Diffusion of Information in Network Structures

Shohreh Shaghaghian

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

September 2017

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

2017/09/15

Abstract

Understanding the process by which a piece of data or information disseminates throughout a network is of great importance in many real world applications. Whether we are in charge of spreading the information or we are merely able to observe its traces, we need to model the process by which the diffusion occurs. In this thesis, we develop frameworks to model the information diffusion processes and propose multiple algorithms to both control and infer them.

We first study how we can proactively control the process of diffusion of information among a set of mobile nodes in the absence of a physical network infrastructure. Data transfer in this setting must rely on unscheduled sporadic meetings between nodes. Therefore, the main challenge is to develop a mechanism based on which nodes can learn to make nearly optimal forwarding decision rules despite having no a priori knowledge of the network topology. The forwarding mechanism should ideally result in a high delivery probability, low average latency, and efficient usage of the network resources. We propose both centralized and decentralized single-copy message forwarding algorithms that, under relatively strong assumptions about the networks behaviour, minimize the expected latencies from any node in the network to a particular destination. After proving the optimality of our proposed algorithms, we develop a decentralized algorithm that involves a recursive maximum likelihood procedure to estimate the meeting rates. We finally propose Bayesian versions of the decentralized algorithm that can take into account some external information about the social ties among the nodes to improve the forwarding decisions.

We also study how we can detect the underlying propagation structure by passively observing the traces of a diffusion process. The required sophistication of the inference approach depends on the type of patterns we want to extract as well as the number of observations that are available to us. We analyze scenarios in which not only the underlying network structure (parental relationships and link strengths) needs to be detected, but also the infection times must be estimated. We assume that our only observation of the diffusion process is a set of time series, one for each node of the network, which exhibit statistical changes when an infection occurs. Modelling the problem in a Bayesian framework, we propose both batch and online inference algorithms.

Sommaire

Comprendre les processus par lequel les données ou l'informations ce diffusent dans un réseau est important dans plusieurs contextes, autant réels que virtuels. Que nous soyons responsables de la diffusion de cette information ou que nous en soyons simplement un observateur, nous devons pouvoir modéliser le processus par lequel la diffusion se produit. Dans cette thèse, nous développons des cadres pour modéliser les processus de diffusion d'informations et proposons de multiples algorithmes pour soit contrôler ou bien inférer ces processus.

Nous étudions d'abord comment nous pouvons contrôler de manière proactive le processus de diffusion d'informations entre noeuds mobiles dans l'absence d'une infrastructure de réseau physique. Le transfert de données dans ce contexte dépend des rencontres sporadiques et non-planifiées entre les noeuds. Par conséquence, le principal défi est de développer un mécanisme par lequel les noeuds peuvent apprendre à créer des règles de décisions d'acheminement quasi-optimales sans aucune connaissance à priori de la topologie du réseau. Le mécanisme d'acheminement doit idéalement mener à une probabilité de livraison élevée, une faible latence moyenne, et une utilisation efficace des ressources du réseau. Nous proposons à la fois des algorithmes centralisés et décentralisés de transmission de messages à *copie unique* qui, dans le contexts de certaines suppositions relativement rigids par rapport au comportement des réseaux, minimisent les latences attendues pour la transmission entre n'importe quel noeud du réseau et une destination spécifique. Après avoir démontré l'optimalité des algorithmes proposés, nous développons un algorithme décentralisé incluant une procédure récursive de maximum de vraisemblance pour estimer les taux de rencontres entre noeuds. Nous proposons aussi des versions Bayésiennes de notre algorithme décentralisé capables de prendre en compte certaines informations externes relatif aux liens sociaux entre les noeuds, afin d'améliorer les décisions d'acheminement.

Nous étudions également comment nous pouvons détecter la structure de propagation sous-jacente en observant les traces d'un processus de diffusion. Le niveau de sophistication de l'approche d'inférence requis dépend du type de modèles que nous voulons extraire, ainsi que du nombre d'observations qui nous sont disponibles. Nous analysons des scénarios dans lesquels non seulement la structure du réseau sous-jacente (relations parentales et forces des liens) doit être identifiée, mais aussi les temps d'infection estimés. Nous présumons que les seules donnée disponibles relatives au processus de diffusion sont un ensemble de séries

temporelles, une pour chaque noeud du réseau, qui présentent des changements statistiques lorsqu'une infection survient. Modélisant le problème dans un cadre Bayésien, nous proposons des algorithmes d'inférence de traitement par lots ainsi qu'en ligne.

Acknowledgments

During my PhD studies, I have had so many moments of feeling lost and confused! At all these moments, there was just one person I have gone to: my advisor, *Professor Mark Coates*, who has patiently provided the vision, encouragement, and advice necessary for me to proceed. After every single meeting I had with you, I felt like I was where exactly I was supposed to be, doing what exactly I was supposed to do in my life. I am also grateful to my professors at McGill University from whom I learned a lot; my thanks go to *Professors Harry Leib, Doina Precup, Michael Rabbat*, and *Derek Ruth*. What I have learned from you has made the foundations of my career and professional life.

I am thankful to all the members of *Computer Networks Laboratory* at McGill for their kind helps. Working with such a smart, hard-working group of people has been an amazing unique experience for me.

I appreciate *Delta Kappa Gamma (DKG) International Society* not only for the generous fellowship award, but also for the positive energy I received from their many encouraging handwritten letters.

This dissertation is dedicated to my parents, *Zahra* and *Akbar*, who have always put my preference and comfort before theirs. There are no words to describe how much I love and respect you. I am also thankful to my uncle, *Daie Ali*. Without your help and support, I could have never applied for the graduate school in the first place. I would like to thank my siblings *Rozita*, *Saeed*, and *Mahboobeh* for being the best role models I could have asked for. Thanks for always being there for me.

I would also like to thank my wonderful friends, *Ahmadreza, Ali(s), Dena, Goli, Golnaz, Mahdi, Niloufar, Samira,* and *Sara* who are like a second family to me. Hats off to all the "Lunch at 1:30 pm?" and "Coffee at 4:30 pm?" texts we exchanged!

Finally, a very special thanks to my sweet *Babak*, who has been a great companion. Beside you, I am the happiest, strongest, kindest, and simply the best version of myself.

Contents

List of Figures

List of Tables

List of Acronyms

ACO	Ant Colony Optimization
AODV	Ad-hoc On-demand Distance Vector
CA	Cultural Algorithm
CAR	Context-Aware Routing
CGrAnt	Cultural Greedy Ant
DMP	Dynamic Message Passing
DSR	Dynamic Source Routing
DTN	Delay Tolerant Network
EBR	Encounter-Based Routing
ER	Epidemic Routing
GPS	Global Positioning System
LP	Linear Programming
MANET	Mobile Ad-hoc NETwork
MAP	Maximum A Posterior
MCMC	Markov Chain Monte Carlo
MEED	Minimal Estimated Expected Delay
MH	Metropolis-Hastings
ML	Maximum Likelihood
MSE	Mean Square Error
MV	Meets and Visits
ODE	Ordinary Differential Equations
PDF	Probability Distribution Function
PRoPHET	Probabilistic Routing Protocol using History of Encounters and Transitivity
SARS	Severe acute respiratory syndrome

SEIR	Susceptible-Exposed-Infected-Recovered
SEPR	Shortest Expected Path Routing
SI	Susceptible-Infected
SIR	Susceptible-Infected-Recovered
SLS	Stochastic Local Search
SMC	Sequential Monte Carlo
SNA	Social Network Analysis
SNC	Saami Network Connectivity
STD	Sexually Transmitted Diseases
SW	Spray and Wait
SWIM	Shared Wireless Infostation Model
TOUR	Time-sensitive Opportunistic Utility-based Routing
TTL	Time To Live
VANET	Vehicular Ad-hoc NETwork
VHF	Very High Frequency

Chapter 1

Introduction

The notion of transmission of information or some sort of contagion from one independent agent to another, exists in many phenomena around us, such as news propagation, virus spreading, communication protocols, political grass-roots campaigning, and activation cascades in biological and neural networks. Although the inherent logic and the propagation patterns are different in each of these phenomena, there is a common rule among all of them. Once an external cause affects a limited number of nodes, it is solely up to the dynamics between individual pairs of nodes to spread the information or infection through the network. We refer to the processes that can be studied in this framework as *Diffusion Processes*. There exist three main components in each diffusion process: *Nodes*, i.e., the set of separate agents; *Infection*, i.e., the change in the state of a node that can be transferred from one node to the other; and *Causality*, i.e., the underlying structure based on which the infection is transferred between nodes. The term *cascade* is often used to refer to the temporal traces left by a diffusion process.

Depending on the context and the application, diffusion processes may be studied with two different objectives. We either want to proactively control the diffusion process, or we intend to passively observe it. We refer to the first study approach as *proactive approach* or an approach with the *proactive perspective*. The goal of the proactive study approach is to take charge of defining transmission rules with respect to the available resources so that a desired performance is achieved. The proactive approach is widely used in telecommunication networks where we try to route messages to their destinations through efficient paths in Delay Tolerant Networks (DTNs). Similarly, we refer to the second study ap-

proach as *passive approach* or an approach with the *passive perspective*. From this passive perspective, the objective is to discover propagation patterns by observing the traces of the diffusion process in a more passive manner. Detecting these patterns not only gives us a valuable insight about the existing dynamics between agents, it also helps us predict, expedite, retard, or prevent future spreads. The passive approach is used to understand the spread of hashtags in social networks which is caused by influence of users over each other. Other examples of the passive study approaches include analyzing the propagation of distortions (caused by external events) among stock returns of different assets in the stock market, and studying the outbreak of a contagious disease in different geographic regions.

In this thesis, we study both perspectives of the diffusion processes. We propose appropriate frameworks to model and analyze these processes in each study approach and develop algorithms to fulfil the requirements of the corresponding applications. In the rest of this section, we discuss the outline and technical contributions of the remaining chapters of this thesis.

1.1 Thesis Organization and Contributions

The organization of the thesis is as follows.

- ◇ In Chapter 2, we illustrate the context in which we are going to pursue each of the aforementioned approaches in studying diffusion processes. For each study approach, we clarify the problem we are trying to solve and its applications. We also justify the contributions we are going to make in the subsequent chapters, by portraying a general insight of the problem background and providing a comprehensive literature review of the existing studies. Finally, we review the studies that have been conducted after ours to give the reader a comprehensive and up to date insight of the topic.

- ◇ In Chapter 3, we study how we can proactively direct a diffusion process in the context of message routing/forwarding in opportunistic Delay Tolerant Networks (DTNs). We declare the assumptions we have made and formulate the routing problem as an optimization problem. We then propose both centralized and decentralized single-copy message forwarding algorithms that, under simplifying assumptions about the network behaviour, minimize the expected latencies from any node in the network

to a particular destination. After proving the optimality of our proposed algorithms, we develop a decentralized algorithm that involves a recursive maximum-likelihood procedure to estimate the meeting rates between nodes. We confirm the improvement that our proposed algorithms make in the system performance through numerical simulations on datasets from synthetic and real-world opportunistic networks. The following publications have resulted from this work:

  - S. Shaghaghian, and M. Coates, "Opportunistic Networks: Minimizing Expected Latency", *Proceedings of IEEE International Conference Wireless and Mobile Computing, Networks and Communications (WiMob)*, Larnaca, Cyprus, October, 2014, pp. 473 - 478.
  - S. Shaghaghian, and M. Coates, "Optimal Forwarding in Opportunistic Delay Tolerant Networks with Meeting Rate Estimations", *IEEE Transactions on Signal and Information Processing over Networks (SIPN)*, vol. 1, no. 2, pp. 104-116, June 2015.

◇ In Chapter 4, we continue the proactive approach by studying the routing problem in a Bayesian framework. We apply the information about the strength of nodes' relationships as prior probability distributions. After proposing two Bayesian versions of the decentralized algorithm of Chapter 3, we evaluate their performance efficiency through simulations on synthetic and real-world datasets. The results of this Chapter are going to be submitted to a conference.

◇ In Chapter 5, we study how we can infer some characteristics of a diffusion process by passively observing it in a Bayesian framework. We assume that our only observation of the diffusion process is a set of time series, one for each node of the network, which exhibit changepoints when an infection occurs. After formulating a model to describe the contagion, and selecting appropriate prior distributions, we seek to find the set of model parameters that best explains our observations. Modelling the problem in a Bayesian framework, we exploit Markov Chain Monte Carlo, Sequential Monte Carlo, and time series analysis techniques to develop batch and online inference algorithms. We evaluate the performance of our proposed algorithms via numerical simulations of synthetic network contagions and analysis of real-world datasets. The results of this chapter are published in the following articles:

  - S. Shaghaghian, and M. Coates, "Bayesian Inference of Diffusion Networks with Unknown Infection Times", *Proceedings of IEEE International Workshop on Statistical Signal Processing (SSP)*, Palma de Mallorca, Spain, June, 2016. pp. 1-5.
  - S. Shaghaghian, and M. Coates, "Online Bayesian Inference of Diffusion Networks", accepted for publication in *IEEE Journal of Selected Topics in Signal Processing, Special Issues on Graph Signal Processing (J-STSP-GSP)*, September 2017.

- ◇ In Chapter 6, we summarize this thesis and make the concluding remarks. We also propose some directions for extending this study.

The only other contributor to the publications mentioned above (and their corresponding sections of this thesis), is Prof. Mark Coates who provided supervision and guidance. He developed Algorithms 1 and 2, and proved Theorems 2 and 3. He also helped in developing Algorithms 3-6. I derived and proved Lemmas 1 and 2, Propositions 1 and 2, and Theorems 1, 3, and 4. I am responsible for formulating the problems, developing algorithms 3-6, and performing and analyzing numerical simulations.

Chapter 2

Information Diffusion: Applications and Literature Review

2.1 Overview

In this chapter, we elaborate on the two perspectives of studying diffusion processes. We begin with the proactive study approach and specify the exact information diffusion process that we intend to proactively study. We state where this proactive study approach can be used in practice. We then move to the passive study approach and review some of the applications that passively observing a diffusion process may have. Moreover, we provide a comprehensive review of the existing studies that have been conducted from both proactive and passive perspectives and indicate the existing gaps that need to be further investigated. We also clarify how we contribute to this literature and overview the studies that have been conducted after the publication of the research results reported in this thesis.

2.2 Proactive Approach

We study proactive information diffusion in the context of *Delay/Disruption Tolerant Networks* (DTNs). DTNs are a class of wireless mobile node networks in which the communication path between any pair of nodes is frequently unavailable. Nodes are thus only intermittently connected. The history of studying DTNs goes back to 1990s when the research community began to explore how the Internet could fit into space communications [1]. Later in 2002, some terrestrial applications were found that could be modeled as DTNs

but with some inherent differences in the regularity of node meetings. These differences are the main criteria for classifying DTNs. Node connections can be either scheduled (thus predictable) or random (hence unpredictable). Space communication networks like the interplanetary Internet project are an example of the first group of DTNs with scheduled node connections. The second group of DTNs which is the main focus of this thesis, is known as *opportunistic networks* because nodes seize the opportunity to transfer data when a communication channel becomes available. In these networks, the meetings between nodes are unpredictable, so an arbitrary node is not aware of the exact time of its next meeting with other nodes. Opportunistic networks have been studied intensively in recent years (e.g., [1–3]) because they can fulfil a number of useful purposes. In Section 2.2.1, we present some of the most well-known applications of opportunistic networks deployed so far.

2.2.1 Applications

Some application scenarios are intrinsically opportunistic, in the sense that it is neither possible nor advisable to provide a more structured network based on legacy routing approaches [4]. In this section, we overview some examples of such applications that have been implemented in the real world.

Emergency Response in Disaster Scenarios

What happens when the infrastructural communication services like the Internet and cellular network become lost or degraded in case of natural or social disasters? When the central communication services become unavailable in such incidents, accessing the necessary information becomes more vital and at the same time more difficult. Under such circumstances, we may still depend on the services of individual mobile devices to build applications based on their available short-range communication facilities like ad-hoc Wi-Fi or Bluetooth. These applications may help first responders to provide a faster triage of the victims and coordinate a more efficient medical status acquisition. [5] presents a survey on multihop ad-hoc network paradigms for disaster scenarios and evaluates their applicability to important tasks in disaster relief operations. Opportunistic forwarding is one of these solutions that enables effective communication and collaboration in crisis situations via mobile devices like smart phones, tablets, laptops without the need for telecommunication infrastructure. Several applications such as ChaosFIRE [6] and Firechat [7] have been

implemented.

Through simulations in realistic disaster scenarios, [8] investigates the efficiency of the best known opportunistic routing protocols. We review these protocols in Section 3.2. The study shows how the different characteristics of an emergency scenario (such as the number of nodes, the number and size of messages) may impact the behaviour of each method in terms of delivery rate, delay, etc.

Non-intrusive Wildlife Tracking

In wildlife monitoring systems, the goal is to track wild species to more thoroughly investigate their behaviour such as migration patterns. These systems also aim to understand the interactions between species and their influences over one another, as well as the species' reaction to the ecosystem changes. Examples include learning about the effects of human development on wilderness areas, and the effects of changes of weather patterns or plant life on the indigenous species [4].

Opportunistic networks provide a reliable, cost-effective, and non-intrusive means of monitoring large populations in vast areas. Studies such as ZebraNet [9] and SWIM [10] have examined the research decisions and design tradeoffs of applying opportunistic forwarding techniques in a mobile sensor network designed to support wildlife tracking for biology research. The Princeton ZebraNet Project is an inter-disciplinary effort to develop, evaluate, implement, and test systems that integrate computing, wireless communication, and non-volatile storage along with Global Positioning Systems (GPS) and other sensors. The system can be modelled as an opportunistic network in which nodes are zebras wearing special collars. Epidemic and history-based forwarding approaches have been used to deliver messages to mobile base stations that periodically move around the area: We study these forwarding approaches thoroughly in Section 2.2.2. In SWIM (Shared Wireless Infostation Model), whales are the wild species to be monitored and the whales carry special devices through which the periodic data monitoring is performed. Data is diffused at pairwise contacts between whales and eventually arrives at the special fixed and mobile SWIM stations.

Provision of Data Communication to Remote and Rural Areas

Over the past decade, communication services have become more and more pervasive. However, due to the lack of fixed infrastructures, communication is still very expensive and inconvenient in some remote villages. Opportunistic networks can provide intermittent Internet connectivity to rural and developing areas where it is not cost-effective to deploy standard Internet access. The most representative village communication network is DakNet [11], which was developed by the MIT Media Lab and has been successfully deployed in some remote areas of India and Cambodia. The DakNet system consists of village Wi-Fi enabled kiosks, Internet access points, and vehicles equipped with mobile access points. Vehicles (e.g., public buses) carry mobile access points between village kiosks and a hub with Internet access. Data automatically uploads and downloads when the vehicle is in range of a kiosk or the hub.

Saami Network Connectivity (SNC) [12] is another project that provides asynchronous communication services to the nomads in Finland by focusing on a pure DTN architecture. Providing network connectivity to the Saami population is a means of protecting and defending their habits, culture, and traditions while also supporting their integration into the modern society of their countries [4].

Traffic Offloading in Cellular Networks

Cellular networks are currently overloaded, due to the increasing popularity of various applications for smartphones. Offloading mobile data traffic through opportunistic communications is a promising solution to partially solve this problem, because there is almost no monetary cost for it. [13] proposes to intentionally delay the delivery of information over cellular networks and offload it through the free opportunistic communications, with the goal of reducing mobile data traffic. The goal is to select the target set with a certain number of users, such that the mobile data traffic over cellular networks can be minimized.

2.2.2 Literature Review

In most opportunistic networks, the nodes are highly mobile and have a short radio range, and the density of nodes is low. In many cases, nodes have limited power and memory resources. These attributes combine with the intermittent connections to make routing traffic challenging. Routing is usually based on a *store-carry-forward* mechanism that

exploits node mobility. In this mechanism, the source transmits its message to a node it meets. This intermediate node stores and then carries the received message until it meets another node to which it can forward the message. This process is repeated until the message reaches its destination. The key ingredients in designing an opportunistic network routing protocol are the forwarding decisions: should a node forward a message to a neighbour it meets? should it retain a copy for itself?

Existing routing methods in opportunistic DTNs can be classified as replication-based, history-based, and social-based algorithms. In the rest of this section, we first describe each of these classes and overview the existing examples of each group. Since the focus of Chapter 3 is on deriving routing algorithms whose optimality can be theoretically proved, we then review the theoretical based studies. These studies try to mathematically optimize a performance metric related to the routing efficiency under certain assumptions about the network behaviour. We finally clarify how we are going to contribute to this literature and briefly overview the related works that have been published after ours.

Replication-based Methods

The first proposed approaches for routing in opportunistic networks were based on extending the concept of flooding to intermittently connected mobile networks. In these replication-based methods, a node forwards a copy of the messages stored in its buffer to all of (or to a fraction of) the nodes it encounters. There is no attempt to evaluate the capability of a given node to expedite the delivery. These routing algorithms have few parameters: they determine only how much replication can occur and which nodes can make copies of packets. One of the earliest algorithms was Epidemic Routing (ER) [14], in which a node forwards a message to any node it meets, provided that node has not previously received a copy of the message. Thus messages are quickly distributed through the connected portions of the network. Some studies have tried to analytically evaluate the performance of ER. [15] models the network as a random graph and derives the probability distribution function of the number of infected nodes, the average number of infected nodes, and the probability of all nodes becoming infected, all as functions of time. [16] derives a partial differential equation model for the dissemination of information via ER.

Other replication-based approaches (e.g., [17–22]) manage to reduce the transmission overhead of ER and improve its delivery performance through modification of the replica-

tion process and prioritization of messages. One of the most representative of such schemes is the two-hop routing proposed in [17]. In this multi-copy routing, whenever the source node encounters another node that does not have a copy of the message, the former forwards a message copy to the latter, referred to as the relay node. These relay nodes are only allowed to forward the message to the destination node. Therefore, all messages reach their destinations in at most two hops. Modelling the node meetings by a Poisson process, [18] studies two-hop forwarding and ER using Markov chain models, and derives the average delay and the number of copies generated at the time of delivery for these two routing schemes. Using Ordinary Differential Equations (ODEs), [19] derives similar parameters for ER and some of its variations.

Spray and Wait (SW) is another representative multi-copy routing scheme proposed in [20], that applies a limit L on the maximum number of relay nodes. Hence, only some of the nodes of the network can receive copies of the message. Two variants of SW, source SW and binary SW, are introduced in [20]. In source SW, only the source node is allowed to hand over copies of the message to other non-destination nodes. Therefore, when L is set to be infinity, the behaviour is equivalent to a two-hop routing scheme. In binary SW, the source of a message initially starts with L message copies. An arbitrary node A that has $n > 1$ message copies is allowed to forward messages to another node B with no copies. When A meets B, it forwards $\lfloor \frac{n}{2} \rfloor$ message copies to B and keeps the other $\lceil \frac{n}{2} \rceil$ ones for itself. When a node is left with just one copy, it can only forward to the destination.

Replication-based routing approaches result in a high probability of message delivery since more nodes have a copy of each message, but they can produce network congestion, increase overall delays, and drain each mobile node's limited battery supply. Some studies such as [23] propose a recovery scheme to delete all copies of the messages that have reached their destinations to decrease the buffer occupancy in replication-based methods.

History-based Methods

A step towards achieving more efficient routing approaches is to consider the history of node contacts in the network instead of blindly forwarding packets. History-based (also called utility-based) routing algorithms assume that nodes' movement patterns are not completely random and that future contacts depend on the frequency and duration of past encounters. Based on these past observations, both the source and the intermediate

nodes decide whether to forward a message to nodes they encounter or to store it and wait for a better opportunity. An early example is [24], which extends ER to situations with limited resources, incorporating a dropping strategy for the case when the buffer of a node is full. The dropping decisions are based on the meeting history of the node. PRoPHET (Probabilistic ROuting Protocol using History of Encounters and Transitivity), proposed in [25], is one of the first well known history-based routing protocols. PRoPHET assigns a delivery probability metric to each node which indicates how likely it is that the message will be delivered to the destination by that particular node. This metric is updated each time two nodes meet, and thus takes into account the history of meetings in the network. Denoting the delivery predictability that node A has for destination node B by $P(A,B) \in [0,1]$, three main update rules are defined in [25]. The first rule ensures that if two nodes are often in contact, they have a high delivery predictability for messages destined for one another.

$$P(A,B)_{new} = P(A,B)_{old} + (1 - P(A,B)_{old})P_{init} \tag{2.1}$$

where $P_{init} \in [0,1]$ is an initial constant. If two nodes do not encounter each other in a while, they are less likely to be good relay candidates to forward the messages destined for one another. Therefore, the second update rule makes sure that the delivery predictability values decay with time by applying an aging rule

$$P(A,B)_{new} = P(A,B)_{old} \times \gamma^k \tag{2.2}$$

where $\gamma \in [0,1)$ is the aging constant and k is the number of time units that have passed since the last time the metric was aged. The value of γ should be chosen according to the scenario and environment in which the protocol will be used. If encounters are expected to be very frequent, a lower value should be chosen for γ than if encounters are expected to be rare [26].

The third rule applies the transitivity effect. If nodes A and B have a high delivery predictability for messages destined for one another, and similarly nodes B and C are good candidates for forwarding the messages destined for each other, then A and C will also have a high delivery predictability.

$$P(A,C)_{new} = P(A,C)_{old} + (1 - P(A,C)_{old}) \times P(A,B) \times P(B,C) \times \beta \tag{2.3}$$

where $\beta \in [0, 1]$ is a scaling constant that decides how large the transitivity impact should be on the delivery predictability. When two nodes A and B meet, the delivery predictability metric $P(A, B)$ is updated. In [25], node A transfers a message it has for a destination node D if $P(A, D) > P(B, D)$. Simulation results in [25] show that in a community based network topology, PRoPHET has a better performance that ER. Also, it is shown that in a completely random topology (for which PRoPHET is not designed), the performance of PRoPHET is still comparable with the performance of ER, but with less communication overhead. In a later modification PRoPHETv2 [27], some update rules are revised. If an arbitrary node E has been met by another node at least once, we refer to it as a known node. The problem with the original transitive update rule is that as long as $\beta > 0$, the delivery predictability for any known node E will increase. This increase in the delivery predictability occurs regardless of whether any node in the network has recently met node E or not. In order to address this issue, PRoPHETv2 has changed the third update rule to,

$$P(A, C)_{new} = \max\Big(P(A, C)_{old}, P(A, B) \times P(B, C) \times \beta\Big) \tag{2.4}$$

By using the maximum, the evolution of the delivery predictability is allowed without subjecting it to exaggerated growth when no new information is available. The β parameter adjusts the weight of the transitive property of PRoPHET. The higher the value of β, the more rapidly encounters will increase delivery predictabilities through the transitivity effect [26]. Simulation results show that PRoPHETv2 performs better than PRoPHET in terms of delivery rate and overhead ratio, especially in heterogeneous network mobility scenarios.

MaxProp [28] is another well known example of a history-based algorithm. It was originally proposed for vehicular DTNs. In these networks, nodes move with higher speeds, reducing the amount of time they are in each other's radio range. Hence, the two main limiting resources are the duration of time that nodes are able to transfer data and their storage capacities. MaxProp uses a variant of Dijkstra's algorithm to estimate delivery likelihoods for each message. These estimated likelihoods are used to define the order in which messages should be transmitted and deleted. At the core of MaxProp is a ranked list of the nodes' stored messages based on a cost assigned to each destination. Messages with highest priority are the first to be transmitted during a transfer opportunity while messages with lowest priority are the first to be deleted to make room for the incoming messages.

Beyond this core, MaxProp applies several complementary mechanisms to increase the delivery probability and reduce the delivery delay. When two nodes meet, first all of the messages destined to the encountered node are transferred. Second, routing information is exchanged. Then, acknowledgements are transferred regardless of their sources and destinations. Afterwards, messages that have not travelled far in the network (i.e., messages with low hop counts) are given priority. Finally, the remaining messages are transmitted based on their delivery probability.

Other algorithms that make use of the history of node encounters to make forwarding decisions are Context-Aware Routing (CAR) [29], Meets and Visits (MV) [30], Shortest Expected Path Routing (SEPR) [31], Minimal Estimated Expected Delay (MEED) [32], and Encounter-Based Routing (EBR) [33].

Social-based Methods

More recently, the consideration of social interactions among nodes of a network have provided a new perspective in designing opportunistic routing protocols. The third group of routing methods for opportunistic networks, referred to as social-based methods, consider the fact that mobile devices are usually carried by members of a society and therefore the contact patterns depend on their social interactions. Social-based routing approaches are based on the assumption that social relations and behaviours among mobile carriers are usually long-term characteristics and less volatile than the node mobility patterns. These methods aim to take advantage of these stabilities to make more reasonable message forwarding decisions.

[34] divides the social characteristics into positive properties which can be used to improve the relay selection (e.g., community, centrality, similarity, and friendship) and negative properties which can hurt the routing performance (e.g., selfishness). Several studies (e.g., [35–38]) strive to take advantage of the positive social properties to predict the node meeting patterns. The main idea is to measure Social Network Analysis (SNA) parameters such as nodes' centralities (i.e., degree, betweenness, closeness) or detect communities to find more efficient forwarding rules.

In SimBet [35], a node decides to forward a message to other nodes based on their betweenness centralities and its social similarity with them. Due to the complexity of the centrality metrics in populated networks, the concept of ego networks is exploited in which

nodes are not required to exchange information about the entire network topology and only locally available information is considered. It has been empirically shown in [39] that locally calculated ego-centric betweenness highly correlates with global socio-centric betweenness. The SimBet utility metric which is used to make forwarding decisions is defined as the linear combination of similarity and betweenness utilities.

BubbleRap [36] makes use of community affiliation and nodes' centrality to make forwarding rules. It first uses the centrality metric to spread out the messages and then uses the community metric to direct the messages towards their destinations. Nodes are ranked based on their betweenness centralities both over the entire network (i.e., global ranks) and within their communities (i.e., local ranks). Overall popularity of a node as well as its forwarding capability for any message in the network is determined by its global rank. The local rank of a node shows its popularity and forwarding ability for messages meant to be delivered within its community. BubbleRap routing is performed by replicating a message to more globally popular nodes. However, when a node that belongs to the same community as the destination of the message is encountered, the message is forwarded to it irrespective of its global rank. Within a community, the message is replicated to more locally popular nodes until it reaches its destination. Community detection is performed by building an unweighted graph that represents the node contacts observed in the network.

The contact graph-based routing algorithm proposed in [37] uses the information derived from a weighted contact graph to make forwarding decisions based on neighbourhood, community, and the degree centrality of nodes. The suitability of a node as a relay is determined by three factors. The first factor is the strength of the node's tie to the destination. The node with a stronger tie to the destination is given preference for receiving a message. The next factor is the node's community affiliation. In order to give more importance to direct ties, this factor is considered only when the destination of a message is not a neighbour of any of the encountering nodes. The last factor is the degree centrality which is used only when both neighbourhood and community information are insufficient to differentiate the forwarding capabilities of the encountering nodes. The performance of the contact graph based routing algorithm shows that the information learned from an appropriately constructed weighted contact graph can improve the delivery performance while lowering other costs.

Other studies focus on the scenarios in which network nodes are uncooperative rational users who attempt to maximize their own utilities and conserve their resources. These

selfish behaviours of entities can significantly impact the routing performance. The second group of social-based studies (e.g., [40–45]) try to develop appropriate incentive mechanisms to stimulate individually selfish nodes to forward messages for all other nodes. Existing incentive mechanisms for DTN routing can be categorized into three categories: reputation-based, tit-for-tat-based, and credit-based (or virtual currency-based). In reputation-based schemes (e.g., [42]), forwarding services are provided to nodes depending on their reputation records. When a node provides services for other nodes, it gains good reputation. Nodes with good reputations can receive services from other nodes. On the contrary, misbehaving nodes get bad reputations and will be denied participation in the network. The fear of detection and punishment motivates nodes to cooperate. In tit-for-tat based schemes (e.g., [43]), every node forwards as many messages for a neighbour as the neighbour forwards for it. In this way, a node autonomously lowers services to a neighbour if it detects that the neighbour is misbehaving. The credit based schemes (e.g., [44, 45]) introduce some form of credit or virtual currency to regulate the message-forwarding relationships among different nodes. Nodes earn virtual currency by forwarding packets for others. These credits can be used to obtain forwarding service from any node in the network. For every forwarding request, the virtual bank charges the sender an extra amount of virtual currency, and the intermediate nodes redeem their rewards at the bank after successful delivery.

Our work focuses on routing a message to a single destination, but there are connections to research that addresses the task of spreading information to multiple nodes in a network. Of particular interest is the gossip-based approach in [46], which greatly reduces the number of message copies in the network while achieving near-optimal dissemination.

Theoretical Approaches

The experimental-based studies demonstrate the efficiency of their proposed methods by running simulations on traces recorded from real world opportunistic networks. Experimental analyses are valuable and take into account practical considerations, but they can leave us with an incomplete understanding of how an algorithm operates and how it will perform in other untested network conditions. For example, the behaviour of PRoPHET has been shown to be very sensitive to parameter choice [27]. It is also useful to design an optimal algorithm under slightly less realistic modeling assumptions, and then consider how it can be adapted to address the practical limitations, without completely losing its

desirable features. More recent studies have focused on deriving a forwarding process whose optimality (in some sense) can be mathematically proved under assumptions about network behaviour. [47] extends the two hop relay strategy of [17] by considering the expected delivery time to the destination as a metric to find the best set of candidate relays. By increasing the number of relaying steps recursively, a centralized single-copy multi-hop opportunistic routing scheme is proposed for sparse DTNs. The main defect of a centralized approach is that global knowledge of the network is required in order to make forwarding decisions.

There have been some efforts towards migrating to decentralized solutions that still provide performance guarantees. [48] proposes a decentralized time-sensitive algorithm called TOUR in which message priority is taken into account in addition to nodes' expected latencies when making forwarding decisions. Although in TOUR each node only needs to be aware of the local information about the rates of contacts with its own set of neighbours, the algorithm assumes that the node knows the exact contact rates. In most practical scenarios, this assumption is not valid.

Some researchers have explored how imprecision in the measurement or estimation of network parameters can impact the performance of opportunistic network routing algorithms. In [49, 50], Boldrini et al. discuss different sources of errors that may exist in parameter estimation like missed encounters, incorrect combination of short contacts, and memory limitations. They model these errors as a random variable with a normal distribution and evaluate the performance of four different forwarding schemes under this model. Although this error analysis is useful, Boldrini et al. do not specify how parameters should be estimated in order to obtain a performance that approaches what can be achieved when perfect a-priori knowledge of the network parameters is available.

Our Contributions

Although much research effort has been devoted to the development of opportunistic network routing algorithms [14–17, 20–25, 27–33, 35–38, 40–50], the algorithms are either centralized, have no performance guarantees, or ignore the need to estimate network parameters. In Chapters 3 and 4 of this thesis, we aim to propose decentralized routing algorithms that can work with no a priori knowledge of the meeting rates among network nodes. Our work focuses on the MANET setting, where node speed is much reduced compared to the

VANET case, and we can assume that there are fewer restrictions on the amount of data that nodes can transfer when they meet. In Chapter 3, we derive a decentralized routing algorithm that has performance guarantees (under simplifying assumptions about the network behaviour). When the meeting times between nodes are independent and exponentially distributed, the routing algorithm minimizes the expected latency in sending a packet from any source node to a specific destination. We examine the behaviour of the routing algorithm when the meeting rates are learned online using a recursive maximum likelihood procedure. We show that, for a stationary network, the decision rules and achieved expected latencies converge to those obtained when there is exact knowledge of the meeting rates. We present the results of simulations that compare the performance of the proposed algorithm to previous approaches, and examine how the algorithm is affected by practical network limitations (finite buffers, restrictions on data exchange, message expiry times).

The routing algorithm proposed in Chapter 3 can be classified as a history-based routing algorithm, because the forwarding rules are mainly based on the history of nodes' contacts. In Chapter 4, we add some social characteristics of the network to improve the meeting rate estimation and routing procedures. We use a Bayesian framework and apply the social properties in prior distributions.

Subsequent Studies

In order to provide the reader with a complete and up-to-date view of the available literature on the topic of routing in opportunistic DTNs, we review the studies that have been published after our work (i.e., since 2015) in the rest of this section.

The precision of the analytical models clearly depends on how accurate the estimation of the meeting rates is. This accuracy directly depends on how well the probability distributions are characterized. [51] proposes new approaches to characterize the distribution of inter-meeting times that result in more precision of analytical models that are developed based on exponential distributions. Three different characterizations of the inter-meeting times are described in [51] and two metrics are used to evaluate them. Relations between these characterizations are also investigated. [52] studies how detecting the abnormal values in time series of inter-meeting times can benefit the decision made for the forwarding rules.

We previously explained that the replication-based routing methods may cause conges-

tion in the network while the single copy routing protocols have on average high message delivery latencies. [53] studies how the message copies can be dynamically allocated to keep a balance between the delay and cost of message delivery. In the routing protocol proposed in [53], the remaining TTL (Time To Live) of a message is used to decide about the minimum number of copies necessary to achieve a given delivery probability. Experiment results show that the proposed adaptive multi-step routing protocol has a higher delivery ratio and a lower delivery cost compared to SW and BubbleRap algorithms.

As a history-based routing protocol, [54] extends the two-hop multi-copy routing scheme proposed in [20] by developing an optimal dynamic relay node selection algorithm. Similar to our approach, the goal is to minimize the expected delivery delay. The performance is evaluated through numerical experiments.

As a social-based routing protocol, CGrAnt [55] uses operational metrics that characterize the opportunistic social connectivity between wireless users. The main idea is that opportunistic networks are dynamic environments for which swarm intelligence methods, including approaches based on Ant Colony Optimization (ACO) [56] and Cultural Algorithms (CAs) [57], can be adopted. The most promising message forwarders are selected via a greedy transition rule based mainly on local information captured from the DTN environment. Whenever global information is available, it can also be used to support decisions. [58] explores the structural vulnerability of social-based forwarding and routing methods in opportunistic networks. It introduces a new problem of assessing the performance reliability of opportunistic routing strategies in DTNs from a community structure point of view. The proposed approach aims to identify the most vulnerable devices in the network whose non-participation (due to out-of-service or permanent out-of-range) transforms the current network community structure to a totally different one.

In [59], a solution is proposed to deal with routing attacks in DTNs. First, a general purpose proactive defence mechanism is proposed for DTNs based on the routing information obtained from the messages forwarded by the relay nodes and the acknowledgements generated by the destination nodes. Then, evolutionary game theory is applied with the proposed defence mechanism to analyze the strategy changes of the nodes in the networks. The result is a routing defence scheme which encourages the nodes not to attack but to cooperate with others.

Studies such as [60] and [61] discuss how maritime networks can be modelled as opportunistic DTNs. [60] uses shipping lane information to predict the meeting opportunities of

the ships to optimize the route selection. [61] proposes an improvement over PRoPHET for underwater communication. [62] evaluates the performance of different DTN routing protocols including ER and PRoPHET in real world vehicular networks. Different scenarios with different degrees of connectivity and mobility are used to test the feasibility and efficiency of routing algorithms in real vehicular environments.

2.3 Passive Approach

We now turn our attention from proactive approaches to passive approaches in studying diffusion processes. Diffusion processes may be observed for different purposes [63]. In some cases (e.g., [64–68]), the goal is to identify the most influential nodes of the network. The main concern in other studies (e.g., [69–71]) is to detect the popular topics that have diffused throughout a network. In this thesis, we pursue the goal of a third group of studies (e.g., [72, 73]), i.e., to infer the causality component of a diffusion process using observations related to the cascades. We define *infection* (or *contagion*) as the change in the state of a node that is triggered by an external source or by another node that has already changed its state. We aim to exploit the available evidence to infer the path that an infection has traversed in order to reach an arbitrary node. Inferring this path not only gives us valuable insight about the existing dynamics between the nodes of the network, it also helps us predict, expedite, or prevent the spread of future infections. In Section 2.3.1, we state some specific areas that can benefit from the result of each of the three purposes of observing diffusion processes. We then review the related research studies that pursue the same goal as ours in Section 2.3.2.

2.3.1 Applications

Propagation of a change in the state or behaviour of individuals or parties is a concept that exists in various domains. Designing frameworks to model and analyze different realizations of this concept can provide us with better understanding of each phenomenon and leads to making better decisions, developing more effective policies, and guaranteeing more balanced growth.

Bioinformatics

One of the challenges in the area of brain research is to discover networks describing the flow of information among communicating neurons in the form of electrophysiological signals. These networks are thought to be responsible for perceiving and learning about the environment, as well as producing behaviour. Monitoring these networks is limited by the number of electrodes that can be placed in the brain of an awake animal [74]. Designing computational tools to detect, infer, and reason about these networks from the available data is of crucial importance. Another similar challenge is in protein signalling networks. These networks play a key role in cellular function, and their dysregulation is central to many diseases, including cancer [75]. Developing statistical approaches to make inferences regarding the network structure is required to elucidate the signalling network topology.

Social Network Analysis (SNA)

Social networks such as Twitter and Facebook have created a new media for individuals to directly or indirectly influence each other's decisions [76]. A tangible example is the mobile applications that people install on their smart phones. Learning about an application that a friend is using can persuade you to install not only because you prefer to use a piece of software or App that is compatible with his/her version (direct effect), but also because of the trust you have regarding your friend's knowledge and choice of technology (indirect effect). This mechanism can have a cascading effect on the spread of a new behaviour or product and is being widely used by different industries as a quick and cheap marketing strategy. As an example, according to [76], Ford Motor Company released its Fiesta model through social media networks using the top 100 video bloggers (Vloggers) in 2009. This resulted in 3000 reservations for the model by March 2010 by first time Ford buyers [77]. On the other hand, the diffusion mechanism in social networks may also be used with malevolent intent to spread malicious gossip or untruthful information. For example, according to [76], in Summer 2012, false rumours about impending attacks on northeast Indian migrant workers, caused panic and a widespread migration of these workers back to their home states in northeastern India [78]. Similarly, following a recent fake tweet about an explosion in the White House, the Dow Jones industrial average dropped 152 points within seconds [76,79]. Because of the significant positive and negative impacts that diffusion processes in social network can have on our lives, it's necessary to thoroughly

understand their dynamics and develop strategies to control them.

Epidemiology

In numerous cases of disease spreading, social contacts are the main factor of transmission. Sexually Transmitted Diseases (STDs) are a good example of diseases that depend solely on personal contacts for dissemination [80]. [81] has studied the propagation patterns of Severe Acute Respiratory Syndrome (SARS) which was first reported on November 16, 2002 in southern China, and later spread throughout the world and seeded outbreaks in Hong Kong, Vietnam, Singapore, and Canada. By March 2003, many countries had instituted general infectious disease control measures, such as quarantine, isolation, and strict hygiene measures in hospitals. According to [81], delaying the institution of control measures by 1 week would have nearly tripled the epidemic size and would have increased the expected epidemic duration by 4 weeks. Understanding such implications is essential to help public health officials to develop appropriate strategies to restrain the spread of the disease.

Finance

In stock markets, distortions in the price of one stock can influence other stocks' prices through supply chains or product competitions. Each of the affected stocks may cause distortions in the returns of a second tier of stocks, each of which can impact others in the same way. This cascade effect can lead to similar behaviour among seemingly irrelevant stocks. Understanding these kinds of interactions helps investors to forecast stock behaviours and make smarter decisions. As an example, it has been shown in [68] that industries that are more central in the network of inter-sectoral trade earn higher stock returns than industries that are less central. In addition, the empirical evidence suggests that sectoral shocks that contribute to aggregate risk are more likely to pass through central industries than peripheral industries. The information diffusion process also exists in investor populations and influences trading behaviour and returns. [82] studies how the so called word-of-mouth effects influence behaviour in various financial market settings. [67] shows that central investors earn higher returns and trade earlier than peripheral investors with respect to information events.

2.3.2 Literature Review

A major factor that differentiates network inference methodologies is the type of available observations and the amount of useful information that can be extracted from them. In most of the social network inference scenarios, for example, the cascade trace is directly observable. The moment of time at which an arbitrary node shows infection symptoms can be easily specified and identified. The majority of the available network diffusion inference literature (e.g., [72, 73, 83–87]) focuses on such frameworks where the cascades are directly and perfectly observed. However, there are other scenarios in which we cannot easily determine when nodes become infected. An example of such a scenario is presented in Figure 2.1. Having access to the number of weekly reported cases of measles and chickenpox in seven major cities of England and Wales for almost 40 years [88], we can observe that different regions become infected at different points in time. The infection then dies away (the region returns to a "susceptible" state). However, we cannot easily pinpoint the exact week when a region becomes infected. Therefore, more sophisticated inference methods are required for cases with limited or indirect observations. [89–92] have studied diffusion processes in which the cascade trace is not directly observable or is partially missing. In [90–92], it is assumed that a portion of the cascade data is directly observable and the authors propose techniques to infer the causality structure from this portion. In [89], the cascade trace is unavailable, but other observable properties of the cascade are used to infer the causality structure. Although these approaches can identify how an infection has traversed the network (in the example of Figure 2.1, which region is primarily responsible for infecting another), they sometimes do not provide all of the information required to take appropriate action to mitigate or expedite the spread of an infection. In the example of a highly contagious disease, we may choose to control transportation and movement between regions or implement stricter health checks. In the example of the stock market, we may choose to invest in a stock that is likely to be affected (infected) soon by an external disruptive influence. To implement such strategies, it is important to estimate when nodes have become infected and to infer model parameters that allow us to construct predictions of when future infections will occur.

In the rest of this section, we review the existing research most closely related to our passive study approach. We first discuss the studies that assume the cascades (infection times) are perfectly observed and propose methods to infer just the network structure.

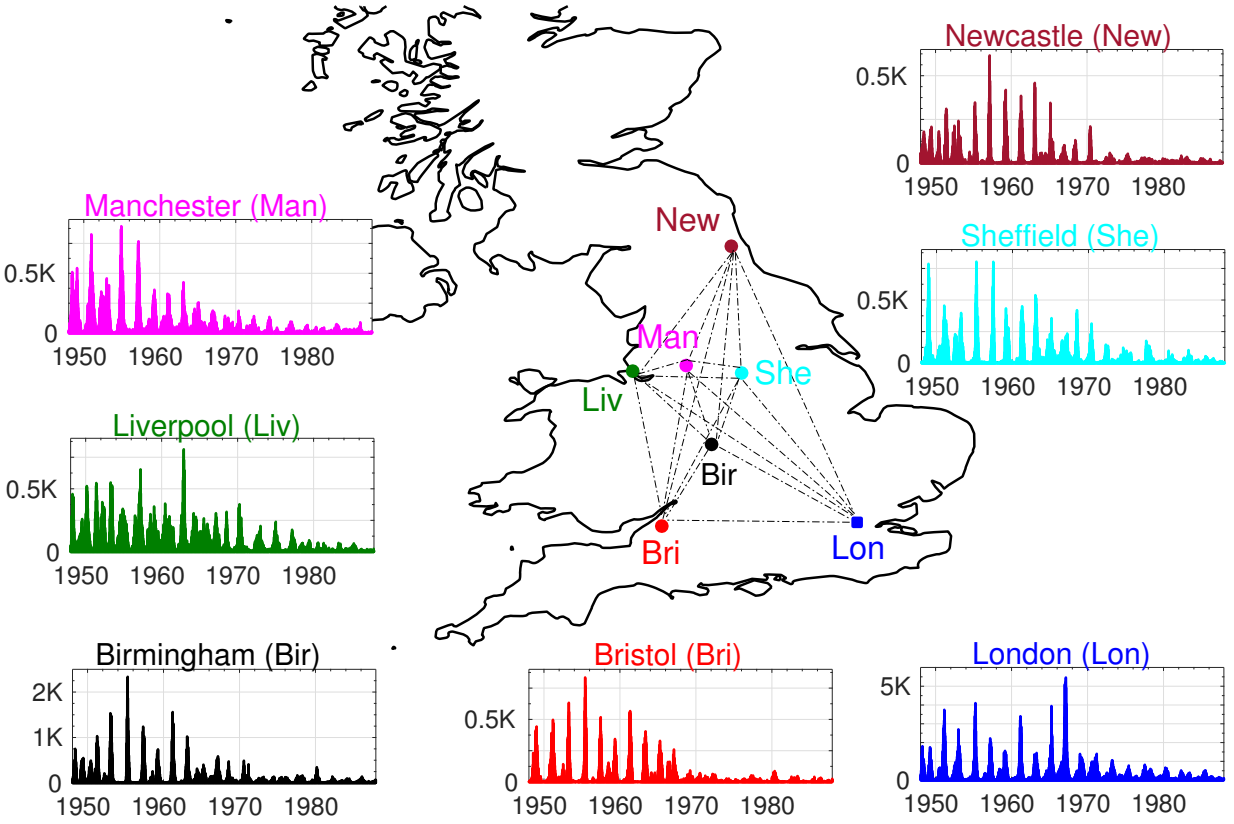


Figure 2.1 Weekly reports of Measles and Chickenpox in 7 major cities of England and Wales: London (Lon), Bristol (Bri), Liverpool (Liv), Manchester (Man), Newcastle (New), Birmingham (Bir), Sheffield (She)

Then, we describe the few existing studies that have the same assumption as our research contribution in Chapter 5, i.e., neither the network structure nor the infection times are perfectly or directly observed. We finally survey methods for detecting the moment of time at which the statistical characteristics of multiple time series change. These methods do not involve any notion of an underlying diffusion network that induces relationships between the time series. We end this section by clarifying how this thesis contributes to this body of literature.

Network Inference with Perfect Cascade Observation

Most of the earlier work exploring diffusion network inference techniques assumes that cascades are perfectly observed, i.e., the infection times are exactly known. [72] models dif-

fusion processes as discrete networks of continuous temporal processes occurring at different rates. Given the infection times, the goal is to infer the parental relationships and estimate the link strengths that maximize the likelihood of the observed data. An algorithm called NETRATE is developed to solve the convex optimization problem. [73] uses a tree-shaped graph of the parental relationships inferred from observed cascades of different contagions to infer the complete set of edges of the graph (for example a friendship graph in a social network). The proposed NETINF algorithm is used to find the maximum likelihood graph conditioned on the set of observed cascades.

[85] uses the same setup as [72], but it assumes that the underlying network structure is not static and infection pathways change over time. A stochastic convex optimization procedure is employed to infer the dynamic network and an online inference algorithm called INFOPATH is developed to solve it. [86] studies how the network evolution affects the diffusion process and proposes a joint continuous-time model to account for co-evolutionary dynamics between these two processes.

[87] considers inferring the network structure as an intermediate task and focuses on estimating joint properties of networks and diffusion processes such as the node influence score of a contagion. The network structure (parental relationships and link strengths) is assumed to be hidden and the infection times are observed. A Bayesian framework is used to calculate the expectation of the hidden parameters under the posterior distribution. Instead of inferring the network structure, [83] and [84] focus on the global influence a node has on the rate of diffusion through the network. The authors develop a linear influence model in which the growth in the number of newly infected nodes is expressed as a function of the infection times of the previously infected nodes. [83] shows that the influence function of each node can be estimated using a simple least squares procedure by modelling it in a non-parametric way. [84] uses the same linear influence model, but introduces sparsity in the estimated influence matrix and applies regularization penalties to take into account the nodes' centralities.

Network Inference without Perfect Cascade Observation

We now review the existing inference techniques for scenarios where cascades are not perfectly observed. Assuming that the infection times are only partially observed and the diffusion trace is incomplete, [90] develops a two-stage framework to pinpoint the infection

source. After learning a continuous-time diffusion network model based on the historical diffusion trace in the first stage, the source of an incomplete diffusion trace is identified by maximizing its likelihood under the learned model. Importance sampling approximation is used to find optimal solutions. Using an SEIR (Susceptible-Exposed-Infected-Recovered) infection model, [89] assumes that partially observed probabilistic information about the state of each node is provided, but the exact state transition times (infection times) are not observed. The underlying network is inferred by minimizing the expected loss over all realizations of the unobservable trace. The loss function is the negative log-likelihood of node state probabilities at each observed time point in a realization of infection times. Although the network structure can be detected using a convex optimization problem, the transition (infection) times are not estimated.

[91] studies the theoretical learnability of tree-like graphs given the initial and final set of infected nodes. The traces are defined as sets of unordered nodes and the authors strive to reconstruct the underlying network. The algorithm proposed in [91] works by observing the relation that a particular vertex's infection has on the likelihood of infection at other locations in the tree. The goal in [92] is to reconstruct the so-called node couplings using Dynamic Message Passing (DMP) equations. The authors assume that the cascade observations are only partially available and define coupling of nodes i and j as the probability that the infected node i transmits the contagion to its susceptible neighbor j.

Changepoint Detection

Another sizeable, related body of literature addresses detecting abrupt changes in the statistical structure of multiple time series. The moments in time that divide time series into distinct homogeneous segments are referred to as changepoints. We refer the reader to [93] for a detailed discussion of the topic.

Most changepoint detection, or time series segmentation, methods strive to detect single and multiple changepoints in univariate ([93–95]) or independent multivariate ([96]) time series. More closely related to our work is the approach in [97], which involves an underlying Gaussian graphical model that captures the correlation structure between multivariate time series. There is no notion of a diffusion process; the model captures contemporaneous correlation structure. In Chapter 5 of this thesis, we strive to detect the changepoints of multiple time series in the context of a background diffusion process that dictates when

the changepoints occur. We combine the indications of change in the statistical characteristics of the observed cascades with the causality relationships between nodes to infer the underlying structure as well as the infection times (i.e., the changepoints). We, however, use the pure statistically detected changepoints as a ground truth to evaluate the infection times we detect for the diffusion process.

Our Contributions

In Chapter 5 of this thesis, we improve upon the existing methods by proposing an approach to simultaneously infer the structure and the cascade trace of a diffusion process when the infection times are completely unavailable. We assume that the only observation we have from each node is a time series whose characteristics provide an indication of the behavioural changes caused by receiving the infection. Modelling the problem in a Bayesian framework, we develop a batch inference algorithm based on Markov Chain Monte Carlo (MCMC) to detect the causality structure and estimate the unobserved infection times. In this inference method, we consider the cases where no infection occurs in a batch of data. We then extend the batch inference model to online scenarios where inference decisions need to be made before the entire time series is available. We develop an online version of the inference algorithm using some Sequential Monte Carlo-based (SMC-based) techniques. We evaluate the performance of our suggested methods using both synthetic and real-world datasets.

Chapter 3

Proactive Approach: Routing in Opportunistic Delay Tolerant Networks

3.1 Overview

As described in Chapter 2, in Delay/Disruption Tolerant Networks (DTNs), a communication path between any two nodes is frequently unavailable and nodes are only intermittently connected. Since end-to-end connections cannot be established at any moment, the conventional routing algorithms for MANETs such as AODV (Ad-hoc On-demand Distance Vector) [98] and DSR (Dynamic Source Routing) [99] do not work properly. Due to this dynamic nature of DTNs, routing is usually performed based on a store-carry-forward mechanism. The source transmits a message to a node it meets; this intermediate node stores the message by buffering it, then carries it (the mobility of the node is exploited), before forwarding it to another node it meets. Opportunistic networks are a subcategory of DTNs in which the node connections are random rather than scheduled. This unpredictability of meetings in opportunistic DTNs makes designing an efficient routing protocol even more challenging.

In this chapter, we study the problem of message forwarding in opportunistic DTNs and propose both centralized and decentralized routing algorithms. Prior to introducing our algorithms, we state the notations and formulate the problem in Section 3.2. In Section

3.3, we assume that a global knowledge of the network is available at a central unit and propose a centralized algorithm to minimize the expected latency from all nodes of the network to a particular destination. In Section 3.4, we assume that no centralized unit is available and each node is only aware of the rates of its own meetings with other nodes (i.e., partial a priori knowledge). Since each node is responsible for finding its optimal forwarding decisions, we propose a decentralized version of the forwarding algorithm and prove that it converges to the same optimal solution of the centralized algorithm. Finally, we propose a decentralized approach with no a priori knowledge in Section 3.5. In this setting, an arbitrary node does not even know its own meeting rates. Therefore, all the meeting rates need to be estimated. The numerical simulation results of applying the proposed algorithms on synthetic and real-world datasets are presented in Section 3.6. We also compare the performance of our proposed algorithms with other existing approaches in terms of latency, delivery rate, hop counts, and buffer occupancy. We summarize the contributions of the chapter in Section 3.7. Proofs of lemmas and theorems stated in this chapter are provided in Section 3.8.

3.2 Problem Statement

We consider a set of N mobile nodes $\mathcal{N} = \{1, \ldots, N\}$ that have short communication ranges. When two nodes i and j are located in the communication range of each other, we say they have met or contacted each other. In this set up, nodes i and j can communicate with each other only if they meet. We refer to the time between two consecutive meetings of nodes i and j as their *inter-meeting time* and denote it by x_{ij}. Based on the key characteristic of opportunistic networks, meeting times are not pre-scheduled. Therefore, x_{ij} should be modelled as a random variable. Although the aggregate inter-meeting distributions of nodes in MANETs often follow a truncated powerlaw distribution [100, 101], there is evidence that the inter-meeting times of individual pairs of nodes can be adequately modelled by exponential distributions with heterogeneous coefficients [102–105]. In particular, Conan et al. [102] and Gao et al. [103] conduct statistical analyses of mobile social network data traces, including the Infocom data set [106] that we analyze in Section 3.6. They demonstrate that most pairs of nodes have inter-meeting times that are approximately exponentially distributed. In [104, 105], approximately exponential distributions of individual meeting times are detected through statistical analyses of car/taxi mobility

traces. We assume that x_{ij} follows an exponential distribution with parameter λ_{ij}, i.e.,

$$f_{x_{ij}}(x|\lambda_{ij}) = \lambda_{ij} e^{-\lambda_{ij} x} \tag{3.1}$$

After n_{ij} meetings between nodes i and j, we assume that we have n_{ij} independent samples of this distribution $\{x_{ij}^1, \ldots, x_{ij}^{n_{ij}}\}$.

We associate with the network a *contact graph* which is formed by adding a link between any two nodes that meet. We assume that the contact graph is connected and denote the set of neighbours of node i in this graph by $\mathcal{S}_i = \{j \in \mathcal{N} | \lambda_{ij} > 0\}$. We focus on the scenarios in which all the N nodes of the network aim to send messages to a particular destination node d and intend to minimize the sum of the expected latencies of all the nodes in the network to the specified destination. We can effectively provide routing decisions for all destinations in the network just by solving the same problem for all $d \in \mathcal{N}$. Since the contacts between nodes are not pre-scheduled, we cannot identify end-to-end paths ahead of time. Hence, solving the routing task is equivalent to identifying the forwarding decisions that nodes should make when meeting each other. In order to address the scenarios in which messages are large and nodes' memory resources are limited, we focus on single-copy routing algorithms. In these algorithms, only one copy of a message exists in the network at any time. Each time node i meets one of its neighbours $j \in \mathcal{S}_i$, it forwards the messages destined for d with probability p_{ij}, or keeps them with probability $1 - p_{ij}$. When a node decides to forward its messages to another node, it deletes them from its own buffer. In Chapter 6, we briefly explain how our proposed single-copy algorithms can be used to develop multi-copy routing algorithms.

Considering the matrix $\mathbf{P}_{N \times N}$ comprised of all pairs i and j, we set $p_{ij} = 0$ if nodes i and j never meet and are thus not neighbours in the contact graph. We denote the forwarding probabilities of node i by the vector $\mathbf{p}_i$; this is the i-th row of the matrix $\mathbf{P}$. Table 3.1 lists the notations used in this chapter.

Symbol(s)	Expression(s)	Definition(s)
$\mathcal{N}$	$\{1,\ldots,N\}$	Set of all the N nodes in the network
x_{ij}^k	-	The random time between $k-1^{th}$ and k^{th} meetings between nodes i and j
λ_{ij}	-	The meeting rate between nodes i and j
n_{ij}	-	The number of meetings between nodes i and j
d	-	The destination node
$\mathcal{S}_i$	$\{j \in \mathcal{N} \mid \lambda_{ij} > 0\}$	Set of neighbours for node i
$\mathbf{P}$	$[p_{ij}]_{N\times N}$	The forwarding probability matrix
$L_{id}(\mathbf{P})$	$\frac{1+\sum_{j\in\mathcal{S}_i} p_{ij}\lambda_{ij}L_{jd}(\mathbf{P})}{\sum_{j\in\mathcal{S}_i} p_{ij}\lambda_{ij}}$	The expected latency from node i to destination d
$\mathbf{B}$	$[b_{ij}]_{N\times N}$	The binary forwarding decision matrix with components b_{ij}
$\mathbf{B}^*$	$\arg\min_{\mathbf{B}} \sum_{i=1}^N L_{id}(\mathbf{B})$	The optimum binary forwarding matrix which is the solution to (3.4)
$\mathcal{A}$	-	The set of nodes whose forwarding decision rules have been made in Algorithm 1
$\hat{L}_{id}(j)$	-	The estimate at node j of the latency from node i to the destination
T_i	-	The earliest time by which node $k>1$ has met all nodes in the set $\{1,\ldots,k-1\}\cap\mathcal{S}_k$ in the time period $(T_{k-1},T_k]$
$\hat{\lambda}^n$	$\frac{n}{\sum_{i=1}^n x_i}$	The ML estimate of meeting rates between two nodes after n meeting
$\widetilde{\mathbf{B}}_t$	-	The binary decision matrix achieved by MinLat-E at time t
$\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i)$	-	The estimate at node i at time t of its expected latency to the destination node in MinLat-E

Table 3.1 Summary of notations used in Chapter 3

The expected latency from node i to destination d is a function of the probability decision matrix $\mathbf{P}$ and we denote it by $L_{id}(\mathbf{P})$. Our goal is to find the matrix $\mathbf{P}^*$ such that the sum of the expected latencies of all the nodes in the network to the specified destination d is minimized. Let us call this utility function $\mathcal{U}(\mathbf{P}) = \sum_{i\in\mathcal{N}} L_{id}(\mathbf{P})$. We assume that nodes' buffer sizes are unlimited and message Time To Live (TTL) is infinity. These assumptions imply that messages never expire and nor are they discarded due to the buffer overflow. We also assume that nodes' speed and message lengths are such that any number of messages can be forwarded during each meeting. The last assumption is that the network topologies and meeting rates are such that the solution $\mathbf{P}^*$ is unique. If not, our algorithms guarantee that we reach one of the optimal matrices, but the proofs are more complicated. The first step towards achieving our goal and finding matrix $\mathbf{P}^*$ is to discover how the expected latency of an arbitrary node i, $L_{id}(\mathbf{P})$, depends on the elements of the probability decision matrix $\mathbf{P}$ in general. Using some characteristics of the exponential distribution, Lemma 1 provides an expression for $L_{id}(\mathbf{P})$ in terms of $\mathbf{P}$ and $\lambda_{ij}, j \in \mathcal{S}_i$. The proof is available in Section 3.8.1.

Lemma 1. *The expected latency of a node $i \in \mathcal{N}$ to the destination d is*

$$L_{id}(\mathbf{P}) = \frac{1 + \sum_{j\in\mathcal{S}_i} p_{ij}\lambda_{ij} L_{jd}(\mathbf{P})}{\sum_{j\in\mathcal{S}_i} p_{ij}\lambda_{ij}} \tag{3.2}$$

Based on the expression derived in Lemma 1 (see Section 3.8.1 for the proof), the expected latency of each node to the destination depends on the expected latencies of its neighbours. This result raises a substantial question: Does the probability decision matrix that minimizes the sum of expected latencies of all nodes of the network, $\mathbf{P}^*$, also minimize the expected latency of each individual node? Before continuing to propose algorithms for finding $\mathbf{P}^*$, we answer this question and make two points about the structure of $\mathbf{P}^*$ through the following theorem. The proof is provided in Section 3.8.2.

Theorem 1. *Suppose* $\mathbf{P}^* = \arg\min_{\mathbf{P}\in[0,1]^{N\times N}} \sum_{i=1}^{N} L_{id}(\mathbf{P})$. *Then:*

1. $\mathbf{P}^*$ *is a binary matrix (its components are either* 0 *or* 1*).*
2. *For any $i \in \mathcal{N}$, the matrix $\mathbf{P}^*$ also minimizes $L_{id}(\mathbf{P})$:*

$$\forall i \in \mathcal{N}: \quad \mathbf{P}^* = \arg \min_{\mathbf{P}\in[0,1]^{N\times N}} L_{id}(\mathbf{P}) \tag{3.3}$$

Theorem 1 shows that the minimization problem is actually a binary problem. Each time node i meets one of its neighbours $j \in \mathcal{S}_i$, it either forwards the message or keeps it. From now on, we change our notation and use the binary indicator matrix $\mathbf{B}$ instead of $\mathbf{P}$ to capture this binary decision. Therefore, the optimization takes the form:

$$\mathbf{B}^* = \arg \min_{\mathbf{B} \in \{0,1\}^{N \times N}} \sum_{i=1}^{N} L_{id}(\mathbf{B}) \tag{3.4}$$

Theorem 1 also states that the optimum solution matrix $\mathbf{B}^*$ can be equivalently achieved by minimizing the expected latency of each of the network nodes to the destination. This is the main idea of developing centralized and decentralized algorithms for finding $\mathbf{B}^*$. In the next three sections, we introduce the algorithms we have proposed for solving this optimization problem and prove that they find the optimal solution.

3.3 Centralized Approach with Global Knowledge

Suppose for each node $i \in \mathcal{N}$, the set of neighbours $\mathcal{S}_i$ and their meeting rates $\lambda_{ij}, j \in \mathcal{S}_i$, are known at a central calculation unit. Algorithm 1 presents an iterative procedure to identify a binary decision matrix $\mathbf{B}$. In this algorithm, we first decide on the forwarding rules of the node that has the most frequent direct contacts with the destination. We refer to this node as node A. In order to achieve the minimum expected latency to the destination, node A should forward its generated or received messages only to the destination, ignoring its meetings with any other nodes. All other nodes that A encounters meet the destination less frequently and, if they forward their messages to the destination through other nodes, these other nodes also meet the destination less frequently than A. In subsequent steps of the algorithm, we consider all the nodes that have direct contacts with the nodes whose forwarding decision rules have already been made (the set $\mathcal{A}$) and calculate the minimum latency that each of them can obtain by forwarding through nodes in $\mathcal{A}$ to the destination. At the end of each iteration, we finalize the forwarding decision for the one node that can achieve the minimum latency and add it to $\mathcal{A}$. We repeat the procedure until the decision has been made for all the nodes of the network and the elements of the binary matrix $\mathbf{B}$ have all been specified. The next theorem states that the binary matrix $\mathbf{B}$ resulting from applying this procedure, as specified concretely in Algorithm 1, achieves the minimum sum

Algorithm 1 Centralized Greedy Latency Minimization with Global Knowledge

1: `// Initialization`
2: $\mathcal{A} = \{d\}$, $\mathbf{B} = \mathbf{0}_{N\times N}$, $L_{dd} = 0$, $L_{jd} = \infty$ for $j \neq d$
3: `// Iteratively add nodes to the set` $\mathcal{A}$
4: **while** $\mathcal{A} \neq \mathcal{N}$ **do**
5: **for** each node $i \notin \mathcal{A}$ **do**
6: Identify $\mathcal{S}_{i\mathcal{A}} = \mathcal{S}_i \cap \mathcal{A}$
7: Calculate $G_{id} = \min_{\mathbf{m}_i \in \{0,1\}^{|\mathcal{S}_{i\mathcal{A}}|}} \frac{1+\sum_{j\in\mathcal{S}_{i\mathcal{A}}} m_{ij}\lambda_{ij}L_{jd}}{\sum_{j\in\mathcal{S}_{i\mathcal{A}}} m_{ij}\lambda_{ij}}$ and identify the minimizing $\mathbf{m}_i^*$
8: Denote $\mathcal{D}_i = \{j | j \in \mathcal{S}_{i\mathcal{A}}, m_{ij}^* = 1\}$
9: **end for**
10: Identify $v = \arg\min_{i\in\mathcal{N}/\mathcal{A}} G_{id}$
11: Set $L_{vd} = G_{vd}$
12: Set $b_{vj} = 1$ for all $j \in \mathcal{D}_i$
13: Set $\mathcal{A} = \mathcal{A} \cup \{v\}$
14: **end while**

of expected latencies to the destination. The proof can be found in Section 3.8.3.

Theorem 2. *Suppose all meeting rates are different and there exists a unique solution* $\mathbf{B}^*$ *for the optimization problem* (3.4)*.*

1. *After each iteration of Algorithm 1,*
   (a) $\forall i \in \mathcal{A}, \forall j \in \mathcal{N}: \quad b_{ij} = b_{ij}^*$
   (b) $\forall i \in \mathcal{A}: \quad L_{id}(\mathbf{B}) = L_{id}(\mathbf{B}^*)$
   (c) $\max_{i\in\mathcal{A}} L_{id}(\mathbf{B}^*) < \min_{i\notin\mathcal{A}} L_{id}(\mathbf{B}^*)$
2. *Upon completion, Algorithm 1 identifies a labelling* $\mathbf{B}$ *and associated expected latencies* L_{id} *such that* $\mathbf{B} = \mathbf{B}^*$

Theorem 2 demonstrates that the iterative optimization procedure expressed in Algorithm 1 finds the solution of the minimization problem in (3.4). If there is not a unique

solution, then at some point in Algorithm 1, there will be multiple $\mathbf{m}_i$s that solve the optimization in line 7. It is straightforward to show that choosing any one of these $\mathbf{m}_i$ vectors leads to a decision matrix $\mathbf{B}$ that achieves the minimum expected latencies.

3.4 Decentralized Approach with Partial a priori Knowledge

Suppose no central unit exists and each node is just aware of its own $\mathcal{S}_i$ and the meeting rates $\lambda_{ij}, j \in \mathcal{S}_i$. Algorithm 2 demonstrates how nodes can make their binary forwarding decisions based on this local information. Since the expected latency of each node depends on the expected latency values of its neighbours, nodes need to have estimates of their neighbours' expected latencies to be able to make forwarding decisions. We denote by $\hat{L}_{id}(j)$ the estimate at node j of the expected latency from node i to the destination. In Algorithm 2, each time two nodes meet, they update these estimates and then recalculate their optimum forwarding rules. Theorem 3 proves that this decentralized approach results in the same global optimum solution. The proof of Theorem 3 is provided in Section 3.8.4. In this proof, we assume that the nodes are labelled in order of ascending expected latency under $\mathbf{B}^*$, i.e.,

$$L_{1d}(\mathbf{B}^*) \leq L_{2d}(\mathbf{B}^*) \leq \cdots \leq L_{(N-1)d}(\mathbf{B}^*) \tag{3.5}$$

We also define a new set of parameters T_k, $k = 1, \ldots, N$. These parameters will be later used to prove the convergence of Algorithm 2. We define T_1 as the moment of time at which node 1 meets the destination node for the first time. Similarly, we define T_k as the earliest time by which node $k > 1$ has met all nodes in the set $\{1, \ldots, k-1\} \cap \mathcal{S}_k$ in the time period $(T_{k-1}, T_k]$.

Theorem 3. *The decision matrix* $\mathbf{B}$ *identified by Algorithm 2 converges to* $\mathbf{B}^*$ *with probability* 1.

We refer to our proposed decentralized greedy latency minimization algorithm (Algorithm 2) as MinLat and evaluate its efficiency in different random and real-world networks based on certain performance metrics in Section 3.6. Regarding the computational complexity of finding the minimum expected latency in MinLat, the following lemma shows that the optimizations in lines 8 and 9 of this algorithm are linear fractional programs and can be solved quickly using variants from linear programming. Further details are available in Section 3.8.5.

Algorithm 2 MinLat: Decentralized Greedy Latency Minimization with Partial a priori Knowledge

1: `// Initialization`
2: $\mathbf{B} = \mathbf{0}_{N\times N}$
3: $\forall i \in \mathcal{N}/d, \forall j \in \mathcal{N}: \quad \hat{L}_{dd}(j) = 0,\ \hat{L}_{id}(j) = \infty$
4: **while** Nodes continue to meet **do**
5: `// Nodes` i `and` j `meet at time` t
6: Set $\hat{L}_{id}(j) = \hat{L}_{id}(i)$
7: Set $\hat{L}_{jd}(i) = \hat{L}_{jd}(j)$
8: Update $\hat{L}_{id}(i) = \min_{\mathbf{m}_i\in\{0,1\}^{|\mathcal{S}_i|}} \frac{1+\sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}\hat{L}_{kd}(i)}{\sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}}$ and identify the minimizing $\mathbf{m}_i^*$
9: Update $\hat{L}_{jd}(j) = \min_{\mathbf{m}_j\in\{0,1\}^{|\mathcal{S}_j|}} \frac{1+\sum_{k\in\mathcal{S}_j} m_{jk}\lambda_{jk}\hat{L}_{kd}(j)}{\sum_{k\in\mathcal{S}_j} m_{jk}\lambda_{jk}}$ and identify the minimizing $\mathbf{m}_j^*$
10: Set $\mathbf{b}_i = \mathbf{m}_i^*$ and $\mathbf{b}_j = \mathbf{m}_j^*$
11: **end while**

Lemma 2. *The minimization problem in Algorithm 2,*

$$\hat{L}_{id}(i) = \min_{\mathbf{m}_i\in\{0,1\}^{|\mathcal{S}_i|}} \frac{1+\sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}\hat{L}_{kd}(i)}{\sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}} \ , \tag{3.6}$$

can be converted to a linear programming problem.

Assuming that (3.6) can be solved in polynomial order $P(|S_i|)$, the worst case complexity order of Algorithm 1 is $O(N^2)P(N)$ because in the ith round of this algorithm, (3.6) should be solved for each of the $N-i$ nodes that are not in the set $\mathcal{A}$. In Algorithm 2, each time node i meets one of its neighbours, it solves a problem of complexity $P(|S_i|)$. The only information that a node needs to share when it meets another node is its estimate of its own expected latency to the destination. In the general case where messages can be destined to any node in the network, this exchangable message could be a length N vector of expected latencies to all nodes.

The following proposition provides a bound on the expected convergence time of the MinLat Algorithm (Algorithm 2). The brief proof is provided in Section 3.8.4. The bound

depends on the slowest meeting rate between each node and its candidate relay nodes. This is a conservative bound, since in practice, a node only needs to meet the relay nodes to which it actually forwards data under the optimum forwarding rule.

Proposition 1. *Assuming that the nodes are labelled in order of ascending expected latency under* $\mathbf{B}^*$*, the expected convergence time,* $\mathbb{E}(T_N)$*, of Algorithm 2 is bounded as*

$$\mathbb{E}(T_N) < \frac{1}{\lambda_{1d}} + \sum_{l=2}^{N} \frac{l-1}{\min\limits_{\substack{i\in\{1,\ldots,l-1\}\\ \lambda_{li}>0}} \lambda_{li}} \tag{3.7}$$

3.5 Decentralized Approach with No a priori Knowledge

In Section 3.5, we assumed that as soon as a node meets another node, it has a perfect knowledge of its meeting rate with that node. In practice, a node will need to estimate its meeting rates with the neighbours and periodically revise the estimation as meetings occur (or fail to occur). Consider an arbitrary pair of nodes i and j that meet each other with rate λ_{ij}. As mentioned in Section 3.2, we denote the k^{th} inter-meeting time between i and j, which is the time between k^{th} and $k+1^{\text{th}}$ meetings, by $x_{ij}^k > 0$. After n_{ij} meetings between i and j, the set $\{x_{ij}^1, \ldots, x_{ij}^{n_{ij}}\}$ is a set of independent samples of an exponentially distributed random variable with parameter λ_{ij}.

In this section, we develop a more practical version of the decentralized MinLat algorithm called MinLat-E. In MinLat-E, nodes do not know their meeting rates and use Maximum Likelihood (ML) estimation to estimate them. We denote the ML estimate of the meeting rate between nodes i and j after n_{ij} meetings by $\hat{\lambda}_{ij}^{n_{ij}}$. This estimate is by definition the meeting rate that maximizes the likelihood function

$$\mathcal{L}(x_{ij}^1, .., x_{ij}^{n_{ij}}; \lambda_{ij}) = f(x_{ij}^1, .., x_{ij}^{n_{ij}} | \lambda_{ij}) = \lambda_{ij}^{n_{ij}} e^{-\lambda_{ij} \sum_{k=1}^{n_{ij}} x_{ij}^k} \tag{3.8}$$

Therefore, after n_{ij} meetings, nodes i and j estimate their meeting rate as

$$\hat{\lambda}_{ij}^{n_{ij}} = \frac{n_{ij}}{\sum_{k=1}^{n_{ij}} x_{ij}^k} \tag{3.9}$$

Hence, under the exponential model, a node only needs to remember the last time it met its neighbour, l_{ij}, and the number of times it has met that neighbour, n_{ij}. With these

two pieces of information, it can update its estimation of the meeting rate ($\hat{\lambda}_{ij}^{n_{ij}}$) from the previously estimated value ($\hat{\lambda}_{ij}^{n_{ij}-1}$) using the following equation.

$$\hat{\lambda}_{ij}^{n_{ij}} = \frac{n_{ij}\hat{\lambda}_{ij}^{n_{ij}-1}}{n_{ij} - 1 + \hat{\lambda}_{ij}^{n_{ij}-1}(t - l_{ij})} \tag{3.10}$$

where t denotes the time since the network began operating.

As shown in Algorithm 3, MinLat-E solves the similar optimization problems as in MinLat using the estimated meeting rates. In this algorithm, the decision matrix achieved by MinLat-E at time t is denoted by $\widetilde{\mathbf{B}}_t$. The i^{th} row of this decision matrix is denoted by $\widetilde{\mathbf{B}}_t^i$. Further, $\widetilde{L}_{jd,t}(\widetilde{\mathbf{B}}_t, i)$ denotes the estimate at node i at time t of the expected latency of node j to the destination, when the forwarding decision matrix is $\widetilde{\mathbf{B}}_t$. This estimate differs from the one obtained in Algorithm 2 , $\hat{L}_{id}(i)$, because MinLat calculates them using estimated meeting rates $\hat{\lambda}_{ij}^{n_{ij}}$.

The main question here is if MinLat-E converges to the same expected latencies as MinLat does. The following theorem answers this question. Before discussing the details of this theorem, we distinguish between two terms: *estimated expected latencies* and *achieved expected latencies*. The term "estimated expected latencies" is used to refer to the expected latencies, $\widetilde{L}_{jd,t}(\widetilde{\mathbf{B}}_t, i)$, that MinLat-E calculates using the estimated meeting rates and the decision matrix $\widetilde{\mathbf{B}}_t$. However, the achieved expected latencies are the latencies that are derived using the true meeting rates and the decision matrix $\widetilde{\mathbf{B}}_t$. We denote the achieved expected latencies by $L_{id}(\widetilde{\mathbf{B}}_t)$. Theorem 4 states that both the achieved and the estimated expected latencies converge in probability to the optimum expected latencies $L_{id}(\mathbf{B}^*)$. The proof is provided in Section 3.8.6.

Theorem 4. *For any node in the network, the sequences of estimated and achieved latencies converge to its optimum expected latency in probability, i.e.,* $\{\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i)\} \xrightarrow{p} L_{id}(\mathbf{B}^*)$ *and* $\{L_{id}(\widetilde{\mathbf{B}}_t)\} \xrightarrow{p} L_{id}(\mathbf{B}^*)$. *More precisely for any* $\epsilon > 0$,

$$\lim_{t->\infty} P(|\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i) - L_{id}(\mathbf{B}^*)| < \epsilon) = 1 \tag{3.11}$$

$$\lim_{t->\infty} P(|L_{id}(\widetilde{\mathbf{B}}_t) - L_{id}(\mathbf{B}^*)| < \epsilon) = 1 \tag{3.12}$$

We check the claims of Theorem 4 and investigate the convergence speed of MinLat-E through simulations in Section 3.6.

Algorithm 3 MinLat-E: Decentralized Greedy Latency Minimization with no a priori Knowledge

// Initialization

$\widetilde{\mathbf{B}}_t = \mathbf{0}_{N\times N}$

$\forall i,j \in \mathcal{N}: \quad n_{ij} = 0, l_{ij} = 0$

$\forall i \in \mathcal{N}/d, \forall j \in \mathcal{N}: \quad \widetilde{L}_{dd,t}(\widetilde{\mathbf{B}}_t, j) = 0,\ \widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, j) = \infty$

while Nodes continue to meet **do**

  // Nodes i and j meet at time t

  Set $n_{ij} = n_{ij} + 1$

  Update $\hat{\lambda}_{ij}^{n_{ij}}$ using (3.10)

  Set $\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, j) = \widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i)$

  Set $\widetilde{L}_{jd,t}(\widetilde{\mathbf{B}}_t, i) = \widetilde{L}_{jd,t}(\widetilde{\mathbf{B}}_t, j)$

  Update $\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i) = \min_{\mathbf{m}_i \in \{0,1\}^{|\mathcal{S}_i|}} \frac{1+\sum_{k\in\mathcal{S}_i} m_{ik}\hat{\lambda}_{ik}^{n_{ik}} \widetilde{L}_{kd,t}(\widetilde{\mathbf{B}}_t,i)}{\sum_{k\in\mathcal{S}_i} m_{ik}\hat{\lambda}_{ik}^{n_{ik}}}$ and identify the minimizing $\mathbf{m}_i^*$

  Update $\widetilde{L}_{jd,t}(\widetilde{\mathbf{B}}_t, j) = \min_{\mathbf{m}_j \in \{0,1\}^{|\mathcal{S}_j|}} \frac{1+\sum_{k\in\mathcal{S}_j} m_{jk}\hat{\lambda}_{jk}^{n_{jk}} \widetilde{L}_{kd,t}(\widetilde{\mathbf{B}}_t,j)}{\sum_{k\in\mathcal{S}_j} m_{jk}\hat{\lambda}_{jk}^{n_{jk}}}$ and identify the minimizing $\mathbf{m}_j^*$

  Set $\widetilde{\mathbf{b}}_t^i = \mathbf{m}_i^*$ and $\widetilde{\mathbf{b}}_t^i = \mathbf{m}_j^*$

  Set $l_{ij} = t$

end while

3.6 Simulation Results

In this section, we investigate the efficiency of our proposed approach in modelling and solving the forwarding/routing problem in different opportunistic network scenarios. We first evaluate the performance of the proposed algorithms on ideal network scenarios where there is no restriction on message life times, buffer sizes and data exchanges in Section 3.6.1. We then study the network behaviour when these practical challenges are added to the simulation set ups in Section 3.6.2.

Expected latency values shown in the simulation results only reflect the delay caused

by carrying the message around before meeting a relay node in the store-carry-forward mechanism. They do not include other sources of delay such as wireless transmission delay, queuing delay, or the delay caused by exchanging the routing information.

3.6.1 Idealized Network Models

We test our algorithms using three different networks to model the contacts between $N = 41$ mobile nodes. The characteristics of the networks are derived from the Infocom05 dataset [106]. This data set is based on an experiment conducted during the IEEE Infocom 2005 conference in Miami where 41 Bluetooth enabled devices (Intel iMote) were carried by attendees for 3 days. The start and end times of each contact between participants were recorded. The average time between node contacts in the Infocom05 dataset is 1.3×10^4 seconds (3.7 hours). In our processing, we only consider the contacts in which both devices recognized each other so that an acknowledged message could be transferred between them.

In the first network, (*Net I*), we construct a contact graph using an evolving undirected network model based on the preferential attachment mechanism. [107] studies how opportunistic networks might be modelled as scale-free networks. We start with a small fully connected graph of $m_0 = 5$ vertices and add vertices to it one by one until the graph consists of $N = 41$ nodes. At each step, the new vertex is connected to $m = 5$ previously existing vertices. The probability that the new vertex is connected to vertex i is $\frac{k_i}{\sum_j k_j}$ where k_i is the degree of i up to this stage. After building the contact graph, we assign a parameter λ_{ij} to each pair of nodes i and j which are connected in the contact graph and assume that they meet with exponentially distributed inter-meeting times with parameter λ_{ij}. We choose the parameters λ_{ij} from a uniform distribution with the same expectation as the average of node meeting rates observed in the Infocom05 dataset.

In *Net II*, we set λ_{ij} to be equal to the inverse of the average inter-meeting time between nodes i and j in the Infocom05 dataset. We are interested in the behaviour of the algorithms in relatively sparse networks, so we limit the number of neighbours of each node: node i is only connected to node j in the contact graph if the meeting rate λ_{ij} is among the largest $K = 10$ meeting rates of either node i or node j. In our simulations, the meeting times between nodes i and j for *Net II* are then chosen from an exponential distribution with parameter λ_{ij}. In the third experimental network, *Net III*, we use the actual meeting times recorded in the Infocom05 dataset. The analysis in [103] indicates that the distribution of

individual inter-meeting times for most pairs of nodes can be approximated reasonably well by an exponential distribution; on the other hand, the aggregate distribution of contact times shows heavy-tailed behaviour and is better approximated using a truncated power distribution [100, 101]. Table 3.2 summarizes the properties of the test networks.

Table 3.2 Test network properties

Network	Contact Graph	Parameters	Inter-meeting Times	Number of nodes
I	Preferential Attachment	$m_0 = 5$ $m = 5$	Exponential λ: uniform	41
II	Sparsified Infocom	$K = 10$	Exponential λ: dataset	41
III	Sparsified Infocom	$K = 10$	Dataset times	41

As mentioned in Section 3.4, we call our proposed decentralized greedy latency minimization algorithm MinLat and refer to its more practical version with meeting rate estimations as MinLat-E. In these two algorithms, the decisions that nodes make for future forwarding rules depend on the (estimated) meeting rates, which are derived from the frequency of past contacts between nodes. Thus, MinLat and MinLat-E can be identified as history-based routing algorithms. In order to evaluate their performance, we compare them with existing history-based routing protocols that can be implemented in a distributed fashion and do not need a priori knowledge of the network topology. As mentioned in Section 3.2, the fixed point algorithm proposed in [47] is proved to result in the minimum expected latency (which is expected to be the same as the result of our proposed centralized Algorithm 1). However, the proposed algorithm in [47] is centralized and needs to be performed in a control unit where the whole topology of the network is known. The TOUR algorithm proposed in [48] is decentralized, but each node needs perfect a priori knowledge of its meeting rates with other nodes. Also, the main focus of TOUR is to find the optimum way to make forwarding decisions based on the priorities of messages. We have chosen PRoPHETv2 [27] and MaxProp [28] as the most appropriate candidates for comparison. We also compare to Epidemic routing (ER) [14], which is expected to result in a high delivery probability at the expense of high usage of network resources. The parameters of PRoPHETv2 are set to those suggested in [25] and [27], i.e., $P_{init} = 0.75$, $\beta = 0.25$, $\gamma = 0.98$, and time step = 1. In order to put the focus on evaluation of the performance

efficiency of forwarding rules and eliminate the effect of the buffer management technique on this performance, the dropping rules proposed for MaxProp in [28] are not applied.

We divide the Infocom05 dataset into slots of 12 hours. In each of these time slots, we build networks *I* to *III* using the nodes that are present in that period. The intermeeting time exponential parameters (λ_{ij}s) are estimated based on the meetings that occurred in that specific time slot and networks *I* and *II* are constructed using these estimated parameters. For each of the first four 12-hour periods (the first two days of the conference), we send 1000 messages, spaced by 5-second intervals, from randomly chosen source nodes to a particular destination. We terminate the simulation at the end of the 3-day period, and calculate the fraction of messages successfully delivered by each of the single copy (PRoPHETv2, MinLat) and multi-copy (Epidemic, MaxProp) forwarding algorithms. For each algorithm, we also calculate the average latency of the messages that are delivered by all four algorithms; the average number of hops that messages pass to reach the destination; and the average buffer occupancy of the nodes. For each of the 41 nodes of the network, we calculate the average of performance metrics over the time slots when that node has been chosen as a destination. Figure 3.1 shows the average and the 95% confidence intervals of the four performance metrics for different destinations in the three test networks using Epidemic, PRoPHETv2, MaxProp, and MinLat forwarding algorithms. There is no restriction on message life time or buffer size that can cause message dropping in these simulations. However, the delivery rates in some cases are less than 1 because the simulations are terminated before all of the generated messages are successfully delivered. We observe that in all the three test networks, MinLat has a better performance than the other existing history-base single copy algorithm, PRoPHETv2, in terms of delivery rate and average latency. Its performance is also comparable to MaxProp which is a history-based multi-copy algorithm. Also, noting that the scale of the vertical axis of Figure 3.1d is logarithmic, we see that MinLat occupies much less memory of nodes' buffers on average. For networks *I* and *II*, where the assumption of exponentially distributed inter-meeting times holds, delivering a higher rate of messages with lower average latencies than PRoPHETv2 is expected from Theorem 3. However, we observe that this result also holds for network *III* where the actual meeting times are used. All algorithms display slightly poorer performance in network *III*; this is probably due to heavy-tailed and non-stationary inter-meeting times.

In order to explore how the incorporation of meeting rate estimations in MinLat-E affects the message delivery performance, we conduct further simulations with a different

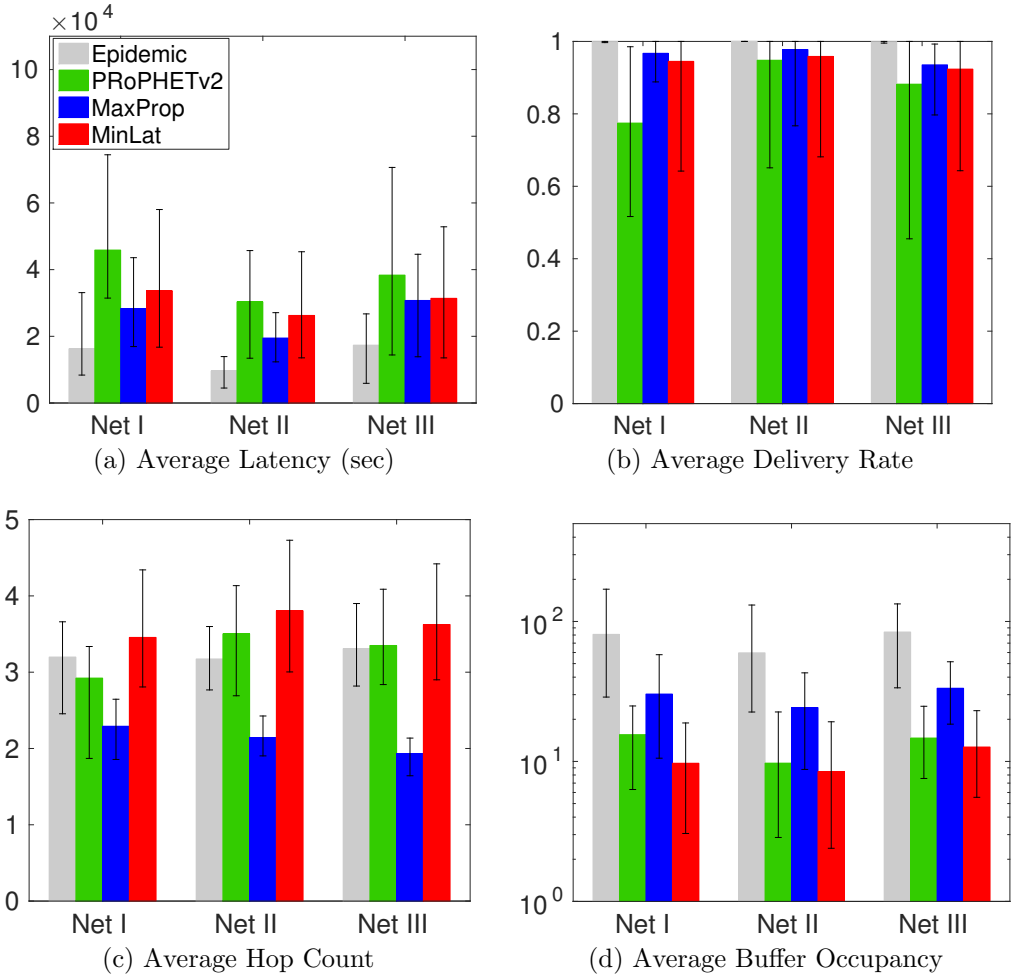
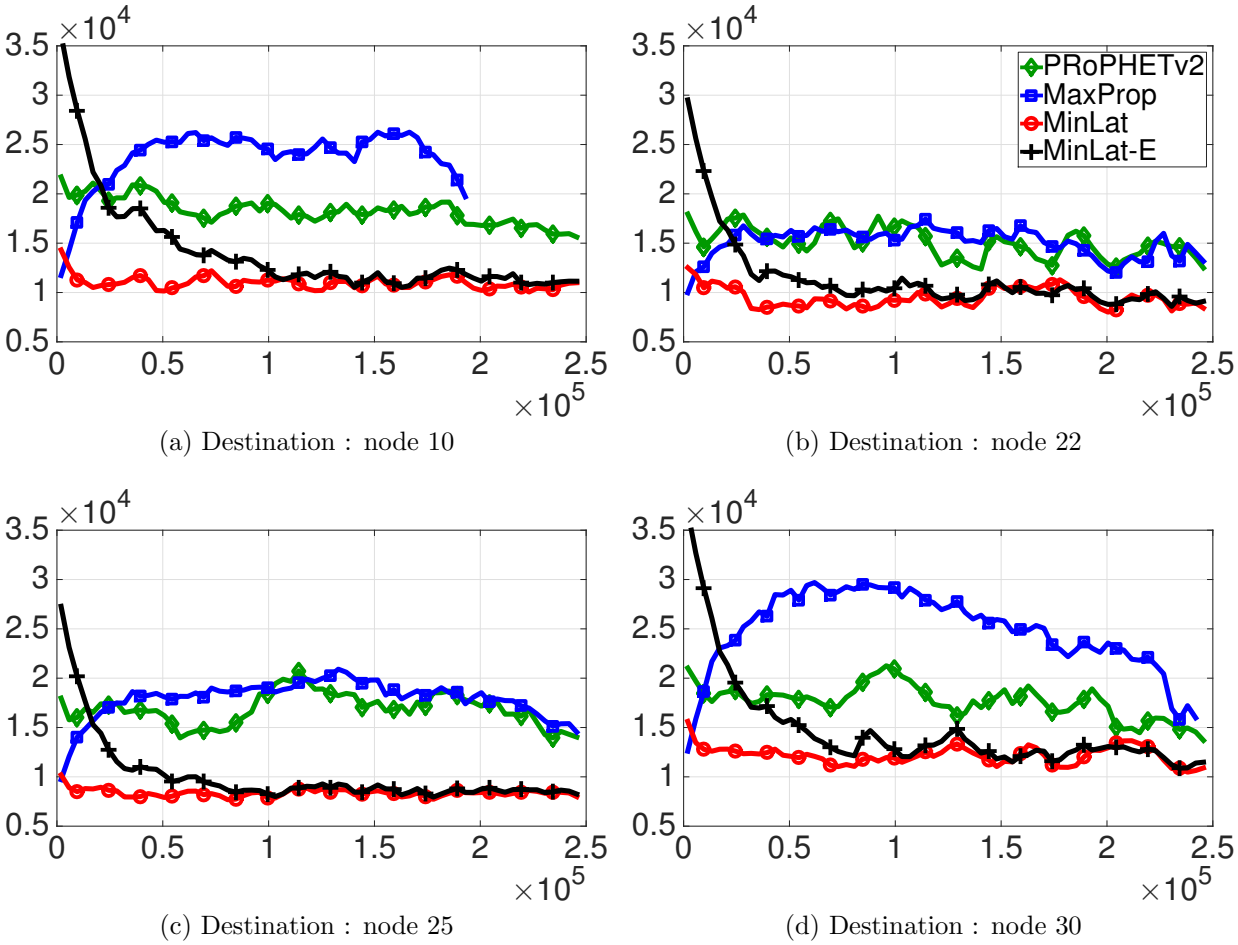


Figure 3.1 Comparison metrics in test networks with $N = 41$

message generation scenario on network *II*. Figure 3.2 displays the average delivery latency as a function of time for the history-based routing algorithms PRoPHETv2, MaxProp, MinLat, and MinLat-E. The delivery latency values are averaged over 32 simulation runs while the destination node and message generation times are fixed in all the 32 runs. The simulation is repeated for four different destinations. Messages are generated with inter-arrival time of t seconds at randomly chosen source nodes where t is uniformly distributed in $[0, 5]$. Each point on the curves represents the average latency of the 1500 most recently sent messages. As Figure 3.2 shows, in all of the three single copy routing algorithms (PRoPHETv2, MinLat, MinLat-E), the average time it takes for a message to be delivered at the destination decreases as time goes by. This decreasing trend is due to the fact that

the forwarding rules discovered by nodes improve as the nodes have more contacts and their information concerning their neighbours' message delivery capabilities becomes more accurate. However, in the multi-copy routing algorithm, MaxProp, the average message delivery latency increases in the beginning. In MaxProp, the weights assigned to the links are initialized to be equal, which means that nodes forward messages to more of their neighbours. There is thus a high level of message replication which leads to messages reaching the destination sooner on average. As time passes, the level of replication decreases and the average delivery latency increases. We also observe that MinLat-E and MinLat eventually achieve the same average delivery latencies, as expected from Theorem 4. However, the convergence to the optimum point is slower in MinLat-E due to the time it takes for nodes to obtain accurate estimates of their meeting rates.

As proved in Theorem 4, estimated and achieved expected latencies in MinLat-E converge to the optimum expected latencies of MinLat in probability. Our next set of simulations shows how the differences between these two expected latencies and the optimum values decrease over time. In addition to the decentralized scheme (MinLat and MinLat-E), we add the recursive maximum likelihood estimation of meeting rates to the centralized scheme (Algorithm 1) as well. Each time two nodes meet and update their estimations of their meeting rates, we run Algorithm 1 with the new set of estimated meeting rates. We conduct experiments using both the centralized and decentralized algorithms operating on the extension of network I to 100 nodes. We select the λ parameters of inter-meeting times from a uniform distribution $U[0, 0.01]$. The destination node is randomly chosen from the N nodes of the network based on a uniform distribution and is fixed throughout the simulation. We run the simulation for 5×10^4 seconds and examine some statistical characteristics of the discrepancies discussed in Theorem 4 over all nodes of the network. Figure 3.3a shows the median, the 25^{th} and 75^{th} percentiles, the highest values, and the outliers of absolute differences between the estimated expected latencies and the minimum expected latencies, i.e., $|\widetilde{L}_{kd,t}(\widetilde{\mathbf{B}}_t, k) - L_{kd}(\mathbf{B}^*)|$ for all $k \in \mathcal{N}$. The central mark of each box is the median, the edges of the box are the 25^{th} and 75^{th} percentiles, the whiskers extend to the highest values not considered as outliers, and outliers are plotted individually. Figure 3.3b shows the same characteristics for absolute differences between the achieved expected latencies and the minimum expected latencies, i.e., $|L_{kd}(\widetilde{\mathbf{B}}_t) - L_{kd}(\mathbf{B}^*)|$ for all $k \in \mathcal{N}$. As expected from Theorem 4, we see that the estimated and achieved discrepancies both decay to zero as the time goes by (for achieved latencies, it is almost zero for all nodes



(a) Destination : node 10

(b) Destination : node 22

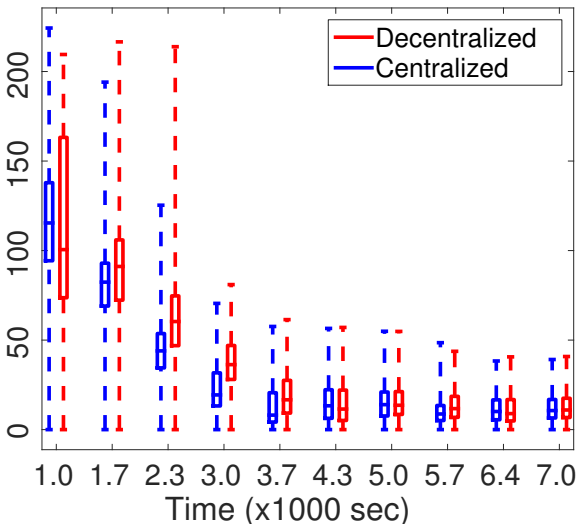
(c) Destination : node 25

(d) Destination : node 30

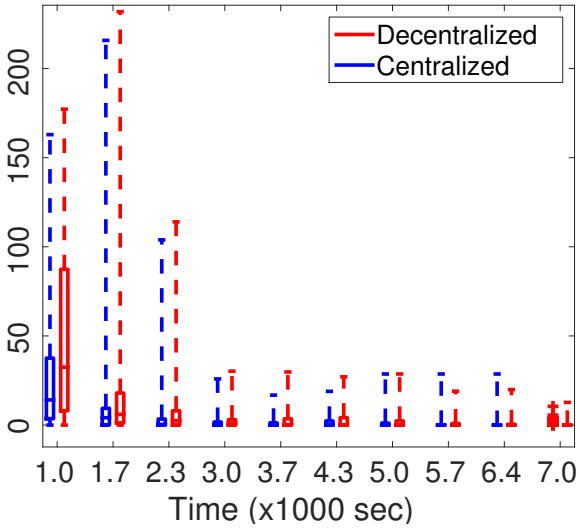
Figure 3.2 Evolution of average delivery latency with time (both in seconds)

of the network after $t = 7000$ seconds). Figure 3.3 indicates that after a certain time the decentralized algorithm achieves almost the same difference between estimated (achieved) and optimum expected latencies as the centralized algorithm. This suggests that the limiting factor is the convergence of the meeting rate estimates rather than the dissemination of latency estimates through the network.

Comparing Figures 3.3a and 3.3b shows that for both centralized and decentralized scenarios, the average difference between the achieved latency and the minimum latency is less than the average difference between the estimated latency and the minimum latency. This is expected, because the estimated latencies are based on incorrect decision matrices and estimated meeting rates, whereas the achieved latencies are derived from the actual



(a) Absolute Difference between Estimated and Minimum Expected Latencies (seconds)



(b) Absolute Difference between Achieved and Minimum Expected Latencies (seconds)

Figure 3.3 Some statistical features of the absolute difference between estimated and achieved latencies and the minimum expected latencies

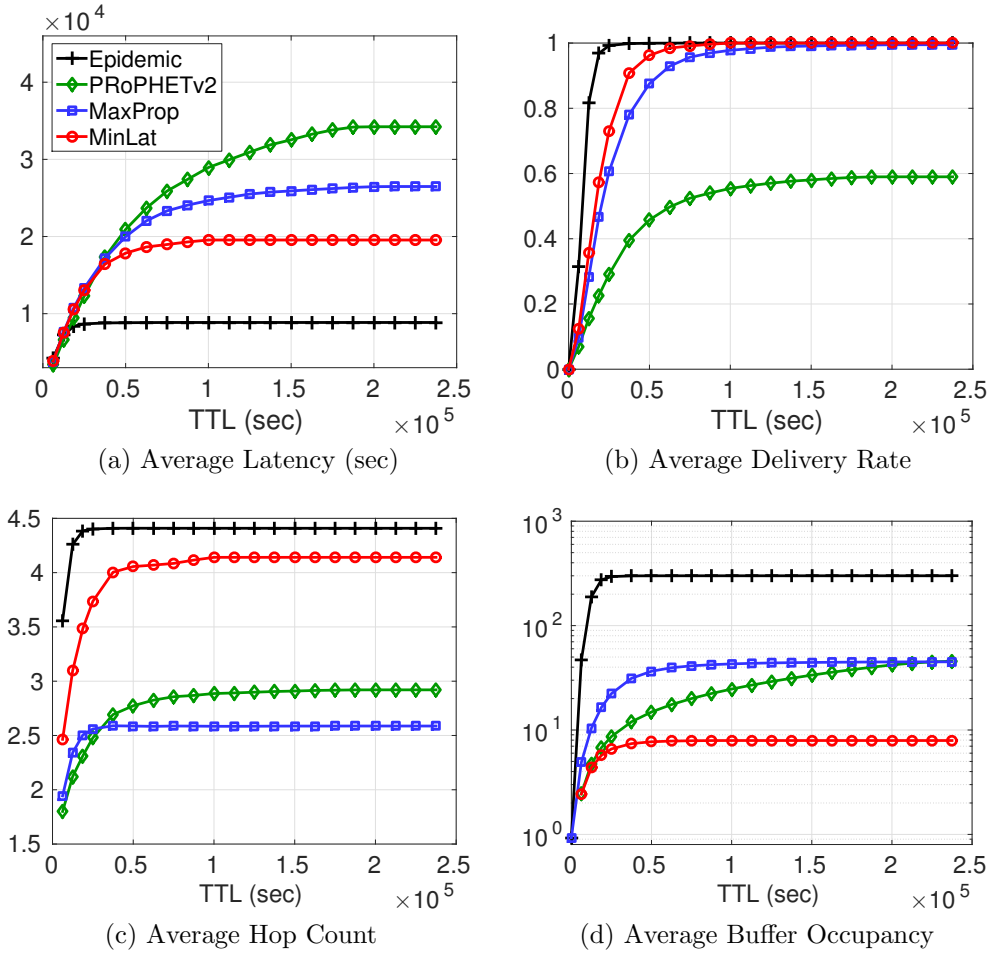
meeting rates. The results suggest that even when there remains substantial inaccuracy in the expected latency estimates (e.g. at time $t = 3000$ seconds), the algorithm identifies a close-to-optimal forwarding decision matrix.

3.6.2 Realistic Network Models

We examine the performance of the forwarding algorithms in larger networks by extending network I to 100 nodes but with the same average λ parameter for exponential inter-meeting times. We also make our model more realistic by adding some practical restrictions to the network model. First, we assume that messages have finite TTL, i.e., a message is discarded when its lifetime exceeds a certain threshold. Figure 3.4 displays the performance of routing algorithms for different values of TTL varying from 0 to as large as the simulation time (almost 2.5×10^5 seconds). Simulations are run 100 times and in each round, a different destination is randomly chosen from network nodes based on a uniform distribution. The average latency and average hop count are calculated only for the messages that reach the destination.

The simulation results in Figure 3.4b show that decreasing the TTL has a similar

overall effect on all of the algorithms. For larger TTLs, the delivery rate increases, but the buffer occupancy, average latency and hop count also increase. MinLat outperforms PRoPHET and MaxProp in terms of delivery rate, average latency, and buffer occupancy even in a scenario with a restricted message life time. Although inter-meeting times are exponentially distributed and the contact graph is based on preferential attachment, in this larger network of 100 nodes, PRoPHET cannot reach 100 percent delivery rate even without any restriction on TTL.



(a) Average Latency (sec)

(b) Average Delivery Rate

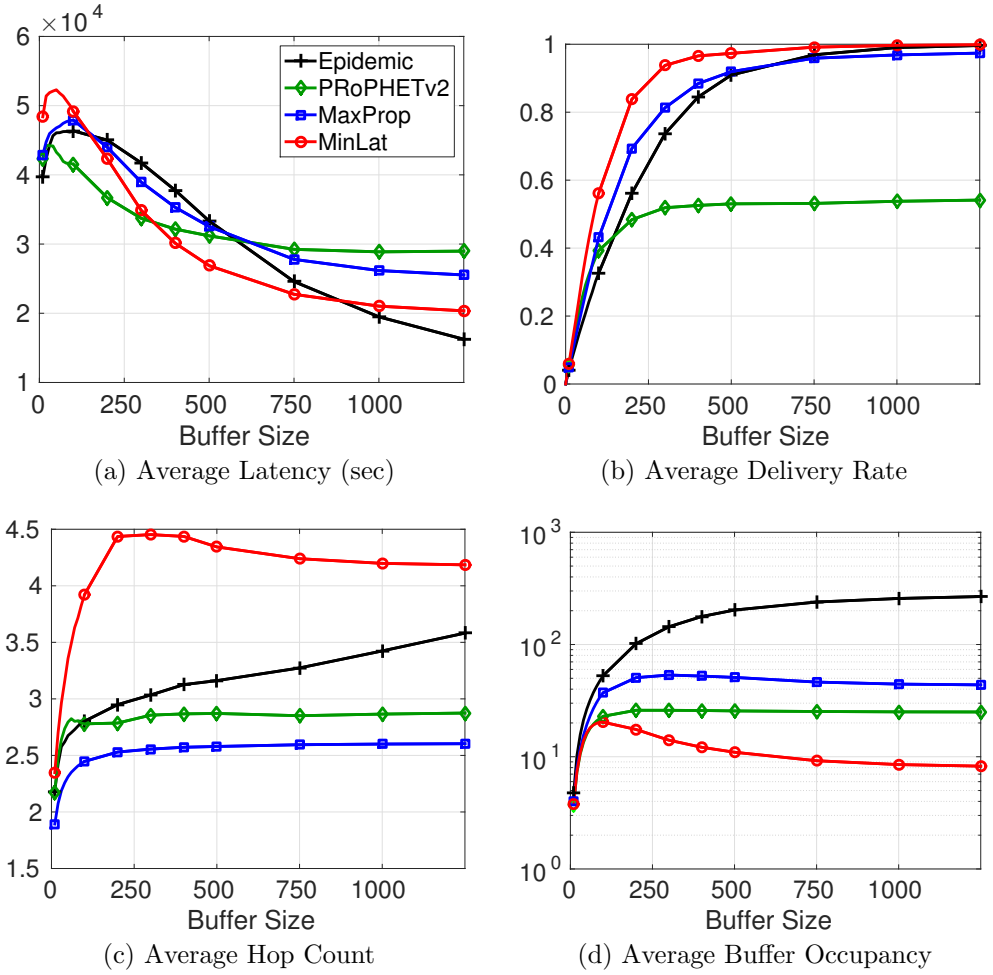
(c) Average Hop Count

(d) Average Buffer Occupancy

Figure 3.4 Effect of TTL on performance metrics

The next step towards a more realistic network model is to consider limits on buffer size. In the next set of simulations, we assume that TTL is 10^5 seconds so that all algorithms reach their best possible delivery rate. We also assume that each node has a limited capacity

for keeping the messages. When the buffer occupancy of a node reaches its limit, messages from other nodes are not forwarded to it. Moreover, any generated messages at the fully occupied node are immediately dropped. Figure 3.5 shows the performance of algorithms for buffer sizes in the range of 0 to 1250 messages. We see that increasing the buffer size improves the delivery rate for all algorithms. It also reduces the average latency because as the buffer size increases, nodes can more frequently follow their optimum forwarding rules.



(a) Average Latency (sec)

(b) Average Delivery Rate

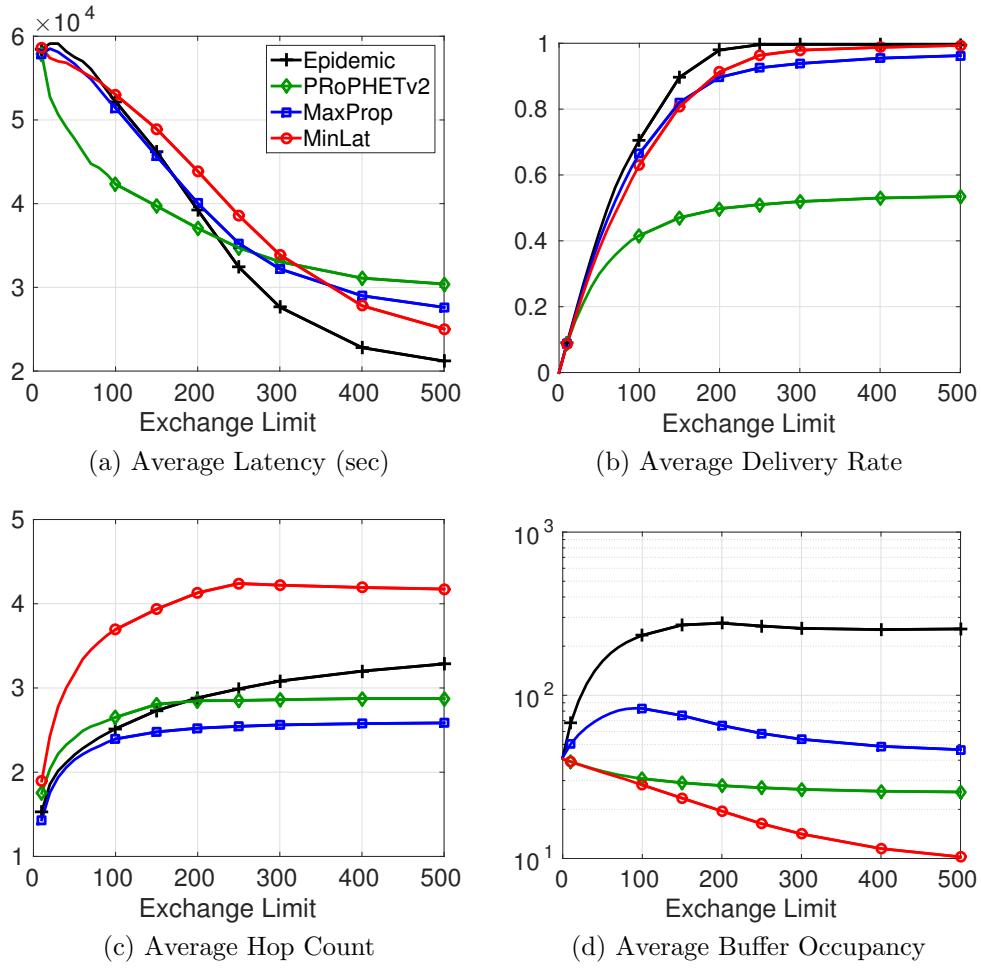
(c) Average Hop Count

(d) Average Buffer Occupancy

Figure 3.5 Effect of buffer size on performance metrics

Finally, we assume that the contact duration is limited so that the number of messages during any meeting is restricted by an exchange limit. We set the buffer size to 1000 messages so that all the algorithms reach their best possible delivery rate. This buffer size implies 50 MB of node memory if each message is 50 KB. We check the effect of varying the

exchange limit on the network performance. Figure 3.6 shows the four comparison metrics for exchange limits in the interval 0 to 500 messages. As we see in the figure, MinLat cannot achieve the optimum average latency for some values of exchange limit, but it still has the best performance in terms of buffer occupancy.



(a) Average Latency (sec)

(b) Average Delivery Rate

(c) Average Hop Count

(d) Average Buffer Occupancy

Figure 3.6 Effect of exchange limit on performance metrics

The simulation results displayed in Figures 3.4-3.6 illustrate that although MinLat has been designed for an ideal network model, it has an acceptable performance when we impose realistic conditions such as finite TTL, buffer size, and exchange limit.

3.7 Summary

In this chapter, we used an analytical framework to model the opportunistic data transfer among mobile devices in DTNs. In our model, the random inter-meeting times of nodes are assumed to be independent and exponentially distributed. We formulated the routing/forwarding problem as an optimization problem in which the goal is to minimize the sum of expected latencies from all nodes of the network to a particular destination. We proved that the solution of this problem is binary, i.e., when an arbitrary node meets any other node in the network, its optimum forwarding rule dictates either to always forward its messages to the encountered node or to never forward any messages to it. We also showed that the solution of this optimization problem minimizes the expected latency from each node of the network to the destination as well. Based on these results, we proposed centralized and distributed versions of an algorithm to find the optimum forwarding decision rules and proved that each of these algorithms result in the same solution. In order to evaluate the performance efficiency of the suggested algorithms in different synthetic and real-world networks, we chose four performance metrics as comparison metrics and compared our proposed decentralized algorithm (MinLat) with the most similar existing approaches. In order to evaluate the performance of MinLat in more realistic scenarios, we conducted simulations in larger networks with practical constraints like limited message life (TTL), buffer size and message exchange.

One of the main contributions of this work is relaxing the condition of having complete knowledge of meeting rates at each node for making the forwarding decisions. We used a recursive maximum likelihood procedure (MinLat-E) to learn the meeting rates online and proved its convergence in probability to the same optimal solution. The validity of this theoretical result was assessed through simulations. Moreover, we compared the convergence speed of the proposed centralized and decentralized algorithms when the meeting rates are estimated online. The simulation results show that the decentralized algorithm has almost the same convergence rate as the centralized algorithm, even though the network topology is not known at individual nodes.

3.8 Proofs

3.8.1 Proof of Lemma 1

Lemma 1: *The expected latency of a node $i \in \mathcal{N}$ to the destination d is*

$$L_{id}(\mathbf{P}) = \frac{1 + \sum_{j \in \mathcal{S}_i} p_{ij} \lambda_{ij} L_{jd}(\mathbf{P})}{\sum_{j \in \mathcal{S}_i} p_{ij} \lambda_{ij}}$$

When node i commences in its routing of a message, it must first wait a time T_w before it meets one of its neighbours. The amount of time before node i meets a specific neighbour j is an exponentially distributed random variable with parameter λ_{ij}. The time T_w is equal to the minimum of the exponentially distributed random variables corresponding to all neighbours $k \in \mathcal{S}_i$ and its expected value is

$$\mathbb{E}(T_w) = \frac{1}{\sum\limits_{k \in \mathcal{S}_i} \lambda_{ik}} \tag{3.13}$$

The probability that j is the first node that i meets is $\frac{\lambda_{ij}}{\sum_{k \in \mathcal{S}_i} \lambda_{ik}}$. Hence $L_{id}(\mathbf{P})$ is

$$L_{id}(\mathbf{P}) = \mathbb{E}(T_w) + \sum_{j \in \mathcal{S}_i} \frac{\lambda_{ij}}{\sum\limits_{k \in \mathcal{S}_i} \lambda_{ik}} [p_{ij} L_{jd}(\mathbf{P}) + (1 - p_{ij}) L_{id}(\mathbf{P})] \tag{3.14}$$

The last term in (3.14) follows from the memoryless property of the distributions. Substituting (3.13) into (3.14) leads to (3.2). □

3.8.2 Proof of Theorem 1

Theorem 1: *Suppose* $\mathbf{P}^* = \arg\min_{\mathbf{P} \in [0,1]^{N \times N}} \sum_{i=1}^{N} L_{id}(\mathbf{P})$. *Then:*

1. $\mathbf{P}^*$ *is a binary matrix (its components are either* 0 *or* 1*).*
2. *For any $i \in \mathcal{N}$, the matrix $\mathbf{P}^*$ also minimizes $L_{id}(\mathbf{P})$:*

$$\forall i \in \mathcal{N}: \quad \mathbf{P}^* = \arg \min_{\mathbf{P} \in [0,1]^{N \times N}} L_{id}(\mathbf{P})$$

Let us assume that the nodes, excluding d, are labelled in ascending order of their expected latency under $\mathbf{P}^*$, i.e.,

$$L_{1d}(\mathbf{P}^*) < L_{2d}(\mathbf{P}^*) < \cdots < L_{(N-1)d}(\mathbf{P}^*)$$

For a given matrix $\mathbf{P}$, we denote by $\mathbf{P}_{\bar{i}}$ a matrix that consists of all rows of $\mathbf{P}$ except the i^{th} one. If we fix $\mathbf{p}_i$ and L_{kd} for $k \in \mathcal{S}_i, k \neq j$ for some $j \in \mathcal{S}_i$, then L_{id} is monotonically increasing with respect to L_{jd} (see (3.2)). This implies that if we start with any $\mathbf{P}$ and change only $\mathbf{p}_j$ to decrease L_{jd}, then all other L_{id} values such that $j \in \mathcal{S}_i$ either decrease or remain the same. The matrix $\mathbf{P}^*$ must therefore satisfy $\mathbf{p}_j^* = \arg\min L_{jd}(\mathbf{P}_{\bar{j}}^*, \mathbf{p}_j)$ for all j. Otherwise we could choose an alternative $\mathbf{p}_j'$ that reduces L_{jd} and hence achieves $\mathcal{U}(\mathbf{P}') < \mathcal{U}(\mathbf{P}^*)$.

We can examine the partial derivative of L_{id} with respect to p_{ij} at $\mathbf{P}' = (\mathbf{P}_{\bar{i}}^*, \mathbf{p}_i)$:

$$\frac{\partial L_{id}}{\partial p_{ij}} = \frac{\lambda_{ij}[\sum_{k\in\mathcal{S}_i} \lambda_{ik}p_{ik}(L_{jd}(\mathbf{P}') - L_{kd}(\mathbf{P}')) - 1]}{(\sum_{k\in\mathcal{S}_i} \lambda_{ik}p_{ik})^2} \tag{3.15}$$

This derivative has the same sign as: $L_{jd}(\mathbf{P}') - \frac{1+\sum_{k\in\mathcal{S}_i} \lambda_{ik}p_{ik}L_{kd}(\mathbf{P}')}{\sum_{k\in\mathcal{S}_i} \lambda_{ik}p_{ik}}$, or equivalently $L_{jd}(\mathbf{P}') - L_{id}(\mathbf{P}')$. This expression for the derivative, together with the requirement that $\mathbf{p}_i^* = \arg\min L_{id}(\mathbf{P}_{\bar{i}}^*, \mathbf{p}_i)$, implies that $\mathbf{p}_{ij}^* = 0$ if $L_{id}(\mathbf{P}^*) < L_{jd}(\mathbf{P}^*)$ and $\mathbf{p}_{ij}^* = 1$ if $L_{id}(\mathbf{P}^*) > L_{jd}(\mathbf{P}^*)$. Our assumption that the solution is unique implies that $L_{id}(\mathbf{P}^*) \neq L_{jd}(\mathbf{P}^*)$. Otherwise, from (3.14), it is clear that we could choose any p_{ij}^* between 0 and 1 and achieve the same $L_{id}(\mathbf{P}^*)$, without affecting any other $L_{jd}(\mathbf{P}^*)$. This establishes statement (1) of the theorem, namely that $\mathbf{P}^*$ is a binary matrix with entries equal to 0 or 1.

Although we have established that $L_{id}(\mathbf{P}_{\bar{i}}^*, \mathbf{p}_i^*) = \min L_{id}(\mathbf{P}_{\bar{i}}^*, \mathbf{p}_i)$, we have not yet shown that $\mathbf{P}^*$ globally minimizes L_{id}. We establish this by contradiction. Suppose $\mathbf{P}^*$ does not minimize the expected latency for some non-empty set of nodes $\mathcal{N}' \subset \mathcal{N}$. In other words, denoting the minimum expected latency achieved via the minimization in (3.3) for node i by L_{id}^*, we have

$$\forall i \in \mathcal{N}' : \quad L_{id}^* < L_{id}(\mathbf{P}^*) \tag{3.16}$$

Let node s be the node in $\mathcal{N}'$ such that $L_{sd}^* < L_{kd}^*$ for all $k \in \mathcal{N}', k \neq s$. Denote by ℓ the ranking of the node with greatest expected latency under $\mathbf{P}^*$ such that $L_{\ell d}(\mathbf{P}^*) < L_{sd}^*$. Based on the discussion above, for each node $i \in \{1, 2, \ldots, \ell\}$, $p_{ik}^* = 0$ for all $k > \ell$ and

hence $p^*_{is} = 0$. Node s must have at least one neighbour in the set $\{d, 1, 2, \ldots \ell\}$. Otherwise, it could not achieve an expected latency under $\mathbf{P}^*$ that is less than all nodes $\ell+1, \ldots, N-1$ (observe from (3.2) that $L_{sd}(\mathbf{P}^*) > \min_{p_{sj}>0} L_{jd}(\mathbf{P}^*)$).

The matrix $\mathbf{P}'$ that achieves the minimum L^*_{sd} must satisfy $p'_{sk} = 0$ for all $k \in \mathcal{N}'$, since for any matrix $\mathbf{P}'$ we have $L_{kd}(\mathbf{P}') \geq L^*_{kd} > L^*_{sd}$. We also have $p'_{sk} = 1$ for $k \in \mathcal{S}_s \cap \{d, 1, 2, \ldots, \ell\}$ if $L_{kd}(\mathbf{P}') < L_{sd}(\mathbf{P}')$. For a fixed choice of $\mathbf{p}'_s$ the value $L_{sd}(\mathbf{P}')$ decreases if we can reduce $L_{kd}(\mathbf{P}')$ for any k such that $p'_{sk} = 1$. The matrix $\mathbf{P}^*$ minimizes L_{kd} for all $k \in \{1, 2, \ldots, \ell\}$, implying that $\mathbf{p}'_k = \mathbf{p}^*_k$ for all $k \in \{1, 2, \ldots, \ell\}$. Since $L_{kd}(\mathbf{P}') = L_{kd}(\mathbf{P}^*)$ for all $k \in \mathcal{S}_s \cap \{d, 1, 2, \ldots, \ell\}$, it follows that $\mathbf{p}'_s = \mathbf{p}^*_s$. For node s, the values of $\mathbf{p}'_j$ for $j \notin \{1, 2, \ldots, \ell, s\}$ have no impact on L_{sd}, so we have $L_{sd}(\mathbf{P}^*) = L_{sd}(\mathbf{P}') = L^*_{sd}$. This contradicts the original assumption that $\mathbf{P}^*$ does not minimize the latency for all nodes $s \in \mathcal{N}'$, and thus establishes statement (2) of the theorem. □

3.8.3 Proof of Theorem 2

Theorem 2: *Suppose all meeting rates are different and there exists a unique solution* $\mathbf{B}^*$ *for the optimization problem* (3.4).

1. *After each iteration of Algorithm 1,*

   (a) $\forall i \in \mathcal{A}, \forall j \in \mathcal{N}: \quad b_{ij} = b^*_{ij}$

   (b) $\forall i \in \mathcal{A}: \quad L_{id}(\mathbf{B}) = L_{id}(\mathbf{B}^*)$

   (c) $\max_{i \in \mathcal{A}} L_{id}(\mathbf{B}^*) < \min_{i \notin \mathcal{A}} L_{id}(\mathbf{B}^*)$

2. *Upon completion, Algorithm 1 identifies a labelling* $\mathbf{B}$ *and associated expected latencies* L_{id} *such that* $\mathbf{B} = \mathbf{B}^*$

We observe that for all $i \in \mathcal{N}$, $G_{id} \geq L^*_{id}$ (since the optimizations are the same). Based on Theorem 1 and its proof, the equality holds only if $j \in \mathcal{A}$ for all $j \in \mathcal{S}_i$ such that $L^*_{jd} < L^*_{id}$. The statements in the theorem follow based on an induction argument.

Suppose, without loss of generality, that the nodes are labelled in ascending order of expected latency under $\mathbf{B}^*$. For node 1, the only neighbour with lower expected latency is the destination. In iteration 1, the destination is included in $\mathcal{A}$ and must be in $\mathcal{S}_1$. Recall that $L^*_{1d} > \min_{b^*_{1j}=1} L^*_{jd}$. Node 1 has the minimum expected latency according to the chosen labelling and Theorem 1, except for the destination itself. The relationship thus implies

that $d \in \mathcal{S}_1$. We therefore have $G_{1d} = L^*_{1d} < L^*_{jd} \leq G_{jd}$, and node 1 is selected to be added to $\mathcal{A}$, with $b_{1d} = 1$ and $b_{jd} = 0$ for all $j \neq d$. Statements 1(a) to 1(c) in the theorem clearly hold after one iteration, i.e., after the addition of node 1 to $\mathcal{A}$.

Assume the same statements hold after the addition of node $k-1$ to $\mathcal{A}$. Then, for node k we must have $j \in \mathcal{A}$ for all $j \in \mathcal{S}_k$ such that $L^*_{jd} < L^*_{kd}$. Again this implies that $G_{kd} = L^*_{kd} < L^*_{jd} \leq G_{jd}$ for all $j > k$. Thus, node k is correctly selected for addition to $\mathcal{A}$ and the statements 1(a) to 1(c) hold at the end of iteration k.

It follows that the statements hold for all iterations of the algorithm, and after completion, when $\mathcal{A} = \mathcal{N}$, the second statement follows. □

3.8.4 Proof of Theorem 3 and Proposition 1

Proof of Theorem 3

Theorem 3: *The decision matrix* $\mathbf{B}$ *identified by Algorithm 2 converges to* $\mathbf{B}^*$ *with probability* 1.

As mentioned in Section 3.4, we assume that the nodes are labelled in order of ascending expected latency under $\mathbf{B}^*$. Based on the definition of T_k, $k = 1, \ldots, N$, and due to the assumption that the inter-meeting times are exponentially distributed, T_N is finite with probability 1.

At T_1, node 1 learns its meeting rate with the destination (λ_{1d}). Since the estimated latencies are initialized to ∞ and due to the update equations in Algorithm 2, the estimation that node 1 has at T_1 of the expected latencies of its neighbours $i \in \mathcal{S}_1$ are upper-bounds, i.e. $\hat{L}_{id}(1) \geq L_{id}(\mathbf{B}^*)$. As discussed in the proof of the previous theorems, the minimizer $\mathbf{b}^*_1$ has $b_{1d} = 1$ and $b_{1j} = 0$ for all $j \neq d$. At time T_1, since the term involving d in the update equation of Algorithm 2 has its minimum value, the vector $\mathbf{m}^*_1 = \mathbf{b}^*_1$ identifies the same minimum latency $\hat{L}_{1d}(1) = L_{1d}(\mathbf{B}^*)$. Hence, immediately after time T_1 we are guaranteed that $\mathbf{b}_1 = \mathbf{b}^*_1$.

At T_k, node k is aware of the minimum expected latencies $\hat{L}_{sd}(k) = L_{sd}(\mathbf{B}^*)$ for the nodes in the set $\mathcal{V}_k = \{d, 1, ..., k\} \cap \mathcal{S}_k$. All other expected latencies are upper bounds, i.e. $\hat{L}_{jd}(k) \geq L_{jd}(\mathbf{B}^*)$ for $j \notin \mathcal{V}_k$. The solution $\mathbf{b}^*_k$ takes value 1 only for nodes in $\mathcal{V}_k$. The minimizer $\mathbf{m}^*_k$ at time T_k is thus equal to $\mathbf{b}^*_k$ and achieves $\hat{L}_{kd}(k) = L_{kd}(\mathbf{B}^*)$. Therefore, immediately after T_k we will have $\mathbf{b}_k = \mathbf{b}^*_k$. This argument applies until just after T_N, at

which point we have $\mathbf{B} = \mathbf{B}^*$. Since T_N is finite with probability 1, the statement of the theorem follows.

Proof of Proposition 1

Proposition 1: *Assuming that the nodes are labelled in order of ascending expected latency under* $\mathbf{B}^*$*, the expected convergence time,* $\mathbb{E}(T_N)$*, of Algorithm 2 is bounded as*

$$\mathbb{E}(T_N) < \frac{1}{\lambda_{1d}} + \sum_{l=2}^{N} \frac{l-1}{\min\limits_{\substack{i\in\{1,\ldots,l-1\}\\ \lambda_{li}>0}} \lambda_{li}}$$

Algorithm 2 has converged when all of the nodes have met all of their relay candidates and have identified their optimum forwarding rules. Thus, $\mathbb{E}(T_N)$, where T_N defined above in the proof of Theorem 3, is the expected convergence time. Considering the worst case where an arbitrary node $k > 1$ is connected to all the nodes in the set $\{1, ..., k-1\}$, we have

$$\mathbb{E}(T_k - T_{k-1}) = \mathbb{E}(\max_{i\in\{1,\ldots,k-1\}} x_i) \tag{3.17}$$

where x_i denotes the inter-meeting time of node k with its neighbour i and follows an exponential distribution with parameter λ_{ki}. Using a standard result for the expected value of the maximum of non-identical independent exponential random variables, we have

$$\mathbb{E}(T_k - T_{k-1}) = \sum_{i=1}^{k-1} \frac{1}{\lambda_{ki}} - \sum_{i=1}^{k-1}\sum_{j=i+1}^{k-1} \frac{1}{\lambda_{ki}+\lambda_{kj}} + \sum_{i=1}^{k-1}\sum_{j=i+1}^{k-1}\sum_{l=j+1}^{k-1} \frac{1}{\lambda_{ki}+\lambda_{kj}+\lambda_{kl}} - \ldots \tag{3.18}$$

An upper bound on this value is $\mathbb{E}(T_k - T_{k-1}) < \sum_{i=1}^{k-1} \frac{1}{\lambda_{ki}} < \frac{k-1}{\min\limits_i \lambda_{ki}}$ Therefore, an upper-bound on the convergence time of Algorithm 2 is $\mathbb{E}(T_N) < \frac{1}{\lambda_{1d}} + \sum_{l=2}^{N} \frac{l-1}{\min\limits_{\substack{i\in\{1,\ldots,l-1\}\\ \lambda_{li}>0}} \lambda_{li}}$. □

3.8.5 Proof of Lemma 2

Lemma 2: *The minimization problem in Algorithm 2,*

$$\hat{L}_{id}(i) = \min_{\mathbf{m}_i\in\{0,1\}^{|\mathcal{S}_i|}} \frac{1 + \sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}\hat{L}_{kd}(i)}{\sum_{k\in\mathcal{S}_i} m_{ik}\lambda_{ik}} \ ,$$

can be converted to a linear programming problem.

Since we want to use the result of this lemma in Chapter 4 as well, we prove it for a more general case where m_{ij} is not necessarily binary for any $i, j \in \mathcal{N}$. Let us return to considering the probability decision variable vector $\mathbf{p}$ instead of binary decision variable vector (We still expect the optimal solution to be of the binary form). Without loss of generality, we relabel the nodes $k \in \mathcal{S}_i$ by labels $1, ..., |\mathcal{S}_i|$. The optimization problem that node i tries to solve in Algorithm 2 follows this general form which is known as linear fractional optimization problems:

$$\begin{aligned} \hat{L}_{id}(i) = \min_{\mathbf{p}^i} \quad & \frac{\mathbf{c}^T\mathbf{p}^i + \alpha}{\mathbf{d}^T\mathbf{p}^i + \beta} \\ \text{subject to} \quad & \mathbf{A}\mathbf{p}^i \leq \mathbf{b} \end{aligned} \tag{3.19}$$

where $\mathbf{p}^i_{|\mathcal{S}_i|\times 1} = [p_{i1}, ..., p_{i|\mathcal{S}_i|}]^T$, $\alpha = 1$, $\mathbf{c}_{|\mathcal{S}_i|\times 1} = [\lambda_{i1}\hat{L}_{1d}(i), ..., \lambda_{i|\mathcal{S}_i|}\hat{L}_{|\mathcal{S}_i|d}(i)]^T$, $\beta = 0$, $\mathbf{d}_{|\mathcal{S}_i|\times 1} = [\lambda_{i1}, ..., \lambda_{i|\mathcal{S}_i|}]^T$, $\mathbf{b}_{2|\mathcal{S}_i|\times 1} = [1, ..., 1, 0, ..., 0]^T$, and the elements of the matrix $\mathbf{A}_{2|\mathcal{S}_i|\times|\mathcal{S}_i|}$ are

$$\mathbf{A}_{ij} = \begin{cases} 1 & \text{if } i < |\mathcal{S}_i| \quad \& \quad j = i \\ -1 & \text{if } i > |\mathcal{S}_i| \quad \& \quad j = i - |\mathcal{S}_i| \\ 0 & \text{otherwise} \end{cases}$$

After applying the following parameter changes (the Charnes-Cooper transformation [108]), we have

$$\mathbf{x} = \frac{1}{\mathbf{d}^T\mathbf{p}^i}\mathbf{p}^i \qquad y = \frac{1}{\mathbf{d}^T\mathbf{p}^i} \tag{3.20}$$

and the optimization problem 3.19 converts to

$$\begin{aligned} \min_{\mathbf{x},y} \quad & \mathbf{c}^T\mathbf{x} + \alpha y \\ \text{subject to} \quad & \mathbf{d}^T\mathbf{x} = 1 \\ & \mathbf{A}\mathbf{x} \leq \mathbf{b}y \\ & y \geq 0 \end{aligned} \tag{3.21}$$

which is a linear optimization problem and can be solved using Linear Programming (LP) solution methods [109]. □

3.8.6 Proof of Theorem 4

Theorem 4: *For any node in the network, the sequences of estimated and achieved latencies converge to its optimum expected latency in probability, i.e.,* $\{\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i)\} \xrightarrow{p} L_{id}(\mathbf{B}^*)$ *and* $\{L_{id}(\widetilde{\mathbf{B}}_t)\} \xrightarrow{p} L_{id}(\mathbf{B}^*)$. *More precisely for any* $\epsilon > 0$,

$$\lim_{t->\infty} P(|\widetilde{L}_{id,t}(\widetilde{\mathbf{B}}_t, i) - L_{id}(\mathbf{B}^*)| < \epsilon) = 1$$

$$\lim_{t->\infty} P(|L_{id}(\widetilde{\mathbf{B}}_t) - L_{id}(\mathbf{B}^*)| < \epsilon) = 1$$

We first prove the statement of the theorem for estimated latencies. For ease of notation, we drop the superscript n_{ij} from the estimate of meeting rates between nodes i and j and denote it by $\hat{\lambda}_{ij}$ in this proof. Without loss of generality, we relabel the nodes such that $L_{1d}(\mathbf{B}^*) < L_{2d}(\mathbf{B}^*) < ... < L_{Nd}(\mathbf{B}^*)$. At node i, MinLat-E forms estimates of the meeting rates with the contact graph neighbours, $\hat{\lambda}_{ik}$ for $k \in \mathcal{S}_i$, and the expected latencies from each neighbour $\widetilde{L}_{kd}(\widetilde{\mathbf{B}}_t, i)$. The result is a sequence of random variables $\{\widetilde{L}_{id}(\widetilde{\mathbf{B}}_t, i)\}$, with a variable being added to the sequence each time node i meets another node. [110] shows that the maximum likelihood estimator of the meeting rates is consistent, i.e., $\forall i, j \in \mathcal{N}, \hat{\lambda}_{ij} \xrightarrow{p} \lambda_{ij}$. More precisely,

$$\forall \epsilon_{ij} > 0 : \lim_{t->\infty} P(|\hat{\lambda}_{ij} - \lambda_{ij}| < \epsilon_{ij}) = 1 \tag{3.22}$$

Equivalently, writing $\epsilon_{ij} = \epsilon_0 \lambda_{ij}$, we have for any $\delta > 0$ and $\epsilon_0 > 0$, there exists a $t_0 > 0$ such that for all $t > t_0$:

$$P\Big((1 - \epsilon_0)\lambda_{ij} < \hat{\lambda}_{ij,t} < (1 + \epsilon_0)\lambda_{ij}\Big) > (1 - \delta) \tag{3.23}$$

where $\hat{\lambda}_{ij,t}$ denotes the estimate of the meeting rate between nodes i and j at time t.

We show that for any set of meeting rates $\{\lambda_{ij}\}_{i,j \in \mathcal{N}}$, there exists an $\epsilon = \epsilon_0$ for which the optimum forwarding decision matrix in MinLat-E ($\widetilde{\mathbf{B}}_t$) is the same as the optimum forwarding decision matrix in MinLat ($\mathbf{B}^*$) with desirably high probability. In order to do so, we find upper and lower bounds (that apply with high probability) on the estimated expected latencies for the optimal decision matrices identified by both MinLat and MinLat-E. We first demonstrate a relationship between the estimated and true expected latencies

that would hold if the nodes employed the optimal decision matrix $\mathbf{B}^*$. We show that there exists a $t_{\delta,i}$ such that for each node $i \in \mathcal{N}$, with probability greater than $1-\delta$, for any positive δ, we have for all $t > t_{\delta,i}$:

$$\frac{(1-\epsilon_0)^{i-1}}{(1+\epsilon_0)^i} L_{id}(\mathbf{B}^*) < \widetilde{L}_{id,t}(\mathbf{B}^*, i) < \frac{(1+\epsilon_0)^{i-1}}{(1-\epsilon_0)^i} L_{id}(\mathbf{B}^*) \tag{3.24}$$

We derive (3.24) by induction. From the arguments made in Theorem 3, we know that under $\mathbf{B}^*$ an arbitrary node i will not forward any messages to a node that does not belong to the set $\{1, .., i-1, d\}$. For $i = 1$, after time $t_{\delta,1}$ such that (3.23) holds, we have, with probability greater than $1-\delta$ for $t > t_{\delta,1}$:

$$\frac{L_{1d}(\mathbf{B}^*)}{1+\epsilon_0} = \frac{1}{1+\epsilon_0}\frac{1}{\lambda_{1d}} < \widetilde{L}_{1d,t}(\mathbf{B}^*, 1) = \frac{1}{\hat{\lambda}_{1d,t}} < \frac{1}{1-\epsilon_0}\frac{1}{\lambda_{1d}} = \frac{L_{1d}(\mathbf{B}^*)}{1-\epsilon_0} \tag{3.25}$$

Suppose (3.24) holds for all the nodes $1, \ldots, k-1$. Denote by $t_{jk} > t$ the last meeting between node j and k that occurs subsequent to $t_{\delta,k-1}$ but prior to a considered time t. Then we can identify a $t^u_{\delta,k}$ so that the following relationship holds with probability greater than $1-\delta$ for $t > t^u_{\delta,k}$:

$$\widetilde{L}_{kd,t}(\mathbf{B}^*, k) = \frac{1}{\Sigma_{j\in\mathcal{S}_k} b^*_{kj}\hat{\lambda}_{kj}}(1 + \Sigma_{j\in\mathcal{S}_k} b^*_{kj}\hat{\lambda}_{kj}\widetilde{L}_{jd,t_{jk}}(\mathbf{B}^*, j)) \tag{3.26a}$$

$$< (\frac{1}{1-\epsilon_0}\frac{1}{\Sigma_{j\in\mathcal{S}_k} b^*_{kj}\lambda_{kj}})(1 + \Sigma_{j\in\mathcal{S}_k} b^*_{kj}(1+\epsilon_0)\frac{(1+\epsilon_0)^{k-2}}{(1-\epsilon_0)^{k-1}} L_{jd}(\mathbf{B}^*)) \tag{3.26b}$$

$$< \frac{(1+\epsilon_0)^{k-1}}{(1-\epsilon_0)^k} L_{kd}(\mathbf{B}^*) \tag{3.26c}$$

Here we have chosen $t^u_{\delta,k}$ to be sufficiently large such that the inequality on the second line holds with probability exceeding $1-\delta$. Similarly we can identify a $t^\ell_{\delta,k}$ so that the lower bound in (3.24) holds with probability greater than $1-\delta$ for $t > t^\ell_{\delta,k}$. By taking $t_{\delta,k} = \max\{t^\ell_{\delta,k}, t^u_{\delta,k}\}$, we see that (3.24) holds for node k as well, and by induction, holds for all nodes $i \in \mathcal{N}$.

We now turn our attention to the forwarding matrix determined by MinLat-E ($\widetilde{\mathbf{B}}_t$). We first consider a scenario where the estimates of the meeting rates are frozen after a certain time t_f. The minimization procedure in MinLat-E is the same as in MinLat, but operates

on the estimates of the meeting rates. If these estimates are held constant, then the results in Theorems 1-3 apply, with the substitution of $\hat{\lambda}_{ij}$ anywhere we make use of λ_{ij}. With probability 1, the optimization algorithm will thus converge after a finite time t', and the estimated expected latencies will be consistent across the network. Hence, there exists a labeling $\{\hat{1}, \hat{2}, ...\hat{N}\}$ for which $\widetilde{L}_{\hat{1}d}(\widetilde{\mathbf{B}}_{t_f+t'}, \hat{1}) < \widetilde{L}_{\hat{2}d}(\widetilde{\mathbf{B}}_{t_f+t'}, \hat{2}) < \cdots < \widetilde{L}_{\hat{N}d}(\widetilde{\mathbf{B}}_{t_f+t'}, \hat{N})$ We can now employ the same argument that was used above for $\mathbf{B}^*$ to determine that there is a finite time $t_{\delta,\hat{k}}$ such that the following bound holds for all $\hat{k} \in \mathcal{N}$ with probability greater than $1-\delta$ for all $t > t_{\delta,\hat{k}}$.

$$\frac{(1-\epsilon_0)^{\hat{k}-1}}{(1+\epsilon_0)^{\hat{k}}} L_{\hat{k}d}(\widetilde{\mathbf{B}}_t) < \widetilde{L}_{\hat{k}d}(\widetilde{\mathbf{B}}_t, \hat{k}) < \frac{(1+\epsilon_0)^{\hat{k}-1}}{(1-\epsilon_0)^{\hat{k}}} L_{\hat{k}d}(\widetilde{\mathbf{B}}_t) \tag{3.27}$$

In the actual MinLat-E algorithm, the meeting rates $\hat{\lambda}$ are not frozen after t_0, but continue to be updated as more meetings occur. This only results in the probabilistic bounds on $\hat{\lambda}$ being tighter, and hence can only tighten the bounds on $L_{\hat{k}d}(\widetilde{\mathbf{B}}_t, \hat{k})$.

To avoid having node-specific bounds on the accuracy of the estimates, we can rewrite the bounds as:

$$\frac{(1-\epsilon_0)^{N-1}}{(1+\epsilon_0)^{N}} L_{kd}(\mathbf{B}) < \widetilde{L}_{kd}(\mathbf{B}, k) < \frac{(1+\epsilon_0)^{N-1}}{(1-\epsilon_0)^{N}} L_{kd}(\mathbf{B}) \tag{3.28}$$

This bound holds for both $\mathbf{B} = \mathbf{B}^*$ and $\mathbf{B} = \widetilde{\mathbf{B}}_t$ with probability at least $1-\delta_0$ after some time t_0.

Our goal is to show that there exists a moment of time after which $\widetilde{\mathbf{B}}_t = \mathbf{B}^*$ is true with desirably high probability. We can accomplish this by showing that there exists an ϵ_0 (and thus an associated time t_0) for which the upper-bound on $\widetilde{L}_{id}(\mathbf{B}^*, i)$ is less than the lower-bound on $\widetilde{L}_{id}(\widetilde{\mathbf{B}}_t, i)$ for any $\widetilde{\mathbf{B}}_t \neq \mathbf{B}^*$ for all $t > t_0$. If this is the case, then with probability exceeding $1-\delta_0$, the optimization procedure that derives $\widetilde{\mathbf{B}}_t$ will set it to $\mathbf{B}^*$, because it minimizes the estimated latencies. Hence, ϵ_0 should satisfy

$$L_{id}(\mathbf{B}^*) < (\frac{1-\epsilon_0}{1+\epsilon_0})^{2N-1} L_{id}(\widetilde{\mathbf{B}}_t) \ , \tag{3.29}$$

which leads to

$$\epsilon_0 < \frac{e^{\frac{\ln(K)}{2N-1}} - 1}{e^{\frac{\ln(K)}{2N-1}} + 1} , \tag{3.30}$$

where $K = \frac{\min_{\mathbf{B}\neq\mathbf{B}^*} L_{id}(\mathbf{B})}{L_{id}(\mathbf{B}^*)}$.

Now that we have established that after a finite amount of time $\widetilde{\mathbf{B}}_t = \mathbf{B}^*$ occurs with a probability desirably close to one, we can show that $\{\widetilde{L}_{id}(\widetilde{\mathbf{B}}_t, i)\} \xrightarrow{p} L_{id}(\mathbf{B}^*)$ for any $i \in \mathcal{N}$ using the following properties:

1. If $X_n \xrightarrow{p(\text{or } d)} X$, then $g(X_n) \xrightarrow{p(\text{or } d)} g(X)$ (Continuous Mapping Theorem [111]);
2. If $X_n \xrightarrow{p} X$, then $X_n \xrightarrow{d} X$;
3. If $X_n \xrightarrow{d} c \in \mathbb{R}$, then $X_n \xrightarrow{p} X$;
4. If $X_n \xrightarrow{d} X$ and $Y_n \xrightarrow{d} c \in \mathbb{R}$, then $g(X_n, Y_n) \xrightarrow{d} g(X, c)$ (Slutsky's Theorem [112]),

where $\xrightarrow{d}$ denotes the convergence in distribution and $g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is an arbitrary continuous function.

Again consider the node labelling such that $L_{1d}(\mathbf{B}^*) < L_{2d}(\mathbf{B}^*) < \cdots < L_{Nd}(\mathbf{B}^*)$. For node 1, $\{\widetilde{L}_{1d}(\widetilde{\mathbf{B}}_t, 1) = \frac{1}{\hat{\lambda}_{1d}}\} \xrightarrow{p} L_{1d}(\mathbf{B}^*) = \frac{1}{\lambda_{1d}}$ which is obvious from property 1. For any node $k > 1$, if $\{\widetilde{L}_{jd}(\widetilde{\mathbf{B}}_t, j)\} \xrightarrow{p} L_{jd}(\mathbf{B}^*)$ holds for any $j \in \{1, ..., k-1\}$, then due to properties 2-4 we have,

$$\forall j \in \{1, ..., k-1, d\} \cap \mathcal{S}_k : \qquad \{\hat{\lambda}_{kj}\widetilde{L}_{jd}(\widetilde{\mathbf{B}}_t, k)\} \xrightarrow{d} \lambda_{kj} L_{jd}(\mathbf{B}^*) \in \mathbb{R} \ , \tag{3.31}$$

and

$$\{\Sigma_{j\in\{1,...,k-1,d\}} \hat{b}^*_{kj}\hat{\lambda}_{kj}\} \xrightarrow{p} \Sigma_{j\in\{1,...,k-1,d\}} b^*_{kj}\lambda_{kj} \ . \tag{3.32}$$

Property 1 in combination with (3.31) and (3.32) results in the statement of the theorem.

For the achieved expected latencies, $L_{kd}(\widetilde{\mathbf{B}}_t)$, as opposed to those estimated at the nodes, the proof is more straightforward. For a given decision matrix, $\widetilde{\mathbf{B}}_t$, the expected latencies $L_{kd}(\widetilde{\mathbf{B}}_t)$s are functions of the true meeting rates λ and are thus not random variables. Thus, the sequence $\{L_{id}(\widetilde{\mathbf{B}}_t)\}$ is only a function of the random sequences $\{\hat{\lambda}_{ij}\}$, $j \in \mathcal{S}_i$ via the optimization that determines $\widetilde{\mathbf{B}}_t$. Since we have established that $\widetilde{\mathbf{B}}_t$ converges in probability to $\mathbf{B}^*$, it follows that $\{L_{id}(\widetilde{\mathbf{B}}_t)\}$ converges in probability to $L_{id}(\mathbf{B}^*)$ due to property 1.

□

Chapter 4

Proactive Approach: Bayesian Routing Using Social Information

4.1 Overview

In Chapter 3, we studied how we can proactively control a diffusion process in the context of solving the routing/forwarding problem in opportunistic Delay Tolerant Networks (DTNs). We designed a decentralized algorithm called MinLat to find the optimal forwarding rules in opportunistic DTNs. We also developed a modified version, MinLat-E, in which no a priori knowledge of the rate of node meetings is required. Based on the classifications presented in Section 2.2.2, MinLat and MinLat-E are both history-based routing algorithms, because they only use the contact histories to make forwarding decisions. But, what if we have some external information about the strength of the relationships between nodes that can be assumed to influence the frequency of their contacts. How can we exploit this external information in making more efficient forwarding rules?

In this chapter, we study the routing problem in a Bayesian framework and incorporate the information about the strength of nodes' relationships through prior probability distributions. After formulating the problem in Section 4.2, we propose two Bayesian versions of the MinLat algorithm, BMinLat-I and BMinLat-II, in Sections 4.3 and 4.4. In Chapter 4.5, we evaluate the performance efficiency of these two proposed algorithms through simulations on synthetic and real-world datasets. We summarize the chapter in Section 4.6.

4.2 Problem Statement

In this chapter, we use the same notations we used in Chapter 3 (see Table 3.1). The additional notations defined in this chapter are listed in Table 4.1. In an ideal scenario, each node knows its meeting rates with all the other nodes it meets. In Chapter 3, we modelled the meeting rates as deterministic parameters and proved that the optimal forwarding rules in these scenarios are binary. It means that a node i either always forwards a message it has to another node j it meets or always keeps the message. However, when we model the parameters as random and take uncertainty into account, it is possible that probabilistic decisions are optimal. For this reason, we use $\mathbf{p}_i$ (instead of $\mathbf{m}_i$) to denote the forwarding rule vector of node i in this chapter. We proposed MinLat in Section 3.4 to find the optimal binary forwarding rules. In MinLat, an arbitrary node i solves the following optimization problem each time it meets another node.

$$\hat{L}_{id}(i) = \min_{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}} \frac{1 + \sum_{k \in \mathcal{S}_i} p_{ik} \lambda_{ik} \hat{L}_{kd}(i)}{\sum_{k \in \mathcal{S}_i} p_{ik} \lambda_{ik}} \tag{4.1}$$

Here, $\hat{L}_{kd}(i)$ denotes the estimate that node i has of the expected latency of node k to the destination node d and $\mathcal{S}_i$ denotes all the neighbours of node i, i.e., $\mathcal{S}_i = \{j | \lambda_{ij} > 0\}$.

In order to allow us to express the Bayesian formulation of the problem in a more concise manner, we define the function $g_i : \mathbb{R}^{|\mathcal{S}_i|} \times \mathbb{R}^{|\mathcal{S}_i|} \times [0,1]^{|\mathcal{S}_i|} \to \mathbb{R}$ for node i as

$$g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big) \triangleq \frac{1 + \sum_{k \in \mathcal{S}_i} p_{ik} \lambda_{ik} \hat{L}_{kd}(i)}{\sum_{k \in \mathcal{S}_i} p_{ik} \lambda_{ik}} = \frac{1 + \boldsymbol{\Psi}_i^T \mathbf{p}_i}{\boldsymbol{\lambda}_i^T \mathbf{p}_i} \tag{4.2}$$

where

$$\boldsymbol{\lambda}_i = \begin{bmatrix} \lambda_{i1} \\ \lambda_{i2} \\ \vdots \\ \lambda_{i|\mathcal{S}_i|} \end{bmatrix} \quad , \quad \boldsymbol{\Psi}_i = \begin{bmatrix} \lambda_{i1} \hat{L}_{1d}(i) \\ \lambda_{i2} \hat{L}_{2d}(i) \\ \vdots \\ \lambda_{i|\mathcal{S}_i|} \hat{L}_{|\mathcal{S}_i|d}(i) \end{bmatrix} \quad , \text{and} \quad \mathbf{p}_i = \begin{bmatrix} p_{i1} \\ p_{i2} \\ \vdots \\ p_{i|\mathcal{S}_i|} \end{bmatrix}$$

In MinLat, node i solves

$$\mathbf{p}_i^* = \arg\min_{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}} g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big) \tag{4.3}$$

and updates its estimate of its optimal expected latency as $\hat{L}_{id}(i) = g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i^*\big)$. In this

Symbol(s)	Expression(s)	Definition(s)
$\boldsymbol{\lambda}_i$	$[\lambda_{i1}, \ldots, \lambda_{i\|\mathcal{S}_i\|}]^T$	Vector of meeting rates of node i with its neighbours
$\boldsymbol{\Psi}_i$	$[\lambda_{i1}L_{1d}(i), \ldots, \lambda_{i\|\mathcal{S}_i\|}L_{\|\mathcal{S}_i\|d}(i)]^T$	Vector of product of meeting rates of node i with its neighbours and their expected latencies
$\hat{\boldsymbol{\lambda}}_i^{\mathbf{n}}$	$[\hat{\lambda}_{i1}^{n_{i1}}, \ldots, \hat{\lambda}_{i\|\mathcal{S}_i\|}^{n_{i\|\mathcal{S}_i\|}}]^T$	Vector of ML estimates of node i's meeting rates
$\hat{\boldsymbol{\Psi}}_i^{\mathbf{n}}$	$[\hat{\lambda}_{i1}^{n_{i1}}L_{1d}(i), \ldots, \hat{\lambda}_{i\|\mathcal{S}_i\|}^{n_{i\|\mathcal{S}_i\|}}L_{\|\mathcal{S}_i\|d}(i)]^T$	Vector of ML estimates of $\boldsymbol{\Psi}_i$
$\mathbf{p}_i^{*1}$	-	The forwarding rule determined by BMinLat-I
$\mathbf{p}_i^{*2}$	-	The forwarding rule determined by BMinLat-II
$\boldsymbol{\zeta}_i$	$[\zeta_{i1}, \ldots, \zeta_{i\|\mathcal{S}_i\|}]^T$	Vector of type of social interactions between node i and its neighbours
n_{ij}	-	Number of meetings between nodes i and j
$\kappa_{ij}^{\zeta}, \theta_{ij}^{\zeta}$	-	The hyperparameters of the Gamma distribution of λ_{ij}
N_s	-	Number of generated samples
$\boldsymbol{\lambda}_i^s$	$[\lambda_{i1}^s, \ldots, \lambda_{i\|\mathcal{S}_i\|}^s]^T$	The s^{th} sample vector of meeting rates of node i with all its neighbours
$\boldsymbol{\Psi}_i^s$	$[\lambda_{i1}^s L_{1d}^s(i), \ldots, \lambda_{i\|\mathcal{S}_i\|}^s L_{\|\mathcal{S}_i\|d}^s(i)]^T$	The s^{th} sample vector of product of meeting rates of node i with its neighbours and their expected latencies
$\mathbb{S}_{\boldsymbol{\lambda}_i}$	$\{\lambda_{ik}^s \mid \forall s = 1, \ldots, N_s, \forall k \in \mathcal{S}_i\}$	Set of all generated vector samples λ_{ik}^s
$\mathbb{S}_{\boldsymbol{\Psi}_i}$	$\{\lambda_{ik}^s L_{kd}^s(i) \mid \forall s = 1, \ldots, N_s, \forall k \in \mathcal{S}_i\}$	Set of all generated vector samples Λ_{ik}^s
$\mathbb{X}_i$	$\{x_{ij}^k \mid \forall j \in \mathcal{S}_i, k = 1, \ldots, n_{ij}\}$	Set of all the inter-meeting times that node i has observed
μ_i^k	-	Node i's estimate of the mean of node k's expected latency
σ_i^k	-	Node i's estimate of the standard deviation of node k's expected latency

Table 4.1 Summary of notations used in Chapter 4

case, $\boldsymbol{\lambda}_i$ and $\boldsymbol{\Psi}_i$ are known deterministic vectors.

In a more realistic scenario where meeting rates are unknown, we model both $\boldsymbol{\lambda}_i$ and $\boldsymbol{\Psi}_i$ by random vectors. We suggested MinLat-E for such scenarios. In MinLat-E, each node i uses Equation (3.9) to calculate the Maximum Likelihood (ML) estimate of its meeting rate $\hat{\lambda}_{ij}^{n_{ij}}$ with another node $j \in \mathcal{S}_i$ based on the observed inter-meeting times $\{x_{ij}^1, \ldots, x_{ij}^{n_{ij}}\}$. Therefore, node i solves a similar optimization as in (4.1) where λ_{ij} is replaced with $\hat{\lambda}_{ij}^{n_{ij}}$. The ML estimated meeting rates get updated as time passes and nodes meet each other. In MinLat-E, node i replaces $\boldsymbol{\lambda}_i$ with its ML estimate, $\hat{\boldsymbol{\lambda}}_i^{\mathbf{n}}$, and solves

$$\hat{\mathbf{p}}_i^* = \underset{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}}{\arg\min} \quad g_i\big(\hat{\boldsymbol{\lambda}}_i^{\mathbf{n}}, \hat{\boldsymbol{\Psi}}_i^{\mathbf{n}}, \mathbf{p}_i\big) \tag{4.4}$$

where

$$\hat{\boldsymbol{\lambda}}_i^{\mathbf{n}} = \begin{bmatrix} \hat{\lambda}_{i1}^{n_{i1}} \\ \hat{\lambda}_{i2}^{n_{i2}} \\ \vdots \\ \hat{\lambda}_{i|\mathcal{S}_i|}^{n_{i|\mathcal{S}_i|}} \end{bmatrix} \quad , \quad \hat{\boldsymbol{\Psi}}_i^{\mathbf{n}} = \begin{bmatrix} \hat{\lambda}_{i1}^{n_{i1}} \hat{L}_{1d}(i) \\ \hat{\lambda}_{i2}^{n_{i2}} \hat{L}_{2d}(i) \\ \vdots \\ \hat{\lambda}_{i|\mathcal{S}_i|}^{n_{i|\mathcal{S}_i|}} \hat{L}_{|\mathcal{S}_i|d}(i) \end{bmatrix}$$

The node then updates its estimate of its optimal expected latency as $\hat{L}_{id}(i) = g_i\big(\hat{\boldsymbol{\lambda}}_i^{\mathbf{n}}, \hat{\boldsymbol{\Psi}}_i^{\mathbf{n}}, \hat{\mathbf{p}}_i^*\big)$. We proved that the expected latencies found by MinLat-E converge in probability to the same values achieved by MinLat. We also verified this result through experimental simulations.

In this chapter, we still focus on scenarios where meeting rates are unknown. Hence, we model them as random variables. Moreover, the estimate that each node has of its own or another node's expected latency should be modelled as a random variable. In order to avoid confusion and emphasize the randomness of the estimate of the expected latencies, we use $L_{id}(j)$ (instead of $\hat{L}_{id}(j)$) to denote the random estimate at node j of the expected latency of node i to the destination d in the rest of this chapter. Assuming $\boldsymbol{\lambda}_i$ and $\boldsymbol{\Psi}_i$ are random vectors, we propose two different methods to solve the optimization problem in (4.3). Since the optimization is over $\mathbf{p}_i$, the result is still a function of $\boldsymbol{\lambda}_i$ and $\boldsymbol{\Psi}_i$. Therefore, we can define the function $h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i)$ as

$$h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i) \triangleq \underset{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}}{\arg\min}\, g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big) \tag{4.5}$$

The first idea that we explore in this chapter is to identify a decision rule that minimizes the expectation of the utility function $g_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i)$ over the random vectors $\boldsymbol{\lambda}_i$ and $\boldsymbol{\Psi}_i$.

$$\mathbf{p}_i^{*1} = \underset{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}}{\arg\min}\, \mathbb{E}_{\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i}\big[g_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i)\big] \tag{4.6}$$

where $\mathbf{p}_i^{*1}$ denotes the optimal forwarding rule matrix. This expectation is with respect to the posterior distributions given the observed inter-meeting times (or the prior distributions when there is no observation). We develop a decentralized Bayesian algorithm, BMinLat-I, in Section 4.3 based on this idea. However, as we are going to see in Section 4.3, solving the optimization problem in (4.6) is computationally expensive. Therefore, we propose a heuristic algorithm, BMinLat-II, that finds an approximate solution that often coincides with BMinLat-I. It is also more computationally efficient. The main idea of this second method is to consider the expectation of $h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i)$ as the forwarding rule. Denoting this forwarding rule by $\mathbf{p}_i^{*2}$, we have,

$$\mathbf{p}_i^{*2} = \mathbb{E}_{\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i}\big[h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i)\big] \tag{4.7}$$

Again, the expectation is with respect to the posterior distributions (and the prior when there is no observation). Details of BMinLat-II and its computational complexity are provided in Section 4.4.

Before going into the details of the algorithm developments, we specify the distribution models we are going to use in the rest of this section.

4.2.1 Priors

We wish to improve the performance of the routing algorithms by using some information related to the social interactions of the nodes in a Bayesian framework. Denoting the type of the social interactions between nodes i and j by the random variable ζ_{ij}, we have

$$f(\lambda_{ij}|x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij}) = \frac{f(x_{ij}^1, \ldots, x_{ij}^{n_{ij}}|\lambda_{ij}, \zeta_{ij}) f(\lambda_{ij}|\zeta_{ij}) f(\zeta_{ij})}{f(x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij})} \tag{4.8}$$

This relationship follows from Bayes' rule and the function $f(.)$ indicates a probability density function. We model the prior distribution $f(\lambda_{ij}|\zeta_{ij})$ by a Gamma distribution to

be able to capture both highly informative and weakly informative knowledge about link strengths. Hence, we have

$$f(\lambda_{ij}|\zeta_{ij}) = \mathbf{\Gamma}(\kappa_{ij}^{\zeta}, \theta_{ij}^{\zeta}) = \frac{{\theta_{ij}^{\zeta}}^{\kappa_{ij}^{\zeta}}}{\Gamma(\kappa_{ij}^{\zeta})} \lambda_{ij}^{\kappa_{ij}^{\zeta}-1} e^{-\lambda_{ij}\theta_{ij}^{\zeta}} \tag{4.9}$$

Due to the prior conjugacy of the exponential and Gamma distributions, the posterior distribution $f(\lambda_{ij}|x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij})$ has also a Gamma distribution. Therefore,

$$f(\lambda_{ij}|x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij}) \propto \lambda_{ij}^{n_{ij}+\kappa_{ij}^{\zeta}-1} e^{-\lambda_{ij}\left(\theta_{ij}^{\zeta}+\sum_{k=1}^{n_{ij}} x_{ij}^k\right)} \tag{4.10}$$

In Chapter 3, we assumed that meeting rates are independent. Therefore, the probability distribution function of $\boldsymbol{\lambda}_i$ is

$$f(\boldsymbol{\lambda}_i|\mathbb{X}_i, \boldsymbol{\zeta}_i) = \prod_{j \in \mathcal{S}_i} f(\lambda_{ij}|x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij}) \tag{4.11}$$

where $\mathbb{X}_i = \{x_{ij}^k | \forall j \in \mathcal{S}_i, k = 1, \ldots, n_{ij}\}$ is the set of all the inter-meeting times that node i has observed and $\boldsymbol{\zeta}_i = [\zeta_{i1}, \ldots, \zeta_{i|\mathcal{S}_i|}]^T$.

The time it takes for a message to go from an arbitrary node i to the destination d on a specific route, consists of the times between meetings of each pair of nodes on the route. The core assumption of exponentially distributed inter-meeting times leads to a distribution for the expected latency $L_{id}(i)$ that is non-Gaussian. We, however, approximate the expected latency estimates, $L_{id}(i)$ for all $i \in \mathcal{N}$, as being normally distributed. In other words, we model the posterior distribution $f(L_{id}(i)|x_{ij}^1, \ldots, x_{ij}^{n_{ij}}; \forall j \in \mathcal{S}_i)$ as

$$f(L_{id}(i)|x_{ij}^1 \ldots, x_{ij}^{n_{ij}}; j \in \mathcal{S}_i) = \frac{1}{\sqrt{2\pi}\sigma_i^i} e^{-\frac{(L_{id}(i)-\mu_i^i)^2}{2{\sigma_i^i}^2}} \tag{4.12}$$

where μ_i^k (or σ_i^k) denotes the estimate that node i has of the mean (or the standard deviation) of node k's expected latency to the destination. When two nodes meet, they update their meeting rate estimations using their most recent inter-meeting times. They also exchange their current posterior distributions for their expected latencies by just exchanging the mean and standard deviation of the approximated normal distribution.

4.3 BMinLat-I

In this section, we introduce our first proposed Bayesian routing algorithm, BMinLat-I. Algorithm 4 shows how BMinLat-I works in detail. As mentioned in Section 4.2, BMinLat-I follows the same logic as in MinLat. However, it tries to deal with the fact that meeting rates and estimated expected latencies are random by finding the forwarding rules that minimize the expectation of estimated expected latencies. BMinLat-I intends to find the solution to the following optimization problem.

$$\mathbf{p}_i^{*1} = \underset{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}}{\arg\min}\, \mathbb{E}_{\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i}\big[g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big)\big] \tag{4.13}$$

where

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i}\big[g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big)\big] &= \int\int g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big) f(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i) d_{\boldsymbol{\lambda}_i} d_{\boldsymbol{\Psi}_i} \\ &= \int\int g_i\big(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i, \mathbf{p}_i\big) f(\boldsymbol{\Psi}_i | \boldsymbol{\lambda}_i, \mathbb{X}_i) f(\boldsymbol{\lambda}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i) d_{\boldsymbol{\lambda}_i} d_{\boldsymbol{\Psi}_i}\end{aligned} \tag{4.14}$$

Since (4.14) does not have a closed form representation, we approximate it with an empirical distribution based on N_s samples, i.e.,

$$f(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i) = \frac{1}{N_s} \sum_{s=1}^{N_s} \delta(\boldsymbol{\lambda}_i - \boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i - \boldsymbol{\Psi}_i^s) \tag{4.15}$$

where

$$\boldsymbol{\lambda}_i^s = \begin{bmatrix} \lambda_{i1}^s \\ \lambda_{i2}^s \\ \vdots \\ \lambda_{i|\mathcal{S}_i|}^s \end{bmatrix} \quad , \quad \boldsymbol{\Psi}_i^s = \begin{bmatrix} \lambda_{i1}^s L_{1d}^s(i) \\ \lambda_{i2}^s L_{2d}^s(i) \\ \vdots \\ \lambda_{i|\mathcal{S}_i|}^s L_{|\mathcal{S}_i|d}^s(i) \end{bmatrix}$$

Samples are independently generated using the posterior distributions described in Section 4.2, i.e.,

$$\forall s = 1, \ldots, N_s : \quad \lambda_{ik}^s \sim f(\lambda_{ij} | x_{ij}^1, \ldots, x_{ij}^{n_{ij}}, \zeta_{ij}) \tag{4.16}$$

and

$$\forall s = 1, \ldots, N_s : \quad L_{kd}^s(i) \sim f(L_{kd}(i) | x_{ij}^1 \ldots, x_{ij}^{n_{ij}}; j \in \mathcal{S}_i) \tag{4.17}$$

Algorithm 4 BMinLat-I

1: `// Initialization`

2: $\mathbf{P}^{*1} = \mathbf{0}_{N\times N}$

3: $\forall i,j \in \mathcal{N}: \quad n_{ij}=0, l_{ij}=0, \kappa_{ij}^{\zeta}=\kappa_0, \theta_{ij}^{\zeta}=\theta_0$

4: $\mu_d^d = 0$, $\sigma_d^d = 0$, $\forall i \in \mathcal{N}/d: \quad \mu_i^i=\infty, \sigma_i^i=0$

5: **while** Nodes continue to meet **do**

6: `// Nodes` i `and` j `meet at time` t

7: Set $\mu_i^j = \mu_i^i$ and $\sigma_i^j = \sigma_i^i$

8: Set $\mu_j^i = \mu_j^j$ and $\sigma_j^i = \sigma_j^j$

9: Update $\kappa_{ik}^{\zeta} = \kappa_{ik}^{\zeta} + 1$ and $\theta_{ik}^{\zeta} = \theta_{ik}^{\zeta} + t - l_{ij}$

10: Generate N_s samples $\{\lambda_{ik}^s\}_{s=1}^{N_s}$ from $\mathbf{\Gamma}(\kappa_{ik}^{\zeta}, \theta_{ik}^{\zeta})$

11: For each $k \in \mathcal{S}_i$, generate N_s samples $\{L_{kd}^s(i)\}_{s=1}^{N_s}$ from $\mathcal{N}(\mu_k^i, \sigma_k^i)$

12: For each $k \in \mathcal{S}_j$, generate N_s samples $\{L_{kd}^s(j)\}_{s=1}^{N_s}$ from $\mathcal{N}(\mu_k^j, \sigma_k^j)$

13: Set $\mathbf{p}_i^* = \arg\min_{\mathbf{p}_i\in[0,1]^{|\mathcal{S}_i|}} G_i\big(\mathbb{S}_{\boldsymbol{\lambda}_i}, \mathbb{S}_{\boldsymbol{\Psi}_i}, \mathbf{p}_i, N_s\big)$ and $L_{id}^s(i) = g_i\big(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s, \mathbf{p}_i^*\big)$

14: Set $\mathbf{p}_j^* = \arg\min_{\mathbf{p}_j\in[0,1]^{|\mathcal{S}_j|}} G_j\big(\mathbb{S}_{\boldsymbol{\lambda}_j}, \mathbb{S}_{\boldsymbol{\Psi}_j}, \mathbf{p}_j, N_s\big)$ and $L_{jd}^s(j) = g_j\big(\boldsymbol{\lambda}_j^s, \boldsymbol{\Psi}_j^s, \mathbf{p}_j^*\big)$

15: Set $\mathbf{p}_i^{*1} = \mathbf{p}_i^*$ and $\mathbf{p}_j^{*1} = \mathbf{p}_j^*$

16: Set $\mu_i^i = \frac{1}{N_s}\sum_{s=1}^{N_s} L_{id}^s(i)$ and $\mu_j^j = \frac{1}{N_s}\sum_{s=1}^{N_s} L_{jd}^s(j)$

17: Set $\sigma_i^i = \sqrt{\frac{1}{N_s}\sum_{s=1}^{N_s}(L_{id}^s(i)-\mu_i^i)^2}$ and $\sigma_j^j = \sqrt{\frac{1}{N_s}\sum_{s=1}^{N_s}(L_{jd}^s(j)-\mu_j^j)^2}$

18: Set $l_{ij} = t$

19: **end while**

We use the notations $\mathbb{S}_{\boldsymbol{\lambda}_i}$ and $\mathbb{S}_{\boldsymbol{\Psi}_i}$ to show the set of generated samples $\boldsymbol{\lambda}_i^s$ and $\boldsymbol{\Psi}_i^s$, i.e.,

$$\mathbb{S}_{\boldsymbol{\lambda}_i} = \{\lambda_{ik}^s | \forall s = 1,\dots,N_s, \forall k \in \mathcal{S}_i\} \tag{4.18}$$

$$\mathbb{S}_{\boldsymbol{\Psi}_i} = \{\lambda_{ik}^s L_{kd}^s(i) | \forall s = 1,\dots,N_s, \forall k \in \mathcal{S}_i\} \tag{4.19}$$

By replacing (4.15) in (4.14), we have

$$\mathbb{E}_{\boldsymbol{\lambda}_i,\boldsymbol{\Psi}_i}\big[g_i\big(\boldsymbol{\lambda}_i,\boldsymbol{\Psi}_i,\mathbf{p}_i\big)\big]=\frac{1}{N_s}\sum_{s=1}^{N_s}g_i\big(\boldsymbol{\lambda}_i^s,\boldsymbol{\Psi}_i^s,\mathbf{p}_i\big) \tag{4.20}$$

Hence, BMinLat-I finds the forwarding rule that minimizes the average of the objective function $g_i\big(\boldsymbol{\lambda}_i,\boldsymbol{\Psi}_i,\mathbf{p}_i\big)$ over N_s generated samples. Denoting this average by

$$G_i\big(\mathbb{S}_{\boldsymbol{\lambda}_i},\mathbb{S}_{\boldsymbol{\Psi}_i},\mathbf{p}_i,N_s\big)=\frac{1}{N_s}\sum_{s=1}^{N_s}g_i\big(\boldsymbol{\lambda}_i^s,\boldsymbol{\Psi}_i^s,\mathbf{p}_i\big) \tag{4.21}$$

the optimization problem (4.13), can be re-written as

$$\mathbf{p}_i^{*1}=\underset{\mathbf{p}_i\in[0,1]^{|\mathcal{S}_i|}}{\arg\min}\quad G_i\big(\mathbb{S}_{\boldsymbol{\lambda}_i},\mathbb{S}_{\boldsymbol{\Psi}_i},\mathbf{p}_i,N_s\big) \tag{4.22}$$

Using the forwarding rule $\mathbf{p}_i^{*1}$, N_s samples of the expected latency of node i will be generated as

$$L_{id}^s(i)=g_i\big(\boldsymbol{\lambda}_i^s,\boldsymbol{\Psi}_i^s,\mathbf{p}_i^{*1}\big) \tag{4.23}$$

The posterior distributions of expected latencies are updated using the generated samples $L_{id}^s(i)$.

The main challenge in performing BMinLat-I is solving the optimization problem (4.22). Some studies such as [113–115] refer to the functions of the form $G_i\big(\mathbb{S}_{\boldsymbol{\lambda}_i},\mathbb{S}_{\boldsymbol{\Psi}_i},\mathbf{p}_i,N_s\big)$ as "sum of linear ratios" and investigate how an optimization problem with an objective function of this form can be solved. In the rest of this section, we overview the solution approach proposed in [113], and then use it to evaluate the computational complexity of BMinLat-I. In Chapter 3, we explained how the optimization problem in (4.3) can be converted to a linear optimization. [113] uses the same parameter conversions (the Charnes-Cooper transformation [108]) used in Lemma 2, i.e.,

$$\begin{aligned}\mathbf{x}_i^s&=\frac{1}{\boldsymbol{\lambda}_i^{sT}\mathbf{p}_i}\mathbf{p}_i\\ y_i^s&=\frac{1}{\boldsymbol{\lambda}_i^{sT}\mathbf{p}_i}\end{aligned} \tag{4.24}$$

Theorem 1 in [113] proves that the optimization problem in (4.22) is equivalent to the following optimization problem.

$$\begin{aligned}
&\arg\min_{\mathbf{x}_i^s, y_i^s} \quad \frac{1}{N_s}\sum_{s=1}^{N_s} \mathbf{\Psi}_i^s \mathbf{x}_i^s + y_i^s \\
&\boldsymbol{\lambda}_i^{sT}\mathbf{x}_i^s = 1 \qquad\qquad\qquad\qquad\qquad\qquad \forall s = 1,\dots,N_s \\
&\mathbf{A}\mathbf{x}_i^s - \mathbf{b} \leq 0 \quad \forall s = 1,\dots,N_s \\
&\frac{1}{\beta_s} \leq y_i^s \leq \frac{1}{\alpha_s} \quad \forall s = 1,\dots,N_s \\
&\mathbf{x}_i^{s_1} y_i^{s_2} = \mathbf{x}_i^{s_2} y_i^{s_1} \quad \forall s_1, s_2 = 1,\dots,N_s
\end{aligned} \tag{4.25}$$

where $\mathbf{A}_{2|\mathcal{S}_i|\times|\mathcal{S}_i|}$ and $\mathbf{b}_{2|\mathcal{S}_i|\times 1}$ are the same as defined in Section 3.8.5, i.e.,

$$\mathbf{A}_{ij} = \begin{cases} 1 & \text{if } i < |\mathcal{S}_i| \quad \& \quad j = i \\ -1 & \text{if } |\mathcal{S}_i| < i < 2|\mathcal{S}_i| \quad \& \quad j = i - |\mathcal{S}_i| \\ 0 & \text{otherwise} \end{cases} \quad , \quad \mathbf{b}_{2|\mathcal{S}_i|\times 1} = [1,...,1,0,...,0]^T$$

and

$$\begin{aligned} \alpha_s = \min\ & \boldsymbol{\lambda}_i^{sT}\mathbf{p}_i \\ & \mathbf{A}\mathbf{p}_i - \mathbf{b} \leq 0 \end{aligned} \tag{4.26}$$

$$\begin{aligned} \beta_s = \max\ & \boldsymbol{\lambda}_i^{sT}\mathbf{p}_i \\ & \mathbf{A}\mathbf{p}_i - \mathbf{b} \leq 0 \end{aligned} \tag{4.27}$$

Therefore, in order to globally solve (4.22), we may solve (4.25). However, the constraints $\mathbf{x}_i^{s_1} y_i^{s_2} = \mathbf{x}_i^{s_2} y_i^{s_1}$ for $s_1, s_2 = 1,\dots,N_s$ in (4.22) are quadratic and non-convex. Hence, this problem belongs to the category of non-convex quadratic programming (which is in general NP-hard).

If we make a relaxation for (4.25) by discarding $\mathbf{x}_i^{s_1} y_i^{s_2} = \mathbf{x}_i^{s_2} y_i^{s_1}$ for $s_1, s_2 = 1,\dots,N_s$, we will have a linear optimization problem whose solution is a lower bound for problem (4.22). [113] devises a linear relaxation of $\mathbf{x}_i^{s_1} y_i^{s_2} = \mathbf{x}_i^{s_2} y_i^{s_1}$ with $-\mathbf{x}_i^s + \mathbf{l} y_i^s \leq 0$ and $\mathbf{x}_i^s - \mathbf{u} y_i^s \leq 0$ where $\mathbf{l} = [l_1,\dots,l_{|\mathcal{S}_i|}]^T$ and $\mathbf{u} = [u_1,\dots,u_{|\mathcal{S}_i|}]^T$ are two boundary vectors in $\mathbb{R}^{|\mathcal{S}_i|}$. The result is the following linear optimization problem that can be used to perform a branch-

and-bound [116] search to find the optimal solution for (4.22).

$$\begin{aligned}
&\arg\min_{\mathbf{x}_i^s, y_i^s} \quad \frac{1}{N_s}\sum_{s=1}^{N_s} \mathbf{\Psi}_i^s \mathbf{x}_i^s + y_i^s \\
&\boldsymbol{\lambda}_i^{sT}\mathbf{x}_i^s = 1 \quad \forall s = 1, \ldots, N_s \\
&\mathbf{A}\mathbf{x}_i^s - \mathbf{b} \leq 0 \quad \forall s = 1, \ldots, N_s \\
&\frac{1}{\beta_s} \leq y_i^s \leq \frac{1}{\alpha_s} \quad \forall s = 1, \ldots, N_s \\
&-\mathbf{x}_i^s + \mathbf{l}y_i^s \leq 0 \quad \forall s = 1, \ldots, N_s \\
&\mathbf{x}_i^s - \mathbf{u}y_i^s \leq 0 \quad \forall s = 1, \ldots, N_s
\end{aligned} \tag{4.28}$$

The j^{th} component of $\mathbf{l}$ and $\mathbf{u}$ can be obtained by solving the following linear programming problems,

$$\begin{aligned}
&l_j = \min p_{ij} \\
&\mathbf{A}\mathbf{p}_i - \mathbf{b} \leq 0
\end{aligned} \tag{4.29}$$

$$\begin{aligned}
&u_j = \max p_{ij} \\
&\mathbf{A}\mathbf{p}_i - \mathbf{b} \leq 0
\end{aligned} \tag{4.30}$$

We can now evaluate the computational complexity of BMinLat-I (Algorithm 4) using some discussions made in [113] about the convergence speed of the proposed branch-and-bound algorithm.

4.3.1 Computational Complexity

Let us denote the Euclidean norm of a vector by $||.||$, and define τ_s, ψ, and μ as

$$\begin{aligned}
\tau_s \triangleq \max_{\mathbf{p}_i} \; &\mathbf{\Psi}_i^{sT}\mathbf{p}_i \\
&\mathbf{A}\mathbf{p}_i - \mathbf{b} \leq 0
\end{aligned} \tag{4.31}$$

$$\psi \triangleq \max_s ||\boldsymbol{\lambda}_i^s|| \quad , \quad \mu \triangleq \max_s ||\mathbf{\Psi}_i^s|| \tag{4.32}$$

Theorem 4 in [113] proves that for a given tolerance ϵ_0, the algorithm proposed in [113] finds a solution $\mathbf{p}_i^{\epsilon}$ of problem (4.25) such that,

$$|G_i(\mathbb{S}_{\boldsymbol{\lambda}_i}, \mathbb{S}_{\boldsymbol{\Psi}_i}, \mathbf{p}_i^{\epsilon}, N_s) - G_i(\mathbb{S}_{\boldsymbol{\lambda}_i}, \mathbb{S}_{\boldsymbol{\Psi}_i}, \mathbf{p}_i^{*}, N_s)| \leq \epsilon_0 \tag{4.33}$$

in at most N_i^I iterations, for

$$N_i^I = (|\mathcal{S}_i| + 1)\Big\lceil \log_2(2N_s \frac{\psi \max_s \tau_s + \mu}{\epsilon_0}) \Big\rceil \tag{4.34}$$

Assuming that at each iteration of the proposed branch and bound algorithm the linear program can be solved in the polynomial order of $P(|\mathcal{S}_i|)$, node i needs to solve a problem with the worst case complexity order of $O(N_i^I)P(|\mathcal{S}_i|)$ whenever it meets another node. Hence, in the worst case scenario where $\mathcal{S}_i = \mathcal{N}$, node i should solve an optimization problem of order $O\big(N \log_2(N_s)\big)P(N)$.

4.4 BMinLat-II

In this section, we develop a heuristic algorithm called BMinLat-II to find the optimal forwarding solution. As mentioned in Section 4.2, we intend to find the expectation of the optimal forwarding rules, $h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i)$, with respect to meeting rates. Expanding Equation (4.7) we have,

$$\begin{aligned} \mathbf{p}_i^{*1} &= \mathbb{E}_{\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i}\big[h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i)\big] \\ &= \int\int h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i) f(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i) d_{\boldsymbol{\lambda}_i} d_{\boldsymbol{\Psi}_i} \\ &= \int\int h_i(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i) f(\boldsymbol{\Psi}_i | \boldsymbol{\lambda}_i, \mathbb{X}_i) f(\boldsymbol{\lambda}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i) d_{\boldsymbol{\lambda}_i} d_{\boldsymbol{\Psi}_i} \end{aligned} \tag{4.35}$$

As explained in Section 4.3, $f(\boldsymbol{\lambda}_i, \boldsymbol{\Psi}_i | \mathbb{X}_i, \boldsymbol{\zeta}_i)$ does not have a closed form representation. We again approximate it with an empirical distribution based on N_s samples. Therefore, we have

$$\mathbf{p}_i^{*2} = \frac{1}{N_s} \sum_{\boldsymbol{\lambda}_i^s \in \mathbb{S}_{\boldsymbol{\lambda}_i}} \sum_{\boldsymbol{\Psi}_i^s \in \mathbb{S}_{\boldsymbol{\Psi}_i}} h_i(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s) \tag{4.36}$$

We need to solve the optimization (4.5) for each sample $s = 1, \ldots, N_s$ independently and

find the binary forwarding rule $\mathbf{p}_i^{*s}$, i.e.,

$$\mathbf{p}_i^{*s} = \underset{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}}{\arg\min} \quad g_i\big(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s, \mathbf{p}_i\big) \tag{4.37}$$

Using these N_s forwarding rules, N_s samples of the expected latency of node i will be generated.

$$L_{id}^s(i) = g_i\big(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s, \mathbf{p}_i^{*s}\big) \tag{4.38}$$

These generated samples are used to update the posterior distribution of (4.12). Then, a non-binary forwarding rule $\mathbf{p}_i^{*2}$ is assigned to node i that equals the average of all the binary forwarding rules $\mathbf{p}_i^{*s}$. Algorithm 5 shows BMinLat-II in detail.

In the rest of this section, we study the computational complexity of BMinLat-II and compare it with BMinLat-I. We then evaluate the performance of the proposed Bayesian algorithm in Section 4.5.

4.4.1 Computational Complexity

In Chapter 3, we showed that for each sample s, the optimization problem in (4.37) can be converted to a linear program and can be solved in polynomial order $P(|\mathcal{S}_i|)$. In Algorithm 5, this optimization problem needs to be solved for each one of the N_s samples. Therefore, node i needs to run an algorithm with computational complexity of order $N_s P(|\mathcal{S}_i|)$. In the worst case, where $\mathcal{S}_i = \mathcal{N} - \{i\}$, the computational complexity of the algorithm is $N_s P(N)$.

Recall that the worst-case complexity of BMinLat-I is $O\big(N \log_2(N_s)\big)P(N)$. Based on the size of the network (i.e., N) and the number of generated samples (i.e., N_s), either BMinLat-I or BMinLat-II may be more computationally efficient. If $N \ll N_s$, then BMinLat-I is a better choice. However, if choosing N_s to be of the same order as N can result in acceptable performance, then it is more computationally efficient to use BMinLat-II.

4.5 Simulation Results

In this section, we compare the performance of the two proposed Bayesian routing algorithms for opportunistic networks. We first implement the algorithms in synthetic networks where the modelling assumptions are exactly met. We then show how these methods can be

Algorithm 5 BMinLat-II

1: `// Initialization`

2: $\mathbf{P} = \mathbf{0}_{N\times N}$

3: $\forall i,j \in \mathcal{N}: \quad n_{ij}=0, l_{ij}=0, \kappa_{ij}^{\zeta}=\kappa_0, \theta_{ij}^{\zeta}=\theta_0$

4: $\mu_d^d = 0$, $\sigma_d^d = 0$, $\forall i \in \mathcal{N}/d: \quad \mu_i^i=\infty, \sigma_i^i=0$

5: **while** Nodes continue to meet **do**

6: `// Nodes` i `and` j `meet at time` t

7: Set $\mu_i^j = \mu_i^i$ and $\sigma_i^j = \sigma_i^i$

8: Set $\mu_j^i = \mu_j^j$ and $\sigma_j^i = \sigma_j^j$

9: Update $\kappa_{ik}^{\zeta} = \kappa_{ik}^{\zeta} + 1$ and $\theta_{ik}^{\zeta} = \theta_{ik}^{\zeta} + t - l_{ij}$

10: Generate N_s samples $\{\lambda_{ik}^s\}_{s=1}^{N_s}$ from $\mathbf{\Gamma}(\kappa_{ik}^{\zeta}, \theta_{ik}^{\zeta})$

11: For each $k \in \mathcal{S}_i$, generate N_s samples $\{L_{kd}^s(i)\}_{s=1}^{N_s}$ from $\mathcal{N}(\mu_k^i, \sigma_k^i)$

12: For each $k \in \mathcal{S}_j$, generate N_s samples $\{L_{kd}^s(j)\}_{s=1}^{N_s}$ from $\mathcal{N}(\mu_k^j, \sigma_k^j)$

13: Set $\mathbf{p}_i^{*s} = \arg\min_{\mathbf{p}_i \in [0,1]^{|\mathcal{S}_i|}} g_i\big(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s, \mathbf{p}_i\big)$ and $L_{id}^s(i) = g_i\big(\boldsymbol{\lambda}_i^s, \boldsymbol{\Psi}_i^s, \mathbf{p}_i^{*s}\big)$

14: Set $\mathbf{p}_j^{*s} = \arg\min_{\mathbf{p}_j \in [0,1]^{|\mathcal{S}_j|}} g_j\big(\boldsymbol{\lambda}_j^s, \boldsymbol{\Psi}_j^s, \mathbf{p}_j\big)$ and $L_{jd}^s(j) = g_j\big(\boldsymbol{\lambda}_j^s, \boldsymbol{\Psi}_j^s, \mathbf{p}_j^{*s}\big)$

15: Set $\mathbf{p}_i = \frac{1}{N_s}\sum_{s=1}^{N_s} \mathbf{p}_i^{*s}$ and $\mathbf{p}_j = \frac{1}{N_s}\sum_{s=1}^{N_s} \mathbf{p}_j^{*s}$

16: Update $\mu_i^i = \frac{1}{N_s}\sum_{s=1}^{N_s} L_{id}^s(i)$ and $\mu_j^j = \frac{1}{N_s}\sum_{s=1}^{N_s} L_{jd}^s(j)$

17: Update $\sigma_i^i = \sqrt{\frac{1}{N_s}\sum_{s=1}^{N_s}(L_{id}^s(i) - \mu_i^i)^2}$ and $\sigma_j^j = \sqrt{\frac{1}{N_s}\sum_{s=1}^{N_s}(L_{jd}^s(j) - \mu_j^j)^2}$

18: Set $l_{ij} = t$

19: **end while**

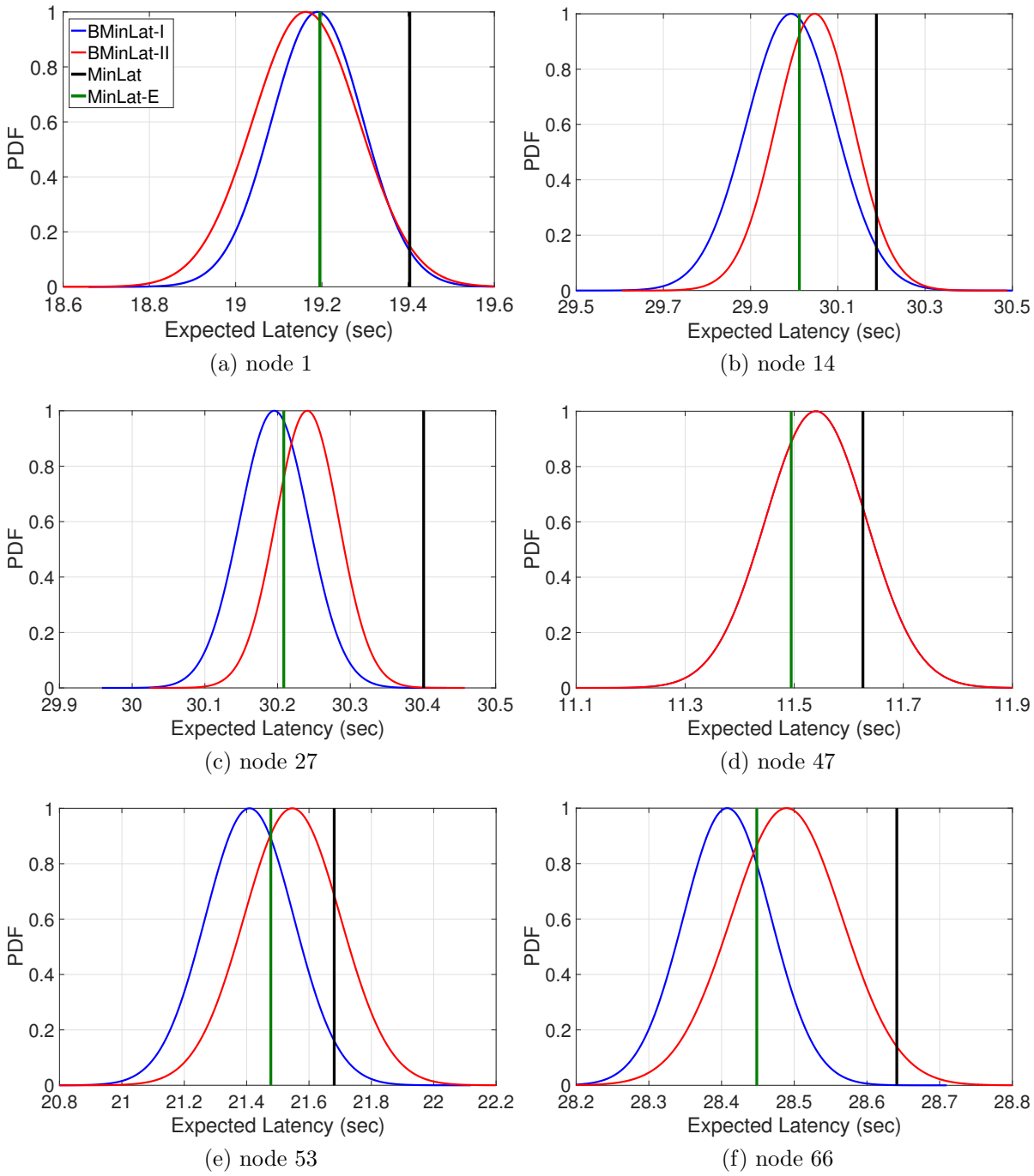
used in practice by testing the algorithms on a dataset generated based on the Sigcomm09 dataset [117].

We implement the two proposed Bayesian algorithms on a preferentially attached network of $N = 100$ nodes. The preferentially attached contact graph is constructed in the same way explained in Section 3.6 with parameters $m_0 = 2$ and $m = 2$. If there exists an edge between nodes i and j in the contact graph, a positive random meeting rate $\lambda_{ij} > 0$ is assigned to the pair. Otherwise, $\lambda_{ij} = 0$. The positive meeting rates are generated

from a Gamma distribution $\mathbf{\Gamma}(1.5, 0.1)$. Therefore, the average meeting rate between pairs of nodes with positive meeting rates is 9 meetings per minute. One of the N nodes is randomly chosen as the destination node based on a uniform probability distribution. We let the nodes meet each other for the equivalent time of almost 28 hours. We then check the posterior probability distribution function that each node has estimated of its expected latency to the destination node using each of the two algorithms. In order to run the algorithms, we use the initial values $\kappa_0 = \theta_0 = 0$ for any $i, j \in \mathcal{N}$ with $\lambda_{ij} > 0$ and generate $N_s = 10$ samples.

Figure 4.1 shows the probability distribution function of the expected latencies of 6 nodes for both BMinLat-I and BMinLat-II at the end of 28 hours of running the simulations on the network. It also demonstrates the expected latencies of MinLat (where the meeting rates are known) and MinLat-E (where Maximum Likelihood (ML) estimation of meeting rates is used). As we see in the figure, at the end of the 28 hours of running the network, the probability distribution of what BMinLat-I and BMinLat-II estimate are tight and close to MinLat-E. However, we see that in the specific case of Figure 4.1, the expected latencies that BMinLat-I, BMinLat-II, and MinLat-E estimate are underestimates of the result of MinLat. This can be explained by taking a closer look at the nodes that have the most frequent meetings with the destination nodes. In the simulation setup of Figure 4.1, node 78 is the destination node. In the random contact graph of this setup, the destination node only meets nodes 30 and 47 with respective rates of 2.00 and 5.16 meetings per minute. Hence, node 47 has the most frequent contacts with the destination and it only forwards its messages to the destination. Most of the other nodes need to pass through this node to send their messages to the destination.

The differences we observe between the estimates of MinLat and the other algorithms (BMinLat-I, BMinLat-II, and MinLat-E) can be justified by a theoretical understanding of the confidence interval associated with the Maximum Likelihood (ML) estimates. The 95% confidence intervals for ML estimates have ranges of $1 \pm \frac{1.96}{\sqrt{n}}$ where n is the number of samples. After 28 hours, a pair of nodes that meet with rate 5.16 meetings per minute have met each other 8668.8 times in average. Therefore, their ML estimate of their meeting rate is between 0.98 and 1.02 of its true value 95% of the time. Hence, what they estimate of the time it takes for a message to reach from one node to the other one is between 11.88 and 11.39 (instead of 11.63) hours. This shows that the difference of about 0.2 hour (12 minutes) is well within the confidence interval.



(a) node 1 (b) node 14

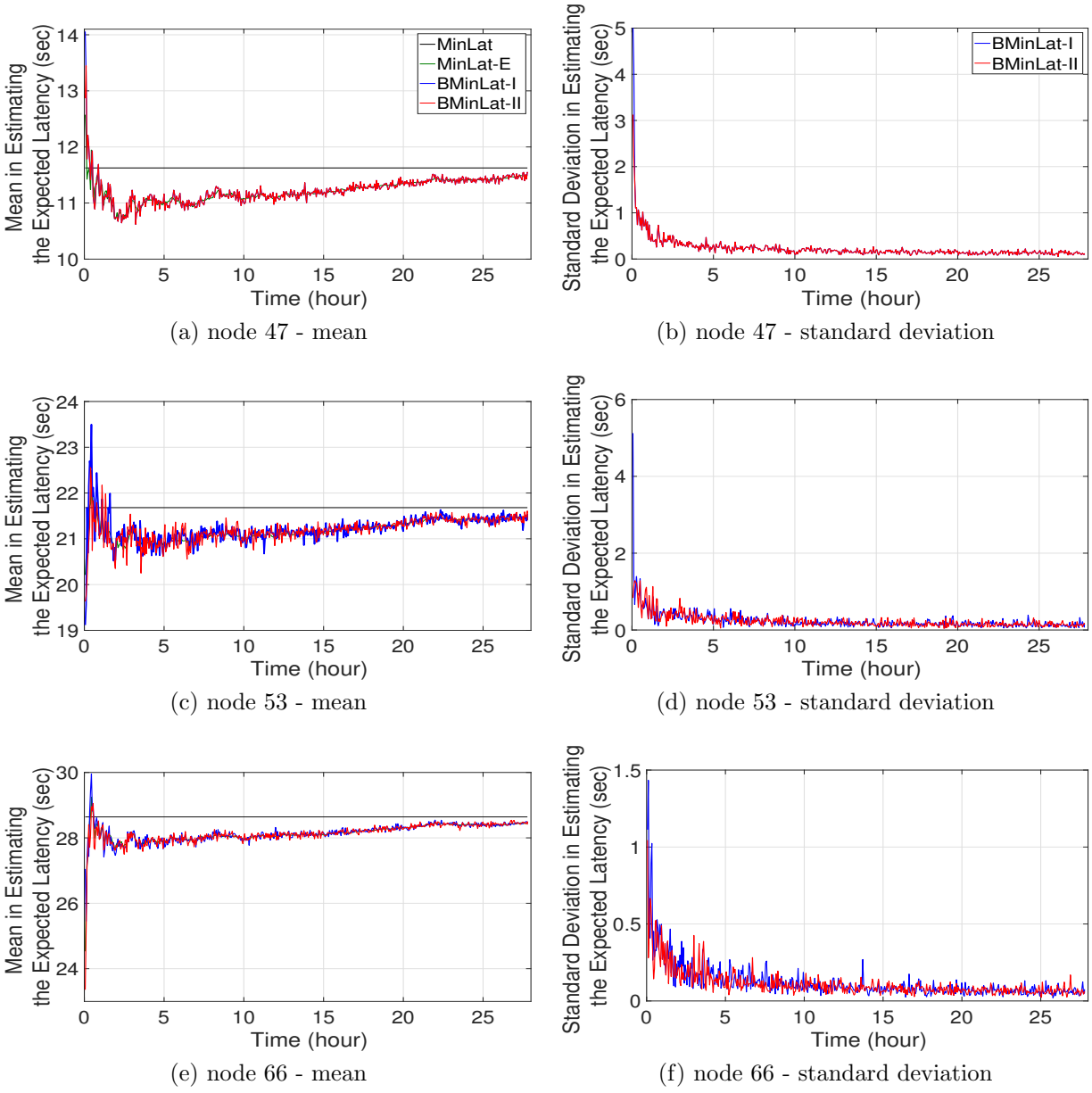
(c) node 27 (d) node 47

(e) node 53 (f) node 66

Figure 4.1 Probability distribution function of expected latencies of 12 nodes estimated by BMinLat-I, BMinLat-II, MinLat, and MinLat-E

As shown in Figure 4.1d, BMinLat-I and BMinLat-II estimate the exact same expected latencies for node 47 and it equals the inverse of the estimated meeting rate between nodes 47 and 78. In order to see how this estimate evolves through time, Figure 4.2 displays the mean and standard deviation of estimates of expected latencies for node 47 and three nodes over time. As we see in Figure 4.2a, the expected latencies that BMinLat-I and BMinLat-II estimate for node 47 converge to the result of MinLat from below. This causes other nodes to underestimate their expected latencies. Moreover, Figure 4.2 shows that not only do the means of the estimated expected latencies get close to what MinLat estimates, but also the standard deviations decrease and the probability distributions get tight.

Figures 4.1 and 4.2 show how individual nodes perform under BMinLat-I, BMinLat-II, and MinLat-E. In order to get an insight of how the entire network performs, Figure 4.3 shows some statistical characteristics of the differences between the expected latencies estimated by MinLat and the two Bayesian algorithms BMinLat-I and BMinLat-II for different nodes. It also displays the difference between MinLat and MinLat-E. The red lines show the median while the boxes display the 25^{th} and 75^{th} percentiles of estimated expected latencies over different nodes. Whiskers show maximum and minimum values. Moreover, 4.3d shows the median of the difference in BMinLat-I, BMinLat-II, and MinLat-E to make it easer to compare the results. As we see in the figure, the difference between the expected latencies estimated by all the three algorithms and MinLat decreases as the time passes. In addition, we see that the result of BMinLat-I and BMinLat-II are close to MinLat-E. This is because we did not use any external or social information about the ties between nodes in the simulations of this section. However, as explained in Section 4.1, the whole point of using a Bayesian framework is to make use of this information to expedite the process of finding the optimal forwarding rules. In the next section, we use a dataset generated based on a real-world dataset to show how we can use the proposed algorithm to achieve this goal. However, before moving on to the next section, we intend to investigate how the results change if we generate more samples.



(a) node 47 - mean

(b) node 47 - standard deviation

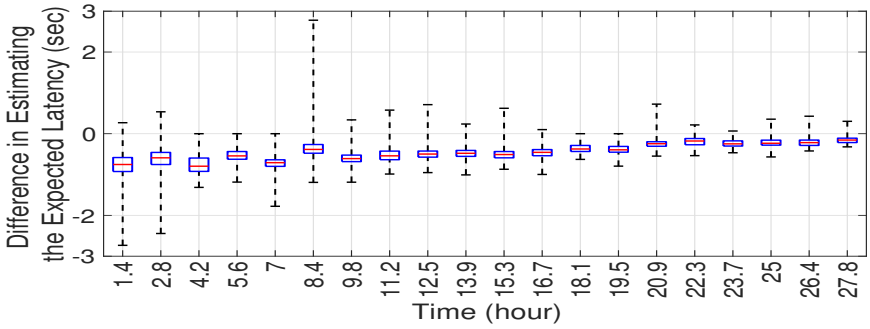
(c) node 53 - mean

(d) node 53 - standard deviation

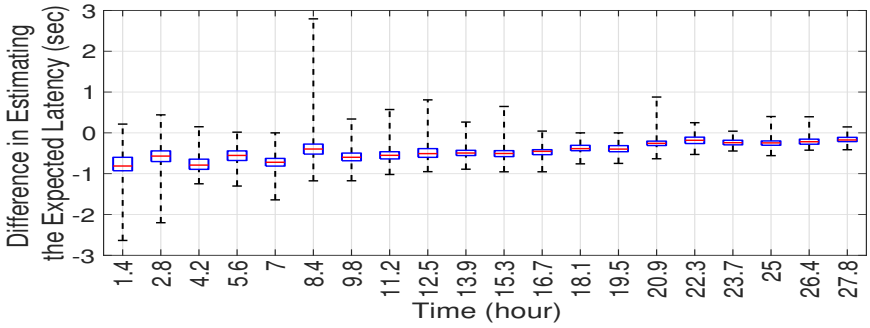
(e) node 66 - mean

(f) node 66 - standard deviation

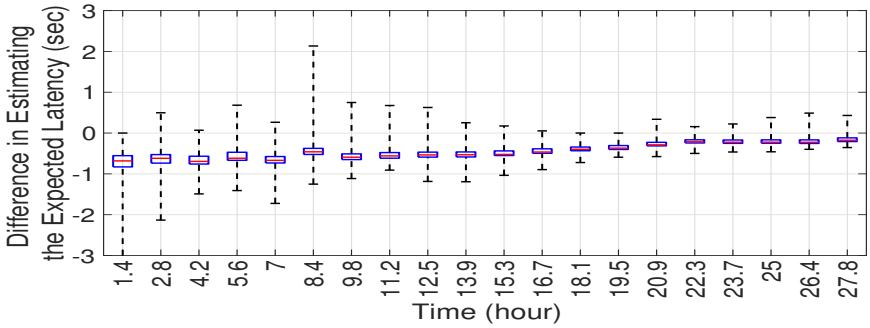
Figure 4.2 Evolution of estimated expected latency with time



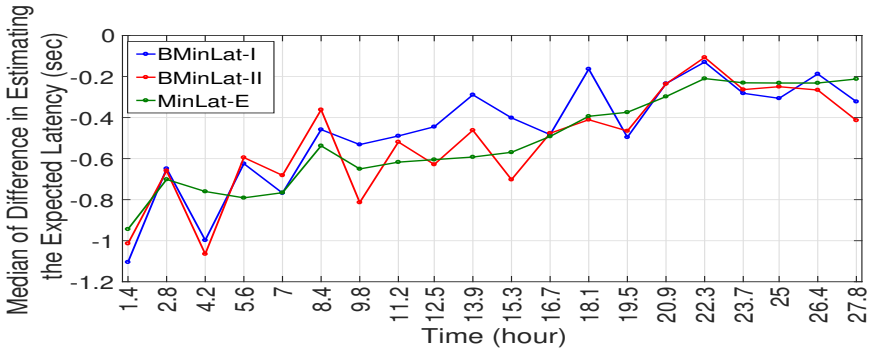
(a) BMinLat-I



(b) BMinLat-II



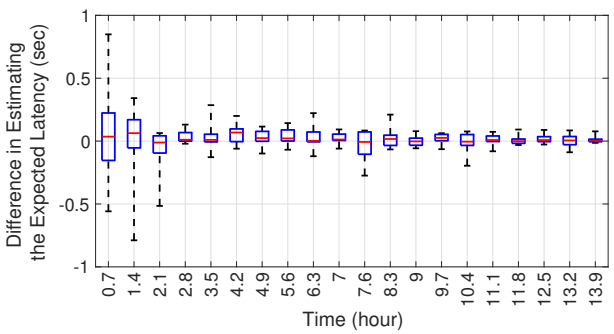
(c) MinLat-E



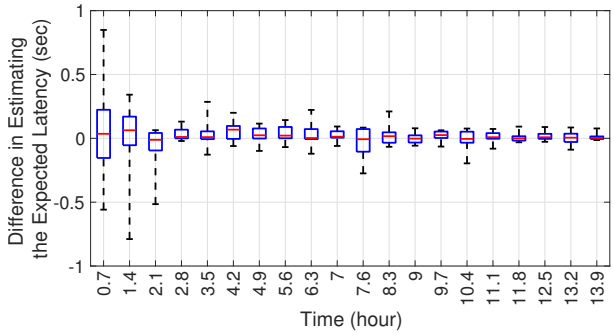
(d) Medians

Figure 4.3 Some statistical properties of the difference between the expected latencies estimated by MinLat and the two Bayesian algorithms BMinLat-I and BMinLat-II across different network nodes

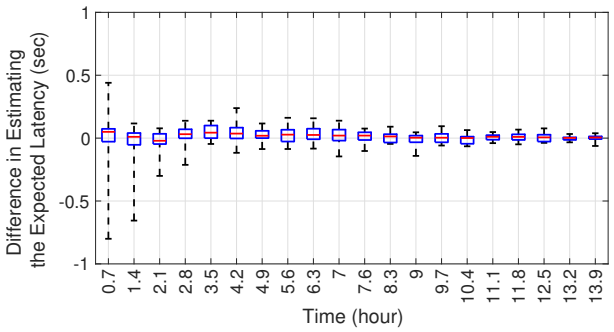
In order to show the impact of the number of generated samples on the results, we compare the performance of BMinLat-I and BMinLat-II for three different values of N_s. Figure 4.4 shows the same statistical characteristics as in Figure 4.3 for a preferentially attached network of $N = 10$ nodes for $N_s = 1, 10, 100$ samples. When only one sample is generated, BMinLat-I and BMinLat-II are exactly the same. When the number of generated samples are increased to 10, we see the difference terms shrink faster. However, the improvement is less significant when we increase the number of generated samples from 10 to 100. Figure 4.5 makes it easier to compare the results by just displaying the median of differences. We see that for higher number of generated samples, the results are closer to MinLat-E. Moreover, we observe that the variability of the estimates also changes with number of samples. Based on Figure 4.5b, the estimates are still changing relatively significantly after 11 hours. With 100 samples, however, there is very little variation.



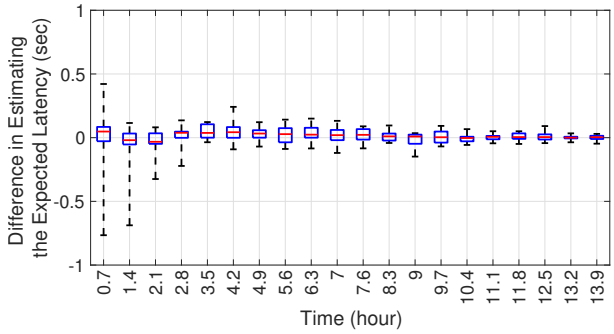
(a) BMinLat-I with 1 sample



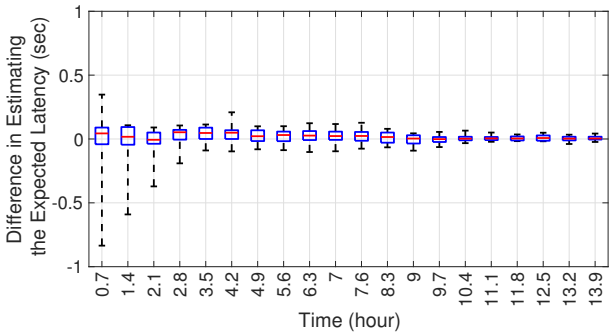
(b) BMinLat-II with 1 sample



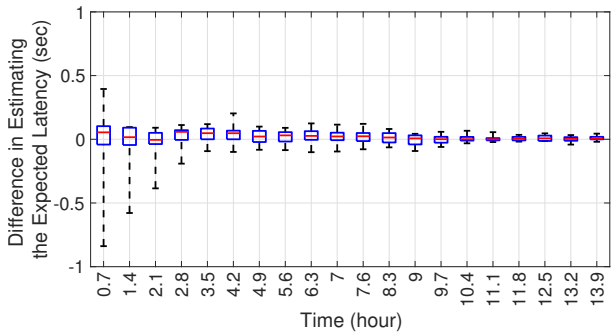
(c) BMinLat-I with 10 samples



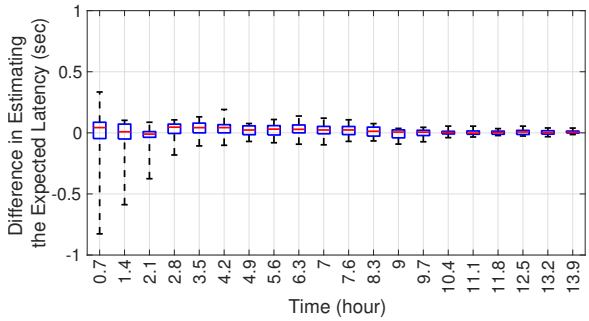
(d) BMinLat-II with 10 samples



(e) BMinLat-I with 100 samples

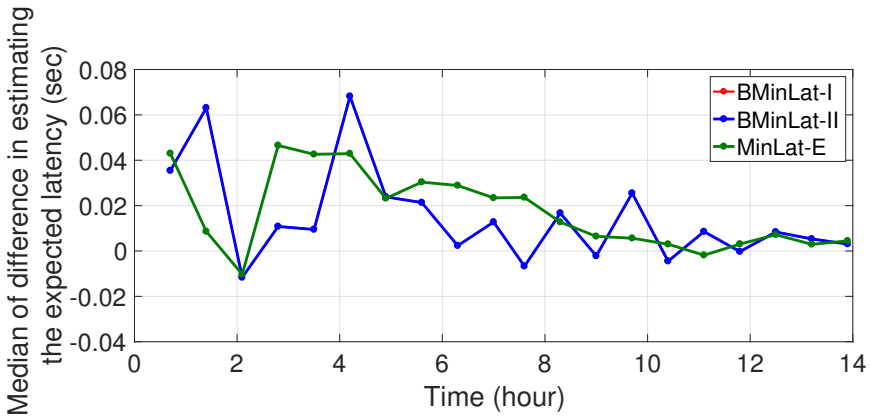


(f) BMinLat-II with 100 samples

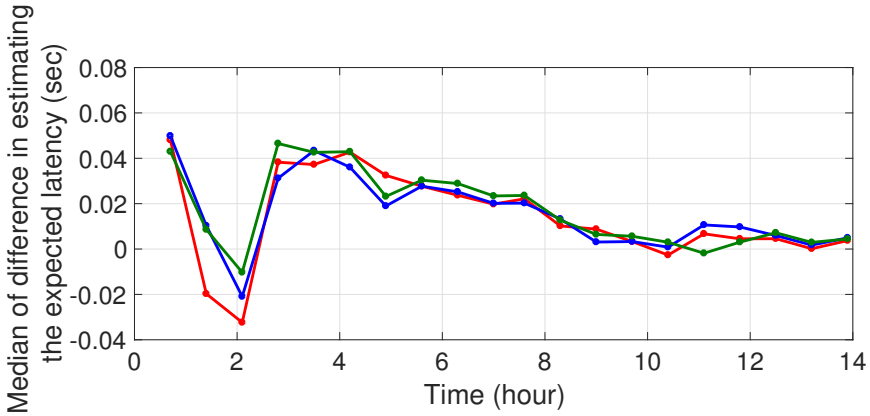


(g) MinLat-E

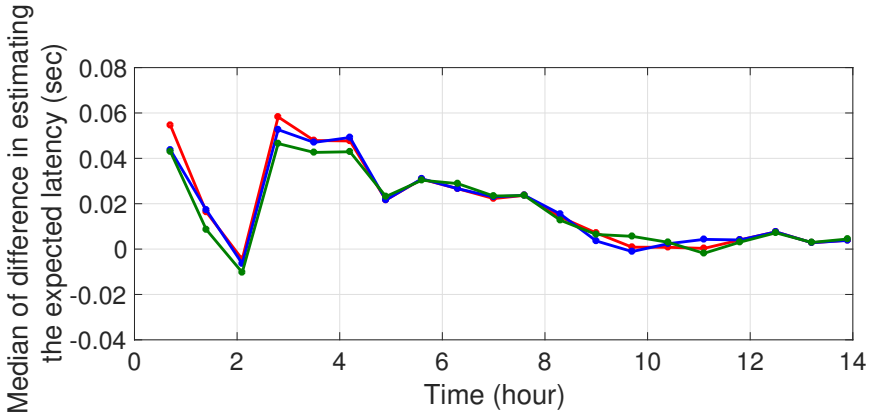
Figure 4.4 Difference between the expected latencies estimated by MinLat and the proposed algorithms BMinLat-I, BMinLat-II, MinLat-E across different network nodes for different number of generated samples



(a) 1 sample



(b) 10 samples



(c) 100 samples

Figure 4.5 Median of the difference between the expected latencies estimated by MinLat and BMinLat-I, BMinLat-II, MinLat-E

4.5.1 Informative Prior Distribution

In order to show how the proposed Bayesian methods can be used in practice, we test them on a dataset generated based on the Sigcomm09 dataset [117] collected at the Sigcomm 2009 conference at Barcelona, Spain. Around 100 smart phones were distributed to a set of volunteers during the first two days of the conference. Each device was initialized with the social profile of a participant including some basic information such as home city, country, and affiliation. In addition, each participant was asked to log in to his/her Facebook profile in order to include the list of Facebook friends and interests in the social profile. The participants were allowed to edit the social profile before it was uploaded on the device and recorded in the traces. Participants were instructed to keep the device with them and powered on at all times, and to use the MobiClique application for mobile social networking during the conference. The final trace contains data from 76 devices that show significant activity during the experiment. We use the rate of meetings in the first 12 hours of the conference and generate a dataset with exponential inter-meeting times for 24 hours. In order to apply the strength of the social ties between nodes, we define four different types of social relationships and use the information in the Sigcomm09 dataset, to assign the parameters for the prior gamma distribution to each group. Table 4.2 summarizes the assigned values. It also shows what percentage of the population each category forms and the average meeting rate in each group.

Table 4.2 Different types of social ties in the Sigcomm09 dataset

Property	ζ_{ij}	κ_0	θ_0	Population Percentage	Average Meeting Rate (1/hour)
i and j are friends on Facebook	1	1.76	1376	4.62	4.68
i and j have the same research interests	2	2.07	1959	17.37	3.81
i and j have both of the above properties	3	1.80	1462	9.14	4.44
i and j have none of the above properties	4	2.68	3026	68.87	3.19

Figure 4.6 shows the prior probability distribution functions for meeting rates of each category. We interpret the friendship on Facebook as a personal relationship and the same research interests as a professional relationship between two nodes. Table 4.2 and Figure 4.6 show that the stronger the personal and professional relationships between two nodes, the more frequently they meet. The hyperparameters shown in Table 4.2 are derived from the dataset. Although this is not a practical approach to chose hyperparameters, we just

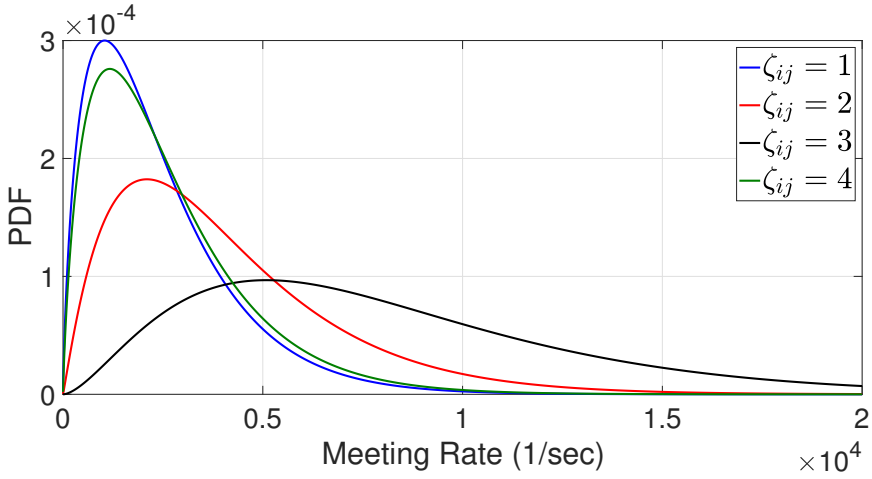
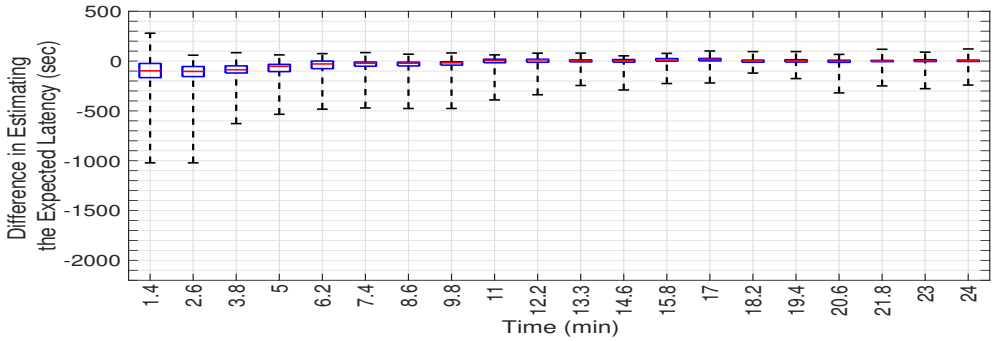


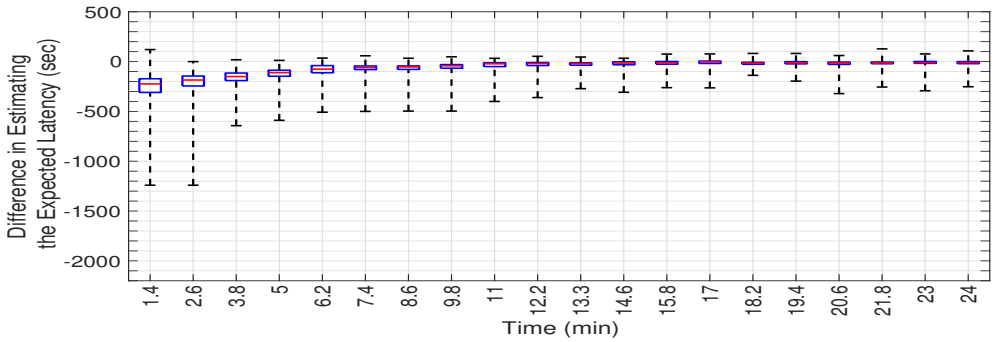
Figure 4.6 Prior probability distribution functions for meeting rates

use it to make the point about our proposed Bayesian framework. In practice, other related sources of information should be used to derive these hyperparameters. In the example of a conference event, the meeting rates of similar participants in the previous years may be used to choose the hyperparameters for prior distributions.

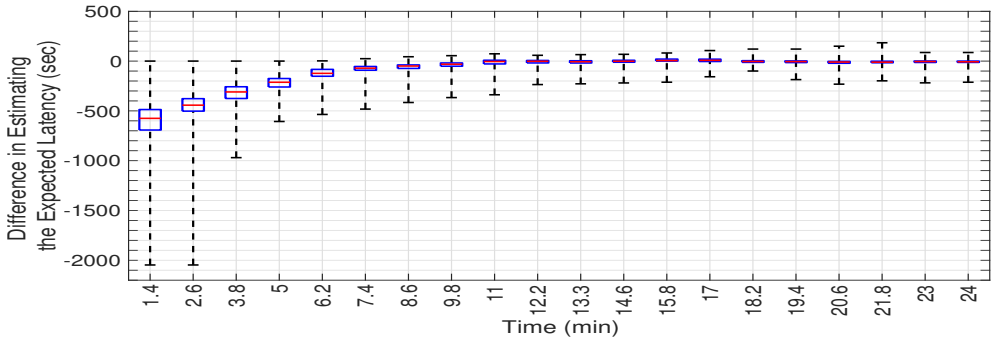
We compare the performance of BMinLat-I and BMinLat-II with these informative prior distributions with the performance of MinLat. Figure 4.7 shows the same statistical characteristics as in Figure 4.3 for the difference between the expected latencies estimated by MinLat and the algorithms BMinLat-I, BMinLat-II, and MinLat-E. As we see in Figure 4.7, the difference terms in BMinLat-I and BMinLat-II are smaller in the beginning compared to MinLat-E. We have shown all the median points in Figure 4.7d to more easily compare the algorithms. As we see in this figure, BMinLat-I has the smallest median values and it is followed by its heuristic version BMinLat-II. MinLat-E has much larger difference values in the beginning. This can be justified by the fact that MinLat-E does not have any prior information about the nodes' meeting rates. However, we see that the prior becomes almost irrelevant after some time and the three algorithms have the same ultimate results. Figure 4.8 shows the evolution of the mean and standard deviation of estimated expected latencies of some of the individual nodes. Again, we see that compared to MinLat-E, BMinLat-I and BMinLat-II result in expected latencies that are more similar to the result of MinLat in the beginning. We see in Figure 4.8 that MinLat-E largely underestimates the expected latencies achieved by MinLat in the beginning. The reason is



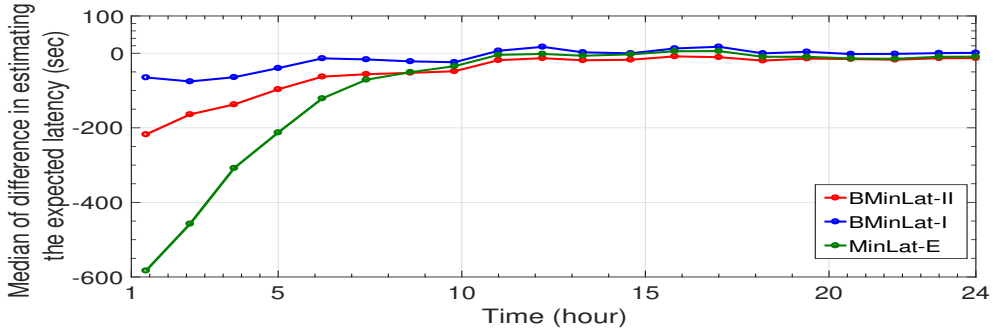
(a) BMinLat-I with informative prior



(b) BMinLat-II with informative prior

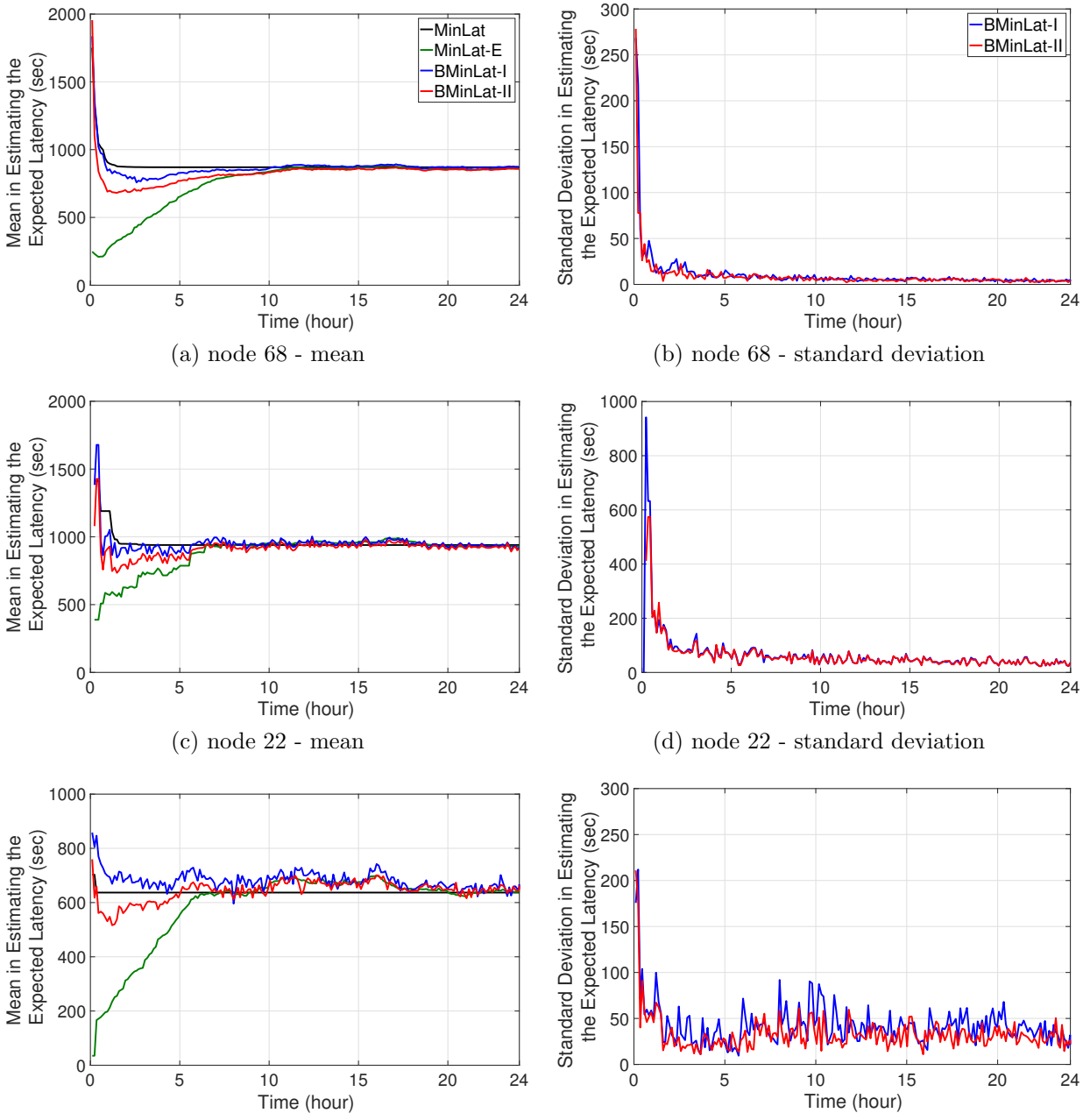


(c) MinLat-E



(d) Medians

Figure 4.7 Some statistical properties of the difference between the expected latencies estimated by MinLat and the proposed algorithms BMinLat-I, BMinLat-II, MinLat-E across different network nodes



(a) node 68 - mean

(b) node 68 - standard deviation

(c) node 22 - mean

(d) node 22 - standard deviation

(e) node 66 - mean

(f) node 66 - standard deviation

Figure 4.8 Evolution of estimated expected latency with time

that in the specific case of this simulation some critical nodes meet more frequently than their average meeting rate in the beginning. Therefore, what they think of their meeting rate is much larger than its true value.

4.6 Summary

In this chapter, we used a Bayesian framework to improve the performance of decentralized routing in opportunistic DTNs in scenarios where the meeting rates are unknown. The goal is to apply some external information about the social ties among the network nodes in addition to the nodes' contact histories. We developed two algorithms, BMinLat-I and BMinLat-II, to find the forwarding rules that nodes should use in order to route their messages to the destination node. The main idea in BMinLat-I is to find a forwarding rule which minimizes the expectation of the utility function with respect to random meeting rates. In order to solve the optimization problem, BMinLat-I samples from the posterior distributions of the meeting rates and expected latencies and identifies the average of objective functions over all the generated samples. The main idea in BMinLat-II, however, is to find the optimal binary solution for each sample independently. The ultimate forwarding rule that is dictated to a node is the average of optimal solutions over the sample set. This final forwarding rule is not binary and the node uses it as the probability of forwarding a message to an encountered node.

We evaluated the performance of the proposed algorithms on a synthetic preferentially attached network. Experimental results show that the probability distribution of the expected latencies that BMinLat-I and BMinLat-II estimate for nodes get tighter as the time passes. Moreover, the means of these probability distributions get close to the expected latencies of MinLat. When no external information is used in the prior distributions, BMinLat-I and BMinLat-II have almost the same results as MinLat-E. We also showed that as we increase the number of generated samples in these two algorithms, their results are more similar to MinLat-E. In order to show how the proposed Bayesian algorithms can be used in practice, we applied them on a dataset generated based on the Sigcomm09 dataset. We used some information about the social ties between nodes provided in the Sigcomm09 dataset to make the prior distributions more informative. We observed that using this information in the prior distribution can expedite the convergence of expected latencies compared to MinLat-E where no social information is used.

Chapter 5

Passive Approach: Inference of Diffusion Networks

5.1 Overview

In Chapters 3 and 4, we focused on how we can proactively control the process of diffusion of information/infection in a network structure. However, as mentioned in Chapter 1, sometimes we want to observe a diffusion process and learn about its dynamic and behavioural patterns rather than control it. These detected patterns may be later used to expedite, retard, or prevent future similar processes. The key factor that determines how the inference should be performed is the type of the changes we observe in the state of the nodes when they become infected.

In this chapter, we intend to passively observe diffusion processes to detect the underlying structure. The focus is on scenarios in which the moment of time that a node becomes infected is unknown. More precisely, our goal is to infer the network structure based on which the infection spreads throughout the network as well as the infection times and the strength of the links between nodes. We assume that our only observation of the diffusion process is a set of time series, one for each node of the network, which exhibit statistical changes when an infection occurs. We first formulate the problem in a Bayesian framework and specify the notations we use in Section 5.2. The network inference problem is modelled as an optimization problem. Since this optimization problem cannot be solved in a closed form, we use Markov Chain Monte Carlo (MCMC) techniques to find the set of network

parameters and temporal cascades that best explain our observations. In Section 5.3, we develop a batch inference algorithm for the scenarios in which the whole time series is observed and can be processed in its entirety. We then migrate to an online version of the inference algorithm which suits the scenarios in which the inference should be performed before all the observations are available. We use Sequential Monte Carlo (SMC) techniques to develop the online algorithm in Section 5.4. The performance of the proposed algorithms are evaluated on both synthetic and real-world datasets in Section 5.5. We summarize the chapter in Section 5.6 and provide the proofs of lemmas and theorems stated in this chapter in Section 5.7.

5.2 Problem Statement

In Chapter 1, we defined a *diffusion process* as the combination of three main components. In this section, we specify how we model each component.

- *Nodes*, i.e., the set of separate agents. We consider a set of N agents and denote this set by $\mathcal{N} = \{1, \ldots, N\}$.

- *Infection* or *Contagion*, i.e., the change in the state of a node that is triggered by an external source or by another node that has already changed its state. When a node changes its state, we say it has received the contagion (or infection). We call the nodes that have received the infection from the external source as the *source nodes* for the set $\mathcal{N}$. In this work, we consider just two states for each node, susceptible and infected, and refer to this model as the SI (Susceptible-Infected) infection scenario. In the susceptible state, a node is prone to receiving the contagion and changing its state to infected. Once an arbitrary node becomes infected, it never changes its state afterwards. However, it can cause other susceptible nodes to change their states (See the next component). Depending on the number and the order of states, different infection scenarios may be considered. We introduce other possible states and briefly discuss how our proposed model can be extended for other infection scenarios in Chapter 6. Considering a discrete time setup, we denote the moment of time at which node i changes its state by t^i and refer to it as node i's infection time. In our work, nodes' infection times are modelled as random variables. We describe the distribution function of these random variables after specifying the third component

of a diffusion process. The term *cascade* is often used to refer to the temporal traces left by a diffusion process i.e., the realization of the random vector $[t^1, \ldots, t^N]^T$.

– *Causality* or *Network Structure*, i.e., the underlying structure based on which nodes cause each other to change state. We model this component by a directed, weighted graph $G = (\mathcal{N}, \mathcal{E}, \boldsymbol{\alpha}_{N\times N})$ where $\mathcal{E}$ is the set of directed edges, and $\boldsymbol{\alpha} = [\alpha^{ij}]_{N\times N}$ is a link strength matrix. When node $j \in \mathcal{N}$ triggers node $i \in \mathcal{N}$ to change its state to infected, we say j has transferred the infection to i or i is infected by j. In this case, node j is referred to as the parent of node i and is denoted by z^i. Each node i has a set of potential (or candidate) parents π^i, but it is infected by only one of them, i.e., $z^i \in \pi^i$. A directed edge $j \to i$ exists in $\mathcal{E}$ if and only if $z^i = j$. The factor that determines which member of π^i transfers the infection to node i is the strength of the relationship between node i and each of its potential parents $j \in \pi^i$. We denote this link strength for nodes i and $j \in \pi^i$ by α^{ij}. The definitions of parents and candidate parents simply imply that $\forall j \in \pi^i : t^j < t^i$ and $\forall j \notin \pi^i : \alpha^{ij} = 0$.

Throughout this chapter, we use $[\cdot]$ to denote vectors and matrices, $\{\cdot\}$ to denote sets, and $(\cdot)$ to denote finite ordered lists.

Now that we have modelled the components of the diffusion process, we characterize the probability distributions that control parent selection and the infection times. Knowing the set of potential parents of node i and the link strengths, z^i is a discrete random variable whose probability mass function (*i.e.*, $f(z^i|\alpha^{ij}, j \in \pi^i)$) depends on the strength of the links between node i and all the nodes $j \in \pi^i$. Moreover, in our model, once z^i is infected, the probability that it transfers the contagion to i decays as time passes and its rate of decay depends on their link strength. Therefore, the probability distribution function of t^i (i.e., $f(t^i|z^i, t^{z^i}, \alpha^{iz^i})$) is a monotonically decreasing function of $t^i - t^{z^i}$. Hence, for each node i we model the joint probability distribution of t^i, z^i and $\alpha^{ij}, j \in \pi^i$ as,

$$f(t^i, z^i, \alpha^{ij}, j \in \pi^i | t^{z^i}) = f(t^i|z^i, t^{z^i}, \alpha^{iz^i}) f(z^i|\alpha^{ij}, j \in \pi^i) f(\alpha^{ij}, j \in \pi^i) \tag{5.1}$$

In Section 5.3.1, we specify the prior distributions we consider in our Bayesian inference algorithms for the terms of (5.1).

As mentioned in Section 5.1, we focus on the scenarios where neither the network structure (i.e., parental relations and link strengths) nor the cascades (i.e., infection times)

are directly observed. We assume that the only observation we get from an arbitrary node $i \in \mathcal{N}$ is a discrete time signal of length N_T denoted by $\mathbf{d}^i = (d_n^1, \ldots, d_n^{N_T})$. We denote the set of all observed time signals by $\mathbf{d} = \{\mathbf{d}^1, \ldots, \mathbf{d}^N\}$.

In order to develop inference algorithms for the case where data arrives in a streaming fashion or in batches, we consider a setting where each node's data arrives in N_B blocks of length M. Denote the vector of time indices in block b by $\mathbf{B}_b$, i.e.,

$$\mathbf{B}_b = \begin{bmatrix} M \times (b-1) + 1 \\ \vdots \\ M \times b \end{bmatrix}$$

and the data in block b for all the nodes in the network by $\mathbf{d}_{\mathbf{B}_b} = \{\mathbf{d}_{\mathbf{B}_b}^1, \ldots, \mathbf{d}_{\mathbf{B}_b}^N\}$. We denote the parent for node i at the end of the b^{th} block by z_b^i. If no infection has occurred by the end of block $b-1$, but it occurs during the b^{th} block, then z_{b-1}^i has a null-value, i.e., $z_{b-1}^i = \phi$, whereas $z_b^i = j$ for some $j \in \pi_i$. Likewise, the infection time for node i and the link strength associated with link ij in block b are respectively denoted by t_b^i and α_b^{ij}. If no infection has occurred then t_b^i also has a null-value ϕ. The parameters at the end of the b^{th} block are denoted by $\mathbf{x}_b = (\mathbf{z}_b, \mathbf{t}_b, \boldsymbol{\alpha}_b)$ where

$$\mathbf{z}_b = \begin{bmatrix} z_b^1 \\ \vdots \\ z_b^N \end{bmatrix} \quad , \quad \mathbf{t}_b = \begin{bmatrix} t_b^1 \\ \vdots \\ t_b^N \end{bmatrix} \quad , \quad \boldsymbol{\alpha}_b = [\alpha_b^{ij}]_{N \times N}$$

While the set of potential parents π^i is the same for all the blocks, two sets $\mathcal{C}_b^i$ and π_b^i are defined as follows for each block b.

$$\mathcal{C}_b^i = \{k | z_b^k = i\} \quad , \quad \pi_b^i = \{k | k \in \pi^i, t_b^k \neq \phi, t_b^k < t_b^i\}$$

$\mathcal{C}_b^i$ is the set of nodes that node i has infected before the end of the b^{th} block, i.e., node i is the established parent of all nodes $j \in \mathcal{C}_b^i$. $\pi_b^i \subseteq \pi^i$ is the subset of potential parents of node i who have been infected by the end of the b^{th} block. Table 5.1 lists the notation used in this chapter.

Symbol(s)	Expression(s)	Definition(s)
$\mathcal{N}$	$\{1,\ldots,N\}$	Set of all the N nodes in the network
π^i	-	Set of potential (or candidate) parents for node i
κ_{ij},θ_{ij}	-	Gamma distribution hyperparameters for link ij
N_T, M, N_B	$N_T = M \times N_B$	Length of observed data, Number of blocks, Length of each block
$\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2$	$\boldsymbol{\gamma}_j = (\gamma_j^1,\ldots,\gamma_j^N)$, $j=1,2$	Hyperparameters of observed data before and after being infected
$\mathbf{B}_b$	$[M(b-1)+1,\ldots,Mb]^T$	Vector of time indices in block b
$\mathbf{d}_{\mathbf{B}_b}$	$\{\mathbf{d}_{\mathbf{B}_b}^1,\ldots,\mathbf{d}_{\mathbf{B}_b}^N\}$	Set of data received in block b
$\mathbf{z}_b$	$[z_b^1,\ldots,z_b^N]^T$	Random vector of parents in block b
$\mathbf{t}_b$	$[t_b^1,\ldots,t_b^N]^T$	Random vector of infection times in block b
$\boldsymbol{\alpha}_b$	$[\alpha_b^{ij}]_{N\times N}$	Random matrix of link strengths in block b
$\mathbf{x}_b$	$(\mathbf{z}_b,\mathbf{t}_b,\boldsymbol{\alpha}_b)$	Infection parameters in block b
p_b^i	$1-e^{-\alpha_b^{iz_b^i}}$	Parameter of geometric distribution in block b
$\mathbf{x}_b^{MAP}$	$(\hat{\mathbf{z}}_b,\hat{\mathbf{t}}_b,\hat{\boldsymbol{\alpha}}_b)$	MAP estimates of infection parameters in block b
$\mathcal{C}_b^i$	$\{k\|z_b^k=i\}$	Set of node i's children in block b
π_b^i	$\{k\|k\in\pi^i, t_b^k\neq\phi, t_b^k<t_b^i\}$	Set of node i's potential parents in block b
$\mathbf{z}_b^{\bar{i}}$	$[z_b^1,\ldots,z_b^{i-1},z_b^{i+1},\ldots,z_b^N]^T$	Vector of parents except node i in block b
$\mathbf{t}_b^{\bar{i}}$	$[t_b^1,\ldots,t_b^{i-1},t_b^{i+1},\ldots,t_b^N]^T$	Vector of infection times except node i in block b
$\boldsymbol{\alpha}_b^{\overline{ij}}$	$[\alpha_b^{i1},\ldots,\alpha_b^{i(j-1)},\alpha_b^{i(j+1)},\ldots,\alpha_b^{iN}]^T$	Vector of node i's link strengths except for link ij in block b
$\mathbf{x}_b^m$	$(\mathbf{z}_b^m,\mathbf{t}_b^m,\boldsymbol{\alpha}_b^m)$	The m^{th} sample from the distribution $f(\mathbf{x}_b\|\mathbf{d}_{\mathbf{B}_{1:b}})$ in block b
r_b	-	Geometric parameter in the proposal for infection times in block b
$\mathbf{t}_b^{ML}$	$[t_b^{ML^1},\ldots,t_b^{ML^N}]^T$	Vector of maximum likelihood changepoints of $\mathbf{d}_{\mathbf{B}_b}$
N_{thin}, N_{burn}	-	Thinning and burn-in factors
N_{MCMC}, N_s	$N_{MCMC} = N_s \times N_{thin} + N_{burn}$	Number of generated and stored particles

Table 5.1 Summary of notations used in Chapter 5

The goal is to infer the set of infection parameters $\mathbf{x}_b$ at the end of the b^{th} block that best explains the received signals up to the end of block b, i.e., $\mathbf{d}_{\mathbf{B}_{1:b}}$. More precisely, we aim to find the most probable parameters $(\hat{\mathbf{z}}_b, \hat{\mathbf{t}}_b, \hat{\boldsymbol{\alpha}}_b)$ conditioned on the received signals $\mathbf{d}_{\mathbf{B}_{1:b}}$. We denote these Maximum A Posteriori (MAP) estimates of infection parameters by $\mathbf{x}_b^{MAP}$:

$$\mathbf{x}_b^{MAP} = (\hat{\mathbf{z}}_b, \hat{\mathbf{t}}_b, \hat{\boldsymbol{\alpha}}_b) = \underset{(\mathbf{z}_b, \mathbf{t}_b, \boldsymbol{\alpha}_b)}{\arg\max} \quad f(\mathbf{z}_b, \mathbf{t}_b, \boldsymbol{\alpha}_b | \mathbf{d}_{\mathbf{B}_{1:b}}) \tag{5.2}$$

Solving the optimization problem in (5.2) is challenging, especially considering the fact that the data arrives in blocks and we need to make decisions before we have access to the entire signals. Hence, we resort to Markov Chain Monte Carlo (MCMC) methods to generate samples from the posterior distribution, $f(\mathbf{z}_b, \mathbf{t}_b, \boldsymbol{\alpha}_b | \mathbf{d}_{\mathbf{B}_{1:b}})$, and assess the underlying diffusion process based on these samples. We first describe the batch inference approach based on Gibbs Sampling. We then extend this framework by considering the cases where no infection time is detected in the interval under study. We use this batch inference method to generate particles in the first received block of data. We then design an online inference algorithm based on SMC techniques to find the infection parameters (i.e., network structure and cascade information) that best explains the observed data at the end of each block $b > 1$. In order to do so, the particles for each block b are obtained by updating particles of block $b-1$ using the received signals $\mathbf{d}_{\mathbf{B}_b}$. To have a unified notation throughout this chapter, we use the b batch subscripts from now on for both the batch and sequential settings. For batch inference scenarios we have $\mathbf{x} = \mathbf{x}_1 = (\mathbf{z}_1, \mathbf{t}_1, \boldsymbol{\alpha}_1)$ since $N_B = 1$ and $M = N_T$.

5.3 Batch Inference

Assuming that the entire data signal is available at each node, we develop a batch (offline) inference algorithm based on Gibbs Sampling. Using Bayes' rule and due to the dependencies we clarified in Section 5.2, the joint conditional distribution $f(\mathbf{x}_1 | \mathbf{d}_{\mathbf{B}_1}) = f(\mathbf{z}_1, \mathbf{t}_1, \boldsymbol{\alpha}_1 | \mathbf{d}_{\mathbf{B}_1})$ is

$$f(\mathbf{x}_1 | \mathbf{d}_{\mathbf{B}_1}) = \frac{f(\mathbf{d}_{\mathbf{B}_1} | \mathbf{t}_1, \mathbf{z}_1, \boldsymbol{\alpha}_1) f(\mathbf{t}_1 | \mathbf{z}_1, \boldsymbol{\alpha}_1) f(\mathbf{z}_1 | \boldsymbol{\alpha}_1) f(\boldsymbol{\alpha}_1)}{f(\mathbf{d}_{\mathbf{B}_1})} \tag{5.3}$$

In the rest of this section, we introduce appropriate prior distributions for components of equation (5.3). The priors are selected to allow flexibility in the incorporation of prior knowledge and to facilitate computation. We then use this Bayesian framework to develop methods for inferring the underlying network structure as well as the cascade traces.

5.3.1 Priors

As opposed to the previous methods and models (e.g., [72, 87]), we accommodate scenarios where an arbitrary node i may never become infected over the study period of length N_T. We indicate this case by assigning null values to the infection time and parent of node i, i.e., $t_1^i = \phi$ and $z_1^i = \phi$. The conditional probability density function $f(t_1^i | z_1^i, \alpha_1^{iz_1^i}, t_1^{z_1^i})$, which models the conditional likelihood that node z_1^i (infected at time $t_1^{z_1^i}$) transfers the infection to node i at time $t_1^i > t_1^{z_1^i}$. Intuitively, the ability of an infected node to transfer the contagion to other nodes is expected to decay as time passes. Hence, the monotonic memoryless exponential distribution is a good candidate. An alternative way to motivate this model is to consider repeated interactions between the nodes, each being a Bernoulli trial with a small probability of infection.

Some authors (e.g., [72]) have studied the effect of heavy tailed Power law or non-monotonic Rayleigh distributions on the inference methods when the cascades are observed. For our prior distribution, we employ the exponential decay assumption. Since our observations $\mathbf{d}$ are assumed to be discrete time series, we choose the discrete counterpart of an exponential distribution for $f(t_1^i | z_1^i, \alpha_1^{iz_1^i}, t_1^{z_1^i})$, i.e., a geometric distribution with parameter $p_1^i = 1 - e^{-\alpha_1^{iz_1^i}}$.

Consider the case where $t_1^1 \geq t_1^2 \geq \cdots \geq t_1^N$, employing the convention that the null infection time $\phi > N_T$. We have $f(\mathbf{t}_1 | \mathbf{z}_1, \boldsymbol{\alpha}_1) = \prod_{i \in \mathcal{N}} f(t_1^i | z_1^i, \boldsymbol{\alpha}_1, t_1^{i+1:N})$ where

$$f(t_1^i | z_1^i, \boldsymbol{\alpha}_1, t_1^{i+1:N}) = \begin{cases} p_1^i (1 - p_1^i)^{t_1^i - t_1^{z_1^i} - 1} & 1 \leq t_1^i \leq N_T, \\ (1 - p_1^i)^{N_T} & t_1^i = \phi. \end{cases} \tag{5.4}$$

The expression for $t_1^i = \phi$ is simply one minus the value of the cumulative distribution function of the random variable t_1^i at time N_T. A similar expression can be obtained for any arbitrary ordering of the changepoints.

We assume that, conditioned on the link strengths, the probability that one of the

nodes in the candidate parent set $v \in \pi_i$ is the actual parent of node i is independent of the probability that a node $r \in \pi_j$ is the parent of node j. The exponential decay assumption and the fact that z_1^i is the first node that transfers the infection to node i makes the multinomial distribution a good candidate for capturing the prior distribution of parents given the link strength values.

$$f(\mathbf{z}_1|\boldsymbol{\alpha}_1) = \prod_{i\in\mathcal{N}} f(z_1^i|\alpha_1^{ij}, j \in \pi^i) = \prod_{i\in\mathcal{N}} \frac{{\alpha_1^{iz_1^i}}}{\sum_{j\in\pi^i} \alpha_1^{ij}} \tag{5.5}$$

As for the prior distribution for strength value of link ij, we choose a Gamma distribution, i.e., $\alpha_1^{ij} \sim \Gamma(\kappa_{ij}, \theta_{ij})$. This choice of prior distribution allows us to model a wide range of α values for different links of the network. We can capture both highly informed knowledge about strong links or the uninformed case where we have little prior information about the strength of links. Therefore, assuming that link strength values of different links are independent, we have

$$f(\boldsymbol{\alpha}_1) = \prod_{i\in\mathcal{N}, j\in\pi^i} f(\alpha_1^{ij}) = \prod_{i\in\mathcal{N}, j\in\pi^i} \frac{x^{\kappa_{ij}-1} e^{-\frac{x}{\theta_{ij}}}}{\Gamma(\kappa_{ij}) {\theta_{ij}}^{\kappa_{ij}}} \tag{5.6}$$

Finally, we assume that node i's observed data, $\mathbf{d}_{\mathbf{B}_1}^i$, are conditionally independent of the observations from all other nodes and that they follow the same prior distribution with two different sets of hyperparameters $\boldsymbol{\gamma}_1 = (\gamma_1^1, \ldots, \gamma_1^N)$, $\boldsymbol{\gamma}_2 = (\gamma_2^1, \ldots, \gamma_2^N)$ before and after being infected at t^i. Hence,

$$\begin{aligned} f(\mathbf{d}_{\mathbf{B}_1}|\mathbf{z}_1, \mathbf{t}_1, \boldsymbol{\alpha}_1) &= \prod_{i\in\mathcal{N}} f(\mathbf{d}_{\mathbf{B}_1}^i|t_1^i) \\ &= \prod_{i\in\mathcal{N}} \prod_{j=1}^{t^i} f(d_j^i; \gamma_1^i) \prod_{j=t^{i+1}}^{N_T} f(d_j^i; \gamma_2^i) \end{aligned} \tag{5.7}$$

The choice of priors $f(d_j^i; \gamma_1^i)$ and $f(d_j^i; \gamma_2^i)$ depends on the application. We test cases where the data are independently drawn from Gaussian and Poisson distributions in our numerical simulations in Section 5.5, but more complex structure can be readily incorporated in the time-series model (autoregressive processes, for example). In the next sections, we use the prior distributions described above to design batch and online inference methods in a

Bayesian framework.

5.3.2 Gibbs Sampling

With the proposed distributions in (5.4)-(5.7), we can calculate the probability of any arbitrary $(\mathbf{z}_1, \mathbf{t}_1, \boldsymbol{\alpha}_1)$ up to a constant $\frac{1}{f(\mathbf{d}_{\mathbf{B}_1})}$ using (5.3). We use Gibbs Sampling to generate N_s samples from the posterior distribution of (5.3). In other words, we use full conditional distributions for each of the infection parameters $z_1^i, t_1^i, \alpha_1^{ij}$ $(i, j \in \mathcal{N})$ to generate samples. We denote the parents and infection times of all the nodes in the network except node i respectively by $\mathbf{z}_1^{\bar{i}}$, $\mathbf{t}_1^{\bar{i}}$. Also, the vector of link strengths of all node i's links except the link between nodes i and j is denoted by $\boldsymbol{\alpha}_1^{\overline{ij}}$. The full conditional probabilities for Gibbs Sampling are as follows.

a) For the parent of node $i \in \mathcal{N}$

$$f(z_1^i|\mathbf{d}_{\mathbf{B}_1}, \mathbf{z}_1^{\bar{i}}, \mathbf{t}_1, \boldsymbol{\alpha}_1) \propto f(t_1^i|z_1^i, \alpha_1^{iz_1^i}, t_1^{z_1^i}) f(z_1^i|\alpha_1^{ij})_{j\in\pi^i} \tag{5.8}$$

b) For the infection time of node $i \in \mathcal{N}$

$$f(t_1^i|\mathbf{d}_{\mathbf{B}_1}, \mathbf{z}_1, \mathbf{t}_1^{\bar{i}}, \boldsymbol{\alpha}_1) \propto f(\mathbf{d}_{\mathbf{B}_1}^i|t_1^i) f(t_1^i|z_1^i, \alpha_1^{iz_1^i}, t_1^{z_1^i}) \prod_{k\in\mathcal{C}_1^i} f(t_1^k|\alpha_1^{ki}, t_1^i) \tag{5.9}$$

c) For the link strength between nodes $i \in \mathcal{N}$ and $j \in \pi^i$

$$f(\alpha_1^{ij}|\mathbf{d}_{\mathbf{B}_1}, \mathbf{z}_1, \mathbf{t}_1, \boldsymbol{\alpha}_1^{\overline{ij}}) \propto f(t_1^i|z_1^i, \alpha_1^{iz_1^i}, t_1^{z_1^i}) f(z_1^i|\boldsymbol{\alpha}_1) f(\alpha_1^{ij}) \tag{5.10}$$

Building upon this batch inference method, we propose an online inference method in the next section. In this online approach, the batch inference method is used to make decisions about diffusion parameters in the first received block of data.

5.4 Online Inference

In the online setting, the goal is to compute the filtering posterior distribution $f(\mathbf{x}_b|\mathbf{d}_{\mathbf{B}_{1:b}})$. In the Bayesian framework we have,

$$f(\mathbf{x}_b|\mathbf{d}_{\mathbf{B}_{1:b}}) \propto \int f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b) f(\mathbf{x}_b|\mathbf{x}_{b-1}) f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}}) d_{\mathbf{x}_{b-1}} \tag{5.11}$$

Since calculating (5.11) is analytically intractable in our application, we use the sequential Markov Chain Monte Carlo framework proposed in [118] to obtain an approximation of this filtering distribution. In this framework, instead of striving to directly sample from $f(\mathbf{x}_b|\mathbf{d}_{\mathbf{B}_{1:b}})$, which has complexity issues due to the need to perform the marginalization captured by the integral in (5.11), we sample from the joint posterior distribution $f(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b}})$ where

$$f(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b}}) = \frac{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b) f(\mathbf{x}_b|\mathbf{x}_{b-1}) f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}})}{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{d}_{\mathbf{B}_{1:b-1}})} \tag{5.12}$$

In the sequential MCMC approach, we maintain a set of particles that provides a sample-based approximation to the distribution of interest $f(\mathbf{x}_b|\mathbf{d}_{\mathbf{B}_{1:b}})$. After processing block $b-1$, since the posterior distribution $f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}})$ does not have a closed form representation, it is approximated with an empirical distribution based on the particle set $\{\mathbf{x}_{b-1}^j | j = 1, \ldots, N_s\}$, i.e.,

$$f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}}) = \frac{1}{N_s} \sum_{j=1}^{N_s} \delta(\mathbf{x}_{b-1} - \mathbf{x}_{b-1}^j) \tag{5.13}$$

Hence, the target distribution $f(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b}})$ that we wish to sample from can be approximated as

$$\begin{cases} \frac{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b) f(\mathbf{x}_b|\mathbf{x}_{b-1}^s)}{\sum_{j=1}^{N_s} f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_{b-1}^j)} & \text{if } \mathbf{x}_{b-1} \in \{\mathbf{x}_{b-1}^s | s = 1, \ldots, N_s\}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.14}$$

Here, $f(\mathbf{x}_b|\mathbf{x}_{b-1}^s)$ is referred to as the transition distribution. We will derive an appropriate expression for this probability distribution in Section 5.4.1.

Algorithm 6 presents our proposed online inference method based on the MCMC-based particle algorithm used in [118]. As mentioned earlier, we use the batch inference method described in section 5.3 to generate a set of N_s particles in the first block of the data and use it as the initial particle set for the SMC procedure of next blocks. In each block $b > 1$ of the data, the SMC procedure consists of two main steps. The first step, *joint draw*, is a joint Metropolis-Hastings (MH) proposal step in which instead of sampling from $f(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b}})$, we sample from the proposal distribution $q(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1})$ and accept the proposed samples with probability ρ. Lines 11-13 of Algorithm 6 describe

the components of the proposal distributions. The proposal distributions and the MH acceptance ratio are respectively derived in Sections 5.4.2 and 5.4.3. The second step of the SMC procedure, *refinement*, is an individual refinement Gibbs Sampling step where $\mathbf{x}_b$ is updated by sampling from $f(\mathbf{x}_b|\mathbf{x}_{b-1}^m, \mathbf{d}_{\mathbf{B}_b})$.

Algorithm 6 SMC-Based Online Inference Method

1: `// First block`
2: Generate N_s samples $\mathbf{x}_1^s|_{s=1,\dots,N_s} \sim f(\mathbf{x}_1|\mathbf{d}_{\mathbf{B}_1})$ using Gibbs Sampling
3: `// Next blocks`
4: Initialize the particle set $\mathcal{P} = \{\mathbf{x}_1^s|s = 1, \dots, N_s\}$
5: Set $N_{MCMC} = N_{thin} \times N_s + N_{burn}$
6: **for** $b = 2, \dots, N_b$ **do**
7: Calculate $\mathbf{t}_b^{ML}$ using (5.20)
8: **for** $m = 1, \dots, N_{MCMC}$ **do**
9: `// Joint Draw`
10: Propose $\mathbf{x}_{b-1}^*$ uniformly from $\mathcal{P}$
11: Propose $\mathbf{t}_b^* \sim h_{\mathbf{t}}(\mathbf{t}_b|\mathbf{x}_{b-1}^*; \mathbf{t}_b^{ML})$ using (5.23)
12: Propose $\boldsymbol{\alpha}_b^* \sim h_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}_b|\mathbf{x}_{b-1}^*, \mathbf{t}_b^*)$ using (5.25)
13: Propose $\mathbf{z}_b^* \sim h_{\mathbf{z}}(\mathbf{z}_b|\mathbf{x}_{b-1}^*, \mathbf{t}_b^*, \boldsymbol{\alpha}_b^*)$ using (5.26)
14: Set $\mathbf{x}_b^* = (\mathbf{z}_b^*, \mathbf{t}_b^*, \boldsymbol{\alpha}_b^*)$
15: Calculate MH acceptance probability ρ using (5.29)
16: Accept $(\mathbf{x}_b^m, \mathbf{x}_{b-1}^m) = (\mathbf{x}_b^*, \mathbf{x}_{b-1}^*)$ with probability ρ
17: `// Refinement`
18: Generate $\mathbf{x}_b^* \sim f(\mathbf{x}_b|\mathbf{x}_{b-1}^m, \mathbf{d}_{\mathbf{B}_b})$ using Gibbs Sampling and set $\mathbf{x}_b^m = \mathbf{x}_b^*$
19: **end for**
20: Set $\mathcal{P} = \{\mathbf{x}_b^{N_{burn}+kN_{thin}}|k = 1, \dots, N_s\}$
21: Set $\mathbf{x}_b^{MAP}$ as the most repeated particle in $\mathcal{P}$
22: **end for**

Finally, *thinning* and *burn-in* procedures are performed by storing one out of every N_{thin} accepted samples after discarding the initial N_{burn} samples. In the rest of this section, we derive the transition and proposal distributions and use them to calculate the MH acceptance ratio. We then explain the details of the refinement step and conclude the section by commenting on the computational complexity of Algorithm 6.

5.4.1 Transition Distribution

Using the framework explained in Section 5.2, we have

$$\begin{aligned} f(\mathbf{x}_b|\mathbf{x}^s_{b-1}) &= f(\mathbf{z}_b, \mathbf{t}_b|\mathbf{x}^s_{b-1}, \boldsymbol{\alpha}_b) f(\boldsymbol{\alpha}_b|\mathbf{x}^s_{b-1}) \\ &= \prod_{i\in\mathcal{N}} f(t^i_b, z^i_b|\mathbf{x}^s_{b-1}, \boldsymbol{\alpha}_b) \prod_{k\in\pi^i} f(\alpha^{ik}_b) \end{aligned} \tag{5.15}$$

Due to the characteristics of the online method, once a non-null value has been assigned to the infection parameters of a node in block b, this decision is respected in the following blocks. More precisely,

$$f(\alpha^{ik}_b) = \begin{cases} \delta(\alpha^{ik}_b - \alpha^{ik}_{b-1}) & \text{if } t^i_{b-1} \neq \phi, \\ \Gamma(\kappa_{ik}, \theta_{ik}) & \text{if } t^i_{b-1} = \phi, \end{cases} \tag{5.16}$$

and

$$f(t^i_b, z^i_b|\mathbf{x}^s_{b-1}, \alpha^{ij}_b)_{j\in\pi^i_b} = \begin{cases} \delta(t^i_b - t^i_{b-1})\delta(z^i_b - z^i_{b-1}) & \text{if } t^i_{b-1} \neq \phi, \\ \sum_{l\in\pi^i_b} \sum_{x=t^l_b+1}^{\mathbf{B}_b[M]} g_1(l,x,i) + \delta(z^i_b - \phi)\times & \\ \delta(t^i_b - \phi)(1 - \sum_{l\in\pi^i_b} \sum_{x=t^l_b+1}^{\mathbf{B}_b[M]} g_1(l,x,i)) & \text{if } t^i_{b-1} = \phi, \end{cases} \tag{5.17}$$

where

$$g_1(l,x,i) = \frac{\alpha^{il}_b}{\sum_{k\in\pi^i} \alpha^{ik}_b} (p^i_b)(1-p^i_b)^{x-t^l_b-1} \delta(z^i_b - l)\delta(t^i_b - x) \tag{5.18}$$

and $p^i_b = 1 - e^{-\alpha^{iz^i_b}_b}$. According to (5.17), if node i has not become infected by the end of block $b-1$, the probability that it becomes infected by node $l \in \pi^i_b$ at some time $x \in [t^l_b, \mathbf{B}_b[M]]$ is $g_1(l,x,i)$. Equation (5.18) shows that this probability equals the product of the geometric and multinomial distributions respectively defined in (5.4) and (5.5). Node

i remains susceptible (i.e., $t_b^i = \phi$) with a probability that equals one minus the sum of the probabilities of being infected by each node $l \in \pi_i$ at each time step $x \in [t_b^l, \mathbf{B}_b[M]]$.

5.4.2 Proposal Distribution

As presented in line 9 of Algorithm 6, the optimal importance sampling distribution (assuming that we adopt the particle-based approximation for $f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}})$) is:

$$\begin{aligned} q(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1}) &= f(\mathbf{x}_b|\mathbf{x}_{b-1}, \mathbf{d}_{\mathbf{B}_b}) f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}}) \\ &= \begin{cases} \frac{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b) f(\mathbf{x}_b|\mathbf{x}_{b-1})}{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_{b-1})} & \text{if } \mathbf{x}_{b-1} \in \{\mathbf{x}_{b-1}^s, s = 1, \dots, N_s\}, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{5.19}$$

Although calculating the predictive density in the denominator of (5.19) is not necessary for sampling, it is eventually required for calculating the acceptance ratio. To avoid numerical integration of the predictive density at every iteration, we benefit from an auxiliary parameter which can be obtained from the data. In each block b, we calculate the maximum likelihood changepoint of each individual time series $\mathbf{d}_{\mathbf{B}_b[1]:\mathbf{B}_b[M]}$ for all $i \in \mathcal{N}$. We denote the vector of these maximum likelihood changepoints by $\mathbf{t}_b^{ML} = [t_b^{ML^1}, \dots, t_b^{ML^N}]^T$, where

$$t_b^{ML^i} \triangleq \arg \max_{\mathbf{B}_b[1] \le t \le \mathbf{B}_b[M]} f(\mathbf{d}_{\mathbf{B}_b[1]:t}; \gamma_1^i) f(\mathbf{d}_{t+1:\mathbf{B}_b[M]}; \gamma_2^i) \tag{5.20}$$

Using this auxiliary parameter, we design the following proposal distribution.

$$\begin{aligned} q(\mathbf{x}_b, \mathbf{x}_{b-1}|\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1}) &= h(\mathbf{x}_b|\mathbf{x}_{b-1}; \mathbf{t}_b^{ML}) f(\mathbf{x}_{b-1}|\mathbf{d}_{\mathbf{B}_{1:b-1}}) \\ &= \begin{cases} \frac{1}{N_s} h(\mathbf{x}_b|\mathbf{x}_{b-1}; \mathbf{t}_b^{ML}) & \text{if } \mathbf{x}_{b-1} \in \{\mathbf{x}_{b-1}^s, s = 1, \dots, N_s\}, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{5.21}$$

where

$$h(\mathbf{x}_b|\mathbf{x}_{b-1}; \mathbf{t}_b^{ML}) = h_{\mathbf{t}}(\mathbf{t}_b|\mathbf{x}_{b-1}; \mathbf{t}_b^{ML}) h_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}_b|\mathbf{t}_b, \mathbf{x}_{b-1}) h_{\mathbf{z}}(\mathbf{z}_b|\mathbf{t}_b, \boldsymbol{\alpha}_b, \mathbf{x}_{b-1}). \tag{5.22}$$

Each component of $h(\mathbf{x}_b|\mathbf{x}_{b-1}; \mathbf{t}_b^{ML})$ is the proposal distribution for one of the infection parameters. For each node $i \in \mathcal{N}$, we independently propose an infection time whose distance from the maximum likelihood $t_b^{ML^i}$ follows a geometric distribution with pre-defined

parameter r_b^i. In other words, we sample from $h_{\mathbf{t}}(\mathbf{t}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML}) = \prod_{i\in\mathcal{N}} h_t^i(t_b^i|\mathbf{x}_{b-1};t_b^{ML^i})$, where

$$h_t^i(t_b^i|\mathbf{x}_{b-1};t_b^{ML^i}) = \begin{cases} \delta(t_b^i - t_{b-1}^i) & t_{b-1}^i \neq \phi, \\ \sum_{n=1}^M g_2(n,i) + (1-\sum_{n=1}^M g_2(n,i))\delta(t_b^i-\phi) & t_{b-1}^i = \phi. \end{cases} \tag{5.23}$$

and

$$g_2(n,i) = \frac{1}{2} r_b^i (1-r_b^i)^{|\mathbf{B}_b[n]-t_b^{ML^i}|}\delta(t_b^i - \mathbf{B}_b[n]). \tag{5.24}$$

Using the proposed $\mathbf{t}_b^*$, we propose $\boldsymbol{\alpha}_b^*$ by generating samples from $h_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}_b|\mathbf{t}_b^*,\mathbf{x}_{b-1}) = \prod_{i\in\mathcal{N}}\prod_{k\in\pi^i} h_\alpha^i(\alpha_b^{ik}|\mathbf{t}_b^*,\mathbf{x}_{b-1})$ where

$$h_\alpha^i(\alpha_b^{ik}|\mathbf{t}_b^*,\mathbf{x}_{b-1}) = \begin{cases} \delta(\alpha_b^{ik} - \alpha_{b-1}^{ik}) & \text{if } t_b^i \neq \phi, t_{b-1}^i \neq \phi, \\ \Gamma(\kappa_{ki},\theta_{ki}) & \text{if } t_{b-1}^i = \phi. \end{cases} \tag{5.25}$$

Finally, we propose $\mathbf{z}_b^*$ by sampling from $h_{\mathbf{z}}(\mathbf{z}_b|\mathbf{t}_b^*,\boldsymbol{\alpha}_b^*,\mathbf{x}_{b-1}) = \prod_{i\in\mathcal{N}} h_z^i(z_b^i|\mathbf{t}_b^*,\boldsymbol{\alpha}_b^*,\mathbf{x}_{b-1})$, i.e.,

$$h_z^i(z_b^i|\mathbf{t}_b^*,\boldsymbol{\alpha}_b^*,\mathbf{x}_{b-1}) = \begin{cases} \delta(z_b^i - z_{b-1}^i) & \text{if } t_b^i \neq \phi, t_{b-1}^i \neq \phi, \\ \delta(z_b^i - \phi) & \text{if } t_b^i = \phi, t_{b-1}^i = \phi, \\ \sum_{l\in\pi_b^i} \frac{\alpha_b^{il}}{\sum_{j\in\pi^i}\alpha_b^{ij}}\delta(z_b^i - l) + & \\ \quad (1 - \frac{\sum_{l\in\pi_b^i}\alpha_b^{il}}{\sum_{j\in\pi^i}\alpha_b^{ij}})\delta(z_b^i-\phi) & \text{if } t_b^i \neq \phi, t_{b-1}^i = \phi. \end{cases} \tag{5.26}$$

Proposition 2 shows that the proposal q is normalized. The proof is provided in Appendix 5.7.1.

Proposition 2. *$h(\mathbf{x}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML})$ is a probability distribution function. In particular,*

$$\int \sum_{\mathbf{t}_b\in\mathcal{S}_1\times\cdots\times\mathcal{S}_1} \sum_{\mathbf{z}_b\in\mathcal{S}_2\times\cdots\times\mathcal{S}_2} h(\mathbf{x}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML}) d_{\boldsymbol{\alpha}_b} = 1 \tag{5.27}$$

where $\mathcal{S}_1 = \{\mathbf{B}_1[1],\ldots,\mathbf{B}_b[M],\phi\}$ and $\mathcal{S}_2 = \pi_b^i \cup \{\phi\}$.

5.4.3 Metropolis-Hastings Acceptance Ratio

The acceptance ratio of the Metropolis-Hastings sampling approach proposed in [119] is

$$\rho = \min\left(1, \frac{f(\mathbf{x}_b^*, \mathbf{x}_{b-1}^*|\mathbf{d}_{\mathbf{B}_{1:b}})}{q(\mathbf{x}_b^*, \mathbf{x}_{b-1}^*|\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1})} \frac{q(\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1}|\mathbf{x}_b^*, \mathbf{x}_{b-1}^*)}{f(\mathbf{x}_b^{m-1}, \mathbf{x}_{b-1}^{m-1}|\mathbf{d}_{\mathbf{B}_{1:b}})}\right) \tag{5.28}$$

Replacing (5.14) and (5.21) in the second argument of (5.28), we have

$$\frac{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b^*)}{f(\mathbf{d}_{\mathbf{B}_b}|\mathbf{x}_b^{m-1})} \frac{f(\mathbf{x}_b^*|\mathbf{x}_{b-1}^*)}{f(\mathbf{x}_b^{m-1}|\mathbf{x}_{b-1}^{m-1})} \frac{h(\mathbf{x}_b^{m-1}|\mathbf{x}_{b-1}^{m-1}; \mathbf{t}_b^{ML})}{h(\mathbf{x}_b^*|\mathbf{x}_{b-1}^*; \mathbf{t}_b^{ML})} \tag{5.29}$$

In evaluating this expression, the first ratio can be calculated directly from the likelihood expressions. The second ratio can be evaluated from (5.15), using (5.16)-(5.18). The third ratio can be calculated from (5.22), using (5.23)-(5.26).

5.4.4 Refinement Step

The Gibbs Sampling procedure in the refinement step is basically the same as the Gibbs Sampling used in the batch inference. For each node $i \in \mathcal{N}$, each one of the diffusion parameters $(t_b^i, z_b^i, \alpha_b^i)$ is treated as a separate parameter block and is sampled using full conditional distributions. However, some of the online inference features need to be accounted for while deriving the full conditional distributions. These distributions are derived as follows.

a) For parent of node $i \in \mathcal{N}$

$$f(z_b^i|\mathbf{x}_{b-1}^m, \mathbf{d}_{\mathbf{B}_b}, \mathbf{z}_b^{\bar{i}}, \mathbf{t}_b, \boldsymbol{\alpha}_b) \propto \begin{cases} \delta(z_b^i - \phi) & \text{if } t^{i^m}_{b-1} = \phi, t_b^i = \phi\,, \\ \delta(z_b^i - z^{i^m}_{b-1}) & \text{if } t^{i^m}_{b-1} \neq \phi, t_b^i \neq \phi\,, \\ f(t_b^i|z_b^i, \alpha_b^{iz_b^i}, t_b^{z_b^i}) f(z_b^i|\alpha_b^{ik})_{k\in\pi^i} & \text{if } t^{i^m}_{b-1} = \phi, t_b^i \neq \phi\,. \end{cases} \tag{5.30}$$

b) For infection time of node $i \in \mathcal{N}$

$$f(t_b^i|\mathbf{x}_{b-1}^m, \mathbf{d}_{\mathbf{B}_b}, \mathbf{z}_b, \mathbf{t}_b^{\bar{i}}, \boldsymbol{\alpha}_b) \propto \begin{cases} \delta(t_b^i - t^{i^m}_{b-1}) & \text{if } t^{i^m}_{b-1} \neq \phi\,, \\ f(\mathbf{d}^i_{\mathbf{B}_b}|t_b^i) f(t_b^i|z_b^i, \alpha_b^{iz_b^i}, t_b^{z_b^i}) \times \prod_{k\in\mathcal{C}_b^i} f(t_b^k|\alpha_b^{ki}, t_b^i) & \text{if } t^{i^m}_{b-1} = \phi\,. \end{cases} \tag{5.31}$$

c) For link strength between nodes $i \in \mathcal{N}$ and $j \in \pi^i$

$$f(\alpha_b^{ij}|\mathbf{x}_{b-1}^m, \mathbf{d}_{\mathbf{B}_b}, \mathbf{z}_b, \mathbf{t}_b, \boldsymbol{\alpha}_b^{\overline{ij}}) \propto \begin{cases} f(\alpha_b^{ij}) & \text{if } {t^i}_{b-1}^m = \phi, t_b^i = \phi\,, \\ \delta(\alpha_b^{ij} - {\alpha^{ij}}_{b-1}^m) & \text{if } {t^i}_{b-1}^m \neq \phi, t_b^i \neq \phi\,, \\ f(t_b^i|z_b^i, \alpha_b^{iz_b^i}, t_b^{z_b^i}) f(z_b^i|\boldsymbol{\alpha}_b) f(\alpha_b^{ij}) & \text{if } {t^i}_{b-1}^m = \phi, t_b^i \neq \phi\,. \end{cases} \tag{5.32}$$

Here ${t^i}_{b-1}^m$, ${z^i}_{b-1}^m$, and ${\alpha^{ij}}_{b-1}^m$ respectively denote the infection time of node i, the parent of node i, and the link strength between nodes i and j in the sample $\mathbf{x}_{b-1}^m$.

5.4.5 Computational Complexity

Line 7 of Algorithm 6 only needs to be performed once per block. In each block b, we need to evaluate the probability distributions of (5.20) for each node at each time $\mathbf{B}_b[1] \leq t \leq \mathbf{B}_b[M]$. Therefore, the computational complexity of finding $\mathbf{t}_b^{ML}$ for all blocks is of order $O(N_b NM)$. Lines 11 to 13 of the algorithm involve generating respectively $N-1$ geometric, $\sum_{i\in\mathcal{N}} |\pi_i|$ gamma, and $N-1$ multinomial random variables in the worst case. These random number generation procedures also exist in the refinement step (Line 18) and the Gibbs Sampling of the first block (Line 2). In practice, we observe that generating a gamma distributed random variable is significantly more computationally expensive than generating the other two random variables. Therefore, if we bound $\sum_{i\in\mathcal{N}} |\pi_i|$ by N^2, we observe that, depending on the values of M, N, and N_{MCMC}, Algorithm 6 has computational complexity of $O(N_b N_{MCMC} N^2)$ or $O(N_b NM)$. The computation grows linearly with the number of MCMC iterations and is proportional to the square of the number of nodes in the network.

5.5 Simulation Results

In this section, we investigate the efficiency of our proposed approach in modelling and solving the inference problem in different diffusion network scenarios. We first test our method in synthetic networks where the modelling assumptions are exactly met and we know the ground truth. Then, we use the approach to analyze two real-world datasets.

5.5.1 Synthetic Data

Since we wish to evaluate the performance of the algorithms in scenarios where the modelling assumptions are exactly met, we generate a dataset based on the model described in Section 5.3. We assume that node 1 is the source of the infection. For each node $i > 1$, we randomly construct π^i by including each $j \in \{1, \dots, i-1\}$ as a potential parent for i (i.e., $j \in \pi^i$) with probability 0.5. Then, we uniformly choose $j \in \pi^i$ for each node i and build a random directed tree $\mathcal{T}$ with edges $j \to i$. We assign the link strength matrix $\boldsymbol{\alpha}^R = [\alpha_{ij}^R]_{N\times N}$ to this model and call them the *true link strengths*. $\alpha_{ij}^R \neq 0$ if and only if $j \in \pi^i$. In this case, α_{ij}^R is drawn from a gamma distribution $\Gamma(\kappa_1, \theta_1)$ if $j \to i \in \mathcal{T}$ and from $\Gamma(\kappa_2, \theta_2)$ if $j \to i \notin \mathcal{T}$. We choose the parent of node i from all the nodes $j \in \pi^i$ based on a random sampling with weights α_{ij}^R. These parents are called *true parents* and are denoted by $\mathbf{z}^R = [z_1^R, \dots, z_N^R]^T$. Knowing the values of z^i and α^{iz^i}, we then generate the *true infection times* $\mathbf{t}^R = [t_1^R, \dots, t_N^R]^T$ based on the geometric distributions described in (5.4). After choosing the random set of true diffusion parameters $(\mathbf{z}^R, \mathbf{t}^R, \boldsymbol{\alpha}^R)$, we choose the length of time series N_T to be 10 samples more than the maximum infection time (i.e., $N_T = 10 + \max_i t_i^R$). Finally, we generate data signals $\mathbf{d} = \{\mathbf{d}^1, \dots, \mathbf{d}^N\}$ of length N_T based on two different Gaussian distributions for before and after being infected. Hyperparameters $\gamma_1^i = (\mu_1, \sigma_1)$ and $\gamma_2^i = (\mu_2, \sigma_2)$ are used for all nodes $i \in \mathcal{N}$.

We aim to investigate whether incorporating the infection diffusion model can improve the estimation of infection times compared to univariate changepoint estimation. Hence, we compare two estimates of infection times: the MAP estimate $\hat{\mathbf{t}}$ based on the model and inference techniques described in this chapter; and an estimate $\hat{\mathbf{t}}' = [\hat{t'}^1, \dots, \hat{t'}^N]^T$ derived by maximizing the likelihood of infection time for individual time-series, i.e,

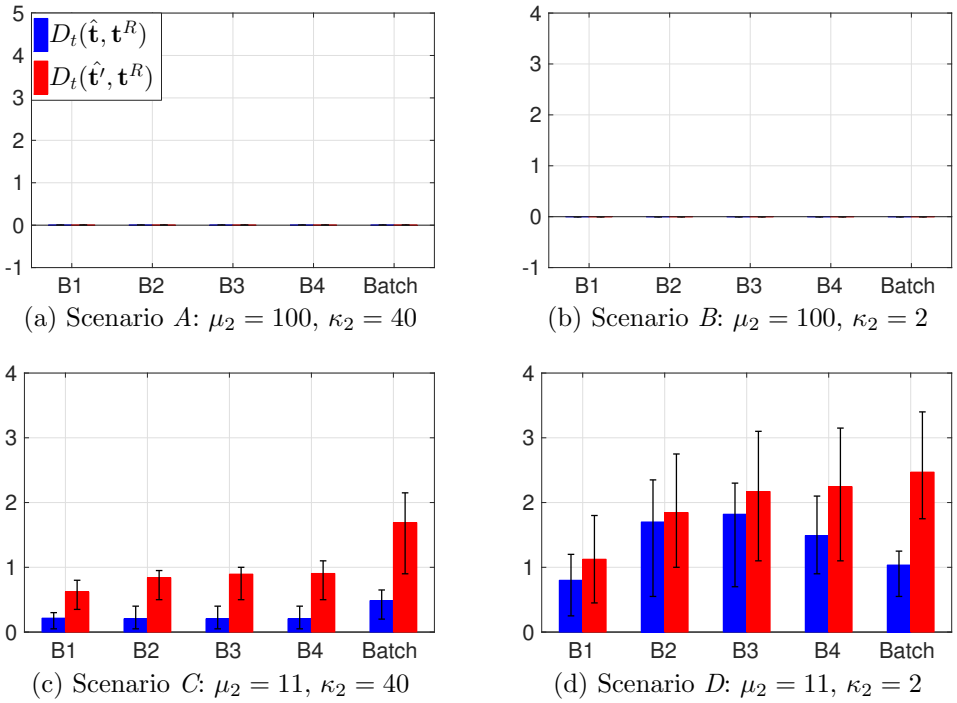
$$t'^i = \arg\max_t \prod_{j=1}^{t} f(d_j^i; \gamma_1^i) \prod_{j=t+1}^{N_T} f(d_j^i; \gamma_2^i) \tag{5.33}$$

We examine batch and online Bayesian inference algorithms in a network of $N = 20$ nodes. For the online setup, data signals are divided into four blocks of equal length ($\mathbf{B}_1$ to $\mathbf{B}_4$), i.e., $M = \lfloor \frac{N_T}{4} \rfloor$. We test inference algorithms in four scenarios. In all of the scenarios, $\mu_1 = 10$, $\sigma_1 = \sigma_2 = 1$, $\kappa_1 = 1$, $\theta_1 = \theta_2 = 0.5$, and $r_b = 0.5$. In the first scenario (Scenario A), $\mu_2 = 100$ and $\kappa_2 = 40$. Hence, not only can the infection times be detected with high likelihood, but also the links that demonstrate the parental relationships can be easily

distinguished from the rest of the links due to their high average strengths. In Scenario B, $\mu_2 = 100$ and $\kappa_2 = 2$. Therefore, the difference between parent and non-parent links are not significant and the underlying network is not easily detected even though infection times are still recognizable with high likelihood. The same trend exists in Scenarios C and D except for the fact that in these two scenarios the means of two Gaussian distributions are close i.e. in Scenario C, $\mu_2 = 11$, $\kappa_2 = 40$ and in Scenario D, $\mu_2 = 11$, $\kappa_2 = 2$.

Figure 5.1 shows the mean and 95% confidence intervals of deviations of the two estimates $\hat{\mathbf{t}}$ and $\hat{\mathbf{t}}'$ from the true infection times $\mathbf{t}^R$ for 100 realizations of true diffusion parameters. The deviation between two infection time vectors (e.g., $D_t(\hat{\mathbf{t}}, \mathbf{t}^R)$) is defined as the average absolute difference between infection times across all nodes. When the true infection time is greater than the time index of the end of a batch b (i.e., $t_i^R > \mathbf{B}_b[Mb]$), we set the true infection time to null for that batch. If either the estimated infection time of a node or its true value is null for a batch b, we replace the null value with the end of the batch when calculating the deviation metric. For each of the algorithms, $N_{MCMC} = 10^5$ samples are generated, the first $N_{burn} = 10^3$ samples are discarded, and every $N_{thin} = 10^{\text{th}}$ of the rest of the samples are kept. The i^{th} component of $\hat{\mathbf{t}}$ denotes the most observed infection time for node i among the stored samples while the i^{th} component of $\hat{\mathbf{t}}'$ shows the maximum likelihood estimation of infection time of node i while ignoring other nodes and the underlying network. As expected, the infection times are perfectly detected in the first two scenarios even though in Scenario B, the link strengths don't have significant difference between parent and non-parent links. As opposed to Scenarios A and B, deviations of detected infection times from their true values are non-zero in Scenario C and they are even larger in Scenario D where neither infection times nor parental relationships are easy to detect. However, we see that for both batch and online inference methods, exploiting the underlying diffusion network in detecting the infection times leads to more accurate results on average.

Next, we wish to investigate how not knowing the infection times affects the performance of detecting the parental relationships and their strengths. In order to do this, we compare our setup in which all the infection parameters are unknown with a setup in which infection times are perfectly known and $\mathbf{z}$ and $\boldsymbol{\alpha}$ are the only parameters to be detected. The MAP estimates for the case of unknown infection times are denoted by $\hat{\mathbf{z}}$ and $\hat{\boldsymbol{\alpha}}$. We use the same Gibbs sampling method to perform inference for the case when the infection times are known (except there is no longer a need to sample infection times). The MAP estimates



(a) Scenario A: $\mu_2 = 100$, $\kappa_2 = 40$

(b) Scenario B: $\mu_2 = 100$, $\kappa_2 = 2$

(c) Scenario C: $\mu_2 = 11$, $\kappa_2 = 40$

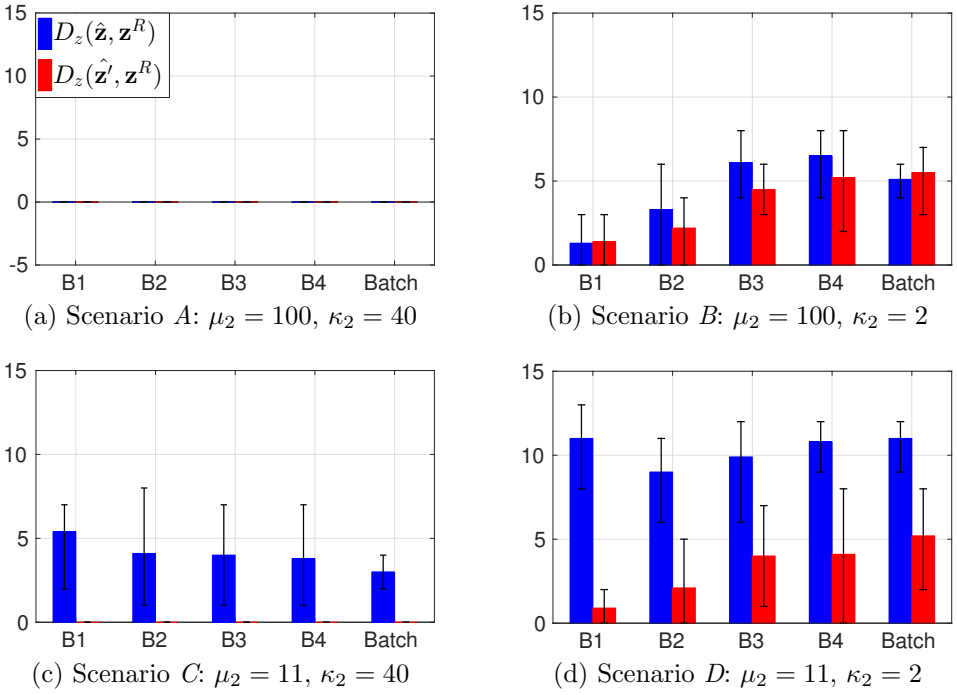
(d) Scenario D: $\mu_2 = 11$, $\kappa_2 = 2$

Figure 5.1 Deviations in detection of infection times

for this latter case are denoted by $\hat{\mathbf{z}}'$ and $\hat{\boldsymbol{\alpha}}'$. We calculate the error metric for parent identification, $D_z(\hat{\mathbf{z}}, \mathbf{z}^R)$, as the average number of nodes whose parents are different in $\hat{\mathbf{z}}$ and $\mathbf{z}^R$. Similarly, $D_\alpha(\hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha}^R)$ denotes the deviation between detected and true link strength values and is equal to the average over all links of the absolute difference between estimated and true α_{ij}.

Figures 5.2 and 5.3 show the error metrics for the estimation of parents and link strengths for the batch and online algorithms. In Scenarios A and B, when infection times are easy to infer, the knowledge of infection times provides little benefits and the error metrics are approximately the same. However, when the infection times are hard to detect (Scenarios C and D), the deterioration in the estimation accuracy is more observable. Figure 5.4 shows the average percentage of samples identifying the correct parents in Scenarios C and D.

In the rest of this section, we apply our inference approach on three different real-world datasets. How we model each dataset depends on the nature of its data and our purpose of studying it. The first two datasets studied in Sections 5.5.2 and 5.5.3 contain



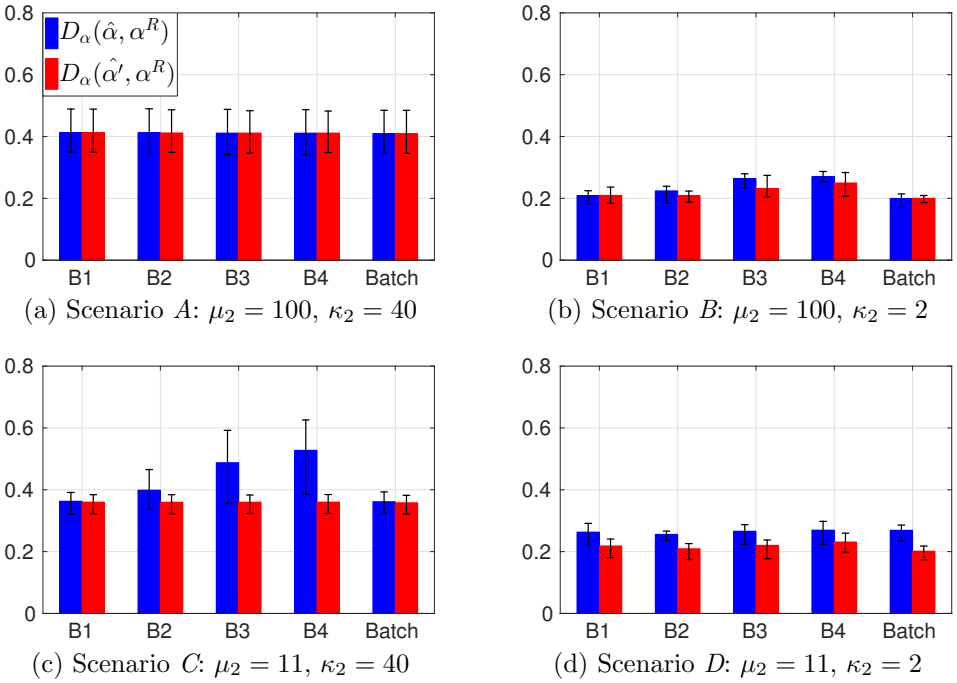
(a) Scenario A: $\mu_2 = 100$, $\kappa_2 = 40$

(b) Scenario B: $\mu_2 = 100$, $\kappa_2 = 2$

(c) Scenario C: $\mu_2 = 11$, $\kappa_2 = 40$

(d) Scenario D: $\mu_2 = 11$, $\kappa_2 = 2$

Figure 5.2 Deviations in detection of parents



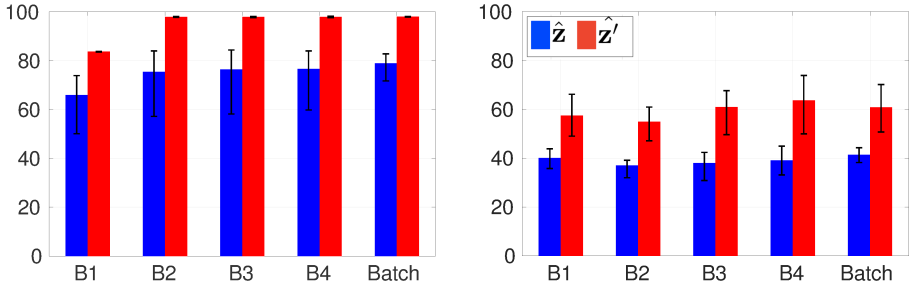
(a) Scenario A: $\mu_2 = 100$, $\kappa_2 = 40$

(b) Scenario B: $\mu_2 = 100$, $\kappa_2 = 2$

(c) Scenario C: $\mu_2 = 11$, $\kappa_2 = 40$

(d) Scenario D: $\mu_2 = 11$, $\kappa_2 = 2$

Figure 5.3 Deviations in detection of link strengths



(a) C: $\mu_2 = 11$, $\kappa_2 = 40$ (b) D: $\mu_2 = 11$, $\kappa_2 = 2$

Figure 5.4 Average percentage of samples identifying the correct parents in Scenarios C and D

epidemiological data. In these two datasets, we define network nodes as the geographical regions in which individuals have been exposed to a contagious disease. In the example of Section 5.5.2, the individuals are birds exposed to avian flu whereas in Section 5.5.3 the individuals are human beings. Our model for these two datasets assumes that an infection commences in one region and then by movement of individuals between two regions emerges in one of the other regions. The model does allow for the possibility of infections arising spontaneously due to unrelated influences in multiple regions. In such a case, multiple nodes (regions) have no parent in the inferred model. This can capture scenarios in which infections are caused by interactions with other regions not included in the model. It can also explain the case when the residual infection in a city gives rise to a new outbreak, with no disease transfer from another region. In Section 5.5.2, our goal is to infer the underlying network based on which the disease is transferred between different regions. In Section 5.5.3, however, we intend to estimate and predict the infection times in different regions using the recursive outbreak pattern of the data. The third dataset, studied in Section 5.5.4, is a seismographic dataset. The network nodes are defined as the seismograph stations that record the vibrations. We use this dataset to find the source of the infection which the epicenter of an earthquake or a tsunami.

5.5.2 Avian Influenza Data

We study the outbreak of Avian Influenza (H5N1 HPAI) [120]. Figure 5.5 shows the observed locations of reported infections for both domestic and wild bird species for the period of January 2004 to February 2016. We divide the observation points into eight main

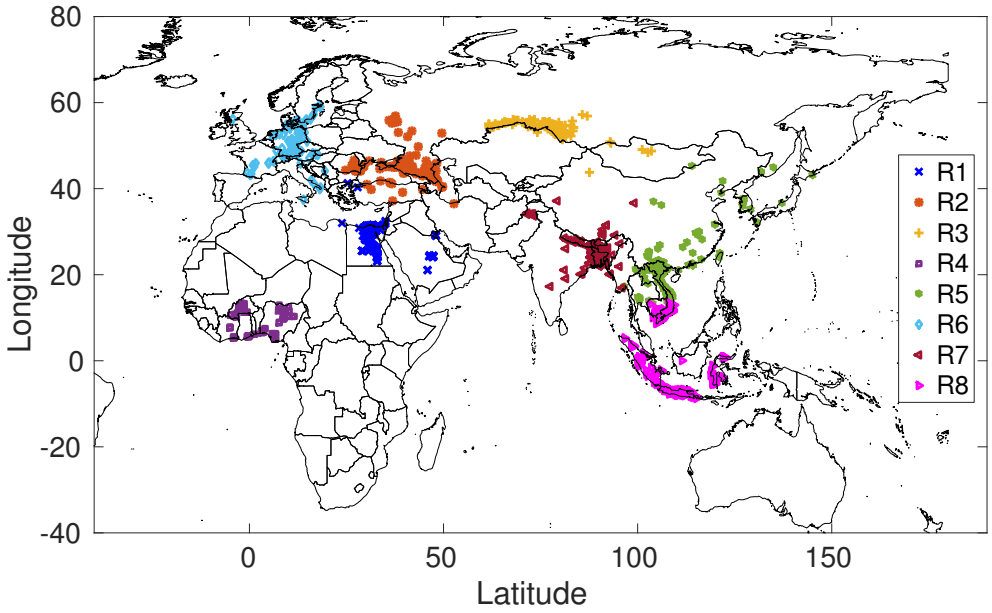


Figure 5.5 H5N1 HPAI outbreak in 2004-2016

regions using K-means clustering and generate a time series $\mathbf{d}^i = \{d_n^i\}_{n=1:N_T}$ to the i^{th} region. The value of this time series at day n (i.e., d_n^i) is the number of separate locations within the region i in which the disease was reported on that day. We model the number of observations in each region by a Poisson distribution. Hence, we have

$$f(\mathbf{d}^i|t^i) = \prod_{n=1}^{t^i-1} \frac{{\lambda_{1i}}^{d_n^i} e^{-\lambda_{1i}}}{d_n^i!} \prod_{n=t^i}^{N_T} \frac{{\lambda_{2i}}^{d_n^i} e^{-\lambda_{2i}}}{d_n^i!} \tag{5.34}$$

where λ_{1i} and λ_{2i} are the average number of events before and after becoming infected. Since the individuals who are exposed to disease are birds, it makes sense to relate the link strengths to the geographic distances between nodes. The link strength parameters κ_{ij} and θ_{ij} of equation (5.6) are derived by fitting a gamma distribution to the inverse of distances between observation points of regions i and j. Figure 5.6 shows the time series for the eight regions. Regions R5 and R8 are the first regions in which the disease is observed. The first infections for these regions were reported on the same day, so we assume that they were both sources of the infection. We infer the infection parameters for the period 2004-2007 by generating $M = 10^6$ samples and discarding the first 10^4 ones. The green line in Figure 5.6 shows the end of the study period. Region R4 has almost no reported

infections for this period so we exclude it when estimating the underlying infection graph. The detected infection times are shown in Figure 5.6 by red vertical lines. Figure 5.7 shows

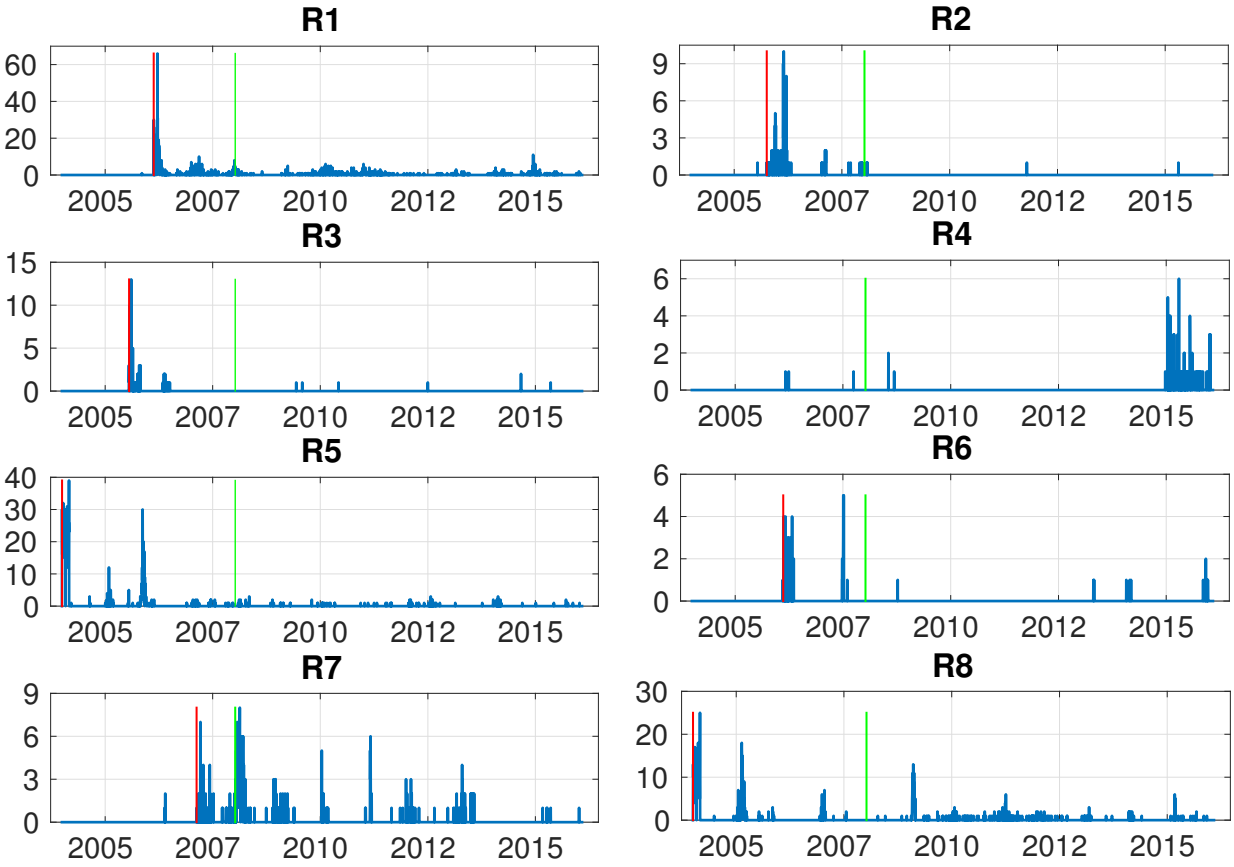
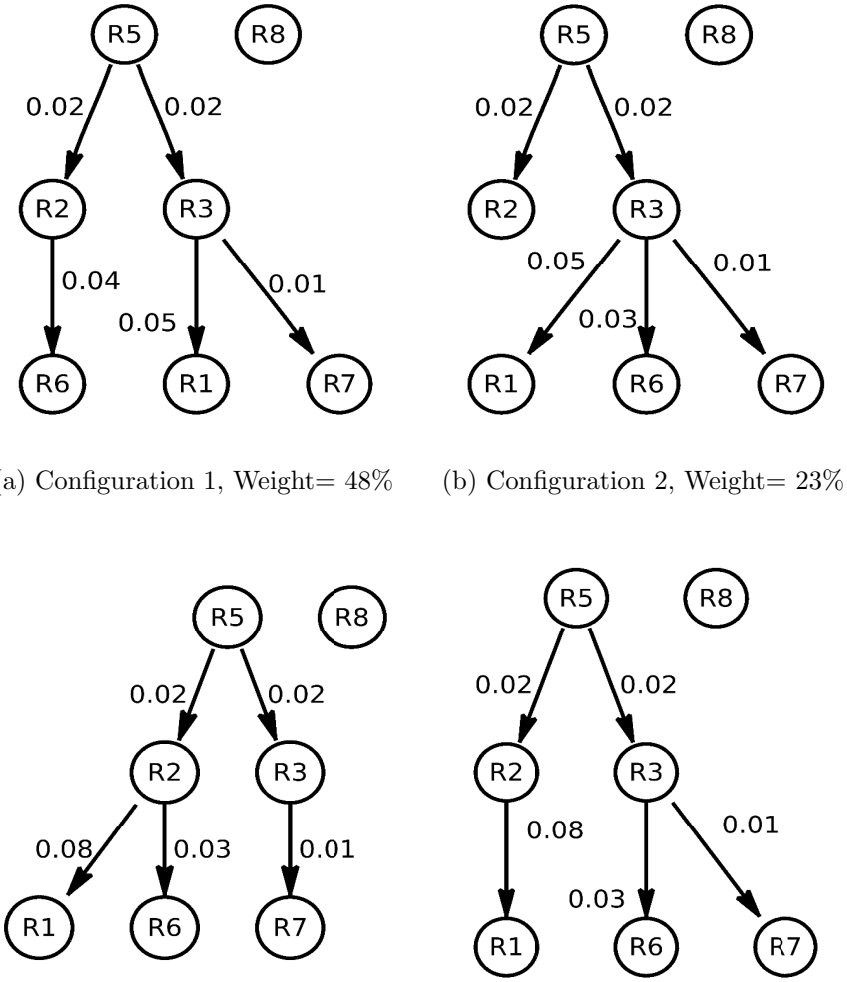


Figure 5.6 Observed time series in the impacted regions. The vertical axis show the number of separate locations within the region in which the disease was reported.

the four most probable configurations of the infection network and their percentages among generated samples. The edge weights in these graphs are estimated link strengths.

5.5.3 Measles and Chickenpox Data

We study the number of weekly reported cases of measles and chickenpox in England and Wales as shown in Figure 2.1. The analysis is based on the data from seven large cities (London, Bristol, Liverpool, Manchester, Newcastle, Birmingham, and Sheffield), with populations ranging from 300,000 to 10 million (out of a total population of approximately 50 million). The primary infections occur every two years in the period from September



(a) Configuration 1, Weight= 48% (b) Configuration 2, Weight= 23%

(c) Configuration 3, Weight= 17% (d) Configuration 4, Weight= 10%

Figure 5.7 Most probable network configurations

to December of the following year. We focus our analysis on these time windows. As mentioned before, our proposed model can capture scenarios in which infections are caused by interactions with other cities not included in the model (e.g., visitors from other countries). It can also explain the case when the residual infection in a city gives rise to a new outbreak, with no disease transfer from another city.

Quantitative studies (e.g., [121]) have explained the decrease in the number of reported cases in the data after widespread use of a vaccine began in 1968. Also, studies such as [122]

have justified the biennial data peaks, claiming that they were caused by exhaustion and subsequent build-up of susceptibles in the population as well as seasonal changes in virus transmission. Finally, [123] shows that once an infection starts in a region, the number of hospitalized cases can be approximated by a log-normal function of the number of days that have passed since the infection began. The number of hospitalized cases in region i at the n^{th} week is denoted by h_n^i and defined as the cumulative number of cases minus the cumulative number of deaths and recoveries. Hence, for $n \geq t_1^i$ we have

$$h_n^i = \frac{A_h^i}{\sqrt{2\pi}\sigma_h^i 7(n - t_1^i)} \exp(-\frac{(\ln[7(n - t_1^i)] - \mu_h^i)^2}{2{\sigma_h^i}^2}). \tag{5.35}$$

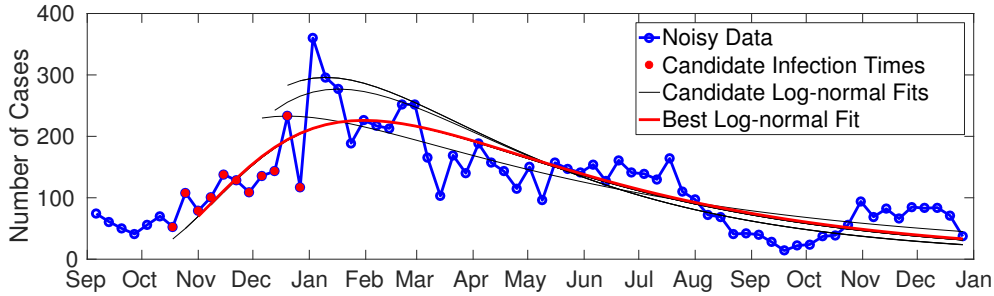
Here A_h^i is a constant, σ_h is the variance parameter of the log-normal, and we set $\mu_h^i = \ln D_h^i + {\sigma_h^i}^2$, where $D_h^i = \arg\max_n h_n^i$. The coefficient 7 is due to the fact that the data is reported weekly.

For each time-series, we use the log-normal function to model the data after an infection. For a candidate infection time, we find σ_h^i and D_h^i such that the Mean Square Error (MSE) is minimized. We also choose A_h^i such that $d_{D_h^i}^i = h_{D_h^i}^i$. The *individual infection time estimate* is then the candidate infection time that has minimum MSE after the log-normal fit. We model the residual using a Gaussian distribution. We also assume that the data follows a normal distribution before the individual infection times, i.e.,

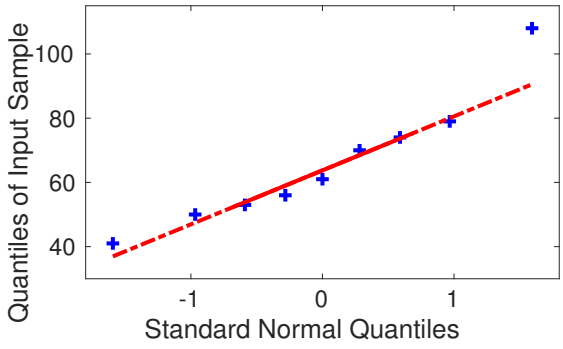
$$d_n^i | t_1^i \sim \begin{cases} \mathcal{N}(\mu_1^i, \sigma_1^i) & \text{if } n > t_1^i, \\ \mathcal{N}(h_n^i + \mu_2^i, \sigma_2^i) & \text{if } n \leq t_1^i. \end{cases} \tag{5.36}$$

In these models, we use empirical variances derived from the previous time-period.

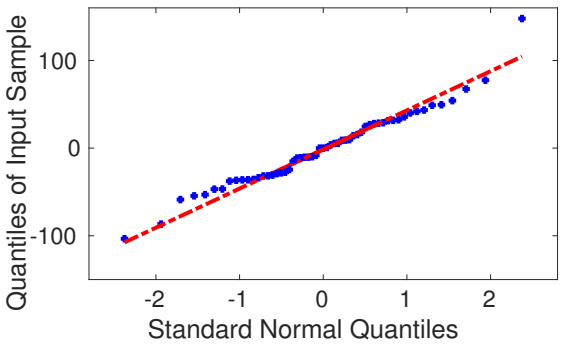
Figure 5.8 shows an example for 1958-1959 in Liverpool. The candidate infection times are shown by red points in Figure 5.8a. In 5.8a, the log-normal fits for each candidate infection time are shown in black. For the example of 5.8a, the individual infection time estimate is November 1^{st}. The quantile-quantile plots in Figures 5.8b and 5.8c provide support for the Gaussian model. Results for other years are provided in Appendix A.



(a) Reported cases for Liverpool in Sep 1958 to Dec 1959



(b) Reported data before the individual infection time



(c) Residual noise after the individual infection time

Figure 5.8 (a) Reported cases time series with best fit log-normals for different candidate infection times, (b) Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution, (c) qq plot of the residual noise after the individual infection time versus standard normal distribution

Estimating Infection Times

Figure 5.9 shows the reported data for the 1958-1959 study period. The individual infection time estimates are indicated by the change of color from green to red. In all the four successive biennial study periods from 1956 to 1963, Liverpool (node 3) has the earliest individual infection time estimate. We consider this node as the source of the infection. Given the entire data for each study period, we intend to infer the underlying network as well as the infection times. We use the prior distributions proposed in (5.4), (5.5), and (5.6). The data is modelled as in (5.36), with h_n^i calculated for each candidate infection time by minimizing the MSE as explained above. In each study period, we choose the hyper-parameters $\mu_1^i, \sigma_1^i, \mu_2^i, \sigma_2^i, \kappa, \theta$ based on the reported data and the individual infection

time estimates for the preceding period.

Any node j whose individual infection time estimate is earlier than the individual infection time estimate of node i in a study period is considered to be a potential parent, i.e. $j \in \pi^i$, for the next study period. We generate $N_{MCMC} = 10^5$ samples, discard the first $N_{burn} = 10^3$, and store one out of every $N_{thin} = 10$ of the rest. The means of the retained samples are used to calculate estimated infection times. These estimated infection times are shown by the black vertical lines in Figure 5.9. The estimated infection times are quite close to the individual estimates.

Predicting Infection Times

In addition to estimating the infection times, we are interested in predicting them beforehand without having access to the reported data. After we have learned a model, we can form a prediction using only the (individual) infection time estimate of the source node. This allows us to detect the onset of the infection in Liverpool and use it to predict when infections will arise in London and Manchester, for example. We use the inferred network structure for each study period to predict the infection times of the next period. For example, the infection times for 1958-1960 are predicted using the model learned by processing 1956-1958 data. For every stored sample from the distribution, we predict the infection time of node i by $t_1^{z_1^i} + \delta^{iz_1^i}$ where $\delta^{iz_1^i}$ is the mean of a geometric distribution with parameter $1 - e^{-\alpha_1^{iz_1^i}}$. The mean and $25 - 75\%$ confidence intervals of these predicted values are respectively shown by the vertical and horizontal blue lines in Figure 5.9. Figure 5.10 shows the absolute difference between the average estimated (predicted) infection times with the individual infection time estimates for all nodes except for the source.

5.5.4 Earthquake Data

In addition to estimating the infection times, our proposed inference approach can be used to detect the underlying network structure. As an example, we use our proposed diffusion framework to study seismic events in different regions of New Zealand [124]. Seismic waves are energy waves generated by earthquakes, volcanic eruptions, and other sources of earth vibration. They travel along layers of the earth and across the surface. Each wave is generated in one location and propagates out to other regions. The *epicenter* of a seismic event is the location on the surface of the earth directly above the cause of the wave. Our

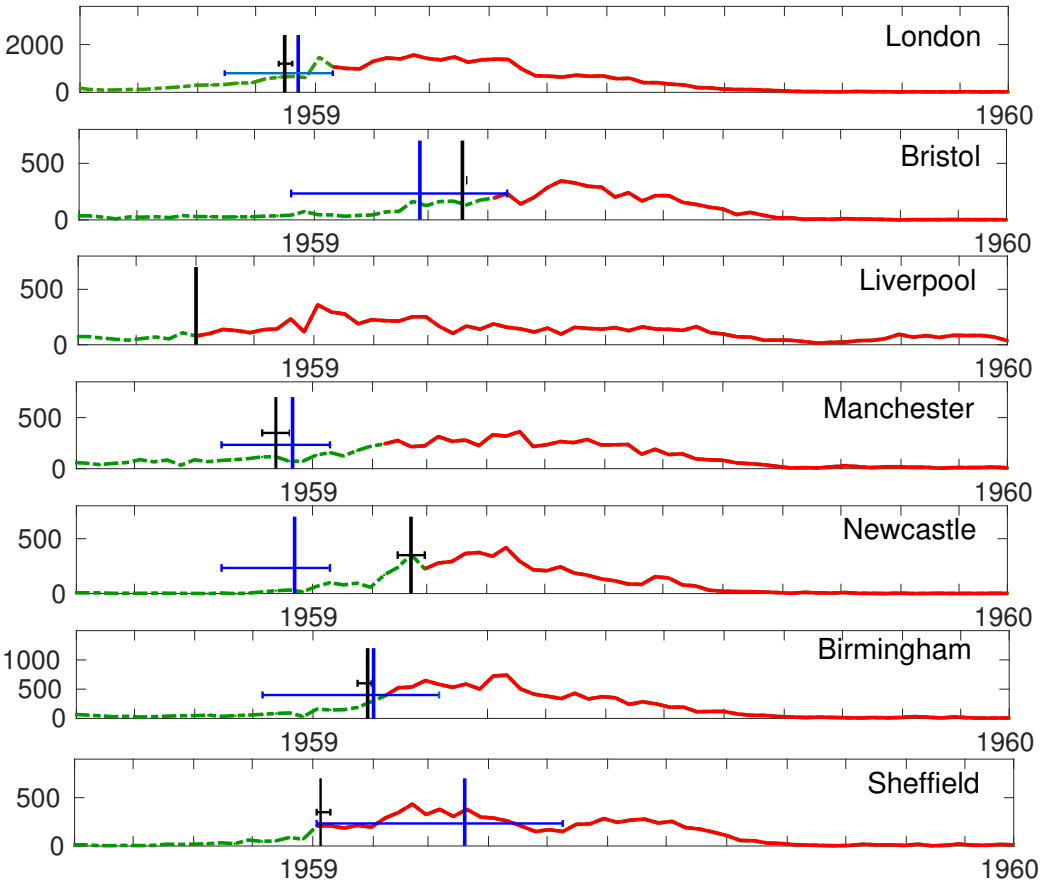


Figure 5.9 Number of reported cases with estimated and predicted infection times in 1958- 1959. dashed green line: susceptible state, solid red line: infected state (after individual infection time), black lines: mean and 25-75% of estimated values, blue lines: mean and 25-75% of predicted values

goal is to locate the epicenter of a seismic event and compare it with the reported real location. A seismograph is a device that records earthquake waves and we consider the measuring stations equipped with seismographs as the nodes of the diffusion network. We approximate the seismic event as a propagation of energy waves between these discrete nodes. A seismogram, the graph drawn by a seismograph, is a record of the ground motion as a function of time. The recorded seismograms are used as the observed data signals in our diffusion model. Three examples are shown in Figure 5.11 for an earthquake that occurred on November 1st 2015.

We choose the prior distribution for link strength values by fitting a Gamma distribution to inverse values of the geographic distance between station pairs. We also assume that

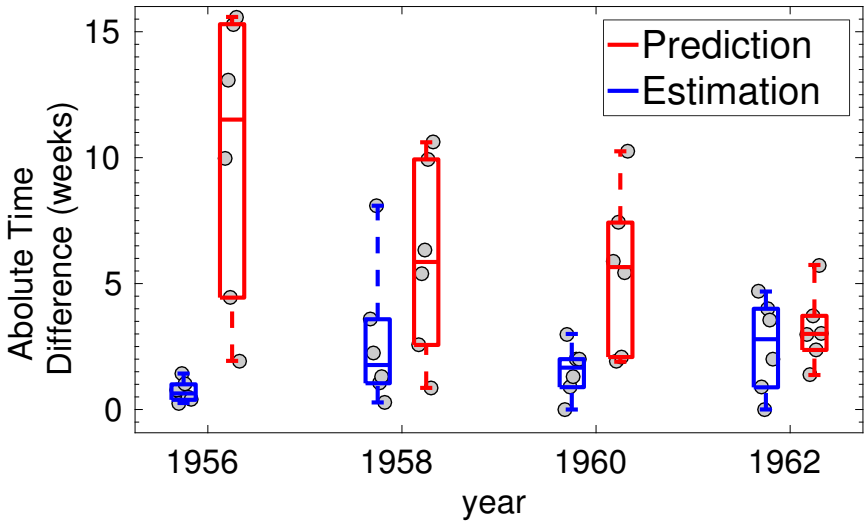


Figure 5.10 The absolute difference between estimated (predicted) infection times and the individual ones. The central mark of each box is the median, the edge of the box are the 25th and 75th percentiles, the whiskers extend to the highest values not considered as outliers.

seismic waves follow two different Gaussian distributions before and after being infected. This ignores the oscillatory structure and correlations in the time series, but is sufficient for the estimation of the arrival of the seismic wave. We denote the individual changepoint estimate of waveform i by $\hat{t'}^i$. Denoting the velocity of the seismic waves in the related depth of the earth by v, we define the set of potential parents for node i as $\pi_i = \{j \in \mathcal{N} | \hat{t'}^j + \frac{D_{ij}}{v} < \hat{t'}^i\}$ where D_{ij} is the distance between stations i and j. More precisely, node i can only become infected by node j if the time difference between their individual changepoints is larger than the time required for the seismic wave to traverse their spatial distance. The precise speed with which seismic waves travel throughout the earth depends on several factors such as composition of the rock, temperature, and pressure. They typically travel at speeds between 1 and 14 $\frac{Km}{s}$ [125].

Since we do not know where the source of the infection is, we assume that there is a dummy node within a radius of D_{i0} of each real node. These dummy nodes are candidate sources of the infection and each is a potential parent of its corresponding node. All dummy nodes become infected at the same time, $\min_i(\hat{t'}^i - \frac{D_{i0}}{v})$. As for the prior distribution of the infection time of node i, we know that it is zero for values less than $\frac{D_{iz_1^i}}{v}$. This probability increases for greater times up to a certain point and then it monotonically increases. We

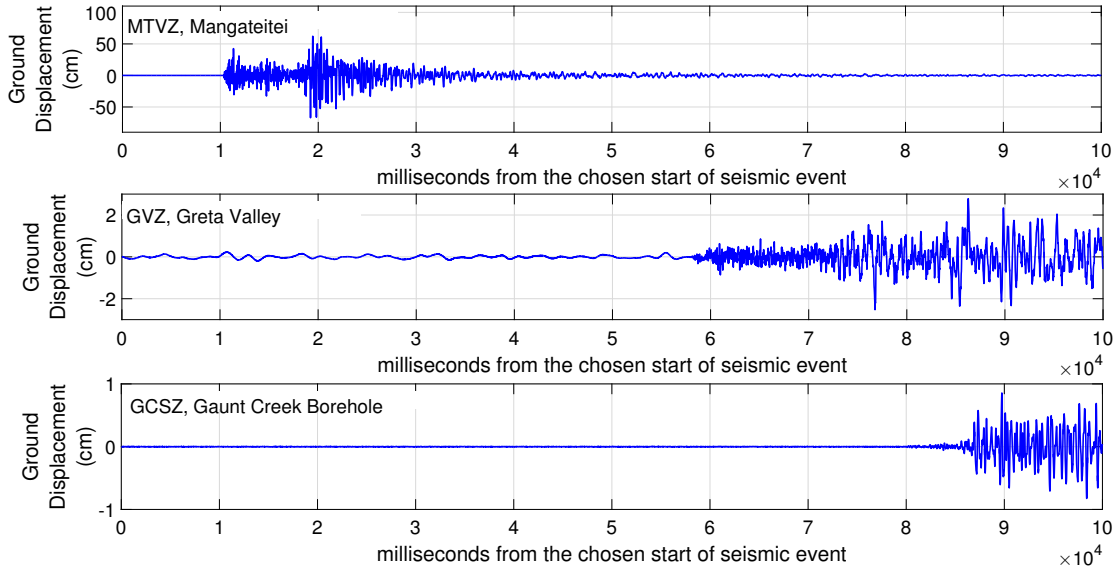


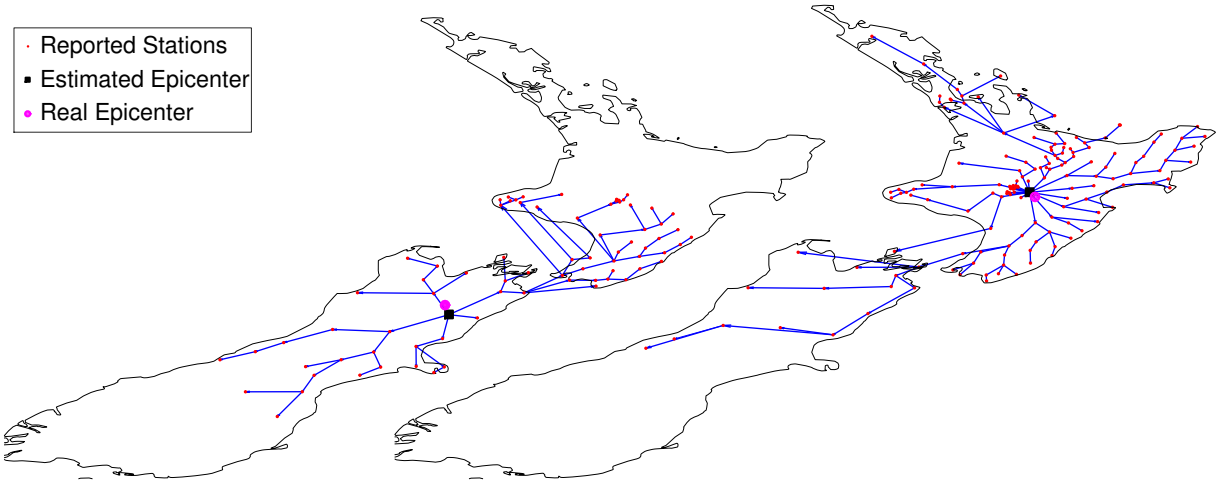
Figure 5.11 Three recorded seismic waveforms for an earthquake happened on November 1st 2015 (Event ID: 2015p822263)

approximate this behaviour with a geometric distribution as in (5.4).

We run the batch inference algorithm on the diffusion network with $N_{MCMC} = 10^5$, $N_{burn} = 10^3$, and $N_{thin} = 10$. For each node, we choose the node that marginally maximizes the posterior distribution as its detected parent. This marginal MAP approximation is the parent that occurs most often in the stored samples. We consider the geographical midpoint of the nodes that are infected by their dummy nodes as the approximate location of the epicenter. The detected and real epicenters of two seismic events are shown in Figure 5.12 where $v = 13\frac{Km}{s}$ and $D_{i0} = 10Km$ is used. We see that a tree-like network structure exists where the root is close to the real epicenter of the seismic event.

5.6 Summary

In this chapter, we presented a Bayesian framework for modelling the diffusion of some sort of a contagion over a graph structure. We formulated the diffusion process by introducing three main groups of parameters: parental relationships, the strength of connections between node pairs, and infection times. The main issue we addressed in this work was to simultaneously infer these three sets of parameters using data signals observed at each of these individual nodes. In order to do so, we applied MCMC techniques to generate



(a) *Event ID*: 2016p105478 (b) *Event ID*: 2015p822263

Figure 5.12 Detected network structure for two seismic events in New Zealand, The distance between real and detected epicenters are (a) $29Km$ and (b) $15Km$

samples from the posterior distribution of these parameters. One of the main contributions of this piece of work is to address applications in which parameters must be estimated before all of the data has been acquired. We addressed this concern by developing an online version of the inference algorithm.

We evaluated the performance of our proposed inference approaches on both synthetic and real-world network scenarios. In the synthetic dataset, model assumptions are exactly met and the ground truth is perfectly known. Simulation results showed that considering the underlying network structure in estimating infection times improves accuracy compared to processing the data at each node individually. We compared the precision of detecting the network structure (parental relationships and link strengths) in scenarios with known and unknown infection times. Finally, we tested our proposed inference algorithm in practical scenarios. First, we showed how the algorithm can use the number of reported cases of a contagious disease as the observed time series to construct estimates and predictions of the time of the year when disease outbreaks occur in a particular region. We then used the inference approach to locate the epicenters of earthquakes using the seismic waveforms recorded at seismic stations.

5.7 Proofs

5.7.1 Proof of Proposition 2

Proposition 2: $h(\mathbf{x}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML})$ *is a probability distribution function. In particular,*

$$\int \sum_{\mathbf{t}_b\in\mathcal{S}_1\times\cdots\times\mathcal{S}_1} \sum_{\mathbf{z}_b\in\mathcal{S}_2\times\cdots\times\mathcal{S}_2} h(\mathbf{x}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML}) d_{\boldsymbol{\alpha}_b} = 1$$

where $\mathcal{S}_1 = \{\mathbf{B}_1[1],\ldots,\mathbf{B}_b[M],\phi\}$ *and* $\mathcal{S}_2 = \pi_b^i \cup \{\phi\}$.

Replacing (5.22)-(5.26) in (5.7.1), we can check that the integrals are separable for different nodes. Hence, (5.7.1) is equal to

$$\prod_{i\in\mathcal{N}} \sum_{t_b^i\in\mathcal{S}_1} h_t(t_b^i|\mathbf{x}_{b-1};t_b^{ML^i}) \times \prod_{k\in\pi^i} h_{\alpha z}^k(t_b^i)\,, \tag{5.37}$$

where

$$h_{\alpha z}^k(t_b^i) = \int_0^\infty h_\alpha(\alpha_b^{ki}|t_b^i,\mathbf{x}_{b-1}) \sum_{z_b^i\in\mathcal{S}_2} h_z(z_b^i|t_b^i,\boldsymbol{\alpha}_b,\mathbf{x}_{b-1}) d_{\alpha_b^{ki}}\,. \tag{5.38}$$

For each node i, we calculate the component in (5.37) for two cases of $t_{b-1}^i = \phi$ and $t_{b-1}^i \neq \phi$ separately. When $t_{b-1}^i \neq \phi$, the integral is equal to

$$\sum_{t_b^i\in\mathcal{S}_1} \delta(t_b^i - t_{b-1}^i) \times \prod_{k\in\pi^i} \int_0^\infty \sum_{z_b^i\in\mathcal{S}_2} \delta(\alpha_b^{ik} - \alpha_{b-1}^{ik}) \delta(z_b^i - z_{b-1}^i) d_{\alpha_b^{ik}} = 1\,. \tag{5.39}$$

When $t_{b-1}^i = \phi$, we start from the innermost integral of (5.38)

$$\sum_{z_b^i\in\mathcal{S}_2} h_z(z_b^i|t_b^i,\boldsymbol{\alpha}_b,t_{b-1}^i) = \begin{cases} 1 & t_b^i = \phi\,, \\ 1 - \frac{\sum_{l\in\pi_b^i}\alpha_b^{il}}{\sum_{j\in\pi^i}\alpha_b^{ij}} + \frac{\sum_{l\in\pi_b^i}\alpha_b^{il}}{\sum_{j\in\pi^i}\alpha_b^{ij}} & t_b^i \neq \phi\,. \end{cases}$$

Hence, $h_{\alpha z}^k(t_b^i) = \int_0^\infty \Gamma(\kappa_{ki},\theta_{ki}) d_{\alpha_b^{ki}} = 1$. Similarly for $t_{b-1}^i = \phi$, $\sum_{t_b^i\in\mathcal{S}_1} h_t(t_b^i) = 1$. Therefore (5.37) always equals 1 and $h(\mathbf{x}_b|\mathbf{x}_{b-1};\mathbf{t}_b^{ML})$ is a probability distribution function. □

Chapter 6

Conclusions and Future Work

In this chapter, we provide concluding remarks and discuss possible directions for future research. This thesis addresses two different perspectives of studying the process of diffusion of information in various network scenarios. We referred to these perspectives as proactive and passive perspectives (or approaches). In a proactive approach, the goal is to control the diffusion process in order to achieve a desirable performance. In a passive approach, the goal is to learn about the dynamics between nodes based on the peripheral measurements of activity and correlation.

Having a proactive perspective, we studied the problem of routing messages in opportunistic Delay Tolerant Networks (DTNs) in Chapters 3 and 4. We overview the contributions we made to the problem of routing in DTNs as well as the conclusions we made from analytical proofs and simulation results in Section 6.1. We then suggest some ideas to further improve or extend our proposed routing methodology. Similarly in Section 6.2, we overview the contributions and conclusions of the passive study perspective we did in Chapter 5 as well as suggestions for future studies.

6.1 Proactive Approach

In Chapter 3, we formulated the routing problem in DTNs as an optimization problem in which we minimized the sum of expected latencies from all nodes of the network to a particular destination. The arguments of this optimization problem were the forwarding probabilities p_{ij} for all nodes i, j. We showed that the solution of this optimization problem is binary (i.e., either $p_{ij} = 0$ or $p_{ij} = 1$). We developed both centralized and decentralized

forwarding algorithms and proved their convergence to the optimal expected latencies. We then made the model more realistic by assuming that no a priori knowledge of meeting rates were available to the network nodes. We added a meeting rate estimation mechanism based on Maximum Likelihood (ML) estimation to the decentralized algorithm. Through mathematical analysis we showed that although this additional mechanism decelerates the convergence process, the resulted expected latencies eventually converge to the optimum values. We compared the performance of our proposed decentralized algorithm with the most similar existing routing approaches on both synthetic and real-world networks. Also, in order to evaluate the performance in non-idealized scenarios, we conducted simulations in larger networks with practical constraints like limited message life (TTL), buffer size and message exchange.

We then studied the routing problem in a Bayesian framework in Chapter 4 and showed how we can apply external information about the social ties to improve meeting rate estimations and expedite the convergence to the optimal expected latencies. We developed two Bayesian versions of the decentralized algorithm proposed in Chapter 3 and evaluated their performance efficiencies through experimental simulations In the rest of this section, we discuss two directions to further improve or extend the work of Chapters 3 and 4.

6.1.1 Improvement of Algorithms' Computational Complexities

The routing algorithms we proposed in Chapters 3 and 4 involve solving at least two optimization problems each time two nodes meet. We showed how each optimization problem can be converted to a linear programming problem in all cases except for BMinLat-II. However, we observed through simulations that solving the linear optimization problem is still the bottleneck for computational complexity of the algorithms. This is more crucial for BMinLat-I in which we solve the optimizations for each sample of the posterior distributions separately.

In this work, we solved the optimization problems each time two nodes meet without considering the solution of the last optimization that the nodes have solved. However, we know that between two successive meetings of node i, at most only two new pieces of information are added to what node i already knows about its meeting rates with the neighbours and their expected latencies. This information may not even change node i's decision about its forwarding decisions. A simple example is in the case where meeting rates are known

(i.e., in the MinLat algorithm). A node can limit its search for optimum forwarding rules to cases where at least one of its neighbours has a new estimate of the expected latency. In other words, if node j has not changed its estimate of its expected latency since the last meeting of i and j, node i can just skip solving the optimization problem and retain its current forwarding rules. In more general cases where we estimate the meeting rates or we work with samples of the expected latencies, the current forwarding rules may still be a good first guess that can be improved. Using Stochastic Local Search (SLS) methods [126], we may be able to expedite the process of finding the optimal forwarding solution each time two nodes meet and reduce the overall computational complexity of the algorithms.

6.1.2 Extension to Multi-copy Routing Algorithms

In Chapters 3 and 4, we focused on the scenarios in which messages are large and nodes' memory resources are limited. In order to address these restrictions, we developed routing algorithms that allow just one copy of each message in the network at any time. As mentioned in Chapter 3, these algorithms are referred to as single-copy algorithms. However, spreading more than one copy of each message in the network may reduce the expected delivery latency of each message and therefore increase the overall message delivery rate. In scenarios where some level of buffer occupancy is tolerable, it is possible to spread more than one copy of a message in the network and use a multi-copy routing algorithm.

The main question here is if the proposed algorithms of this thesis can be modified to be used as multi-copy routing algorithms. Assuming that at most M copies of each message are allowed to be spread in the network, it may not be straightforward to express the expected latency of a node i to the destination (i.e., L_{id}). However, we may be able to use the Bayesian framework introduced in Chapter 4 to make forwarding decisions based on the estimated expected latencies in the single-copy algorithm (i.e, Equation (3.2)). In this setup, node i may decide to give a copy of a message it has to node j based on the overlap between the two posterior distributions $f(\hat{L}_{id}(i)|x_{ik}^1, \ldots, x_{ik}^{n_{ik}}; k \in \mathcal{S}_i)$ and $f(\hat{L}_{jd}(j)|x_{jk}^1, \ldots, x_{jk}^{n_{jk}}; k \in \mathcal{S}_j)$.

6.2 Passive Approach

Having a passive perspective, we studied how we can infer the underlying network structure of a diffusion process and estimate the infection times. We assumed that we merely

observe a related time series from each node of the network. We considered the parental relationships, link strengths, and infection times as the network parameters and developed inference algorithms to simultaneously infer these three sets of parameters using data signals observed at each of the individual nodes. In order to do so, we applied Markov Chain Monte Carlo (MCMC) techniques to generate samples from the posterior distribution of network parameters conditioned on the observed time series. We then clarified how these particles can be updated as we receive more data and developed an online version of the inference algorithm. Simulation results on synthetic data showed that considering the underlying network structure in estimating infection times improves accuracy compared to processing the data at each node individually. We also showed how we can practically use the proposed framework through multiple examples on real-world datasets.

6.2.1 Extension to Multi-state Infection Models

In Chapter 5, we focused on the simple SI (Susceptible-Infected) model in which once an arbitrary node becomes infected, it never recovers or becomes susceptible again. However, as we saw in Section 5.5 (e.g., the Measles and Chickenpox dataset), the infection models can be more complicated in practice. It would be interesting to extend our methodology and develop more sophisticated algorithms that account for more complicated infection models. Here, we review some other infection models introduced in the epidemiology literature (e.g., [127]). Although we explain the models in the context of contagious diseases, these models can be used in other applications (mentioned in Section 2.3) as well.

- SIR (Susceptible-Infected-Recovered): This model is used to describe a disease from which infected nodes recover with immunity against reinfection.

- SIS (Susceptible-Infected-Susceptible): In this model, an infected node recovers after some time and cannot transfer the infection to another node. However, it is not immune against becoming reinfected. Hence, it returns to the susceptible state again.

- SEIR (Susceptible-Exposed-Infected-Recovered): In some infectious diseases there is an exposed period after the transmission of infection from susceptible nodes to potentially infected nodes but before these potentially infected nodes can transmit the infection. If the exposed period is short, it is often neglected in modelling. A longer exposed period could perhaps lead to significantly different model predictions [127].

– MSIR (iMmune-Susceptible-Infected-Recovered): There are several diseases where an individual node is born with a passive immunity.

The first step towards extending the model is to define other parameters to describe the transition times between states and choose appropriate prior probability distributions for these parameters. Then, the conditional probability distributions (i.e., Equations (5.8)-(5.10)) should be revised to address the probability of transitioning between these new states.

6.2.2 Extension to Dynamic Networks

In the model we studied in Chapter 5, we assumed that the set of potential parents for node i (i.e., π^i) is known and fixed during the study period. The next step to make the model more practical is to consider π^i (or π_b^i in the online setup) as another unknown random variable and derive its probability distribution conditioned on other variables. Besides, this can help modelling a dynamic network in which nodes can enter or leave the network or the set of potential parents changes over time.

Bibliography

[1] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, "Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 2, pp. 607–640, 2012.

[2] Y. Cao and Z. Sun, "Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 654–677, 2013.

[3] K. Wei, X. Liang, and K. Xu, "A survey of social-aware routing protocols in delay tolerant networks: applications, taxonomy and design-related issues," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 556–578, 2014.

[4] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: Data forwarding in disconnected mobile ad hoc networks," *IEEE Commun. Mag*, vol. 44, no. 11, pp. 134–141, 2006.

[5] N. Chakchouk, "A survey on opportunistic routing in wireless communication networks," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2214–2241, 2015.

[6] B. E. Pataki and L. Kovács, "Sensor data collection experiments with chaoster in the fed4fire federated testbeds," in *IEEE Wireless and Mobile Comput., Net., and Comm. (WiMob)*, (Larnaca, Cyprus), Oct. 2014.

[7] OpenGarden, "Case studies." `https://www.opengarden.com/case-studies.html`, Jun. 2016.

[8] A. Martín-Campillo, J. Crowcroft, E. Yoneki, and R. Martí, "Evaluating opportunistic networks in disaster scenarios," *J. Netw. and Comput. Appl.*, vol. 36, no. 2, pp. 870–880, 2013.

[9] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet," in *ACM Sigplan Notices*, vol. 37, pp. 96–107, 2002.

[10] T. Small and Z. J. Haas, "The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way)," in *Proc. ACM Int. Symp. Mobile Ad hoc Netw. and Comput. (MobiHoc)*, (Annapolis, MD, USA), pp. 233–244, Jun. 2003.

[11] A. Pentland, R. Fletcher, and A. Hasson, "Daknet: Rethinking connectivity in developing nations," *IEEE Comput.*, vol. 37, no. 1, pp. 78–83, 2004.

[12] A. Doria, M. Uden, and D. Pandey, "Providing connectivity to the saami nomadic community," *Proc. Int. Conf. Open Collaborative Design for Sustain. Innovation*, Dec. 2002.

[13] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. Mobile Comput.*, vol. 11, pp. 821–834, May 2012.

[14] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," tech. rep., Duke Univ., Durham, NC, USA, 2000.

[15] Q. Wang and Z. J. Haas, "Analytical model of epidemic routing for delay-tolerant networks," in *Proc. ACM Int. Workshop High Perf. Mobile Opportunistic Sys.*, (Paphos, Cyprus), pp. 1–8, Oct. 2012.

[16] D. J. Klein, J. Hespanha, and U. Madhow, "A reaction-diffusion model for epidemic routing in sparsely connected MANETs," in *Proc. IEEE Infocom*, (San Diego, CA, USA), pp. 1–9, Mar. 2010.

[17] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *Proc. IEEE Infocom*, vol. 3, (Anchorage, AL, USA), pp. 1360–1369, Apr. 2001.

[18] R. Groenevelt, P. Nain, and G. Koole, "Message delay in MANET," in *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, pp. 412–413, 2005.

[19] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," *Comput. Netw.*, vol. 51, no. 10, pp. 2867–2891, 2007.

[20] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proc. ACM Sigcomm Workshop Delay Tolerant Netw.*, (Philadelphia, PA, USA), pp. 252–259, Aug. 2005.

[21] R. Ramanathan, R. Hansen, P. Basu, R. Rosales-Hain, and R. Krishnan, "Prioritized epidemic routing for opportunistic networks," in *Proc. ACM Int. Workshop Mobile Opportunistic Netw.*, (San Juan, Puerto Rico), pp. 62–66, Jun. 2007.

[22] M. Khouzani, S. Eshghi, S. Sarkar, N. B. Shroff, and S. S. Venkatesh, "Optimal energy-aware epidemic routing in DTNs," in *Proc. ACM Int. Symp. Mobile Ad hoc Netw. and Comput. (MobiHoc)*, (Hilton Head Island, SC, USA), pp. 175–182, Jun. 2012.

[23] T. Matsuda and T. Takine, "(p,q)-epidemic routing for sparsely populated mobile ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 5, 2008.

[24] J. A. Davis, A. H. Fagg, and B. N. Levine, "Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks," in *Proc. IEEE Int. Symp. Wearable Comput.*, (Zurich, Germany), pp. 141–148, Oct. 2001.

[25] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Comput. and Commun. Rev.*, vol. 7, pp. 19–20, Jul. 2003.

[26] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic routing protocol for intermittently connected networks." RFC 6693, Aug. 2012.

[27] S. Grasic, E. Davies, A. Lindgren, and A. Doria, "The evolution of a dtn routing protocol-prophetv2," in *Proc. ACM Workshop Challenged Netw.*, (Las Vegas, NV, USA), pp. 27–30, Sept. 2011.

[28] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks.," in *Proc. IEEE Infocom*, vol. 6, (Barcelona, Spain), pp. 1–11, Apr. 2006.

[29] M. Musolesi, S. Hailes, and C. Mascolo, "Adaptive routing for intermittently connected mobile ad hoc networks," in *IEEE Int. Symp. World of Wireless Mobile and Multimedia Net. (WoWMoM)*, (Taormina , Italy), pp. 183–189, Jun. 2005.

[30] B. Burns, O. Brock, and B. Neil Levine, "Mv routing and capacity building in disruption tolerant networks," in *Proc. IEEE Infocom*, (Miami, FL, USA), march 2005.

[31] K. Tan, Q. Zhang, and W. Zhu, "Shortest path routing in partially connected ad hoc networks," in *IEEE Globecom*, vol. 2, (San Francisco, CA, USA), pp. 1038–1042, 2003.

[32] E. P. Jones, L. Li, J. K. Schmidtke, and P. A. Ward, "Practical routing in delay-tolerant networks," *IEEE Trans. Mobile Comp.*, vol. 6, no. 8, pp. 943–959, 2007.

[33] S. C. Nelson, M. Bakht, and R. Kravets, "Encounter-based routing in dtns," in *Infocom*, (Rio de Janeiro, Brazil), pp. 846–854, Apr. 2009.

[34] Y. Zhu, B. Xu, X. Shi, and Y. Wang, "A survey of social-based routing in delay tolerant networks: Positive and negative social effects," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 387–401, 2013.

[35] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proc. ACM Int. Symp. Mobile Ad hoc Net. Comput. (MobiHoc)*, (Montreal, Canada), pp. 32–40, Sep. 2007.

[36] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 11, pp. 1576–1589, 2011.

[37] D. A. Sharma and M. Coates, "Contact graph based routing in opportunistic networks," in *Proc. IEEE Global Conf. Sig. and Info. Proc. (GlobalSIP)*, (Austin, TX, USA), Dec. 2013.

[38] M. Xiao, J. Wu, and L. Huang, "Community-aware opportunistic routing in mobile social networks," *IEEE Trans. Comput.*, vol. 63, no. 7, pp. 1682–1695, 2013.

[39] P. V. Marsden, "Egocentric and sociocentric measures of network centrality," *Social networks*, vol. 24, no. 4, pp. 407–422, 2002.

[40] Y. Li, G. Su, D. O. Wu, D. Jin, L. Su, and L. Zeng, "The impact of node selfishness on multicasting in delay tolerant networks," *IEEE Trans. Veh. Tech.*, vol. 60, no. 5, pp. 2224–2238, 2011.

[41] P. Sermpezis and T. Spyropoulos, "Understanding the effects of social selfishness on the performance of heterogeneous opportunistic networks," *Comput. Commun.*, vol. 48, pp. 71–83, 2014.

[42] A. Mei and J. Stefa, "Give2get: Forwarding in social mobile wireless networks of selfish individuals," *IEEE Trans. Dependable and Secure Comput.*, vol. 9, no. 4, pp. 569–582, 2012.

[43] U. Shevade, H. H. Song, L. Qiu, and Y. Zhang, "Incentive-aware routing in dtns," in *IEEE Int. Conf. Netw. Protocols (ICNP)*, (Orlando, FL, USA), pp. 238–247, Oct. 2008.

[44] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "Smart: A secure multilayer credit-based incentive scheme for delay-tolerant networks," *IEEE Trans. Veh. Tech.*, vol. 58, no. 8, pp. 4628–4639, 2009.

[45] B. B. Chen and M. C. Chan, "Mobicent: a credit-based incentive system for disruption tolerant network," in *Proc. IEEE Infocom*, (San Diego, CA, USA), pp. 1–9, Mar. 2010.

[46] H. Zhang, Z. Zhang, and H. Dai, "Gossip-based information spreading in mobile networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 11, pp. 5918–5928, 2013.

[47] V. Conan, J. Leguay, and T. Friedman, "Fixed point opportunistic routing in delay tolerant networks," *IEEE J. Sel. Areas in Commun.*, vol. 26, no. 5, pp. 773–782, 2008.

[48] M. Xiao, J. Wu, C. Liu, and L. Huang, "Tour: Time-sensitive opportunistic utility-based routing in delay tolerant networks," in *Proc. IEEE Infocom*, (Turin, Italy), pp. 2085–2091, Apr. 2013.

[49] C. Boldrini, M. Conti, and A. Passarella, "Modelling social-aware forwarding in opportunistic networks," in *Perform. Eval. of Comput. and Commun. Syst. Milestones and Future Challenges*, (Vienna, Austria), pp. 141–152, Oct. 2010.

[50] C. Boldrini, M. Conti, and A. Passarella, "Performance modelling of opportunistic forwarding with imprecise knowledge," in *Proc. Int. Symp. Model. and Opt. in Mobile, Ad Hoc and Wireless Net. (WiOpt)*, (Paderborn, Germany), pp. 216–223, 2012.

[51] E. Hernández-Orallo, J. C. Cano, C. T. Calafate, and P. Manzoni, "New approaches for characterizing inter-contact times in opportunistic networks," *Ad Hoc Netw.*, vol. 52, pp. 160–172, 2016.

[52] Y. Feng, "Detect method of time series,Äô abnormal value for predictive model," in *Int. Conf.Intell. Comput. (ICIC)*, (Lanzhou,China), pp. 735–742, Aug. 2016.

[53] J. Miao, O. Hasan, S. B. Mokhtar, L. Brunie, and G. Gianini, "A delay and cost balancing protocol for message routing in mobile delay tolerant networks," *Ad Hoc Netw.*, vol. 25, pp. 430–443, 2015.

[54] T. Kimura and C. Premachandra, "Optimal relay node selection in two-hop routing for intermittently connected manets," *J Wireless Mobile Netw., Ubiquitous Comput., and Dependable Appl.*, vol. 7, no. 1, pp. 23–38, 2016.

[55] A. C. K. Vendramin, A. Munaretto, M. R. Delgado, M. Fonseca, and A. C. Viana, "A social-aware routing protocol for opportunistic networks," *Expert Syst. with Appl.*, vol. 54, pp. 351–363, 2016.

[56] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Congress Evol. Comput. (CEC)*, vol. 2, pp. 1470–1477, 1999.

[57] R. G. Reynolds, "An introduction to cultural algorithms," in *Proc. Evol. Program.*, vol. 131139, (San Diego, CA, USA), Feb. 1994.

[58] M. A. Alim, X. Li, N. P. Nguyen, M. T. Thai, and A. Helal, "Structural vulnerability assessment of community-based routing in opportunistic networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 12, pp. 3156–3170, 2016.

[59] H. Guo, X. Wang, H. Cheng, and M. Huang, "A routing defense mechanism using evolutionary game theory for delay tolerant networks," *Applied Soft Comput.*, vol. 38, pp. 469–476, 2016.

[60] X. Geng, Y. Wang, H. Feng, and L. Zhang, "Lanepost: lane-based optimal routing protocol for delay-tolerant maritime networks," *China Commun.*, vol. 14, no. 2, pp. 65–78, 2017.

[61] N. Rajpoot and R. S. Kushwah, "An improved prophet routing protocol for underwater communication," in *Proc. Int. Conf. Commun. Netw. (ICCN)*, (Gwalior, India), pp. 27–32, Nov. 2015.

[62] R. Monteiro, L. Guedes, T. Condeixa, F. Neves, S. Sargento, L. Guardalben, and P. Steenkiste, "Lessons learned from a real vehicular network deployment of delay-tolerant networking," in *Proc. Int. Conf. Commun. Workshop (ICCW)*, (London, UK), pp. 2489–2494, Jun. 2015.

[63] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM Sigmod Record*, vol. 42, no. 2, pp. 17–28, 2013.

[64] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley, and H. A. Makse, "Identification of influential spreaders in complex networks," *Nature physics*, vol. 6, no. 11, pp. 888–893, 2010.

[65] D. M. Romero, W. Galuba, S. Asur, and B. A. Huberman, "Influence and passivity in social media," in *Mach. Learn. and Knowl. Discov. in Databases*, (Athens, Greece), pp. 18–33, Sep. 2011.

[66] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. ACM Int. Conf. Knowl. Discov. and Data Min.*, (Washington, DC, USA), pp. 137–146, Aug. 2003.

[67] H. N. Ozsoylev, J. Walden, M. D. Yavuz, and R. Bildik, "Investor networks in the stock market," *Rev. Financ. Stud.*, vol. 27, no. 5, pp. 1323–1366, 2014.

[68] K. R. Ahern, "Network centrality and the cross section of stock returns," *Available at SSRN 2197370*, 2013.

[69] M. Cataldi, L. Di Caro, and C. Schifanella, "Emerging topic detection on twitter based on temporal and social terms evaluation," in *Proc. Int. Workshop Multimedia Data Min.*, (Washington, DC, USA), p. 4, Jul. 2010.

[70] J. Leskovec, L. Backstrom, and J. Kleinberg, "Meme-tracking and the dynamics of the news cycle," in *Proc. ACM Int. Conf. Knowl. Discov. and Data Min.*, (Paris, France), pp. 497–506, Jun. 2009.

[71] D. A. Shamma, L. Kennedy, and E. F. Churchill, "Peaks and persistence: modeling the shape of microblog conversations," in *Proc. ACM Conf. Comput. Supported Cooperative Work and Social Comput. (CSCW)*, (Hangzhou, China), pp. 355–358, Mar. 2011.

[72] M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 561–568, Jun.

[73] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *ACM Trans. Knowl. Discov. from Data*, vol. 5, no. 4, p. 21, 2012.

[74] V. A. Smith, J. Yu, T. V. Smulders, A. J. Hartemink, and E. D. Jarvis, "Computational inference of neural information flow networks," *PLoS Comput. Biol.*, vol. 2, no. 11, p. e161, 2006.

[75] S. M. Hill, Y. Lu, J. Molina, L. M. Heiser, P. T. Spellman, T. P. Speed, J. W. Gray, G. B. Mills, and S. Mukherjee, "Bayesian inference of signaling network topology in a cancer cell line," *Bioinf.*, vol. 28, no. 21, pp. 2804–2810, 2012.

[76] A. Louni and K. Subbalakshmi, "Diffusion of information in social networks," in *Social Netw.*, pp. 1–22, 2014.

[77] K. Barry, "Ford bets the fiesta on social networking." `https://www.wired.com/2009/04/how-the-fiesta/`, Apr. 2009.

[78] "Northeast rumours: 254 websites blocked." `http://www.news18.com/news/india/govt-latest-500164.html`, Aug. 2012.

[79] G. Strauss, A. Shell, R. Yu, and B. Acohido, "Sec, fbi probe fake tweet that rocked stocks." `http://www.usatoday.com/story/news/nation/2013/04/23/hack-attack-on-associated-press-shows-vulnerable-media/2106985/`, Apr. 2013.

[80] F. Liljeros, C. R. Edling, and L. A. N. Amaral, "Sexual networks: implications for the transmission of sexually transmitted infections," *Microbes and Infection*, vol. 5, no. 2, pp. 189–196, 2003.

[81] J. Wallinga and P. Teunis, "Different epidemic curves for severe acute respiratory syndrome reveal similar impacts of control measures," *American J. Epidemiology*, vol. 160, no. 6, pp. 509–516, 2004.

[82] H. Hong, J. D. Kubik, and J. C. Stein, "Thy neighbor's portfolio: Word-of-mouth effects in the holdings and trades of money managers," *J. Financ.*, vol. 60, no. 6, pp. 2801–2824, 2005.

[83] J. Yang and J. Leskovec, "Modeling information diffusion in implicit networks," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, (Sydney, Australia), pp. 599–608, Dec. 2010.

[84] Y. Wang, G. Xiang, and S.-K. Chang, "Sparse linear influence model for hot user selection on mining a social network," in *Proc. Int. Conf. Soft. Eng. and Knowl. Eng. (SEKE)*, (San Francisco, CA, USA), pp. 1–6, Jul. 2012.

[85] M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf, "Structure and dynamics of information pathways in online media," in *Proc. ACM Int. Conf. Web Search and Data Min. (WSDM)*, (Rome, Italy), pp. 23–32, Feb. 2013.

[86] M. Farajtabar, Y. Wang, M. Gomez-Rodriguez, S. Li, H. Zha, and L. Song, "Coevolve: A joint point process model for information diffusion and network co-evolution," in *Proc. Advances in Neural Info. Process. Syst. (NIPS)*, (Montreal, Canada), pp. 1945–1953, Dec. 2015.

[87] V. R. Embar, R. K. Pasumarthi, and I. Bhattacharya, "A bayesian framework for estimating properties of network diffusions," in *Proc. ACM Int. Conf. Knowl. Discov. and Data Min. (KDD)*, (New York City, NY, USA), pp. 1216–1225, Aug. 2014.

[88] B. Bolker, "Infectious disease data." `https://ms.mcmaster.ca/bolker/measdata.html`.

[89] E. Sefer and C. Kingsford, "Convex risk minimization to infer networks from probabilistic diffusion data at multiple scales," in *Proc. Int. Conf. Data Eng. (ICDE)*, (Seoul, South Korea), pp. 663–674, Apr. 2015.

[90] M. Farajtabar, M. Gomez-Rodriguez, N. Du, M. Zamani, H. Zha, and L. Song, "Back to the past: Source identification in diffusion networks from partially observed cascades," in *Proc. Int. Conf. Artif. Intell. and Stat. (AISTATS)*, (Fort Lauderdale, FL, USA), pp. 232–240, Apr. 2015.

[91] K. Amin, H. Heidari, and M. Kearns, "Learning from contagion (without timestamps)," in *Proc. Int. Conf. Mach. Learn. (ICML)*, (Beijing, China), pp. 1845–1853, Jun. 2014.

[92] A. Y. Lokhov and T. Misiakiewicz, "Efficient reconstruction of transmission probabilities in a spreading process from partial observations," *arXiv preprint arXiv:1509.06893*, 2015.

[93] I. A. Eckley, P. Fearnhead, and R. Killick, "Analysis of changepoint models," *Bayesian Time Series Model.*, pp. 205–224, 2011.

[94] P. Fearnhead, "Exact Bayesian curve fitting and signal segmentation," *IEEE Trans. Signal Process.*, vol. 53, no. 6, pp. 2160–2166, 2005.

[95] R. Killick, P. Fearnhead, and I. Eckley, "Optimal detection of changepoints with a linear computational cost," *J. American Stat. Assoc.*, vol. 107, no. 500, pp. 1590–1598, 2012.

[96] D. S. Matteson and N. A. James, "A nonparametric approach for multiple change point analysis of multivariate data," *J. American Stat. Assoc.*, vol. 109, no. 505, pp. 334–345, 2014.

[97] X. Xuan and K. Murphy, "Modeling changing dependency structure in multivariate time series," in *Proc. Int. Conf. Mach. Learn. (ICML)*, (Corvallis, OR, USA), pp. 1055–1062, Jun. 2007.

[98] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (aodv) routing," Tech. Rep. RFC 3561. 2003, Nokia Research Center, University of California, Santa Barbara, University of Cincinnati, 2003.

[99] D. Johnson, Y.-c. Hu, and D. Maltz, "The dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4," tech. rep., 2007.

[100] H. Cai and D. Y. Eun, "Crossing over the bounded domain: from exponential to power-law intermeeting time in mobile ad hoc networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1578–1591, 2009.

[101] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Trans. Mob. Comput.*, vol. 6, pp. 606–620, Jun. 2007.

[102] V. Conan, J. Leguay, and T. Friedman, "The heterogeneity of inter-contact time distributions: its importance for routing in delay tolerant networks," *arXiv preprint cs/0609068*, 2006.

[103] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: A social network perspective," in *Proc. ACM MobiHoc*, (New Orleans, LA, USA), pp. 299–308, May 2009.

[104] K. Lee, Y. Y. amd J. Jeong, H. Won, I. Rhee, and S. Chong, "Max-contribution: On optimal resource allocation in delay tolerant networks," in *Proc. IEEE Infocom*, (San Diego, CA, USA), pp. 1–9, Mar. 2010.

[105] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. M. Ni, "Recognizing exponential inter-contact time in VANETs," in *Proc. IEEE Infocom*, (San Diego, CA, USA), pp. 1–5, Mar. 2010.

[106] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD data set cambridge/haggle (v. 2006-01-31)." Downloaded from `http://crawdad.org/cambridge/haggle/`, Jan. 2006.

[107] S. Ferretti, V. Ghini, and F. Panzieri, "Scale-free opportunistic networks: Is it possible?," in *Proc. IEEE Int. Workshop Pervasive Comput. and Commun. (PERCOM)*, pp. 625–630, 2012.

[108] A. Charnes and W. W. Cooper, "Programming with linear fractional functionals," *Naval Research logistics quarterly*, vol. 9, no. 3-4, pp. 181–186, 1962.

[109] R. Vanderbei, *Linear programming: foundations and extensions*, vol. 196. Springer, fourth ed., 2014.

[110] R. H. Berk, "Consistency and asymptotic normality of MLE's for exponential models," *The Annals of Math. Stat.*, vol. 43, no. 1, pp. 193–204, 1972.

[111] H. B. Mann and A. Wald, "On stochastic limit and order relationships," *The Annals of Math. Stat.*, vol. 14, no. 3, pp. 217–226, 1943.

[112] H. J. Bierens, *Introduction to the mathematical and statistical foundations of econometrics*. Cambridge University Press, 2004.

[113] J. G. Carlsson and J. Shi, "A linear relaxation algorithm for solving the sum-of-linear-ratios problem with lower dimension," *Operations Research Letters*, vol. 41, no. 4, pp. 381–389, 2013.

[114] Y. Hu, J. Shi, and S. Watanabe, "A revised algorithm for solving the sum of linear ratios problem with lower dimension using linear relaxation," *Int. J. Operations Research*, vol. 11, no. 1, pp. 28–39, 2014.

[115] Y. Gao and S. Jin, "A global optimization algorithm for sum of linear ratios problem," *J. Applied Math.*, vol. 2013, 2013.

[116] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica: J. Econ. Soc.*, pp. 497–520, 1960.

[117] A.-K. Pietilainen and C. Diot, "CRAWDAD dataset thlab/sigcomm2009 (v. 2012-07-15)." Downloaded from `http://crawdad.org/thlab/sigcomm2009/20120715`, Jul. 2012.

[118] F. Septier, S. K. Pang, A. Carmi, and S. Godsill, "On MCMC-based particle methods for Bayesian filtering: Application to multitarget tracking," in *Proc. IEEE Int. Workshop Comput. Advances in Multi-Sensor Adaptive Process. (CAMSAP)*, (Aruba, Dutch Antilles), pp. 360–363, Dec. 2009.

[119] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[120] "EMPRES-i global animal disease information system." Downloaded from `http://empres-i.fao.org`, Jan. 2016.

[121] B. Bolker and B. Grenfell, "Impact of vaccination on the spatial correlation and persistence of measles dynamics," *Proc. Nat. Academy Sci.*, vol. 93, no. 22, pp. 12648–12653, 1996.

[122] B. Bolker, "Chaos and complexity in measles models: a comparative numerical study," *Math. Med. and Biol.*, vol. 10, no. 2, pp. 83–95, 1993.

[123] W. Wang, Z. Wu, C. Wang, and R. Hu, "Modelling the spreading rate of controlled communicable epidemics through an entropy-based thermodynamic model," *Science China Phys., Mech., and Astro.*, vol. 56, no. 11, pp. 2143–2150, 2013.

[124] "Geonet." `http://www.geonet.org.nz/quakes`, Oct. 2016.

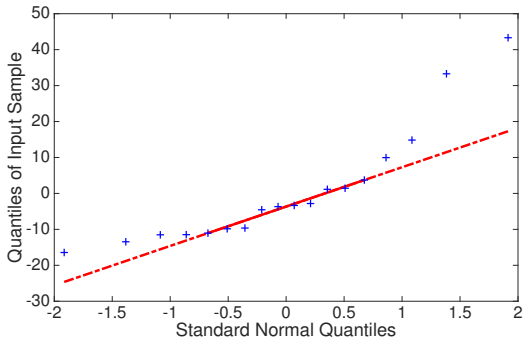
[125] G. R. Helffrich and B. J. Wood, "The earth's mantle," *Nature*, vol. 412, no. 6846, pp. 501–507, 2001.

[126] H. H. Hoos and T. Stützle, *Stochastic local search: Foundations and applications*. Elsevier, 2004.

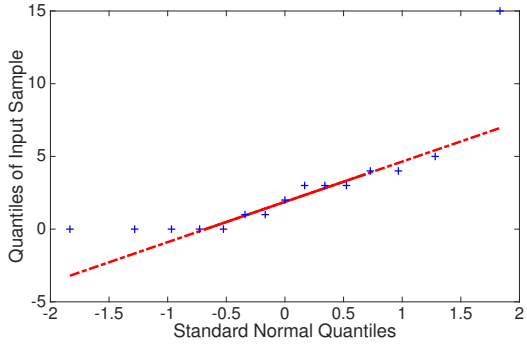
[127] L. Allen, F. Brauer, P. Van den Driessche, and J. Wu, "Mathematical epidemiology," *Lecture Notes in Math.*, vol. 1945, pp. 81–130, 2008.

Appendix A

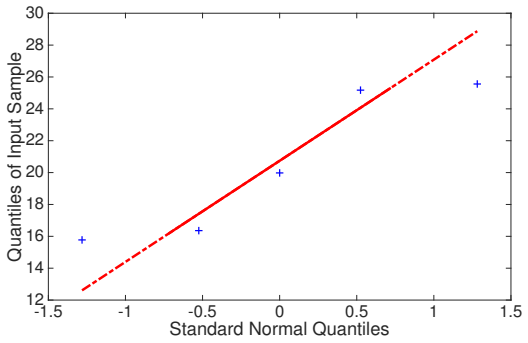
Figures A.1 to A.14 show the quantile-quantile plots for all the seven nodes (cities) in seven successive biennial study periods (1952-1966) to provide support for the Gaussian model.



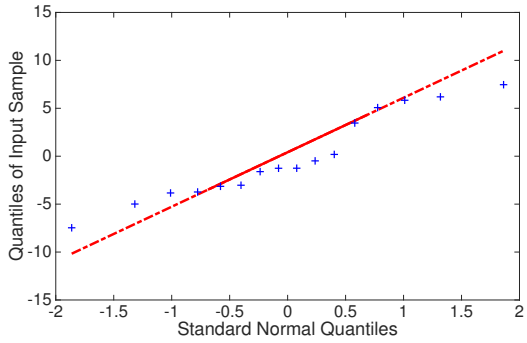
(a) London



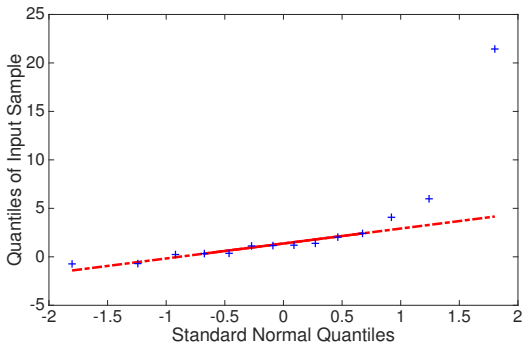
(b) Bristol



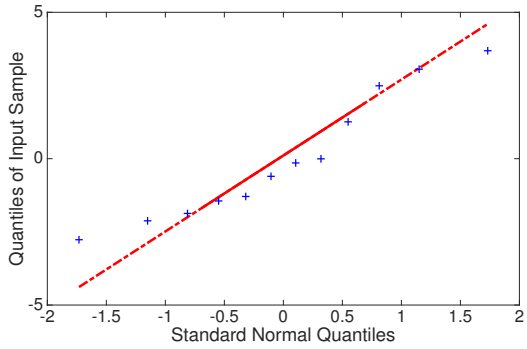
(c) Liverpool



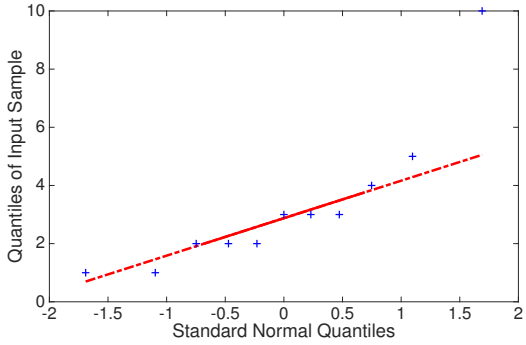
(d) Manchester



(e) Newcastle

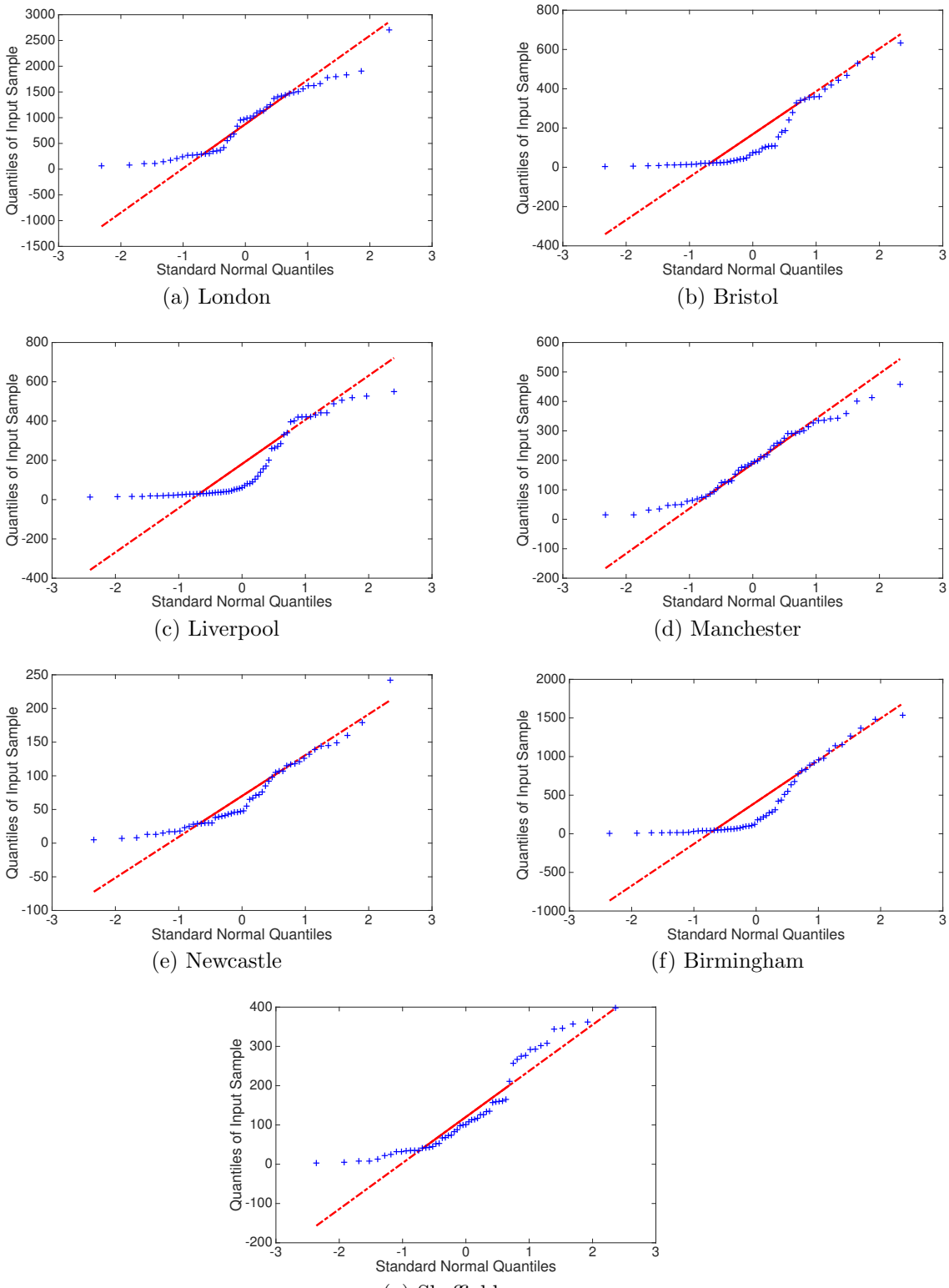


(f) Birmingham



(g) Sheffield

Figure A.1 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1952-1954



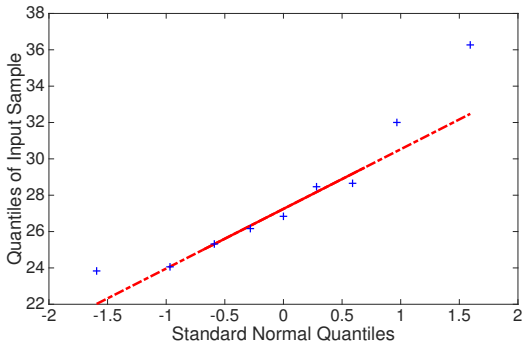
(a) London (b) Bristol

(c) Liverpool (d) Manchester

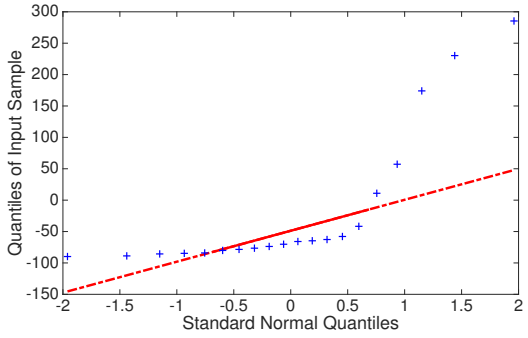
(e) Newcastle (f) Birmingham

(g) Sheffield

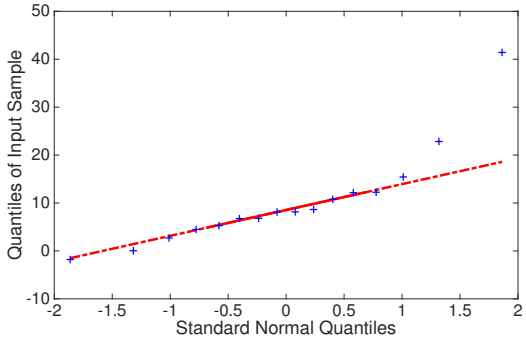
Figure A.2 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1952-1954



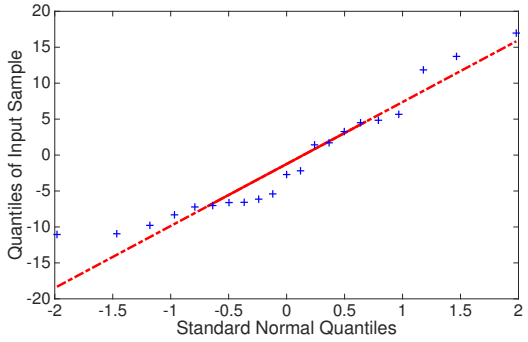
(a) London



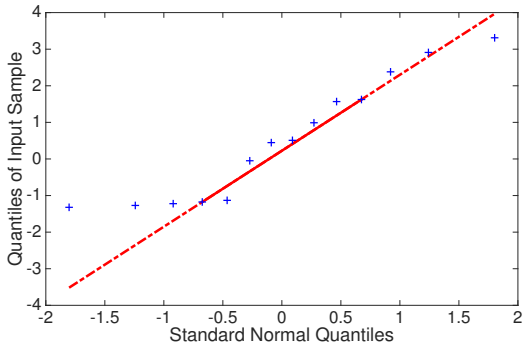
(b) Bristol



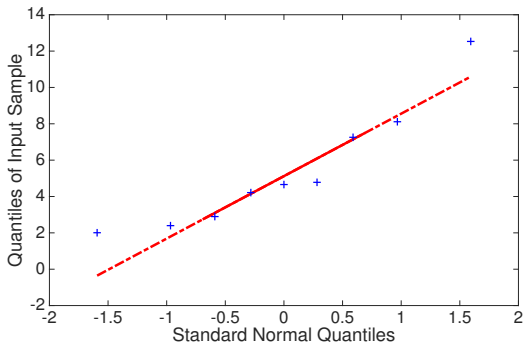
(c) Liverpool



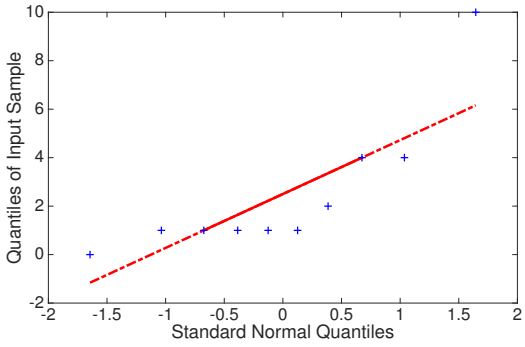
(d) Manchester



(e) Newcastle

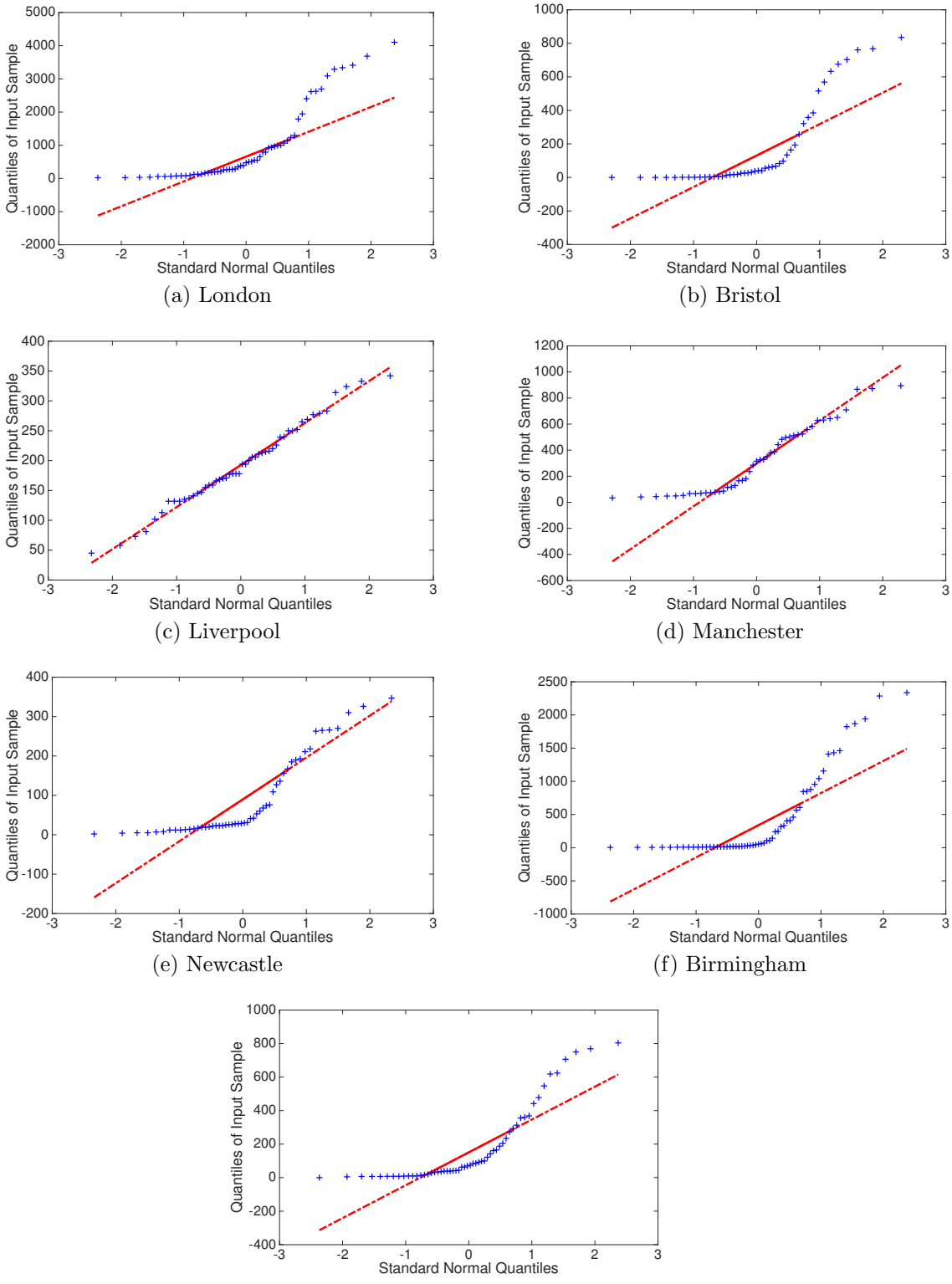


(f) Birmingham



(g) Sheffield

Figure A.3 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1954-1956



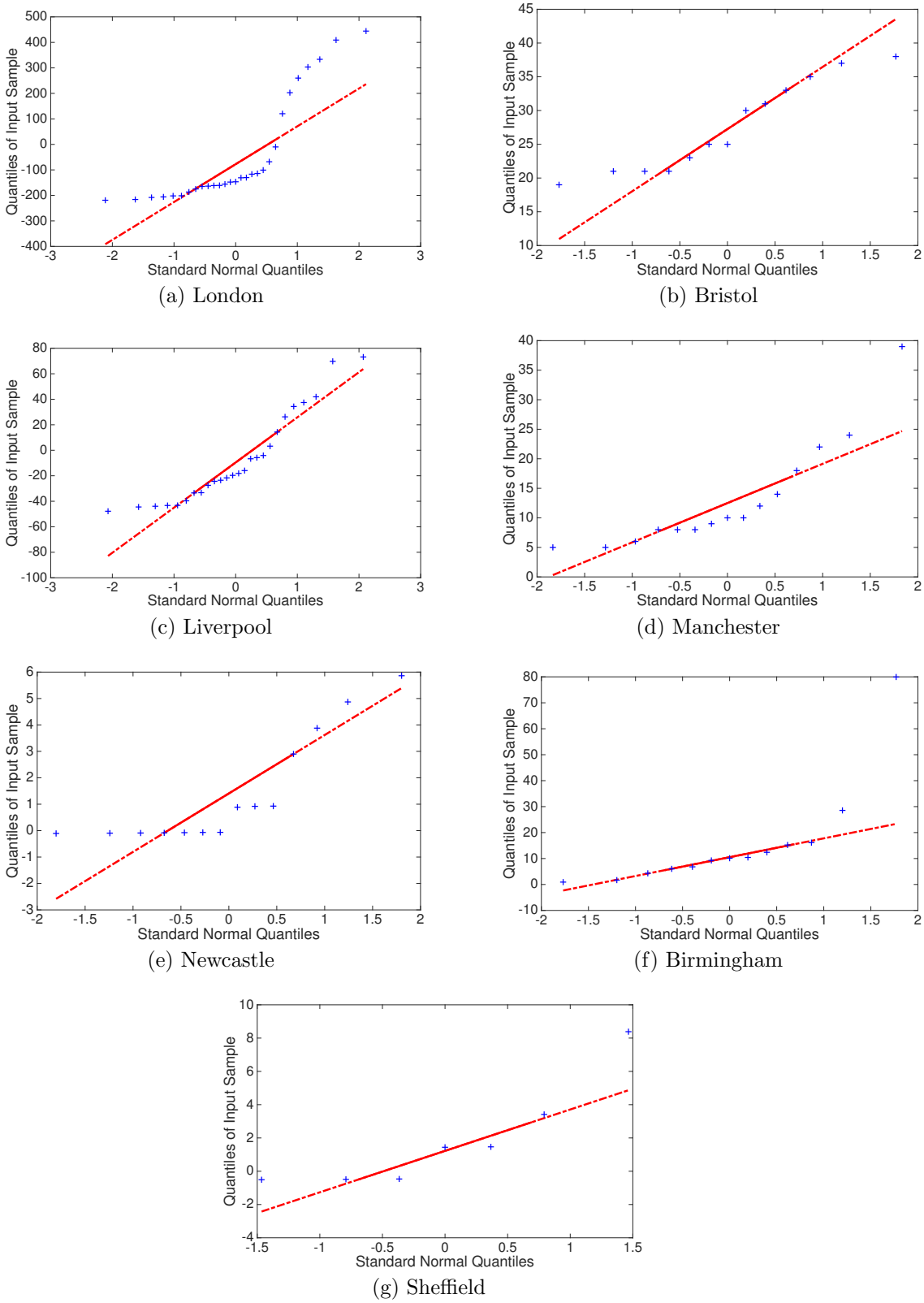
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.4 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1954-1956



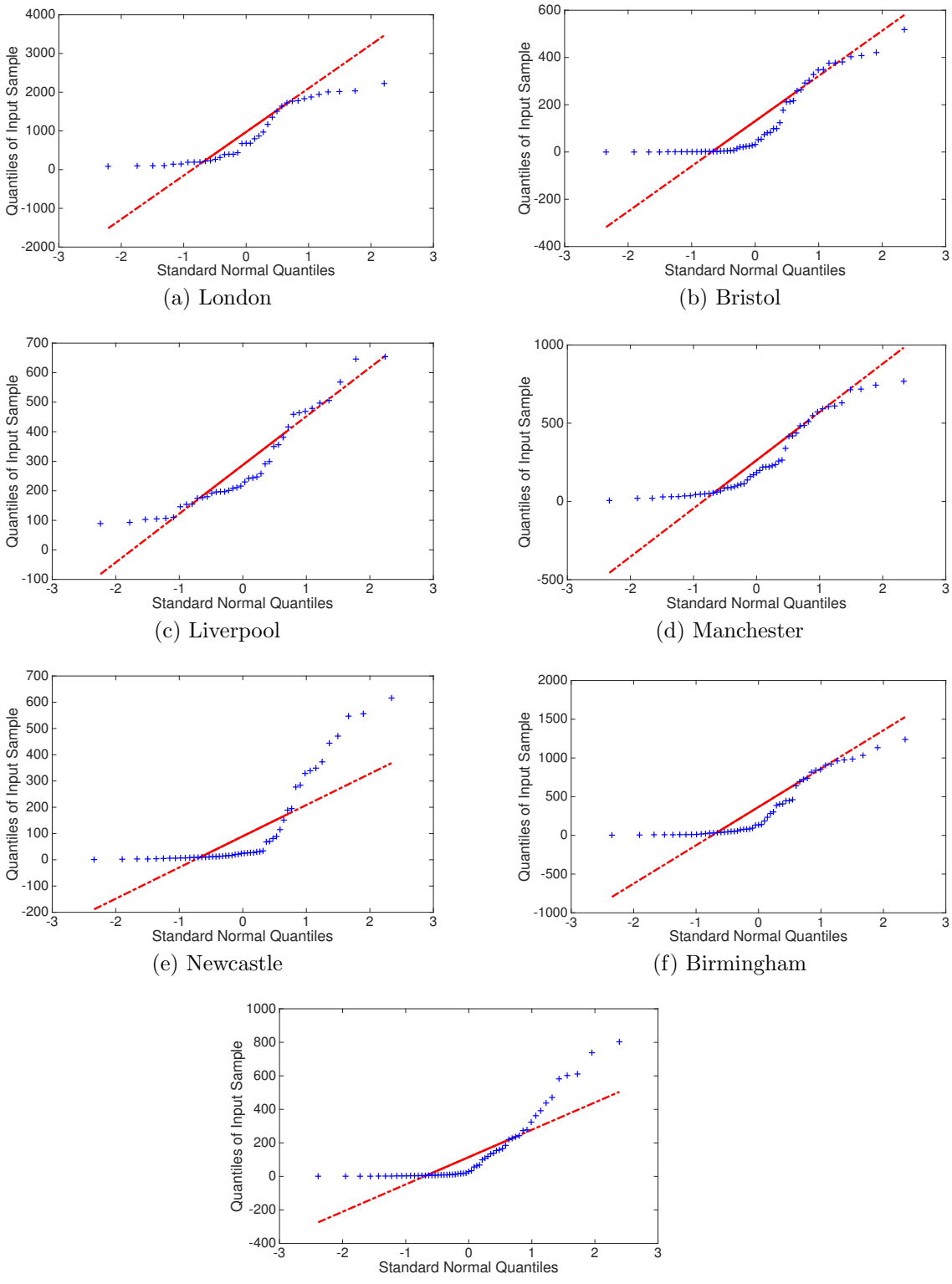
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.5 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1956-1958



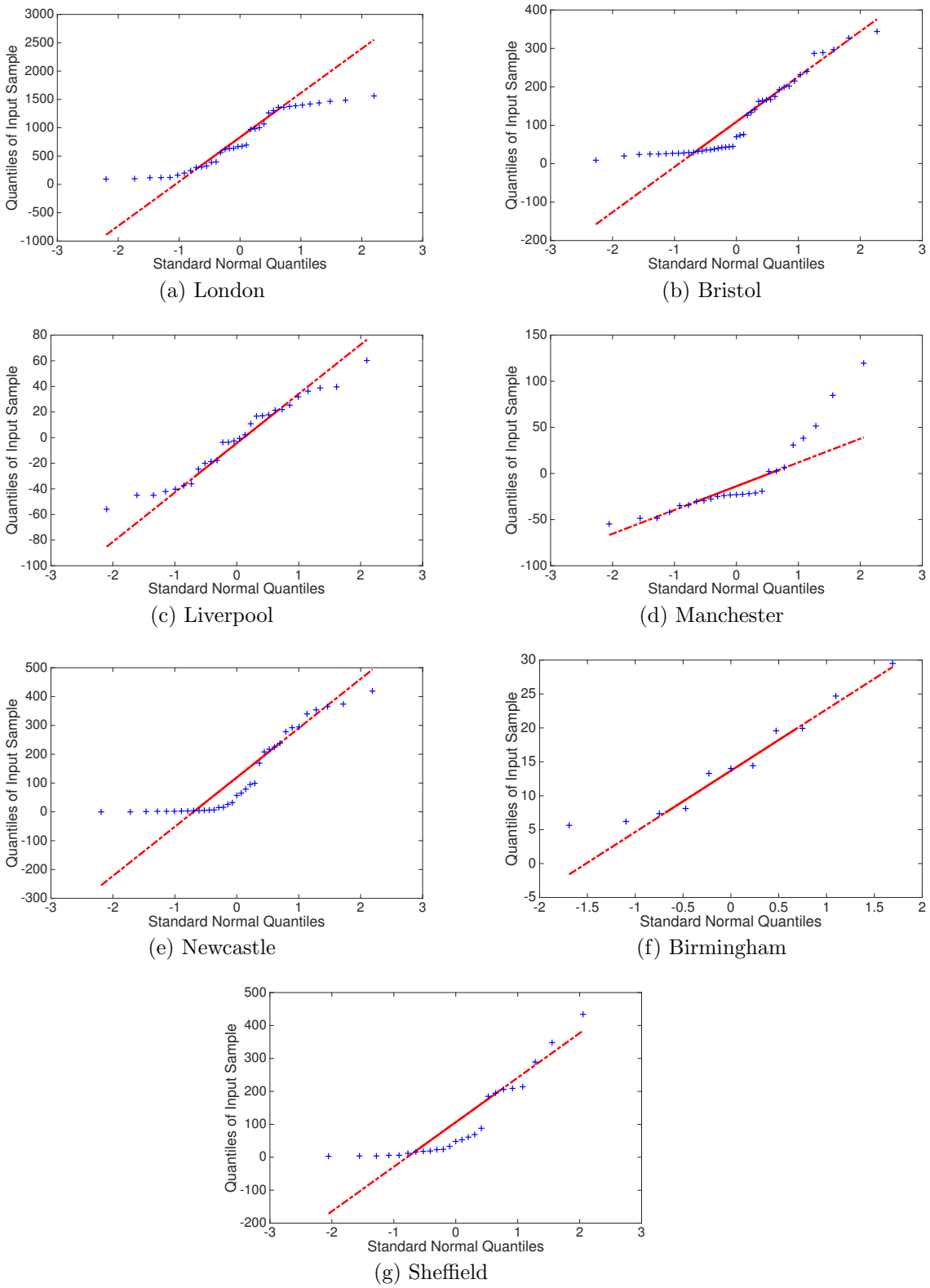
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.6 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1956-1958



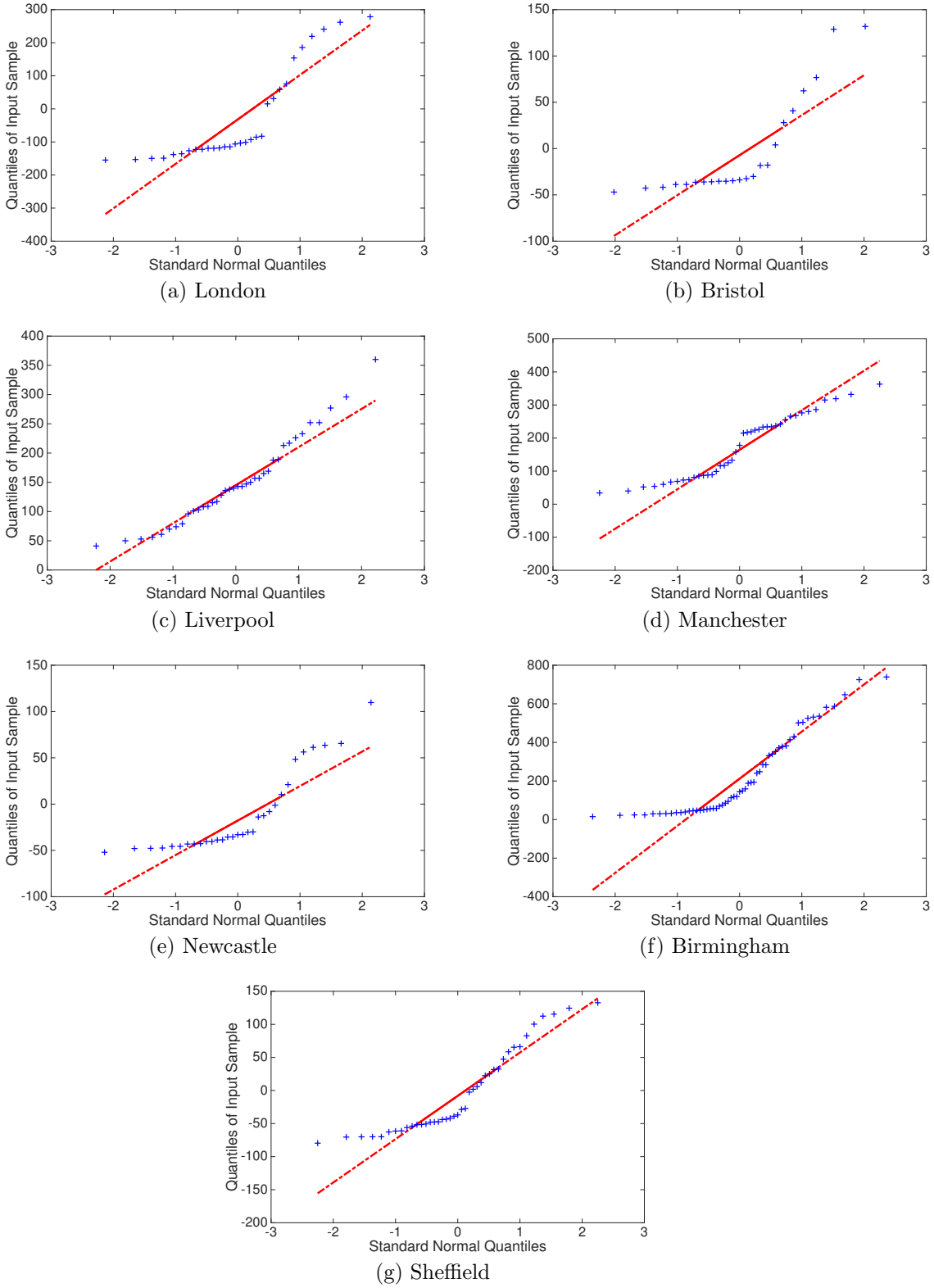
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.7 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1958-1960



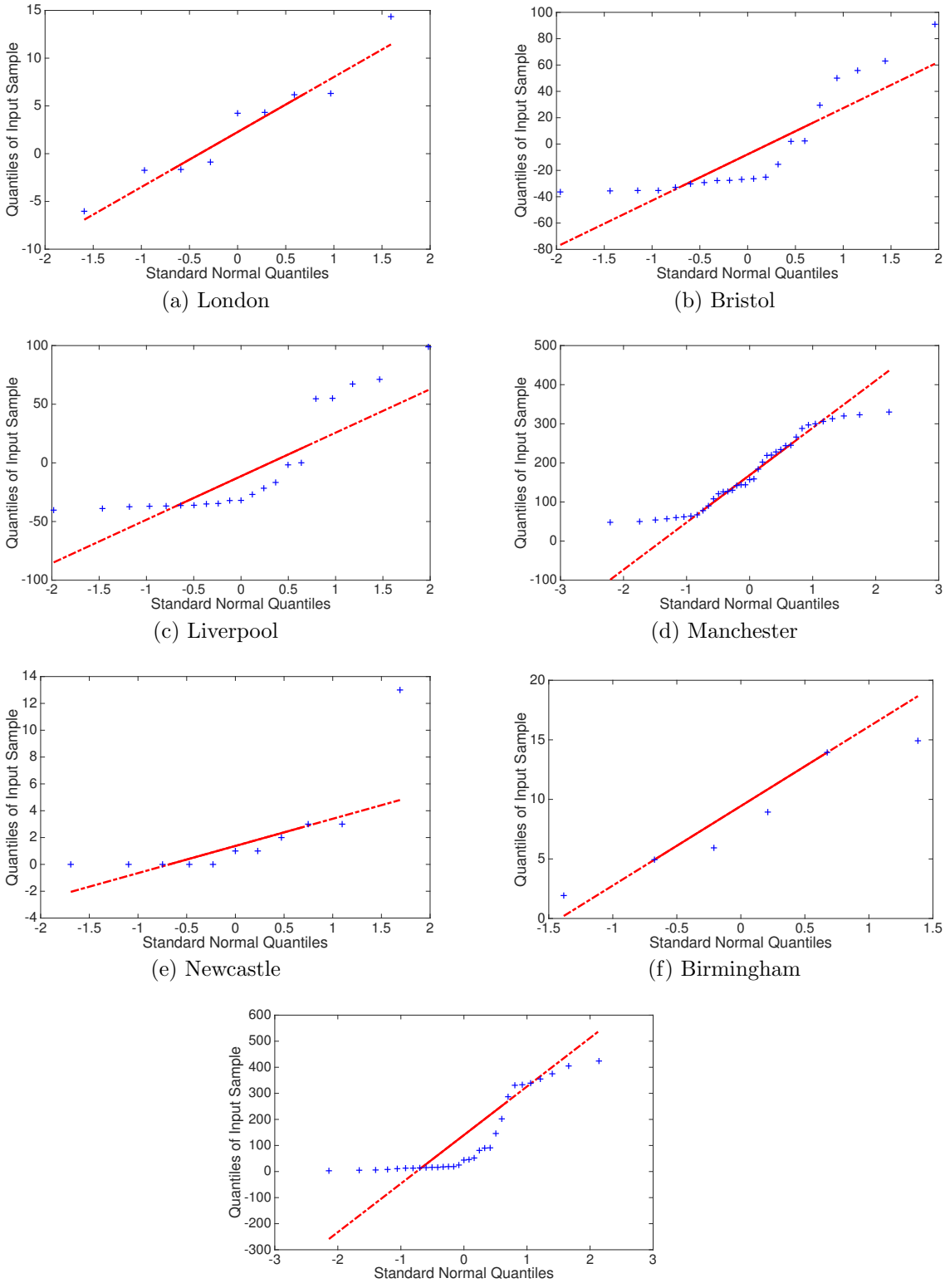
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.8 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1958-1960



(a) London

(b) Bristol

(c) Liverpool

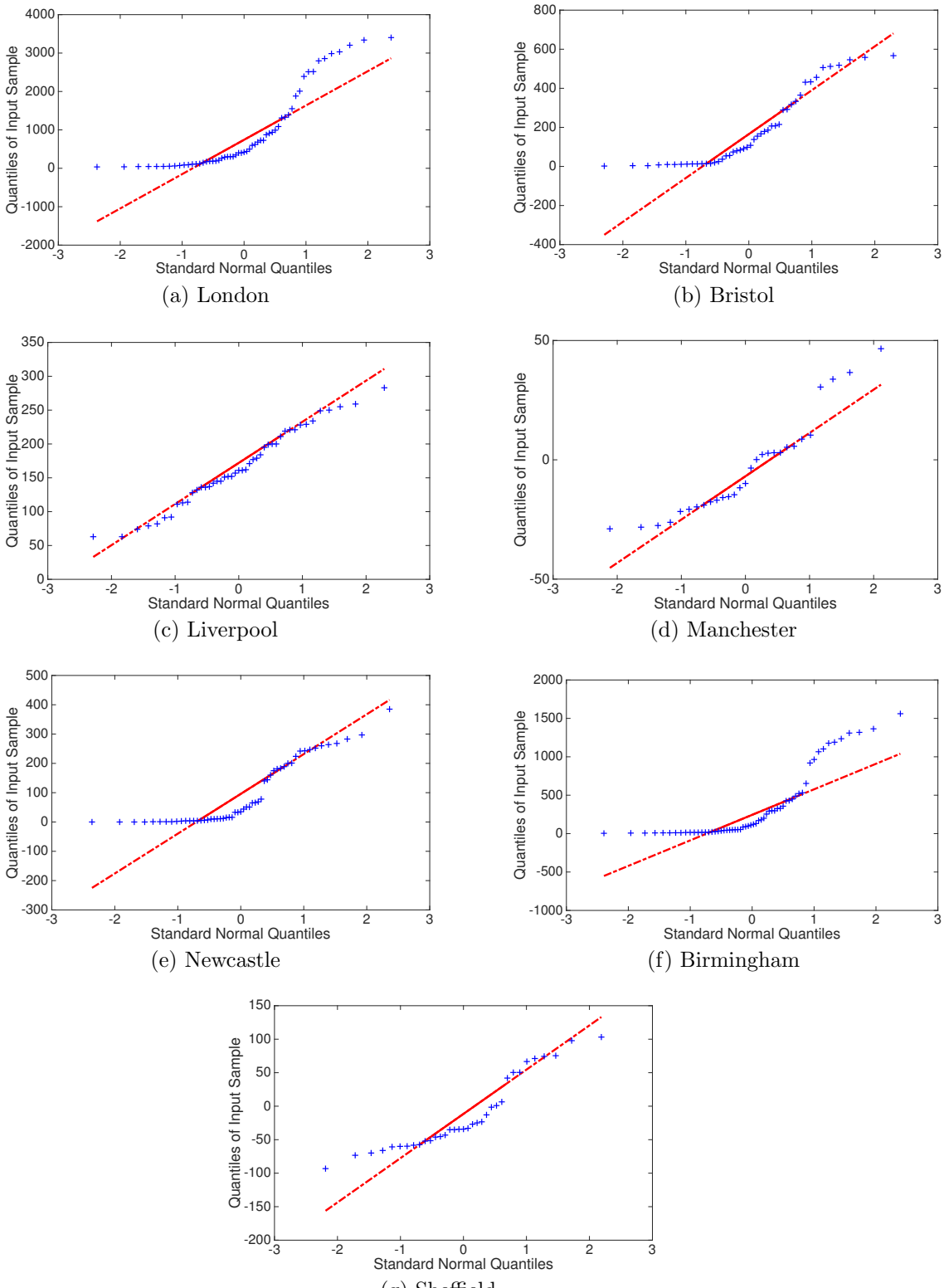
(d) Manchester

(e) Newcastle

(f) Birmingham

(g) Sheffield

Figure A.9 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1960-1962



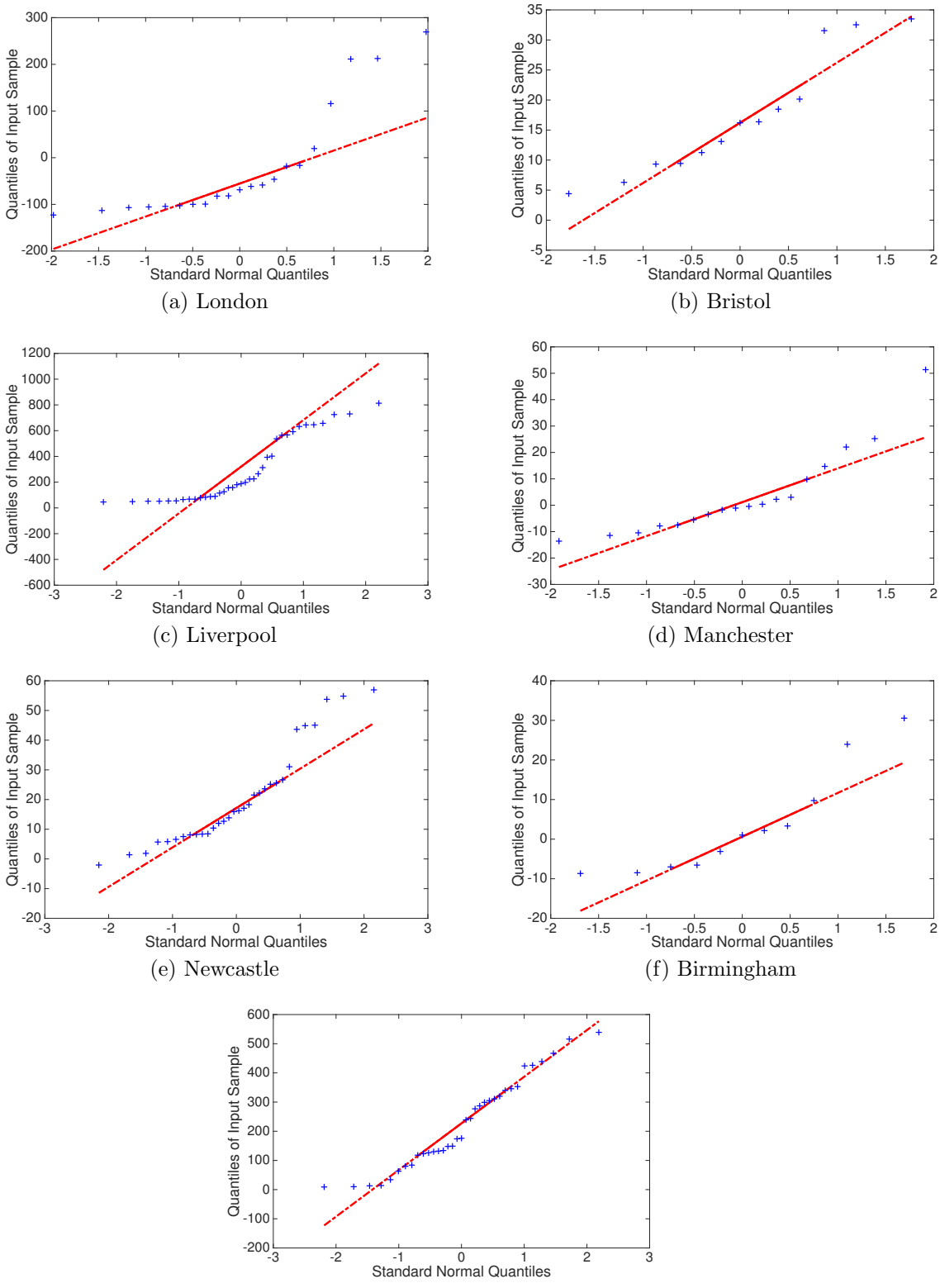
(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.10 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1960-1962



(a) London

(b) Bristol

(c) Liverpool

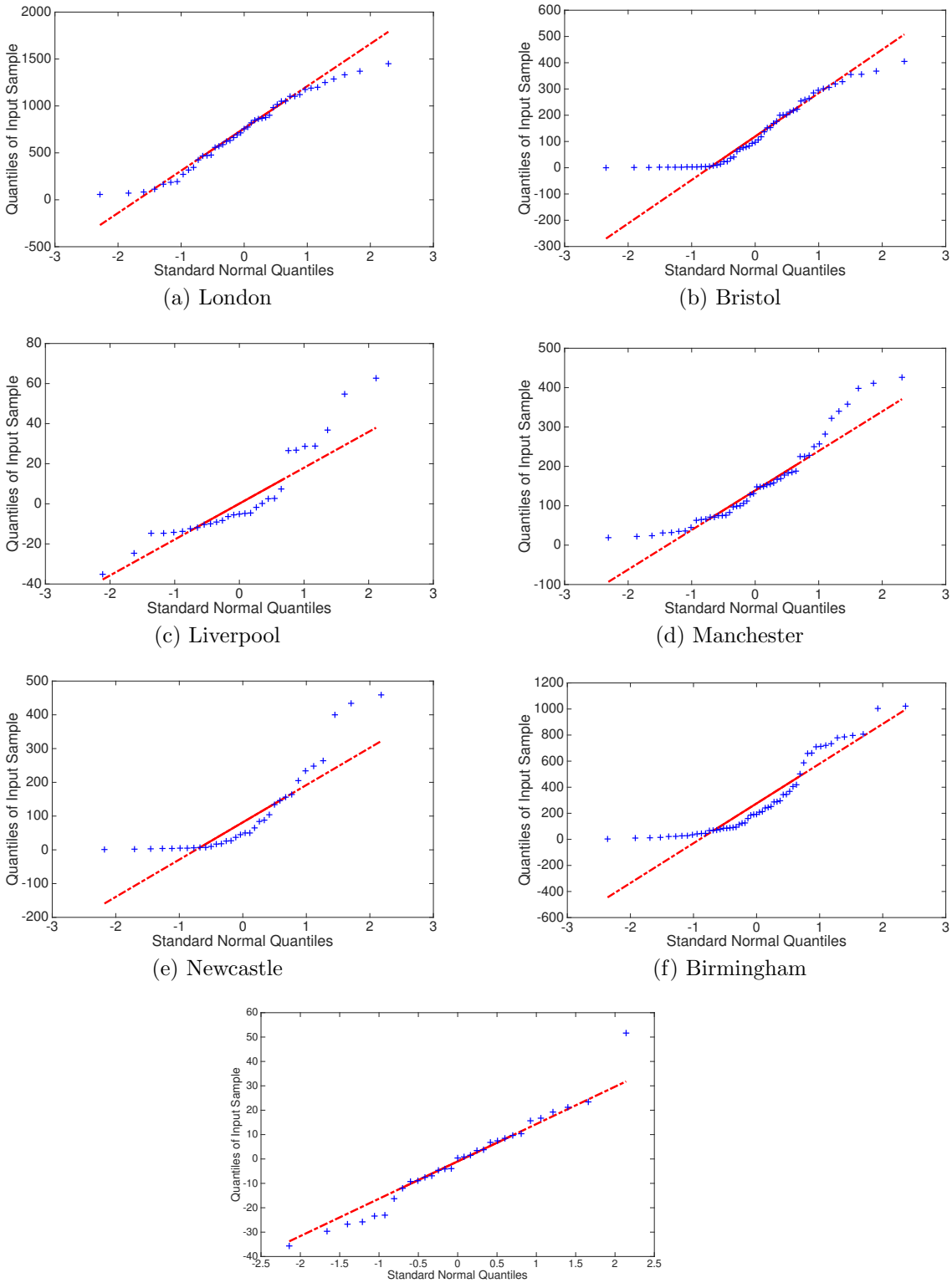
(d) Manchester

(e) Newcastle

(f) Birmingham

(g) Sheffield

Figure A.11 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1962-1964



(a) London

(b) Bristol

(c) Liverpool

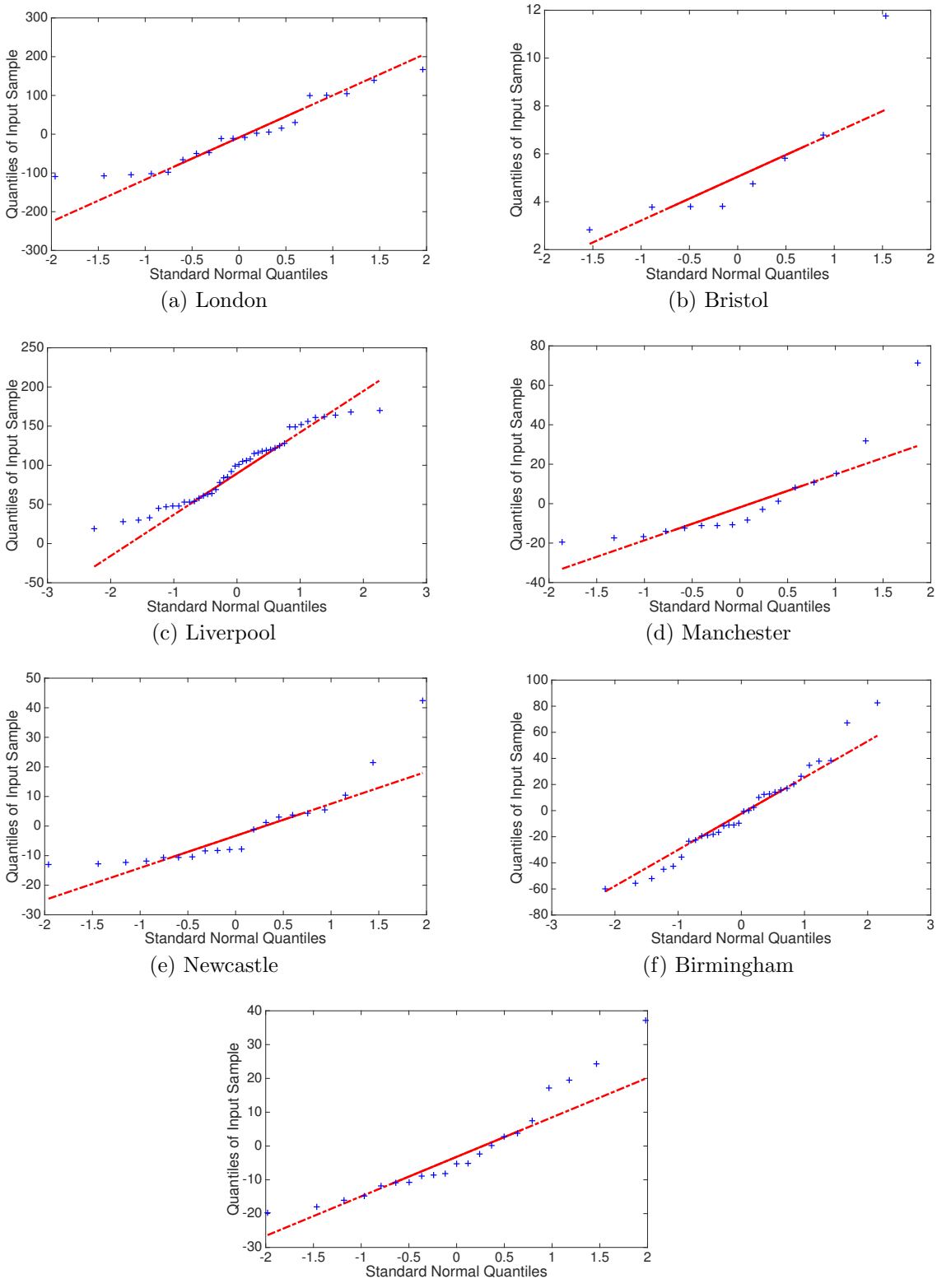
(d) Manchester

(e) Newcastle

(f) Birmingham

(g) Sheffield

Figure A.12 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1962-1964



(a) London

(b) Bristol

(c) Liverpool

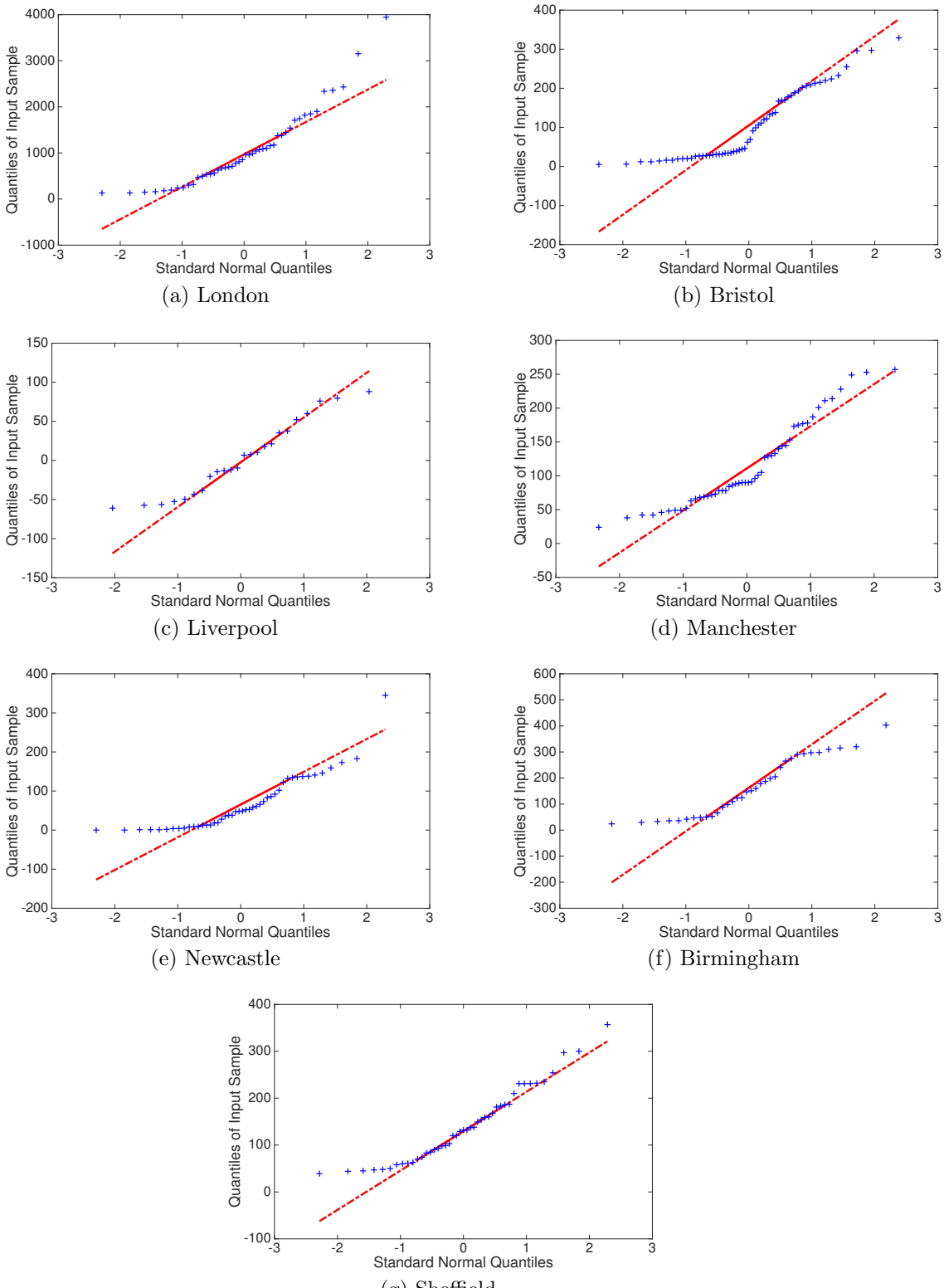
(d) Manchester

(e) Newcastle

(f) Birmingham

(g) Sheffield

Figure A.13 Quantile-quantile (qq) plot of the reported data before the individual infection time versus standard normal distribution in 1964-1966



(a) London (b) Bristol

(c) Liverpool (d) Manchester

(e) Newcastle (f) Birmingham

(g) Sheffield

Figure A.14 Quantile-quantile (qq) plot of the residual noise after the individual infection time versus standard normal distribution in 1964-1966