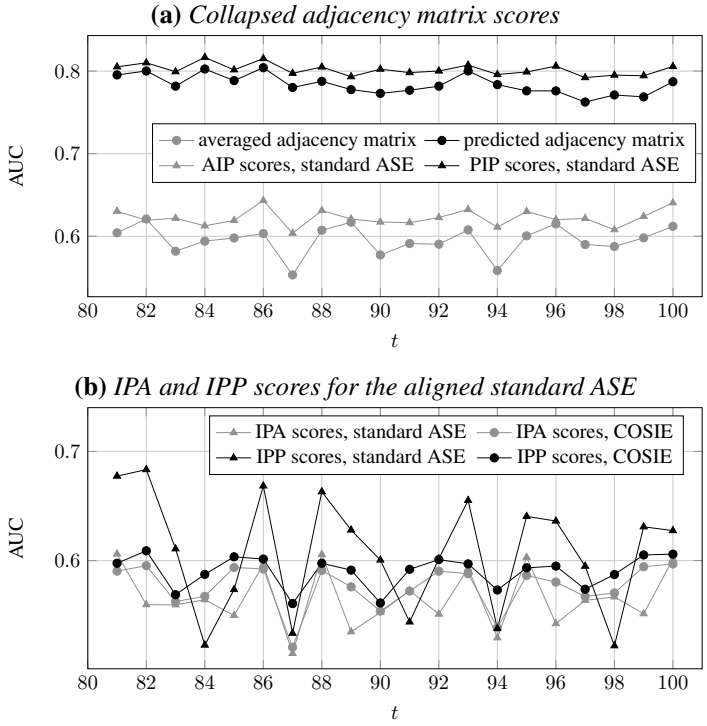


Figure 7.2.: *Comparison between four of the link prediction models in Figure 7.1, and their extensions using the methods in Section 7.2, on the synthetic SBM data.*

issues related to the modelling of binary time series pointed out in Section 7.2. On the other hand, the improvements obtained using the COSIE-based scores and the IPP score (7.10) seem to be less significant compared to the two other methods. This aspect will also be confirmed on real data examples in the next section. In general, it is clear from the plots in Figure 7.2 that adding temporal dynamics to the network via time series modelling is beneficial for link prediction purposes. In particular, including edge specific information from the time series of estimated link probabilities, or from the binary time series of links, has significantly improved link prediction.

7.3.2. Santander bikes

The Santander Cycles bike sharing network was introduced in Section 5.6.2. In this example, data from 7 March, 2018 to 19 March, 2019 were used, for a total of $T = 378$ days. Each bike sharing station is considered as a node, and an undirected edge (i, j, t) is drawn if at least one journey between the stations i and j is completed on day t. The total number of docking stations active in London during the observation period is $n = 840$. The daily graphs are fairly dense, with an average edge density of approximately 10% across the T networks. The first $T' = 250$ graphs are used as training set. Initially, the methods compared are four of the techniques used to produce Figure 7.1.

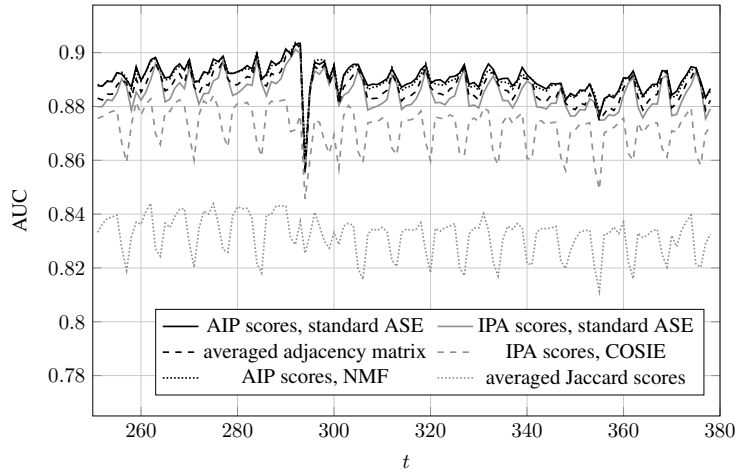


Figure 7.3.: *Results of the link prediction procedure on the Santander Cycles network.*

For the Santander Cycles network, the results are reported in Figure 7.3. To provide further comparison, the methods proposed in Section 7.1 are also compared to other classic methods used for temporal link prediction, namely Adamic-Adar, Jaccard and preferential attachment coefficients (used, for example, in Güneş, Gündüz-Öğüdücü, and Çataltepe, 2016), and non-negative matrix factorisation (NMF) (see, for example, Chen et al., 2017; Yu, Aggarwal, and Wang, 2017). In order to demonstrate that the proposed methodology could be readily extended to any embedding technique, the non-negative matrix factorisation scores were calculated from each of the adjacency matrices $\mathbf{A}_1, \ldots, \mathbf{A}_T$, and prediction scores were obtained using the AIP methodology, akin to (7.3). Similarly, for the simple Adamic-Adar, Jaccard and preferential attachment coefficients, the scores were calculated for each of the T' adjacency matrices, and then averaged to obtain predicted scores. Of the three aforementioned coefficients, only the best performing score, the Jaccard method, was reported in Figure 7.3.

In Figure 7.3, the performance of the classification procedure drops around day 294. This corresponds to Christmas day, which has a different behaviour compared to non-festive days. It is also interesting to point out that COSIE methods tends to perform better on weekdays than weekends, whereas the other methods more accurately predict the links on weekends compared to weekdays. The results of the link prediction procedure in Figure 7.3 suggest that the data might not have a long term trend, but only a seasonal component, since the performance does not significantly decrease over time, and the parameters obtained using a training set of size $T' = 250$ seem to reliably predict the structure of the adjacency matrix even at $T = 378$.

Overall, this example seems to confirm that the method of AIP scores (7.3) based on standard ASE has the best performance for link prediction purposes when time dynamics are not included. Such scores also have a similar performance to the AIP scores obtained using the excellent non-negative matrix factorisation link prediction technique, widely used in the literature (Chen et al.,

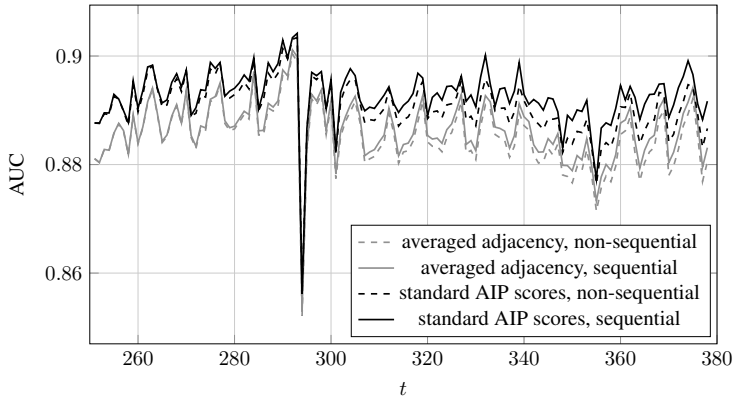


Figure 7.4.: *Results for the AIP scores* (7.3) *and averaged adjacency matrix scores for the Santander Cycles network, with and without sequential updates.*

2017). All of the proposed methodologies perform better than the Adamic-Adar, Jaccard and preferential attachment coefficients, which seem to be largely outperformed by spectral embedding methods.

As discussed in the introduction to this chapter, it should be noted that the methodologies of AIP and IPA scores proposed in Section 7.1, and the corresponding PIP and IPP extensions in Section 7.2, could be applied to *any* sequence of individual embeddings, not necessarily obtained using ASE, but also with other embedding methods for static networks, for example NMF. Since one of the main objectives of this chapter is comparing different embedding methods based on RDPGs, the focus in subsequent examples will be only on RDPG-based techniques.

So far, the embeddings learned using the initial T' snapshots have been used to predict *all* the remaining $T - T'$ adjacency matrices. In practical applications, it would be necessary to sequentially update the scores when new snapshots become available over time, improving the predictive performance. This is demonstrated in Figure 7.4, where the AIP scores and average adjacency matrix scores from Figure 7.3 are compared with their sequential counterparts, obtained when the model is updated using each new observation $\mathbf{A}_t$ in the test set. Clearly, updating the scores sequentially is beneficial, especially towards the end of the test set, whereas the difference between the methodologies is negligible in the initial snapshots of the test set. Both methods seem to reliably predict even k-steps ahead network snapshots, since the non-sequential curves in Figure 7.4 are fairly close to their sequential counterparts. The method of AIP scores based on standard ASE outperforms the scores based on the averaged adjacency matrix, commonly used in the literature, including sequential settings. Furthermore, the *non-sequential* AIP scores (7.3), also outperform the *sequential* averaged adjacency matrix. This result is quite remarkable, since the former only use the initial T' snapshots of the network for training, whereas the latter is sequentially updated with the snapshots in the test set.

The performance of the classifiers can be improved using some of the time series model in

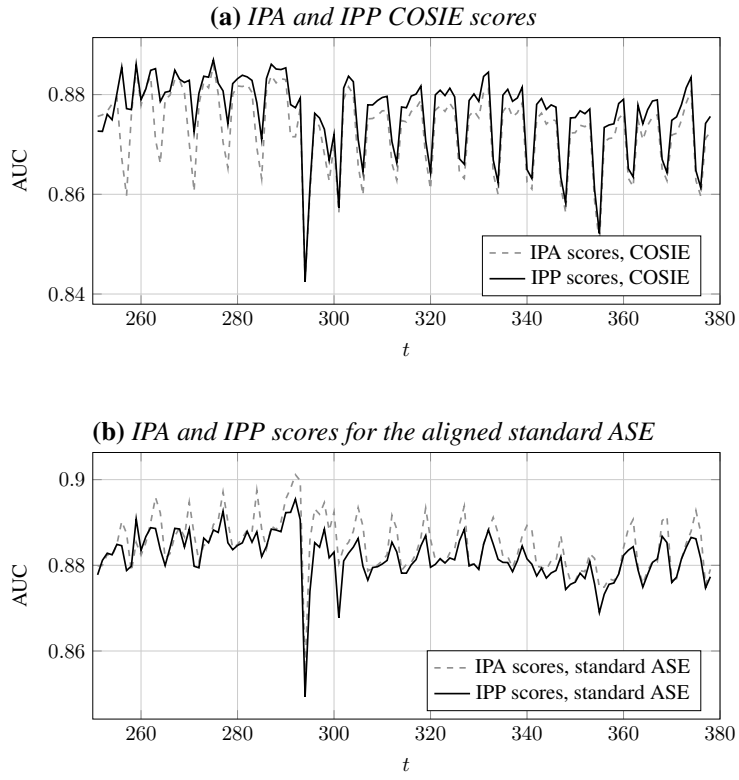


Figure 7.5.: *Comparison between two of the link prediction models in Figure 7.3, and their extensions using the methods in Section 7.2, on the Santander Cycles network.*

Section 7.2. Figure 7.5a show the results obtained from the prediction of subsequent COSIE correction matrices. The predictive performance is slightly improved by the extended time series models. Again, it is empirically confirmed that adding temporal dynamics is beneficial for the performance of random dot product graph based classifiers.

On the other hand, predicting the subsequent adjacency spectral embeddings from the time series of aligned embeddings $\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \ldots, \hat{\mathbf{X}}_{T'}$ does not seem to improve the predictive performance. The results are presented in Figure 7.5b, and confirm the findings in Figure 7.2b, where the improvements on the simulated network were less significant compared to other methods. In this case, the time series models are not able to capture the dynamics of the aligned embeddings, and the predictive performance does not improve in AUC.

The limited improvements in the results seem to suggest that the network does not have a strong dynamic component. The tradeoff between performance and the computational effort required to fit multiple independent time series simultaneously, would suggest use of the AIP scores (7.3) based on standard ASE in practical applications.

7.3.3. LANL 2017 data

In this section, a bipartite graph obtained from the LANL 2017 dateset (see Section 2.1.2, or Turcotte, Kent, and Hash, 2018) is used. From the host event logs, 90 daily user-authentication bipartite graphs have been constructed, writing $A_{ijt} = 1$ if the user i initiates a connection authenticating to computer j, on day t. This graph is known as the *User – Destination* graph (*cf.* Section 2.1.2). A total of $|V_1| = 12{,}222$ users, $|V_2| = 5{,}047$ hosts, and $85{,}020$ pairs (i, j) are observed, corresponding to approximately 0.137% of all possible links. The first $T' = 56$ matrices are used as training set. Note that it is computationally difficult to calculate $|V_1| \times |V_2|$ scores for each adjacency matrix, and storing such large dense matrices in memory is also not efficient. Therefore, an estimate of the AUC can be obtained by subsampling the negative class at random from the zeroes in the test set adjacency matrices. Two subsampling techniques are used to construct the negative class for prediction of $\mathbf{A}_t$:

(1) the negative class is constructed by sampling pairs (i, j) such that $A_{ijt} = 0$,

(2) the negative class contains randomly selected pairs (i, j) such that $A_{ijt} = 0$, *and* all pairs (i, j) such that $A_{ijt} = 0$ and $A_{ijt'} = 1$ for at least 1 value of $t' \in \{1, \ldots, T\}$.

For simplicity, the two subsampling techniques are denoted with the numbers *(1)* and *(2)* in Figure 7.6. The former method clearly provides an estimate of the ROC curve for the entire matrix, since the scores are sampled at random from the distribution of all scores. On the other hand, the latter method includes in the negative class more elements that tend to have associated high scores, represented by the pairs (i, j, t) such that $A_{ijt} = 0$ and $A_{ijt'} = 1$ for at least 1 value of $t' \in \{1, \ldots, T\}$, giving an unbalanced sampling procedure and therefore a biased estimate of the ROC curve. Clearly, higher AUC scores should be obtained using the first subsampling procedure.

Interestingly, in Figure 7.6a, the COSIE-based AIP scores seem to have the best predictive performance across the different methods. In particular, COSIE scores tend to largely outperform the other methods during weekdays, whereas the performance during weekends seems almost equivalent, and sometimes inferior, to the AIP scores (7.3) based on the standard ASE. On the other hand, in Figure 7.6b, COSIE scores have the worst performance among the methods, except a spike on day 62. In Figure 7.6b, AIP scores (7.3) based on the standard ASE once again give the best predictive performance. The results of Figure 7.6b are of particular interest since these allow for a comparison of the classifiers on a more challenging negative class compared to Figure 7.6a. Therefore, it is reasonable to conclude that the AIP scores (7.3) emerge again as the most suitable method for link prediction based on random dot product graphs.

For this example, it is also demonstrated how the proposed methods can be extended for streaming applications. For example, the method of combining inner products of individual embeddings is particularly suitable for implementation in a streaming fashion, since the inner products are easily updated on the run when new snapshots of the graph are observed. A demonstration using the method of *fixed forgetting factors* (Gama, Sebastião, and Rodrigues, 2013) is given here. The

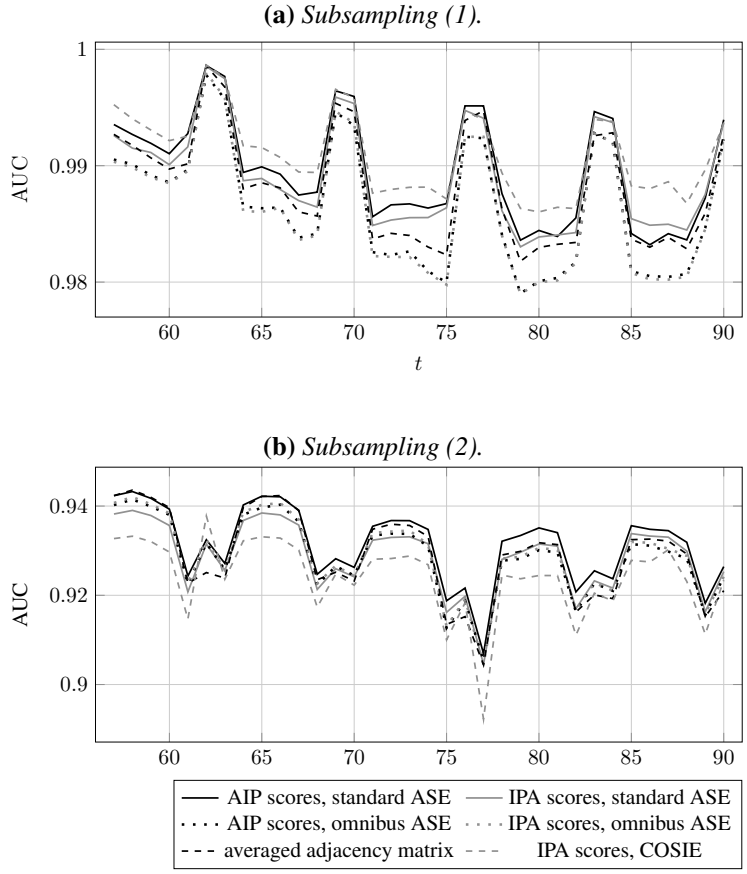


Figure 7.6.: *Results of the link prediction procedure on the LANL network. In both cases, AUCs are calculated from* $\approx 150{,}000$ *links per graph.*

matrix of scores $\mathbf{S}_t$ for prediction of $\mathbf{A}_{t+1}$ is updated in streaming as follows:

$$\begin{aligned} w_t &= \lambda w_{t-1} + 1, \\ \mathbf{S}_t &= \left(1 - \frac{1}{w_t}\right) \mathbf{S}_{t-1} + \frac{1}{w_t} \hat{\mathbf{X}}_t \hat{\mathbf{X}}_t^\top, \end{aligned} \tag{7.12}$$

starting from $w_1 = 1$ and $\mathbf{S}_1 = \hat{\mathbf{X}}_1 \hat{\mathbf{X}}_1^\top$. The forgetting factor $\lambda \in [0, 1]$ is usually chosen to be close to 1 (Gama, Sebastião, and Rodrigues, 2013). Note that $\lambda = 1$ corresponds to the sequential AIP scores in (7.3), calculated as in Figure 7.4, whereas smaller values of λ give more weight to recent observations. In particular, $\lambda = 0$ only gives weight to the most recent snapshot. Schemes similar to (7.12) could be also implemented to update the collapsed adjacency matrix (7.1), obtaining the scores from its ASE at each point in time, or to update the matrix $\mathbf{R}_t$ in (7.1) for the COSIE embeddings, or the rotated embeddings $\hat{\mathbf{X}}_t$. The forgetting factor approach might be interpreted as a simplification of the time series scheme proposed in Section 7.2, where the same

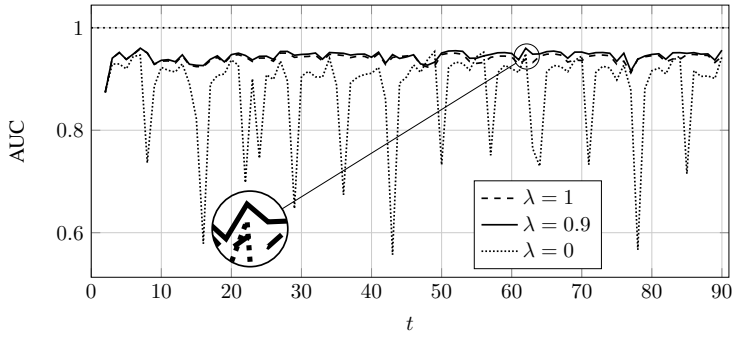


Figure 7.7.: *Results for the AIP scores* (7.3) *on the LANL network with streaming updates. AUCs are calculated from* $\approx 150{,}000$ *links per graph, using subsampling (2).*

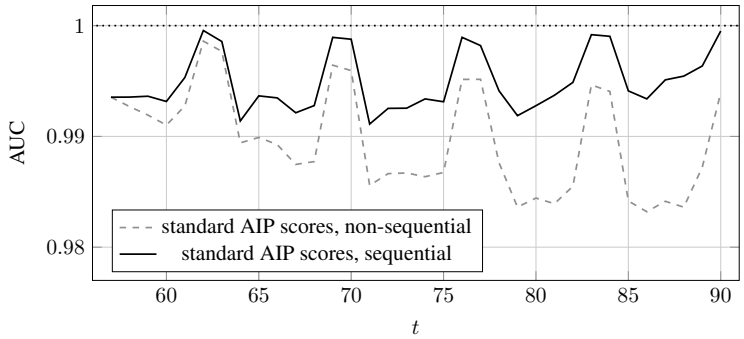


Figure 7.8.: *Results for the AIP scores* (7.3) *for the LANL network, with and without sequential updates. AUCs are calculated from* $\approx 150{,}000$ *links per graph, using subsampling (1).*

weight is given to each edge.

The results for the entire observation period for three different values of λ are plotted in Figure 7.7. The best performance is achieved with the forgetting factor approach with $\lambda = 0.9$. The performance for $\lambda = 0$ clearly drops around the weekends, since the network has a seasonal component which is not accounted for in the prediction. The difference between the curves for $\lambda = 1$ and $\lambda = 0.9$ suggests that the graph has temporal dynamics which is captured by the forgetting factor approach, which down-weights past observations in favour of more recent snapshots of the graph. This impression is confirmed by Figure 7.8, which shows the sequential and non-sequential AIP scores based on the standard ASE, akin to Figure 7.4. The predictive performance slightly deteriorates for snapshots that are further away in time, whereas 1-step ahead predictions based on the sequential scores consistently give better results.

The performance of the different methods can be again improved using the time series models in Section 7.2. Figure 7.9 shows the results obtained using the two different subsampling schemes for the collapsed adjacency matrix scores and the scores based on standard and COSIE embeddings. From Figures 7.9a, 7.9c and 7.9e, it is evident that the extensions do not improve the performance of the classifier when the subsampling scheme *(1)* is used. On the other hand, Figures 7.9b, 7.9d and

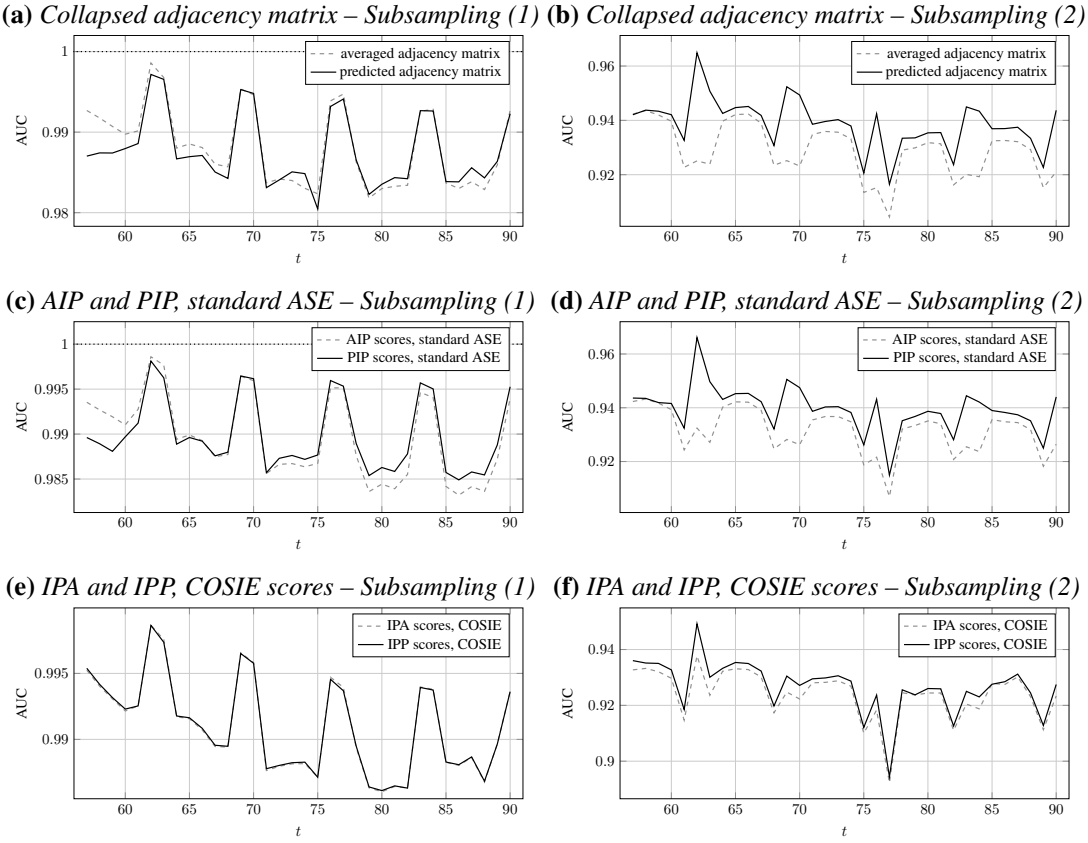


Figure 7.9.: *Comparison between three of the link prediction models in Figure 7.6, and their extensions using the methods in Section 7.2, on the LANL network. AUCs are calculated from* $\approx 150{,}000$ *links per graph.*

7.9f show that relevant improvements (especially on day 62) are obtained when the subsampling method *(2)* is used, which represents a more difficult classification task. Again, the results confirm that the performance of RDPGs for link prediction can be enhanced by time series models.

7.3.4. Imperial College network flow data

The methodologies proposed in Section 7.1 are also tested on a large computer network, demonstrating that the spectral embedding techniques are scalable to graphs of large size. A directed graph with $n = 95{,}220$ has been constructed using the network flow data collected at Imperial College London (Section 2.2), limiting the connections to internal hosts only. The data have been collected over $T = 47$ days, between 1st January and 16th February, 2020. An edge between a client i and a server j is drawn on day t if the two hosts have connected at least once during the corresponding day. The number of edges ranges from a minimum of $344{,}565$, observed on 1st January, to a maximum of $912{,}984$ on 6th February.

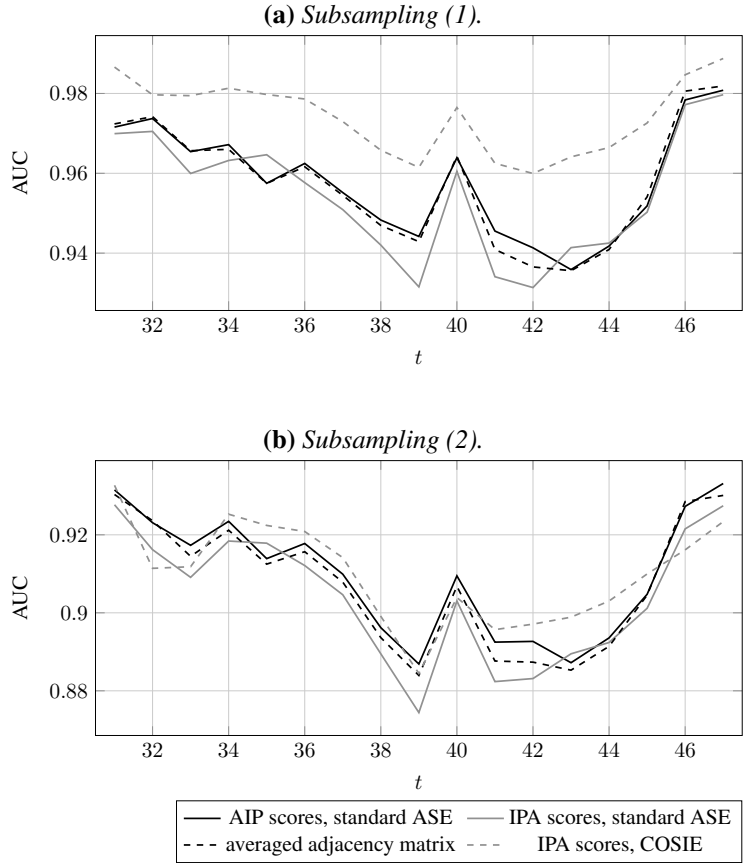


Figure 7.10.: *Results of the link prediction procedure on the Imperial College network. In both cases, AUCs are calculated from* 6.7 *million links per graph.*

The results are plotted in Figure 7.10. In this case, the best performance is achieved by the COSIE scores, similarly to the LANL network in Figure 7.6a. The traditional methodology of the averaged adjacency matrix is also outperformed by the AIP scores, which supports the results obtained in all the previous real data examples. As pointed out in Section 7.2, fitting the time series models on the sequence $\hat{\mathbf{R}}_1, \ldots, \hat{\mathbf{R}}_{T'}$ of COSIE weighting matrices is inexpensive, and it is therefore possible to scale the time series extension of the IPA scores to a fairly large network. The results of this procedure are plotted in Figure 7.11, which shows again that the time series extension is beneficial for link prediction purposes.

7.4. Conclusion and discussion

In this chapter, simple link prediction techniques based on random dot product graphs have been presented, discussed and compared. In particular, link prediction methods based on sequences of

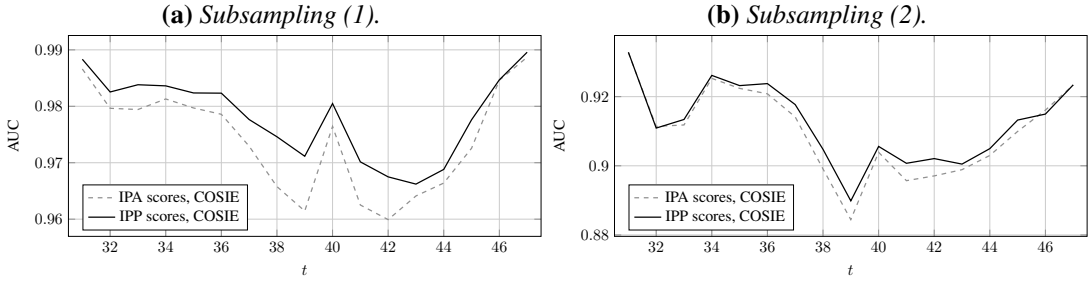


Figure 7.11.: *IPA and IPP scores using COSIE embeddings on the Imperial College network. In both cases, AUCs are calculated from* 6.7 *million links per graph.*

embeddings, COSIE models, and omnibus embeddings have been considered. Applications on simulated and real world data have shown that one of the most common approaches used in the literature, the decomposition of a collapsed adjacency matrix $\tilde{\mathbf{A}}$, is usually outperformed by other methods for multiple graph inference in random dot product graphs.

Estimating link probabilities with the average of the inner products from sequences of individual embeddings of different snapshots of the graph has given the best performance in terms of AUC scores across multiple datasets. This result is particularly appealing for practical applications: calculating the individual ASEs is computationally inexpensive using algorithms for large sparse matrices, and the method seems particularly suitable for implementation in a streaming fashion, since the average inner product could be easily updated on the run when new snapshots of the graph are observed, as demonstrated in Section 7.3.3.

The methods discussed in the chapter have then been further extended to include temporal dynamics, using time series models. The extensions have shown improvements over standard random dot product graph based link prediction techniques, especially when the graph exhibits a strong dynamic component. The techniques presented in this chapter could also be readily extended to *any* embedding method for static graphs, following the framework for calculating AIP and IPA scores presented in Section 7.1, and the PIP and IPP extensions in Section 7.2. Overall, this chapter provides valuable guidelines for practitioners for using random dot product graphs as tools for link prediction in networks, providing insights into the predictive capability of such statistical network models.

7.4.1. Limitations and future work

So far, it has been assumed that nodes do not have associated covariates and are therefore exchangeable. When nodal covariates are observed, the random dot product graph probability function (5.4) must be adapted to include that information. As discussed in Section 7.1, one of the main advantages of spectral-based methods is the computational efficiency of estimation via ASE or

LSE, which has complexity approximately proportional to the number of observed links $\mathrm{nnz}(\mathbf{A})$ for a sparse adjacency matrix $\mathbf{A}$. Unfortunately, if covariates are included in (5.4), estimates of the latent positions are not as straightforward to obtain, and likelihood-based methodologies should be considered instead. This might represent a computational burden in large graphs, since evaluating the likelihood $L(\mathbf{X}) \propto \prod_{ij}(\boldsymbol{x}_i^\top \boldsymbol{x}_j)^{A_{ij}}(1-\boldsymbol{x}_i^\top \boldsymbol{x}_j)^{1-A_{ij}}$ in RDPG models has complexity $\mathcal{O}(|V|^2)$, which makes this approach not scalable to large $|V|$. To address this issue, efficient likelihood approximations in latent space models have been proposed in the literature, for example the case-control approximation of Raftery et al. (2012). In contrast, random dot product graph models with covariates have been successfully used in the literature, for fairly small graphs. For example, Marchette and Priebe (2008) use a random dot product graph model with covariates for missing link prediction. Mele et al. (2019) propose a spectral approach for estimation of the stochastic blockmodel with covariates (Choi, Wolfe, and Airoldi, 2012; Roy, Atchadé, and Michailidis, 2019). Mu et al. (2020) discusses two model-based spectral algorithms for clustering vertices in stochastic blockmodel graphs with nodal covariates.

Motivated by application to the LANL 2017 computer network, the next chapter will address the issues of scalability and inclusion of covariates. To this purpose, a Poisson matrix factorisation model with nodal covariates will be proposed, such that the complexity of evaluating the likelihood exactly is only $\mathcal{O}(\mathrm{nnz}(\mathbf{A}))$. The model will be shown to outperform the RDPG for link prediction purposes, even when covariates are not available.

8

Graph link prediction using Poisson matrix factorisation

In Chapter 7, it was demonstrated that the random dot product graph could be efficiently used for temporal link prediction purposes. In the literature, Poisson matrix factorisation (PMF) (Canny, 2004; Dunson and Herring, 2005; Cemgil, 2009; Gopalan, Hofman, and Blei, 2015) also emerged as a suitable model in the link prediction framework for dyadic count data. The methodological contribution of this chapter is to present extensions of the PMF model, suitably adapted to scenarios which are commonly encountered in cyber-security applications. Importantly, a framework for including categorical covariates within the PMF model is introduced, which also allows for modelling of new nodes appearing within a network. Traditionally, Poisson matrix factorisation methods are used on partially observed adjacency matrices of natural numbers representing, for example, ratings of movies provided by different users. In cyber-security applications, the matrix is fully observed, and the counts associated with network edges are complicated by repeated observations, polling at regular intervals, and the intrinsic burstiness of the events (Heard, Rubin-Delanchy, and Lawson, 2014). This is demonstrated in Figure 8.1, which displays the frequency distribution of the number of connections on each edge in the first 56 days of observation on the LANL 2017 network, described in Section 2.1.2. Therefore, each edge is usually represented by a binary indicator expressing whether at least one connection between the corresponding nodes was observed. Consequently, the standard PMF model, where counts associated with links are assumed to follow a Poisson distribution with unbounded support, cannot be applied directly. Instead, indicator functions are applied in this chapter, leading to an extension for PMF on binary adjacency matrices.

Initially, the focus will be on a model for a single realisation of a training set adjacency matrix, used to predict a single future realisation in a testing period. This framework will then be extended, presenting a PMF model that incorporates seasonal dynamics. *New links*, corresponding to previ-

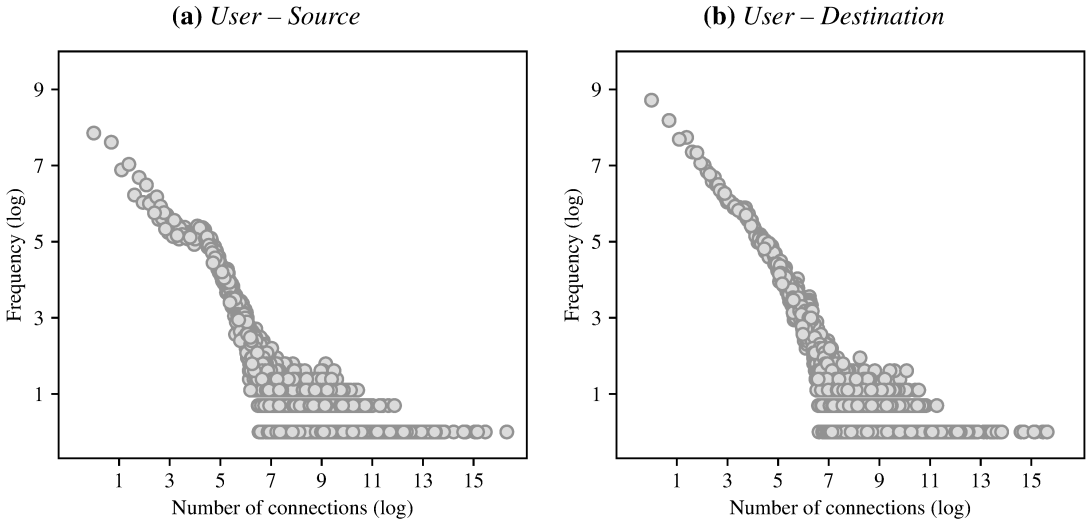


Figure 8.1.: *Frequency distribution on a log-log scale of the number of connections on each edge in the LANL 2017 network, in the first* 56 *days of observation.*

ously unobserved relationships, will be given particular attention, as many attack behaviours such as lateral movement (Neil et al., 2013), phishing, and data retrieval, can create new edges between network entities (Metelli and Heard, 2019).

The methodologies in this chapter have been developed to provide insight into authentication data extracted from the LANL 2017 dataset (Turcotte, Kent, and Hash, 2018). As described in Section 2.1.2, two bipartite graphs $\mathbb{G} = \{(V_1, V_2), E\}$ are generated from the data: first, the network users and the computers from which they authenticate, denoted *User – Source*; second, the same users and the computers or servers they are connecting to, denoted *User – Destination*. The two graphs are first analysed separately, before a joint model is explored. For each user and computer, a list of categorical covariates are available: 6 covariates for the users, and 3 for the computers. In total, there are $K = 1{,}064$ factor levels available for the user credentials and $H = 735$ factor levels for the computers. Section 7.4.1 discussed the scalability issues in the estimation of random dot product graphs with nodal covariates. One objective of this chapter is to present a scalable methodology for incorporating such covariates within Poisson matrix factorisation. The extended PMF model proposed in this chapter is also a special case of latent position model (Hoff, Raftery, and Handcock, 2002), assuming a particular form of kernel function (5.1), $\kappa(\boldsymbol{x}, \boldsymbol{y}) = 1 - \exp(-\boldsymbol{x}^\top \boldsymbol{y})$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d_+$, with nodal covariates.

The rest of the chapter is organised as follows: Section 8.1 formally introduces Poisson matrix factorisation for network link prediction, and Section 8.2 discusses the proposed PMF model for binary matrices and labelled nodes. A seasonal extension is described in Section 8.3. Finally, results of the analysis are presented in Section 8.4.

8.1. Background on Poisson matrix factorisation

Let $\mathbf{N} \in \mathbb{N}^{|V_1| \times |V_2|}$ be a matrix of non-negative integers N_{ij}. For recommender system applications, N_{ij} could represent information about how a user i rated an item j, or a count of the times they have clicked on or purchased the item. The cyber-security application has a major difference: in recommender systems, if user i never interacted with the item j, then N_{ij} is considered as a *missing observation*, implying that the that the number of observations is a possibly very small fraction of $|V_1||V_2|$. On the other hand, in cyber-security, the absence of a link from i to j is itself an observation, namely $N_{ij} = 0$, and $|V_1||V_2|$ data points are observed. Such a difference has practical consequences, particularly regarding scalability in cyber-security of models borrowed from the recommender systems literature.

The hierarchical Poisson factorisation model (Gopalan, Hofman, and Blei, 2015) specifies a distribution for N_{ij} using a Poisson link function with rate given by the inner product between user-specific latent features $\boldsymbol{x}_i \in \mathbb{R}_+^d$ and host-specific latent features $\boldsymbol{y}_j \in \mathbb{R}_+^d$, for a positive integer $d \geq 1$:

$$N_{ij} \sim \text{Poisson}(\boldsymbol{x}_i^\top \boldsymbol{y}_j) = \text{Poisson}\left(\sum_{r=1}^{d} x_{ir} y_{jr}\right). \tag{8.1}$$

If two latent features are *close* in the latent space, the corresponding nodes are expected to exhibit similar connectivity patterns. The specification of the model is completed with gamma hierarchical priors:

$$\begin{aligned} x_{ir} &\sim \text{Gamma}(a^{(x)}, \zeta_i^{(x)}),\ i = 1, \ldots, |V_1|,\ r = 1, \ldots, d, \\ y_{jr} &\sim \text{Gamma}(a^{(y)}, \zeta_j^{(y)}),\ j = 1, \ldots, |V_2|,\ r = 1, \ldots, d. \\ \zeta_i^{(x)} &\sim \text{Gamma}(b^{(x)}, c^{(x)}),\ \zeta_j^{(y)} \sim \text{Gamma}(b^{(y)}, c^{(y)}), \end{aligned} \tag{8.2}$$

Each of the gamma distribution parameters $a^{(x)}, b^{(x)}, c^{(x)}, a^{(y)}, b^{(y)}, c^{(y)}$ is a positive real number which must be specified.

In the cyber-security context there are no missing observations in the matrix $\mathbf{N}$. Consequently, an advantage of PMF over other models for link prediction (for example, Salakhutdinov and Mnih, 2007) is that the likelihood function only depends on the observed links, meaning evaluating the likelihood exactly is $\mathcal{O}(\text{nnz}(\mathbf{N}))$, compared to $\mathcal{O}(|V_1||V_2|)$ for most statistical network models. In cyber-security, networks tend to be very large in the number of nodes, but extremely sparse: $\text{nnz}(\mathbf{N}) \ll |V_1||V_2|$. Hence, PMF appears to be a particularly appealing modelling framework for this application.

The PMF model has been used as a building block for multiple extensions. For example, Chaney, Blei, and Eliassi-Rad (2015) developed *social Poisson factorisation* to include latent *social influences* in personalised recommendations. Gopalan, Charlin, and Blei (2014) developed *collaborative*

topic Poisson factorisation, which adds a document topic offset to the standard PMF model to provide content-based recommendations and thereby tackle the challenge of recommending new items, referred to in the literature as cold starts. These ideas of combining collaborative filtering and content-based filtering are further developed in Zhang and Wang (2015), Singh and Gordon (2008) and Silva, Langseth, and Ramampiaro (2017), where social influences are added as constraints in the latter.

In general, most PMF-based methods presented in this section model *binary* adjacency matrices using the Poisson link function for convenience. This approach is computationally advantageous, but implies an incorrect model for the range: the entries A_{ij} of the adjacency matrix are binary, whereas the Poisson distribution has support over the natural numbers.

Dynamical extensions to PMF have also been studied. Charlin et al. (2015) use Gaussian random walk updates on the latent features to dynamically correct the rates of the Poisson distributions. Schein et al. (2015) and Schein et al. (2016) propose a temporal version of PMF using the two main tensor factorisation algorithms: canonical polyadic and Tucker decompositions. Hosseini et al. (2020) combine the PMF model with the Poisson process to produce dynamic recommendations. Dynamic network models have also been widely studied outside the domain of matrix factorisation techniques (for a survey, see Kim et al., 2018). For example, Sewell and Chen (2015) extend latent position models to a temporal setting. The dynamic models described above could handle generic network dynamics, but seasonality, a special case of temporal structure, has not been explicitly accounted for in the PMF literature. This chapter further aims to fill this gap and propose a viable seasonal PMF model.

8.2. PMF with labelled nodes and binary adjacency matrices

Suppose that there are K covariates associated with each user and H covariates for each host. Let the value of the covariate k for user i be denoted as u_{ik}. Similarly, let the value of the covariate h for host j be v_{jh}. In cyber-security applications, most available information types are categorical, indicating memberships of known groupings or clusters of nodes. For the remaining of this chapter, the covariates will be assumed to be binary indicators representing categorical variables. This type of encoding of categorical variables is commonly known in the statistical literature as *dummy variable encoding*, whereas the term *one-hot encoding* is used in the machine learning community. Several approaches for including nodal covariates in recommender systems using non-probabilistic matrix factorisation methods, such as the SVD, have been discussed in the literature (for some examples, see Nguyen and Zhu, 2013; Fithian and Mazumder, 2018; Dai et al., 2019). Regression methods for network data with covariates are also studied in Hoff (2005). In Section 8.1, it has been remarked that, for binary adjacency matrices, the standard PMF model for count data cannot be applied directly, since the observations are binary, whereas the Poisson distribution has support

on the natural numbers. To model binary links, it is assumed here that the count N_{ij} is a latent random variable, and the binary indicator $A_{ij} = \mathbb{1}_{\mathbb{N}_+}(N_{ij})$ is a censored Poisson draw with a corresponding Bernoulli distribution. This type of link has been referred to in the literature as the Bernoulli-Poisson (BerPo) link (Acharya et al., 2015; Zhou, 2015). The full extended model is

$$\begin{aligned} A_{ij}|N_{ij} &= \mathbb{1}_{\mathbb{N}_+}(N_{ij}), \\ N_{ij}|\boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{\Phi} &\sim \text{Poisson}\left(\boldsymbol{x}_i^\top \boldsymbol{y}_j + \mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i \boldsymbol{v}_j^\top)\mathbf{1}_H\right) \\ &= \text{Poisson}\left(\sum_{r=1}^{d} x_{ir} y_{jr} + \sum_{k=1}^{K}\sum_{h=1}^{H} \phi_{kh} u_{ik} v_{jh}\right), \end{aligned} \tag{8.3}$$

where $\mathbf{1}_n$ is a vector of n ones, $\odot$ is the Hadamard element-wise product, and $\boldsymbol{u}_i$, $\boldsymbol{v}_j$ are K and H-dimensional binary vectors of covariates. The d-dimensional latent features $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$, appear in the traditional PMF model given in (8.1), and $\boldsymbol{\Phi} = \{\phi_{kh}\} \in \mathbb{R}_+^{K\times H}$ is a matrix of interaction terms for each combination of the covariates. Under model (8.3),

$$\mathbb{P}(A_{ij} = 1) = 1 - \exp\left(-\sum_{r=1}^{d} x_{ir} y_{jr} - \sum_{k=1}^{K}\sum_{h=1}^{H} \phi_{kh} u_{ik} v_{jh}\right). \tag{8.4}$$

To provide intuition for these extra terms, assume for the cyber-security application that a binary covariate for job title `manager` is provided for the users, and that a binary covariate for the location `research lab` for the hosts. If user i is a manager and host j is located in a research lab, then ϕ_{kh} expresses a correction to the rate $\boldsymbol{x}_i^\top \boldsymbol{y}_j$ for a manager connecting to a machine in a research lab. The covariate term is inspired by the bilinear mixed-effects models for network data in Hoff (2005).

The same hierarchical priors (8.2) are used for $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ and the following prior distribution completes the specification of the model:

$$\phi_{kh}|\zeta^{(\phi)} \sim \text{Gamma}(a^{(\phi)}, \zeta^{(\phi)}),\ k = 1, \ldots, K,\ h = 1, \ldots, H, \quad \zeta^{(\phi)} \sim \text{Gamma}(b^{(\phi)}, c^{(\phi)}).$$

Note that this model provides a natural way for handling what the literature commonly refers to as cold starts, where new users or hosts appear in the network. Provided that covariate-level information is known about new entities, then the estimates for $\boldsymbol{\Phi}$ can be used to make predictions about links where $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ for new user i or new host j could be initialised from the prior or some other global statistic based on other users and hosts. Another significant advantage of the model proposed in (8.3) is that the likelihood is $\mathcal{O}(\text{nnz}(\mathbf{A}))$, analogously to the standard PMF model in (8.2), implying that the model scales well to large sparse networks.

Proposition 8.1. The likelihood of the PMF model with covariates in (8.3) is $\mathcal{O}(\mathrm{nnz}(\mathbf{A})$:

$$\log L(\mathbf{A}) = \sum_{i,j:A_{ij}>0} \log\left(e^{\psi_{ij}} - 1\right) - \left(\sum_{i=1}^{|V_1|} \boldsymbol{x}_i\right)^\top \left(\sum_{j=1}^{|V_2|} \boldsymbol{y}_j\right) - \sum_{k,h}^{K,H} \phi_{kh}\tilde{u}_k\tilde{v}_h,$$

where $\psi_{ij} = \boldsymbol{x}_i^\top \boldsymbol{y}_j + \mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i\boldsymbol{v}_j^\top)\mathbf{1}_H$, $\tilde{u}_k = \sum_{i=1}^{|V_1|} u_{ik}$, and $\tilde{v}_h = \sum_{j=1}^{|V_2|} v_{jh}$.

Proof. Since $\mathbb{P}(A_{ij} = 1) = 1 - e^{-\psi_{ij}}$ from (8.4), the likelihood becomes:

$$\begin{aligned}
\log L(\mathbf{A}) &= \sum_{i,j} \left\{ A_{ij} \log\left(1 - e^{-\psi_{ij}}\right) - (1 - A_{ij})\psi_{ij} \right\} \\
&= \sum_{i,j} \left\{ A_{ij} \left[\log\left(1 - e^{-\psi_{ij}}\right) + \psi_{ij} \right] - \psi_{ij} \right\} \\
&= \sum_{i,j:A_{ij}>0} \log\left\{ e^{\psi_{ij}} \left(1 - e^{-\psi_{ij}}\right) \right\} - \sum_{i,j}^{|V_1|,|V_2|} \psi_{ij} \\
&= \sum_{i,j:A_{ij}>0} \log\left(e^{\psi_{ij}} - 1\right) - \sum_{i,j}^{|V_1|,|V_2|} \sum_{r=1}^{d} x_{ir}y_{jr} - \sum_{i,j}^{|V_1|,|V_2|} \sum_{k,h}^{K,H} \phi_{kh}u_{ik}v_{jh} \\
&= \sum_{i,j:A_{ij}>0} \log\left(e^{\psi_{ij}} - 1\right) - \left(\sum_{i=1}^{|V_1|} \boldsymbol{x}_i\right)^\top \left(\sum_{j=1}^{|V_2|} \boldsymbol{y}_j\right) - \sum_{k,h}^{K,H} \phi_{kh}\tilde{u}_k\tilde{v}_h,
\end{aligned}$$

where the last line holds by linearity of the inner product. □

8.2.1. Bayesian inference

Given an observed matrix $\mathbf{A}$, inferential interest is on the marginal posterior distributions of the parameters $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ for all the users and hosts, and the parameters $\boldsymbol{\Phi}$ for the covariates, since these govern the predictive distribution for the edges observed in the future.

A common approach for performing inference is adopted, where additional latent variables are introduced. Given the (assumed) unobserved count N_{ij}, a further set of latent counts Z_{ijl}, $l = 1, \ldots, d + KH$, are used to represent the contribution of each component l to the total latent count, such that $N_{ij} = \sum_l Z_{ijl}$. For $l \leq d$, $Z_{ijl} \sim$ Poisson$(x_{il}y_{jl})$. Otherwise, l refers to a (k, h) covariate pair, and $Z_{ijl} \sim$ Poisson(ϕ_{kh}). This construction ensures that N_{ij} has precisely the Poisson distribution specified in (8.3).

Inference using Gibbs sampling is straightforward, as the full conditionals all have closed form expressions, but sampling-based methods do not scale well with network size. Instead, a variational inference procedure is proposed in the next section. Variational inference schemes have already been commonly and successfully used in the literature for network models (Nakajima, Sugiyama, and Tomioka, 2010; Seeger and Bouchard, 2012; Salter-Townshend and Murphy, 2013; Hernández-

Lobato, Houlsby, and Ghahramani, 2014), despite the issue of introducing bias and potentially reducing estimation accuracy, in particular on the posterior variability (see, for example, Huggins et al., 2019).

Gibbs sampling also presents an additional difficulty in PMF models: the inner product $\boldsymbol{x}_i^\top \boldsymbol{y}_j$ is invariant to permutations of the latent features. In particular, $(\mathbf{Q}\boldsymbol{x}_i)^\top(\mathbf{Q}\boldsymbol{y}_j) = \boldsymbol{x}_i^\top \boldsymbol{y}_j$ for any *permutation* matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$, which makes the posterior invariant under such transformation. This implies that the posterior is highly multimodal, a well-known burden for MCMC-based inference in Bayesian factor models (Papastamoulis and Ntzoufras, 2020). Hence, parameter estimates obtained as averages from MCMC samples from the posterior could be meaningless, since the algorithm could have switched among different modes. On the other hand, variational inference is well understood to be a *"mode-seeking"* algorithm, a desirable property for this problem. A practical comparison of the estimates of the inner products for prediction purposes will be briefly illustrated in Section 8.4.2.

8.2.2. Variational inference

Variational inference (see, for example, Blei, Kucukelbir, and McAuliffe, 2017) is an optimisation based technique for approximating intractable distributions, such as the joint posterior density $p(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}, \boldsymbol{\zeta}, \mathbf{N}, \mathbf{Z}|\mathbf{A})$, with a proxy $q(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}, \boldsymbol{\zeta}, \mathbf{N}, \mathbf{Z})$ from a given distributional family $\mathcal{Q}$, and then finding the member $q^\star \in \mathcal{Q}$ that minimises the Kullback-Leibler (KL) divergence to the true posterior. Usually the KL-divergence cannot be explicitly computed, and therefore an equivalent objective, called the *evidence lower bound* (ELBO), is maximised instead:

$$\text{ELBO}(q) = \mathbb{E}^q[\log p(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}, \boldsymbol{\zeta}, \mathbf{N}, \mathbf{Z}, \mathbf{A})] - \mathbb{E}^q[\log q(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}, \boldsymbol{\zeta}, \mathbf{N}, \mathbf{Z})], \tag{8.5}$$

where the expectations are taken with respect to $q(\cdot)$. The proxy distribution $q(\cdot)$ is usually chosen to be of much simpler form than the posterior distribution, so that maximising the ELBO is tractable. As in Gopalan, Hofman, and Blei (2015) the *mean-field variational family* is used, where the latent variables in the posterior are considered to be independent and governed by their own distribution, so that:

$$\begin{aligned} q(\mathbf{X}, \mathbf{Y}, \boldsymbol{\zeta}, \mathbf{\Phi}, \mathbf{N}, \mathbf{Z}) = &\prod_{i,r} q(x_{ir}|\lambda_{ir}^{(x)}, \mu_{ir}^{(x)}) \prod_{j,r} q(y_{jr}|\lambda_{jr}^{(y)}, \mu_{jr}^{(y)}) \prod_{k,h} q(\phi_{kh}|\lambda_{kh}^{(\phi)}, \mu_{kh}^{(\phi)}) \\ &\times \prod_{i} q(\zeta_i^{(x)}|\nu_i^{(x)}, \xi_i^{(x)}) \prod_{j} q(\zeta_j^{(y)}|\nu_j^{(y)}, \xi_j^{(y)}) q(\zeta^{(\phi)}|\nu^{(\phi)}, \xi^{(\phi)}) \prod_{i,j} q(N_{ij}, \mathbf{Z}_{ij}|\theta_{ij}, \boldsymbol{\chi}_{ij}). \end{aligned} \tag{8.6}$$

The objective function (8.5) is optimised using *coordinate ascent mean field variational inference* (CAVI), whereby each density or variational factor is optimised while holding the others fixed (for details, see Bishop, 2006; Blei, Kucukelbir, and McAuliffe, 2017).

Proposition 8.2 (Bishop (2006); Blei, Kucukelbir, and McAuliffe (2017))**.** Using CAVI, the optimal form of each variational factor under a mean-field variational family is:

$$q^\star(w_j) \propto \exp\left\{\mathbb{E}^q_{-j}\left[\log p(w_j|\boldsymbol{w}_{-j}, \mathbf{A})\right]\right\}, \tag{8.7}$$

where w_j is an element of a *partition* of the full set of parameters $\boldsymbol{w}$, and the expectation is taken with respect to the variational densities that are currently held fixed for $\boldsymbol{w}_{-j}$, defined as $\boldsymbol{w}$ excluding the parameters in the subset w_j.

Proof. Denote the model parameters with $\boldsymbol{w}$. Fixing $\boldsymbol{w}_{-j}$ under the mean field assumption gives:

$$\begin{aligned}
\text{ELBO}(q) &= \mathbb{E}^q[\log p(\boldsymbol{w}, \mathbf{A})] - \mathbb{E}^q[\log q(\boldsymbol{w})] = \mathbb{E}^q_j\left\{\mathbb{E}^q_{-j}[\log p(\boldsymbol{w}, \mathbf{A})]\right\} - \mathbb{E}^q_j[\log q(w_j)] + k_1 \\
&= \mathbb{E}^q_j\left\{\mathbb{E}^q_{-j}[\log p(w_j|\boldsymbol{w}_{-j}, \mathbf{A})]\right\} - \mathbb{E}^q_j[\log q(w_j)] + k_1 + k_2 \\
&= -\text{KL}\left\{q(w_j)\middle\| \exp\left[\mathbb{E}^q_{-j}\{\log p(w_j|\boldsymbol{w}_{-j}, \mathbf{A})\}\right]\right\} + k_1 + k_2,
\end{aligned} \tag{8.8}$$

where k_1 and k_2 are constants with respect to $\boldsymbol{w}_j$, and $p(\boldsymbol{w}, \mathbf{A}) = p(w_j|\boldsymbol{w}_{-j}, \mathbf{A})p(\boldsymbol{w}_{-j}, \mathbf{A})$ was used. The KL-divergence $\text{KL}\{q\|p\} = \mathbb{E}^q\{\log q(w) - \log p(w)\}$ reaches its minimum when its arguments correspond to the *same* distribution. Therefore, from (8.8), maximising the ELBO for $\boldsymbol{w}_{-j}$ fixed in CAVI gives precisely the update in (8.7). □

Convergence of the CAVI algorithm is determined by monitoring the change in the ELBO over subsequent iterations. Since the prior distributions are chosen to be conjugate, the full conditionals in (8.7) are all available analytically. Full details are given in Appendix C.1. Also, since all the conditionals are exponential families, each $q(w_j)$ obtained from (8.7) is from the same exponential family (Blei, Kucukelbir, and McAuliffe, 2017). Hence, with the exception of $q(N_{ij}, \mathbf{Z}_{ij}|\theta_{ij}, \boldsymbol{\chi}_{ij})$, the proxy distributions in (8.6) are all gamma; for example, $q(x_{ir}|\lambda^{(x)}_{ir}, \mu^{(x)}_{ir}) = \text{Gamma}(\lambda^{(x)}_{ir}, \mu^{(x)}_{ir})$. The update equations for the parameters $\{\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\xi}, \boldsymbol{\theta}, \boldsymbol{\chi}\}$ of the variational approximation can be obtained using (8.7), which is effectively the expected parameter of the full conditional with respect to q. The full variational inference algorithm is detailed in Algorithm 8.1. Note that each update equation only depends upon the elements of the matrix where $A_{ij} > 0$, providing computational efficiency for large sparse matrices. Further details concerning the update equations for the Poisson and multinomial parameters $\boldsymbol{\theta}$ and $\boldsymbol{\chi}$ are also given in Appendix C.2.

8.2.3. Link prediction

Given the optimised values of the parameters of the variational approximation $q^\star(\cdot)$ to the posterior, a Monte Carlo posterior model estimate of $\mathbb{P}(A_{ij} = 1)$ can be obtained by averaging (8.4) over M

Algorithm 8.1: Variational inference for binary PMF with covariates.

initialise $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$ from the prior,

set $\nu_i^{(x)} = b^{(x)} + da^{(x)},\ \nu_j^{(y)} = b^{(y)} + da^{(y)},\ \nu^{(\phi)} = b^{(\phi)} + KHa^{(\phi)}$,

calculate $\tilde{u}_k = \sum_{i=1}^{|V_1|} u_{ik},\ k = 1, \ldots, K$ and $\tilde{v}_h = \sum_{j=1}^{|V_2|} v_{jh}$,

repeat

for each entry of $\mathbf{A}$ such that $A_{ij} > 0$, update the rate θ_{ij}:

$$\theta_{ij} = \sum_{r=1}^{d} \exp\left\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)})\right\} + \sum_{k=1}^{K}\sum_{h=1}^{H} u_{ik}v_{jh} \exp\left\{\Psi(\lambda_{kh}^{(\phi)}) - \log(\mu_{kh}^{(\phi)})\right\},$$

where $\Psi(\cdot)$ is the digamma function,

for each entry of $\mathbf{A}$ such that $A_{ij} > 0$, update χ_{ijl}:

$$\chi_{ijl} \propto \begin{cases} \exp\left\{\Psi(\lambda_{il}^{(x)}) - \log(\mu_{il}^{(x)}) + \Psi(\lambda_{jl}^{(y)}) - \log(\mu_{jl}^{(y)})\right\} & l \leq d, \\ u_{ik}v_{jh} \exp\left\{\Psi(\lambda_{kh}^{(\phi)}) - \log(\mu_{kh}^{(\phi)})\right\} & l > R, \end{cases}$$

where, for $l > R$, l corresponds to a covariate pair (k, h),

update the user-specific first-level parameters:

$$\lambda_{ir}^{(x)} = a^{(x)} + \sum_{j=1}^{|V_2|} \frac{A_{ij}\theta_{ij}\chi_{ijr}}{1 - e^{-\theta_{ij}}},\ \mu_{ir}^{(x)} = \frac{\nu_i^{(x)}}{\xi_i^{(x)}} + \sum_{j=1}^{|V_2|} \frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}},$$

update the host-specific first-level parameters:

$$\lambda_{jr}^{(y)} = a^{(y)} + \sum_{i=1}^{|V_1|} \frac{A_{ij}\theta_{ij}\chi_{ijr}}{1 - e^{-\theta_{ij}}},\ \mu_{jr}^{(y)} = \frac{\nu_j^{(y)}}{\xi_j^{(y)}} + \sum_{i=1}^{|V_1|} \frac{\lambda_{ir}^{(x)}}{\mu_{ir}^{(x)}},$$

update the covariate-specific first-level parameters:

$$\lambda_{kh}^{(\phi)} = a^{(\phi)} + \sum_{i,j=1}^{|V_1|,|V_2|} \frac{A_{ij}\theta_{ij}\chi_{ijl}}{1 - e^{-\theta_{ij}}},\ \mu_{kh}^{(\phi)} = \frac{\nu^{(\phi)}}{\xi^{(\phi)}} + \tilde{u}_k\tilde{v}_h,$$

update the second-level parameters:

$$\xi_i^{(x)} = c^{(x)} + \sum_{r=1}^{d} \frac{\lambda_{ir}^{(x)}}{\mu_{ir}^{(x)}},\ \xi_j^{(y)} = c^{(y)} + \sum_{r=1}^{d} \frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}},\ \xi^{(\phi)} = c^{(\phi)} + \sum_{k,h}^{K,H} \frac{\lambda_{kh}^{(\phi)}}{\mu_{kh}^{(\phi)}}.$$

until *convergence*;

samples from $q^\star(x_{ir}|\lambda_{ir}^{(x)}, \mu_{ir}^{(x)})$, $q^\star(y_{jr}|\lambda_{jr}^{(y)}, \mu_{jr}^{(y)})$, and $q^\star(\phi_{kh}|\lambda_{kh}^{(\phi)}, \mu_{kh}^{(\phi)})$:

$$\hat{\mathbb{P}}(A_{ij}=1) = 1 - \frac{1}{M}\sum_{m=1}^{M} \exp\left(-\sum_{r=1}^{d} x_{ir}^{(m)} y_{jr}^{(m)} - \sum_{h,k}^{K,H} \phi_{kh}^{(m)} u_{ik} v_{jh}\right). \tag{8.9}$$

Alternatively, a computationally fast way to approximate $\mathbb{P}(A_{ij}=1)$ uses the parameters of the estimated variational distributions:

$$\tilde{\mathbb{P}}(A_{ij}=1|\hat{x}_{ir}, \hat{y}_{jr}, \hat{\phi}_{kh}) = 1 - \exp\left(-\sum_{r=1}^{d} \hat{x}_{ir}\hat{y}_{jr} - \sum_{h,k}^{K,H} \hat{\phi}_{kh} u_{ik} v_{jh}\right), \tag{8.10}$$

where, for example, $\hat{x}_{ir} = \lambda_{ir}^{(x)}/\mu_{ir}^{(x)}$, the mean of the gamma proxy distribution. Note that (8.10) clearly gives a biased estimate, and by Jensen's inequality $\hat{\mathbb{P}}(A_{ij}=1) \leq \tilde{\mathbb{P}}(A_{ij}=1)$ in expectation, but it carries a much lower computational burden. The approximation in (8.10) has been successfully used for link prediction and network anomaly detection purposes in Turcotte et al. (2016).

8.3. Seasonal PMF

The previous sections have been concerned with making inference from a single adjacency matrix $\mathbf{A}$. Now, consider observing a sequence of adjacency matrices $\mathbf{A}_1, \ldots, \mathbf{A}_T$, representing snapshots of the same network over time. Further, suppose this time series of adjacency matrices has seasonal dynamics with some known fixed seasonal period, s; for example, s could be one day, one week or one year. To recognise time dependence, a third index t is required, such that A_{ijt} denotes the (i,j)-th element of the matrix $\mathbf{A}_t$, $t = 1, \ldots, T$.

As in Section 8.2, there are assumed to be underlying counts N_{ijt} which are treated as latent variables, and the sequence of observed adjacency matrices is obtained by $A_{ijt} = \mathbb{1}_{\mathbb{N}_+}(N_{ijt})$. To account for seasonal repetition in connectivity patterns, the model proposed for the latent counts is:

$$\begin{aligned} N_{ijt} &\sim \text{Poisson}\left(\sum_{r=1}^{d} x_{ir}\gamma_{it'r}y_{jr}\delta_{jt'r} + \sum_{k=1}^{K}\sum_{h=1}^{H} \phi_{kh}u_{ik}v_{jh}\right) \\ &= \text{Poisson}\left((\boldsymbol{x}_i \odot \boldsymbol{\gamma}_{it'})^\top(\boldsymbol{y}_j \odot \boldsymbol{\delta}_{jt'}) + \mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i\boldsymbol{v}_j^\top)\mathbf{1}_H\right), \end{aligned} \tag{8.11}$$

where, for example, $t' = 1 + (t \bmod s)$. In general, more complicated functions for t' might be required, as in Section 8.4.7.

The priors on x_{ir} and y_{jr} are those given in (8.2); these parameters represent a baseline level of activity, which is constant over time. The two additional parameters $\gamma_{it'r}$ and $\delta_{jt'r}$ represent corrections to these rates for the seasonal segment $t' \in \{1, \ldots, s\}$. Note that for some applications,

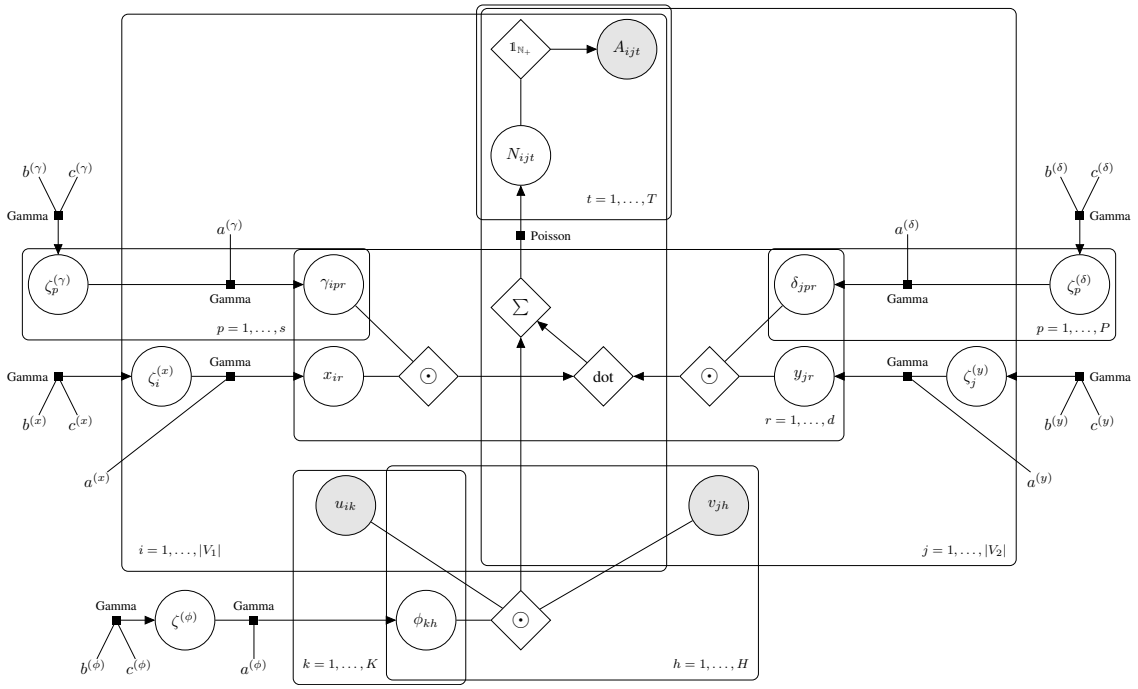


Figure 8.2.: *Graphical model representation of the seasonal extended PMF model.*

it may be anticipated that there is a seasonal adjustment to the rate for the interaction terms of the covariates, in which case temporal adjustments could be also added to $\mathbf{\Phi}$. For identifiability, it is necessary to impose constraints on the seasonal adjustments so that, for example, for all i, j, r, $\gamma_{i1r} = \delta_{j1r} = 1$.

For $t' > 1$, the following hierarchical priors are placed on $\gamma_{it'r}$ and $\delta_{jt'r}$:

$$\gamma_{it'r} \sim \text{Gamma}(a^{(\gamma)}, \zeta_{t'}^{(\gamma)}),\ \zeta_{t'}^{(\gamma)} \sim \text{Gamma}(b^{(\gamma)}, c^{(\gamma)}),$$
$$\delta_{jt'r} \sim \text{Gamma}(a^{(\delta)}, \zeta_{t'}^{(\delta)}),\ \zeta_{t'}^{(\delta)} \sim \text{Gamma}(b^{(\delta)}, c^{(\delta)}).$$

A graphical model representation of the seasonal extended PMF model is given in Figure 8.2. Inference for the seasonal model can be performed following the same principles of Section 8.2.2; full details are given in Appendix C.3. The constraint is implemented in the variational inference framework by setting the variational approximation to γ_{i1r} and δ_{j1r} to a delta function centred at 1.

8.4. Results

The extensions to the PMF model detailed in Sections 8.2 and 8.3 are now used to analyse the LANL 2017 authentication data, in particular the *User – Source* and *User – Destination* graphs described in Section 2.1.2. In order to assess the predictive performance of the models, the data are

Table 8.1.: *Summary of training and test sets for User – Destination and User – Source.*

	User – Destination		*User – Source*	
	Training set	Test set	Training set	Test set
Users	11,688	534 new	12,027	507 new
Hosts	3,801	1,246 new	15,881	1,236 new
Links	82,517	76,240	60,059	50,412
New links		11,418		12,080
Cold starts		3,401		3,014

split into a training set corresponding to the first 56 days of activity, and a test set corresponding to days 57 through 82. During the latter time period, LANL conducted a *red-team* exercise, where the security team test the robustness of the network by attempting to compromise other network hosts; labels of known compromised authentication events will be used for evaluating the model performance in anomaly detection. The parameters are estimated from the training set adjacency matrix, constructed by setting $A_{ij} = 1$ if a connection from user i to host j is observed during the training period, and $A_{ij} = 0$ otherwise. The predictive performance of the model is then evaluated on the test set adjacency matrix, constructed similarly using the connections observed in the last 25 days.

Summary statistics about the data are provided in Table 8.1, where "cold starts" refer to links originating from new users and hosts in the test data. Figure 8.3 shows binary heat map plots of the adjacency matrices obtained from the training period for each the two graphs, *User – Source* and *User – Destination*. In all analyses, variational inference is used to estimate the parameters based on the the training data, with a threshold for convergence being 10^{-5} for relative difference between two consecutive values of the ELBO (8.5). Details about the calculation of the ELBO in PMF models are given in Appendix C.5. The prior hyperparameters are set to $a^* = b^* = 1$ and $c^* = 0.1$, although the algorithm is fairly robust to the choice of these parameters. The number of latent features $d = 20$ and was chosen using the criterion of the elbow in the scree-plot of singular values.

8.4.1. Including covariates

First, results are presented for the extended PMF (EPMF) model with covariates, discussed in Section 8.2. Performance is evaluated by the ROC curve and the corresponding AUC. The AUC is used as a measure of quality of classification and will allow for the predictive power of the different models to be ranked. The link probabilities are estimated using (8.10). Due to the large computational effort of scoring all entries in the adjacency matrix for EPMF, mostly due to the calculation of $\mathbf{1}_K^\top(\mathbf{\Phi} \odot \boldsymbol{u}_i \boldsymbol{v}_j^\top)\mathbf{1}_H$, the AUC is estimated by subsampling the negative class at

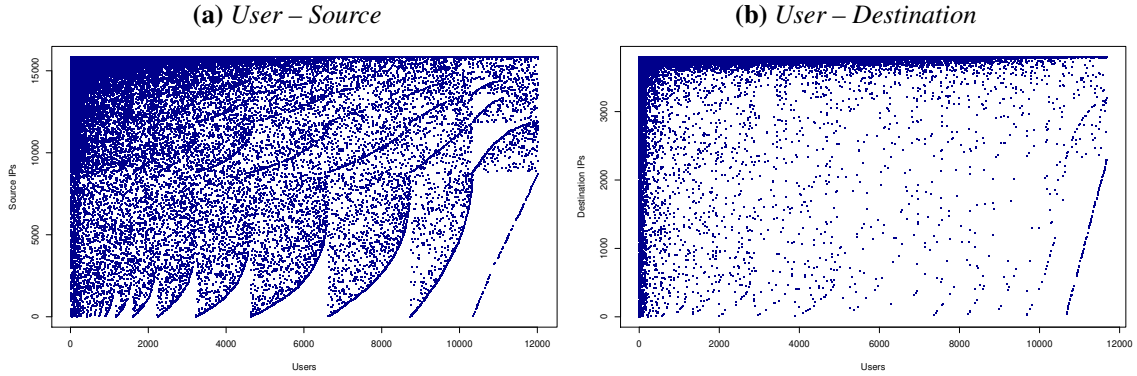


Figure 8.3.: *Training set adjacency matrices for the two graphs (spy-plot). Nodes are sorted by in-degree and out-degree.*

Table 8.2.: *AUC scores for prediction of all and new links using standard and extended PMF. Number of latent features:* $d = 20$.

	User – Destination		*User – Source*	
	PMF	EPMF	PMF	EPMF
All links	0.98479	0.98707	0.85797	0.96602
New links	0.95352	0.95474	0.89759	0.95260

random from the zeros in the adjacency matrix formed from the test data, similarly to Section 7.3.3; the sample sizes is chosen to be three times the size of the number of edges in the test set. In general, if the sample size is chosen to be at least on the same order of magnitude as $\mathrm{nnz}(\mathbf{A})$, this procedure leads to reliable estimates of the AUC, as demonstrated via simulation at the end of this subsection. The estimated AUC scores are summarised in Table 8.2. For evaluating the AUC scores for new links (edges in the test set not present in the training set), the negative class was also restricted to entries in the training adjacency matrix for which $A_{ij} = 0$.

Table 8.2 shows that the AUC for *User – Destination* does not change significantly when the extended model is used; however, for *User – Source*, the extended PMF model offers a significant improvement. The difference in the results between the two networks can be explained by the contrasting structures of the adjacency matrices. The edge density for *User – Destination* is 0.184% and for *User – Source*, 0.031%. However, despite *User – Destination* having a higher density, the links are concentrated on a small number of dominant nodes, as can be seen in Figure 8.3b. Therefore, the prediction task is relatively easy: the probability of a link is roughly approximated by a function of the degree of the node, and adding additional information is not particularly beneficial. For *User – Source*, as can be seen by Figure 8.3a, the links are more evenly distributed between the nodes, and the prediction task is more difficult. Hence, in this setting, including additional information about known groupings is crucial to improve the predictive capability of the model.

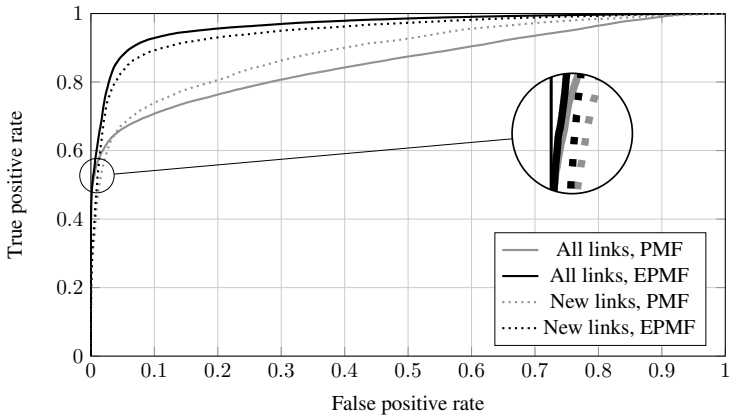


Figure 8.4.: *ROC curves for standard PMF and extended PMF on User – Source,* $d = 20$.

The ROC curves for the *User – Source* graph are shown in Figure 8.4.

Some of the covariates might be more predictive than others. To evaluate such differences, the AUC for all links on *User – Source* has been recalculated excluding each user and host covariate in turn. The difference between the AUC for EPMF fitted with all the covariates, and for EPMF with covariate k removed, could then be used to quantify the predictive power of covariate k. According to this methodology, the most relevant user covariates (using the covariate codes in the anonymised publicly released version of the dataset) are *Covariate 1*, with a loss in AUC of 0.05442 when it is removed from the model, followed by *Covariate 2*, with 0.03001. Similarly, for the hosts, the most predictive covariates appear to be *Covariate 2*, with score 0.05761, and *Covariate 0*, with 0.03116.

In order to assess the accuracy of the AUC approximation obtained from a subsample of the negative class, 500 simulations have been carried out. For standard PMF on *User – Destination*, estimates of the AUC for all links across different subsampled negative classes had standard deviation $\approx 4.3 \cdot 10^{-5}$ for a sample size of $3\mathrm{nnz}(\mathbf{A})$. If $\mathrm{nnz}(\mathbf{A})$ or $0.1\mathrm{nnz}(\mathbf{A})$ are used, the standard deviation increases to $\approx 7.6 \cdot 10^{-5}$ and $\approx 2.4 \cdot 10^{-4}$ respectively, whereas using $5\mathrm{nnz}(\mathbf{A})$ gives a value of $\approx 3.5 \cdot 10^{-5}$.

The prediction accuracy of the estimates of the link probabilities (8.9) and (8.10) can also be evaluated. Table 8.3 reports the AUC scores obtained from link probabilities estimated using (8.9) and $M = 1000$ samples from the variational approximation to the posterior. The results are similar to Table 8.2. Therefore, for the remainder of this chapter, the approximation in (8.10), computationally more efficient, is used.

8.4.2. Comparison with Gibbs sampling

As discussed in Section 8.2.1, a possible drawback of variational inference is the introduction of bias in the posterior estimates, in particular regarding the variability. Therefore, it is necessary to assess the loss in predictive performance caused by switching to the proposed variational approximation

Table 8.3.: *AUC scores for prediction of all and new links using standard and extended PMF. Link probabilities are estimated using* (8.9) *and* $M = 1000$ *samples from the variational approximation to the posterior. Number of latent features:* $d = 20$.

	User – Destination		*User – Source*	
	PMF	EPMF	PMF	EPMF
All links	0.98622	0.98707	0.85884	0.96624
New links	0.95650	0.95780	0.89103	0.94941

Table 8.4.: *AUC scores for prediction using Gibbs sampling. Number of latent features:* $d = 20$.

	User – Destination		*User – Source*	
	PMF	EPMF	PMF	EPMF
All links	0.98737	0.98835	0.87130	0.95809
New links	0.94311	0.94768	0.83979	0.92723

to the posterior. This task is not straightforward, since performing full Bayesian inference on the the standard PMF and EPMF model is computationally extremely demanding when the graph is large, because many posterior samples (usually in the order of tens of thousands) are required to confidently estimate the parameters.

Appendix C.1 gives details about the conditional distributions required to develop a Gibbs sampler. Table 8.4 presents the results obtained from 10,000 posterior samples with burnin 1,000. The link probabilities have been estimated using the unbiased score (8.9). No issues with convergence of the Markov chain were observed, and the performance seems comparable to the results obtained using variational inference, demonstrating that the loss in performance due to the variational approximation seems to be minimal.

It must be noted that variational inference converges in our application in less than 200 iterations. For example, for PMF on *User – Source IP*, the computation time is ≈ 4.5 seconds per iteration on a MacBook Pro 2017 with a 2.3GHzIntel Core i5 dual-core processor. On the other hand, Gibbs sampling, despite the lower cost of $\approx 2.6s$ per iteration, requires many more posterior samples. Hence, the computational advantages guaranteed by the variational inference procedure are particularly relevant in this context.

8.4.3. Cold starts

As discussed in Section 8.2, the extended PMF model allows for prediction of new entities or nodes in the network (cold starts). To assess performance on links in the test set involving new users or hosts, the estimates of the covariate coefficients $\hat{\phi}_{kh} = \lambda_{kh}^{(\phi)}/\mu_{kh}^{(\phi)}$ from the training period are used. The latent feature values are set equal to the mean of all users and hosts observed in the training set. For comparison against a baseline model, the regular PMF model is used where the latent features

Table 8.5.: *AUC scores for prediction of cold starts.*

	User – Destination		*User – Source*	
	PMF	EPMF	PMF	EPMF
New Users	0.96826	0.97785	0.73362	0.93148
New Hosts	0.81789	0.82715	0.79541	0.91138

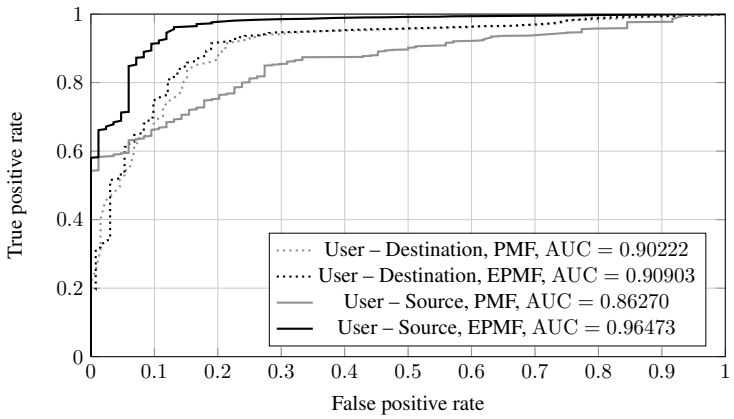


Figure 8.5.: *ROC curves for prediction of red-team events for standard PMF and extended PMF.*

are set as above; this has the effect of comparing against the global mean.

Cold starts can be divided between new *users* and new *hosts*, and the AUC scores for prediction for each case are presented in Table 8.5. To calculate the AUC, the negative class is randomly sampled from the rows and columns corresponding to the new users and hosts, respectively. Again, there are only minor performance gains for *User – Destination*, and the regular PMF model using the global average of the latent features provides surprisingly good results. As discussed above, this can be explained by the prediction task being much simpler, and well approximated by a simple degree-based model. In contrast, for the *User – Source* graph the extended PMF model shows very good predictive performance for cold starts.

8.4.4. Red-team

The motivation for this work is the detection of cyber attacks; to assess performance from an anomaly detection standpoint, the event labels from the red-team attack (see Section 2.1.2) are used as a binary classification problem. Figure 8.5 plots the ROC curves and AUC scores from the standard and extended PMF models, and improvements in detection capability are obtained using EPMF. Similarly to the previous cases, the predictive performance gain is most notable for *User – Source*.

8.4.5. Comparisons with alternative prediction methods

The results in Table 8.2 are compared in this section with alternative link prediction methods:

i. *Probabilistic matrix factorisation* (Salakhutdinov and Mnih, 2007): $A_{ij} = \boldsymbol{x}_i^\top \boldsymbol{y}_j + \varepsilon_{ij}$, $\varepsilon_{ij} \overset{iid}{\sim} \mathbb{N}(0, \sigma^2)$, with $x_{ir} \sim \mathbb{N}(0, \sigma_x^2)$ and $y_{jr} \sim \mathbb{N}(0, \sigma_y^2)$. The parameters were obtained by maximum a posteriori estimation using gradient-based optimisation techniques (Kingma and Ba, 2015). The optimisation was run independently for 25 times starting from different initialisation points, and reporting only the best results in terms of AUC.

ii. *Logistic matrix factorisation* (Johnson, 2014): $A_{ij} \sim$ Bernoulli(p_{ij}) where logit$(p_{ij}) = \boldsymbol{x}_i^\top \boldsymbol{y}_j$, with $x_{ir} \sim \mathbb{N}(0, \sigma_x^2)$ and $y_{jr} \sim \mathbb{N}(0, \sigma_y^2)$. The parameters were estimated using the same procedure described for probabilistic matrix factorisation.

iii. *Bipartite random dot product graph* (Athreya et al., 2018): $A_{ij} \sim$ Bernoulli $\left(\boldsymbol{x}_i^\top \boldsymbol{y}_j\right)$. The latent positions $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ are estimated using DASE (see Definition 4). A comparison with the truncated Katz scores (*tKatz*, Dunlavy, Kolda, and Acar, 2011) is also provided. The latent positions are estimated similarly to DASE, replacing the diagonal entries of the matrix $\mathbf{S}$ with a transformation: $f(s_{ii}) = (1 - \eta s_{ii})^{-1} - 1$, with $\eta = 10^{-4}$.

iv. *Non-negative matrix factorisation* (for example, see Chen et al., 2017): nodes are assigned features $\mathbf{W} \in \mathbb{R}_+^{|V_1| \times d}$ and $\mathbf{H} \in \mathbb{R}_+^{|V_2| \times d}$ obtained as solutions to $\|\mathbf{A} - \mathbf{W}\mathbf{H}^\top\|_{\mathrm{F}}^2$, where $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. The score associated with any link (i, j) is then obtained as $\boldsymbol{w}_i^\top \boldsymbol{h}_j$, where $\boldsymbol{w}_i^\top$ and $\boldsymbol{h}_j^\top$ are respectively the i-th and j-th row of $\mathbf{W}$ and $\mathbf{H}$.

v. *Degree-based model*, where the probability of a link is approximated as $\mathbb{P}(A_{ij} = 1) = 1 - \exp(-d_i^{\text{out}} d_j^{\text{in}})$, where d_i^{out} and d_j^{in} are the out-degree and in-degree of each node.

The results are presented in Table 8.6. Overall, when compared to the results of PMF with Ber-Po link in Table 8.2, the PMF models achieve better results compared to other popular techniques, especially for new link prediction. On *User – Source*, non-negative matrix factorisation and random dot product graph methods seem to slightly outperform standard PMF when predicting all links, but their performance for new link prediction, a more interesting and difficult task, is significantly worse than PMF. It must be remarked that some of the alternative methods have been also used for complex applications and purposes other than link prediction. This chapter does *not* claim that PMF is globally better than such methods, but only that PMF appears to have a better performance for link prediction on the LANL network.

In principle, it would be also possible to add covariates to the probabilistic models analysed in this section, including the term $\mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i \boldsymbol{v}_j^\top)\mathbf{1}_H$ within the link functions. However, this would be extremely unpractical, since the likelihood for such models is $\mathcal{O}(|V_1||V_2|)$, and the calculation of $\boldsymbol{u}_i \boldsymbol{v}_j^\top$ for all pairs (i, j) is computationally expensive and carries a large memory requirement (18 gigabytes on *User – Source*), in the order of magnitude of $\mathcal{O}(|V_1||V_2|KH)$. Therefore, in this section, only the standard models, without the covariate extension, were compared. On the other hand, fitting EPMF in (8.3) only requires to calculate $\boldsymbol{u}_i \boldsymbol{v}_j^\top$ for all pairs such that $A_{ij} > 0$, which is

Table 8.6.: *AUC scores for different link prediction algorithms on the two data sets, with* $d = 20$.

	User – Destination		*User – Source*	
	All links	New links	All links	New links
Probabilistic matrix factorisation	0.93886	0.61006	0.83022	0.64154
Logistic matrix factorisation	0.98079	0.95357	0.85252	0.84330
Random dot product graph, DASE	0.95050	0.69184	0.86202	0.60935
Random dot product graph, tKatz	0.95392	0.71754	0.86247	0.61120
Non-negative matrix factorisation	0.93985	0.61675	0.86941	0.61107
Degree-based model	0.95433	0.72505	0.82719	0.70393

$\mathcal{O}(\mathrm{nnz}(\mathbf{A})KH)$. This operation only takes 6.5 megabytes in memory for *User – Source*, since $\mathbf{A}$ is extremely sparse in the cyber-security application. This is a very significant practical advantage of the proposed model.

8.4.6. Comparison with a joint model

In the previous sections, the graphs *User – Source* and *User – Destination* were analysed separately. In this section, the predictive performance of the two individual models is compared to a joint PMF model, where the user-specific latent features are shared between the two graphs. In particular, assume that $\mathbf{A} \in \{0,1\}^{|V_1|\times|V_2|}$ represents the adjacency matrix for *User – Source*, and $\mathbf{A}' \in \{0,1\}^{|V_1|\times|V_2'|}$ for *User – Destination*. The users are assigned latent positions $\boldsymbol{x}_i,\ i = 1,\ldots,|V_1|$ and covariates $\boldsymbol{u}_i$, whereas the source and destination hosts are given latent positions $\boldsymbol{y}_j,\ j = 1,\ldots,|V_2|$ and $\boldsymbol{y}'_j,\ j = 1,\ldots,|V_2'|$ respectively, and covariates $\boldsymbol{v}_j$ and $\boldsymbol{v}'_j$. The joint extended PMF model (JEPMF) assumes:

$$\begin{aligned} A_{ij} &= \mathbb{1}_{\mathbb{N}_+}(N_{ij}),\ N_{ij} \sim \text{Poisson}\left(\boldsymbol{x}_i^\top \boldsymbol{y}_j + \mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i\boldsymbol{v}_j^\top)\mathbf{1}_H\right), \\ A'_{ij} &= \mathbb{1}_{\mathbb{N}_+}(N'_{ij}),\ N'_{ij} \sim \text{Poisson}\left(\boldsymbol{x}_i^\top \boldsymbol{y}'_j + \mathbf{1}_K^\top(\boldsymbol{\Phi}' \odot \boldsymbol{u}_i\boldsymbol{v}_j'^\top)\mathbf{1}_{H'}\right). \end{aligned} \tag{8.12}$$

Similarly, a standard joint PMF model would have the same structure as (8.12), without the covariate term. Variational inference for the joint model proceeds similarly to Algorithm 8.1, with minor differences in the updates for $\lambda_{ir}^{(x)}$ and $\mu_{ir}^{(x)}$. More details are given in Appendix C.4.

The results are presented in Table 8.7. The AUC scores were obtained fitting the joint model and then assessing the predictive performance on *User – Source* and *User – Destination* separately. Comparing the results with the predictions in Table 8.2 for the two individual models, joint PMF produces similar results to the individual models. This suggests that users have a similar behaviour across the two graphs, since adding the constraint of identical user features does not significantly decrease the predictive performance.

Table 8.7.: *AUC scores for prediction using the joint PMF model. Number of latent features:* $d = 20$.

	User – Destination		*User – Source*	
	PMF	EPMF	PMF	EPMF
All links	0.98206	0.98290	0.86061	0.95585
New links	0.94563	0.94565	0.89414	0.94637

Table 8.8.: *AUC scores for prediction of all and new links using seasonal PMF.*

	User – Destination		*User – Source*	
	EPMF	SEPMF	EPMF	SEPMF
All links	0.96550	0.96205	0.93559	0.92829
New links	0.87107	0.89337	0.85009	0.85748

8.4.7. Seasonal modelling

To investigate dynamic modelling, binary adjacency matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{82}$ are calculated for each day across the train and test periods. The seasonal PMF model with the inclusion of covariates (SEPMF) (8.11) is then compared against EPMF; for EPMF, the adjacency matrices are assumed to be independent realisations randomly generated from a fixed set of latent features. Due to a "9 day-80 hour" work schedule operated at LANL, whereby employees can elect to take vacation every other Friday, the seasonal period is assumed to be comprised of four segments: weekdays (Monday - Thursday), weekends (Saturday and Sunday), and two separate segments for alternating Friday's. For each model, binary classification is performed using the model predictive scores calculated across the entire period. For the positive class, scores are calculated for all user-host pairs (i, j) such that $A_{ijt} = 1$ for at least one t in the test set; for the negative class, a random sample of (i, j) pairs such that $A_{ijt} = 0$ for all t in the test set are obtained, with sample size equal to three times the total number of observed links.

Table 8.8 presents the resulting AUC scores. For both networks, the seasonal model does not globally outperform the extended PMF model for *all links*. However, improvements are obtained for prediction of the *new links*. One explanation for the weaker overall performance could be the reduced training sample size implied for the seasonal model: EPMF in a dynamic setting assumes that the all daily graphs have been sampled from the same process, whereas if the seasonal model is used then the daily graphs are only informative for the corresponding seasonal segments. In addition, as mentioned in the introduction to this chapter, elements of the data exhibit strong polling patterns, often due to computers automatically authenticating on users' behalves (Turcotte, Kent, and Hash, 2018); some of the links that exhibit polling will not exhibit seasonal patterns, as the human behaviour has not been separated from the automated behaviour.

On the other hand, improvements in the estimation of new links, despite the reduced training

sample size, demonstrates that it can be beneficial to understand the temporal dynamics of the network for these cases. Considering the context of the application, it might be perfectly normal for a user to authenticate to a computer during the week; however, that same authentication would be extremely unusual on the weekends when the user is not present at work. Without the seasonal model this behavioural difference in would be missed.

8.5. Conclusion and discussion

In this chapter, extensions of the standard Poisson matrix factorisation model have been proposed, motivated by applications to computer network data, in particular the LANL 2017 enterprise computer network. The extensions are threefold: handling binary matrices, including covariates for users and hosts in the PMF framework, and accounting for seasonal effects. The counts N_{ijt} have been treated as censored, and it has been assumed that only the binary indicator $A_{ijt} = \mathbb{1}_{\mathbb{N}_+}(N_{ijt})$ is observed. Starting from the hierarchical Poisson matrix factorisation model of Gopalan, Hofman, and Blei (2015), which only includes the latent features $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$, covariates have been included through the matrix of coefficients $\boldsymbol{\Phi}$. Seasonal adjustments for the coefficients are obtained through the variables $\boldsymbol{\gamma}_{it'}$ and $\boldsymbol{\delta}_{jt'}$. A variational inference algorithm is proposed, suitably adapted for the Bernoulli-Poisson link. This chapter mainly considered categorical covariates, but the methodology could be extended to include other forms of covariates, with minimal modifications to the inferential algorithms.

Despite the focus on bipartite graphs here, the proposed methodologies could be readily adapted to undirected and general directed graphs. For an undirected graph, the PMF model with Ber-Po link would assume:

$$A_{ij} = \mathbb{1}_{\mathbb{N}_+}(N_{ij}),\ N_{ij} \sim \text{Poisson}\left(\boldsymbol{x}_i^\top \boldsymbol{x}_j\right),\ i < j,\ A_{ij} = A_{ji}. \tag{8.13}$$

For directed graphs on the same node set, it could be assumed that each node has the same behaviour as source and destination of the connection, implying equation (8.13) for undirected graphs applies, removing the constraint $A_{ij} = A_{ji}$. Alternatively, each node could be given two latent features: $\boldsymbol{x}_i$ for its behaviour as source and $\boldsymbol{y}_i$ for its behaviour as destination. This is a special case of the bipartite graph model, where $V_1 \equiv V_2$, hence (8.3) applies.

The results show improvements over alternative models for link prediction purposes on the real computer network data. Including covariates provides significant uplift in predictive performance and allows for prediction for new nodes arriving into the network. The seasonal model provides time-varying anomaly scores and offers marginal improvements for predicting new links, which are of primary interest in cyber-security applications. It must be also noted that the latent features estimated using PMF could be easily embedded within the dynamic link prediction framework discussed in Chapter 7.

8.5.1. Limitations and future work

One of the main limitations of the extended PMF model presented in this chapter is that the coefficients x_{ir}, y_{jr} and ϕ_{kh} are non-negative, and they all contribute to the summation in the Poisson rate. Furthermore, the number of latent features d must be specified in advance. In this chapter, the choice of d was based on the elbow of the scree-plot of singular values (Jolliffe, 2002), but model based approaches could be alternatively used, similarly to Chapters 5 and 6, allowing d to be in principle unbounded. In future work, the PMF model in (8.3) could be generalised by introducing binary coefficients $\Delta \in \{0, 1\}$, used to switch on or off the latent variables x_{ir} and y_{jr}, allowing for a sparse representation. Covariate terms could also be removed using a similar approach, resulting in the following model:

$$A_{ij} = \mathbb{1}_{\mathbb{N}_+}(N_{ij}),\ N_{ij} \overset{d}{\sim} \text{Poisson}\left(\sum_{r=1}^{\infty} x_{ir}\Delta_{ir}^{(x)} y_{jr}\Delta_{jr}^{(y)} + \sum_{k=1}^{K}\sum_{h=1}^{H} \phi_{kh} u_{ik}\Delta_{k}^{(u)} v_{jh}\Delta_{h}^{(u)}\right), \tag{8.14}$$

where $\Delta_{ir}^{(x)}, \Delta_{jr}^{(y)}, \Delta_{k}^{(u)}, \Delta_{h}^{(v)} \in \{0, 1\}$. A suitable prior on the infinite matrices $\mathbf{\Delta}^{(x)} = \{\Delta_{ir}^{(x)}\}$ and $\mathbf{\Delta}^{(y)} = \{\Delta_{jr}^{(y)}\}$ could be the Indian Buffet Process (IBP) (Ghahramani and Griffiths, 2006; Griffiths and Ghahramani, 2011), widely used in the Bayesian non-parametric literature primarily for network models (Meeds et al., 2007; Miller, Griffiths, and Jordan, 2010). The IBP process is the infinite limit for $m \to \infty$ of a Beta-Bernoulli process (Thibaux and Jordan, 2007):

$$\begin{aligned}
\Delta_{ir}^{(x)} | \pi_r^{(x)} &\sim \text{Bernoulli}(\pi_r^{(x)}),\ \pi_r^{(x)} \sim \text{Beta}(\alpha/m, 1),\ r = 1, \dots, m, \\
\Delta_{jr}^{(y)} | \pi_r^{(y)} &\sim \text{Bernoulli}(\pi_r^{(y)}),\ \pi_r^{(y)} \sim \text{Beta}(\beta/m, 1),\ r = 1, \dots, m.
\end{aligned}$$

If m is fixed to a large value, using the same idea in Chapters 5 and 6, the value of d could be set to the largest $r < m$ such that at least one $\Delta_{ir}^{(x)} = 1$ or $\Delta_{jr}^{(y)} = 1$. A similar Beta-Bernoulli hierarchical prior could be also placed on the binary variables corresponding to the covariates, $\Delta_{k}^{(u)}$ and $\Delta_{h}^{(v)}$. Introducing the latent binary coefficients Δ implies that the CAVI algorithm cannot be trivially applied. Alternative options are a Gibbs sampler, or structured stochastic variational inference (Hoffman and Blei, 2015), which would significantly increase the computational cost associated with the inferential procedure. Future work should investigate methods for performing inference efficiently under the model (8.14), even for large networks, and test whether sparsity of the latent features, or the random value of d, have a significant impact for link prediction.

Finally, the sequence of adjacency matrices $\mathbf{A}_1, \dots, \mathbf{A}_T$ for *User – Source* or *User – Destination* could be interpreted as a $(|V_1| \times |V_2| \times T)$-tensor. Following Schein et al. (2015), the extended PMF model presented in this chapter could be extended to a tensor-based version, including static or time-varying covariate coefficients using the same mathematical formulation in (8.3), and using a similar variational inference algorithm.

9

Conclusion

In this thesis, latent factor models have been demonstrated to be a versatile modelling framework for network analysis, and especially useful for cyber-security applications. A number of different models have been proposed, tackling different challenges, motivated by specific characteristics of computer networks. Overall, this work provides contributions towards a unified Bayesian model for cyber-security applications, with the ultimate aim to predict dynamically the connectivity of IP addresses, or users and hosts, in a computer network.

First, this thesis has discussed models for scoring individual events within a dynamic network. In particular, contributions in two under-researched areas are given: network-wide modelling of sequences of events, and classification of periodic activity within individual network edges.

In Chapter 3, a novel network-wide Bayesian nonparametric model based on the Pitman-Yor process has been proposed, and used in an anomaly detection study. The Pitman-Yor process admits power laws, which are commonly observed in computer networks both in the degree distributions and on the the frequencies of clients connecting to a given server, and vice versa. A parameter estimation technique based on the method of moments has been discussed, and suitable adaptations for uniform discrete base measures have been addressed. Compromised nodes in the LANL 2015 network were identified by combination of p-values of individual events.

In Chapter 4, a novel methodology for classification of periodic arrivals in event time data has been discussed. Existing frameworks for classification of polling activity deem the *entire* edge as periodic or non-periodic. The proposed model classifies individual *events*, a refinement over existing methodologies. The model utilised a periodicity estimated using Fourier analysis, which is used to transform the observed arrival times to the unit circle. Parametric wrapped distributions are used to model such periodic arrivals. This information is then combined with the daily arrival time of each event, which is assumed to be distributed according to an unknown step function density. The model parameters are estimated using Bayesian inferential methods. Simulated and real world

examples demonstrate that the model is able to correctly separate a significant proportion of the periodic and non-periodic activity on mixed-type edges.

The second and third parts of the thesis discussed models for the entire graph, motivated by two important tasks for computer network security: graph clustering and link prediction. Graph clustering is useful to identify nodes with similar connectivity patterns. For anomaly detection, abrupt changes in the community membership of a user might imply that its credentials have been compromised. On the other hand, link prediction represents a mathematical framework to understand the normal process of link formation, which is crucial to detect a cyber-attack. Link prediction is particularly important since malicious behaviour is often associated with the appearance of new connections, and it is therefore paramount to have a sound understanding of the normal connectivity patterns across the network.

In Chapter 5, a novel method for simultaneous estimation of the latent dimensionality and number of communities in spectral clustering has been proposed, under a generalised random dot product graph interpretation of the stochastic blockmodel. Traditional methods for selection of the number of communities in SBMs rely on the assumption of correct specification of the underlying latent dimensionality of the model. If the number of blocks is selected conditional on the wrong number of latent features, the clustering performance is suboptimal. The proposed methodology instead provides a *simultaneous* estimation framework for the two quantities. The model has a good performance at recovering the correct values of the parameter in synthetic networks, and gives sensible results on real world graphs.

For cyber-security applications, in particular network flow data, the model from Chapter 5 might represent an oversimplification of the underlying block structure. Therefore, in Chapter 6, this methodology has been extended to degree corrected stochastic blockmodels, which admit heterogeneous within-community degree distributions. The extension is based on a novel approximation of the distribution of the spherical coordinates of spectral embeddings, which has been shown to work well in simulated and real world scenarios. The approach is aimed at transforming towards normality the embeddings obtained from DCSBMs, which, according to the theory, are noisy *rays* passing through the origin of the latent space, as demonstrated in Figure 6.1b. The modelling assumptions are then incorporated within a model selection framework akin to Chapter 5, providing a method for jointly estimating the latent space dimension and number of communities also in DCSBMs.

Chapter 7 extends the discussion of generalised random dot product graphs to the link prediction framework. The performance for dynamic link prediction from GRDPG models for multiple graph inference has been analysed, providing guidelines to practitioners on the most appropriate choices for combining individual embeddings. RDPGs are demonstrated to be a scalable and powerful tool for link prediction in large networks. Furthermore, a time series extension was presented, which could be used to account for the temporal dynamics of link probabilities.

Finally, Chapter 8 has proposed a novel extension of Poisson matrix factorisation, a popular model in the literature of recommender systems. The contributions are threefold: a framework for including nodal covariates within the PMF model, a fast variational inference algorithm for PMF on binary matrices, and a seasonal PMF model. The proposed model and inferential procedure are scalable to very large sparse networks because of the convenient form of the likelihood function, which only depends on the non-zero entries of the adjacency matrix. On the LANL 2017 network, the extended PMF model appears to have a better predictive performance than other statistical network models, which, combined with its property of scalability, makes it an ideal candidate for modelling adjacency matrices arising from user-authentication or network flow data.

Limitations of each of the methodologies presented in this thesis have been discussed in detail at the end of each chapter, and directions for future work have also been given. More generally, it has been shown in this thesis that models for individual events and global models for the entire network can both provide valuable insights on the normal behaviour of a system. A natural extension of such methodologies would be to combine modelling ideas from these two different approaches. For example, the edge-based mutually exciting model (4.28) of Price-Williams and Heard (2019) could be given a network structure, parametrising the edge intensities using only functions of node-specific latent factors. Also, a recent body of literature is exploring the advantages of topic modelling for computer networks. In particular, an application to the LANL 2015 network is described in Heard, Palla, and Skoularidou (2016). For each client host on a computer network, for example, a day of connection events can be interpreted as a document, with each destination computer regarded as a word, and the entire set of records as a corpus. In NetFlow data, each flow could be assumed to be generated by an element of a vector of latent *devices* (corresponding to the *topics* in the topic modelling language). The model for separation of human and automated activity in Chapter 4 could be considered as a special case of this framework, where the latent devices generating the flows are simply a human and a machine.

In summary, computer network data have unique characteristics that must be accounted for in the model building process. This thesis provides ideas towards a comprehensive statistical model for a computer network, proposing data-driven methodologies at different levels of resolution, from the individual events to the entire network connectivity. Despite the heavy focus on computer network modelling of this thesis, it must be remarked that the proposed methodologies could extend well beyond cyber-security applications, and used to obtain insights on dynamic network data from any application domain.

Bibliography

Abbe, E. (2018). “Community Detection and Stochastic Block Models: Recent Developments”. In: *Journal of Machine Learning Research* 18(177), pp. 1–86.

Abbe, E., Bandeira, A. S., and Hall, G. (2016). “Exact Recovery in the Stochastic Block Model”. In: *IEEE Transactions on Information Theory* 62, pp. 471–487.

Acharya, A. et al. (2015). “Gamma Process Poisson Factorization for Joint Modeling of Network and Documents”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, pp. 283–299.

Adams, N. and Heard, N. A. (2014). *Data Analysis for Network Cyber-Security*. Imperial College Press.

Adams, N. and Heard, N. A. (2016). *Dynamic Networks and Cyber-Security*. World Scientific.

Adomavicius, G. and Tuzhilin, A. (2005). “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions”. In: *IEEE Transactions on Knowledge and Data Engineering* 17(6), pp. 734–749.

Agarwal, D., Zhang, L., and Mazumder, R. (Sept. 2011). “Modeling item-item similarities for personalized recommendations on Yahoo! front page”. In: *Annals of Applied Statistics* 5(3), pp. 1839–1875.

Ahmed, M., Naser Mahmood, A., and Hu, J. (2016). “A survey of network anomaly detection techniques”. In: *Journal of Network and Computer Applications* 60, pp. 19–31.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2008). “Mixed Membership Stochastic Blockmodels”. In: *Journal of Machine Learning Research* 9, pp. 1981–2014.

Akoglu, L., Tong, H., and Koutra, D. (2015). “Graph based anomaly detection and description: a survey”. In: *Data Mining and Knowledge Discovery* 29(3), pp. 626–688.

Aldous, D. (1985). “Exchangeability and related topics”. In: *École d’Été de Probabilités de Saint-Flour XIII-1983*, pp. 1–198.

Amini, A. A., Chen, A., Bickel, P. J., and Levina, E. (2013). "Pseudo-likelihood methods for community detection in large sparse networks". In: *Annals of Statistics* 41(4), pp. 2097–2122.

Amit, I. et al. (2019). "Machine Learning in Cyber-Security - Problems, Challenges and Data Sets". In: *AAAI-19 Workshop on Engineering Dependable and Secure Machine Learning Systems*.

Anderson, B., Vejman, M., McGrew, D., and Paul, S. (2018). "Towards Generalisable Network Threat Detection". In: *Data Science for Cyber-Security*. World Scientific. Chap. 4, pp. 77–94.

Anderson, T. W. (1971). *The Statistical Analysis of Time-Series*. New York: John Wiley & Sons.

Arroyo-Relión, J. D., Kessler, D., Levina, E., and Taylor, S. F. (2019). "Network classification with applications to brain connectomics". In: *Annals of Applied Statistics* 13(3), pp. 1648–1677.

Arroyo-Relión, J. D. et al. (2020). "Inference for multiple heterogeneous networks with a common invariant subspace". In: *Journal of Machine Learning Research* (to appear).

AsSadhan, B. and Moura, J. M. F. (2014). "An efficient method to detect periodic behavior in botnet traffic by analyzing control plane traffic". In: *Journal of Advanced Research* 5(4), pp. 435–448.

Athreya, A. et al. (2016). "A Limit Theorem for Scaled Eigenvectors of Random Dot Product Graphs". In: *Sankhya A* 78(1), pp. 1–18.

Athreya, A. et al. (2018). "Statistical Inference on Random Dot Product Graphs: a Survey". In: *Journal of Machine Learning Research* 18(226), pp. 1–92.

Athreya, A., Tang, M., Park, Y., and Priebe, C. E. (2020). "On estimation and inference in latent structure random graphs". In: *Statistical Science* (to appear).

Barabási, A. L. (2005). "The origin of bursts and heavy tails in human dynamics". In: *Nature* 435(207–211).

Barbosa, R. R. R., Sadre, R., and Pras, A. (2012). "Towards periodicity based anomaly detection in SCADA networks". In: *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012)*, pp. 1–4.

Bartlett, G., Heidemann, J., and Papadopoulos, C. (2011). "Low-rate, flow-level periodicity detection". In: *2011 IEEE Conference on Computer Communications Workshops*, pp. 804–809.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). "A Neural Probabilistic Language Model". In: *Journal of Machine Learning Research* 3, pp. 1137–1155.

Benjamin, M. A., Rigby, R. A., and Stasinopoulos, D. M. (2003). "Generalized Autoregressive Moving Average Models". In: *Journal of the American Statistical Association* 98(461), pp. 214–223.

Bernardo, J. M. and Smith, A. F. M. (1994). *Bayesian Theory*. Wiley Series in Probability & Statistics. Wiley.

Bickel, P. J. and Chen, A. (2009). "A nonparametric view of network models and Newman–Girvan and other modularities". In: *Proceedings of the National Academy of Sciences* 106(50), pp. 21068–21073.

Bickel, P. J. and Sarkar, P. (2016). “Hypothesis testing for automated community detection in networks”. In: *Journal of the Royal Statistical Society: Series B* 78(1), pp. 253–273.

Biernacki, C., Celeux, G., and Govaert, G. (2000). “Assessing a mixture model for clustering with the integrated completed likelihood”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(7), pp. 719–725.

Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., and Kruegel, C. (2012). “DISCLOSURE: Detecting botnet command and control servers through large-scale netflow analysis”. In: *ACSAC 2012, 28th Annual Computer Security Applications Conference*.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112(518), pp. 859–877.

Blumenson, L. E. (1960). “A Derivation of n-Dimensional Spherical Coordinates”. In: *The American Mathematical Monthly* 67(1), pp. 63–66.

Brockwell, P. J. and Davis, R. A. (1987). *Time Series: Theory and Methods*. Springer Series in Statistics. Springer-Verlag.

Buczak, A. L. and Guven, E. (2016). “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection”. In: *IEEE Communications Surveys Tutorials* 18(2), pp. 1153–1176.

Buntine, W. and Hutter, M. (2010). “A Bayesian View of the Poisson-Dirichlet Process”. In: *ArXiv e-prints*. arXiv: 1007.0296.

Cai, H., Zheng, V. W., and Chang, K. C. (2018). “A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications”. In: *IEEE Transactions on Knowledge and Data Engineering* 30, pp. 1616–1637.

Canny, J. (2004). “GaP: A Factor Model for Discrete Data”. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’04, pp. 122–129.

Cape, J., Tang, M., and Priebe, C. E. (2019). “On spectral embedding performance and elucidating network structure in stochastic block model graphs”. In: *Network Science* 7(3), pp. 269–291.

Casella, G. and Berger, R. L. (2002). *Statistical Inference*. Duxbury Advanced Series in Statistics and Decision Sciences. Thomson Learning.

Celisse, A., Daudin, J., and Pierre, L. (2012). “Consistency of maximum-likelihood and variational estimators in the stochastic block model”. In: *Electronic Journal of Statistics* 6, pp. 1847–1899.

Cemgil, A. T. (2009). “Bayesian Inference for Nonnegative Matrix Factorisation Models”. In: *Computational Intelligence and Neuroscience*.

Chandola, V., Banerjee, A., and Kumar, V. (2009). “Anomaly Detection: A Survey”. In: *ACM Computing Surveys* 41(3).

Chaney, A. J. B., Blei, D. M., and Eliassi-Rad, T. (2015). "A Probabilistic Model for Using Social Networks in Personalized Item Recommendation". In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys '15, pp. 43–50.

Charlin, L., Ranganath, R., McInerney, J., and Blei, D. M. (2015). "Dynamic Poisson Factorization". In: *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 155–162.

Chatterjee, S. (2015). "Matrix estimation by universal singular value thresholding". In: *The Annals of Statistics* 43(1), pp. 177–214.

Chaudhuri, K., Chung, F., and Tsiatas, A. (2012). "Spectral Clustering of Graphs with General Degrees in the Extended Planted Partition Model". In: *Proceedings of the 25th Annual Conference on Learning Theory*. Vol. 23.

Chen, B., Li, F., Chen, S., Hu, R., and Chen, L. (2017). "Link prediction based on non-negative matrix factorization". In: *PLOS ONE* 12(8), pp. 1–18.

Chen, C., Du, L., and Buntine, W. (2011). "Sampling Table Configurations for the Hierarchical Poisson-Dirichlet Process". In: *Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 296–311.

Chen, H. and Li, J. (2018). "Exploiting Structural and Temporal Evolution in Dynamic Link Prediction". In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 427–436.

Chen, K. and Lei, J. (2018). "Network Cross-Validation for Determining the Number of Communities in Network Data". In: *Journal of the American Statistical Association* 113(521), pp. 241–251.

Chen, L. M., Hsiao, S. W., Chen, M. C., and Liao, W. (2016). "Slow-Paced Persistent Network Attacks Analysis and Detection Using Spectrum Analysis". In: *IEEE Systems Journal* 10(4), pp. 1326–1337.

Chen, Y., Li, X., and Xu, J. (2018). "Convexified modularity maximization for degree-corrected stochastic block models". In: *The Annals of Statistics* 46(4), pp. 1573–1602.

Choi, D. S., Wolfe, P. J., and Airoldi, E. M. (2012). "Stochastic blockmodels with a growing number of classes". In: *Biometrika* 99(2), pp. 273–284.

Cicuttin, A., Colavita, A. A., Cerdeira, A., Mutihac, R., and Turrini, S. (1998). "A Simple Method for Detecting Periodic Signals in Sparse Astronomical Event Data". In: *The Astrophysical Journal* 498(2), pp. 666–670.

Claise, B. (2004). "Cisco Systems NetFlow Services Export Version 9". In: *RFC 3954, Internet Engineering Task Force*.

Clauset, A., Moore, C., and Newman, M. E. J. (2008). "Hierarchical structure and the prediction of missing links in networks". In: *Nature* 453.

Côme, E. and Latouche, P. (2015). "Model selection and clustering in stochastic block models based on the exact integrated complete data likelihood". In: *Statistical Modelling* 15(6), pp. 564–589.

Comer, D. (2014). *Internetworking with TCP/IP: Principles, protocols, and architecture*. 6th ed. Vol. 1. Pearson.

Dahl, D. B. (2003). *An improved merge-split sampler for conjugate Dirichlet process mixture models*. Tech. rep. 1086. Department of Statistics, University of Wisconsin, Madison.

Dai, B., Wang, J., Shen, X., and Qu, A. (2019). "Smooth neighborhood recommender systems". In: *Journal of Machine Learning Research* 20(16), pp. 1–24.

de Lichtenberg, U. et al. (2005). "Comparison of computational methods for the identification of cell cycle-regulated genes". In: *Bioinformatics* 21(7), pp. 1164–1171.

Dellaportas, P. and Papageorgiou, I. (2006). "Multivariate mixtures of normals with unknown number of components". In: *Statistics and Computing* 16(1), pp. 57–68.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the Royal Statistical Society, Series B* 39(1), pp. 1–38.

Deng, D. et al. (2016). "Latent Space Model for Road Networks to Predict Time-Varying Traffic". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1525–1534.

Dhillon, I. S. (2001). "Co-clustering Documents and Words Using Bipartite Spectral Graph Partitioning". In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '01, pp. 269–274.

Dong, X., Frossard, P., Vandergheynst, P., and Nefedov, N. (2014). "Clustering on Multi-Layer Graphs via Subspace Analysis on Grassmann Manifolds". In: *IEEE Transactions on Signal Processing* 62(4), pp. 905–918.

Donoho, D. L. and Johnstone, I. M. (1994). "Ideal spatial adaptation by wavelet shrinkage". In: *Biometrika* 81(3), pp. 425–455.

Dryden, I. L. and Mardia, K. V. (2016). *Statistical Shape Analysis, with Applications in R*. John Wiley and Sons.

Dunlavy, D. M., Kolda, T. G., and Acar, E. (2011). "Temporal link prediction using matrix and tensor factorizations". In: *ACM Transactions on Knowledge Discovery from Data* 5(2).

Dunson, D. B. and Herring, A. H. (2005). "Bayesian latent variable models for mixed discrete outcomes". In: *Biostatistics* 6(1), pp. 11–25.

Durante, D. and Dunson, D. B. (2014). "Nonparametric Bayes dynamic modelling of relational data". In: *Biometrika* 101(4), pp. 883–898.

Durante, D. and Dunson, D. B. (2016). "Locally adaptive dynamic networks". In: *Annals of Applied Statistics* 10(4), pp. 2203–2232.

Durante, D. and Dunson, D. B. (2018). "Bayesian Inference and Testing of Group Differences in Brain Networks". In: *Bayesian Analysis* 13(1), pp. 29–58.

Durante, D., Dunson, D. B., and Vogelstein, J. T. (2017). "Nonparametric Bayes Modeling of Populations of Networks". In: *Journal of the American Statistical Association* 112(520), pp. 1516–1530.

Escobar, M. D. and West, M. (1995). "Bayesian Density Estimation and Inference Using Mixtures". In: *Journal of the American Statistical Association* 90(430), pp. 577–588.

Eslahi, M. et al. (2015). "Periodicity classification of HTTP traffic to detect HTTP Botnets". In: *2015 IEEE Symposium on Computer Applications Industrial Electronics*, pp. 119–123.

Ferguson, T. S. (1973). "A Bayesian Analysis of Some Nonparametric Problems". In: *Annals of Statistics* 1, pp. 209–230.

Fienberg, S. E. (2012). "A Brief History of Statistical Models for Network Analysis and Open Challenges". In: *Journal of Computational and Graphical Statistics* 21(4), pp. 825–839.

Fisher, R. A. (1929). "Tests of Significance in Harmonic Analysis". In: *Proceedings of the Royal Society of London. Series A.* 125(796), pp. 54–59.

Fisher, R. A. (1934). *Statistical Methods for Research Workers*. 4th ed. Edinburgh: Oliver & Boyd.

Fithian, W. and Mazumder, R. (2018). "Flexible Low-Rank Statistical Modeling with Side Information". In: *Statistical Science* 33(2), pp. 238–260.

Fortunato, S. (2010). "Community detection in graphs". In: *Physics Reports* 486(3), pp. 75–174.

Fowlkes, E. B., Gnanadesikan, R., and Kettenring, J. R. (1988). "Variable selection in clustering". In: *Journal of Classification* 5(2), pp. 205–228.

Fraley, C. and Raftery, A. E. (2002). "Model-Based Clustering, Discriminant Analysis, and Density Estimation". In: *Journal of the American Statistical Association* 97(458), pp. 611–631.

Fritsch, A. and Ickstadt, K. (2009). "Improved criteria for clustering based on the posterior similarity matrix". In: *Bayesian Analysis* 4(2), pp. 367–391.

Gama, J., Sebastião, R., and Rodrigues, P. P. (2013). "On evaluating stream learning algorithms". In: *Machine Learning* 90(3), pp. 317–346.

Gao, C., Ma, Z., Zhang, A. Y., and Zhou, H. H. (2018). "Community detection in degree-corrected block models". In: *Annals of Statistics* 46(5), pp. 2153–2185.

Gao, J. et al. (2010). "On Community Outliers and Their Efficient Detection in Information Networks". In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '10, pp. 813–822.

Gao, S., Denoyer, L., and Gallinari, P. (2011). "Temporal Link Prediction by Integrating Content and Structure Information". In: *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pp. 1169–1174.

Gelman, A. et al. (2013). *Bayesian Data Analysis*. Second edition. Chapman and Hall/CRC.

Geman, S. and Geman, D. (1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6(6), pp. 721–741.

Ghahramani, Z. and Griffiths, T. L. (2006). “Infinite latent feature models and the Indian buffet process”. In: *Advances in Neural Information Processing Systems 18*. MIT Press, pp. 475–482.

Ghashami, M., Liberty, E., and Phillips, J. M. (2016). “Efficient Frequent Directions Algorithm for Sparse Matrices”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 845–854.

Ginestet, C. E., Li, J., Balachandran, P., Rosenberg, S., and Kolaczyk, E. D. (2017). “Hypothesis testing for network data in functional neuroimaging”. In: *Annals of Applied Statistics* 11(2), pp. 725–750.

Girvan, M. and Newman, M. E. J. (2002). “Community structure in social and biological networks”. In: *Proceedings of the National Academy of Sciences* 99(12), pp. 7821–7826.

Givens, G. H. and Hoeting, J. A. (2012). *Computational Statistics*. Second edition. John Wiley & Sons, Inc.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. (2009). “A survey of statistical network models”. In: *Foundations and Trends in Machine Learning* 2, pp. 129–233.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2005). “Interpolating Between Types and Tokens by Estimating Power-law Generators”. In: *Proceedings of the 18th International Conference on Neural Information Processing Systems*. MIT Press, pp. 459–466.

Gopalan, P., Charlin, L., and Blei, D. M. (2014). “Content-based Recommendations with Poisson Factorization”. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems*. Vol. 2. NIPS’14. MIT Press, pp. 3176–3184.

Gopalan, P., Hofman, J. M., and Blei, D. M. (2015). “Scalable Recommendation with Hierarchical Poisson Factorization”. In: *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*. UAI’15. AUAI Press, pp. 326–335.

Govaert, G. and Nadif, M. (2010). “Latent Block Model for Contingency Table”. In: *Communications in Statistics - Theory and Methods* 39(3), pp. 416–425.

Gower, J. C. (1975). “Generalized Procrustes analysis”. In: *Psychometrika* 40(1), pp. 33–51.

Green, P. J. (1995). “Reversible Jump Markov Chain Monte Carlo computation and Bayesian model determination”. In: *Biometrika* 82(4), pp. 711–732.

Griffiths, T. L. and Ghahramani, Z. (2011). “The Indian Buffet Process: An Introduction and Review”. In: *Journal of Machine Learning Research* 12, pp. 1185–1224.

Grover, A. and Leskovec, J. (2016). “node2vec: Scalable Feature Learning for Networks”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864.

Gu, G., Zhang, J., and Lee, W. (2008). “BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic”. In: *Proceedings of the 15th Annual Network and Distributed System Security Symposium*.

Gulikers, L., Lelarge, M., and Massoulié, L. (2017). "A spectral method for community detection in moderately sparse degree-corrected stochastic block models". In: *Advances in Applied Probability* 49(3), pp. 686–721.

Güneş, I., Gündüz-Öğüdücü, Ş., and Çataltepe, Z. (2016). "Link prediction using time series of neighborhood-based node similarity scores". In: *Data Mining and Knowledge Discovery* 30(1), pp. 147–180.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). "Inductive Representation Learning on Large Graphs". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035.

Handcock, M. S., Raftery, A. E., and Tantrum, J. M. (2007). "Model-based clustering for social networks". In: *Journal of the Royal Statistical Society: Series A* 170(2), pp. 301–354.

Hastings, W. K. (1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". In: *Biometrika* 57(1), pp. 97–109.

He, X., Papadopoulos, C., Heidemann, J., Mitra, U., and Riaz, U. (2009). "Remote detection of bottleneck links using spectral and statistical methods". In: *Computer Networks* 53(3), pp. 279–298.

Heard, N., Palla, K., and Skoularidou, M. (2016). "Topic modelling of authentication events in an enterprise computer network". In: *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pp. 190–192.

Heard, N. A. and Rubin-Delanchy, P. (2016). "Network-wide anomaly detection via the Dirichlet process". In: *Proceedings of the IEEE Workshop on Big Data Analytics for Cyber-security Computing*.

Heard, N. A. and Rubin-Delanchy, P. (2018). "Choosing between methods of combining p-values". In: *Biometrika* 105(1), pp. 239–246.

Heard, N. A., Rubin-Delanchy, P. T. G., and Lawson, D. J. (2014). "Filtering automated polling traffic in computer network flow data". In: *Proceedings - 2014 IEEE Joint Intelligence and Security Informatics Conference, JISIC 2014*, pp. 268–271.

Heard, N. A., Weston, D. J., Platanioti, K., and Hand, D. J. (2010). "Bayesian anomaly detection methods for social networks". In: *Annals of Applied Statistics* 4(2), pp. 645–662.

Heard, N. A., Adams, N., Rubin-Delanchy, P., and Turcotte, M. J. M. (2018). *Data Science for Cyber-Security*. World Scientific.

Herlau, T., Schmidt, M. N., and Mørup, M. (2014). "Infinite-degree-corrected stochastic block model". In: *Physical Review E* 90 (3).

Hernández-Lobato, J. M., Houlsby, N., and Ghahramani, Z. (2014). "Stochastic Inference for Scalable Probabilistic Modeling of Binary Matrices". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning*. Vol. 32, pp. 379–387.

Hernandez-Stumpfhauser, D., Breidt, F. J., and van der Woerd, M. J. (2017). "The General Projected Normal Distribution of Arbitrary Dimension: Modeling and Bayesian Inference". In: *Bayesian Analysis* 12(1), pp. 113–133.

Higdon, D. M. (1998). "Auxiliary Variable Methods for Markov Chain Monte Carlo with Applications". In: *Journal of the American Statistical Association* 93(442), pp. 585–595.

Hoff, P. D. (2005). "Bilinear Mixed-Effects Models for Dyadic Data". In: *Journal of the American Statistical Association* 100(469), pp. 286–295.

Hoff, P. D., Raftery, A. E., and Handcock, M. S. (2002). "Latent space approaches to social network analysis". In: *Journal of the American Statistical Association* 97(460), pp. 1090–1098.

Hoffman, M. D. and Blei, D. M. (2015). "Structured stochastic variational inference". In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*. Vol. 38, pp. 361–369.

Hofstede, R. et al. (2014). "Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX". In: *IEEE Communications Surveys Tutorials* 16(4), pp. 2037–2064.

Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). "Stochastic blockmodels: First steps". In: *Social Networks* 5(2), pp. 109 –137.

Hosseini, S. A. et al. (2020). "Recurrent Poisson factorization for temporal recommendation". In: *IEEE Transactions on Knowledge and Data Engineering* 32(1), pp. 121–134.

Hu, J., Qin, H., Yan, T., and Zhao, Y. (2019). "Corrected Bayesian Information Criterion for Stochastic Block Models". In: *Journal of the American Statistical Association* 0(0), pp. 1–13.

Huang, Z. and Zeng, D. D. (2006). "A Link Prediction Approach to Anomalous Email Detection". In: *2006 IEEE International Conference on Systems, Man and Cybernetics*. Vol. 2, pp. 1131–1136.

Hubballi, N. and Goyal, D. (2013). "FlowSummary: Summarizing Network Flows for Communication Periodicity Detection". In: *Pattern Recognition and Machine Intelligence*. Springer, pp. 695–700.

Hubert, L. and Arabie, P. (1985). "Comparing partitions". In: *Journal of Classification* 2(1), pp. 193–218.

Huggins, J. H., Campbell, T., Kasprzak, M., and Broderick, T. (2019). "Scalable Gaussian Process Inference with Finite-data Mean and Variance Guarantees". In: *Proceedings of Machine Learning Research*. Vol. 89, pp. 796–805.

Huynh, N. A., Ng, W. K., Ulmer, A., and Kohlhammer, J. (2016). "Uncovering periodic network signals of cyber attacks". In: *2016 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pp. 1–8.

Hyndman, R. and Khandakar, Y. (2008). "Automatic Time Series Forecasting: The forecast Package for R". In: *Journal of Statistical Software, Articles* 27(3), pp. 1–22.

Idé, T. and Kashima, H. (2004). “Eigenspace-Based Anomaly Detection in Computer Systems”. In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’04, pp. 440–449.

Ishiguro, K., Iwata, T., Ueda, N., and Tenenbaum, J. B. (2010). “Dynamic Infinite Relational Model for Time-varying Relational Data Analysis”. In: *Advances in Neural Information Processing Systems 23*, pp. 919–927.

Ishwaran, H. and James, L. F. (2001). “Gibbs sampling methods for stick-breaking priors”. In: *Journal of the American Statistical Association* 96(453), pp. 161–173.

Jain, S. and Neal, R. M. (2004). “A Split-Merge Markov Chain Monte Carlo Procedure for the Dirichlet Process Mixture Model”. In: *Journal of Computational and Graphical Statistics* 13(1), pp. 158–182.

Jasra, A., Holmes, C. C., and Stephens, D. A. (2005). “Markov Chain Monte Carlo Methods and the Label Switching Problem in Bayesian Mixture Modeling”. In: *Statistical Science* 20(1), pp. 50–67.

Jaynes, E. T. (1987). “Maximum Entropy and Bayesian Spectral Analysis and Estimation Problems”. In: *Bayesian Spectrum & Chirp Analysis*. Dordrecht, pp. 1–37.

Jenkins, G. M. and Priestley, M. B. (1957). “The Spectral Analysis of Time-Series”. In: *Journal of the Royal Statistical Society. Series B* 19(1), pp. 1–12.

Jeske, D. R., Stevens, N. T., Tartakovsky, A. G., and Wilson, J. D. (2018). “Statistical methods for network surveillance”. In: *Applied Stochastic Models in Business and Industry* 34(4), pp. 425–445.

Ji, T., Yang, D., and Gao, J. (2013). “Incremental Local Evolutionary Outlier Detection for Dynamic Social Networks”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 1–15.

Jin, J. (2015). “Fast community detection by SCORE”. In: *Annals of Statistics* 43(1), pp. 57–89.

Johnson, C. C. (2014). “Logistic matrix factorization for implicit feedback data”. In: *Advances in Neural Information Processing Systems* 27.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer.

Jurafsky, D., Martin, J. H., Norvig, P., and Russell, S. (2014). *Speech and Language Processing*. Pearson Education.

Karrer, B. and Newman, M. E. J. (2011). “Stochastic blockmodels and community structure in networks”. In: *Physical Review E* 83(1).

Kass, R. E. and Raftery, A. E. (1995). “Bayes Factors”. In: *Journal of the American Statistical Association* 90(430), pp. 773–795.

Kauppi, H. and Saikkonen, P. (2008). “Predicting U.S. Recessions with Dynamic Binary Response Models”. In: *The Review of Economics and Statistics* 90(4), pp. 777–791.

Kent, A. D. (2016). “Cybersecurity data sources for dynamic network research”. In: *Dynamic Networks and Cyber-Security*. World Scientific.

Khanna, R., Zhang, L., Agarwal, D., and Chen, B. C. (2013). “Parallel matrix factorization for binary response”. In: *IEEE International Conference on Big Data 2013*, pp. 430–438.

Khor, S. (2010). “Concurrency and network disassortativity”. In: *Artificial Life* 16(3), pp. 225–232.

Kim, B., Lee, K. H., Xue, L., and Niu, X. (2018). “A review of dynamic network models with latent variables”. In: *Statistics Surveys* 12, pp. 105–135.

Kim, Y. and Levina, E. (2019). “Graph-aware Modeling of Brain Connectivity Networks”. In: *arXiv e-prints*, arXiv:1903.02129. arXiv: 1903.02129.

Kingma, D. P. and Ba, J. (2015). “Adam: a Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR*.

Kintzel, U. (2005). “Procrustes problems in finite dimensional indefinite scalar product spaces”. In: *Linear Algebra and its Applications* 402, pp. 1–28.

Kocak, M., George, E., Pyne, S., and Pounds, S. (2013). “An empirical Bayes approach for analysis of diverse periodic trends in time-course gene expression data”. In: *Bioinformatics* 29(2), pp. 182–188.

Kolaczyk, E. D. (2009). *Statistical Analysis of Network Data: Methods and Models*. 1st edition. Springer Publishing Company, Incorporated.

Korwar, R. M. and Hollander, M. (1973). “Contributions to the Theory of Dirichlet Processes”. In: *Annals of Probability* 1(4), pp. 705–711.

Krivitsky, P. N. and Handcock, M. S. (2014). “A separable model for dynamic networks”. In: *Journal of the Royal Statistical Society: Series B* 76(1), pp. 29–46.

Kumar, R. S. S., Wicker, A., and Swann, M. (2017). “Practical Machine Learning for Cloud Intrusion Detection: Challenges and the Way Forward”. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. AISec ’17, pp. 81–90.

Lambert, D. and Liu, C. (2006). “Adaptive Thresholds”. In: *Journal of the American Statistical Association* 101(473), pp. 78–88.

Lancaster, H. (1952). “Statistical control of counting experiments”. In: *Biometrika* 39, pp. 419–422.

Latouche, P., Birmelé, E., and Ambroise, C. (2012). “Variational Bayesian inference and complexity control for stochastic block models”. In: *Statistical Modelling* 12(1), pp. 93–115.

Lau, J. W. and Green, P. J. (2007). “Bayesian Model-Based Clustering Procedures”. In: *Journal of Computational and Graphical Statistics* 16(3), pp. 526–558.

Law, M. H. C., Figueiredo, M. A. T., and Jain, A. K. (2004). “Simultaneous feature selection and clustering using mixture models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(9), pp. 1154–1166.

Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., and Srivastava, J. (2003). "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection". In: *Proceedings of the 2003 SIAM International Conference on Data Mining*, pp. 25–36.

Lee, W., McCormick, T. H., Neil, J., and Sodja, C. (2019). "Anomaly Detection in Large Scale Networks with Latent Space Models". In: *arXiv e-prints*. arXiv: 1911.05522 [stat.ME].

Lei, J. (2016). "A goodness-of-fit test for stochastic block models". In: *The Annals of Statistics* 44(1), pp. 401–424.

Lei, J. and Rinaldo, A. (2015). "Consistency of spectral clustering in stochastic block models". In: *Annals of Statistics* 43(1), pp. 215–237.

Levin, K. et al. (2017). "A central limit theorem for an omnibus embedding of multiple random graphs and implications for multiscale network inference". In: *arXiv e-prints*. arXiv: 1705.09355.

Li, T., Levina, E., and Zhu, J. (2020). "Network cross-validation by edge sampling". In: *Biometrika* 107(2), pp. 257–276.

Li, X. et al. (2014). "A Deep Learning Approach to Link Prediction in Dynamic Networks". In: *Proceedings of the 2014 SIAM International Conference on Data Mining*, pp. 289–297.

Li, Z., Ding, B., Han, J., Kays, R., and Nye, P. (2010). "Mining Periodic Behaviors for Moving Objects". In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '10, pp. 1099–1108.

Liao, H.-J., Richard Lin, C.-H., Lin, Y.-C., and Tung, K.-Y. (2013). "Intrusion detection system: A comprehensive review". In: *Journal of Network and Computer Applications* 36(1), pp. 16 –24.

Liben-Nowell, D. and Kleinberg, J. (2007). "The Link-prediction Problem for Social Networks". In: *Journal of the American Society for Information Science and Technology* 58(7), pp. 1019–1031.

Liu, J. (1994). "The Collapsed Gibbs Sampler in Bayesian Computations with Applications to a Gene Regulation Problem". In: *Journal of the American Statistical Association* 89(427), pp. 958–966.

Lü, L. and Zhou, T. (2011). "Link prediction in complex networks: A survey". In: *Physica A: Statistical Mechanics and its Applications* 390(6), pp. 1150 –1170.

Ludkin, M., Eckley, I., and Neal, P. (2018). "Dynamic stochastic block models: parameter estimation and detection of changes in community structure". In: *Statistics and Computing* 28(6), pp. 1201–1213.

Lv, Y. and Zhai, C. X. (2009). "Positional Language Models for Information Retrieval". In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pp. 299–306.

Ma, S., Su, L., and Zhang, Y. (2018). "Determining the Number of Communities in Degree-corrected Stochastic Block Models". In: *arXiv e-prints*. arXiv: 1809.01028.

MacDonald, I. L. and Zucchini, W. (1997). *Hidden Markov and Other Models for Discrete-valued Time Series*. Taylor & Francis.

Malliaros, F. D. and Vazirgiannis, M. (2013). "Clustering and community detection in directed networks: A survey". In: *Physics Reports* 533(4), pp. 95 –142.

Marchette, D. J. (2018). "Discussion of *Statistical methods for network surveillance*". In: *Applied Stochastic Models in Business and Industry* 34(4), pp. 449–451.

Marchette, D. J. (2001). *Computer Intrusion Detection and Network Monitoring. A Statistical Viewpoint*. Statistics for Engineering and Information Science. Springer.

Marchette, D. J. and Priebe, C. E. (2008). "Predicting unobserved links in incompletely observed networks". In: *Computational Statistics & Data Analysis* 52(3), pp. 1373–1386.

Mardia, K. V. (1970). "Measures of multivariate skewness and kurtosis with applications". In: *Biometrika* 57(3), pp. 519–530.

Matias, C. and Miele, V. (2017). "Statistical clustering of temporal networks through a dynamic stochastic block model". In: *Journal of the Royal Statistical Society: Series B* 79(4), pp. 1119–1141.

Maugis, C., Celeux, G., and Martin-Magniette, M. L. (2009). "Variable selection for clustering with Gaussian mixture models". In: *Biometrics* 65(3), pp. 701–709.

McLachlan, G. J. and Krishnan, T. (2007). *The EM Algorithm and Extensions*. Second edition. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc.

McPherson, S. and Ortega, A. (2011). "Detecting low-rate periodic events in Internet traffic using renewal theory". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4336–4339.

Medvedovic, M., Yeung, K. Y., and Bumgarner, R. E. (2004). "Bayesian mixture model based clustering of replicated microarray data". In: *Bioinformatics* 20(8), pp. 1222–1232.

Meeds, E., Ghahramani, Z., Neal, R. M., and Roweis, S. T. (2007). "Modeling Dyadic Data with Binary Latent Factors". In: *Advances in Neural Information Processing Systems 19*. MIT Press, pp. 977–984.

Mele, A., Hao, L., Cape, J., and Priebe, C. E. (2019). "Spectral inference for large Stochastic Blockmodels with nodal covariates". In: *arXiv e-prints*. arXiv: 1908.06438.

Mengersen, K. and Robert, C. (1996). "Testing for mixtures: a Bayesian entropic approach (with discussion)". In: *Bayesian Statistics*. Ed. by J. Berger, J. Bernardo, A. Dawid, D. Lindley, and A. Smith. Oxford University Press.

Menon, A. and Elkan, C. (2011). "Link Prediction via Matrix Factorization". In: *Machine Learning and Knowledge Discovery in Databases: Proceedings of the ECML PKDD 2011*. Springer Berlin Heidelberg, pp. 437–452.

Metelli, S. (2018). "New edge activity and anomaly detection in a large computer network". PhD thesis. Imperial College London.

Metelli, S. and Heard, N. A. (2019). "On Bayesian new edge prediction and anomaly detection in computer networks". In: *Annals of Applied Statistics* 13(4), pp. 2586–2610.

Meyer, C. D. (2000). *Matrix analysis and applied linear algebra*. SIAM.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. (2010). "Recurrent neural network based language model". In: *Proceedings of the 11th Annual Conference of the International Speech Communication Association*. Vol. 9, pp. 1045–1048.

Miller, J. W. and Harrison, M. T. (2018). "Mixture Models With a Prior on the Number of Components". In: *Journal of the American Statistical Association* 113(521), pp. 340–356.

Miller, K., Griffiths, T., and Jordan, M. I. (2010). "Nonparametric latent feature models for link prediction". In: *Advances in Neural Information Processing Systems (NIPS) 22*.

Mu, C. et al. (2020). "On identifying unobserved heterogeneity in stochastic blockmodel graphs with vertex covariates". In: *arXiv e-prints*. arXiv: 2007.02156.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

Nakajima, S., Sugiyama, M., and Tomioka, R. (2010). "Global Analytic Solution for Variational Bayesian Matrix Factorization". In: *Advances in Neural Information Processing Systems 23*. Curran Associates, Inc., pp. 1768–1776.

Navlakha, S., Gitter, A., and Bar-Joseph, Z. (2012). "A Network-based Approach for Predicting Missing Pathway Interactions". In: *PLOS Computational Biology* 8(8), pp. 1–13.

Neal, R. M. (2000). "Markov Chain Sampling Methods for Dirichlet Process Mixture Models". In: *Journal of Computational and Graphical Statistics* 9(2), pp. 249–265.

Neil, J., Hash, C., Brugh, A., Fisk, M., and Storlie, C. B. (2013). "Scan Statistics for the Online Detection of Locally Anomalous Subgraphs". In: *Technometrics* 55(4), pp. 403–414.

Newman, M. (2010). *Networks: An Introduction*. OUP Oxford.

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

Newman, M. E. J. and Reinert, G. (2016). "Estimating the Number of Communities in a Network". In: *Physical Review Letters* 117.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). "On Spectral Clustering: Analysis and an Algorithm". In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. NIPS'01. MIT Press, pp. 849–856.

Nguyen, J. and Zhu, M. (2013). "Content-boosted matrix factorization techniques for recommender systems". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 6(4), pp. 286–301.

Nielsen, A. M. and Witten, D. (2018). "The Multiple Random Dot Product Graph Model". In: *arXiv e-prints*. arXiv: 1811.12172.

Nobile, A. (2004). "On the posterior distribution of the number of components in a finite mixture". In: *Annals of Statistics* 32(5), pp. 2044–2073.

Nowicki, K. and Snijders, T. A. B. (2001). "Estimation and Prediction for Stochastic Blockstructures". In: *Journal of the American Statistical Association* 96(455), pp. 1077–1087.

Papastamoulis, P. and Ntzoufras, I. (2020). "On the identifiability of Bayesian factor analytic models". In: *arXiv e-prints*. arXiv: 2004.05105.

Paquet, U. and Koenigstein, N. (2013). "One-class Collaborative Filtering with Random Graphs". In: *Proceedings of the 22nd International Conference on World Wide Web*. WWW '13. ACM, pp. 999–1008.

Patcha, A. and Park, J.-M. (2007). "An overview of anomaly detection techniques: Existing solutions and latest technological trends". In: *Computer Networks* 51(12), pp. 3448 –3470.

Pearson, K. (1933). "On a method of determining whether a sample of size n supposed to have been drawn from a parent population having a known probability integral has probably been drawn at random". In: *Biometrika* 25(3-4), pp. 379–410.

Peixoto, T. P. (2018). "Bayesian stochastic blockmodeling". In: *Advances in Network Clustering and Blockmodeling*. Ed. by P. Doreian, V. Batagelj, and A. Ferligoj. Wiley.

Peng, L. and Carvalho, L. (2016). "Bayesian degree-corrected stochastic blockmodels for community detection". In: *Electronic Journal of Statistics* 10(2), pp. 2746–2779.

Pensky, M. and Zhang, T. (2019). "Spectral clustering in the dynamic stochastic block model". In: *Electronic Journal of Statistics* 13(1), pp. 678–709.

Percival, D. B. and Walden, A. T. (1993). *Spectral Analysis for Physical Applications*. Cambridge University Press.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). "DeepWalk: Online Learning of Social Representations". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14, pp. 701–710.

Perry, P. O. and Wolfe, P. J. (2013). "Point process modelling for directed interaction networks". In: *Journal of the Royal Statistical Society: Series B* 75(5), pp. 821–849.

Perusquía, J. A., Griffin, J. E., and Villa, C. (2020). "Bayesian Models Applied to Cyber Security Anomaly Detection Problems". In: *arXiv e-prints*. arXiv: 2003.10360.

Pitman, J. (2002). *Combinatorial stochastic processes*. Tech. rep. 621. Department of Statistics University of California at Berkeley.

Pitman, J. and Yor, M. (1997). "The two-parameter Poisson-Dirichlet distribution derived from a stable sub-ordinator". In: *Annals of Probability* 25, pp. 855–900.

Price-Williams, M. and Heard, N. A. (2019). "Nonparametric self-exciting models for computer network traffic". In: *Statistics and Computing* 30, pp. 209–220.

Price-Williams, M., Heard, N. A., and Turcotte, M. J. M. (2017). "Detecting Periodic Subsequences in Cyber Security Data". In: *2017 European Intelligence and Security Informatics Conference (EISIC)*, pp. 84–90.

Priebe, C. E., Conroy, J. M., Marchette, D. J., and Park, Y. (2005). "Scan Statistics on Enron Graphs". In: *Computational & Mathematical Organization Theory* 11(3), pp. 229–247.

Priebe, C. E. et al. (2019). "On a two-truths phenomenon in spectral graph clustering". In: *Proceedings of the National Academy of Sciences* 116(13), pp. 5995–6000.

Qiao, Y., Yang, Y., He, J., Liu, B., and Zeng, Y. (2012). "Detecting Parasite P2P Botnet in eMule-like Networks through Quasi-periodicity Recognition". In: *Information Security and Cryptology - ICISC 2011*. Springer Berlin Heidelberg, pp. 127–139.

Qiao, Y., Yang, Y.-X., He, J., Tang, C., and Zeng, Y.-Z. (2013). "Detecting P2P bots by mining the regional periodicity". In: *Journal of Zhejiang University Science C* 14(9), pp. 682–700.

Qin, T. and Rohe, K. (2013). "Regularized Spectral Clustering under the Degree-Corrected Stochastic Blockmodel". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems*. Vol. 2. NIPS'13. Curran Associates Inc., pp. 3120–3128.

Raftery, A. E. and Dean, N. (2006). "Variable Selection for Model-Based Clustering". In: *Journal of the American Statistical Association* 101(473), pp. 168–178.

Raftery, A. E., Niu, X., Hoff, P. D., and Yeung, K. Y. (2012). "Fast Inference for the Latent Space Network Model Using a Case-Control Approximate Likelihood". In: *Journal of Computational and Graphical Statistics* 21(4), pp. 901–919.

Ranshous, S. et al. (2015). "Anomaly detection in dynamic networks: a survey". In: *WIREs Computational Statistics* 7(3), pp. 223–247.

Richardson, S. and Green, P. J. (1997). "On Bayesian Analysis of Mixtures with an Unknown Number of Components". In: *Journal of the Royal Statistical Society: Series B* 59(4), pp. 731–792.

Rohe, K., Qin, T., and Yu, B. (2016). "Co-clustering directed graphs to discover asymmetries and directional communities". In: *Proceedings of the National Academy of Sciences* 113(45), pp. 12679–12684.

Rohe, K., Chatterjee, S., and Yu, B. (2011). "Spectral clustering and the high-dimensional stochastic blockmodel". In: *Annals of Statistics* 39(4), pp. 1878–1915.

Rosenfeld, R. (1996). "A maximum entropy approach to adaptive statistical language modelling". In: *Computer Speech & Language* 10(3), pp. 187 –228.

Roy, S., Atchadé, Y., and Michailidis, G. (2019). "Likelihood Inference for Large Scale Stochastic Blockmodels With Covariates Based on a Divide-and-Conquer Parallelizable Algorithm With Communication". In: *Journal of Computational and Graphical Statistics* 28(3), pp. 609–619.

Rubin-Delanchy, P., Adams, N. M., and Heard, N. A. (2016). "Disassortativity of computer networks". In: *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pp. 243–247.

Rubin-Delanchy, P., Heard, N. A., and Lawson, D. J. (2018). "Meta analysis of mid-p-values: some new results based on the convex order". In: *Journal of the American Statistical Association*.

Rubin-Delanchy, P., Cape, J., Tang, M., and Priebe, C. E. (2017). "A statistical interpretation of spectral embedding: the generalised random dot product graph". In: *arXiv e-prints*. arXiv: 1709.05506.

Salakhutdinov, R. and Mnih, A. (2007). "Probabilistic Matrix Factorization". In: *Proceedings of the 20th International Conference on Neural Information Processing Systems*. NIPS'07, pp. 1257–1264.

Saldaña, D. F., Yu, Y., and Feng, Y. (2017). "How Many Communities Are There?" In: *Journal of Computational and Graphical Statistics* 26(1), pp. 171–181.

Salter-Townshend, M. and Murphy, T. B. (2013). "Variational Bayesian inference for the Latent Position Cluster Model for network data". In: *Computational Statistics & Data Analysis* 57(1), pp. 661–671.

Sanna Passino, F. and Heard, N. A. (2019). "Modelling dynamic network evolution as a Pitman-Yor process". In: *Foundations of Data Science* 1(3), pp. 293–306.

Sanna Passino, F. and Heard, N. A. (2020a). "Bayesian estimation of the latent dimension and communities in stochastic blockmodels". In: *Statistics and Computing* 30(5), pp. 1291–1307.

Sanna Passino, F. and Heard, N. A. (2020b). "Classification of periodic arrivals in event time data for filtering computer network traffic". In: *Statistics and Computing* 30(5), pp. 1241–1254.

Sanna Passino, F., Heard, N. A., and Rubin-Delanchy, P. (2020). "Spectral clustering on spherical coordinates under the degree-corrected stochastic blockmodel". In: *arXiv e-prints*. arXiv: 2011.04558.

Sanna Passino, F., Turcotte, M. J. M., and Heard, N. A. (2020). "Graph link prediction in computer networks using Poisson matrix factorisation". In: *arXiv e-prints*. arXiv: 2001.09456.

Sanna Passino, F., Bertiger, A. S., Neil, J. C., and Heard, N. A. (2020). "Link prediction in dynamic networks using random dot product graphs". In: *arXiv e-prints*. arXiv: 1912.10419.

Sarkar, P., Chakrabarti, D., and Jordan, M. (2014). "Nonparametric link prediction in large scale dynamic networks". In: *Electronic Journal of Statistics* 8(2), pp. 2022–2065.

Sarkar, P. and Moore, A. W. (2006). "Dynamic Social Network Analysis using Latent Space Models". In: *Advances in Neural Information Processing Systems 18*, pp. 1145–1152.

Schein, A., Paisley, J., Blei, D. M., and Wallach, H. (2015). "Bayesian Poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1045–1054.

Schein, A., Zhou, M., Blei, D. M., and Wallach, H. (2016). "Bayesian Poisson Tucker decomposition for learning the structure of international relations". In: *Proceedings of the 33rd International Conference on Machine Learning*.

Scheinerman, E. R. and Tucker, K. (2010). "Modeling graphs using dot product representations". In: *Computational Statistics* 25(1), pp. 1–16.

Schönemann, P. H. (1966). "A generalized solution of the orthogonal Procrustes problem". In: *Psychometrika* 31(1), pp. 1–10.

Schwarz, G. (1978). "Estimating the Dimension of a Model". In: *Annals of Statistics* 6(2), pp. 461–464.

Scott, S. and Smyth, P. (2003). "The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic modeling". In: *Bayesian Statistics 7*. Ed. by J. M. Bernardo et al. Oxford University Press.

Seeger, M. and Bouchard, G. (2012). "Fast variational Bayesian inference for non-conjugate matrix factorization models". In: *Artificial Intelligence and Statistics*, pp. 1012–1018.

Sengupta, S. and Chen, Y. (2018). "A block model for node popularity in networks with community structure". In: *Journal of the Royal Statistical Society: Series B* 80(2), pp. 365–386.

Sethuraman, J. (1994). "A constructive definition of Dirichlet priors". In: *Statistica Sinica* 4, pp. 639–650.

Sewell, D. K. and Chen, Y. (2015). "Latent Space Models for Dynamic Networks". In: *Journal of the American Statistical Association* 110(512), pp. 1646–1657.

Sewell, D. K. and Chen, Y. (2017). "Latent Space Approaches to Community Detection in Dynamic Networks". In: *Bayesian Analysis* 12(2), pp. 351–377.

Sharan, U. and Neville, J. (2008). "Temporal-Relational Classifiers for Prediction in Evolving Domains". In: *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pp. 540–549.

Shiga, M. and Mamitsuka, H. (2012). "A Variational Bayesian Framework for Clustering with Multiple Graphs". In: *IEEE Transactions on Knowledge and Data Engineering* 24(4), pp. 577–590.

Siegel, A. F. (1980). "Testing for Periodicity in a Time Series". In: *Journal of the American Statistical Association* 75(370), pp. 345–348.

Silva, E. d. S. da, Langseth, H., and Ramampiaro, H. (2017). "Content-Based Social Recommendation with Poisson Matrix Factorization". In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, pp. 530–546.

Silverman, B. W. (1986). *Density Estimation*. London: Chapman and Hall.

Singh, A. P. and Gordon, G. J. (2008). "Relational Learning via Collective Matrix Factorization". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08, pp. 650–658.

Snijders, T. A. B. and Nowicki, K. (1997). "Estimation and Prediction for Stochastic Blockmodels for Graphs with Latent Block Structure". In: *Journal of Classification* 14(1), pp. 75–100.

Sperotto, A. et al. (2010). "An Overview of IP Flow-Based Intrusion Detection". In: *IEEE Communications Surveys Tutorials* 12(3), pp. 343–356.

Stephens, M. (2000). “Bayesian analysis of mixture models with an unknown number of components–an alternative to reversible jump methods”. In: *Annals of Statistics* 28(1), pp. 40–74.

Stouffer, S. A., Suchman, E. A., DeVinney, L. C., Star, S. A., and Williams, R. M. (1949). *The American Soldier. Adjustment During Army Life*. Princeton University Press: Princeton, New Jersey.

Sussman, D. L., Tang, M., and Priebe, C. E. (2014). “Consistent Latent Position Estimation and Vertex Classification for Random Dot Product Graphs”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36(1), pp. 48–57.

Sussman, D. L., Minh, T., Fishkind, D. E., and Priebe, C. E. (2012). “A Consistent Adjacency Spectral Embedding for Stochastic Blockmodel Graphs”. In: *Journal of the American Statistical Association* 107(499), pp. 1119–1128.

Tadesse, M. G., Sha, N., and Vannucci, M. (2005). “Bayesian Variable Selection in Clustering High-Dimensional Data”. In: *Journal of the American Statistical Association* 100(470), pp. 602–617.

Tang, J. et al. (2015). “LINE: Large-Scale Information Network Embedding”. In: *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077.

Tang, M. and Priebe, C. E. (2018). “Limit theorems for eigenvectors of the normalized Laplacian for random graphs”. In: *Annals of Statistics* 46(5), pp. 2360–2415.

Tang, M., Sussman, D. L., and Priebe, C. E. (2013). “Universally consistent vertex classification for latent positions graphs”. In: *Annals of Statistics* 41(3), pp. 1406–1430.

Tang, W., Lu, Z., and Dhillon, I. S. (2009). “Clustering with Multiple Graphs”. In: *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*. ICDM ’09, pp. 1016–1021.

Teh, W. Y. (2006). “A hierarchical Bayesian language model based on Pitman-Yor processes”. In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association of Computational Linguistics*, pp. 985–992.

Thibaux, R. and Jordan, M. I. (2007). “Hierarchical Beta Processes and the Indian Buffet Process”. In: *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*. Vol. 2, pp. 564–571.

Tippett, L. H. C. (1931). *The Methods of Statistics*. London: Williams and Norgate.

Tong, H., Faloutsos, C., and Pan, J.-Y. (2006). “Fast Random Walk with Restart and Its Applications”. In: *Proceedings of the Sixth International Conference on Data Mining*. ICDM ’06, pp. 613–622.

Turcotte, M., Heard, N. A., and Neil, J. (2014). “Detecting Localised Anomalous Behaviour in a Computer Network”. In: *Advances in Intelligent Data Analysis XIII*. Springer, pp. 321–332.

Turcotte, M., Moore, J., Heard, N. A., and McPhall, A. (2016). “Poisson factorization for peer-based anomaly detection”. In: *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pp. 208–210.

Turcotte, M. J. M., Kent, A. D., and Hash, C. (2018). "Unified Host and Network Data Set". In: *Data Science for Cyber-Security*. World Scientific. Chap. 1, pp. 1–22.

van der Pas, S. L. and van der Vaart, A. W. (2018). "Bayesian Community Detection". In: *Bayesian Analysis* 13(3), pp. 767–796.

Verizon (2020). *2020 Data Breach Investigations Report*. Tech. rep. Verizon Enterprises.

von Luxburg, U. (2007). "A tutorial on spectral clustering". In: *Statistics and Computing* 17(4), pp. 395–416.

Wallach, S. M., Jensen, S. T., Dicker, L., and Heller, K. A. (2010). "An Alternative Prior Process for Nonparametric Bayesian Clustering". In: *Proceedings of the Thireenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, pp. 892–899.

Wang, S., Arroyo-Relión, J. D., Vogelstein, J. T., and Priebe, C. E. (2019). "Joint Embedding of Graphs". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (to appear).

Wang, W. J. and Wong, G. Y. (1987). "Stochastic Blockmodels for Directed Graphs". In: *Journal of the American Statistical Association* 82(397), pp. 8–19.

West, M., Müller, P., and Escobar, M. D. (1994). "Hierarchical Priors and Mixture Models, with Applications in Regression and Density Estimation". In: *Aspects of Uncertainty: A Tribute to D. V. Lindley*, pp. 363–386.

Witten, D. M. and Tibshirani, R. (2010). "A Framework for Feature Selection in Clustering". In: *Journal of the American Statistical Association* 105(490), pp. 713–726.

Wolfe, P. J. and Olhede, S. C. (2013). "Nonparametric graphon estimation". In: *arXiv e-prints*. arXiv: 1309.5936.

Wu, Z. et al. (2020). "A Comprehensive Survey on Graph Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21.

Wyse, J. and Friel, N. (2012). "Block clustering with collapsed latent block models". In: *Statistics and Computing* 22(2), pp. 415–428.

Wyse, J., Friel, N., and Latouche, P. (2017). "Inferring structure in bipartite networks using the latent blockmodel and exact ICL". In: *Network Science* 5(1), pp. 45–69.

Xing, E. P., Fu, W., and Song, L. (2010). "A state-space mixed membership blockmodel for dynamic network tomography". In: *Annals of Applied Statistics* 4(2), pp. 535–566.

Xu, K. S. and Hero III, A. O. (2013). "Dynamic Stochastic Blockmodels: Statistical Models for Time-Evolving Networks". In: *Social Computing, Behavioral-Cultural Modeling and Prediction*. Springer Berlin Heidelberg, pp. 201–210.

Xu, K. S. and Hero III, A. O. (2014). "Dynamic Stochastic Blockmodels for Time-Evolving Social Networks". In: *IEEE Journal of Selected Topics in Signal Processing* 8(4), pp. 552–562.

Yan, X. et al. (2014). "Model selection for degree-corrected block models". In: *Journal of Statistical Mechanics: Theory and Experiment* 5.

Yang, C., Priebe, C. E., Park, Y., and Marchette, D. J. (2020). "Simultaneous dimensionality and complexity model selection for spectral graph clustering". In: *Journal of Computational and Graphical Statistics* (to appear).

Young, S. J. and Scheinerman, E. R. (2007). "Random Dot Product Graph Models for Social Networks". In: *Algorithms and Models for the Web-Graph*. Springer, pp. 138–149.

Yu, W., Aggarwal, C. C., and Wang, W. (2017). "Temporally Factorized Network Modeling for Evolutionary Network Analysis". In: *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 455–464.

Yu, W., Cheng, W., Aggarwal, C. C., Chen, H., and Wang, W. (2017). "Link Prediction with Spatial and Temporal Consistency in Dynamic Networks". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 3343–3349.

Zhang, M. and Chen, Y. (2018). "Link Prediction Based on Graph Neural Networks". In: *Advances in Neural Information Processing Systems 31*, pp. 5165–5175.

Zhang, W. and Wang, J. (2015). "A Collective Bayesian Poisson Factorization Model for Cold-start Local Event Recommendation". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15, pp. 1455–1464.

Zhao, Y., Levina, E., and Zhu, J. (2011). "Community extraction for social networks". In: *Proceedings of the National Academy of Sciences* 108(18), pp. 7321–7326.

Zhao, Y., Levina, E., and Zhu, J. (2012). "Consistency of community detection in networks under degree-corrected stochastic block models". In: *Annals of Statistics* 40(4), pp. 2266–2292.

Zheng, Q. and Skillicorn, D. B. (2015). "Spectral embedding of directed networks". In: *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 432–439.

Zhou, M. (2015). "Infinite Edge Partition Models for Overlapping Community Detection and Link Prediction". In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS*.

Zhu, L., Guo, D., Yin, J., Steeg, G. V., and Galstyan, A. (2016). "Scalable Temporal Latent Space Inference for Link Prediction in Dynamic Social Networks". In: *IEEE Transactions on Knowledge and Data Engineering* 28(10), pp. 2765–2777.

Zhu, M. and Ghodsi, A. (2006). "Automatic dimensionality selection from the scree plot via the use of profile likelihood". In: *Computational Statistics & Data Analysis* 51(2), pp. 918 –930.

A

Parametrisation of the probability distributions used in this thesis

This section reports the density functions of some of the distributions used in this thesis.

i. Multivariate normal – $\mathbf{X} \sim \mathbb{N}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$, $\boldsymbol{\Sigma} \succ 0$:

$$f(\boldsymbol{x}) = (2\pi)^{-m/2} \det(\boldsymbol{\Sigma})^{-1/2} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right\},$$

where $\det(\cdot)$ is the determinant and $\boldsymbol{\Sigma} \succ 0$ denotes a positive definite matrix.

ii. Multivariate Student's t – $\mathbf{X} \sim t_\nu(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$, $\boldsymbol{\Sigma} \succ 0$, $\nu > 0$:

$$f(\boldsymbol{x}) = \frac{\Gamma(\nu/2 + m/2)}{\Gamma(\nu/2)} \frac{\det(\boldsymbol{\Sigma})^{-1/2}}{(\nu\pi)^{m/2}} \left(1 + \frac{1}{\nu} (\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right)^{-(\nu+m)/2}.$$

iii. Inverse Wishart – $\mathbf{X} \sim \text{IW}_m(\nu, \boldsymbol{\Psi})$, $\nu \geq m$, $\boldsymbol{\Psi} \in \mathbb{R}^{m \times m}$, $\boldsymbol{\Psi} \succ 0$:

$$f(\boldsymbol{x}) = \frac{\det(\boldsymbol{\Psi})^{\nu/2}}{2^{\nu m/2} \Gamma_m(\nu/2)} \det(\mathbf{X})^{-(\nu+m+1)/2} \exp\left\{ -\frac{1}{2} \text{tr}(\boldsymbol{\Psi} \mathbf{X}^{-1}) \right\},$$

where $\Gamma_m(\nu/2) = \pi^{m(m-1)/4} \prod_{i=1}^m \Gamma\{(\nu + 1 - i)/2\}$, and $\text{tr}(\cdot)$ is the trace.

iv. Normal inverse-Wishart – $(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \sim \text{NIW}_m(\boldsymbol{\mu}_0, \kappa, \nu, \boldsymbol{\Psi}) = \mathbb{N}_m(\boldsymbol{\mu} | \boldsymbol{\mu}_0, \boldsymbol{\Sigma}/\kappa) \text{IW}_m(\boldsymbol{\Sigma} | \nu, \boldsymbol{\Psi})$.

v. Inverse gamma – $X \sim \text{IG}(\alpha, \beta)$, $\alpha, \beta > 0$:

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} (1/x)^{\alpha+1} \exp(-\beta/x).$$

In this work, the inverse χ^2 distribution is also used: $X \sim \text{Inv-}\chi^2(\lambda, \sigma^2) \equiv \text{IG}(\lambda/2, \lambda\sigma^2/2)$.

vi. Normal-inverse gamma – $(\mu, \sigma^2) \sim \text{NIG}(\mu_0, \lambda, \alpha, \beta) = \mathbb{N}(\mu | \mu_0, \sigma^2/\lambda) \text{IG}(\sigma^2 | \alpha, \beta)$.

B

Procrustes alignment of embeddings

As discussed in Section 7.1, for prediction of the future latent positions based on individual embeddings $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_T$, it is first necessary to align the embeddings. This section discusses a popular method for aligning two matrices: Procrustes analysis (Dryden and Mardia, 2016) and its generalisation to T matrices (Gower, 1975). The alignment step is required because the latent positions $\mathbf{X}_t$ of a single graph are not identifiable up to orthogonal transformations, which leave the inner product unchanged: for an orthogonal matrix $\mathbf{Q}_t$, $(\mathbf{X}_t\mathbf{Q}_t)(\mathbf{X}_t\mathbf{Q}_t)^\top = \mathbf{X}_t\mathbf{X}_t^\top$. Therefore $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_T$ only represent estimates of a rotation of the embeddings. Given two embeddings $\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2 \in \mathbb{R}^{n\times d}$, Procrustes analysis aims to find the optimal rotation of $\hat{\mathbf{X}}_2$ on $\hat{\mathbf{X}}_1$. The following minimisation criterion is utilised:

$$\min_{\mathbf{Q}} \left\| \hat{\mathbf{X}}_1 - \hat{\mathbf{X}}_2\mathbf{Q} \right\|_{\mathrm{F}}, \tag{B.1}$$

where $\mathbf{Q}$ is an orthogonal matrix, and $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. The solution of the minimisation problem has been derived in Schönemann (1966), and is based on the SVD decomposition $\hat{\mathbf{X}}_2^\top\hat{\mathbf{X}}_1 = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top$. The solution is $\mathbf{Q}^\star = \tilde{\mathbf{U}}\tilde{\mathbf{V}}^\top$, and it follows that the optimal rotation of $\hat{\mathbf{X}}_2$ onto $\hat{\mathbf{X}}_1$ is $\hat{\mathbf{X}}_2\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\top$.

Similarly, a set of T embeddings $\hat{\mathbf{X}}_t \in \mathbb{R}^{n\times r},\ t = 1, \ldots, T$ can be superimposed using *generalised Procrustes analysis* (GPA, Gower, 1975), which uses the minimisation criterion:

$$\min_{\mathbf{Q}_j} \sum_{j=1}^{T} \left\| \hat{\mathbf{X}}_j\mathbf{Q}_j - \tilde{\mathbf{X}} \right\|_{\mathrm{F}}^2 \text{ s.t. } \sum_{j=1}^{T} \mathrm{S}^2(\hat{\mathbf{X}}_j) = \sum_{j=1}^{T} \mathrm{S}^2(\hat{\mathbf{X}}_j\mathbf{Q}_j), \tag{B.2}$$

where, similarly to (B.1), $\mathbf{Q}_j$ are orthogonal matrices. Additionally, $\tilde{\mathbf{X}} \in \mathbb{R}^{n\times r}$ is a reference matrix, shared across the T matrices, and $\mathrm{S}(\cdot)$ is the *centroid size* $\mathrm{S}(\mathbf{M}) = \|(\mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top)\mathbf{M}\|_{\mathrm{F}}$.

The GPA algorithm solves (B.2) by iterating standard Procrustes analysis (Dryden and Mardia, 2016), after a suitable initialisation of the reference matrix:

i. *update* the embeddings $\hat{\mathbf{X}}_t$, performing a standard Procrustes superimposition of each $\hat{\mathbf{X}}_j$ on $\tilde{\mathbf{X}}$:
$$\hat{\mathbf{X}}_j \leftarrow \hat{\mathbf{X}}_j \tilde{\mathbf{U}}_j \tilde{\mathbf{V}}_j^\top,$$
where $\hat{\mathbf{X}}_j^\top \tilde{\mathbf{X}} = \tilde{\mathbf{U}}_j \tilde{\mathbf{D}}_j \tilde{\mathbf{V}}_j^\top$,

ii. *update* the reference embedding: $\tilde{\mathbf{X}} = \sum_{t=1}^{T} \hat{\mathbf{X}}_t$,

iii. *repeat* steps 1 and 2 until the difference between two consecutive values of (B.2) is within a tolerance η.

The final value of the reference embedding $\tilde{\mathbf{X}}$ can be interpreted as the average of rotations of the initial embeddings. The alignment step increases the computational cost of the operations described in Section 7.1 by a factor of $\mathcal{O}(nd^2)$, caused by the repeated SVD decompositions and matrix multiplications in the GPA algorithm.

When $d_- \neq 0$ in the GRDPG setting, the problem is known as *indefinite Procrustes problem*, and does not have closed form solution (Kintzel, 2005). In this setting, the criterion (B.1) must be optimised numerically for $\mathbf{Q} \in \mathbb{O}(d_+, d_-)$, the indefinite orthogonal group with signature (d_+, d_-). The optimisation routine could be applied iteratively in the GPA algorithm to obtain an *indefinite GPA*.

On the other hand, for directed and bipartite graphs, the criteria (B.1) and (B.2) must be optimised jointly for the two embeddings obtained using DASE in Definition 4. For two embeddings $(\hat{\mathbf{X}}_1, \hat{\mathbf{Y}}_1)$ and $(\hat{\mathbf{X}}_2, \hat{\mathbf{Y}}_2)$ obtained from a directed graph, the solution is simply given by aligning the stacked matrices $[\hat{\mathbf{X}}_1^\top, \hat{\mathbf{Y}}_1^\top]^\top \in \mathbb{R}^{2n \times r}$ and $[\hat{\mathbf{X}}_2^\top, \hat{\mathbf{Y}}_2^\top]^\top \in \mathbb{R}^{2n \times r}$ using (B.1). The procedure could be also iterated for more than two embeddings to obtain a *joint generalised Procrustes algorithm*.

C

Inference in the extended, seasonal and joint PMF models

C.1. Full conditional distributions in the extended PMF model

First note that, conditional on N_{ij}, $\mathbf{Z}_{ij} = (Z_{ij1}, \ldots, Z_{ij(d+KH)})$ has a multinomial distribution,

$$\mathbf{Z}_{ij}|N_{ij}, \boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{\Phi} \sim \text{Multinomial}\left(N_{ij}, \boldsymbol{\pi}_{ij}\right),$$

where $\boldsymbol{\pi}_{ij}$ is the probability vector proportional to

$$(x_{i1}y_{j1}, \ldots, x_{iR}y_{jR}, \phi_{11}x_{i1}y_{j1}, \ldots, \phi_{KH}x_{iK}y_{jH}).$$

The result can be obtained using Bayes' theorem on $p(\mathbf{Z}_{ij}|N_{ij}, \boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{\Phi})$. Therefore:

$$p(N_{ij}, \mathbf{Z}_{ij}|\boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{\Phi}, \mathbf{A}) = \begin{cases} \text{Poisson}_+(\psi_{ij})\text{Multinomial}(N_{ij}, \boldsymbol{\pi}_{ij}) & A_{ij} > 0, \\ \delta_0(N_{ij})\delta_{\mathbf{0}}(\mathbf{Z}_{ij}) & A_{ij} = 0, \end{cases} \tag{C.1}$$

where $\text{Poisson}_+(\cdot)$ denotes the zero-truncated Poisson distribution, and $\psi_{ij} = \boldsymbol{x}_i^\top \boldsymbol{y}_j + \mathbf{1}_K^\top(\boldsymbol{\Phi} \odot \boldsymbol{u}_i\boldsymbol{v}_j^\top)\mathbf{1}_H$. The user and host latent features complete conditionals are gamma, where

$$x_{ir}|\boldsymbol{y}, \zeta_i^{(x)}, \mathbf{Z} \sim \text{Gamma}\left(a^{(x)} + \sum_{j=1}^{|V_2|} Z_{ijr}, \zeta_i^{(x)} + \sum_{j=1}^{|V_2|} y_{jr}\right),$$
$$y_{jr}|\boldsymbol{x}, \zeta_j^{(y)}, \mathbf{Z} \sim \text{Gamma}\left(a^{(y)} + \sum_{i=1}^{|V_1|} Z_{ijr}, \zeta_j^{(y)} + \sum_{i=1}^{|V_1|} x_{ir}\right),$$

and

$$\zeta_i^{(x)}|\boldsymbol{x}_i \sim \text{Gamma}\left(b^{(x)} + da^{(x)}, c^{(x)} + \sum_{r=1}^{d} x_{ir}\right),$$
$$\zeta_j^{(y)}|\boldsymbol{y}_j \sim \text{Gamma}\left(b^{(y)} + da^{(y)}, c^{(y)} + \sum_{r=1}^{d} y_{jr}\right). \tag{C.2}$$

Similarly,

$$\phi_{kh}|\zeta^{(\phi)}, \mathbf{Z} \sim \text{Gamma}\left(a^{(\phi)} + \sum_{i=1}^{|V_1|}\sum_{j=1}^{|V_2|} Z_{ijl},\ \zeta^{(\phi)} + \sum_{i=1}^{|V_1|} u_{ik}\sum_{j=1}^{|V_2|} v_{jh}\right),$$
$$\zeta^{(\phi)}|\boldsymbol{\Phi} \sim \text{Gamma}\left(b^{(\phi)} + KHa^{(\phi)},\ c^{(\phi)} + \sum_{k=1}^{K}\sum_{h=1}^{H}\phi_{kh}\right),$$

where l is the index corresponding to the covariate pair (k, h).

C.2. Variational inference in the extended PMF model

As all of the factors in the variational approximation given in (8.6) take the same distributional form of the complete conditionals,

$$q(N_{ij}, \mathbf{Z}_{ij}|\theta_{ij}, \boldsymbol{\chi}_{ij}) = \begin{cases} \text{Poisson}_+(\theta_{ij})\text{Multinomial}(N_{ij}, \boldsymbol{\chi}_{ij}) & A_{ij} > 0, \\ \delta_0(N_{ij})\delta_{\mathbf{0}}(\mathbf{Z}_{ij}) & A_{ij} = 0. \end{cases} \tag{C.3}$$

Let ψ_{ij} denote the rate $\sum_{r=1}^{d} x_{ir}y_{jr} + \sum_{k=1}^{K}\sum_{h=1}^{H}\phi_{kh}u_{ik}v_{jh}$ of the Poisson distribution for N_{ij}, and $\psi_{ijl},\ l = 1, \ldots, d + KH$, represent the individual elements in the sum. Equation (8.7) gives the updates for $\boldsymbol{\theta}$ and $\boldsymbol{\chi}$ for $A_{ij} > 0$:

$$\mathbb{E}^q_{-N_{ij},\mathbf{Z}_{ij}}\left\{\log p(N_{ij}, \mathbf{Z}_{ij}|\boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{\Phi})\right\} = \sum_l \left\{Z_{ijl}\mathbb{E}^q_{-N_{ij},\mathbf{Z}_{ij}}\left(\log\psi_{ijl}\right) - \log(Z_{ijl}!)\right\} + k, \tag{C.4}$$

where k is a constant with respect to N_{ij} and $\mathbf{Z}_{ij}$. Hence:

$$q^\star(N_{ij}, \mathbf{Z}_{ij}) \propto \prod_{l=1}^{d+KH} \frac{1}{Z_{ijl}!}\exp\left\{\mathbb{E}^q_{-N_{ij},\mathbf{Z}_{ij}}(\log\psi_{ijl})\right\}^{Z_{ijl}}.$$

with domain of $\mathbf{Z}_{ij}$ constrained to have $\sum_l Z_{ijl} > 0$. Multiplying and dividing the expression by $N_{ij}!$ and $[\sum_l \exp\{\mathbb{E}^q_{-N_{ij},\mathbf{Z}_{ij}}(\log\psi_{ijl})\}]^{N_{ijl}}$ gives a distribution which has the same form of (C.3). Therefore the rate θ_{ij} of the zero truncated Poisson is updated using $\sum_l \exp\{\mathbb{E}^q_{-N_{ij},\mathbf{Z}_{ij}}(\log\psi_{ijl})\}$, see step 5 in Algorithm 8.1 for the resulting final expression. The update for the vector of

probabilities $\boldsymbol{\chi}_{ij}$ is given by a slight extension of the standard result for variational inference in the PMF model (Gopalan, Hofman, and Blei, 2015) to include the covariate terms, see step 6 of Algorithm 8.1. The remaining updates are essentially analogous to standard PMF (Gopalan, Hofman, and Blei, 2015), and use the fact that, if $X \sim \text{Gamma}(\alpha, \beta)$, then $\mathbb{E}(\log X) = \Psi(\alpha) - \log(\beta)$, where $\Psi(\cdot)$ is the digamma function.

C.3. Variational inference in the seasonal PMF model

The inferential procedure for the seasonal model follows the same guidelines used for the non-seasonal model. Given the unobserved count N_{ijt}, latent variables Z_{ijtl} are added, representing the contribution of the component l to the total count N_{ijt}: $N_{ijt} = \sum_l Z_{ijtl}$. The full conditional for N_{ijt} and Z_{ijt} follows (C.1), except the rate for the Poisson and probability vectors for the multinomial will now depend on the seasonal parameters γ_{itr}, δ_{jtr}, and ω_{kht}. Letting p denote a seasonal segment in $\{1, \ldots, s\}$ the full conditionals for the rate parameters are:

$$
\begin{aligned}
x_{ir}|\mathbf{Z}, \mathbf{Y}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \zeta_i^{(x)} &\sim \text{Gamma}\left(a^{(x)} + \sum_{j=1}^{|V_2|}\sum_{t=1}^{T} Z_{ijtr}, \zeta_i^{(x)} + \sum_{t=1}^{T}\gamma_{itr}\sum_{j=1}^{|V_2|} y_{jr}\delta_{jtr}\right), \\
\gamma_{ipr}|\mathbf{Z}, \mathbf{X}, \mathbf{Y}, \boldsymbol{\delta}, \zeta_p^{(\gamma)} &\sim \text{Gamma}\left(a^{(\gamma)} + \sum_{j=1}^{|V_2|}\sum_{t:t=p} Z_{ijtr}, \zeta_p^{(\gamma)} + x_{ir}\sum_{j=1}^{|V_2|} y_{jr}\sum_{t:t=p}\delta_{jtr}\right), \\
\phi_{kh}|\mathbf{Z}, \zeta^{(\phi)} &\sim \text{Gamma}\left(a^{(\phi)} + \sum_{i=1}^{|V_1|}\sum_{j=1}^{|V_2|}\sum_{t=1}^{T} Z_{ijtl}, \zeta^{(\phi)} + T\tilde{u}_k\tilde{v}_h\right),
\end{aligned}
$$

where $\tilde{u}_k = \sum_{i=1}^{|V_1|} u_{ik}$ and $\tilde{v}_h = \sum_{j=1}^{|V_2|} v_{jh}$. Similar results are available for y_{jr} and δ_{jpr}. Also:

$$
\zeta_p^{(\gamma)}|\boldsymbol{\gamma} \sim \text{Gamma}\left(b^{(\gamma)} + |V_1|da^{(\gamma)}, c^{(\gamma)} + \sum_{i=1}^{|V_1|}\sum_{r=1}^{d}\gamma_{ipr}\right),
$$

and similarly for $\zeta_p^{(\delta)}$. For $\zeta_i^{(x)}$ and $\zeta_j^{(y)}$, the conditional distribution is equivalent to (C.2). The mean-field variational family is again used implying a factorisation similar to (8.6), so that

$$
\begin{aligned}
q(\mathbf{X}, \mathbf{Y}, \boldsymbol{\Phi}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \mathbf{N}, \mathbf{Z}) = &\prod_{i,j,t} q(N_{ijt}, \mathbf{Z}_{ijt}|\theta_{ijt}, \boldsymbol{\chi}_{ijt}) \prod_{i,r} q(x_{ir}|\lambda_{ir}^{(x)}, \mu_{ir}^{(x)}) \prod_{j,r} q(y_{jr}|\lambda_{jr}^{(y)}, \mu_{jr}^{(y)}) \\
&\times \prod_{k,h} q(\phi_{kh}|\lambda_{kh}^{(\phi)}, \mu_{kh}^{(\phi)}) \prod_{i} q(\zeta_i^{(x)}|\nu_i^{(x)}, \xi_i^{(x)}) \prod_{j} q(\zeta_j^{(y)}|\nu_j^{(y)}, \xi_j^{(y)}) q(\zeta^{(\phi)}|\nu^{(\phi)}, \xi^{(\phi)}) \\
&\times \prod_{i,q,r} q(\gamma_{ipr}|\lambda_{ipr}^{(\gamma)}, \mu_{ipr}^{(\gamma)}) \prod_{j,p,r} q(\delta_{jpr}|\lambda_{jpr}^{(\delta)}, \mu_{jpr}^{(\delta)}) \prod_{p} q(\zeta_p^{(\gamma)}|\nu_p^{(\gamma)}, \xi_p^{(\gamma)}) q(\zeta_p^{(\delta)}|\nu_p^{(\delta)}, \xi_p^{(\delta)}).
\end{aligned}
$$

As in Section 8.2.2, each $q(\cdot)$ has the same form of the full conditional distributions for the corresponding parameter or group of parameters. Again the variational parameters are updated using CAVI and a similar algorithm is obtained to that detailed in Algorithm 8.1, where steps 7, 8, 9 and 10 are modified to include the time dependent parameters. It follows that for the user-specific parameters the update equations take the form:

$$\lambda_{ir}^{(x)} = a^{(x)} + \sum_{j=1}^{|V_2|}\sum_{t=1}^{T} \frac{A_{ijt}\theta_{ijt}\chi_{ijtr}}{1-e^{-\theta_{ijt}}},\ \mu_{ir}^{(x)} = \frac{\nu_i^{(x)}}{\xi_i^{(x)}} + \sum_{t=1}^{T}\frac{\lambda_{itr}^{(\gamma)}}{\mu_{itr}^{(\gamma)}}\sum_{j=1}^{|V_2|}\frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}}\frac{\lambda_{jtr}^{(\delta)}}{\mu_{jtr}^{(\delta)}},$$
$$\lambda_{ipr}^{(\gamma)} = a^{(\gamma)} + \sum_{j=1}^{|V_2|}\sum_{t:t=p} \frac{A_{ijt}\theta_{ijt}\chi_{ijtr}}{1-e^{-\theta_{ijt}}},\ \mu_{ipr}^{(\gamma)} = \frac{\nu_p^{(\gamma)}}{\xi_p^{(\gamma)}} + \frac{\lambda_{ir}^{(x)}}{\mu_{ir}^{(x)}}\sum_{j=1}^{|V_2|}\frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}}\sum_{t:t=p}\frac{\lambda_{jtr}^{(\delta)}}{\mu_{jtr}^{(\delta)}},$$

and similar results can be obtained for the host-specific parameters $\lambda_{ir}^{(y)}$, $\mu_{jr}^{(y)}$, $\lambda_{jpr}^{(\delta)}$ and $\mu_{jpr}^{(\delta)}$. The updates for $\nu_i^{(x)}$, $\xi_i^{(x)}$, $\nu_j^{(y)}$ and $\xi_j^{(y)}$ are identical to steps 7 and 8 in Algorithm 8.1. For the covariates

$$\lambda_{kh}^{(\phi)} = a^{(\phi)} + \sum_{i=1}^{|V_1|}\sum_{j=1}^{|V_2|}\sum_{t=1}^{T}\frac{A_{ijt}\theta_{ijt}\chi_{ijtl}}{1-e^{-\theta_{ijt}}},\ \mu_{kh}^{(\phi)} = \frac{\nu^{(\phi)}}{\xi^{(\phi)}} + \tilde{u}_k\tilde{v}_h T.$$

The updates for $\nu^{(\phi)}$ and $\xi^{(\phi)}$ are the same as step 9 in Algorithm 8.1. Finally, for the time dependent hyperparameters:

$$\nu_p^{(\gamma)} = b^{(\gamma)} + |V_1|da^{(\gamma)},\ \xi_p^{(\gamma)} = c^{(\gamma)} + \sum_{i=1}^{|V_1|}\sum_{r=1}^{d}\frac{\lambda_{ipr}^{(\gamma)}}{\mu_{ipr}^{(\gamma)}},$$

and similarly for $\nu_p^{(\delta)}$ and $\xi_p^{(\delta)}$. Finally, the updates for θ_{ijt} and $\boldsymbol{\chi}_{ijt}$ are similar to Appendix C.2. The required expectation $\mathbb{E}^q_{-N_{ijt},\mathbf{Z}_{ijt}}\{\log p(N_{ijt},\mathbf{Z}_{ijt}|\boldsymbol{x}_i,\boldsymbol{y}_j,\boldsymbol{\gamma}_{it},\boldsymbol{\delta}_{jt},\boldsymbol{\Phi}\}$ can be expanded similarly to (C.4), and the update equations for θ_{ijt} and χ_{ijtr} can be derived similarly to Appendix C.2:

$$\theta_{ijt} = \sum_{r=1}^{d}\exp\Big\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)}) + \Psi(\lambda_{itr}^{(\gamma)}) - \log(\mu_{itr}^{(\gamma)}) + \Psi(\lambda_{jtr}^{(\delta)}) - \log(\mu_{jtr}^{(\delta)})\Big\} + \sum_{k=1}^{K}\sum_{h=1}^{H}u_{ik}v_{jh}\exp\Big\{\Psi(\lambda_{kh}^{(\phi)}) - \log(\mu_{kh}^{(\phi)})\Big\},$$

$$\chi_{ijtl} \propto \begin{cases} \exp\Big\{\Psi(\lambda_{il}^{(x)}) - \log(\mu_{il}^{(x)}) + \Psi(\lambda_{jl}^{(y)}) - \log(\mu_{jl}^{(y)}) \\ \qquad + \Psi(\lambda_{itl}^{(\gamma)}) - \log(\mu_{itl}^{(\gamma)}) + \Psi(\lambda_{jtl}^{(\delta)}) - \log(\mu_{jtl}^{(\delta)})\Big\} & l \le R, \\ u_{ik}v_{jh}\exp\Big\{\Psi(\lambda_{kh}^{(\phi)}) - \log(\mu_{kh}^{(\phi)})\Big\} & l > R. \end{cases}$$

C.4. Variational inference in the joint PMF model

Variational inference for the joint PMF model presented in Section 8.12 proceeds similarly to Algorithm 8.1. The conditional posterior distributions are essentially the same as PMF and EPMF. The only exception is the conditional posterior distribution of x_{ir}:

$$x_{ir}|\mathbf{Z},\mathbf{Z}',\mathbf{Y},\mathbf{Y}',\zeta_i^{(x)} \sim \text{Gamma}\left(a^{(x)} + \sum_{j=1}^{|V_2|} Z_{ijr} + \sum_{j=1}^{|V_2'|} Z'_{ijr},\ \zeta_i^{(x)} + \sum_{j=1}^{|V_2|} y_{jr} + \sum_{j=1}^{|V_2'|} y'_{jr}\right).$$

For variational inference, a factorisation similar to (8.6) is assumed, further multiplied by the approximation for the posteriors of the additional components of the model: mainly $q(y'_{jr}|\lambda_{jr}^{(y')},\mu_{jr}^{(y')})$, $q(\phi'_{kh}|\lambda_{kh}^{(\phi')},\mu_{kh}^{(\phi')})$, $q(N'_{ij},\mathbf{Z}'_{ij}|\theta'_{ij},\boldsymbol{\chi}'_{ij})$. Therefore variational inference is also essentially unchanged, except the CAVI update for the parameters of variational approximation for the components x_{ir}, which become:

$$\lambda_{ir}^{(x)} = a^{(x)} + \sum_{j=1}^{|V_2|} \frac{A_{ij}\theta_{ij}\chi_{ijr}}{1-e^{-\theta_{ij}}} + \sum_{j'=1}^{|V_2'|} \frac{A'_{ij}\theta'_{ij}\chi'_{ijr}}{1-e^{-\theta'_{ij}}}, \qquad \mu_{ir}^{(x)} = \frac{\nu_i^{(x)}}{\xi_i^{(x)}} + \sum_{j=1}^{|V_2|} \frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}} + \sum_{j'=1}^{|V_2'|} \frac{\lambda_{jr}^{(y')}}{\mu_{jr}^{(y')}}.$$

Updates for the approximations of the additional components $N'_{ij}, Z'_{ijr}, y'_{jr}$ and ϕ'_{kh} follow from the updates for the variational parameters $\theta_{ij}, \chi_{ijr}, \lambda_{jr}^{(y)}, \mu_{jr}^{(y)}, \lambda_{kh}^{(\phi)}$ and $\mu_{kh}^{(\phi)}$ in Algorithm 8.1.

C.5. Evidence lower bound in PMF models

This section describes the calculations required to obtain the evidence lower bound for the standard PMF model. The ELBO is defined as:

$$\text{ELBO}(q) = \mathbb{E}^q\left\{\log\frac{p(\mathbf{X},\mathbf{Y},\boldsymbol{\zeta},\mathbf{N},\mathbf{Z},\mathbf{A})}{q(\mathbf{X},\mathbf{Y},\boldsymbol{\zeta},\mathbf{N},\mathbf{Z})}\right\}.$$

The two joint densities can be factorised as:

$$p(\mathbf{X},\mathbf{Y},\boldsymbol{\zeta},\mathbf{N},\mathbf{Z},\mathbf{A}) = p(\mathbf{A}|\mathbf{N},\mathbf{Z})p(\mathbf{Z}|\mathbf{N},\mathbf{X},\mathbf{Y})p(\mathbf{N}|\mathbf{X},\mathbf{Y})p(\mathbf{X}|\boldsymbol{\zeta})p(\mathbf{Y}|\boldsymbol{\zeta})p(\boldsymbol{\zeta}),$$
$$q(\mathbf{X},\mathbf{Y},\boldsymbol{\zeta},\mathbf{N},\mathbf{Z}) = q(\mathbf{N},\mathbf{Z})q(\mathbf{X})q(\mathbf{Y})q(\boldsymbol{\zeta}).$$

Using the above decompositions, the ELBO can be rewritten as:

$$\mathbb{E}^q\left\{\log\frac{p(\mathbf{A}|\mathbf{N},\mathbf{Z})p(\mathbf{Z}|\mathbf{N},\mathbf{X},\mathbf{Y})p(\mathbf{N}|\mathbf{X},\mathbf{Y})}{q(\mathbf{N},\mathbf{Z})}\right\} + \mathbb{E}^q\left\{\log\frac{p(\mathbf{X}|\boldsymbol{\zeta})p(\mathbf{Y}|\boldsymbol{\zeta})}{q(\mathbf{X})q(\mathbf{Y})}\right\} + \mathbb{E}^q\left\{\log\frac{p(\boldsymbol{\zeta})}{q(\boldsymbol{\zeta})}\right\}.$$

Therefore, three main components can be identified in the ELBO: the likelihood component, the latent feature component, and the hyperparameter component.

Likelihood component. The likelihood component of the ELBO is:

$$\mathbb{E}^q \left\{ \log \frac{p(\mathbf{A}|\mathbf{N},\mathbf{Z})p(\mathbf{Z}|\mathbf{N},\mathbf{X},\mathbf{Y})p(\mathbf{N}|\mathbf{X},\mathbf{Y})}{q(\mathbf{N},\mathbf{Z})} \right\}.$$

In the above expression, $p(\mathbf{A}|\mathbf{N},\mathbf{Z})$ is deterministic, $p(\mathbf{Z}|\mathbf{N},\mathbf{X},\mathbf{Y})$ is multinomial, and $p(\mathbf{N}|\mathbf{X},\mathbf{Y})$ is Poisson. If the Ber-Po link is used, $q(\mathbf{N},\mathbf{Z})$ is Poisson$_+$-Multinomial. With the Ber-Po link, the above expression results in:

$$\begin{aligned}
&\text{Numerator:} \quad \prod_{ij} \left\{ \exp\left(-\sum_r x_{ir}y_{jr}\right) \frac{(\sum_r x_{ir}y_{jr})^{N_{ij}}}{N_{ij}!} \frac{N_{ij}!}{\prod_r Z_{ijr}!} \prod_r \left(\frac{x_{ir}y_{jr}}{\sum_r x_{ir}y_{jr}} \right)^{Z_{ijr}} \right\}, \\
&\text{Denominator:} \quad \prod_{ij:A_{ij}>0} \left\{ \frac{\exp(-\theta_{ij})}{1-\exp(-\theta_{ij})} \frac{\theta_{ij}^{N_{ij}}}{N_{ij}!} \frac{N_{ij}!}{\prod_r Z_{ijr}!} \prod_r \chi_{ijr}^{Z_{ijr}} \right\}. \qquad \text{(C.5)}
\end{aligned}$$

If the Ber-Po link is not used, the denominator $q(\mathbf{Z})$ is simply multinomial. Note that, if the Ber-Po link is not used, $\mathbf{A} = \mathbf{N}$, and therefore it is not necessary add a variational distribution on $\mathbf{N}$. Taking the log-transformation of (C.5), after simplifications and cancellations, gives:

$$\begin{aligned}
&\text{Numerator:} \quad \sum_{ij:A_{ij}>0} \sum_r Z_{ijr} \log(x_{ir}y_{jr}) - \sum_r \left(\sum_i x_{ir} \right) \left(\sum_j y_{jr} \right), \\
&\text{Denominator:} \quad \sum_{ij:A_{ij}>0} \left\{ \sum_r Z_{ijr} \log(\chi_{ijr}) - \theta_{ij} - \log[1-\exp(-\theta_{ij})] + N_{ij} \log(\theta_{ij}) \right\}.
\end{aligned}$$

Taking the expectation with respect to $q(\cdot)$ finally gives:

$$\begin{aligned}
&-\sum_{r=1}^{d} \left(\sum_i \frac{\lambda_{ir}^{(x)}}{\mu_{ir}^{(x)}} \right) \left(\sum_j \frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}} \right) + \sum_{ij:A_{ij}>0} \left\{ \theta_{ij} \left(1 - \frac{\log(\theta_{ij})}{1-\exp(-\theta_{ij})} \right) + \log[1-\exp(-\theta_{ij})] \right. \\
&\left. + \frac{\theta_{ij}}{1-\exp(-\theta_{ij})} \left[\sum_r \chi_{ijr} \{ \Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)}) - \log(\chi_{ijr}) \} \right] \right\}.
\end{aligned}$$

Note that if $\chi_{ijr} = \theta_{ijr}/\theta_{ij}$, where θ_{ijr} represents the unnormalised χ_{ijr} values, and the ELBO is updated immediately after the update $\chi_{ijr} \propto \exp\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)})\} = \theta_{ijr}$, then the second summation in the expression clearly reduces to:

$$\sum_{ij:A_{ij}>0} \{\theta_{ij} + \log[1-\exp(-\theta_{ij})]\} = \sum_{ij:A_{ij}>0} \log[\exp(\theta_{ij}) - 1].$$

When the Ber-Po link is not used, the resulting portion of the ELBO is:

$$\sum_{ijr:A_{ij}>0} \chi_{ijr}\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)}) - \log(\chi_{ijr})\} - \sum_{r=1}^{d} \sum_{i} \frac{\lambda_{ir}^{(x)}}{\mu_{ir}^{(x)}} \sum_{j} \frac{\lambda_{jr}^{(y)}}{\mu_{jr}^{(y)}}.$$

which is obtained by ignoring the Poisson term for N_{ij} in (C.5), and noting that $N_{ij}! = A_{ij}! = 1\ \forall i, j$ and therefore the term can be ignored in the logarithm. If $\chi_{ijr} = \theta_{ijr}/\theta_{ij}$, where $\theta_{ijr} = \exp\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)}) + \Psi(\lambda_{jr}^{(y)}) - \log(\mu_{jr}^{(y)})\}$ and $\theta_{ij} = \sum_r \theta_{ijr}$, and the ELBO is calculated immediately after updating χ_{ijr}, then the first summation reduces to:

$$\sum_{ij:A_{ij}>0} \log(\theta_{ij}).$$

Latent feature component. The latent feature components correspond to the expectations $\mathbb{E}^q[\log p(x_{ir}|\zeta_i^{(x)}) - \log q(x_{ir})]$ and $\mathbb{E}^q[\log p(y_{jr}|\zeta_j^{(y)}) - \log q(y_{jr})]$. In this case:

$$\begin{aligned}\mathbb{E}^q\left\{\log \frac{p(x_{ir}|\zeta_i^{(x)})}{q(x_{ir})}\right\} = \log \frac{\Gamma(\lambda_{ir}^{(x)})}{\Gamma(a^{(x)})} + a^{(x)}\left\{\Psi(\nu_i^{(x)}) - \log(\xi_i^{(x)})\right\} - \lambda_{ir}^{(x)} \log(\mu_{ir}^{(x)}) \\ + (a^{(x)} - \lambda_{ir}^{(x)})\left\{\Psi(\lambda_{ir}^{(x)}) - \log(\mu_{ir}^{(x)})\right\} + \lambda_{ir}^{(x)}\left(1 - \frac{\nu_i^{(x)}}{\xi_i^{(x)}} \frac{1}{\mu_{ir}^{(x)}}\right).\end{aligned}$$

Hyperparameter component. The hyperparameter components are given by the expectations $\mathbb{E}^q[\log p(\zeta_i^{(x)}) - \log q(\zeta_i^{(x)})]$ and $\mathbb{E}^q[\log p(\zeta_j^{(y)}) - \log q(\zeta_j^{(y)})]$. The ELBO results in:

$$\begin{aligned}\mathbb{E}^q\left\{\log \frac{p(\zeta_i^{(x)})}{q(\zeta_i^{(x)})}\right\} = \log \frac{\Gamma(\nu_i^{(x)})}{\Gamma(b^{(x)})} + b^{(x)} \log(c^{(x)}) - \nu_i^{(x)} \log(\xi_i^{(x)}) \\ + (b^{(x)} - \nu_i^{(x)})\left\{\Psi(\nu_i^{(x)}) - \log(\xi_i^{(x)})\right\} + \nu_i^{(x)}\left(1 - \frac{c^{(x)}}{\xi_i^{(x)}}\right).\end{aligned}$$

The calculations in this section could be easily extended to the extended PMF model, where nodal covariates are introduced, using an identical procedure.

"The goofiness you must get yourself into to get where you have to go, the extent of the mistakes you are required to make! If they told you beforehand about all the mistakes, you'd say no, I can't do it, you'll have to get somebody else, I'm too smart to make all those mistakes. […] you will make mistakes on a scale you can't even dream of now – because there is no other way to reach the end."

Philip Roth - Sabbath's Theater (1995)