

Survival Theory Modelling for Information Diffusion

by

Akshay Aravamudan

Bachelor of Science
Computer Engineering
Florida Institute of Technology
2018

A thesis
submitted to the department of
Computer Engineering and Sciences,
Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Masters of Science
in
Computer Engineering

Melbourne, Florida
July, 2019

© Copyright 2019 Akshay Aravamudan
All Rights Reserved

The author grants permission to make single copies.

We the undersigned committee
hereby approve the attached thesis

Survival Theory Modelling for Information Diffusion
by Akshay Aravamudan

Georgios Anagnostopoulos, Ph.D.
Associate Professor
Electrical and Computer Engineering
Committee Chair

Jewgeni Dshalalow, Dr.rer.nat.
Professor
Mathematical Sciences

Adrian Peter, Ph.D.
Associate Professor
Engineering Systems

Philip Bernhard, Ph.D.
Associate Professor and Department Head
Computer Engineering and Sciences

ABSTRACT

Title:

Survival Theory Modelling for Information Diffusion

Author:

Akshay Aravamudan

Major Advisor:

Georgios Anagnostopoulos, Ph.D.

Information diffusion is the spread of information within a network. In this thesis, we model information diffusion as a survival process. We have adopted an existing algorithm called NETRATE for modelling information diffusion. This model involves finding the distribution of transmission time between two nodes in the network. We modify NETRATE's concave-down log-likelihood expression by adding partial parentage information and formulate an Expectation-Minimization (EM) algorithm to learn the parameters. We also describe a simulation scheme for NETRATE inspired by point process simulation strategies. Using the assumptions of the NETRATE model, we derive a method to model popularity as a function of time. In order to showcase the insights offered by NETRATE, we explore a real-world example involving two kinds of software vulnerabilities: Exploited and non-exploited vulnerabilities. Finally, We derive a scheme for transforming infection times so that a goodness of fit test can be performed using Kolmogorov-Smirnov (KS) test statistic.

Table of Contents

Abstract	iii
List of Figures	viii
List of Tables	xii
Abbreviations	xiv
Notations	xv
Acknowledgments	xvi
Dedication	xvii
1 Introduction	1
1.1 Contributions	6
2 Related Works and Literature Review	8
2.1 Survival Theory Modelling	8
2.2 Point-Process based Modelling	10
2.3 Hybrid Modelling Techniques	14

3	Background	15
3.1	Notation	15
3.2	Information Diffusion	16
3.2.1	Stochastic Point Processes	16
3.2.1.1	The Intensity Function λ	17
3.2.1.2	The HAWKES Point Process	18
3.2.2	Survival Analysis Fundamentals	19
3.2.2.1	Survival Function	19
3.2.2.2	Hazard Function	20
3.2.3	Kernels	24
3.2.4	On the Incompleteness of Data	29
3.2.5	Cascade Definition	31
3.3	The NETRATE Information Diffusion Model	32
3.3.1	More on NETRATE nodes' Conditional Intensity Functions (CIFs)	34
3.3.2	NETRATE nodes' CCIFs	35
3.3.3	NETRATE nodes' CPDFs, CCDFs & CSFs	36
3.3.4	NETRATE Likelihood	39
3.3.4.1	The First Log-Likelihood Term	49
3.3.4.2	The Second Log-Likelihood Term	50
3.3.4.3	The Third Log-Likelihood Term	51
3.3.4.4	The Fourth Log-Likelihood Term	53
3.3.4.5	Final Form of Log-Likelihood	54
4	NETRATE Training	58
4.1	Regularized Likelihood	59

4.2	Regarding Gradient Method	62
4.3	EM Algorithm	63
4.3.1	EM Setup	65
4.3.2	EM updates	65
4.4	Evaluating the Log-Likelihood	65
4.4.1	Some Observations	69
4.5	Experiment: Effect of Parentage Information	69
5	Time Rescaling and Goodness of Fit	71
5.1	Regarding Goodness of Fit for Large Samples	75
6	Simulation	77
6.1	Simulation Scheme	79
6.2	Case study: Goodness of Fit for Simulation	79
7	Predicting Dynamic Virality	82
8	Real World Data: Software Vulnerabilities	87
8.1	Software Vulnerability Data	87
9	Results: NETRATE for software vulnerabilities	92
9.1	Common Vulnerabilities and Exploit (CVE): Exploited	92
9.2	CVE: Non-Exploited	95
10	Conclusions and Future Work	98
	References	101

A	Proofs & Derivations	107
A.1	Proof of Proposition 1	107
A.2	Proof of Proposition 2	109
A.3	Proof of Proposition 3	110
A.4	Proof of Proposition 4	111

List of Figures

1.1	Influence of real world networks on popularity w.r.t stocks and political candidacy.	2
3.1	Graph used for illustrative example. The transmission rates are present as wedge weights.	40
3.2	Example of a cascade on the example network. The dotted lines indicates possible parentage for infection events, i.e which node can infect the given node.	40
3.3	Intensity functions $\lambda(t)$ for nodes B,C and D. After infection of each node, their respective intensities go to zero.	41
3.4	Integrated intensity functions $\Lambda(t)$ for nodes B,C and D. After infection of each node, their respective integrated intensities remain constant.	42
3.5	The lifetime distribution $f(t)$ for nodes B,C and D after the infection of node A.	43

- 3.6 A visual depiction of a cascade c contributing to the first term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_1(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_1(i)$. For such type of cascades, i 's infection is observed but not as the first infected node. Furthermore, no parent node information is taken into account. 50
- 3.7 A visual template of a cascade c contributing to the second term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_2(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_2(i)$. The red arrow conveys that one of the nodes j is the observed parent of i in this cascade. Again, for such type of cascades, i 's infection is observed but not as the first infected node. 52
- 3.8 A visual template of a cascade c contributing to the third term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_3(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. In this case, i 's infection has been observed. Highlighted in green are the node indices that will belong to $\mathcal{J}_3(i)$. The red arrow conveys that i 's parent has not been observed in this cascade. This necessarily means that $j_1^c, j_2^c \neq i$, *i.e.*, it was neither the first or second infected node in the cascade. . . . 53

3.9	A visual template of a cascade c contributing to the second term of the i^{th} sub-problem's log-likelihood, <i>i.e.</i> , $c \in \mathcal{C}_4(j i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_4(i)$. This is a cascade, where i 's infection is not observed by the right-censoring time T	54
4.1	P-P plot generated using transformed times from simulation when there are no survived nodes in all cascades. Generated using 1000 data points selected at random	70
6.1	P-P plot generated using transformed times from simulation when there are no survived nodes in all cascades. Generated using 1000 data points selected at random	81
6.2	P-P plot generated using transformed times from simulation when cascades contain some survived nodes. Generated using 1000 data points selected at random	81
7.1	Probability Mass Function (PMF)s generated for future infections as we consider n infections have already happened in the cascade. .	86
8.1	Example of an event from twitter that contributes to the CVE cascade	89
8.2	Example of an event from Reddit that contributes to the CVE cascade	89
8.3	Example of an event from Github that contributes to the CVE cascade	90
9.1	Inter Platform average transmission rates for Exploited CVEs. The average rate was obtained by looking only at those edges that go from one platform to another. (G - Github, R - Reddit, T-Twitter)	94

9.2	Intra Platform average transmission rates for Exploited CVEs.	
	The average rates were obtained by observing edges that stay within a platform. (G - Github, R - Reddit, T-Twitter)	94
9.3	Inter Platform average transmission rates for Non-Exploit CVEs.	
	The average rate was obtained by looking only at those edges that go from one platform to another. (G - Github, R - Reddit, T-Twitter)	96
9.4	Intra Platform average transmission rates for Non-Exploit CVEs.	
	The average rates were obtained by observing edges that stay within a platform. (G - Github, R - Reddit, T-Twitter)	96

List of Tables

3.1	Various Memory Kernels (MKs) with their corresponding Integrated Memory Kernels (IMKs) and the resulting Probability Density Functions (PDFs). As a reminder, all above quantities are defined for $t \geq 0$, except for the Power Law case, where they are defined for $t \geq \beta$. All β 's are scale parameters of the corresponding PDFs, while all γ 's, as well as δ , are shape parameters of the corresponding PDFs.	26
3.2	What terms each cascade type contributes to the log likelihood according to (3.72) for the i^{th} sub-problem. Indices of nodes are depicted as cells in an array and are assumed sorted in chronological order of infection. Red arrows indicate parentage information and a parent node index of u reflects an unknown parent. Node indices highlighted in green reflect the set of j indices summed over in the corresponding expression of the right column.	55
9.1	Validation Likelihood of kernels: Exploited CVEs	93
9.2	Average incoming and outgoing transmission rate: Exploited CVEs	93
9.3	Validation Likelihood of kernels: Non-exploit	95
9.4	Average incoming and outgoing transmission rate: Non-Exploit	97

List of Algorithms

1	NETRATE EM Setup	66
2	NETRATE EM Updates	67
3	NETRATE Log-Likelihood	68
4	Infection Time Transformation	75
5	NETRATE Simulation	78

List of Abbreviations

The abbreviations showed below apply to the Netrate process and its application to the data

CVE	Common Vulnerabilities and Exposures
w.r.t	with respect to
API	Application Programming Interface
EM	Expectation Maximization
PDF	Probably Density Function
PMF	Probability Mass Function
RV	Random Variable
IID	Independent, Identically Distributed
K.S	Kolmogorov Smirnov
CIF	Conditional Intensity Function
CCIF	Coniditional Cumulative Intensity Function
CSF	Conditional Survival Function
CPDF	Conditional Probability Density Function
CCDF	Conditional Cumulative Density Function
MK	Memory Kernel
IMK	Integrated Memory Kernel
EM	Expectation Maximization
SI	Susceptible Infected
I-SP	Intensity Survival Probability
FMOI	Finite Mixture of Intensities
P-P	Probability – Probability
CI	Confidence Interval

List of Symbols/Notations

The Notation used in this paper are described below:

$\lambda(t \mathcal{H}_t)$	Conditional Intensity of process
$\boldsymbol{\lambda}$	Vector of intensities of multiple processes
$\Lambda(t)$	Cumulative intensity of process
\mathbb{R}	set of real numbers
\mathbb{R}^N	cartesian set products of real number sets
\mathbb{R}_+	set of postive real numbers
\mathbb{R}_+^N	cartesian set products of positive real number sets
$\llbracket \text{predicate} \rrbracket$	evaluates as 1 if predicate is true and 0 otherwise
$S(t)$	Survival function
$F(t)$	Cumulative density function
$\phi(t)$	Memory kernel
$\psi(t)$	Integrated Memory kernel
\mathcal{N}	set of all nodes in a network
a_{ji}	transmission rate from node j to i
$\mathcal{I}^c(t)$	Set of infected nodes in a cascade
\wedge	minimum over set of values
$pa(T)$	parent node of infection taking place at T
ν	regularizer
$p_{j,i}^c$	probability that j is parent node of i in cascade c

Acknowledgements

I want to thank Dr. Georgios Anagnostopoulos for providing the topic of my thesis as well as for future research. His notes, advise and enlightening conversations formed the foundation of this thesis. I also thank Xi Zhang for her interesting conversations related to point processes as well as experiences with her own implementation which I found invaluable to my thesis. This work has been supported by Defense Advanced Research Projects Agency under grant number FA8650-18-C-7823. This has also been supported by "Deep Agent: A Framework for Information Spread and Evolution in Social Networks", and Walt Disney Attractions, LLC (in collaboration with UCF), "Disney-UCF Life Long Learning Program: Deep Learning Intrapreneurship Pilot".

Dedication

Dedicated to Harish, mom and dad for their patience and understanding in my pursuit of higher education.

Chapter 1

Introduction

Information diffusion is the process of information propagating to individuals or groups within a network. An episode of information diffusion keeps track of "to whom" and "when" the information reaches. These diffusion episodes are referred to as information cascades. Information cascades are nuggets of data that include metadata about the hops of a single piece of information within a particular platform, or even across platforms. This information could be a tweet, a paper published in a journal or even a Facebook post. The hops in these relevant domains could manifest as a retweet by another user, a citation by another publisher and a comment to the Facebook post respectively. There could also be cascades that spread information across multiple platforms such as from Twitter to Facebook, Reddit to Twitter..etc. These information cascades are rich in terms of the insights they can provide. However, modelling the network on which these information cascades operate using an information diffusion model helps us get a lot more insights about the network and its actors. Information diffusion models take information cascades as inputs and produce an output that helps us extract useful information

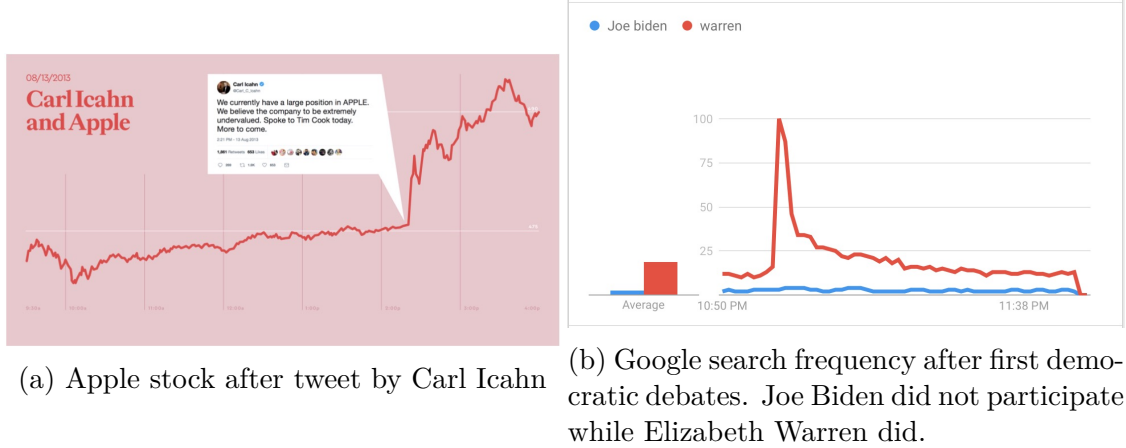


Figure 1.1: Influence of real world networks on popularity w.r.t stocks and political candidacy.

about the network and the actors within them.

Modelling networks gives us valuable information about how each actor in the network interacts with other actors. This information could be useful to tasks like marketing campaigns for advertising products and mitigation of fake news. Finding out users who create and share fake news is often a trivial task because it is a never-ending endeavor. However, finding users with high influence in their respective information context is useful. These users act as a bottleneck within the network due to their large influence over other entities in the same network. This concept of finding the influential users/nodes translates to a variety of applications. For instance Figure 1.1a shows how the Apple stock value increases after a tweet by Carl Icahn, a well-known investor. In this situation, it could be valuable to find out the network over which this tweet propagated. This could help us preemptively figure out if Apple stock will show significant change. Figure 1.1b show the increase in Google searches after the first democratic primary debates. In such a situation, the Warren campaign might be interested in knowing the network

which contributed to this temporary increase in popularity. This can help them target people for fund-raising purposes. This could also be useful for national security purposes, journalism, stock-market investments, operating system security and so on. For instance, a stock investment firm might be interested in knowing how influential a negative campaign against a stock might be. This will help them determine if there is adequate reason to sell the stock at an earlier stage to mitigate potential losses.

The literature on information diffusion is built on the graph inference problem, whether it is obtaining a social network or an influence graph. Gomez et al. [1] proposes NETRATE: a probabilistic infection model where inferring the network is a consequence of estimating the parameters. Farajtabar et al. propose using a point process model [2] to jointly evolve the network as well as the information diffusion process by modelling them as a HAWKES process and a hybrid of HAWKES and survival process. [3] models the spread of cascades in a social network as a multidimensional marked point process. Based on the time period of the available data, we can also infer dynamic networks by solving a network inference problem for cascades within every consecutive time period. This provides insights on how the network edges evolve over time. This is especially relevant for news articles to observe how public perception has changed, or even to observe the attention span of the general public in relation to a news topic. We have included more relevant papers in chapter 2.

We have examined one specific model for information diffusion namely the NETRATE model. The NETRATE model is a survival based model for information diffusion which treats the concept of information spreading as infections. Although still falling under the broad category of point process based models [4], the NE-

TRATE model is characterized by its base assumption that nodes can only be infected once in a cascade. This assumption is driven by the fact that we are more interested in whether a user participates in a cascade, hence a survival theory approach is more appropriate. Relaxing this assumptions leads us to another class of point process based models which is best captured by the HAWKES model. We will touch on the HAWKES model to see how it compares with NETRATE. Another important distinction of the NETRATE model is that it doesn't explain the first infection in the cascade, i.e. it only deals with infections after the first node is infected. NETRATE uses a probabilistic distribution to model the infection times between two nodes in a cascade, assuming one node is already infected. It is treated as a survival process wherein the infection time between two nodes is represented as a probability distribution which is a function of the memory kernel $\phi(.)$, the integrated memory kernel $\psi(.)$ and the transmission rate a_{ji} .

$$f(t_i|t_j) = a_{ji}\phi(t_i - t_j)e^{-a_{ji}\psi(t_i-t_j)} \quad (1.1)$$

For the NETRATE model, once the parameters are estimated, Gomez Rodriguez et al. [5] simulate the cascades by using an inversion method. We here will also do a similar simulation, but without using an inversion technique. We use an efficient simulation algorithm that is inspired by Ogata's modified thinning algorithm [6] for the HAWKES point process. With a recent trend towards popularity prediction techniques, especially in the context fake news, there have been a lot of attempts to appropriately model these results using an information diffusion model. There are a variety of popularity prediction techniques associated with the HAWKES model. [7] for instance uses a single HAWKES point process to model the popularity. They attempt to find the final number of re-shares of a given post which is based on a

metric they define called the infectiousness parameter. [8] comes up with a closed form solution for the expected number of forwarding of a micro-blog within a time t . We will come up with something similar along the lines of characterizing the dynamic virality of a cascade.

To further demonstrate the NETRATE model, we run it on a dataset of software vulnerabilities. We use existing CVEs and see how they traverse across three platforms: Reddit, Twitter and Github. This helps us investigate the role of different platforms in the life-cycle of software vulnerabilities. To contrast two differing CVE types, we divided the dataset into 2 types: exploited and non-exploited. Exploit CVEs are software vulnerabilities which malicious actors have taken advantage of. Naturally, these CVEs have a higher risk. So, there is a need to understand how these two types of CVEs propagate and how it contributes to access violations by hackers. In both cases, we perform parameter estimation to draw some conclusions regarding the role of these platforms in the propagation of information regarding software vulnerabilities.

The thesis is organized as follows. Chapter 2 provides a view of the existing and related works that is relevant to this thesis as well as in understanding the NETRATE model. Chapter 3 provides some background information regarding point processes, survival models and eventually derive the log-likelihood expression for the NETRATE model. Chapter 4 discusses the procedure for learning the parameters using an EM algorithm. Chapter 8 and 9 cover the examples dealing with exploited and non-exploited CVEs. The time transformation for goodness of fit is discussed in chapter 5. The simulation strategy is covered in chapter 6. Finally, the method for predicting dynamic virality and its analysis is discussed in chapter 7.

1.1 Contributions

Firstly, this thesis aims to investigate the NETRATE information diffusion model in more detail than has been done in its seminal and companion papers [1]. For example, during this endeavor, we layout the probabilistic assumptions and rules that govern the diffusion dynamics of contagions over networks according to NETRATE. Unlike NETRATE’s introduction in its seminal paper, this thesis frames NETRATE as a multi-variate point process. A concrete contribution that is made in this thesis is the extension of the standard NETRATE model to one that leverages potentially knowing the source/cause of a node’s infection. Indeed, in some online social media settings, such information may be (perhaps, occasionally) available. For example, a user B’s reply to a user’s A message may indicate that B’s response was instigated by A’s stimulus. Real-world examples of this are comments to users’ posts in Reddit and replies to users’ tweets in Twitter. Rather than discarding or ignoring such infection parentage information¹, utilizing it for training a model is reasonably expected to increase the quality of the resulting fit, while at the same time improving inference time. With this new element in mind, we eventually derive the log-likelihood expression for this modified model’s parameters given training observations. In terms of parameter estimation, we oriented ourselves towards an EM-based training algorithm that is suitable for inferring parameter values of large to very large networks. In order to be able to address such large scale problems with efficiency, our implementation first subdivides NETRATE’s training into independent sub-problems and, secondly, pre-computes and

¹In future chapters we will say that node (user) i ’s parent node is j , if it has been observed that j ’s infection caused i ’s infection.

appropriately stores certain quantities required for optimizing each sub-problem. The former feature allows for the training to be performed in a computationally distributed manner.

Moreover, inspired by the random time change theorem for point processes [9], we derive a similar time transformation result for our modified NETRATE model, which enables us to perform a goodness of fit test. We also describe a simulation scheme for our NETRATE model that is based on a well-known thinning method rather than the inversion method previously applied to such survival models [5]. Finally, we present a method to compute the PMF of the excess number of infections based on the prior observed evolution of a contagion. This allows us to make predictions about the number of future infections within a given forecast horizon. Such capability is important for tasks such influence maximization and popularity prediction.

Chapter 2

Related Works and Literature

Review

Information diffusion has become an increasingly relevant endeavor over the years. It has applications in a variety of situations; Whether it is the need to mitigate fake news, malicious users or even propaganda, there is an inherent need to understand the network over which these communications take place. In recent years, there have mainly been three broad categories of papers in literature that have dealt with this process. Survival models, Point process models and hybrid techniques. Although the three categories are not mutually exclusive [[10], [4]], this distinction helps us understand their approach. Below we mention some of the relevant papers.

2.1 Survival Theory Modelling

Survival Theory models consider the transfer of information as a survival process wherein the transfer of information follows a Susceptible Infected (SI) behavior;

Once a node has been infected, it cannot be susceptible to another infection forever in that diffusion episode. Most of these approaches involve solving a graph inference problem.

Kempe et al introduced the independent cascade model (ICM) [11] which provides the foundations for all the other models described below. ICM works on the philosophy that when a node becomes active, it has a single chance of activating its neighbors with a given probability. Gomez-Rodriguez et al proposed NETINF [12] in which they "recover the unobserved network structure by exploring correlations in node infection times". They incorporate a feature vector as a part of the information diffusion process. They build upon the ICM to describe information diffusion using a probabilistic generative model. They propose an efficient algorithm for inferring a near-optimal set of k edges since maximizing the likelihood to find the optimal set k is an NP-hard problem. However the transmission rate between nodes is assumed to be fixed.

Myers et al introduced CONNIE [13] wherein they infer the connectivity along with a prior probability for infection. Although their formulation is mainly applied to SI models, where if once a node is infected, it stays infected forever, they claim that their model is robust in the sense that it is applicable to both Susceptible-Infected-Susceptible (SIS) and Susceptible-Infected-Recovery (SIR) models as well. In SIS models, once a node is infected, it can become susceptible to infection again in the future. However in SIR models, once a node is infected, it can recover and then never get infected again. This interpretation depends on the context of the data and the platforms over which it propagates.

NETRATE was proposed by Gomez-Rodriguez et al. [1]. This is done by formulating an optimization problem where the likelihood of a node A infecting

another node B is modelled by a PDF. like NETINF, they use a probabilistic generative model with each node-node pair having its own transmission rate α_{ji} . They have to make parametric assumptions about the nature of transmission of information between nodes. They claim that their models naturally impose a sparsity constraint as opposed to forcing it by using an ℓ_1 regularizer (also referred to as lasso regression) as in NETINF and CONNIE.

Gomez-Rodriguez et al [5] proposed to use a survival model for information diffusion, much like NETRATE. They treat information spread as a contagion, again just like the NETRATE model. They attempt to learn the hazard function of the survival model based on a predetermined shape (hazard function described in 3). More specifically, they propose an additive and multiplicative model for the hazard function. They perform cascade size prediction and cascade duration by performing a simulation of the survival process using the inversion method to calculate the subsequent infection times. They compare their simulation results for both the additive and multiplicative models by comparing their Conditional Cumulative Density Function (CCDF) with the real cascade data.

2.2 Point-Process based Modelling

Point-process based models treat the information diffusion process as a renewal process. In such processes, an infected node can become susceptible to infection again. So these fall under the category of SIS models. However, if this assumption is relaxed, it still falls under the category of point processes [10]. This has found a lot of applications in a social network setting due to the concept of re-infections. Most of the works presented below use the HAWKES process or the Poisson process in

some in some form.

Yang et al. [10] model the viral diffusion process of memes by simultaneously addressing diffusion network inference and context tracking by using a mixture of HAWKES processes. SEISMIC [7] develops a statistical model based on the HAWKES process to predict final size of cascades. They estimate the intensity associated with a single cascade as a doubly stochastic process by incorporating an infectiousness parameters that is related to the number of followers of individual users, thus incorporating network structure.

Farajtabar et al. [2] proposed COEVOLVE: a joint process model for information diffusion and network evolution. They merge the two processes. They consider twitter as the main example. They model the information diffusion process as a multivariate HAWKES process and the link creation process as a combination of survival and HAWKES process. They also propose an efficient simulation similar to Ogata’s simulation algorithm for both the information diffusion as well as networks evolution process. They also go on to predict links and activity which performed better than their baselines. They perform model checking to transform the times in order to compare the quantiles of the model to the quantiles of the time change theorem.

[14] learn the causal influence in a network of multivariate linear HAWKES processes. They claim that the estimation of the support of the excitation matrix is sufficient to learn the associated directed information graph (DIG). They come up with a minimal generative model which is equivalent to the causal relationships implied by the excitation matrix. They use a numerical method to learn these processes. They use an exact analytical solution for the learning approach, which they claim to be more robust and computationally efficient compared to other

numerical methods in the same field.

The kernel function is highly relevant for both the HAWKES process and survival process. This determines the shape of the deterministic function that defines how the intensity function changes over time. One option is to have a parametric distribution, which is what we have used in this thesis (Exponential, Rayleigh and Power-law). Another way that has been used in literature is using a non-parametric kernel (using all data points). Yang et al. [15] develop an online learning algorithm to learn the triggering functions (referred as kernel distribution or impact function) associated with the multivariate HAWKES process. They estimate the background intensity and the kernel function together. The triggering function has a positivity constraint, which leads them to apply a semi-definite programming strategy to find the optimal function. The computation cost per iteration is $\mathcal{O}(p^2)$ where p is the number of dimensions of the HAWKES process. [16] non-parametrically estimate the shape of the kernel shape for symmetric HAWKES processes. It involves computing the empirical correlation matrix using fourier transforms. It is more relevant to financial and earthquake data where there are a lot of events. Once the shape of the kernel is estimated using these fourier transforms, a regular maximum likelihood can be maximized to solve for the parameters of the model. The main advantage this paper offers is that the maximum likelihood approach can be used with a high level of confidence because the kernel has been chosen based on empirical evidence. Xu et al. [17] represent the impact function as a sum of (fixed number of) basis functions. This is in an attempt to find the Granger causality in a multivariate HAWKES process. Since this method is a strictly non-parametric method of estimating the impact function, it performs better than other known non-parametric methods.

There are some papers that leverage the point process model to propose effective strategies for fake news mitigation. These papers deal with specific situations of fake news mitigation. Farajtabar et al. [18] propose a framework that depends upon certain actors in the social-network, these are users in the network who can determine whether or not a piece of news is fake. They formulate their objective function by trying to figure out when to call upon these actors to cull the fake news. Obviously these moderators are incentivized to act as moderators. They develop a policy iteration method to multivariate point process for maximizing total reward (maximum fake news mitigated) under budgetary constraints. They model it as two processes $F(t)$ and $M(t)$ which is the fake news process and the mitigation process respectively. The problem is formulated by manipulating the reward function by ensuring correlation maximization: if a user is exposed to fake news, they must also be exposed to the true news. They use a Least Squared Temporal difference (LSTD) policy iteration scheme to find the best policy for intervention. Kim et al. [19] on the other hand leverage the crowd to mitigate fake news rather than predetermined moderators embedded within the network. This model depends upon users themselves flagging content as fake news. The objective function tries to compromise between misinformation evidence and determining which stories to send for checking. They find which stories to send for checking by solving an optimal control problem for Stochastic Differential Equations (SDE) with jumps. They propose an algorithm called CURB to determine which news to send for checking for efficient fake news mitigation which involves finding the fact-checking intensities.

2.3 Hybrid Modelling Techniques

Under the umbrella of hybrid techniques combining information diffusion and feature driven techniques, some have used a deep-learning approach to predicting information cascades. Li et al proposed DeepCas [20] which used deep learning methods to provide a the size increment of a cascade in a given time period. They represent each cascade as a graph and use the node embedding using node2vec (originally proposed by Grover et al [21]). They come up with a representation fixed dimensional representation of the cascade layers down the machine learning pipeline. Cao et al [22] proposed DeepHawkes which is also a deep-learning pipeline modified to use the interpretable factors of HAWKES process. DeepHawkes is used to predict the size increment of a cascade within a time period. Mei et al. [23] model the stream of events as a neurally self-modulating multivariate point process. The intensities of each individual process is controlled by an LSTM cell in the deep-learning pipeline. Gao et al [24] propose a reinforced Poisson process model (RPP) to predict the final size using the concept of "rich get richer" concept. [25] improves upon the single memory kernel of RPP by introducing it as a HAWKES process and they find popularity as the expectation of the intensity function. [26] proposes modelling twitter hashtag popularity using a reinforcement and competitive modification to the HAWKES point process. There are a few methods that perform popularity prediction based on features of an information cascade or the nodes. Wu et al [27] propose PRENETS which employs a critic and interpreter model as is prevalent in WGANs. This is both a feature driven and point process based model for popularity prediction. They have the critic (feature driven) and interpreter (point process based) to compete against each other in a minimax game to produce the best results.

Chapter 3

Background

In order to derive the NETRATE model, we need to introduce some basic concepts involving stochastic point processes and survival analysis fundamentals. The following sections will explore these topics in its appropriate depth and finally arrive at a likelihood that describes the NETRATE model.

3.1 Notation

We will use \mathbb{R} to denote the set of real numbers and \mathbb{R}_+ to denote the set of non-negative real numbers. Additionally, we will define the Cartesian set products $\mathbb{R}^N \triangleq \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{N \text{ times}}$ and \mathbb{R}_+^N , which is similarly defined. All other sets, which we will define, are denoted by capitalized caligraphic letters. Furthermore, we are going to make use of the Iverson bracket, which is defined as

$$\llbracket \text{predicate} \rrbracket = \begin{cases} 1 & \text{if the predicate evaluates to true} \\ 0 & \text{otherwise} \end{cases} \quad (3.1)$$

We will use \triangleq to define quantities and \equiv in lieu of $=$ to emphasize tautological equalities. \succcurlyeq will stand for the element-wise \geq relationship between two vectors.

3.2 Information Diffusion

Tasks like network inference and popularity prediction work well when applied under a context of an information diffusion model. The model serves to explain the time-dependent actions that take place within these networks. To that end, we shall explore a specific framework for information diffusion based on survival theory, which falls under the broad category of stochastic point process based methods.

3.2.1 Stochastic Point Processes

Point processes provide a useful framework for modelling events points that fall on the time axis. Point processes can be applied to a variety of situations. For example, they can represent a new entry to the population, a tweet or a retweet, a user sharing a piece of information in some social media platform and even the tremors of an earthquake. Point processes therefore form the basis for any survival process. In general, point process do not care about the re-occurrence of a an event in a process. Although not the main topic of this thesis, we will show that using a NETRATE model amounts a point process under some assumptions of survival and the like.

3.2.1.1 The Intensity Function λ

Consider the simplest point process: Poisson process. It is well known for its memory-less property, which means there is no need to keep track of the history. The Poisson process contains a sequence of inter-arrival times τ_i which are Identically Independently Distributed (IID) exponential random variables, each with the same rate parameter λ . The Poisson process enjoys the following properties:

1. Stationary increment property. The number of arrivals on any time interval is independent of the exact time, but on the duration of time interval.
2. Independent increments property. This follows from the IID exponential RVs that the inter-arrival times are sampled from.

The Poisson process, although powerful for modelling, is still a simple point process. Situations in reality often requires that the value of the intensity function λ to vary over time. i.e. $\lambda = \lambda(t)$. This brings us to the definition of a non-homogeneous point process. Just as the λ value is sufficient to characterize a homogeneous-Poisson process, for a non-homogeneous Poisson process, it is sufficient to model the conditional intensity function.

$$\lambda(t | \mathcal{H}_t) = \lim_{h \rightarrow 0} \frac{\mathcal{P}\{N_{t+h} - N_t = 1 | \mathcal{H}_t\}}{h} \quad (3.2)$$

Where \mathcal{H}_t contains the history. Which is a collection of subsets which is closed under complement and under countable unions. In stochastic process literature, the history is embedded within the natural filtration of the process \mathcal{F}_t . The

process is adapted to the natural filtration. i.e. the process will keep a record of its past history. For notational and practical convenience, we will simply refer to natural filtration as its history \mathcal{H}_t from here on.

3.2.1.2 The HAWKES Point Process

A majority of events that occur in reality (under any applicable context) arrive in two ways. They can arrive independently at a constant rate, or they can arrive as a consequence of excitation due to the other events in the history \mathcal{H}_t . This is best exemplified by tweets in the social media platform Twitter. A user can tweet of his own volition or tweet as a response to another tweet. Although we lose the Stationary and independent increments property associated with the homogeneous-Poisson process, we get a rich model that can capture the self and mutually exciting behavior of point processes. The HAWKES models this behavior with a specific form of the intensity function for a single process. So, the HAWKES process is a counting process whose intensity depends on the past events. Such a process proves useful for the study of seismicity [[28], [29]]. For more information on HAWKES processes, please refer to the tutorial paper by Rizoïu et al. [30].

$$\lambda(t \mid \mathcal{H}_t) = \mu + \sum_{i:t > T_i} g(t - T_i) \quad (3.3)$$

where μ is the background intensity, T_i records the occurrence of previous points (contained within the history \mathcal{H}_t) [we then say that the process is adapted to the natural filtration \mathcal{H}_t]. $g()$ is called the memory kernel. It is a deterministic base intensity whose form must be known beforehand. There are parametric forms of this

function like exponential and power-law distributions. However there are some non-parametric approaches to finding the shape of this function which have been covered in 2.2 [[31]. [17], [15]]. Knowing the form of the function also helps to determine various statistics like expected values, dynamic contagion impact, etc.. about the process itself [32]

The above HAWKES process is a single point process. To account for multiple actors within a network, we will consider a process analogous to a multi-dimensional HAWKES process for NETRATE modelling.

3.2.2 Survival Analysis Fundamentals

Survival analysis is applicable whenever there exists a concept of finality. This is exemplified in the following examples: time until death due to a disease, time until AIDS for an HIV patient (cite) and time until acquisition of knowledge due to information transfer. We will be dealing with the latter case in this thesis. Without loss of generality, we here introduce some of the basic concepts of survival analysis that we will use throughout this paper. Below you will find the same intensity functions mentioned earlier for stochastic point processes. However, being a survival point process, it takes a different form whose origins can be seen in the HAWKES model intensity equation (3.3). Consider a PDF of the time of death $f(t)$, whose is given by $S(t)$ and Cumulative intensity function is given by $\mathcal{F}(t)$.

3.2.2.1 Survival Function

The survival function of an entity at a given time is defined as the probability that the entity has survived beyond that time. Consider that the entity dies at time T ,

then the survival function is defined as:

$$S(t) = \mathcal{P}\{T > t\} \quad (3.4)$$

$$= 1 - \mathcal{F}(t) \quad (3.5)$$

$$= \int_t^{+\infty} f(\tau) d\tau \quad (3.6)$$

where $\mathcal{F}(t)$ is the CDF of the time of death. The survival function is translated as the probability that the entity can survive beyond the time t .

3.2.2.2 Hazard Function

The Hazard rate or function is defined as the instantaneous rate of arrival of infection events, conditioned on the fact that it has yet to be infected.

the formal definition of the hazard function is

$$\lambda(t) = \lim_{\Delta t \rightarrow 0} \frac{\mathcal{P}\{t < T \leq t + \Delta t \mid T > t\}}{\Delta t} \quad (3.7)$$

As NETRATE is a diffusion model that is based on point processes and point processes themselves can be viewed under a lifetime modeling prism from one event to another, in this section we will provide some background material related to the concepts of Intensity–Survival Probability (I-SP) and Finite Mixtures of Intensities (FMOI) representations. These notions and some accompanying results will be useful in our upcoming exposition of NETRATE.

In survival analysis, *i.e.*, in the field of modeling lifetime distributions, one of the prominent Random Variables (RVs) is the time to an event of interest. Without loss of generality, such times can be regarded as non-negative RVs. Therefore,

in such analysis, it is more convenient to express and manipulate the PDFs of such RVs (assuming that they exist) in their I-SP representations¹. The next statement, whose facts are very well known in survival analysis, reveals this type of representation.

Proposition 1 (I-SP Representation). *Let $\lambda : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ and define $\Lambda(t) \triangleq \int_0^t \lambda(\tau) d\tau$. If λ is defined on the entire set \mathbb{R}_+ and $\lim_{t \rightarrow +\infty} \Lambda(t) = +\infty$, then the function f defined as*

$$f(t) \triangleq \lambda(t) \exp\{-\Lambda(t)\} \quad (3.8)$$

is a PDF of a non-negative RV, say T , with \mathbb{R}_+ as its support. λ and Λ are called the intensity and cumulative or integrated intensity function of T . And, vice versa: if T is a non-negative RV with PDF f , then its intensity and cumulative intensity function are given respectively as

$$\lambda(t) \triangleq \frac{f(t)}{S(t)} \quad (3.9)$$

and

$$\Lambda(t) \triangleq -\ln S(t) \quad (3.10)$$

where

$$S(t) \triangleq \mathbb{P}\{T > t\} = \int_t^{+\infty} f(\tau) d\tau \quad (3.11)$$

and where S is T 's survival function.

¹These particular representations do not appear to have a standard name in the literature. The nomenclature used here reflects our own preferences.

Note that, in the survival analysis literature, the intensity and cumulative intensity are more commonly referred to as *hazard rate* and *cumulative hazard rate* respectively and the distribution of T as *lifetime distribution*. Moreover note that both of these quantities are non-negative and either one of them uniquely defines (aside from a subset of \mathbb{R}_+ of measure 0) the PDF.

Proposition 2. *Assume a collection of J mutually independent, non-negative RVs $\{T_j\}_{j=1}^J$ with PDFs and I-SP representations $\{f_j(t) = \lambda_j(t) \exp\{-\Lambda_j(t)\}\}_{j=1}^J$ respectively. Define $T \triangleq \bigwedge_{j=1}^J T_j$. Then, the PDF f of T is given as*

$$f(t) = \left[\sum_{j=1}^J \lambda_j(t) \right] \exp \left\{ - \sum_{j=1}^J \Lambda_j(t) \right\} \quad (3.12)$$

Again, the proof for this proposition is given in A.2. The previous proposition provide us a way to draw samples from f : we only need to independently draw samples t_j 's from the f_j 's and then yield $t = \bigwedge_{j=1}^J t_j$, which will now be a sample from f . This proposition also plays a significant role when it comes to the simulation (drawing samples) of point processes.

Another useful concept is the potential FMOI representation of the PDF of a non-negative RV.

Definition 1 (Finite Mixture of Intensities Representation). We will say that a non-negative RV T admits a *finite mixture of intensities representation* of J components, if its PDF f can be expressed as

$$f(t) = \left[\sum_{j=1}^J \lambda_j(t) \right] \exp \left\{ - \sum_{j=1}^J \Lambda_j(t) \right\} \quad (3.12)$$

where the λ_j 's are intensity functions and the Λ_j 's are the respective cumulative intensities.

An alternative form of the same PDF can be easily obtained by multiplying each element of the sum with the exponential term and then utilizing (3.10):

$$f(t) = \sum_{j=1}^J f_j(t) \prod_{k \neq j} S_k(t) \quad (3.13)$$

where we have defined

$$f_j(t) \triangleq \lambda_j(t) S_j(t) \quad (3.14)$$

and

$$S_j(t) \triangleq \exp\{-\Lambda_j(t)\} \quad (3.15)$$

This way we observe that f can be written as a sum of J terms and, at first, it does bare a resemblance to ordinary finite mixtures of distributions. However, it does so only superficially, as, in general, $\sum_{j=1}^J \prod_{k \neq j} S_k(t) \neq 1$ for all t .

The nature of the above mixture PDF is revealed clearer by the next statements, which are also widely known in the survival analysis and point process literature.

Proposition 3. *Consider the same setting as in Proposition 2 and define the RV $\text{pa}(T) \triangleq \arg \min_{j=1, \dots, J} T_j$. Then,*

$$p_j(t) \triangleq \mathbb{P}\{\text{pa}(T) = j | T = t\} = \frac{\lambda_j(t)}{\sum_{k=1}^J \lambda_k(t)} \quad (3.16)$$

$$\mathbb{P}\{\text{pa}(T) = j\} = \int_{\tau=0}^{+\infty} f_j(\tau) \prod_{k \neq j} S_k(\tau) d\tau \quad (3.17)$$

and

$$f(t, \text{pa}(T) = j) = f_j(t) \prod_{k \neq j} S_k(t) \quad (3.18)$$

Once again, the proof can be found in Section A.3 Based on this proposition, it follows immediately that

$$(3.17), (3.18) \Rightarrow f(t|\text{pa}(T) = j) = \frac{f_j(t) \prod_{k \neq j} S_k(t)}{\int_{\tau=0}^{+\infty} f_j(\tau) \prod_{k \neq j} S_k(\tau) d\tau} \quad (3.19)$$

Hence, it is obvious now that f can indeed be written in traditional mixture form as

$$(3.13), (3.18) \Rightarrow f(t) = \sum_{j=1}^J f(t, \text{pa}(T) = j) = \sum_{j=1}^J f(t|\text{pa}(T) = j) \mathbb{P}\{\text{pa}(T) = j\} \quad (3.20)$$

Also as a sidenote, when the source $\text{pa}(T)$ of T 's value is observed, then it is useful to keep in mind that

$$f(t, \text{pa}(T) = j) = \mathbb{P}\{\text{pa}(T) = j|T = t\} f(t) \quad (3.21)$$

In survival analysis as well as in the realm of point processes, intensities such as the λ_j 's we just considered are typically constructed or modelled by conically² combining so-called memory kernels, which we define next.

3.2.3 Kernels

Definition 2 (MK). We will call a function $\phi : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ a MK, iff $\int_{\mathbb{R}_+} \phi(\tau) d\tau = +\infty$.

²A conic combination of quantities is a linear combination of those quantities featuring non-negative coefficients.

Definition 3 (IMK). We will call a function $\psi : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ a IMK, iff $\psi(t) \triangleq \int_0^t \phi(\tau) d\tau$ for some MK ϕ .

Proposition 4. *If ϕ is a MK and ψ its associated IMK, then $f(t) \triangleq \phi(t) \exp\{-\psi(t)\} \mathbb{I}[t \geq 0]$ is a valid PDF of a non-negative RV.*

The proof for the above can be found in Section A.4. An *intensity memory kernel*, or, in our context, simply referred to as *memory kernel* (MK), is a function $\phi : \mathbb{R}_+ \rightarrow \mathbb{R}_+$, such that its associated *integrated intensity memory kernel*, or, plainly, *integrated memory kernel* (IMK) $\psi(t) \triangleq \int_{\tau=0}^{+\infty} \phi(\tau) d\tau$ is divergent, *i.e.*, $\lim_{t \rightarrow +\infty} \psi(t) = +\infty$.

By consulting Proposition 1, it becomes obvious that an (integrated) memory kernel can serve as a (cumulative) intensity for a lifetime distribution. It is interesting to note that, due to its definition, $\psi(0) = 0$ for all IMKs; a similar statement also holds for cumulative intensity functions. Finally, often-considered kernels are shown in Table 3.1; in general, each one of them gives rise to a different lifetime PDF. Note that decisions about the parametric shape of the memory kernel must be made qualitatively. There are a few non-parametric methods to estimate the shape of the IMK, $\psi()$ function. There are some non-parametric methods to estimate the ψ function, however, that is an entirely different field of study that we shall skip at this moment. We make a choice for the IMK based on the likelihoods produced on the data.

Another element that we are going to need for our exposition of NETRATE is a minorizer for the log-likelihood of a set of samples drawn from a FMOI, such as f . This will allow us in the future to derive an EM-based algorithm to train NETRATE models. We will reach this goal with the aid of the next proposition.

MK $\phi(t)$	IMK $\psi(t)$	PDF $f(t) = \phi(t) \exp\{-\psi(t)\}$
1	t	unit scale (rate) Exponential
t	$\frac{t^2}{2}$	unit scale Rayleigh
$\frac{1}{t} \llbracket t \geq \beta \rrbracket$	$\ln\left(\frac{t}{\beta}\right) \llbracket t \geq \beta \rrbracket$	unit shape Power Law $\beta > 0$
$\gamma t^{\gamma-1}$	t^γ	unit scale Weibull $\gamma > 0$
$\frac{\delta \frac{\gamma}{\beta} \left(\frac{t}{\beta}\right)^{\gamma-1}}{e\left(\frac{t}{\beta}\right)^\gamma - 1}$	$-\ln\left(1 - \left(1 - e^{-\left(\frac{t}{\beta}\right)^\gamma}\right)^\delta\right)$	Exponentiated Weibull $\beta, \gamma, \delta > 0$
$e^{\gamma t}$	$\frac{1}{\gamma} (e^{\gamma t} - 1)$	special case of $\gamma > 0$ Gompertz
$\frac{\beta}{1 + (\gamma - 1)e^{-\beta t}}$	$\ln\left(1 + \frac{e^{\beta t} - 1}{\gamma}\right)$	special case of Gamma/- $\beta, \gamma > 0$ Gompertz
$\frac{\frac{\gamma}{\beta} \left(\frac{t}{\beta}\right)^{\gamma-1}}{1 + \left(\frac{t}{\beta}\right)^\gamma}$	$\ln\left(1 + \left(\frac{t}{\beta}\right)^\gamma\right)$	special case of Fisk $\beta, \gamma > 0$ (log-logistic)
$\beta + e^{-\gamma t}$	$\beta t + \frac{1}{\gamma} (1 - e^{-\gamma t})$	$\beta, \gamma > 0$

Table 3.1: Various MKs with their corresponding IMKs and the resulting PDFs. As a reminder, all above quantities are defined for $t \geq 0$, except for the Power Law case, where they are defined for $t \geq \beta$. All β 's are scale parameters of the corresponding PDFs, while all γ 's, as well as δ , are shape parameters of the corresponding PDFs.

Proposition 5 (Jensen-type Inequality for Logarithms of Sums). *Let $\boldsymbol{\lambda} \in \mathbb{R}^J$ be a vector, such that $\boldsymbol{\lambda} \succ \mathbf{0}$, and $\mathcal{S}^{J-1} \triangleq \{\mathbf{p} \in \mathbb{R}_+^J : \mathbf{1}^T \mathbf{p} = 1\}$ be the $(J - 1)$ -dimensional probability simplex, where $\mathbf{1}$ is the appropriate all-ones vector. Then, it holds that*

$$\max_{\mathbf{p} \in \mathcal{S}^{J-1}} \sum_{j=1}^J p_j \ln \left(\frac{\lambda_j}{p_j} \right) = \ln \sum_{j=1}^J \lambda_j \quad (3.22)$$

for

$$\arg \max_{\mathbf{p} \in \mathcal{S}^{J-1}} \sum_{j=1}^J p_j \ln \left(\frac{\lambda_j}{p_j} \right) = \frac{\boldsymbol{\lambda}}{\mathbf{1}^T \boldsymbol{\lambda}} \quad (3.23)$$

Hence, it also holds that

$$\sum_{j=1}^J p_j \ln \left(\frac{\lambda_j}{p_j} \right) \leq \ln \sum_{j=1}^J \lambda_j \quad \text{for any } \mathbf{p} \in \mathcal{S}^{J-1} \quad (3.24)$$

when following the information-theoretic convention of $0 \ln 0 = 0$.

The facts of the previous proposition are considered well-known in the literature. It can be easily established using the fact that $\ln(\cdot)$ is concave and by applying Jensen's inequality.

With this preliminary fact in place, let's suppose that the lifetime distribution of an event time T is given as an FMOI of J components, where the intensities $\{\lambda_j(t|\boldsymbol{\theta})\}_{j=1}^J$ and corresponding cumulative intensities $\{\Lambda_j(t|\boldsymbol{\theta})\}_{j=1}^J$ are functions of some parameter $\boldsymbol{\theta} \in \mathbb{R}^P$. Moreover, let's suppose that we would like to estimate $\boldsymbol{\theta}$ from data via Maximum Likelihood Estimation (MLE). For the purposes of deriving a minorizer of T 's log-likelihood function, we only need to consider a data set consisting of only of one observed sample of T , say t . Then, the log-likelihood

we need to consider takes the form

$$\ell(\boldsymbol{\theta}|t) = \ln \left(\sum_{j=1}^J \lambda_j(t|\boldsymbol{\theta}) \right) - \sum_{j=1}^J \Lambda_j(t|\boldsymbol{\theta}) \quad (3.25)$$

which, via (3.16), is easily shown to be equivalent to

$$\ell(\boldsymbol{\theta}|t) = \sum_{j=1}^J \text{pa}_j(t|\boldsymbol{\theta}) \ln \left(\frac{\lambda_j(t|\boldsymbol{\theta})}{\text{pa}_j(t|\boldsymbol{\theta})} \right) - \sum_{j=1}^J \Lambda_j(t|\boldsymbol{\theta}) \quad (3.26)$$

If we wish to employ an EM algorithm to numerically maximize the above likelihood, perhaps, because of the inconvenient presence of the sum-of-log terms in the log-likelihood, Proposition 6 provides us with the EM minorizer $\bar{\ell}(\boldsymbol{\theta}|\boldsymbol{\theta}', t)$ of $\ell(\boldsymbol{\theta}|t)$ via Proposition 5, meaning that

$$\bar{\ell}(\boldsymbol{\theta}|\boldsymbol{\theta}', t) < \ell(\boldsymbol{\theta}|t) \quad \text{for } \boldsymbol{\theta}' \neq \boldsymbol{\theta} \quad (3.27)$$

and

$$\bar{\ell}(\boldsymbol{\theta}|\boldsymbol{\theta}, t) = \ell(\boldsymbol{\theta}|t) \quad (3.28)$$

Proposition 6 (EM-based Minorizer). *Application of the E-step of the EM methodology yields*

$$\bar{\ell}(\boldsymbol{\theta}|\boldsymbol{\theta}', t) \triangleq \sum_{j=1}^J \text{pa}_j(t|\boldsymbol{\theta}') \ln \left(\frac{\lambda_j(t|\boldsymbol{\theta})}{\text{pa}_j(t|\boldsymbol{\theta}')} \right) - \sum_{j=1}^J \Lambda_j(t|\boldsymbol{\theta}) \quad (3.29)$$

The proof for the above can be found in the seminal paper for the EM algorithm [33]. If we maximize $\bar{\ell}(\boldsymbol{\theta}|\boldsymbol{\theta}', t)$ with respect to $\boldsymbol{\theta}$ to obtain the maximizer $\boldsymbol{\theta}^*$, which will be a function of $\boldsymbol{\theta}'$, we can formulate the relevant EM algorithm as the iterative map $\boldsymbol{\theta}' \mapsto \boldsymbol{\theta}^*$.

3.2.4 On the Incompleteness of Data

Whenever dealing with survival analysis studies, we need to discuss the notion of incomplete data. Survival Analysis literature mainly discuss two types of events that cause incompleteness in the data: Censoring and Truncation [34]. When the observation is terminated as a pre-assigned time, it is referred to as truncation. Whereas, data is censored when the observation is terminated after a reaching a fixed number of infections. Censoring occurs in all observations. This is because of the constraints that drive how long the observation period is. Censoring basically establishes a window within the entire lifetime of event. The right censoring time plays an important role in survival analysis. As is the situation with our data, we have a (random) right-censored data. Motivated by practical settings, we only consider the events before a pre-determined right-censoring time, *right-censoring time* $t_{rc} > 0$. Hence, some events will not be observed in contrast to our assumptions so far. These circumstances imply that the log-likelihood of (3.25) ceases to be applicable. One could, of course, consider only uncensored event times t , *i.e.*, ones that have been observed and proceed to estimating $\boldsymbol{\theta}$ via (3.25). Nevertheless, doing so discards observed information: the fact that some event times have not been observed or, equivalently, the fact that $T > t_{rc}$ for some event. This translates to less accurate estimates of $\boldsymbol{\theta}$.

Perhaps interestingly, right-censoring can be elegantly addressed using an FMOI. Assume that the right-censoring time T_{rc} is a RV independent of the event time T and whose distribution has \mathbb{R}_+ as support. What is actually observed is the value t_{obs} of $T_{\text{obs}} \triangleq T \wedge T_{rc}$, which, based on what we have discussed about FMOIs, has

a PDF of

$$\begin{aligned}
f_{T_{\text{obs}}}(t_{\text{obs}}) &= [f_T(t_{\text{obs}})S_{T_{\text{rc}}}(t_{\text{obs}})]^{\llbracket t_{\text{obs}}=t \rrbracket} [f_{T_{\text{rc}}}(t_{\text{obs}})S_T(t_{\text{obs}})]^{\llbracket t_{\text{obs}}=t_{\text{rc}} \rrbracket} = \\
&= [f_T(t)S_{T_{\text{rc}}}(t)]^{\llbracket t_{\text{obs}}=t \rrbracket} [f_{T_{\text{rc}}}(t_{\text{rc}})S_T(t_{\text{rc}})]^{\llbracket t_{\text{obs}}=t_{\text{rc}} \rrbracket} = \\
&= f_T(t)^{\llbracket t_{\text{obs}}=t \rrbracket} S_T(t_{\text{rc}})^{\llbracket t_{\text{obs}}=t_{\text{rc}} \rrbracket} \underbrace{f_{T_{\text{rc}}}(t_{\text{rc}})^{\llbracket t_{\text{obs}}=t_{\text{rc}} \rrbracket} S_{T_{\text{rc}}}(t)^{\llbracket t_{\text{obs}}=t \rrbracket}}_{g_{T_{\text{rc}}}(t_{\text{obs}}, t, t_{\text{rc}}) \triangleq} \quad (3.30)
\end{aligned}$$

which yields a likelihood of

$$\begin{aligned}
\ln f_{T_{\text{obs}}}(t_{\text{obs}}) &= \llbracket t_{\text{obs}} = t \rrbracket \ln f_T(t) + \llbracket t_{\text{obs}} = t_{\text{rc}} \rrbracket \ln S_T(t_{\text{rc}}) + \\
&\quad + \ln g_{T_{\text{rc}}}(t_{\text{obs}}, t, t_{\text{rc}}) \quad (3.31)
\end{aligned}$$

The $g_{T_{\text{rc}}}$ term depends solely on the right-censoring time's distribution. Since, we typically are not interested in modeling this distribution, we treat it as a constant and ignore it by writing the log-likelihood without it simply as

$$\ln f_{T_{\text{obs}}}(t_{\text{obs}}) = \llbracket t_{\text{obs}} = t \rrbracket \ln f_T(t) + \llbracket t_{\text{obs}} = t_{\text{rc}} \rrbracket \ln S_T(t_{\text{rc}}) \quad (3.32)$$

The previous expression, now, allow us to estimate lifetime distribution parameters for right-censored data. Uncensored data contribute a $\ln f_T(t)$ term to the log-likelihood, while censored data contribute a term of $\ln S_T(t_{\text{rc}})$.

Finally, let us remark that, in our discussion in this section, we assumed all event time (lifetime) distributions to have \mathbb{R}_+ as support and that their intensities are unconditional, that is, they do not depend on other observed RVs. These are not particularly restrictive assumptions and what we have presented so far still applies after only minor, notational modifications.

3.2.5 Cascade Definition

We here standardize the definition of an information cascade and infection events. Information cascades are defined by a sequence of events. Each events contains the information about which node is activated and the time at which it activates. This generalization works with both the survival process definition for NETRATE and the point process activations for the HAWKES process.

Assume a network of $N \geq 2$ nodes and define the set

$$\mathcal{N} \triangleq \{1, 2, \dots, N\} \quad (3.33)$$

By an (*infection*) *event*, we will mean a pair (i, t) , which indicates the index $i \in \mathcal{N}$ of a node that got infected and its associated infection time. For brevity, we will denote such an event as t_i .

By (*information*) *cascade*, we will mean the set of infection events $\mathcal{H}_{T+} \triangleq \{t_{j_1}, t_{j_2}, \dots, t_{j_n}\}$ ordered in chronological precedence and corresponding to the spread of a single unit of information over the network that infected n nodes by the right-censoring time T . In the aforementioned definition, for $k = 1, 2, \dots, n$, j_k is the index of the k^{th} infected node and $0 \leq t_{j_k} \leq T$ is its infection time obtained by considering the same right-censoring time. A typical set of available observations will consist of $C \geq 1$ cascades. We will denote the c^{th} cascade of this collection as \mathcal{H}_{T+}^c ³.

Furthermore, for the c^{th} cascade, we will denote the set of indices of nodes

³This is not to be confused with the complement of \mathcal{H}_T

infected prior to time t as

$$\mathcal{I}^c(t) \triangleq \{j \in \mathcal{N} : t_j^c < t\} \quad (3.34)$$

respectively.

3.3 The NETRATE Information Diffusion Model

These are probabilistic models built on top of a pairwise transmission likelihood between two entities(nodes) in the network. We are adopting a method called NETRATE, originally proposed by Rodriguez et al [1]. They treat information diffusion as a contagion spreading within a domain. By convention, for a given piece of information, we will say that a network node got *infected*, when the node broadcasts this information to the network. Without loss of generality, we will assume that infection times are non-negative. Also, we will assume that we will observe infections up to a *right-censoring time* $T > 0$.

From here on, all defined functions assume a domain of \mathbb{R}_+ .

NETRATE makes the following modeling assumptions:

A.1 An information diffusion network of N users is modelled by a symmetric directed graph of N vertices (nodes), where, for every ordered pair of nodes (j, i) , the corresponding edge from node j to node i is weighted by a *transmission rate* $a_{j,i} \geq 0$. Moreover, $a_{j,i} = 0$ signifies the absence of an edge from j to i .

A.2 For any given time t , future survival or infection events of a set of so-far

uninfected nodes are mutually independent given all infections that occurred prior to time t .

A.3 Any node of the network can be infected (can transmit) only once per cascade.

A.4 The lifetime distribution of each network node $i \notin \mathcal{I}^c(t)$, is governed by the following conditional hazard rate:

$$\begin{aligned} h_i(t|\mathcal{H}_t^c) &\triangleq \sum_{j \in \mathcal{I}^c(t)} a_{j,i} \phi(t - t_j^c) \mathbb{I}[t > t_j^c] = \\ &= \sum_{j \in \mathcal{I}^c(t)} a_{j,i} \phi(t - t_j^c) \quad i \notin \mathcal{I}^c(t) \end{aligned} \quad (3.35)$$

where ϕ is a predetermined memory kernel. By convention, if $\mathcal{I}^c(t) = \emptyset$ for some t , then the value of the latter sum is taken to equal 0.

Since $i \notin \mathcal{I}^c(t) \Leftrightarrow t \leq t_i^c$, assumptions A.3 and A.4 can be consolidated in the following equivalent assumption:

A.5 The infection dynamics of the entire network (which network node gets infected and when) is modelled by a *multi-variate survival point process*, whose CIFs are given as

$$\begin{aligned} \lambda_i(t|\mathcal{H}_t^c) &\triangleq \mathbb{I}[t \leq t_i^c] h_i(t|\mathcal{H}_t^c) \stackrel{(3.35)}{=} \\ &= \mathbb{I}[t \leq t_i^c] \sum_{j \in \mathcal{I}^c(t)} a_{j,i} \phi(t - t_j^c) \quad i \in \mathcal{N} \end{aligned} \quad (3.36)$$

Remark 1. It should be clear from (3.36) that, for any node $i \in \mathcal{N}$, pre-infection CIF values coincide with conditional hazard values and, most importantly, post-

infection CIF values are 0, *i.e.*, $\lambda_i(t|\mathcal{H}_t^c) = 0$ for $t > t_i^c$. Hence, any network node can be infected only once.

3.3.1 More on NETRATE nodes' CIFs

(3.36) can be re-written as

$$\lambda_i(t|\mathcal{H}_t^c) = \sum_{j \in \mathcal{N}} \lambda_i(t|t_j^c) \quad i \in \mathcal{N} \quad (3.37)$$

where

$$\lambda_i(t|t_j^c) \triangleq a_{j,i} \phi(t - t_j^c) \mathbb{I}[t_j^c < t \leq t_i^c] \quad i \in \mathcal{N} \quad (3.38)$$

The latter quantity represents the hazard rate of any network node i , if node j were the only node of the network infected during the c^{th} cascade. Also, (3.37) and (3.38) in tandem confirm the left-continuous nature of the aforementioned CIFs. Finally, the previous equations can be re-written as

$$(3.38) \Rightarrow \lambda_i(t|t_j^c) = \begin{cases} a_{j,i} \phi(t - t_j^c) \mathbb{I}[t_j^c < t] & i \notin \mathcal{I}^c(t) \\ 0 & i \in \mathcal{I}^c(t) \end{cases} \quad (3.39)$$

for any $j \in \mathcal{N}$, and

$$(3.37) \Rightarrow \lambda_i(t|\mathcal{H}_t^c) = \sum_{j \in \mathcal{I}^c(t)} \lambda_i(t|t_j^c) \stackrel{(3.38)}{=} \begin{cases} \sum_{j \in \mathcal{I}^c(t)} a_{j,i} \phi(t - t_j^c) & i \notin \mathcal{I}^c(t) \\ 0 & i \in \mathcal{I}^c(t) \end{cases} \quad (3.40)$$

We notice from (3.40) that the CIFs $\lambda_i(t|\mathcal{H}_t^c)$ are piece-wise left-continuous

with possible jump discontinuities occurring at past node infection times.

3.3.2 NETRATE nodes' CCIFs

The Conditional Cumulative Intensity Function (CCIF) of any node i is defined as

$$\Lambda_i(t|\mathcal{H}_t^c) \triangleq \int_0^t \lambda_i(\tau|\mathcal{H}_\tau^c) d\tau \quad i \in \mathcal{N} \quad (3.41)$$

and, therefore,

$$\lambda_i(t|\mathcal{H}_t^c) = \frac{d\Lambda_i(t|\mathcal{H}_t^c)}{dt} \quad i \in \mathcal{N} \quad (3.42)$$

at any time t , where $\Lambda_i(t|\mathcal{H}_t^c)$ is differentiable. Based on (3.37) and (3.38), if ψ is the IMK of ϕ , (3.41) takes the following form:

$$\begin{aligned} (3.41) \stackrel{(3.37),(3.38)}{\Rightarrow} \Lambda_i(t|\mathcal{H}_t^c) &= \sum_{j \in \mathcal{N}} a_{j,i} \int_0^t \phi(\tau - t_j^c) \mathbb{I}[t_j^c < \tau \leq t_i^c] d\tau = \\ &= \sum_{j \in \mathcal{N}} a_{j,i} \int_0^{+\infty} \phi(\tau - t_j^c) \underbrace{\mathbb{I}[t_j^c < \tau \leq t_i^c] \mathbb{I}[\tau \leq t]}_{=\mathbb{I}[t_j^c < \min\{t, t_i^c\}] \mathbb{I}[t_j < \tau \leq \min\{t, t_i^c\}]} d\tau \\ &= \sum_{j \in \mathcal{I}^c(\min\{t, t_i^c\})} a_{j,i} \int_{t_j^c}^{\min\{t, t_i^c\}} \phi(\tau - t_j^c) d\tau \Leftrightarrow \\ \Leftrightarrow \Lambda_i(t|\mathcal{H}_t^c) &= \sum_{j \in \mathcal{I}^c(\min\{t, t_i^c\})} a_{j,i} \psi(\min\{t, t_i^c\} - t_j^c) \quad i \in \mathcal{N} \quad (3.43) \end{aligned}$$

Furthermore, (3.43) can be re-written as

$$\Lambda_i(t|\mathcal{H}_t^c) = \sum_{j \in \mathcal{N}} \Lambda_i(t|t_j^c) \quad i \in \mathcal{N} \quad (3.44)$$

where

$$\Lambda_i(t|t_j^c) \triangleq a_{j,i} \psi(\min\{t, t_i^c\} - t_j^c) \mathbb{I}[t_j^c < \min\{t, t_i^c\}] \quad i, j \in \mathcal{N} \quad (3.45)$$

For any network node i , the latter quantity represents its cumulative hazard of infection, if node j is the only other node present in the network; nevertheless, see Remark 2 for some nuances. We see that $\Lambda_i(t|t_j^c)$ is differentiable at all times except past node infection times. Finally, the previous CCIF expressions can be re-written as

$$(3.45) \Rightarrow \quad \Lambda_i(t|t_j^c) = a_{j,i}\psi(t - t_j^c)\mathbb{I}[t_j^c < t] \quad i \notin \mathcal{I}^c(t) \quad (3.46)$$

for all $j \in \mathcal{N}$, and

$$(3.44) \Rightarrow \quad \Lambda_i(t|\mathcal{H}_t^c) \stackrel{(3.46)}{=} \sum_{j \in \mathcal{I}^c(t)} a_{j,i}\psi(t - t_j^c) \quad i \notin \mathcal{I}^c(t) \quad (3.47)$$

3.3.3 NETRATE nodes' CPDFs, CCDFs & CSFs

As node lifetime-related functions of t , such as Conditional Survival Functions (CSFs), CCDF and Conditional Probability Density Functions (CPDFs) are only defined for nodes that remain uninfected at time t , all expressions in the subsections hold for nodes $i \notin \mathcal{I}^c(t)$, unless indicated otherwise.

Under the NETRATE model, the CSF of any uninfected node i by time t is given as

$$S_i(t|\mathcal{H}_t^c) = \exp\{-\Lambda_i(t|\mathcal{H}_t^c)\} \stackrel{(3.47)}{=} \exp\left\{-\sum_{j \in \mathcal{I}^c(t)} a_{j,i}\psi(t - t_j^c)\right\} \quad (3.48)$$

which represents the probability of node i surviving up to time t despite the infection events in \mathcal{H}_t^c just prior to time t . Stated another way, it is the probability of i getting infected beyond time t given \mathcal{H}_t^c . Hence, were this node to get infected by time t under the same circumstances, the probability of such an event would

be given by its CCDF value

$$F_i(t|\mathcal{H}_t^c) = 1 - S_i(t|\mathcal{H}_t^c) \quad (3.49)$$

It is useful to define the following quantities:

$$S_i(t|t_j^c) \triangleq \exp\{-\Lambda_i(t|t_j^c)\} \quad j \in \mathcal{N} \quad (3.50)$$

For a given network node j , the previous expression provides the probability of an uninfected node i to get infected beyond time t given that j is the only other node present in the network. Using this definition, we obtain that

$$(3.48) \Rightarrow S_i(t|\mathcal{H}_t^c) = \prod_{j \in \mathcal{N}} S_i(t|t_j^c) = \prod_{j \in \mathcal{I}^c(t)} S_i(t|t_j^c) \quad (3.51)$$

Furthermore, node i 's CPDF (conditional lifetime PDF) can be readily obtained as

$$f_i(t|\mathcal{H}_t^c) = \lambda_i(t|\mathcal{H}_t^c) S_i(t|\mathcal{H}_t^c) \stackrel{(3.40), (3.51)}{=} \sum_{j \in \mathcal{I}^c(t)} \lambda_i(t|t_j^c) \prod_{k \in \mathcal{I}^c(t)} S_i(t|t_k^c) \quad (3.52)$$

Hence, the CPDF for an uninfected node i has the form of the one in Proposition X. If we define

$$f_i(t|t_j^c) \triangleq \lambda_i(t|t_j^c) S_i(t|t_j^c) \quad j \in \mathcal{N} \quad (3.53)$$

then, using (3.53), we can re-express (3.52) as

$$\begin{aligned}
(3.52) \Rightarrow f_i(t|\mathcal{H}_t^c) &= \sum_{j \in \mathcal{I}^c(t)} \lambda_i(t|t_j^c) \prod_{k \in \mathcal{I}^c(t)} S_i(t|t_k^c) = \\
&= \sum_{j \in \mathcal{I}^c(t)} \underbrace{\lambda_i(t|t_j^c) S_i(t|t_j^c)}_{\stackrel{(3.53)}{=} f_i(t|t_j^c)} \prod_{\substack{k \in \mathcal{I}^c(t) \\ k \neq j}} S_i(t|t_k^c) \Leftrightarrow \\
\Leftrightarrow f_i(t|\mathcal{H}_t^c) &= \sum_{j \in \mathcal{I}^c(t)} f_i(t|t_j^c) \prod_{\substack{k \in \mathcal{I}^c(t) \\ k \neq j}} S_i(t|t_k^c) \tag{3.54}
\end{aligned}$$

All this implies that node i will be infected by only one parent node in $\mathcal{I}^c(t)$. Moreover, this implies that, the probability of node pa_i^c infecting i at time t is given by

$$p_{j,i}^c(t) \triangleq \mathbb{P}\left\{\text{pa}_i^c(t) = j \middle| t, \mathcal{H}_t^c\right\} = \frac{\lambda_i(t|t_j^c)}{\sum_{k \in \mathcal{I}^c(t)} \lambda_i(t|t_k^c)} \quad j \in \mathcal{I}^c(t) \tag{3.55}$$

and, therefore, the most probable parent node of i , if i gets infected at time t , is given by

$$\text{pa}_i^{c*}(t) \triangleq \arg \max_{j \in \mathcal{I}^c(t)} p_{j,i}^c(t) = \arg \max_{j \in \mathcal{I}^c(t)} \lambda_i(t|t_j^c) \tag{3.56}$$

Remark 2. The discussion in this sub-section focused on the case, where $i \notin \mathcal{I}^c(t) \Leftrightarrow t \leq t_i^c$. In other words, each of the aforementioned node-specific quantities hold for times t up to node i 's instance of infection. In order to extend these quantities for post-infection times ($t > t_i^c \Leftrightarrow i \in \mathcal{I}^c(t)$), one can coherently argue that $S_i(t|\mathcal{H}_t^c) = 0$ and, hence, perhaps that $\Lambda_i(t|\mathcal{H}_t^c) = +\infty$, instead of the finite constant implied by (3.45). Due to (3.36), we already have that $\lambda_i(t|\mathcal{H}_t^c) = 0$ for post-infection times. But this reasoning implies a post-infection CPDF f_i , which

is 0 everywhere past t_i^c , but integrates to 1. One could take this “paradox” to mean that there will not be a re-infection of i in the future. Nevertheless, all these post-infection considerations are tangential to what we will try to accomplish in the material subsequently presented.

Let us consider an example case with small network of 4 nodes as shown in Figure 3.1. Using this structure, we constructed a simple cascade as shown in figure 3.2. This cascade was constructed to showcase two situations. How the lifetime is affected when there is a single connection and when there are more than one connections. The intensity function $\lambda(t)$, integrated intensity function $\Lambda(t)$ and the lifetime PDF which is obtained as per 1. These are available in Figures 3.3, 3.4 and 3.5. For ease of interpretation, we have used an exponential distribution for the memory kernel ϕ and integrated memory kernel ψ . After the first infection A has taken place, the survival process of both B and C are initiated with constant intensities which are maintained until the respective node is infected. When B is infected, C gains an updated intensity value on account of a connection between B and C. For node D, the only connection is from B. As a result, D’s survival process is started when B is infected. D remains unaffected by C’s infection because there is no connection from C to D.

3.3.4 NETRATE Likelihood

Assume that the c^{th} observed cascade contains n^c events and takes the form $\mathcal{H}_T^c = \{t_{j_1^c}^c, t_{j_2^c}^c, \dots, t_{j_{n^c}^c}^c\}$, where j_n^c for $n = 1, 2, \dots, n^c$ is the index of the chronologically n^{th} infected node of the cascade, while the rest of the nodes survived by the right-censoring time T . For simplicity, we will start by considering the case, where no parent node information is available for the infected nodes.

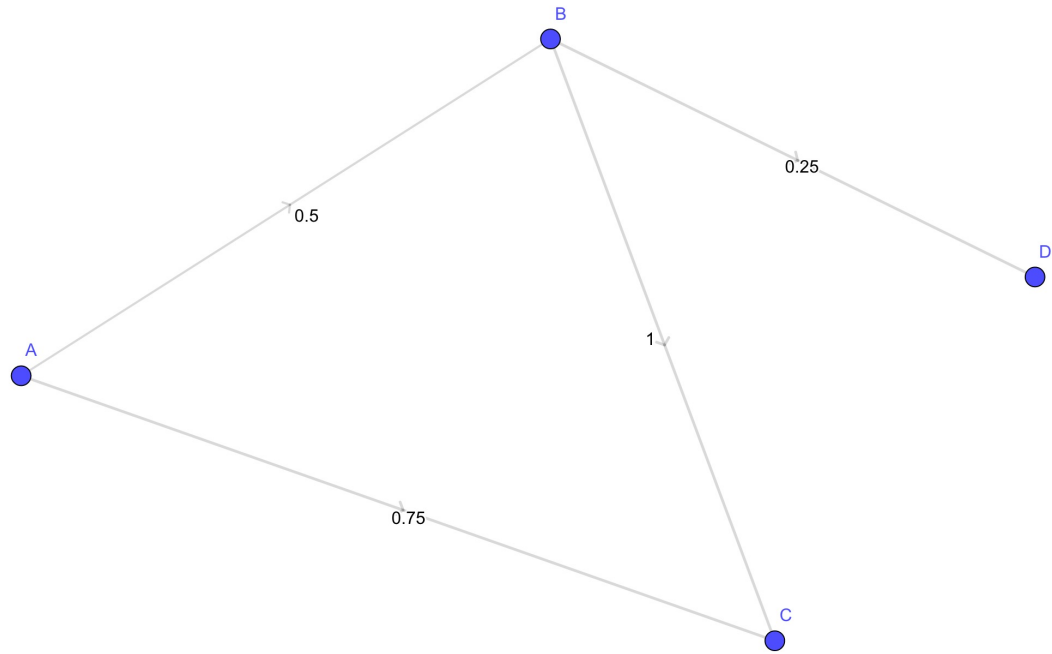


Figure 3.1: Graph used for illustrative example. The transmission rates are present as wedge weights.

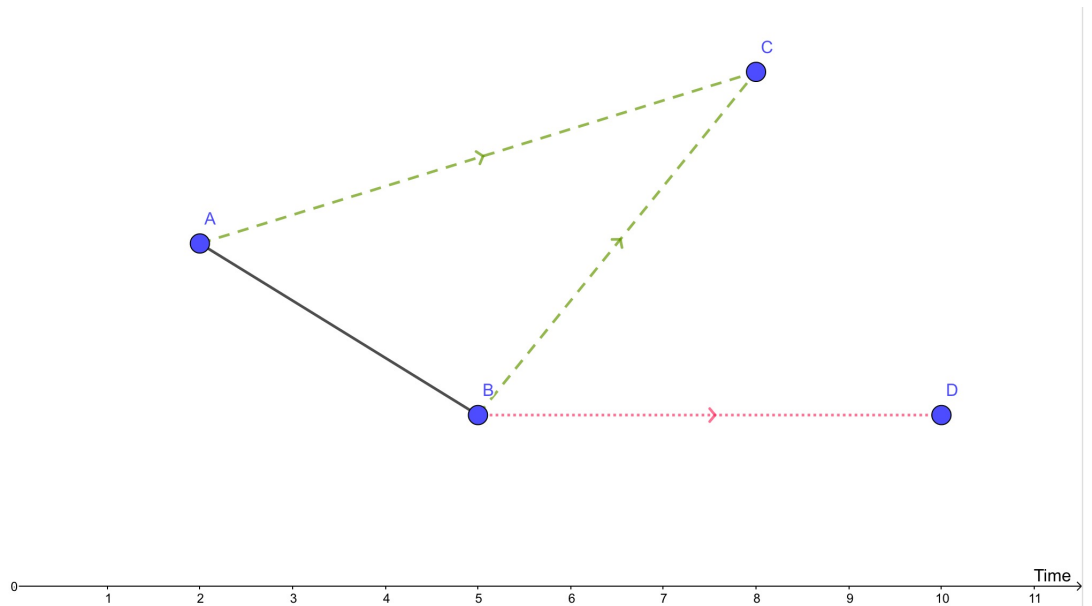
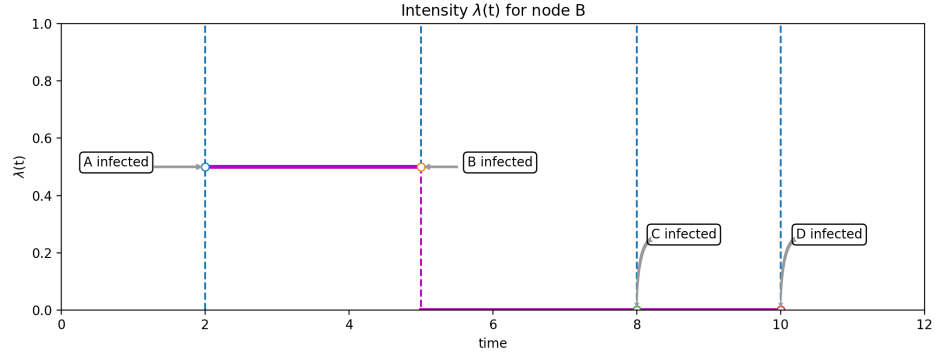
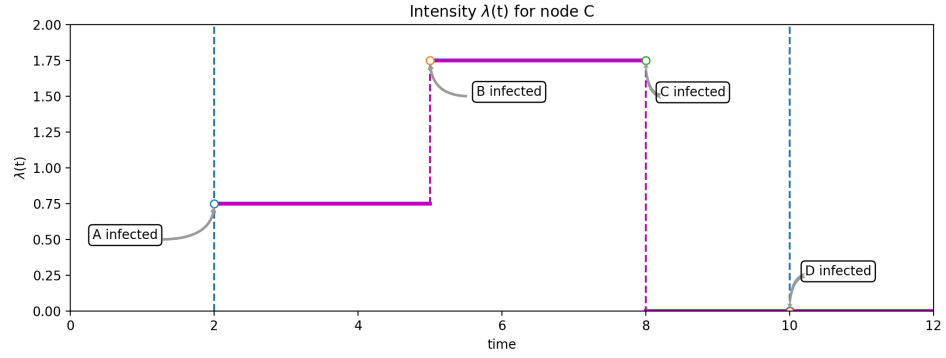


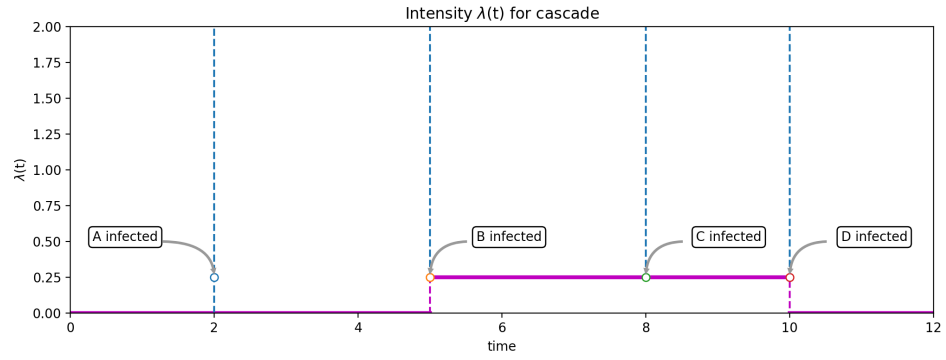
Figure 3.2: Example of a cascade on the example network. The dotted lines indicate possible parentage for infection events, i.e. which node can infect the given node.



(a) Since A is directly linked to B, infection of A jump starts the survival process of B, maintaining a constant intensity until B is infected.

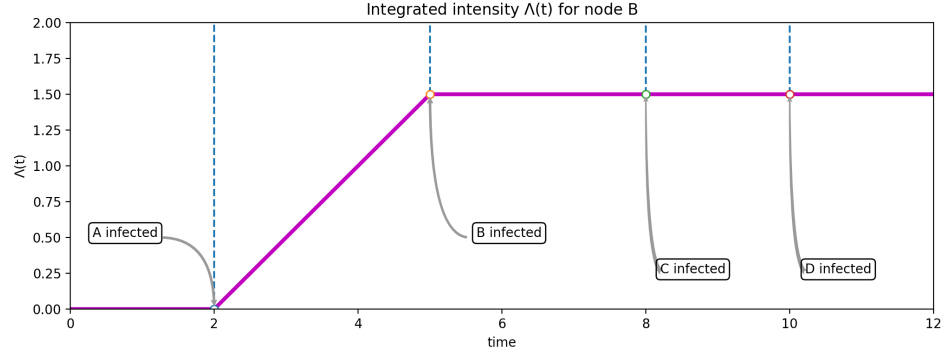


(b) Since A is directly linked to C, infection A jump starts the survival process of C. After the subsequent infection of B, the intensity of C increases since B is also connected to C.

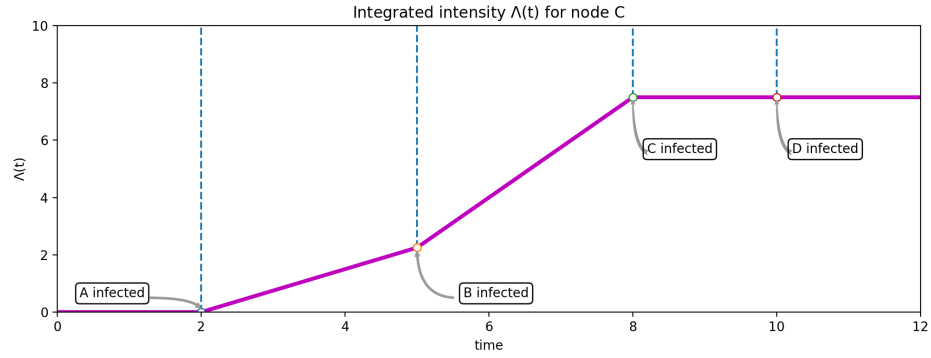


(c) D is only connected to node B. So, once B is infected, D's survival process begins and it maintains a constant intensity until it finally gets infected.

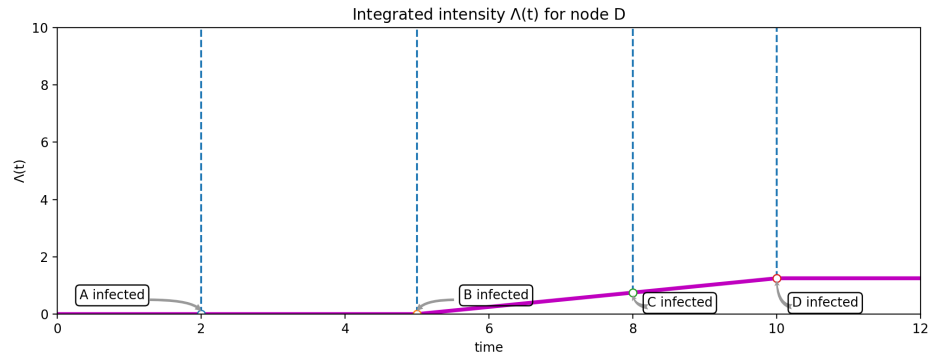
Figure 3.3: Intensity functions $\lambda(t)$ for nodes B,C and D. After infection of each node, their respective intensities go to zero.



(a) After infection of A, the integrated intensity starts increasing until node B is infected, after which point, it remains constant.

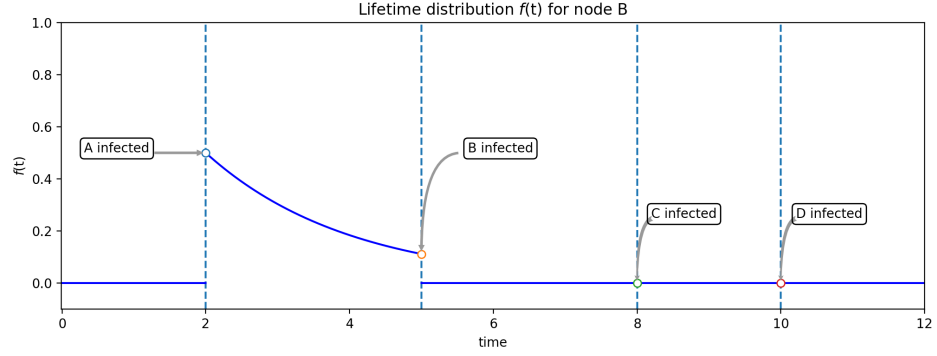


(b) After infection of A, the integrated intensity starts increasing. Once B gets infected, the slope of the Integrated intensity increases since B is also connected to C.

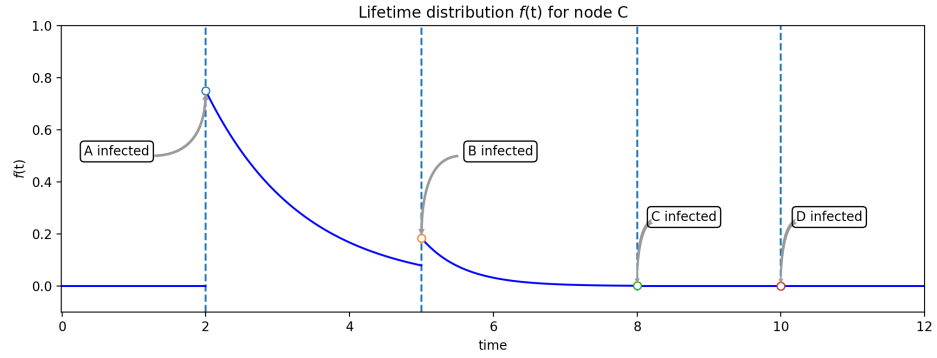


(c) B is the only node connected to D. the integrated intensity starts increasing after infection of B. It remains unchanged when C is infected because its not connected to D.

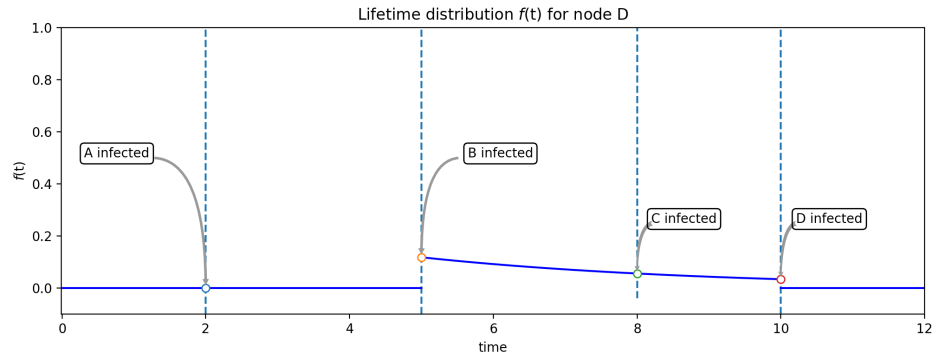
Figure 3.4: Integrated intensity functions $\Lambda(t)$ for nodes B,C and D. After infection of each node, their respective integrated intensities remain constant.



(a) Lifetime distribution of B. It starts when A is infected and has the same shape until it gets infected. It retains shape because no other nodes influence B.



(b) Lifetime distribution of C. It starts when A gets infected. The shape changes after the infection of B because both A and B influence C.



(c) Lifetime distribution of D. It only starts after the infection of B and the shape remains unchanged with the infection of C because there is no connection to D.

Figure 3.5: The lifetime distribution $f(t)$ for nodes B,C and D after the infection of node A.

Noticing that $\mathcal{I}^c(T)$ stands for the index set of all infected nodes appearing in cascade \mathcal{H}_T^c , let us define the index set $\mathcal{I}_{\setminus 1}^c(T) \triangleq \mathcal{I}^c(T) \setminus \{j_1^c\}$, which excludes the index of the first infected node (j_1^c) from the set of all infected nodes. If T_k^c stands for the RV of the k^{th} node's infection time and if the conditioning on n^c is notationally suppressed, the likelihood of observing \mathcal{H}_T^c takes the form ⁴:

$$\begin{aligned}
L(\mathbf{A}|\mathcal{H}_T^c) &\triangleq f\left(\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)}, \bigcap_{k \notin \mathcal{I}^c(T)} \{T_k^c > T\} \middle| t_{j_1^c}^c\right) = \\
&= \mathbb{P}\left\{ \bigcap_{k \notin \mathcal{I}^c(T)} \{T_k^c > T\} \middle| \underbrace{\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)}, t_{j_1^c}^c}_{\equiv \mathcal{H}_T^c} \right\} f\left(\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)} \middle| t_{j_1^c}^c\right) = \\
&= f\left(\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)} \middle| t_{j_1^c}^c\right) \mathbb{P}\left\{ \bigcap_{k \notin \mathcal{I}^c(T)} \{T_k^c > T\} \middle| \mathcal{H}_T^c \right\} \tag{3.57}
\end{aligned}$$

Using the chain rule of conditioning, the first term becomes:

$$\begin{aligned}
f\left(\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)} \middle| t_{j_1^c}^c\right) &= f\left(t_{j_2^c}^c \middle| t_{j_1^c}^c\right) f\left(t_{j_3^c}^c \middle| t_{j_1^c}^c, t_{j_2^c}^c\right) \cdots \\
&\cdots f\left(t_{j_{n^c}^c}^c \middle| \{t_{j_n^c}^c\}_{n=1}^{n^c-1}\right) = \\
&= \prod_{n=2, \dots, n^c} f\left(t_{j_n^c}^c \middle| \mathcal{H}_{t_{j_n^c}^c}^c\right) = \prod_{i \in \mathcal{I}_{\setminus 1}^c(T)} f_i\left(t_i^c \middle| \mathcal{H}_{t_i^c}^c\right) \tag{3.58}
\end{aligned}$$

On the other hand, the second term simplifies to

$$\mathbb{P}\left\{ \bigcap_{k \notin \mathcal{I}^c(T)} \{T_k^c > T\} \middle| \mathcal{H}_T^c \right\} \stackrel{(A.2)}{=} \prod_{k \notin \mathcal{I}^c(T)} \mathbb{P}\{t_k^c > T | \mathcal{H}_T^c\} \equiv$$

⁴The function $f()$ is a density-probability mixture. It infected terms constitute a probability density, whereas the survival terms in the same expression correspond to a probability

$$\equiv \prod_{k \notin \mathcal{I}^c(T)} S_k(T|\mathcal{H}_T^c) \quad (3.59)$$

Finally, the likelihood takes the form:

$$(3.57) \stackrel{(3.58),(3.59)}{\Rightarrow} L(\mathbf{A}|\mathcal{H}_T^c) = \prod_{i \in \mathcal{I}_{\setminus 1}^c(T)} f_i(t_i^c|\mathcal{H}_{t_i^c}^c) \prod_{k \notin \mathcal{I}^c(T)} S_k(T|\mathcal{H}_T^c) \quad (3.60)$$

and, hence, the log-likelihood for the same cascade is given as

$$\begin{aligned} \ell(\mathbf{A}|\mathcal{H}_T^c) &\triangleq \ln L(\mathbf{A}|\mathcal{H}_T^c) \stackrel{(3.48),(3.52)}{=} \\ &= \sum_{i \in \mathcal{I}_{\setminus 1}^c(T)} \left(\ln \lambda_i(t_i^c|\mathcal{H}_{t_i^c}^c) - \Lambda_i(t_i^c|\mathcal{H}_{t_i^c}^c) \right) - \sum_{i \notin \mathcal{I}^c(T)} \Lambda_i(T|\mathcal{H}_T^c) \end{aligned} \quad (3.61)$$

where we renamed k to i .

On the other hand, if the parent node index $\text{pa}_i^c \triangleq \text{pa}_i^c(t_i^c)$ of a node i , which got infected at time t_i^c , is known, then $f_i(t_i^c|\mathcal{H}_{t_i^c}^c)$ in (3.60) needs to be replaced with

$$f_i(t_i^c, \text{pa}_i^c|\mathcal{H}_{t_i^c}^c) = \lambda_i(t_i^c|t_{\text{pa}_i^c}^c) S_i(t_i^c|\mathcal{H}_{t_i^c}^c) \quad (3.62)$$

Let us define the sets

$$\mathcal{I}_{o, \setminus 1}^c(T) \triangleq \left\{ i \in \mathcal{I}_{\setminus 1}^c(T) : \text{pa}_i^c \text{ observed} \right\} \quad (3.63)$$

$$\mathcal{I}_{u, \setminus 1}^c(T) \triangleq \left\{ i \in \mathcal{I}_{\setminus 1}^c(T) : \text{pa}_i^c \text{ unobserved} \right\} \quad (3.64)$$

Note that, if pa_i^c is observed, then $\text{pa}_i^c = j$, for some j s.t. $t_j^c < t_i^c$. In the opposite

case, we are going to use the following convention:

$$\text{pa}_i^c \text{ not observed} \quad \Leftrightarrow \quad \text{pa}_i^c = u \quad (\text{C.1})$$

for some convenient choice of $u \notin \mathcal{N}$ indicating an unknown parent node. In light of the previous set definitions, the log-likelihood of the cascade shown in (3.61) will be modified to

$$\begin{aligned} \ell(\mathbf{A}|\mathcal{H}_T^c) = & \sum_{i \in \mathcal{I}_{o, \setminus 1}^c(T)} \ln \lambda_i(t_i^c | t_{\text{pa}_i^c}^c) + \sum_{i \in \mathcal{I}_{u, \setminus 1}^c(T)} \ln \lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) - \sum_{i \in \mathcal{I}_{\setminus 1}^c(T)} \Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) - \\ & - \sum_{i \notin \mathcal{I}^c(T)} \Lambda_i(T | \mathcal{H}_T^c) \end{aligned} \quad (3.65)$$

If we denote the i^{th} column of \mathbf{A} by \mathbf{a}_i , then the previous log-likelihood can be decomposed as

$$\ell(\mathbf{A}|\mathcal{H}_T^c) = \sum_{i \in \mathcal{N}} \ell_i(\mathbf{a}_i|\mathcal{H}_T^c) \quad (3.66)$$

where

$$\ell(\mathbf{a}_i|\mathcal{H}_T^c) \triangleq \begin{cases} -\Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) & + \begin{cases} \ln \lambda_i(t_i^c | \text{pa}_i^c) & i \in \mathcal{I}_{o, \setminus 1}^c(T) \\ \ln \lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) & i \in \mathcal{I}_{u, \setminus 1}^c(T) \end{cases} \\ -\Lambda_i(T | \mathcal{H}_T^c) & i \notin \mathcal{I}^c(T) \\ 0 & \text{otherwise, i.e., when } i = j_1^c \end{cases} \quad (3.67)$$

and the first main case is applicable for $i \in \mathcal{I}_{o, \setminus 1}^c(T) \cup \mathcal{I}_{u, \setminus 1}^c(T) = \mathcal{I}_{\setminus 1}^c(T)$. In other words, $\ell(\mathbf{A}|\mathcal{H}_T^c)$ is decomposable into log-likelihood terms $\ell(\mathbf{a}_i|\mathcal{H}_T^c)$, one for each node $i \in \mathcal{N}$.

The quantities involved in (3.67) can be computed as follows: for $t = t_i^c$, (3.38) yields

$$(3.38) \xRightarrow{t=t_i^c} \lambda_i(t_i^c | t_{\text{pa}_i^c}^c) = a_{\text{pa}_i^c, i} \phi(t_i^c - t_{\text{pa}_i^c}^c) \quad i \in \mathcal{I}_{o, \setminus 1}^c(T) \quad (3.68)$$

Notice that $t_i^c > t_{\text{pa}_i^c}^c$ should always hold. Similarly, (3.40) yields

$$(3.40) \xRightarrow{t=t_i^c} \lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) = \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j, i} \phi(t_i^c - t_j^c) \quad i \in \mathcal{I}_{u, \setminus 1}^c(T) \quad (3.69)$$

Also, (3.47) gives

$$(3.47) \xRightarrow{t=t_i^c} \Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) = \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j, i} \psi(t_i^c - t_j^c) \quad i \in \mathcal{I}_{\setminus 1}^c(T) \quad (3.70)$$

Finally, again from (3.47) we obtain

$$(3.47) \xRightarrow{t=T} \Lambda_i(T | \mathcal{H}_T^c) = \sum_{j \in \mathcal{I}^c(T)} a_{j, i} \psi(T - t_j^c) \quad i \notin \mathcal{I}^c(T) \quad (3.71)$$

Substituting our latest findings of (3.68), (3.69), (3.70) and (3.71) into (3.66) finally yields

$$\ell(\mathbf{a}_i | \mathcal{H}_T^c) = \begin{cases} - \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j, i} \psi(t_i^c - t_j^c) & + \begin{cases} \ln a_{\text{pa}_i^c, i} + \text{const} & i \in \mathcal{I}_{o, \setminus 1}^c(T) \\ \ln \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j, i} \phi(t_i^c - t_j^c) & i \in \mathcal{I}_{u, \setminus 1}^c(T) \end{cases} \\ - \sum_{j \in \mathcal{I}^c(T)} a_{j, i} \psi(T - t_j^c) & i \notin \mathcal{I}^c(T) \\ 0 & \text{otherwise, i.e., when } i = j_1^c \end{cases} \quad (3.72)$$

An immediate observation from the last expression is that cascades, in which i is

infected first ($j_1^c = i$), do not contribute to $\ell(\mathbf{a}_i|\mathcal{H}_T^c)$.

Assuming an observed collection of C cascades $\mathcal{H}_T \triangleq \{\mathcal{H}_T^c\}_{c \in \mathcal{C}}$, the log-likelihood corresponding to i based on this set is given as the sum of four terms:

$$\begin{aligned}
\ell(\mathbf{a}_i|\mathcal{H}_T) &= \sum_{c \in \mathcal{C}} \ell(\mathbf{a}_i|\mathcal{H}_T^c) \stackrel{(3.72)}{=} \\
&= - \underbrace{\sum_{j \in \mathcal{N}} a_{j,i} \sum_{c \in \mathcal{C}} \psi(t_i^c - t_j^c) \mathbb{I}[i \in \mathcal{I}_{\setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)]}_{\ell_1(\mathbf{a}_i|\mathcal{H}_T) \triangleq} + \\
&\quad + \underbrace{\sum_{j \in \mathcal{N}} (\ln a_{j,i}) \sum_{c \in \mathcal{C}} \mathbb{I}[i \in \mathcal{I}_{o, \setminus 1}^c(T), j = \text{pa}_i^c]}_{\ell_2(\mathbf{a}_i|\mathcal{H}_T) \triangleq} + \\
&\quad + \underbrace{\sum_{c \in \mathcal{C}} \mathbb{I}[i \in \mathcal{I}_{u, \setminus 1}^c(T)] \ln \sum_{j \in \mathcal{N}} \mathbb{I}[j \in \mathcal{I}^c(t_i^c)] a_{j,i} \phi(t_i^c - t_j^c)}_{\ell_3(\mathbf{a}_i|\mathcal{H}_T) \triangleq} - \\
&\quad - \underbrace{\sum_{j \in \mathcal{N}} a_{j,i} \sum_{c \in \mathcal{C}} \psi(t_i^c - t_j^c) \mathbb{I}[i \notin \mathcal{I}^c(T), j \in \mathcal{I}^c(T)]}_{\ell_4(\mathbf{a}_i|\mathcal{H}_T) \triangleq} \tag{3.73}
\end{aligned}$$

We see that $\ell(\mathbf{a}_i|\mathcal{H}_T)$ is comprised of 4 terms involving elements of \mathbf{a}_i . Terms ℓ_1 , ℓ_2 and ℓ_3 jointly concern infections of i by time T due to other previously-infected network nodes. Term ℓ_4 corresponds to cascades c and their encompassing events that i survives within the observational window $(t_{j_1^c}^c, T]$. In the next subsections, we will examine and simplify, to whatever extend, each one of these terms. 3.73 is concave (down) due to the linearity and composition with concave (down) functions for the ϕ and ψ functions described in 3.1.

3.3.4.1 The First Log-Likelihood Term

Let us define the sets

$$\mathcal{C}_1(j|i) \triangleq \{c \in \mathcal{C} : i \in \mathcal{I}_1^c(T), j \in \mathcal{I}^c(t_i^c)\} \quad (3.74)$$

$$\mathcal{J}_1(i) \triangleq \{j \in \mathcal{N} : \exists c \in \mathcal{C}_1(j|i)\} \quad (3.75)$$

The set $\mathcal{C}_1(j|i)$ contains all indices c of cascades in \mathcal{H}_T , for which node i 's infection was observed and node j was infected prior to i . On the other hand, $\mathcal{J}_1(i)$ consists of all indices j of nodes that are infected before i 's across all cascades in \mathcal{H}_T . $\mathcal{J}_1(i) = \emptyset$ if and only if either (i) $j_1^c = i$ for all $c \in \mathcal{C}$, *i.e.* i appears as the first infected node in all cascades of \mathcal{H}_T or (ii) $i \notin \mathcal{I}(T)$, *i.e.*, i 's infection is not observed in any cascade of \mathcal{H}_T .

Using these sets, the first log-likelihood term of (3.73) can be expressed as

$$\ell_1(\mathbf{a}_i|\mathcal{H}_T) = - \sum_{j \in \mathcal{J}_1(i)} a_{j,i} D_1(j|i) \quad (3.76)$$

where

$$D_1(j|i) \triangleq \sum_{c \in \mathcal{C}_1(j|i)} \psi(t_i^c - t_j^c) \quad (3.77)$$

Using the convention of sums over empty index sets equal 0, we can write

$$D_1(j|i) = \begin{cases} \sum_{c \in \mathcal{C}_1(j|i)} \psi(t_i^c - t_j^c) & j \in \mathcal{J}_1(i) \\ 0 & \text{otherwise} \end{cases} \quad (3.78)$$

Figure 3.6 shows the type of cascade c that belongs to $\mathcal{C}_1(j|i)$.

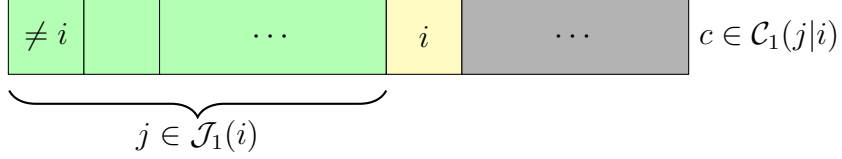


Figure 3.6: A visual depiction of a cascade c contributing to the first term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_1(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_1(i)$. For such type of cascades, i 's infection is observed but not as the first infected node. Furthermore, no parent node information is taken into account.

3.3.4.2 The Second Log-Likelihood Term

Similarly, let us define the sets

$$\mathcal{C}_2(j|i) \triangleq \{c \in \mathcal{C} : i \in \mathcal{I}_{o \setminus 1}^c(T), \text{pa}_i^c = j\} \quad (3.79)$$

$$\mathcal{J}_2(i) \triangleq \{j \in \mathcal{N} : \exists c \in \mathcal{C}_2(j|i)\} \quad (3.80)$$

The first set, $\mathcal{C}_2(j|i)$, encompasses all indices of cascades in \mathcal{H}_T , for which j is the observed parent node of infected node i , while $\mathcal{J}_2(i)$ contains all node indices, across all cascades in \mathcal{H}_T , that are observed parents of i , when i is infected in an event other than the first one. This last constraint can be dispensed with due to the earlier convention C.1, *i.e.*,

$$j_1^c = i \quad \Rightarrow \quad \text{pa}_i^c = u \quad (\text{C.1})$$

Furthermore, notice that, for binary-event cascades of the form $\{t_{j_1^c}, t_i^c\}$, *i.e.*, when i 's infection is recorded as the second event in a cascade c , then, due to lack of any alternative, $\text{pa}_i^c = j_1^c$, *i.e.*, the node j_1^c is the parent node of i . Hence, we will

count on the convention that

$$j_2^c = i \quad \Rightarrow \quad \text{pa}_i^c = j_1^c \quad (\text{C.2})$$

Note that $\mathcal{J}_2(i) = \emptyset$ only in the following 2 cases: (i) no parents of i 's infection have been observed. Due to (C.2), this would imply that \mathcal{H}_T does not include any binary-event cascades with $j_2^c = i$; (ii) node i survives all cascades in \mathcal{H}_T , *i.e.*, its infection is never observed.

Based on these sets the second term of (3.73) can be written as

$$\ell_2(\mathbf{a}_i | \mathcal{H}_T) = \sum_{j \in \mathcal{J}_2(i)} D_2(j|i) \ln a_{j,i} \quad (3.81)$$

where

$$D_2(j|i) \triangleq \begin{cases} |\mathcal{C}_2(j|i)| & j \in \mathcal{J}_2(i) \\ 0 & \text{otherwise} \end{cases} \quad (3.82)$$

Figure 3.7 shows the type of cascade c that would belong to $\mathcal{C}_2(j|i)$.

3.3.4.3 The Third Log-Likelihood Term

While there is particular simplification that can be done to $\ell_3(\mathbf{a}_i | \mathcal{H}_T)$ at this point, it is useful to determine the set of j indices it involves, as this determines, in turn, which elements of \mathbf{a}_i take part in it. The answer is supplied by the following sets:

$$\mathcal{C}_3(j|i) \triangleq \{c \in \mathcal{C} : i \in \mathcal{I}_{u, \setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)\} \quad (3.83)$$

$$\mathcal{J}_3(i) \triangleq \{j \in \mathcal{N} : \exists c \in \mathcal{C}_3(j|i)\} \quad (3.84)$$

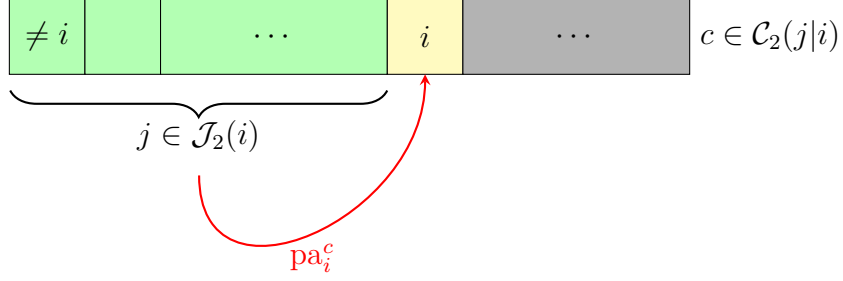


Figure 3.7: A visual template of a cascade c contributing to the second term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_2(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_2(i)$. The red arrow conveys that one of the nodes j is the observed parent of i in this cascade. Again, for such type of cascades, i 's infection is observed but not as the first infected node.

Above, $\mathcal{C}_3(j|i)$ contains all indices c of cascades in \mathcal{H}_T , where node i 's infection has been observed, but not its parent node, it is not the cascade's first infection, and node j is infected before i . Hence, $\mathcal{J}_3(i)$ accrues all indices j from all cascades in \mathcal{H}_T that are infected before i , when i 's infection has been observed, its parent has not been observed and it is not the first node to get infected. It is interesting to note that, due to (C.2), $\mathcal{J}_3(i)$ will not contain any binary-event cascades c with $j_2^c = i$. This latter fact will prove consequential, when we discuss NETRATE's EM-based training algorithm. Finally, it is, perhaps, worth noting that $\mathcal{J}_3(i) = \emptyset$, if and only if in every cascade of \mathcal{H}_T (i) when i 's infection has been observed, its parent node has not been observed or (ii) i 's infection is not observed.

Figure 3.8 shows the type of cascade c that would belong to $\mathcal{C}_3(j|i)$.

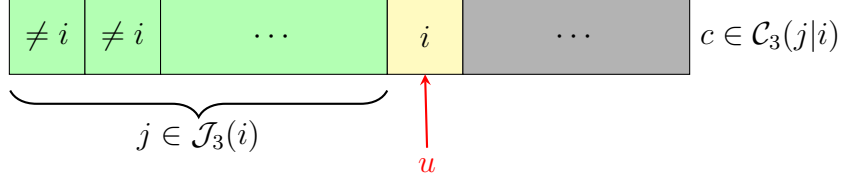


Figure 3.8: A visual template of a cascade c contributing to the third term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_3(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. In this case, i 's infection has been observed. Highlighted in green are the node indices that will belong to $\mathcal{J}_3(i)$. The red arrow conveys that i 's parent has not been observed in this cascade. This necessarily means that $j_1^c, j_2^c \neq i$, *i.e.*, it was neither the first or second infected node in the cascade.

3.3.4.4 The Fourth Log-Likelihood Term

Similarly to the previous terms, it is helpful to define the sets

$$\mathcal{C}_4(j|i) \triangleq \{c \in \mathcal{C} : i \notin \mathcal{I}^c(T), j \in \mathcal{I}^c(T)\} \quad (3.85)$$

$$\mathcal{J}_4(i) \triangleq \{j \in \mathcal{N} : \exists c \in \mathcal{C}_4(j|i)\} \quad (3.86)$$

Here, $\mathcal{C}_4(j|i)$ encompasses the cascade indices c , for which i 's infection is not observed, while node j 's is. Also, $\mathcal{J}_4(i)$ includes all node indices j that take part in any cascade that is survived by i . Given these two sets, the fourth log-likelihood term of (3.73) can be written as

$$\ell_4(\mathbf{a}_i|\mathcal{H}_T) = - \sum_{j \in \mathcal{J}_4(i)} a_{j,i} D_4(j|i) \quad (3.87)$$

where

$$D_4(j|i) \triangleq \begin{cases} \sum_{c \in \mathcal{C}_4(j|i)} \psi(T^c - t_j^c) & j \in \mathcal{J}_4(i) \\ 0 & \text{otherwise} \end{cases} \quad (3.88)$$

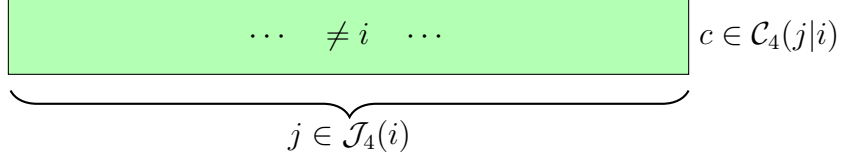


Figure 3.9: A visual template of a cascade c contributing to the second term of the i^{th} sub-problem's log-likelihood, *i.e.*, $c \in \mathcal{C}_4(j|i)$. The figure depicts the indices of infected nodes as cells of an array sorted in chronological order of infection. Highlighted in green are the node indices that will belong to $\mathcal{J}_4(i)$. This is a cascade, where i 's infection is not observed by the right-censoring time T .

Figure 3.9 shows the type of cascade c that would belong to $\mathcal{C}_4(j|i)$.

Having examined each one of the log-likelihood terms and which type of cascades contribute to them, Table 3.2 summarizes the contributions of each type of cascade to the three non-zero cases of (3.72).

3.3.4.5 Final Form of Log-Likelihood

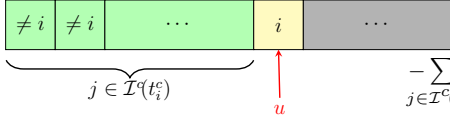
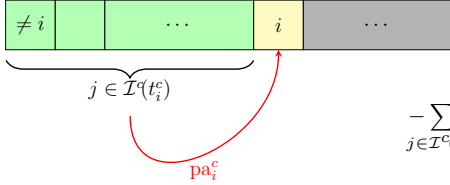
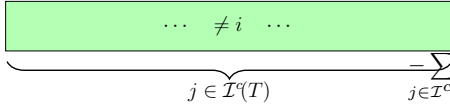
Given our finding thus far, (3.73) can now be written (up to a constant) as

$$\begin{aligned}
\ell(\mathbf{a}_i|\mathcal{H}_T) &= \ell_1(\mathbf{a}_i|\mathcal{H}_T) + \ell_2(\mathbf{a}_i|\mathcal{H}_T) + \ell_3(\mathbf{a}_i|\mathcal{H}_T) + \ell_4(\mathbf{a}_i|\mathcal{H}_T) = \\
&= - \sum_{j \in \mathcal{J}_1(i)} a_{j,i} D_1(j|i) + \sum_{j \in \mathcal{J}_2(i)} D_2(j|i) \ln a_{j,i} + \\
&+ \sum_{c \in \mathcal{C}} \mathbb{I}[i \in \mathcal{I}_{u,\setminus 1}^c(T)] \ln \sum_{j \in \mathcal{N}} \mathbb{I}[j \in \mathcal{I}^c(t_i^c)] a_{j,i} \phi(t_i^c - t_j^c) - \\
&- \sum_{j \in \mathcal{J}_4(i)} a_{j,i} D_4(j|i)
\end{aligned} \tag{3.89}$$

At this point it is useful to investigate the relationships, if any, between the j 's appearing in the four log-likelihood terms we examined so far.

In particular, the most interesting relationship arises between the index sets that depend on cascades, in which i 's infection was observed, namely, $\mathcal{J}_1(i)$, $\mathcal{J}_2(i)$

Table 3.2: What terms each cascade type contributes to the log likelihood according to (3.72) for the i^{th} sub-problem. Indices of nodes are depicted as cells in an array and are assumed sorted in chronological order of infection. Red arrows indicate parentage information and a parent node index of u reflects an unknown parent. Node indices highlighted in green reflect the set of j indices summed over in the corresponding expression of the right column.

Cascade type	Contribution to $\ell(\mathbf{a}_i \mathcal{H}_T^c)$
	$-\sum_{j \in \mathcal{I}^c(t_i^c)} a_{j,i} \psi(t_i^c - t_j^c) + \ln \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j,i} \phi(t_i^c - t_j^c)$
	$-\sum_{j \in \mathcal{I}^c(t_i^c)} a_{j,i} \psi(t_i^c - t_j^c) + \ln a_{\text{pa}_i^c, i} + \text{const}$
	$-\sum_{j \in \mathcal{I}^c(T)} a_{j,i} \psi(T - t_j^c)$

and $\mathcal{J}_3(i)$. It is not difficult to come up with examples of cascade collections, in which $\mathcal{J}_2(i)$ and $\mathcal{J}_3(i)$ either share elements or do not have a common intersection. However, for a given cascade collection \mathcal{H}_T , it is always the case that

$$\mathcal{J}_2(i) \cup \mathcal{J}_3(i) = \mathcal{J}_1(i) \quad i \in \mathcal{I}_{\setminus 1}^c(T) \quad (3.90)$$

In order to show this, it suffices to show that $\mathcal{C}_2(j|i) \cup \mathcal{C}_3(j|i) = \mathcal{C}_1(j|i)$ for all pertinent i . Recall from (3.79) that the condition $c \in \mathcal{C}$ needs to meet in order to belong to $\mathcal{C}_2(j|i)$ is

$$(i \in \mathcal{I}_{o, \setminus 1}^c(T), \text{pa}_i^c = j) = (j_1^c \neq i, t_i^c < T, \text{pa}_i^c = j) \stackrel{(S1)}{=}$$

$$= (j_1^c \neq i, t_j^c < t_i^c < T, \text{pa}_i^c = j) \quad (3.91)$$

where in step (S1) we used⁵ the implication $\text{pa}_i^c = j \Rightarrow t_j^c < t_i^c$. Similarly, from (3.83), cascade c is a member of $\mathcal{C}_3(j|i)$ only if

$$(i \in \mathcal{I}_{u, \setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)) = (j_1^c \neq i, t_j^c < t_i^c < T, \text{pa}_i^c = u) \quad (3.92)$$

Thus, we obtain that

$$\begin{aligned} & (i \in \mathcal{I}_{o, \setminus 1}^c(T), \text{pa}_i^c = j) \cup (i \in \mathcal{I}_{u, \setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)) = \\ & \stackrel{(3.91), (3.92)}{=} (j_1^c \neq i, t_j^c < t_i^c < T, \text{pa}_i^c = j) \underbrace{((\text{pa}_i^c = j) \cup (\text{pa}_i^c = u))}_{=\text{true}} = \\ & = (j_1^c \neq i, t_j^c < t_i^c < T) = (i \in \mathcal{I}_{\setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)) \end{aligned} \quad (3.93)$$

Hence, from the last result, we get

$$\mathcal{C}_2(j|i) \cup \mathcal{C}_3(j|i) \stackrel{(3.93)}{=} \{c \in \mathcal{C} : i \in \mathcal{I}_{\setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)\} \stackrel{(3.74)}{=} \mathcal{C}_1(j|i)$$

which is what was to be shown.

As a quick note is worth mentioning that $\mathcal{J}_2(i)$ and $\mathcal{J}_3(i)$ may or may have not any common elements; in this sense, they are unrelated. The same thing applies to $\mathcal{J}_4(i)$ and any of the other three sets.

We will end this subsection with determining the elements of \mathbf{a}_i , which $\ell(\mathbf{a}_i|\mathcal{H}_T)$

⁵Hence, it is important to validate any cascade collection used for training by ensuring that $t_{\text{pa}_i^c}^c < t_i^c$, whenever pa_i^c is observed.

depends on. Let's refer to this set as $\mathcal{D}(i)$. Clearly, $\mathcal{D}(i)$ can be computed as

$$\begin{aligned} \mathcal{D}(i) &\triangleq \mathcal{J}_1(i) \cup \underbrace{\mathcal{J}_2(i) \cup \mathcal{J}_3(i) \cup \mathcal{J}_4(i)}_{\stackrel{(3.90)}{=} \mathcal{J}_1(i)} = \mathcal{J}_1(i) \cup \mathcal{J}_4(i) \quad \Rightarrow \\ \Rightarrow \quad \mathcal{D}(i) &= \{j \in \mathcal{N} : \exists c \in \mathcal{C}_1(j|i) \cup \mathcal{C}_4(j|i)\} \end{aligned} \quad (3.94)$$

From (3.74) and (3.74), the latter union of $\mathcal{C}_1(j|i)$ and $\mathcal{C}_4(j|i)$ contains the cascades $c \in \mathcal{C}$ s.t.

$$\begin{aligned} &\left(i \in \mathcal{I}_{\setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c) \right) \cup \left(i \notin \mathcal{I}^c(T), j \in \mathcal{I}^c(T) \right) = \\ &= \left(j_1^c \neq i, t_j^c < t_i^c < T \right) \cup \left(t_j^c < T < t_i^c \right) \stackrel{(S1)}{=} \\ &= (j_1^c \neq i) \left[(t_j^c < t_i^c < T) \cup (t_j^c < T < t_i^c) \right] = \\ &= \left(j_1^c \neq i, t_j^c < t_i^c \wedge T \right) = \mathcal{I}_{\setminus 1}^c(t_i^c \wedge T) \end{aligned} \quad (3.95)$$

where in step (S1) we used the implication $T < t_i^c \Rightarrow j_1^c \neq i$. Therefore, the set sought after is given as

$$\mathcal{D}(i) \triangleq \left\{ j \in \mathcal{N} : \exists c \in \mathcal{C} \text{ s.t. } j \in \mathcal{I}_{\setminus 1}^c \left(\min \left\{ t_i^j, T \right\} \right) \right\} = \mathcal{J}_1(i) \cup \mathcal{J}_4(i) \quad (3.96)$$

and contains all (across cascades) indices j of nodes, whose observed infection time never appears after i 's observed infection time. This implies that $\ell_i(\mathbf{a}_i|\mathcal{H}_T)$ does not depend on any $a_{j,i}$, for which $j \notin \mathcal{D}(i)$. For example, it never depends on $a_{i,i}$, since $i \notin \mathcal{D}(i)$ for any $i \in \mathcal{N}$. A sufficiently rich collection of cascades \mathcal{H}_T would be such that $\mathcal{D}(i) = \mathcal{N} \setminus \{i\}$. Otherwise, \mathcal{H}_T will be uninformative for the transmission rates $a_{j,i}$ for $j \notin \mathcal{D}(i)$, when attempting to estimate them from the data. Finally, notice that $\mathcal{D}(i) = \emptyset$ if and only if $j_1^c = i$ for all cascades $c \in \mathcal{C}$.

Chapter 4

NETRATE Training

For the given model, we need to extract some information from the data. Based on the likelihood equation derived in the previous section, we design a learning algorithm to update these parameters so that they reach the optimal value. Before we find a learning algorithm, we need to slightly modify the likelihood to avoid over-fitting as shown in the next section.

Given a training set \mathcal{H}_T consisting of C cascades, NETRATE's parameters, *i.e.*, its transmission matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, can be estimated via the MLE method. The former is accomplished by solving the following constrained optimization problem:

$$\max_{\mathbf{A} \in \mathbb{R}_+^{N \times N}} \ell(\mathbf{A}|\mathcal{H}_T) \quad (4.1)$$

Due to (3.66) the previous problem can be decomposed over the network's nodes and is equivalent to the following set of N independent problems:

$$\max_{\mathbf{a}_i \in \mathbb{R}_+^N} \ell(\mathbf{a}_i|\mathcal{H}_T) \quad i \in \mathcal{N} \quad (4.2)$$

where $\ell_i(\mathbf{a}_i|\mathcal{H}_T)$ is given by (3.73).

4.1 Regularized Likelihood

Nevertheless, in many real-world settings, available cascade data depicting the dynamics of a network may be quite limited relative to the size of the network, *i.e.*, the network's number of nodes. This results to small (in terms of cardinality) sets $\mathcal{D}(i)$ w.r.t. N , which implies that many transmission rates will not be estimatable from the data and, hence, will remain undetermined. Obviously, this would lead to a clearly over-fitted infection model for the corresponding nodes i . Hence, instead of considering the previous problem, we will consider a penalized version of it, which employs an ℓ_1 -norm regularizer as in

$$\max_{\mathbf{a}_i \in \mathbb{R}_+^N} \underbrace{[\ell(\mathbf{a}_i|\mathcal{H}_T) - \nu_i \|\mathbf{a}_i\|_1]}_{\ell^R(\mathbf{a}_i|\mathcal{H}_T) \triangleq} \quad i \in \mathcal{N} \quad (4.3)$$

where $\nu_i > 0$ is a penalty (hyper-)parameter of the model. Under a Bayesian inference prism, it can be thought of as a Maximum A-Posteriori estimation problem, where an independent Laplace (a.k.a. double-exponential) prior distribution is put on every element of \mathbf{a}_i with a common scale parameter $1/\nu_i$.

The choice of ℓ_1 -norm regularization (also referred to as lasso regression) is motivated by its sparsity-inducing properties (for example, see [35]); admittedly, though, the primary reason behind this choice is mathematical convenience, as it will soon become apparent. High values of ν_i force the transmission rates $a_{j,i}$ to tend to 0 and, thus, simplify the model by reducing, in some sense, node inter-connectivities. At the same time, since this norm is concave-up in \mathbf{a}_i , using it

as a regularizer as shown above maintains the convex nature of the optimization problem. Now, notice that, by solving the aforementioned problem for each i , we are solving the following overall problem:

$$\max_{\mathbf{A} \in \mathbb{R}_+^{N \times N}} \underbrace{\left[\ell(\mathbf{A}|\mathcal{H}_T) - \sum_{i \in \mathcal{N}} \nu_i \|\mathbf{a}_i\|_1 \right]}_{\ell^R(\mathbf{A}|\mathcal{H}_T) \triangleq} \quad (4.4)$$

i.e., the overall problem uses a weighted ℓ_1 -norm regularizer.

As we discussed in the previous section, $\ell(\mathbf{a}_i|\mathcal{H}_T)$ only depends on $a_{j,i}$'s, for which $j \in \mathcal{D}(i)$. Hence, if we define $\mathbf{a}_{\mathcal{D}(i)}$ and $\mathbf{a}_{\mathcal{D}(i)^c}$ as the parameter vectors with elements $\{a_{j,i}\}_{j \in \mathcal{B}(i)}$ and $\{a_{j,i}\}_{j \notin \mathcal{B}(i)}$ respectively, Prob. (4.3) can be expressed more specifically as

$$\begin{aligned} & \max_{\mathbf{a}_i \in \mathbb{R}_+^N} \underbrace{\left[\ell(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) - \nu_i \|\mathbf{a}_{\mathcal{D}(i)}\|_1 - \nu_i \|\mathbf{a}_{\mathcal{D}(i)^c}\|_1 \right]}_{\equiv \ell^R(\mathbf{a}_i|\mathcal{H}_T)} = \\ & = \max_{\mathbf{a}_{\mathcal{D}(i)} \succcurlyeq \mathbf{0}} \underbrace{\left[\ell(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) - \nu_i \|\mathbf{a}_{\mathcal{D}(i)}\|_1 \right]}_{\ell^R(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) \triangleq} - \min_{\mathbf{a}_{\mathcal{D}(i)^c} \succcurlyeq \mathbf{0}} \nu_i \|\mathbf{a}_{\mathcal{D}(i)^c}\|_1 \quad i \in \mathcal{N} \end{aligned} \quad (4.5)$$

Clearly, for any $\nu_i > 0$, the optimal value for $\mathbf{a}_{\mathcal{D}(i)^c}$ is given as

$$\mathbf{a}_{\mathcal{D}(i)^c}^* = \mathbf{0} \quad \nu_i > 0 \quad (4.6)$$

and the problem that remains to be solved becomes

$$\begin{aligned} & \max_{\mathbf{a}_{\mathcal{D}(i)} \succcurlyeq \mathbf{0}} \ell^R(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) \equiv \max_{\mathbf{a}_{\mathcal{D}(i)} \succcurlyeq \mathbf{0}} \left[\ell(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) - \nu_i \|\mathbf{a}_{\mathcal{D}(i)}\|_1 \right] = \\ & = \max_{\mathbf{a}_{\mathcal{D}(i)} \succcurlyeq \mathbf{0}} \left[\ell(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T) - \nu_i \sum_{j \in \mathcal{D}(i)} a_{j,i} \right] \quad i \in \mathcal{N} \end{aligned} \quad (4.7)$$

Since $\ell(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T)$ above is the same as $\ell(\mathbf{a}_i|\mathcal{H}_T)$ in (3.89), the objective function $\ell_i^R(\mathbf{a}_{\mathcal{D}(i)}|\mathcal{H}_T)$ of the last problem takes now the form

$$\begin{aligned}
\ell_i^R(\mathbf{a}_i|\mathcal{H}_T) &\stackrel{(3.89)}{=} - \sum_{j \in \mathcal{J}_1(i)} a_{j,i} D_1(j|i) + \sum_{j \in \mathcal{J}_2(i)} D_2(j|i) \ln a_{j,i} + \\
&+ \sum_{c \in \mathcal{C}} \mathbb{I}[i \in \mathcal{I}_{u,\setminus 1}^c(T)] \ln \sum_{j \in \mathcal{N}} \mathbb{I}[j \in \mathcal{I}^c(t_i^c)] a_{j,i} \phi(t_i^c - t_j^c) - \\
&- \sum_{j \in \mathcal{J}_4(i)} a_{j,i} D_4(j|i) - \underbrace{\nu_i \sum_{j \in \mathcal{D}(i)} a_{j,i}}_{\ell_5(\mathbf{a}_i|\mathcal{H}_T) \triangleq} \quad (4.8)
\end{aligned}$$

It will be helpful in our subsequent analysis to consider a few extreme cases of \mathcal{H}_T and their effect on the maximization of $\ell^R(\mathbf{a}_i|\mathcal{H}_T)$.

Case I (most uninformative): Assume that \mathcal{H}_T contains cascades, for which $j_1^c = i$ for all $c \in \mathcal{C}$. In that case, as we recently discussed, this directly implies $\mathcal{D}(i) = \emptyset$ all terms/sums vanish from (4.8) and $\ell^R(\mathbf{a}_i|\mathcal{H}_T) = 0$ for all $\mathbf{a}_i \in \mathbb{R}^N$. In other words, the entire \mathbf{a}_i vector cannot be estimated from the data. By a convention, though, that we will adopt, we will assume that, in this case, $\mathbf{a}_i^* = \mathbf{0}$ to reflect the fact that i 's observed infections are due to causes exogenous to the network.

Case II (no observed infections): Assume, now, that \mathcal{H}_T consists only of cascades, for which node i 's infection was never observed; in other words, i survives all cascades of \mathcal{H}_T . In that case, again based on a previous discussion, $\mathcal{J}_1(i) = \emptyset$ and, due to (3.90), $\mathcal{J}_2(i) = \mathcal{J}_3(i) = \emptyset$ with the associated terms of $\ell^R(\mathbf{a}_i|\mathcal{H}_T)$ vanishing. Additionally, due to (3.96), we will have $\mathcal{D}(i) = \mathcal{J}_4(i)$ and it is easy to verify that $\mathcal{J}_4(i) \neq \emptyset$, if \mathcal{H}_T consists of at least one cascade. Under all these

circumstances, the regularized log-likelihood becomes:

$$\begin{aligned}
(4.8) \Rightarrow \quad \ell^R(\mathbf{a}_i|\mathcal{H}_T) &= - \sum_{j \in \mathcal{J}_4(i)} a_{j,i} D_4(j|i) - \nu_i \sum_{j \in \mathcal{D}(i)=\mathcal{J}_4(i)} a_{j,i} \\
&= - \sum_{j \in \mathcal{J}_4(i)} a_{j,i} [D_4(j|i) + \nu_i] \tag{4.9}
\end{aligned}$$

Since all $D_4(j|i) > 0$ almost surely due to (3.88), the coefficients appearing in (4.9) are all positive. Hence, the maximum of $\ell^R(\mathbf{a}_i|\mathcal{H}_T)$ (which equals 0) is attained for $\mathbf{a}_{\mathcal{D}(i)}^* = \mathbf{0}$.

4.2 Regarding Gradient Method

Based on the log-likelihood equation, we can derive the gradients and use it to do a gradient descent to approach the solution. However, the constraint $a_{ji} > 0$ introduces a problem. In a general projected gradient descent approach, whenever we reach a region of infeasibility, we project the solution back into the feasible region. In this case, whenever $a_{ji} > 0$, we set $a_{ji} = 0$. This sets in in a loop where solution keeps escaping the feasible region. A way to deal with this is by using a line search method wherein we keep searching for learning rate for descent step until the log-likelihood increases. However, this is very expensive and slows the training. Therefore, we derive the EM algorithm which works on a minorizer of the log-likelihood.

4.3 EM Algorithm

As a notational convention, primed quantities are assumed to be computed using the estimated parameter values of the EM algorithm's previous iteration.

Based on earlier discussion, we can lower-bound the inner summation of $\ell_3(\mathbf{a}_i|\mathcal{H}_T)$ as

$$\ln \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j,i} \phi(t_i^c - t_j^c) \geq \sum_{j \in \mathcal{I}^c(t_i^c)} (p_{j,i}^c)' \ln a_{j,i} + \text{const} \quad (4.10)$$

where const is independent of \mathbf{a}_i . Also, $(p_{j,i}^c)'$ stands for the probability of j being the parent node of node i , if i was observed to be infected at time t_i^c (also, see (3.55)):

$$(p_{j,i}^c)' \triangleq \frac{a'_{j,i} \phi(t_i^c - t_j^c)}{\sum_{k \in \mathcal{I}^c(t_i^c)} a'_{k,i} \phi(t_i^c - t_k^c)} \quad j \in \mathcal{I}^c(t_i^c) \quad (4.11)$$

Given (4.10), the third term of the regularized log-likelihood is lower-bounded by

$$\begin{aligned} \bar{\ell}_3(\mathbf{a}_i|\mathbf{a}'_i, \mathcal{H}_T) &\triangleq \sum_{j \in \mathcal{N}} \ln a_{j,i} \sum_{c \in \mathcal{C}} \mathbb{I}[i \in \mathcal{I}_{u,\setminus 1}^c(T), j \in \mathcal{I}^c(t_i^c)] (p_{j,i}^c)' + \text{const} = \\ &= \sum_{j \in \mathcal{J}_3(i)} D'_3(j|i) \ln a_{j,i} + \text{const} \end{aligned} \quad (4.12)$$

where

$$D'_3(j|i) \triangleq \begin{cases} \sum_{c \in \mathcal{C}_3(j|i)} (p_{j,i}^c)' & j \in \mathcal{J}_3(i) \\ 0 & \text{otherwise} \end{cases} \quad (4.13)$$

Although $D'_3(j|i)$ depends on \mathbf{a}'_i , we do not show this dependence explicitly. (4.12)

prompts the following EM-stemming minorizer¹ of $\ell^R(\mathbf{a}_i|\mathcal{H}_T)$:

$$\begin{aligned}
\bar{\ell}^R(\mathbf{a}_i|\mathbf{a}'_i, \mathcal{H}_T) &\triangleq \ell_1(\mathbf{a}_i|\mathcal{H}_T) + \ell_2(\mathbf{a}_i|\mathcal{H}_T) + \bar{\ell}_3(\mathbf{a}_i|\mathbf{a}'_i, \mathcal{H}_T) + \ell_4(\mathbf{a}_i|\mathcal{H}_T) + \ell_5(\mathbf{a}_i|\mathcal{H}_T) = \\
&= - \sum_{j \in \mathcal{J}_1(i)} a_{j,i} D_1(j|i) + \sum_{j \in \mathcal{J}_2(i)} D_2(j|i) \ln a_{j,i} + \sum_{j \in \mathcal{J}_3(i)} D'_3(j|i) \ln a_{j,i} - \\
&\quad - \sum_{j \in \mathcal{J}_4(i)} a_{j,i} D_4(j|i) - \nu_i \sum_{j \in \mathcal{D}(i)} a_{j,i} + \text{const}
\end{aligned} \tag{4.14}$$

If we have a cascade collection \mathcal{H}_T , such that $\mathcal{D}(i) \neq \emptyset$ and $\mathcal{J}_1(i) \neq \emptyset$, *i.e.*, we do not fall into the cases I and II we examined earlier, then, the optimal value $a_{k,i}^*$ for node $k \in \mathcal{D}(i)$, which maximizes $\bar{\ell}^R(\mathbf{a}_i|\mathbf{a}'_i, \mathcal{H}_T)$ is computed from (4.14) as

$$a_{k,i}^* = \frac{D_2(k|i) + D'_3(k|i)}{\nu_i + D_4(k|i) + D_1(k|i)} \quad k \in \mathcal{D}(i) \tag{4.15}$$

and, when $\nu_i > 0$ as discussed earlier,

$$a_{k,i}^* = 0 \quad k \notin \mathcal{D}(i) \tag{4.16}$$

The sequence of solutions produced by the EM algorithm increases the likelihood and converges to some optimal value [36]. The EM algorithm can be divided into independent sub-problems as discussed earlier. For each sub-problem $i \in \mathcal{N}$, the EM algorithm is divided into two stages: Setup and Updates. The advantage of this becomes evident when working with a large number of cascades. An EM setup for each node sub-problem ensures that it sees only the cascades relevant to the node. This also enables the algorithm to run in parallel more efficiently since all the sub problems have all the relevant sets stored in their local memory.

¹Such a minorizer is known as the Q function in the EM literature.

4.3.1 EM Setup

The setup involves generating The following sets.

1. The node index sets: $\mathcal{J}_1(i)$, $\mathcal{J}_2(i)$, $\mathcal{J}_3(i)$ and $\mathcal{J}_4(i)$
2. The sets with pre-calculated ϕ, ψ values: $D_1(j|i)$, $D_2(j|i)$ and $D_3(j|i) \quad \forall j \in \mathcal{N}$
3. The cascade index set \mathcal{C}_3 . This is because the update equation for $D_3(j|i)$ $\forall j \in \mathcal{N}$ itself depends on the previous value of α_i vector. As a result, $\mathcal{C}_3(j | i)$ must be used to calculate $D_3(j|i)$ at every iteration of the update. We store \mathcal{C}_3 as a dictionary so that it can be iterated through to obtain the value of $D_3(j|i)$ per EM update iteration.

Algorithm 1 describes the pseudo-code for generating these sets. Keep in mind that the setup has to be performed for both the training and the validation set separately. This is because log-likelihood calculations for the validation set will need these pre-calculated sets.

4.3.2 EM updates

The update algorithm is designed to run efficiently mainly because all the required sets have been pre-calculated and simply involves updating the values of α_i given by the EM update equation. The updates are designed to stop when $\Delta\alpha_i$ is less than a given tolerance ϵ . Algorithm 2 details the update method for α_i .

4.4 Evaluating the Log-Likelihood

The EM setup also gives us the opportunity to quickly evaluate the log-likelihood. In order to find the best regularizer, it is done by using a grid search on a set of

Algorithm 1 NETRATE EM Setup

Input: cascades: C , Subproblem node ID: \mathbf{i} , Memory Kernel $\phi(t_i, t_j)$, Integrated Memory Kernel $\psi(t_i, t_j)$, Node set: \mathcal{N} , Right Censoring Time : T
 $\mathcal{J}_1(i) \leftarrow \emptyset, \mathcal{J}_2(i) \leftarrow \emptyset, \mathcal{J}_3(i) \leftarrow \emptyset, \mathcal{J}_4(i) \leftarrow \emptyset, \mathcal{D}(i) \leftarrow \emptyset, \mathcal{C}_3 \leftarrow \emptyset, \mathcal{D} \leftarrow \emptyset$
 $D_1(j | \mathbf{i}) = 0, D_2(j | \mathbf{i}) = 0, D_4(j | \mathbf{i}) = 0 \forall j \in \mathcal{N}$
for $cas \in C$ **do**
 $\mathcal{D} \leftarrow \{j \mid j \text{ is an infected node in } cas\}$
 if node \mathbf{i} has survived in cas **then**
 $j_{found} = \{j \mid j \text{ is an infected node in } cas\}$
 $\mathcal{J}_4(i) \leftarrow j_{found}$
 for $j \in j_{found}$ **do**
 $D_4(j | i) \leftarrow D_4(j | i) + \psi(T - t_j)$
 end for
 end if
 if node \mathbf{i} is infected in cas and first infection of cascade $\neq \mathbf{i}$ **then**
 $j_{found} \leftarrow \{j \mid j \text{ is infected before } \mathbf{i} \text{ in } cas\}$
 $\mathcal{J}_1(i) \leftarrow j_{found}$
 for $j \in j_{found}$ **do**
 $D_1(j | i) \leftarrow D_1(j | i) + \psi(t_i - t_j)$
 end for
 if node \mathbf{i} has a parent event in cas **then**
 $j_{found} \leftarrow \{j \mid j \text{ is the infected parent node of } \mathbf{i} \text{ in } cas\}$
 $\mathcal{J}_2(i) \leftarrow j_{found}$
 for $j \in j_{found}$ **do**
 $D_2(j | i) \leftarrow D_2(j | i) + \psi(t_i - t_j)$
 end for
 else
 $j_{found} \leftarrow \{j \mid j \text{ is infected before } \mathbf{i} \text{ in } cas\}$
 $\mathcal{J}_3(i) \leftarrow j_{found}$
 for $j \in j_{found}$ **do**
 $\mathcal{C}_3(cas).append(j, \phi(t_i - t_j))$
 end for
 end if
 end if
end for
Output: $\mathcal{J}_1(i), \mathcal{J}_2(i), \mathcal{J}_3(i), \mathcal{J}_4(i), \mathcal{C}_3, D_1(j | i), D_2(j | i), D_4(j | i) \forall j \in \mathcal{N}$.

Algorithm 2 NETRATE EM Updates

Input: cascades: C , Subproblem node ID: \mathbf{i} , Node set: \mathcal{N} , Right Censoring Time : T , Regularization coefficient: ν , Number of EM iterations: $iters$, alpha value tolerance: ϵ , initial alpha values: a_i , \mathcal{D} , $\mathcal{J}_1(i)$, $\mathcal{J}_2(i)$, $\mathcal{J}_3(i)$, $\mathcal{J}_4(i)$, \mathcal{C}_3 , $D_1(j | \mathbf{i})$, $D_2(j | \mathbf{i})$, $D_4(j | \mathbf{i}) \forall j \in \mathcal{N}$

for $iteration \in iters$ **do**

$a_{i,previous} \leftarrow a_i$

for $j \in \mathcal{N}$ **do**

if $j \in \mathcal{D}$ **then**

$a_{ji-num} \leftarrow 0$

$a_{ji-den} \leftarrow \nu$

if $j \in \mathcal{J}_1(i)$ **then**

$a_{ji-den} \leftarrow a_{ji-den} + D_1(j | \mathbf{i})$

end if

if $j \in \mathcal{J}_2(i)$ **then**

$a_{ji-num} \leftarrow a_{ji-num} + D_2(j | \mathbf{i})$

end if

if $j \in \mathcal{J}_4(i)$ **then**

$a_{ji-den} \leftarrow a_{ji-den} + D_4(j | \mathbf{i})$

end if

if $j \in \mathcal{J}_3(i)$ **then**

$D_3(j | \mathbf{i}) \leftarrow 0$

for $cas \in \mathcal{C}_3$ **do**

if j is infected in cas **then**

$p_{ji-den} \leftarrow 0$

$j_{found} \leftarrow \{ j | j \text{ is infected before } \mathbf{i} \text{ in } cas \}$

for $k \in j_{found}$ **do**

$p_{ji-den} \leftarrow p_{ji-den} + a_i[k] \phi(t_i - t_k)$

if $k = j$ **then**

$p_{ji-num} \leftarrow a_i[k] \phi(t_i - t_k)$

end if

end for

$D_3(j | i) \leftarrow D_3(j | i) + \frac{p_{ji-num}}{p_{ji-den}}$

end if

end for

$a_{ji-num} \leftarrow a_{ji-num} + D_3(j | \mathbf{i})$

end if

$a_i[j] \leftarrow \frac{a_{ji-num}}{a_{ji-den}}$

else

$a_i[j] \leftarrow 0$

end if

end for

if $\|a_{i,previous} - a_i\|_\infty$ **then**

break

end if

end for

Output: a_i

predetermined ν values. The best ν value is set to be the one which maximizes the validation log likelihood. Algorithm 3 shows the method for evaluating the regularized log likelihood. This is not the evaluation for the minorizer.

Algorithm 3 NETRATE Log-Likelihood

Input: cascades: C , Subproblem node ID: \mathbf{i} , Node set: \mathcal{N} , Right Censoring Time : T , Regularization coefficient: ν , Number of EM iterations: iters , transmission rate vector: a_i , \mathcal{D} , $\mathcal{J}_1(i)$, $\mathcal{J}_2(i)$, $\mathcal{J}_3(i)$, $\mathcal{J}_4(i)$, \mathcal{C}_3 , $D_1(j | \mathbf{i})$, $D_2(j | \mathbf{i})$, $D_4(j | \mathbf{i}) \forall j \in \mathcal{N}$

$\logLikelihood = 0$

for $j \in \mathcal{N}$ **do**

if $j \in \mathcal{D}$ **then**

$\logLikelihood \leftarrow \logLikelihood - \nu a_i[j]$

end if

if $j \in \mathcal{J}_1(i)$ **then**

$\logLikelihood \leftarrow \logLikelihood - D_1(j | \mathbf{i}) a_i[j]$

end if

if $j \in \mathcal{J}_2(i)$ **then**

$\logLikelihood \leftarrow \logLikelihood - D_2(j | \mathbf{i}) \ln a_i[j]$

end if

if $j \in \mathcal{J}_4(i)$ **then**

$\logLikelihood \leftarrow \logLikelihood - D_4(j | \mathbf{i}) a_i[j]$

end if

if $j \in \mathcal{J}_3(i)$ **then**

for $\text{cas} \in \mathcal{C}_3$ **do**

$\text{totalIntensityCascade} \leftarrow 0$

$j_{\text{found}} \leftarrow \{ j \mid j \text{ is infected before } \mathbf{i} \text{ in cas} \}$

for $k \in j_{\text{found}}$ **do**

$\text{totalIntensityCascade} \leftarrow \text{totalIntensityCascade} + a_i[j] \phi(t_i - t_k)$

$\triangleright \phi(t_i - t_k)$ value is present in \mathcal{C}_3 after setup

end for

$\logLikelihood \leftarrow \logLikelihood + \ln(\text{totalIntensityCascade})$

end for

end if

end for

Output: \logLikelihood

4.4.1 Some Observations

In order to avoid inconsistent data samples, the following relationships between training and validation sets must be satisfied.

1. $(\mathcal{J}_{4,training} - \mathcal{J}_{1,training}) \cap \mathcal{J}_{1,validation} \neq \emptyset$
2. $\mathcal{D}(i, validation) - \mathcal{D}(i, training) = \emptyset$
3. $(\mathcal{J}_{1,validation} \cap \mathcal{J}_{4,training}) - \mathcal{J}_{validation} = \emptyset$

One can interpret these conditions as the characteristics of the fundamental set of cascades required for training. Alternatively, this can be interpreted as: all the infections that take place in the validation set must be explainable by infections taking place in the training set. This is to avoid a situation where in the training set, due to the conditions of the EM presented in section 4.3, $a_{ji} \leftarrow 0$ and there happens to be in the validation set an infection event between j and i . This might result in a $\ln 0$ situation while evaluating the likelihood of the validation set.

4.5 Experiment: Effect of Parentage Information

The effect of incorporating parentage information was evaluated by running the NETRATE algorithm on data sources simulated with varying probabilities of parentage information. 5000 cascades were created using the simulation algorithm present in chapter 6. Parent information was added with a probability. This probability was varied from 0 to 1, while the tolerance parameter ϵ for converge was fixed at 10^{-7} . Figure 4.1 shows that the number of iterations decreases as more parent

Number of iterations to converge as more parent information is added

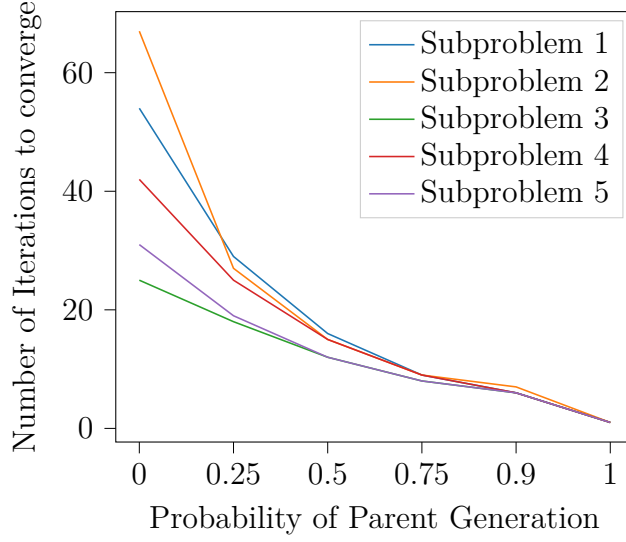


Figure 4.1: P-P plot generated using transformed times from simulation when there are no survived nodes in all cascades. Generated using 1000 data points selected at random

information is added. When the infection events from all the information cascades have parent information, we observe the NETRATE completes in a single iteration, i.e. there is a closed form solution. This agrees with our assumptions because the iterations are caused by terms present in $\mathcal{J}_3(j | i)$. So, what the parent information essentially does is that it transfers terms to $\mathcal{J}_2(j | i)$ from $\mathcal{J}_3(j | i)$. This is why the presence of parent information, even partially, would speedup the algorithm.

Chapter 5

Time Rescaling and Goodness of Fit

Once NETRATE's parameters $\{a_{ji}, j, i \in \mathcal{N}\}$ have been inferred, an important question that arises is how well the model fits the data. In the absence of an alternative model to compare to (by comparing likelihood function values of the two models, as computed on the same training set), one needs some type of estimation-stemming result, whose distribution is known. Such a result is supplied by a very specific time rescaling of infection times, derived as follows. This is similar to the time transformation using the random time change theorem [9] applied to stochastic point processes.

In Section 3.3.4, we already derived that for a single cascade c for NETRATE it holds that

$$f\left(\{t_i^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)}, \bigcap_{k \notin \mathcal{I}^c(T)} \{T_k^c > T\} \middle| t_{j_1}^c\right) = \prod_{i \in \mathcal{I}_{\setminus 1}^c(T)} f_i(t_i^c | \mathcal{H}_{t_i^c}^c) \prod_{k \notin \mathcal{I}^c(T)} S_k(T | \mathcal{H}_T^c) \quad (3.60)$$

where, if we assume that no parent node has been observed,

$$f_i(t_i^c | \mathcal{H}_{t_i^c}^c) = \lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) \exp\{-\Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c)\} \quad (5.1)$$

and

$$\prod_{k \notin \mathcal{I}^c(T)} S_k(T | \mathcal{H}_T^c) = \prod_{k \notin \mathcal{I}^c(T)} \exp\{-\Lambda_k(T | \mathcal{H}_T^c)\} \quad (5.2)$$

Consider, now, the non-linear time re-scaling transform:

$$\tau_i^c(t) \triangleq \Lambda_i(t | \mathcal{H}_t^c) \quad (5.3)$$

and the new RVs $\tau_i(t_i^c)$ for any infected node i in the cascade, except the one infected first. The Jacobian $|\frac{\partial t_i^c}{\partial \tau_i^c}|$ of the transform connecting a specific t_i^c to the corresponding $\tau_i^c(t_i^c)$ can be computed as

$$|J_i| = \frac{1}{\lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c)} \quad (5.4)$$

which yields the PDF for node i

$$f(\tau_i^c(t_i^c)) = f_i(t_i^c | \mathcal{H}_{t_i^c}^c) |J_i| \stackrel{(5.1), (5.4)}{=} \exp\{-\Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c)\} = \exp\{-\tau_i^c(t_i^c)\} \quad (5.5)$$

This means that the joint PDF of the transformed times for $i \in \mathcal{I}_{\setminus 1}^c(T)$ is

$$f\left(\{\tau_i(t_i^c)^c\}_{i \in \mathcal{I}_{\setminus 1}^c(T)} \middle| t_{j_1^c}^c\right) \stackrel{(5.5)}{=} \prod_{i \in \mathcal{I}_{\setminus 1}^c(T)} \exp\{-\tau_i^c(t_i^c)\} \quad (5.6)$$

Also, we have that

$$(5.2) \stackrel{(5.3)}{\Rightarrow} \prod_{k \notin \mathcal{I}^c(T)} S_k(T | \mathcal{H}_T^c) = \prod_{k \notin \mathcal{I}^c(T)} \exp\{-\tau_k^c(T)\} \quad (5.7)$$

Combining our previous results yields

$$\begin{aligned}
(3.60) & \stackrel{(5.6),(5.7)}{\Rightarrow} f \left(\left\{ \tau_i(t_i^c)^c \right\}_{i \in \mathcal{I}_1^c(T)}, \left\{ \tau_k^c(T) \right\}_{k \notin \mathcal{I}^c(T)} \middle| t_{j_1}^c \right) = \\
& = \prod_{i \in \mathcal{I}_1^c(T)} \exp\{-\tau_i^c(t_i^c)\} \prod_{k \notin \mathcal{I}^c(T)} \exp\{-\tau_k^c(T)\} \quad (5.8)
\end{aligned}$$

which implies that the random variables $\{\tau_i^c(t_i^c)\}_{i \in \mathcal{I}_1^c(T)}$ and $\{\tau_i^c(T)\}_{i \notin \mathcal{I}^c(T)}$ are mutually IID exponential 1-distributed. So, in summary, the following transformed variables can be computed from each cascade c and any node $i \in \mathcal{N}$:

$$\tau_i^c = \Lambda_i(t_i^c | \mathcal{H}_{t_i^c}^c) = \sum_{j \in \mathcal{I}^c(t_i^c)} a_{j,i} \psi(t_i^c - t_j^c) \quad \text{for } i \in \mathcal{I}_1^c(t_i^c) \quad (5.9)$$

$$\tau_i^c = \Lambda_i(T | \mathcal{H}_T^c) = \sum_{j \in \mathcal{I}^c(T)} a_{j,i} \psi(T - t_j^c) \quad \text{for } i \notin \mathcal{I}^c(T) \quad (5.10)$$

These are the time transformation that must be performed on the infection times in each cascade. The associated algorithm is described in 4.

Up to this point we assumed that no parent node indices are observed apart, of course, of those corresponding to nodes that get infected after the first event in every cascade. For those infections that are accompanied by observed parent node indices, some minor modifications need to be made in our previous analysis. Assume that in cascade c we observed pa_i^c for a node i . Then, the corresponding factor $f_i(t_i^c | \mathcal{H}_{t_i^c}^c)$ in (3.60) needs to be replaced with

$$f_i(t_i^c, \text{pa}_i^c | \mathcal{H}_{t_i^c}^c) \stackrel{(3.21)}{=} \mathbb{P}\left\{ \text{pa}_i^c(T_i^c) = \text{pa}_i^c | T_i^c = t_i^c, \mathcal{H}_{t_i^c}^c \right\} f_i(t_i^c | \mathcal{H}_{t_i^c}^c) \quad (5.11)$$

In essence this amounts to including factors

$\mathbb{P}\left\{ \text{pa}_i^c(T_i^c) = \text{pa}_i^c | T_i^c = t_i^c, \mathcal{H}_{t_i^c}^c \right\}$ to (3.60) above for every infection which is accom-

panied by observed parentage information except the cases of chronologically-second infections; in that case, $\mathbb{P}\{\text{pa}_i^c(T_i^c) = \text{pa}_i^c | T_i^c = t_i^c, \mathcal{H}_{t_i^c}^c\} = 1$. It also means that (5.9) applies regardless of whether parent node indices have been observed or not.

Based on (5.9) and (5.10) and the fact that both of these quantities are samples from an exponential-1, a qualitative goodness-of-fit vehicle can be construed via a Probability–Probability (P-P) plot. Assume that we have accumulated all such transformed times across all cascades and end up with L samples $\{\tau_\ell\}_{\ell=1}^L$. Then, the P-P plot consists of all pairs $\left\{\left(\frac{\ell}{L}, F(\tau_{(\ell)})\right)\right\}_{\ell=1}^L$, where $\tau_{(\ell)}$ is the ℓ^{th} sample order statistic of the τ 's (*i.e.*, the ℓ^{th} τ , if the τ 's are sorted in ascending order). Major deviations from the 45° line would imply that the trained NETRATE model, which the transformed times τ 's are connected to, lacks a good fit to the observed training data. We also use the single sample KS test statistic, which in our case is used to compare the transformed samples with the exponential-1 distribution. It is to be noted that for models involving a large number of samples, the p-value approaches zero [37]. This is due to the fact that as the number of samples increases, the p-value will be either 0 or 1. It can only be 1 if the calculated test-statistic is equal to the true value to the infinite decimal point, which due to numerical imprecision, is never possible.

Algorithm 4 Infection Time Transformation

Input: transmission matrix: $A = \{a_{ji} \mid j, i \in \mathcal{N}\}$, node ID: i , cascade cas , right Censoring time T , Integrated Memory kernel $\psi(t)$

$\tau_i^c \leftarrow 0$

if i is infected in cas **then**

for $j \mid j \in \text{infected nodes in } cas$ **do**

if $t_j < t_i$ **then**

$\tau \leftarrow \tau + a_{ji}\psi(t_i - t_j)$

end if

end for

else

for $j \mid j \in \text{infected nodes in } cas$ **do**

$\tau \leftarrow \tau + a_{ji}\psi(T - t_j)$

end for

end if

Output: transformed time τ_i^c

5.1 Regarding Goodness of Fit for Large Samples

When training the model for a large number of cascades, the learned parameters does not provide a good fit to the training cascades. This is mainly because of two situations we have encountered during training

1. When the size of a cascade is small compared to the total size of the network \mathcal{N} , i.e. a large number of nodes have survived in most cascades, the learned parameters deviate from the ground truth. This is caused by a penalty imposed on the transmission rate due to these nodes having survived these cascades. This was tested by running simulation while changing the right censoring time. (described in chapter 6). So, obviously the learned parameters in this case would provide a terrible fit.
2. The p-value for the KS test will almost surely be zero for large number of

cascades when the network is also large. This situation has been discussed in [37] by Lin et al. They claim that in the case of consistent estimators like the K-L test statistic, the limiting distribution of the estimator (say $\hat{\beta}$) has all its mass on the exact value β . This means that unless the estimated value is exactly equal to the ground truth to the infinite decimal place, the p-value will be zero. As a result, performing a goodness of fit using the P-P plot is trivial for the real world example described in chapter 9.

Chapter 6

Simulation

Simulation plays an important role in survival model. Simulating a cascade based on the transmission matrix $\{a_{ji}, i, j \in \mathcal{N}\}$ helps us investigate how the information spreads. This simulation method we employ is a slightly modified version of Ogata's Modified Thinning Algorithm [6]. Ogata's thinning algorithm is applied to single and multi-dimensional Hawkes processes. The general idea of the thinning algorithm is that we simulate samples based on an upper bound of the intensity value. Then, we accept or reject each of these points based on a thinning condition. With multi-dimensional processes, there is an extra step to decide the identity of the node that has been infected. The above mentioned procedure can be applied to a survival process by a simple modification: Once a node has been infected, its intensity after that point reduces to zero. Algorithm 5 describes the process for generating samples based on the NETRATE model.

Algorithm 5 NETRATE Simulation

Input: cascade Seed: C_{seed} , Memory Kernel $\phi(t_i, t_j)$, Node set: \mathcal{N} , Right Censoring Time : T ,
Transmission Matrix: $\{a_{ji}, i, j \in \mathcal{N}\}$ (of size $|\mathcal{N}|^2$)
 $InfectedNodes \leftarrow \{j \mid j \text{ is infected in } C_{seed}\}$
 $t \leftarrow$ Final Infection time in C_{seed}
 $Cas \leftarrow C_{seed}$
while $t < T$ & $|InfectedNodes| < |\mathcal{N}|$ **do**
 $\triangleright f(t)$ is the pdf of infection times associated with the memory kernel $\phi(t)$
 if $f'(t) \leq 0 \forall t \in [0, \infty)$ **then** $\triangleright \phi()$ is a strictly decreasing function
 $\phi \leftarrow \llbracket [i \notin InfectedNodes] \phi_i(t - t_i) \mid i \in \mathcal{N} \rrbracket$ \triangleright Vector of $\phi()$ values
 else $\triangleright \phi()$ is not a strictly decreasing function
 $\phi \leftarrow \llbracket [i \notin InfectedNodes] \phi_i(T - t_i) \mid i \in \mathcal{N} \rrbracket$ \triangleright Vector of $\phi()$ values
 end if
 $\lambda \leftarrow ComputeConditionalIntensity(\phi)$ \triangleright upper bound vector of intensities calculated as per (3.40)
 $initialIntensitySum \leftarrow \sum_{i \in \mathcal{N}} \lambda_i, i \in \mathcal{N}$
 $u_1, u_2, u_3 \sim N(\mu = 0, \sigma = 1)$
 $waitingTime = -\frac{\ln u_1}{initialIntensitySum}$
 $t \leftarrow t + waitingTime$
 $\phi \leftarrow \llbracket [i \notin InfectedNodes] \phi_i(t - t_i) \mid i \in \mathcal{N} \rrbracket$ \triangleright calculate updated ϕ
 $\lambda \leftarrow ComputeConditionalIntensity(\phi)$ \triangleright updated vector of intensities calculated as per (3.40)
 $IntensitySum \leftarrow \sum_{i \in \mathcal{N}} \lambda_i, i \in \mathcal{N}$
 if $u_2 \leq \frac{IntensitySum}{initialIntensitySum}$ **then**
 $cumSum \leftarrow 0$
 for $i \in \mathcal{N}$ **do**
 $cumSum \leftarrow cumSum + \frac{\lambda_i}{initialIntensitySum}$
 if $u_2 > cumSum$ **then**
 break
 end if
 $newInfectedNode \leftarrow i$
 $\phi \leftarrow \llbracket [i \in InfectedNodes] \phi_i(t - t_i) \mid i \in \mathcal{N} \rrbracket$ \triangleright calculate ϕ for infected nodes
 $parentProbability \leftarrow ComputeConditionalIntensities(\phi)$
 $k \leftarrow \operatorname{argmax}(u_3 \leq parentProbability)$
 $parentOfInfectingNode \leftarrow infectedNodes[k]$
 end for
 if $t < T$ **then**
 $Cas.append(newInfectedNode, t)$
 $infectedNodes.append(newInfectedNode)$
 end if
 end if
end while
Output: Cas

6.1 Simulation Scheme

The explanation for the simulation algorithm will be based on the formal proof provided in Ogata's paper [6] on the modified thinning algorithm for HAWKES process.

1. Find the intensity of the ground process $\lambda = \sum_{i=1}^{\mathcal{N}} \lambda_i$
2. Find the **upper bound intensity** of the ground process $\hat{\lambda} \geq \lambda$
3. **Simulate points** (waiting time) for the upper bound intensity $\hat{\lambda}$
4. Recalculate intensities based on new waiting time.
5. **thinning** accept points by simulation by comparing ground process of upper bound and updated intensity based on waiting time.
6. **randomly sample a node ID** of uninfected nodes with a probability proportional to the conditional intensities evaluated at that time.
7. **randomly sample a parent node ID** of infected nodes with a probability proportional to the conditional intensities evaluated at that time.
8. **repeat until** right censoring time T.

6.2 Case study: Goodness of Fit for Simulation

In order to provide a visual verification to the validity of the simulation model, We simulated 10,000 cascades for a 3 node network with a randomized ground truth alpha matrix. The simulation was run under two scenarios.

1. A large right censoring time so that none of the nodes have survived the cascades.
2. A small right censoring time, enough that some of the cascades contain survived nodes.

The simulated infection times were then transformed as described in section 5. For the first case (as shown in image 6.1) The p-value obtained is large, indicating that the null hypothesis cannot be rejected and therefore the transformed times indeed come from the exponential distribution, which agrees with the conclusion from the previous chapter 5.8 that indeed the the transformed times come from the exponential 1 distribution.

In the second case however, that does not appear to be the case. It appears the the right censoring time plays a role in the goodness of fit. We simulated cascades using the same ground truth value but with a small right censoring time, ensuring that some nodes survive some of the cascades. figure 6.2 shows the P-P plot when using 1000 sample points from the simulated cascades. We observe that when there are nodes that remain uninfected in some of the cascades. There is slight deviation from the 45° , however, it still provides a reasonable fit to the exponential-1 distribution.

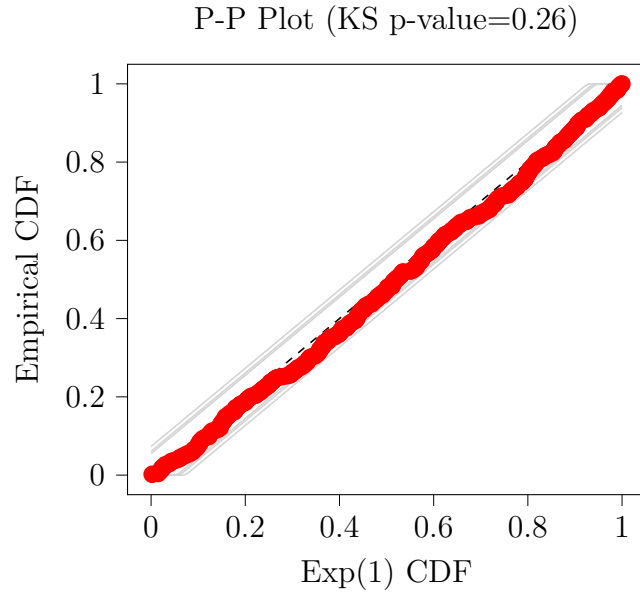


Figure 6.1: P-P plot generated using transformed times from simulation when there are no survived nodes in all cascades. Generated using 1000 data points selected at random

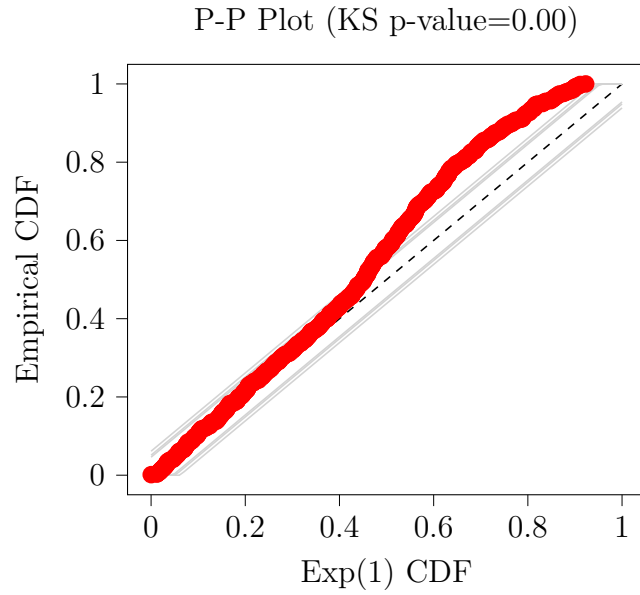


Figure 6.2: P-P plot generated using transformed times from simulation when cascades contain some survived nodes. Generated using 1000 data points selected at random

Chapter 7

Predicting Dynamic Virality

NETRATE models the infection time distribution between two nodes in the network. So, we can extract useful insights regarding the time of infection. More specifically, we shall attempt to find the dynamic virality of an information cascade. Assume we are given with an information cascades whose users are part of a previously trained NETRATE model. Assume that we have observed the dynamics of a given NETRATE model until some right-censoring time T via a cascade \mathcal{H}_T and that we observed $N(T) = |\mathcal{I}(T)| < N$ infections, where we define as $N(t)$ the number of node infections until time t . Assuming also that no network node is immune to a contagion (Following the SI model, the node is susceptible to an infection but not yet infected.), so that $N(+\infty) = N$, it may be of interest to investigate how many additional nodes $N_x(t_x|T) \triangleq N(T + t_x) - N(T)$ will get infected in the future time interval $[T, t_x)$ for some $t_x > T$. Due to the stochastic nature of NETRATE's dynamics, $N_x(t_x|T)$ will be an RV with a certain PMF with support on $\{0, \dots, \bar{N}_x\}$, where $\bar{N}_x \triangleq N - N(T)$. Identifying the latter PMF will allow us to compute quantities like the expected value or the mode of $N_x(t_x|T)$ as

well as various types of confidence intervals for it.

For that purpose consider the events $\mathcal{E}_i \triangleq \{t_i \leq T + t_x\}$ for $i \notin \mathcal{I}(T)$, where, as usual, t_i denotes node i 's infection time. These correspond to infection events at some time in $(T, T + t_x]$ of nodes i that have survived until time T . By assumption A.2, these events are mutually independent given \mathcal{H}_T . This means that the RVs $N_i(T + t_x) \triangleq \mathbb{I}\{t_i \leq T + t_x\}$ are also mutually independent given \mathcal{H}_T . Their expected value is given as

$$\pi_i(t_x|T, \mathcal{H}_T) \triangleq \mathbb{E}\{N_i(T + t_x)|\mathcal{H}_T\} = \mathbb{P}\{t_i \leq T + t_x|\mathcal{H}_T\} \quad (7.1)$$

and, hence,

$$N_i(T + t_x)|\mathcal{H}_T \stackrel{\text{iid}}{\sim} \text{Bernoulli}(\pi_i(t_x|T, \mathcal{H}_T)) \quad (7.2)$$

Notice that

$$N_x(t_x|T) = \sum_{i \notin \mathcal{I}(T)} N_i(T + t_x) \quad (7.3)$$

and, therefore, $N_x(t_x|T)|\mathcal{H}_T$ is distributed as a Poisson-Binomial RV with parameters $\{\pi_i(t_x|T, \mathcal{H}_T)\}_{i \notin \mathcal{I}(T)}$, whose PMF is given as

$$p_{N_x(t_x|T)}(\cdot|\mathcal{H}_T) = \bigstar_{i \notin \mathcal{I}(T)} p_{N_i(T+t_x)}(\cdot|\mathcal{H}_T) \quad (7.4)$$

where $*$ denotes discrete convolution and $p_{N_i(T+t_x)}(\cdot|\mathcal{H}_T)$ is the PMF of $N_i(T + t_x)|\mathcal{H}_T$, which, being Bernoulli-distributed, is given as

$$p_{N_i(T+t_x)}(n|\mathcal{H}_T) = \pi_i(t_x|T, \mathcal{H}_T)\mathbb{I}\{n = 1\} + (1 - \pi_i(t_x|T, \mathcal{H}_T))\mathbb{I}\{n = 0\} \quad (7.5)$$

7.4 provides the PMF we were seeking and which can be computed by using Fast Fourier Transform-based methods at a computational cost of $\mathcal{O}(\bar{N}_x \log \bar{N}_x)$. The conditional expectation of $N_x(t_x|T)$ can be simply computed as

$$(7.3) \Rightarrow \mathbb{E}\{N_x(t_x|T)|\mathcal{H}_T\} = \sum_{i \notin \mathcal{I}(T)} \pi_i(t_x|T, \mathcal{H}_T) \quad (7.6)$$

What remains to be determined is the precise expression of $\pi_i(t_x|T, \mathcal{H}_T)$, which we do next. For all $i \notin \mathcal{I}(T)$, it holds

$$\begin{aligned} \pi_i(t_x|T, \mathcal{H}_T) &= 1 - \mathbb{P}\{t_i > T + t_x | \mathcal{H}_T\} \stackrel{(S1)}{=} 1 - \frac{S_i(T + t_x | \mathcal{H}_T)}{S_i(T | \mathcal{H}_T)} \stackrel{(3.48)}{=} \\ &= 1 - \exp\{-[\Lambda_i(T + t_x | \mathcal{H}_T) - \Lambda_i(T | \mathcal{H}_T)]\} \stackrel{(3.47)}{=} \\ &= 1 - \exp\left\{-\sum_{j \in \mathcal{I}(T)} a_{j,i} [\psi(T + t_x - t_j) - \psi(T - t_j)]\right\} \end{aligned} \quad (7.7)$$

where in step (S1) we used the fact that one can deduce the event $t_i > T$ from \mathcal{H}_T , that is, we used the fact that $\mathbb{P}\{t_i > T + t_x | \mathcal{H}_T\} = \mathbb{P}\{t_i > T + t_x | t_i > T, \mathcal{H}_T\}$. This results into an expected number of excess infected nodes given by

$$\begin{aligned} (7.6) &\stackrel{(7.7)}{\Rightarrow} \mathbb{E}\{N_x(t_x|T)|\mathcal{H}_T\} = \\ &= \bar{N}_x - \sum_{i \notin \mathcal{I}(T)} \exp\left\{-\sum_{j \in \mathcal{I}(T)} a_{j,i} [\psi(T + t_x - t_j) - \psi(T - t_j)]\right\} \end{aligned} \quad (7.8)$$

As expected, it is evident from the previous expression that, (i) $\mathbb{E}\{N_x(0|T)|\mathcal{H}_T\} = 0$ and (ii) $\mathbb{E}\{N_x(+\infty|T)|\mathcal{H}_T\} = \bar{N}_x$, since $\psi(+\infty) = +\infty$.

(7.4) can be used for predicting the dynamic (as a function of an increasing t_x value) virality of a contagion across the network in a couple of ways: (i) by

retrieving the mode of $N_x(t_x|T)|\mathcal{H}_T$'s distribution to obtain the most likely number of excess infections, where the PMF must be computed via (7.4), or (ii) using (7.8) to measure popularity via its expected value over time. Obviously, the latter option is more computationally attractive, as being less computationally expensive. However the PMF is obviously more rigorous and we can get information like the confidence interval for the excess number of infections in a given time period.

Since equations 7.8 and 7.7 clearly depends on learned parameters a_{ji} , any deviations from the ground truth would result in a bad prediction model. So, based on the reasons discussed in section 5.1, the prediction model would fail to give accurate predictions. However, in the case when the learning algorithm approaches the ground truth, we can investigate the effectiveness of these predictions. We trained the model using cascades simulated using an exponential kernel (10 nodes). In order for the training to approach the ground truth, we extended the right censoring time for the simulation to a large value in order to ensure that all the nodes are infected in the cascade. We then tested it with some test cascades. For a given cascade, we iterated through multiple cases: predict future size given first infection, second infection and so on. 7.1 shows the PMFs generated in each case. For notational purposes, we consider n to be the number of infection considered for generating the PMF. The quality of the prediction improves as more infection are observed in the test cascade. The PMFs were generated as per the convolution presented in equation 7.4.

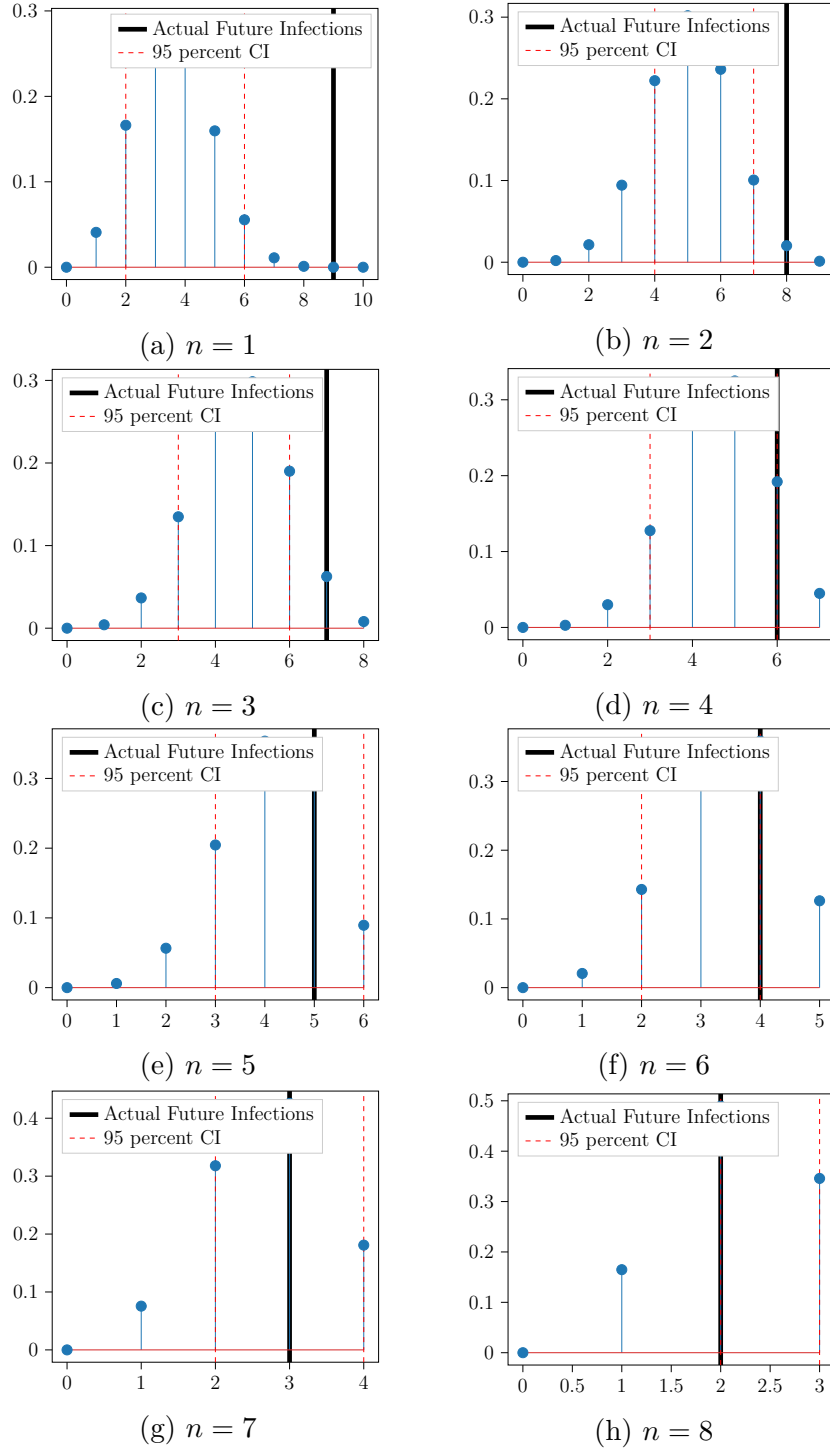


Figure 7.1: PMFs generated for future infections as we consider n infections have already happened in the cascade.

Chapter 8

Real World Data: Software Vulnerabilities

In order to showcase how the NETRATE model applies to real world situations, we have obtained data concerning software vulnerabilities. More specifically, we have obtained, for each CVE, an information cascade. For the task of graph inference, the data source provides the context for what the inferred graph represents and how we interpret the resulting graph. Below we will discuss the dataset and the implications of various platforms that the CVE cascade traverses through.

8.1 Software Vulnerability Data

We have a list of 13,069 CVEs and conversation about these CVEs across 3 social networks: Reddit, Twitter and Github. Before we characterize how this builds to a cascade, we must look at the role of these social networks in the life cycle of the CVE. A CVEs is issued and stored in a national database whenever a

vulnerability has been identified. This database is maintained by National Security FFRDC (Federally Funded Research and Development center) and is operated by the MITRE corporation. It was created in September 1999.

When a CVE is created, there is an entry first created in the national vulnerabilities database and can be accessed through a public API. However, at this point the cascade hasn't been initiated yet. CVEs are rarely fixed without a discussion among peers about the issues surrounding the CVE. This is where the our information cascades are created. People create posts in one of the three social networks: Github, Reddit and Twitter posting the CVE number and their opinions or updates regarding the CVE. For example, they could complain about it or a developer can post an update about a particular CVE number. This forms a sequence of events about a particular CVE number. It is to be noted that there may be other platforms that play a role in these fixes, or even they may take place completely in the privacy of an organization. However, we are only interested in those cascades that exist in these platforms.

CVE cascades with Twitter

Twitter is a social network wherein the user can send out posts or tweets with a character limit of 200. There can also be replies to these tweets, referred to as retweets. w.r.t to the CVEa, this means that there can't be a detailed discussion about a CVE. Figure 8.1 shows an example of how CVE related events take place in twitter. So this platform is mostly used for sending updates or complaints regarding the status of a CVE. Thus, it is likely for this platform to be an initiator of the cascade.



Figure 8.1: Example of an event from twitter that contributes to the CVE cascade

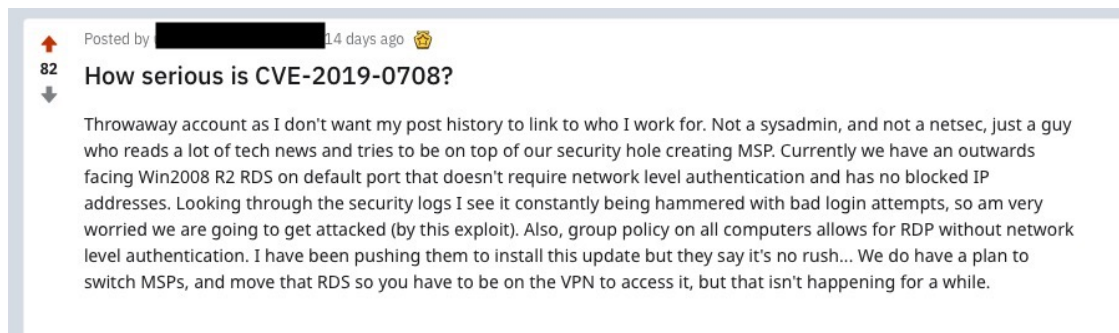



Figure 8.2: Example of an event from Reddit that contributes to the CVE cascade

CVE Cascades with Reddit

Reddit is another social network where the posts are public and can be highly detailed. They can have replies as well. This enables the users to have a detailed discussion about the dynamics of the CVE. Figure 8.2 shows an example of how CVE related events take place in Reddit. They may even go into discussions about how to go about fixing the CVE. It is to be noted that Reddit threads are usually moderated, thus ensuring the quality of the CVE mentions.

CVE-2018-1002105: proxy request handling in kube-apiserver can leave vulnerable TCP connections #71411

 Closed

opened this issue on Nov 26, 2018 · 49 comments

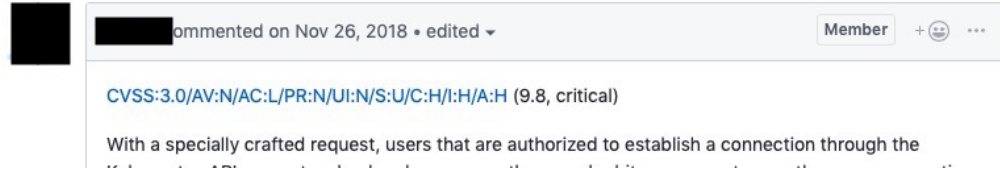


Figure 8.3: Example of an event from Github that contributes to the CVE cascade

CVE Cascades with Github

Github is a hybrid social network in the sense that it allows for people to post updates and comment on these updates, but most of the time, it is linked to a particular piece of code. Obviously this implies that the base software on which the vulnerability is based on is open-source. Figure 8.3 shows an example of how CVE related events take place in Github. If there were any fixes taking place for the CVE, it would be reflected on Github. Therefore, a CVE cascade reaching Github is indicative of an active effort to fix the issue.

Based on the dataset, we will be looking at two types of CVEs: Exploited and Non-exploited. The NETRATE model is trained with cascades of these two types separately to compare and contrast their behaviors. Exploited CVEs refer to the list of CVEs that have been exploited by malicious actors or hackers. Clearly, these CVEs have a higher risk factor and the hypothesis is that links associated with diffusion of exploited CVE are weaker and hence contribute to them getting exploited. One of the base assumptions of the NETRATE model is that a node can only get infected once in a cascade. This translates to CVE domain as follows: A user can only participate once in a cascade for a particular CVE. The data

obtained also reflects this assumption. In order to obtain an acceptable goodness of fit, we concluded in chapter 6 that we would need a large number of samples relative to the number of nodes. This is clearly not the case for the CVE data because for exploited cascades we have 940 cascades with 4348 nodes, while in the non-exploited case, we have 12,129 cascades with 6737 nodes. Although 12,129 cascades is large, this is nowhere near the number of cascades needed to train the model and obtain accurate results. So, this naturally acts as an argument for regularization.

Chapter 9

Results: NETRATE for software vulnerabilities

This chapter compiles and interprets the results obtained after using the NETRATE model on both exploited and non-exploited CVEs. For both the cases, we ran the EM algorithm 2 using exponential, Rayleigh and the power-law kernel. For the following analyses, we used the memory kernel that produced the highest likelihood on the validation set.

9.1 CVE: Exploited

9.1 shows the likelihoods produced in the validation dataset. The power-law distribution produced the best likelihoods and therefore was used for the following analyses. We found best incoming and outgoing average transmission rate per platform. Table 9.2 describes these values. Github had the highest average outgoing transmission rate. This makes sense since all updates and code changes involved

with the CVE take place in github and are propagated outwards. We observed that Reddit had a higher average incoming transmission rate indicating that the Reddit platform is more active when on the receiving end. This is a special case for exploited CVEs resulting due to the fringe community setting that Reddit provides.

Figure 9.1 shows the average transmission rates obtained for links that jump across various platforms. Links between Github and Reddit seems to be stronger than other platform links. Exploited CVEs propagate best from Github to Reddit. We posit that this is due to the discussion about exploiting CVEs in community or subreddits, which is often severely under-moderated. Figure 9.2 indicates that among the intra platform links, Github has the highest transmission rates on average. This makes sense because of the discussion involved with commits and pull-requests associated with software vulnerabilities. The higher intra-platform rate implies the large magnitude of discussion in Github and it fails to leak to the other platforms, hence it is relatively isolated.

Table 9.1: Validation Likelihood of kernels: **Exploited** CVEs

Kernel	Validation log-Likelihood
Exponential	-5027.45
Rayleigh	-34380805.59
Power-law	-418.97

Table 9.2: Average incoming and outgoing transmission rate: **Exploited** CVEs

Platform	Average incoming $a_{ji}s$	Average outgoing $a_{ji}s$
Reddit	0.00068	0.00058
Twitter	0.00043	0.00045
Github	0.00056	0.00072

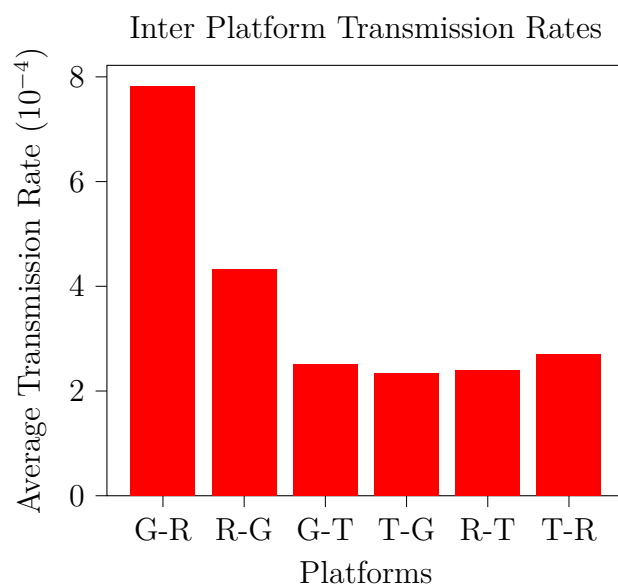


Figure 9.1: Inter Platform average transmission rates for **Exploited** CVEs. The average rate was obtained by looking only at those edges that go from one platform to another. (G - Github, R - Reddit, T-Twitter)

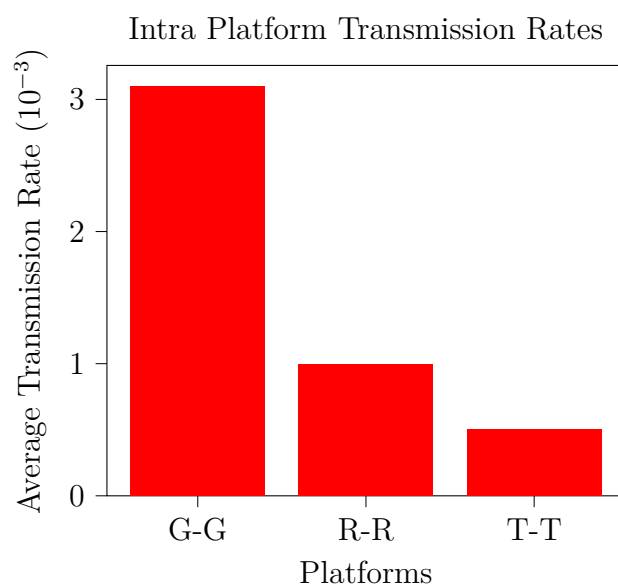


Figure 9.2: Intra Platform average transmission rates for **Exploited** CVEs. The average rates were obtained by observing edges that stay within a platform. (G - Github, R - Reddit, T-Twitter)

9.2 CVE: Non-Exploited

Table 9.3 shows the likelihoods produced in the validation dataset. The power-law distribution has once again produced the best likelihoods. Table 9.4 describes the average incoming and outgoing transmission rates for the three platforms. These values indicate that Github acts like a central hub for Non-exploit CVEs since it has highest values for both incoming and outgoing transmission rates.

Figure 9.3 shows the average transmission rates obtained for links that jump across various platforms. Interestingly, the rates are not as diverse as for the exploit CVEs. Although Github as a source as a higher average transmission rate, all the other platforms seems to equally involved with these CVEs. This indicates that CVEs that have a balanced presence in the platforms tend to not be exploited. Figure 9.4 differs from the exploited case where the intra-platform transmission rates are roughly balanced.

An interesting observation is that Exploited CVEs have a lower average transmission rate compared to the Non-Exploit case (based on the tables 9.2 and 9.4). This could possibly be due to less involvement among multiple developers that deal with exploited CVEs. This again emphasizes why the CVEs were exploited in the first place.

Table 9.3: Validation Likelihood of kernels: **Non-exploit**

Kernel	Validation log-Likelihood
Exponential	-87818.49
Rayleigh	-854721702.47
Power-law	-2908.73

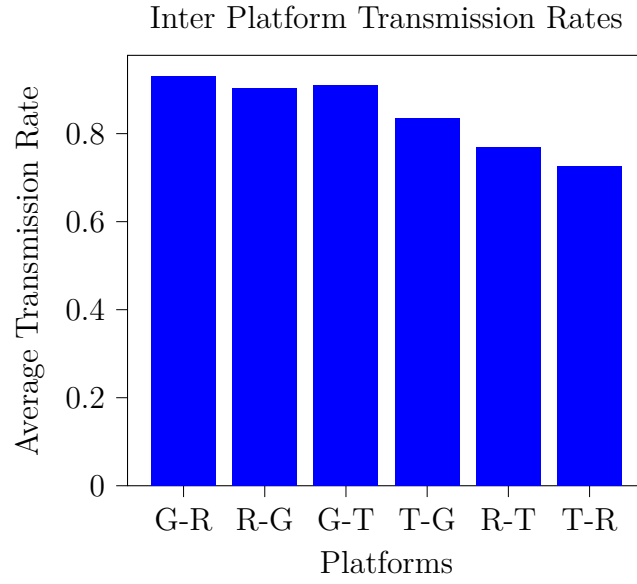


Figure 9.3: Inter Platform average transmission rates for **Non-Exploit** CVEs. The average rate was obtained by looking only at those edges that go from one platform to another. (G - Github, R - Reddit, T-Twitter)

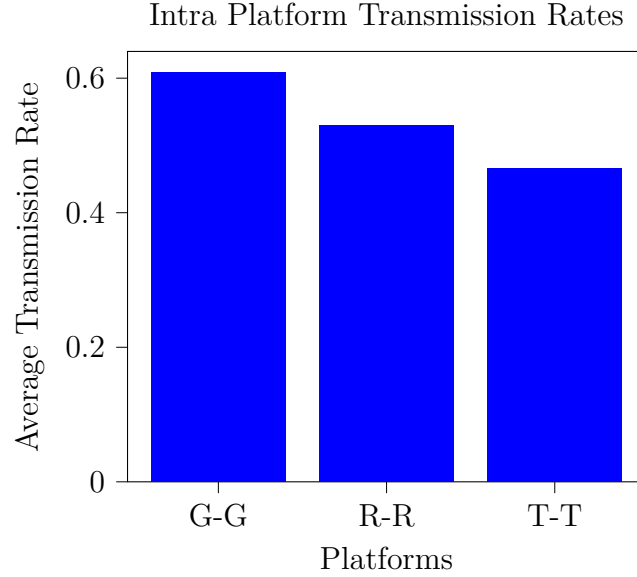


Figure 9.4: Intra Platform average transmission rates for **Non-Exploit** CVEs. The average rates were obtained by observing edges that stay within a platform. (G - Github, R - Reddit, T-Twitter)

Table 9.4: Average incoming and outgoing transmission rate: **Non-Exploit**

Platform	Average incoming a_{ji} s	Average outgoing a_{ji} s
Reddit	0.69	0.73
Twitter	0.58	0.54
Github	0.79	0.84

Chapter 10

Conclusions and Future Work

The field of information propagation has had various approaches in the recent years. We have approached the spread of information as a survival based point process. We have provided a fresh perspective to the NETRATE model for information diffusion by infusing information about the parentage of events. We have also derived the EM algorithm for the same and have provided an efficient implementation the scales well to distributed systems. We have used a time transformation for the infection times and have proved that the transformed times come from an exponential-1 distribution. This has enabled us to perform a goodness of fit test to validate NETRATE model based on ground truth transmission rates. The simulation algorithm in chapter 6 formed the basis for testing and debugging the learning algorithm based on ground truth. Finally, we formulated an expression for the PMF that describes number of future infections in a cascade. We have shown using a test cascade that for cases where the learned parameters are close to the ground truth parameters, the PMF provides a good fit.

We ran the NETRATE model to investigate the nature of information propa-

gation of two types of CVE cascades: Exploited and Non-Exploited. We observed some interesting results by their respective transmission rate matrices. In both cases, the power-law distribution produced the highest likelihood on the validation set. Interestingly, the power-law model is also used to model infectious disease spread [38]. Leskovec et al. [39] found that blog posts citing each other exhibit a power-law characteristic for its popularity. This also agrees with Bauckhage et al. [40]. They explain that for modelling memes and other pieces of information, Weibull, Gompertz and Frechet provide the best models. The power-law distribution is a special case of the Weibull distribution.

Exploited CVEs are outliers in the sense that they seem to involve the Reddit platform more than the Non-Exploit CVEs. In both cases, the link between users in Github to users in Reddit are strong, indicating the volume of information flow in that direction. A final and interesting observation was that Non-exploited CVEs have, on average, higher transmission rates compared to exploit. This, we conjecture, may be the cause of why CVEs were exploited in the first place. They flew under the radar, giving opportunity for malicious actors to take advantage.

There are several issues encountered during this thesis that helps lay the foundation for future work. The choice for the memory kernel was decided based on the likelihood on the validation set, however, estimating the exact shape of the ψ function or IMK requires the use of some non-parametric techniques. For instance Aalen et al. have used the Nelson-Aalen estimator [4] as a non parametric estimators for survival processes. Certain considerations have to be made when finding such an estimator for large number of processes. Secondly, we noticed that the sizes of CVE information cascades were significantly smaller than the size of the network. The context of the information cascade determines how it spreads [41].

This meant that although we get some reasonable estimates to the edge weights, it did not approach the ground truth. This was verified based on our experiments by simulation with smaller networks. Essentially, each CVE cascade has its own behavior and can perhaps be better modelled by incorporating features of the CVE into the model. This would help explain why some users are immune to some CVEs even though there is a link between users. This will in turn help improve the prediction of excess infection within a cascade.

Bibliography

- [1] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. *CoRR*, abs/1105.0697, 2011.
- [2] Mehrdad Farajtabar, Yichen Wang, Manuel Gomez-Rodriguez, Shuang Li, Hongyuan Zha, and Le Song. COEVOLVE: A joint point process model for information diffusion and network co-evolution. *CoRR*, abs/1507.02293, 2015.
- [3] Ali Zarezade, Ali Khodadadi, Mehrdad Farajtabar, Hamid R Rabiee, and Hongyuan Zha. Correlated Cascades: Compete or Cooperate. In Satinder P Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 238–244. {AAAI} Press, 2017.
- [4] Odd O. Aalen, Årnluf Borgan, and Hakon Gjessing. *Survival and Event History Analysis: A Process Point of View*. Springer, 01 2008.
- [5] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. *CoRR*, abs/1305.3616, 2013.
- [6] Y. Ogata. On lewis’ simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, January 1981.

- [7] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1513–1522, New York, NY, USA, 2015. ACM.
- [8] Hua-Wei Shen, Dashun Wang, Chaoming Song, and Albert-László Barabási. Modeling and predicting popularity dynamics via reinforced poisson processes, 2014.
- [9] Timothy C. Brown and M. Gopalan Nair. A simple proof of the multivariate random time change theorem for point processes. *Journal of Applied Probability*, 25(1):210–214, 1988.
- [10] Shuang-Hong Yang and Hongyuan Zha. Mixture of mutually exciting processes for viral diffusion. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages II–1–II–9. JMLR.org, 2013.
- [11] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.
- [12] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. *CoRR*, abs/1006.0234, 2010.
- [13] Seth A. Myers and Jure Leskovec. On the convexity of latent social network inference. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2*, NIPS'10, pages 1741–1749, USA, 2010. Curran Associates Inc.

- [14] Jalal Etesami, Negar Kiyavash, Kun Zhang, and Kushagra Singhal. Learning network of multivariate hawkes processes: A time series approach. *CoRR*, abs/1603.04319, 2016.
- [15] Yingxiang Yang, Jalal Etesami, Niao He, and Negar Kiyavash. Nonparametric hawkes processes: Online estimation and generalization bounds. *NIPS*, 01 2018.
- [16] E. Bacry, K. Dayri, and J. F. Muzy. Non-parametric kernel estimation for symmetric hawkes processes. application to high frequency financial data. *The European Physical Journal B*, 85(5):157, May 2012.
- [17] Hongteng Xu, Mehrdad Farajtabar, and Hongyuan Zha. Learning granger causality for hawkes processes. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1717–1726, New York, New York, USA, 20–22 Jun 2016. PMLR.
- [18] Mehrdad Farajtabar, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Boutros Khalil, Shuang Li, Le Song, and Hongyuan Zha. Fake news mitigation via point process based intervention. *CoRR*, abs/1703.07823, 2017.
- [19] Jooyeon Kim, Behzad Tabibian, Alice Oh, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Leveraging the crowd to detect and reduce the spread of fake news and misinformation. *CoRR*, abs/1711.09918, 2017.
- [20] Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. Deepcas: an end-to-end predictor of information cascades. *CoRR*, abs/1611.05373, 2016.
- [21] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. *CoRR*, abs/1607.00653, 2016.

- [22] Qi Cao, Huawei Shen, Keting Cen, Wentao Ouyang, and Xueqi Cheng. Deep-hawkes: Bridging the gap between prediction and understanding of information cascades. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 1149–1158, New York, NY, USA, 2017. ACM.
- [23] Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *CoRR*, abs/1612.09328, 2016.
- [24] Shuai Gao, Jun Ma, and Zhumin Chen. Modeling and predicting retweeting dynamics on microblogging platforms. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 107–116, New York, NY, USA, 2015. ACM.
- [25] Peng Bao, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. Modeling and predicting popularity dynamics of microblogs using self-excited hawkes processes. *CoRR*, abs/1503.02754, 2015.
- [26] Bidisha Samanta, Abir De, Abhijnan Chakraborty, and Niloy Ganguly. Lmpp: A large margin point process combining reinforcement and competition for modeling hashtag popularity. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2679–2685, 2017.
- [27] Qitian Wu, Chaoqi Yang, Hengrui Zhang, Xiaofeng Gao, Paul Weng, and Guihai Chen. Adversarial training model unifying feature driven and point process perspectives for event popularity prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 517–526, New York, NY, USA, 2018. ACM.

- [28] Y. Ogata. Seismicity analysis through point-process modeling: A review. *pure and applied geophysics*, 155(2):471–507, Aug 1999.
- [29] A Helmstetter and Didier Sornette. Subcritical and supercritical regimes in epidemic models of earthquake aftershocks. *Journal of Geophysical Research*, 107, 10 2002.
- [30] Marian-Andrei Rizoiu, Young Lee, Swapnil Mishra, and Lexing Xie. Frontiers of multimedia research. *Frontiers of Multimedia Research*, pages 191–218, 2018.
- [31] Ke Zhou, Hongyuan Zha, and Le Song. Learning triggering kernels for multi-dimensional hawkes processes. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1301–1309, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.
- [32] Angelos Dassios and Hongbiao Zhao. A dynamic contagion process. *Advances in Applied Probability*, 43, 09 2011.
- [33] Arthur Dempster, Natalie Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38, 01 1977.
- [34] Regina C. Elandt-Johnson and Norman L. Johnson. *Survival models and data analysis / Regina C. Elandt-Johnson, Norman L. Johnson*. John Wiley and Sons New York, 1980.
- [35] Hadi Daneshmand, Manuel Gomez-Rodriguez, Le Song, and Bernhard Schölkopf. Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm. *CoRR*, abs/1405.2936, 2014.

- [36] C. F. Jeff Wu. On the convergence properties of the em algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.
- [37] Mingfeng Lin, Henry C. Lucas, and Galit Shmueli. Too big to fail: Large samples and the p-value problem. *Information Systems Research*, 24:906–917, 12 2013.
- [38] Sebastian Meyer and Leonhard Held. Power-law models for infectious disease spread. *The Annals of Applied Statistics*, 8:1612–1639, 10 2014.
- [39] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie Glance, and Matthew Hurst. Cascading behavior in large blog graphs patterns and a model. *SIAM International Conference on Data Mining (SDM), 2007*, 05 2007.
- [40] Christian Bauckhage, Kristian Kersting, and Fabian Hadiji. Mathematical models of fads explain the temporal dynamics of internet memes. In *ICWSM*, 2013.
- [41] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th International Conference on World Wide Web, WWW '11*, pages 695–704, New York, NY, USA, 2011. ACM.

Appendix A

Proofs & Derivations

A.1 Proof of Proposition 1

Proof. We have the definition for the cumulative hazard function:

$$\Lambda(t) = \int_0^t \lambda(\tau) d\tau$$

The survival function is represented as

$$S(t) \triangleq \exp\{-\Lambda(t)\} \tag{A.1}$$

The probability density function of death (or infection) also referred to as the lifetime distribution (assuming that the entity has survived until that time) is given in terms of the conditional intensity function $\lambda(t)$ as:

$$f(t)dt \triangleq \lambda(t)S(t)dt \tag{A.2}$$

The the pdf of the lifetime can therefore be represented as

$$f(t) = \lambda(t)S(t) \tag{A.3}$$

using A.1, we get

$$\tag{A.4}$$

$$f(t) = \lambda(t) \exp\{-\Lambda(t)\} \tag{A.5}$$

The above proof can also be applied when working vice-versa. i.e. assuming that the PDF of the lifetime is $f(t) = \lambda(t) \exp\{-\Lambda(t)\}$, we can show that the survival function $S(t) \triangleq \exp\{-\Lambda(t)\}$.

$$S(t) \triangleq \int_t^{+\infty} f(\tau) d\tau \tag{A.6}$$

using the fact that $f(t) = \lambda(t) \exp\{-\Lambda(t)\}$, we get

$$S(t) = \int_t^{\infty} \lambda(\tau) \exp\{-\Lambda(\tau)\} d\tau \tag{A.7}$$

substituting $u = \exp\{-\Lambda(t)\}, du = -\exp\{-\Lambda(\tau)\} \lambda(\tau) d\tau$,

$$\begin{aligned} S(t) &= \int_0^1 -1 du = \\ S(t) &= u = \exp\{-\Lambda(t)\} \end{aligned} \tag{A.8}$$

□

A.2 Proof of Proposition 2

Proof. We have a collection of J mutually independent, non-negative RVs $\{T_j\}_{j=1}^J$. T is the lifetime of the entire survival process as a whole such that $T \triangleq \bigwedge_{j=1}^J T_j$, where \bigwedge is the 'mimumum' set operator.

We have a survival process of J components. The individual lifetime of a single process is represented using the I-SP for the lifetime PDF

$$f_i(t) \triangleq \lambda_i(t) \exp\{-\Lambda_i(t)\}, i \in \{1, 2, \dots, J\} \quad (\text{A.9})$$

We define the total hazard rate and the total cumulative hazard rate as follows:

$$\lambda(t) \triangleq \sum_{j=1}^J \lambda_j(t) \quad (\text{A.10})$$

$$\Lambda(t) \triangleq \sum_{j=1}^J \Lambda_j(t) \quad (\text{A.11})$$

The survival function of the lifetime distribution T is

$$\begin{aligned} S(t) &= \mathbb{P}\{T > t\} = \\ &\stackrel{T \triangleq \bigwedge_{j=1}^J T_j}{\implies} \mathbb{P}\{T_1 > t, T_2 > t, \dots\} = \\ &\stackrel{\text{Mutual Independence}}{\implies} \prod_{j=1}^J \mathbb{P}\{T_j > t\} = \\ &= \prod_{j=1}^J \exp\{-\Lambda_j(t)\} = \\ &= \exp\left\{-\sum_{j=1}^J \Lambda_j(t)\right\} = \end{aligned}$$

$$\stackrel{\{A.11\}}{\implies} = \exp\{-\Lambda(t)\} \quad (\text{A.12})$$

The PDF of the combined lifetime can be represented using the I-SP representation. The terms have a common survival function because anytime an event occurs, the σ -algebra changes (\mathcal{H}_t changes), as a result, the individual conditional intensities $\lambda_j(t)$ changes.

$$f(t) = \sum_{j=1}^J \lambda_j(t) \exp\{\Lambda(t)\} \quad (\text{A.13})$$

$$f(t) = \sum_{j=1}^J \lambda_j(t) \exp\left\{\sum_{j=1}^J \Lambda_j(t)\right\} \quad (\text{A.14})$$

□

A.3 Proof of Proposition 3

Proof. For the same collection of J mutually independent, non-negative RV where $\{T_j\}_{j=1}^J$, where T is the lifetime of the entire survival process such that $T \triangleq \bigwedge_{j=1}^J T_j$. The PDF of each T_j is $f_j(t)$. The RV is defined as $pa(T) \triangleq \arg \min_{j=1, \dots, J} T_j$. We define the probability that a new event occurring at T is of index j is:

$$\begin{aligned} \mathbb{P}\{pa(T) = j \mid T = t\} &= \mathbb{P}\{pa(T) = j \mid t < T \leq t + dt\}, T_j = T \\ \xrightarrow{\text{Applying conditional probability formula}} &= \frac{\mathbb{P}\{pa(T) = j \cap t < T \leq t + dt\}}{\underbrace{\mathbb{P}\{t < T \leq t + dt\}}_{f(t)dt}} \end{aligned} \quad (\text{A.15})$$

where $f(t)$ is the lifetime distribution

$$\begin{aligned}
\text{A.15} \quad & \xrightarrow{\text{Independence of lifetimes of each process}} = \frac{\mathbb{P}\{t < T_j \leq t + dt\} \prod_{k \neq j} \mathbb{P}\{T_k > t + dt\}}{f(t)dt} \\
& = \frac{f_j(t) \prod_{k \neq j} S_k(t)}{f(t)} \\
& \xrightarrow{\text{from Proposition A.2}} = \frac{\lambda_j(t) \exp\left\{-\sum_{k=1}^J \Lambda_k(t)\right\}}{\sum_{k=1}^J \lambda_k(t) \exp\left\{-\sum_{k=1}^J \Lambda_k(t)\right\}} \\
& = \frac{\lambda_j(t)}{\sum_{k=1}^J \lambda_k(t)} \tag{A.16}
\end{aligned}$$

The other two statements from the proposition naturally follow using the independence assumption for the multi-variate survival process.i.e.

$$f(t, pa(T) = j) = f_j(t) \prod_{k \neq j} S_k(t) \tag{A.17}$$

$$\begin{aligned}
\mathbb{P}\{pa(T) = j\} &= \int_{\tau=0}^{\infty} f(t, pa(T) = j) dt = \\
&= \int_{\tau=0}^{\infty} f_j(t) \prod_{k \neq j} S_k(t) dt \tag{A.18}
\end{aligned}$$

□

A.4 Proof of Proposition 4

Proof. Since $\phi(t) \geq 0$ by definition, $f(t) \geq 0$. Next,

$$\begin{aligned}
\int_{\mathbb{R}_+} f(t) dt &= \int_{\mathbb{R}_+} \phi(t) \exp\{-\psi(t)\} dt = - \int_{\mathbb{R}_+} d \exp\{-\psi(t)\} = \\
&= \exp\{-\psi(t)\} \Big|_{+\infty}^0 = 1 - 0 = 1
\end{aligned}$$

where we used the fact that, by its very definition, for any IMK ψ , we have that $\psi(0) = 0$ and $\psi(+\infty) = +\infty$.

□