Statistical Analysis in Cyberspace

Data veracity, completeness, and clustering

Roeling, M.P.

DOI
10.4233/uuid:f495fd3f-d131-40c9-a51e-e4a8bcc12c84

Publication date
2021

Document Version
Final published version

Citation (APA)
Roeling, M. P. (2021). *Statistical Analysis in Cyberspace: Data veracity, completeness, and clustering*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:f495fd3f-d131-40c9-a51e-e4a8bcc12c84

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

MARK PATRICK ROELING

STATISTICAL ANALYSIS IN CYBERSPACE

DATA VERACITY, COMPLETENESS AND CLUSTERING

Statistical Analysis in Cyberspace

Data veracity, completeness, and clustering

STATISTICAL ANALYSIS IN CYBERSPACE

DATA VERACITY, COMPLETENESS, AND CLUSTERING

Dissertation

for the purpose of obtaining the degree of doctor
at the Delft University of Technology
by the authority of the Rector Magnificus Prof.dr.ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates
to be defended publicly on
Monday 7 June 2021 at 15:00 o'clock

by

Mark Patrick ROELING

Master of Science in Behavior Genetics, Vrije Universiteit Amsterdam, the Netherlands
Master of Science in Genetic Epidemiology, Erasmus University Rotterdam, the Netherlands
born in Rhenen, the Netherlands

This dissertation has been approved by the promotors.

Composition of the Doctoral Committee:

Rector Magnificus	chairperson
Prof.dr.ir. J. van den Berg	Leiden University, and Delft University of Technology, promotor
Prof.dr.ir. R.L. Lagendijk	Delft University of Technology, promotor
Dr.ir. S.E. Verwer	Delft University of Technology, promotor

Independent members:

Prof.dr. G.K. Nicholls	University of Oxford
Prof.dr. S. van Buuren	TNO Research / Utrecht University
Prof.dr.ir. G. Jongbloed	Delft University of Technology
Prof.dr. M. Conti	Delft University of Technology
Dr. P. Rubin-Delanchy	University of Bristol

Prof.dr. G.K. Nicholls, Prof.dr.ir. J. van den Berg, and Dr.ir. S.E. Verwer contributed substantially to this thesis.

This thesis is partially supported by the Engineering and Physical Sciences Research Council (EPSRC).

Keywords: Cybersecurity, Unsupervised Learning, Imputation, Data Veracity

ISBN 978-94-6423-299-8

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

CONTENTS

List of Figures

LIST OF SUPPLEMENTARY FIGURES

LIST OF TABLES

1

INTRODUCTION

This is a thesis about statistical learning in the context of cyber security. Statistical learning is a large area of research with a long history, providing analytical techniques, methods, concerns, and tools for the estimation (learning) of parameters to either draw inference from, or predict patterns in, data [1]. Cyber security refers to the body of technologies, processes, and practices designed to protect networks, devices, programs, and data from attack, damage, or unauthorized access [2]. There is also an import behavioural element to cyber security as it deals with the security of cyber activities, or: the security of cyber behaviour [3]. Activities in cyberspace can be disentangled in a multidimensional and multilayer organisation with a (1) technical (IT) layer, (2) socio-technical layer, and (3) governance layer, each with their own security requirements and implications [4].

With the increasing digitization of society, there is an inherent need to understand risks, weaknesses, and other security properties related to digital-infrastructures. For example, online shopping websites usually have customer databases, but the implementation of how customer data are collected or stored is complex and often has security weaknesses, which can be exploited so that private data can be accessed by unauthorized persons (hackers) and can be used for extortion [5] or identity theft [6]. Another example concerns the (covert) installation of malicious software on a computer, so that the behaviour of a (user on a) computer can be monitored or influenced [7]. Many factors influence the likelihood for these *infections*, such as the use of outdated software, mistakes in the implementation of security measures, users not adequately protecting their personal computers or personal information. Inadequate protection of personal information can increase the risk for social engineering attacks (i.e., where somebody uses personalized information to adopt a false identity and convince the user to allow / support an operation) [8]. Critical infrastructures that are strongly automated are also known to be vulnerable to cyber incidents, such as energy and water supply [9], transport [10], health care [11], and SCADA systems [12]. There have been many efforts to prevent occurrence or escalation of these kinds of unwanted events, such as government-supported awareness and information campaigns [13, 14]. However, raising awareness has (up till now) not been sufficient, as the damage due to phishing in online-banking

increased from €1.05 million in 2017 to €3.81 million in 2018 [15], the average damage of a malware attack on a company in 2019 is estimated at $2.6 million [16], data breaches exposed 4.1 billion records in the first half of 2019 [17], ransomware damage costs have risen to $11.5 billion in 2019 and one business is expected to fall victim to a ransomware attack every 14 seconds in 2020 [18]. These examples underline the potential impact of cyberincidents and have set off a clear interest to prevent occurrence or escalation of cyber-events as well as attribute these to an entity. Important tools in understanding, preventing and managing risks in cyber space are risk management models. One risk management model that is applicable to the management of cyber risks is the *bowtie* model [19, 4]. Stemming from the 1960s, bowtie became the a popular model for analyzing and managing risks in the early 1990s. Central to bowtie thinking is the operationalization of a risk as the product of the likelihood of an incident and its impact. This allows to model incidents and impact with fault trees, attack trees, event trees, and/or Bayesian networks. The bowtie way of thinking allows sufficient flexibility to cope with the variation in cyber incidences: an intruder may use a large number of pathways to obtain their objectives [20]. To understand these pathways, many different research paradigms have been explored and applied to cyber security, ranging from studies focusing on psychological factors to understand decision making, to abstract computational research. Popular examples of psychological factors in cyber security are the privacy paradox; where most people report favourable attitudes towards privacy but nevertheless share their data freely [21] and the online disinhibition effect; which is the removal or reduction of the social and psychological restraints that individuals experience in everyday face to face interaction [22]. Other examples of research in cyber security are vulnerability / penetration testing of (SCADA) systems [12], mathematically proving the quality of (the implementation of) protocols [23, 24], attribution and interpretation of cyber incidents in the context of international relations [25, 26] and robustness of trusted platform modules [27, 28]. Cyber security therefore has become a broad and interdisciplinary field where insights from many fields (computer science, mathematics, social sciences, and humanities) are used to understand certain phenomena, technical aspects, and the interaction between the online environment and human behaviour.

1.1. RESEARCH MOTIVATION

One of the directions that has been powerful and efficient in describing attack scenario's and incident detection, in the cyber context, is the use of statistical analyses [29, 30, 31, 32, 33]. For 50 years, there has been a continuous interaction between mathematics and computer science, flowing to (and allowing) developments in statistics and machine learning. There has been remarkable progress in the past decade in the application of anomaly detection methods to detect fraud [34, 35], malware [36], intrusion detection [37], and infected devices in networks known as bots [38, 39, 40]. These early successes kindled the enthusiasm to develop statistical methods tailored to cyber security problems and are currently shaping a new field of scientific inquiry: *Statistical Cyber-Security* [41].

Previous studies experimenting with the application of existing algorithms to (large) datasets obtained in various aforementioned cyber-scenario's, manifested the need for methods that can cope with the volume(s) and types of data typically available in the

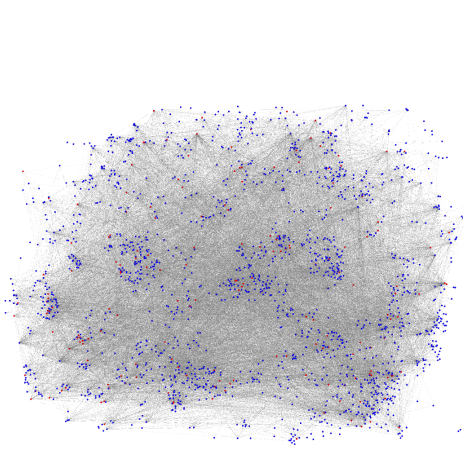
cyber setting. Many statistical algorithms and methods have been designed and constructed for applications in epidemiology [42, 43, 44], behavioural sciences [45], marketing [46], operational research, etc., but are not necessarily appropriate for immediate application across contexts. Especially in parametric statistical models (where we assume that the sample we study can be modeled by one or more probability distribution(s) with a fixed set of parameters), the statistical model may be specifically tuned to a specific parameterization (e.g. capturing certain types of distributions). In other settings the loss function to optimize (a set of) parameters may work in one setting but not in another (examples below). Consequently, the misfit between the suitability of a given method and the context in which it is applied can be a complexity, which is a potential shortcoming in many cyber security studies, catalyzing inaccurate inferences.

One large area of the field where this is particularly apparent is the (quantitative) analyses of networked data, which ranges from community detection in social networks focused on cyber-bullying (e.g. [47]) to analyzing captures of packets send from one computer to another to detect infected computers (e.g. [48]). As illustrated by Figure 1.1, the structure of a social network (described in Chapter 3) is markedly different from the structure of a computer network (described in Chapter 5). The social network data is more densely connected: there are more often highly connected subgroups (e.g. friends or members of a club). In contrast, a computer network typically consist of nodes that do not form highly connected clusters because devices / servers do not need to communicate to other devices to interact with the internet. Opposite to social networks; nodes in the network that are similar in terms of their role in the network (behaviour / connectivity) are often relatively unlikely to connect, a phenomenon known as disassortativity [49]. There is also a difference in the amount of times a connection is endorsed over time (the number of send messages in the social network is lower compared to computer connections). Popular network clustering methods often aim to detect highly connected clusters with more links / edges / connections within groups than between groups (e.g. modularity based clustering; [50] or spectral clustering), but these can perform suboptimal in networks where nodes are loosely connected to each other but highly connected to a specific group of hubs [51] such as in computer networks [49]. Finally, many studies that focus on classification of computers in networks require a labelled dataset, but the cyber security setting often lacks the presence of a label (e.g. in botnets), limiting operational usefulness. These examples of problems specific to a cyber security setting commend the development of suitable statistical models to understand phenomena captured by data, encountered in a setting where data are retrieved or analyzed online, machine generated, or linked by specific modes of (inter)dependence.

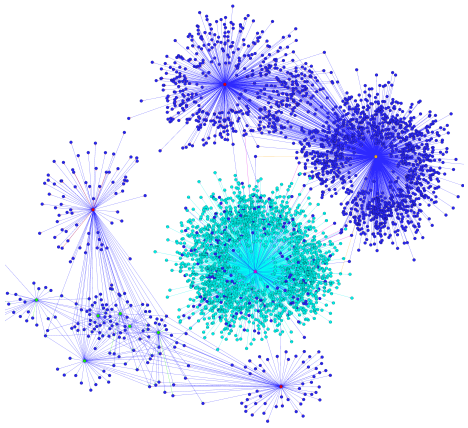
1.2. WHAT THIS THESIS ADDRESSES

This thesis addresses several methodological complexities relevant to the analyses of data in cyber-space, by illustrating the application of existing methods in a new context, and by furthering existing methods. We selected three major topics because of the growing relevance of analyzing networked data and the seriousness of the security threat posed by for example malware and botnets. The three topics are: 1. Handling missing data in social networks; 2. Network clustering: the detection of computers infected with malware by using a method that works even in the absence of labels indicating infection

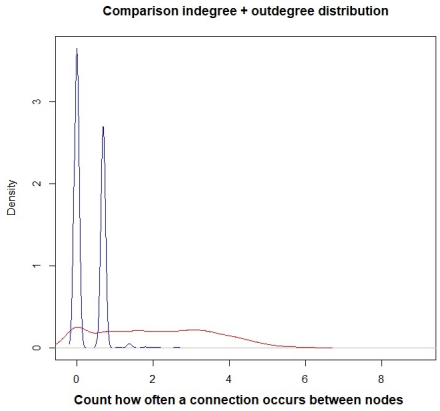
1



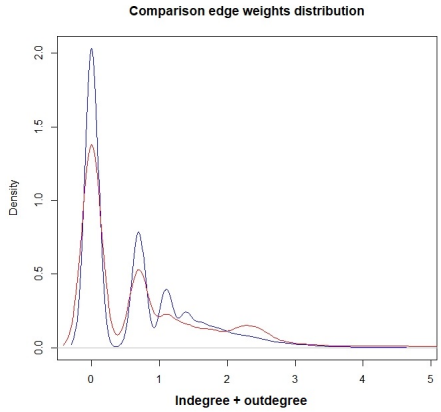
(a) Illustration of a Social Network



(b) Illustration of a Computer Network



(c) Distribution of edge weights between a social (blue) and computer (red) network.



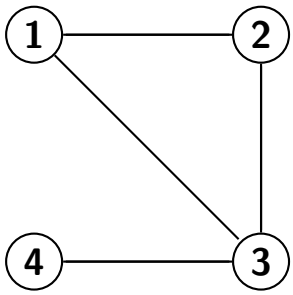
(d) Distribution of indegree+outdegree between a social (blue) and computer (red) network.

Figure 1.1: Network plots of a social network (data used in Chapter 3) and a computer network (data used in Chapter 5). We can see that the social network (a) is more densely connected compared to the computer network and suggests assortativity (b), but in the computer network the central nodes have more endpoints, and suggests disassortativity. This difference is visualised in (c) by plotting the distribution of total (receiving + sending) connections per node, where blue = social network, and red = computer network. However, counting how often a connection occurs between nodes in the network (edge frequency) as depicted in (d) results in identical distributions between the two different networks.

status (unsupervised). We used a combination of a node clustering method with a connection clustering method, and; 3. Data veracity (conformity of observed data to facts) in online application systems. These topics are introduced below.

1.2.1. INCOMPLETE DATA IN NETWORKS

The internet has moved part of our (social) interactions to an online environment, which deals with identity and contacts differently to a purely physical environment. For example, are we certain that the LinkedIn or Facebook profile we are connecting with is a true person? There is some evidence that fake Facebook profiles often have missing data in attributes [52]. With 982 Facebook profiles (781 real and 201 fake), there were missing values attributes of 61 (7.8%) real and 25 (12.4%) fake profiles. To keep the online environment secure, we need statistical models that can cope with these missing data, which is why this thesis focused on imputation of missing data in attributes from linked observations. Missing and incomplete data is a common problem across research fields and can result in more data loss (if data from incomplete cases are removed) or biased parameter estimates. In networks, the consequences of missing data can be more severe compared to independent cases [53, 54, 55, 56], for example when complete-case analyses excludes a person with missing data and all the edges to his peers (and information) are dropped. In extreme examples where missing data correlates with network structure, entire segments of the network can drop-out. Networks (or graphs) are usually represented by edges and vertices. The weights on the edges can be denoted in a adjacency matrix which values may indicate a (number of) connections (e.g. communication or friendship) between nodes, and no value may indicate the absence of a connection. This representation has allowed the development of many clustering methods to either cluster network nodes or ties. Given that network clustering methods rely on the network matrix, previous research has focused on estimating unobserved or missing ties [57]. We are interested in missingness in X, the matrix with node attributes where covariate (the features) values of the nodes are stored. Figure 1.2 presents and example where there is a missing value (indicated by ?) in the gender variable. In our setting the network structure complements a collection of attribute data, such as gender, income,



$$C = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$X = \begin{array}{c|cc} \text{ID} & \text{age} & \text{gender} \\ 1 & 13 & 1 \\ 2 & 20 & 0 \\ 3 & 16 & ? \\ 4 & 19 & 1 \end{array}$$

Figure 1.2: Undirected graph with a symmetric adjacency matrix C, indicating all connections are bidirectional. The diagonal of C is always zero implying the absence of self-loops. X is the covariate matrix with variables age and gender. Since all four observations are part of a social network, the variables in X are node attributes. In X, one observation ($ID = 3$) has with missing gender.

age, etc. There are many ways to replace the ? with a value, such as taking the mean (or in categorical data the mode) of the complete cases. In the observed data we observe one male ($gender = 0$) and two females ($gender = 1$) so the mode = 1 and we could decide to replace the missing value with a 1, indicating a female. However, mean imputation can be a problem in statistical analyses, because we add observations, but do not add variation. We can easily understand why mean imputation is a problem if we consider the formula for the standard deviation in a sample;

$$s^2 = \frac{\sum(x - \bar{x})^2}{n - 1} \tag{1.1}$$

where s^2 indicates the variance, n is the sample size, x is the sample value, and $\bar{x}$ is the sample mean. With mean imputation we increase n but for every observation that is given the mean as a value the product $\sum(x - \bar{x})^2$ is zero, so s^2 deceases. There are also other options with their own advantages and disadvantages [58], but the general consensus is that statistical modelling (via imputation methods) should provide an accurate and unbiased way to estimate missing values [59].

The estimation of missing attribute data when observations are linked is more challenging (compared to independent observations) and not well understood. A large body of research on imputation presents methods to estimate missing covariates in independent observations [60], but none of these methods support networked data. We propose Bayesian Autocorrelation Regression Models (ARMs) as a sensible method for the construction of a posterior distribution from which values can be inferred to replace the missing data. ARMs are based on regular regression models and thereby provide a straightforward and formally tractable framework for the analyses of attributes [61, 62]. We acquired data via dr. Tore Opsahl who collected data from a Facebook like messaging app used by graduate students, so that the network represents who send or received messages during a 6-month period. Every node also has attributes such as gender, age or year of study, and we introduced missing data in *gender* to test our imputation performance.

1.2.2. NETWORK CLUSTERING

Network data implies links or connections between nodes, captured in an adjacency or connection matrix: C (see Figure 1.2). When C only indicates the presence or absence of a connection $a \rightarrow b$, then $C_{ab} = 1$ if there is a connection between nodes a and b and 0 otherwise. Sometimes C contains weights (e.g. amount of money transferred between clients), indicating that C in essence can take any value from any distribution. Frequently, every node $i \in \mathcal{N}$ also has some attribute data, captured by X, holding p covariates for the n nodes or observations.

One popular area where data always follow a network structure is the analyses of computer networks that are infected with malware, so that infected internet-connected computers become *zombies* or bots. Large networks of bots are known as *botnets*, and are controlled by one or more botnet controllers [63] performing malicious tasks. The activity on such a network can be captured with *packet capture* software (e.g. Wireshark), and typically results in a long stream of data with variables such as source, destination, port, packet length, and time. This spatio (which node in the network) - temporal (ac-

tivity over time) structure has been proven a challenge for statistical analyses. In short this entails; 1. loss of variation with studies collapsing the information captured over time for every node into a simplified variable, or by removing streams of data that only occur once; 2. many current classification methods applied to (streaming) data in cyber security rely on preprocessing of the data, and 3. the requirement of a labelled dataset, whereas the cyber security setting often lacks the presence of a label, such as in captures of network activity.

The above-mentioned limitations directed the efforts in this thesis to present a robust, valid, and reliable procedure to detect botnet infected machines. Chapters 4 and 5 apply Stochastic BlockModels (SBMs) to botnet data with the aim of identifying infected clusters without the need for a labelled dataset. At the start of this project (late 2015), the overwhelming majority of studies focusing on the detection of computers that were / are infected with malicious software, made use of supervised clustering. In supervised clustering, a labelled dataset is used to allow the computer to learn patterns of behaviour (in terms of variance, covariance, or interactions) between (statistical projections of) features or covariates. In many operational settings, these labels are not readily available when we create are statistical model, which in botnet detection is a rationale for using statistical / machine learning models. Consequently, we rely on another group of learning methods called unsupervised learning, where patterns in the data are learned (by the computer) without the availability of examples or labels. SBMs are attractive because they seek highly connected blocks in network connections while allowing the inclusion of covariates, in a statistically tractable way (variational Expectation Maximization). Hence, there is no need to choose between analyzing the node structure or node attributes as both are considered in the same model. To prevent collapse of covariates and loss of temporal resolution, we experiment with the use of features derived from Dynamic Time Warping techniques (distance measure of time sequences) and Ngrams (distance measure for strings). This also reduces the required number of covariates since all variation is (assumed to be) captured by these distance features. The method does not require a priori (manual) host or sequence filtering, and we experiment with different packet thresholds to show which data-specific settings are optimal. We first test the principles on synthetic data, followed by captures from the wild. Finally, the main finding is replicated in another (larger) capture of network activity, with different botnets.

1.2.3. DATA VERACITY

STATE ESTIMATION

Concerns regarding cyber security often apply to systems that process data or information. Processing these data often involves the analyses of states of a system. For example, supplying sufficient electricity to a city requires strict monitoring of supply and demand, since power consumption changes constantly. Underproduction of energy results in power-loss, but overproduction of energy wastes resources. In another context, we might want to govern how much water should run through a dam in order to maintain proper flood (water) levels beyond the dam for residents in downstream area's. For these kinds of applications, state estimators have been developed. One of the most famous state estimators is the Kalman Filter (KF; [64]), which was developed by NASA to predict and guide the position of the 1977 Mariner Jupiter-Saturn mission. Kalman Filters (KF)

are recursive state estimation algorithms capable of combining and weighting different variables to estimate the state of a system [64]. Recursive reflects the property that every iteration incorporates information from previous observations and predictions [65]. Many versions of the KF have been proposed and in the area of state estimation or particle filtering [66]. They work by comparing a state matrix against a change matrix of a dynamical model to predict the real state of a system (correcting for measurement and prediction error).

Since KFs are so widely implemented in all kinds of cyber-physical systems, there as been growing interest to understand how vulnerable state-estimators are to attack models where the functioning of KFs is subjected to various scenario's where it's convergence is compromised. Recent studies have shown that KFs are vulnerable to data-injection attacks [67, 68, 69], where attackers inject data by supplying extra change matrices (as fake measurements) to subvert the state estimate and control the behaviour of a system. In chapter 6, we test the vulnerability of Kalman Filters to data injection in the context of ADS-B (Automatic Dependent Surveillance-Broadcast) systems, used in airplanes for navigation. This context is relevant as ADS-B is the communications protocol are part of next generation air transportation systems and feed many modern air traffic control systems, making it essential in the protection of two billion passengers per year. The inherent lack of security measures in the ADS-B protocol has long been a topic in both the aviation circles and in the academic community [70]. Usually ADS-B exists as an on-board system that takes input from satellites to determine the position of a moving (flying) object, and broadcasts that position to other (proximate) aircraft (and vice versa) to create positional awareness. To understand the vulnerability in the ADS-B context, we present a sensitivity study where different adversarial scenario's are tested to investigate the influence of false data injection in the Kalman Filter. Data-injection in KFs was successfully applied in power system estimation [68] and we replicate that study by injecting location data in ADS-B data to increase the prediction error in the location of an aircraft, showing the potency of data-injection in changing the estimated location of an aircraft.

ONLINE DATA COLLECTION

In the field of epidemiology, medical studies usually make use of tests or questionnaires administered to patients under (direct) supervision of an examiner. The direct interaction and monitoring between the test-subject and the person who oversees or conducts the analyses is time intensive but ensures correct interpretation of reported values and test outcomes. A more efficient way of collecting data is through online questionnaires, with many obvious benefits; the volume of potential respondents who can be reached with relatively little resources, clients can have 24/7 access to services, and data do not require to be digitized after collection. But this efficiency-increase comes at a price: less control over data collection, which can lead to problems in data veracity (the conformity to facts). These problems can be caused by misinterpretation or wrong interpretations of the question, for example when the measurement instrument is badly constructed [71]. Another well-known phenomenon with online data collection is a substantial non-response or drop-out rate compared to interviews or *live* administration of a questionnaire, resulting in missing data.

One setting where online data collection is an integrated part of the operational procedure is in a financial context, when clients from a bank apply for a consumer credit.

Consumer credits are a product common in financial organisations to provide a loan to earn long-term interest. Client's who require funds can apply for a loan via the website of a bank [72] where they are required to fill in a set of questions assessing their personal (e.g. marital status and number of children) and financial situation (e.g. income and years in job). Consumer credits belong to a group of products vulnerable to fraud [73], which is a wrongful or criminal deception intended to result in financial or personal gain (Oxford Dictionary). There is significant interest in the early detection of suspicious behaviour indicative to fraud, and statistical analyses are known to have merit for fraud detection [74, 34, 75]. These methods typically work by analyzing features (or variables) that can be combined to create a fraud-indicator value.

The key problem with current statistical methods to detect fraud is that feature-data are taken as valid, while potential fraudsters have an obvious interest in manipulating the input data [76], by providing wrong or false answers in the application process to prevent detection. This phenomenon has been studied in the context of identity theft and/or credit-card application fraud [34], where fraudsters adopt the identity of another (often familiar) person (e.g. family member) by using personal information (social Security number, address, health insurance information or job history). This information is then (ab)used to apply for credit by taking out loans or opening new accounts in the victim's name. One identity theft prevalence study ranked *new loans* as third most frequent, behind *new telephone services* and *new credit card accounts* [77]. There are many settings where intensive checks (two-way verification) or personal security verification questions by phone are performed to ensure valid authentication, but in online application systems this remains challenging. Clearly, applicants can also report completely false answers, without the use of personal details of a victim. In another context, using personal details or a specifically tuned context of a victim to earn another person's or organisation's trust is a common phenomenon in spear phishing [78].

In chapter 7 we present a collaborative project with ING bank which investigated whether it was possible to identify surprising values on consumer credit applications, and which surprising values were indicative for fraud status. These measures of surprise not necessarily imply dishonest answers but present some measure of inconsistency between values from different variables. From ING, we used a large dataset from non-fraud (controls) to train generalized linear models for predicting the feature values in a *non-fraud* setting. The parameters from that model were used to calculate predicted feature values in a test set including both fraud and non-fraud clients so that residuals could be inferred (observed - expected), as a measure of surprise. Case-control comparison revealed which feature-residuals were indicative for fraud, and we tested whether adding that information to the canonical fraud detection model was sensible.

1.3. THESIS STRUCTURE

This thesis is structured as follows, we present an introduction in statistics relevant to this thesis in chapter 2, and continue the thesis with two parts; network analysis and data veracity. Part I, themed network analysis, consists of chapters 3, 4, and 5 where we respectively present a model for the imputation of attributes in networked data, and an unsupervised approach to cluster spatio-temporal data with mixture models. The main contributions of part I are:

- an extension of autocorrelation regression models to impute missing data in attributes from linked observations with and without flow from imputed observations,
- an unsupervised model to detect computers infected with P2P malware, and
- an unsupervised model to analyse spatio-temporal data, in the context of botnets, where nodes as well as connections are clustered.

In part II we present two chapters (6, 7) focused on data veracity where we inject false data in state estimators (Kalman Filters), and show how to detect anomalies in the context of credit fraud applications. The main contributions of part II are:

- the effects of (false) data injection on the convergence of Kalman Filter estimates in the context of ADS-B systems used in aircraft navigation, and
- a proof of concept to quantify unexpected responses in "online" applications of consumer credits, an the merit of this quantification to the detection of fraud.

Conclusions and a discussion of the results presented in this thesis are put forth in chapter 8.

2

AMUSE-BOUCHE OF STATISTICAL TECHNIQUES APPLIED

This chapter presents a short introduction and explanation of the statistical methods used in this thesis, provided for the researcher who is less-familiar with the concepts of imputation, mixture models and network clustering.

We will use the following dataset to illustrate our examples:

Person	Gender	Height	Weight
Susan	f	1.73	58
Luke	m	1.82	83
Jasmin	f	1.68	55
Mary	f	1.78	72
Hank	m	1.90	95
James	m	1.85	86
Laura	f	1.76	64
Kevin	m	1.77	78
Mark	m	1.83	86
Yara	f	1.71	61

Table 2.1: An example dataset, as used in this chapter

2.1. DISTRIBUTIONS

This thesis makes use of the exponential family of probability distributions, mostly the normal- and chi-squared distributions. Given a measure θ, we define an exponential family of probability distributions as those distributions whose density (relative to θ) have the following general form:

$$p(x|\theta) = h(x)\exp(\eta(\theta)\ T(x) - A(\theta)) \tag{2.1}$$

where θ is the canonical parameter, and $T(x)$, $h(x)$, $\eta(\theta)$, and $A(\theta)$ are known functions ([79], p.312). A large part of the imputation and mixture models literature is written around the normal distribution where the variable-to-impute or the distribution of the mixture component is a Gaussian. We therefore discuss the normal distribution and likelihood below. In the normal distribution, the data form a random sample from the normal distribution; we treat the data $y_1,\ldots,y_n$ as the observed values of $Y_1,\ldots,Y_n$ where the Y_j are independently taken from the normal density

$$f(y|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}}\exp(-\frac{(y-\mu)^2}{2\sigma^2}), \quad -\infty < y < \infty \tag{2.2}$$

with mean $\mu \in \mathbf{R}$ and variance $\sigma^2 \in \mathbf{R}^+$ [80]. We can standardize this distribution via $z = (y-\mu)/\sigma$ so that the corresponding random variable $Z = (Y - \mu/\sigma)$ has density

$$\phi(z) = (2\pi)^{-1/2}\exp(-\frac{1}{2}z^2), \quad -\infty < y < \infty \tag{2.3}$$

which is the density of the standard normal random variable Z. The density is symmetric about $z = 0$, the expectation $E(Z) = 0$ and $\text{var}(Z) = 1$. The mean and variance of $Y = \mu + \sigma Z$ are respectively μ and σ^2. The notation $Y \sim N(\mu,\sigma^2)$ refers to variable Y has the normal distribution with mean μ and variance σ^2 [79]. The standard normal has distribution function

$$\Phi(z) = (2\pi)^{-1/2}\int_{-\infty}^{z}\exp(-\frac{1}{2}u^2)du \tag{2.4}$$

and has the useful property that $z_{0.025} = -1.96$ and $z_{0.05} = -1.65$ and is symmetric around $z = 0$ (see Figure 2.1).

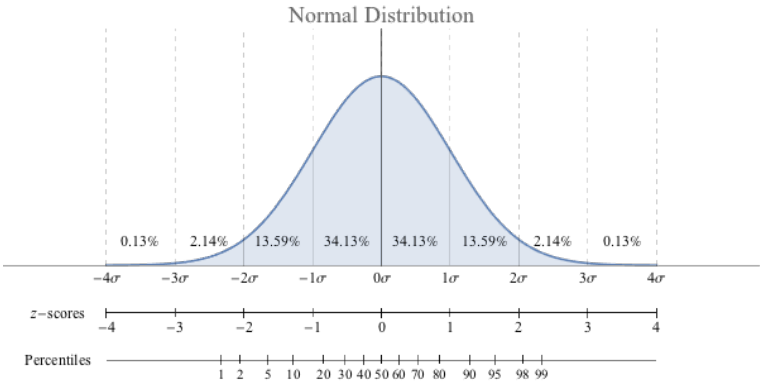


Figure 2.1: Normal distribution

When $Z_1, \ldots, Z_k$ are independent standard normal random variables, $W = Z_1^2 + \cdots + Z_k^2$ has the chi-squared distribution on k degrees of freedom; $W \sim \chi_v^2$ ([79], p. 63) defined (in terms of the probability density function) as

$$f(x,k) = \begin{cases} \frac{x^{\frac{k}{2}-1} e^{-\frac{x}{2}}}{2^{\frac{k}{2}} \Gamma(\frac{k}{2})}, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\Gamma\frac{k}{2}$ denotes the gamma function, which has closed-form values for integer k.

2.2. IMPUTATION OF MISSING DATA

Sometimes data are not completely observed because they contain one or more missing value(s) and we need to decide what to do with these partly-observed cases; we need a missing data strategy.

2.2.1. MISSING DATA MECHANISMS

To understand which missing data strategy is appropriate, we need to decide to which of the typical missing data mechanisms (formalized by Rubin in 1976 [81]) we attribute the missing values; missing data completely at random (MCAR), missing data not at random (MNAR), and missing data at random (MAR). Missing completely at random occurs if the probability of being missing is the same for all cases (e.g., all subjects are equally likely to be missing). This implies there are no systematic differences between subjects with observed and unobserved values meaning that the observed values can be treated as a random sample of the population. For example, if we use a laptop to collect data but some of the data are missing due to a empty battery [58, 82]. In missing at random, the likelihood of a value to be missing depends on other, observed variables. Hence, any systematic difference between missing and observed values can be attributed to observed data. That is, the relationships observed in the data can be utilized to 'recover' the missing data. For example, a laptop to collect data may be malfunctioning more often in a moist environment [82]. However, if we correct for the environment, it may be possible that within all measurements collected in a moist environment, the probability of a missing value is identical and we have MAR. Missing not at random happens when the probability of being missing varies for reasons that are unknown to the researcher. For example when we use a laptop to collect data but the performance of the processor declines over time presenting us with more missing data but we don't recognize this pattern. This means there is extra information associated with the missing data that cannot be recovered by (the relationships observed in) the collected data [82].

According to Papageorgiou et al. [82], the missing data mechanism should be regarded as an assumption that either supports an analysis or not, rather than as an inherent and identifiable feature of a dataset (see Table 2.2). For example, complete-case analyses can result in a loss of information if we drop data from partly observed cases, while including missing values can complicate data analyses if statistical methods are not able to cope with missing data. Also, missing data can potentially bias analyses due to systemic differences between observed and missing values [83].

With a missing data mechanism in mind, we can select one of the different strategies to deal with partly observed / missing data [84, 82, 59, 85, 58, 86], mainly

- complete-case analyses; neglecting the observations that are not fully observed,
- apply weighting procedures in an attempt to adjust for nonresponse as if it were part of the sample design,
- imputation; filling in one or multiple values for each missing value, and
- model based procedures where a model is defined for the observed data and the inferences are based on the likelihood or posterior distributions under that model.

This thesis focuses on imputation, which refers to a process where missing values are replaced with a (substituted) value [59, 82, 58, 86]. Imputation is a large field of scientific research and many flavours exist, as presented in different reviews [58, 59, 86]. A central problem in imputation is how to choose the substitute value to replace the observed missing value. A plethora of imputation procedures has been developed which can roughly be divided in single- (replace the missing value with one value) and multiple imputation (replace the missing value with multiple values) [82].

2.2.2. SINGLE AND MULTIPLE IMPUTATION

Single imputation ranges from hot-deck imputation; where recorded units in the sample are used as substitute values, and mean imputation; where the mean is calculated in sets of recorded data to become the substitute values, to regression imputation; where the missing values are estimated by predicted values from the regression on (usually) the complete observations [59, 86]. Important limitations of single imputations include the negligence of 1) sampling variability about the actual value and 2) additional uncertainty when more than one model is being used [59]. These factors can influence (co)variance estimates and thus bias parameters [87, 88, 89, 86, 90].

Multiple imputation revolves around the idea of choosing multiple values (m) for every missing value. This means that every missing value is imputed m-times, resulting in m complete datasets, which can be analysed with standard complete-data procedures (that ignore whether an observation was imputed or observed). Rubin mentions three important advantages to multiple imputation over single imputation [59]. First, when substitute values are randomly drawn in an attempt to represent the distribution of the data, multiple imputations increases the efficiency of estimation. Second, when multiple imputations represent repeated random draws under a model for nonresponse, valid inferences are obtained simply by combining complete-data inferences in a straightforward manner. Third, by generating repeated randomly drawn imputations under more than one model, it allows the straightforward study of the sensitivity of inferences to various models for nonresponse (by using the complete-data methods repeatedly).

This all builds to a method that ensures the proper estimation of estimand Q, which is a quantity of scientific interest that can be calculated in the population and does not change its value depending on the data collection design used to measure it [91]. Examples of estimands are the population mean and (co)variance, regression coefficients and factor loadings (not sample means, standard errors, and test statistics). What is proper

was initially outlined by Newman [92] and mainly implies an unbiased and confidence valid estimate [58, 91]. Unbiased means that the average estimates of $\hat{Q}$ over all possible samples Y from the population is equal to Q (the population parameter(s));

$$E(\hat{Q}|Y) = Q \tag{2.5}$$

Confidence validity is achieved when the average of U; the estimate variance-covariance matrix of $\hat{Q}$, over all possible samples Y is equal or larger than the variance of $\hat{Q}$;

$$E(U|Y) \geq V(\hat{Q}|Y) \tag{2.6}$$

where $V(\hat{Q}|Y)$ denotes the variance caused by the sampling process. A procedure is confidence valid if a statistical test, with a stated nominal rejection rate of 5% should reject the null hypothesis in at most 5% of the cases when in fact the null hypothesis is true. Other important underlying conceptual documentation, explaining why certain Bayesian approaches to repeated imputations are improper if misspecified (and solutions to that problem [93]), which sources of variation occur in missing data, how we can be confident that imputation produces valid and robust estimates, and alternatives to multiple imputation, is provided elsewhere (e.g. [91, 94, 85, 58, 86]). Overall, the goal of multiple imputation is to find an estimate $\hat{Q}$ of Q with correct statistical properties and if we have no missing data, the pair $(\hat{Q}, U)$ holds all information we know about Q.

2.2.3. MULTIPLE IMPUTATION MODEL

We will continue by explaining the technical procedure (outlined in [58], p.57 and [59], p. 167) of multiple imputation by presenting some notation. Let X be a $n \times p$ dimensional matrix holding all the available data, including n_0 observations with missing values and n_1 completely observed observations. From X we select Y, which is a $n \times q$ dimensional matrix (q covariates with missing data) for imputation. In Y we have observed values stored in vector Y_{obs} ($n_1 \times q$) and (partly) missing values in Y_{mis} ($n_0 \times q$). From the remaining part of X we subset X_{obs} ($n_1 \times p$) consisting of the predictor covariate data from the complete observations, and X_{mis} ($n_0 \times p$) containing the predictor covariate data from observations for which Y is (partly) missing. Hence, X_{obs} and X_{mis} contain no missing values. For a univariate Y we write lowercase y. Predictor selection for imputation is discussed elsewhere [58].

Our modelling task is to draw m substitute values from a the posterior distribution of Y_{mis} under the chosen model to create m imputations. We specify $Pr(Y_{mis}|X, Y_{obs})$ to indicate that the posterior distribution of Y_{mis} follows from X and Y_{obs}. If our univariate y is normally distributed and we create imputations under the normal linear model, we can apply a Bayesian multiple imputation model as

$$\dot{y} = \dot{\beta}_0 + X_{mis}\dot{\beta}_1 + \dot{\epsilon} \tag{2.7}$$

where $\dot{\epsilon} \sim N(0, \dot{\sigma}^2)$ and $\dot{\beta}_0, \dot{\beta}_1$ and $\dot{\sigma}$ are random draws from their posterior distribution, given the data ([58], p. 57). The dot above a parameter indicates this is a drawn value. For completeness; ϵ is the normally distributed error, $\dot{\beta}$ is the intercept, and $\dot{\beta}_1$ the regression coefficient(s). Under the normal linear model it is typical to use the standard

non-informative priors for each of the parameters [59]. A posteriori, σ^2 is $\hat{\sigma}^2(n_1 - q)$ divided by a $\chi^2_{n_1-q}$ random variable, and β given σ^2 is normal with mean $\hat{\beta}_1$ and variance-covariance matrix $\sigma^2 V$. In all subsequent models the intercept and coefficients are combined in one vector and we add a column of value 1 to X_{obs} for proper estimation. In line with least squares statistics we have

$$V = [X^t_{obs} X_{obs}]^{-1}, \tag{2.8}$$

$$\hat{\beta} = V X^t_{obs} y_{obs}, \tag{2.9}$$

with X^t_{obs} indicating the transpose of X_{obs}. Note that this notation skips the use of a scalar to prevent singular matrices as defined in [58]. We draw $\dot{g} \sim \chi^2_v$ with $v = n_1 - q$ to calculate

$$\dot{\sigma}^2 = \frac{(y_{obs} - X_{obs}\hat{\beta})^t (y_{obs} - X_{obs}\hat{\beta})}{\dot{g}}. \tag{2.10}$$

We continue with drawing q independent $N(0,1)$ variates to create a q-component vector $\dot{z}_1$ and let

$$\dot{\beta} = \hat{\beta} + \dot{\sigma}\dot{z}_1 V^{1/2} \tag{2.11}$$

where $V^{1/2}$ is the square root of V such as the triangular square root obtained by Cholesky factorization. We calculate the n_0 values with

$$\dot{y} = X_{mis}\dot{\beta} + \dot{z}_2\dot{\sigma} \tag{2.12}$$

where $\dot{z}_2$ is a vector of n_0 independent $N(0,1)$ variates drawn independently. A new imputed value for y_{mis} is initiated by drawing a new value of the parameter $\dot{\sigma}^2$, implying that for m imputations the steps to calculate $\dot{\sigma}$, $\dot{\beta}$, and $\dot{y}$ are repeated m times.

2.2.4. MULTIPLE IMPUTATION ON OUR EXAMPLE DATA

We use the example data presented in the start of this chapter, which was fully observed, and introduce missingness into this data by removing the gender of James (person 6). Gender is recoded to 0 and 1, so that $m \rightarrow 0$ and $f \rightarrow 1$. Hence, data $\rightarrow X$ so X is a $n \times p$ dimensional matrix (10 rows and 3 covariates):

$$X = \begin{bmatrix} 1 & 1.73 & 58 \\ 0 & 1.82 & 83 \\ 1 & 1.68 & 55 \\ 1 & 1.78 & 72 \\ 0 & 1.90 & 95 \\ ? & 1.85 & 86 \\ 1 & 1.76 & 64 \\ 0 & 1.77 & 78 \\ 0 & 1.83 & 86 \\ 1 & 1.71 & 61 \end{bmatrix}$$

Our goal is to impute the gender of observation 6 (James) so we extract the variable gender from X to become y. The remaining part of X is used as predictors and becomes X_{obs} for the cases with observed gender ($n_0 = 9$) and X_{mis} for James ($n_1 = 1$). From our newly created gender variable y we extract the observed values to y_{obs}, and $y_{mis} = [?]$. We add a 1 to X_{obs} and X_{mis} to estimate β with matrix multiplication.

$$X_{obs} = \begin{bmatrix} 1 & 1.73 & 58 \\ 1 & 1.82 & 83 \\ 1 & 1.68 & 55 \\ 1 & 1.78 & 72 \\ 1 & 1.90 & 95 \\ 1 & 1.76 & 64 \\ 1 & 1.77 & 78 \\ 1 & 1.83 & 86 \\ 1 & 1.71 & 61 \end{bmatrix} \quad Y_{obs} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad X_{mis} = \begin{bmatrix} 1 & 1.85 & 86 \end{bmatrix}$$

Because gender is a categorical variable with 2 levels (dichotomous) we cannot use a normal linear model for imputation but use a generalized linear model, which is a generalization of ordinary linear regression to allow response variables with error distribution models other than a normal distribution [95]. The categorical imputation applied to our data is different in three steps:

- Our 2-level variable gender (0 and 1) is reason to select an logit-link function, to link our linear predictor to the mean of the (logistic) distribution function. Following previous work [59] we calculate the probability of being a female;

$$\dot{p}_i = \frac{1}{1 + e^{-(X_i \dot{\beta})}} \tag{2.13}$$

  with $i \in mis$.

- We calculate $\dot{\beta}$ as defined in Equation 2.11 but here $V^{1/2}$ is the unscaled covariance matrix (estimated with *glm.fit*)[1].

- We draw n_0 independent uniform (0, 1) random numbers, u_i and if $u_i > \text{logit}^{-1} X_i \theta$) we impute $\dot{y}_i = 0$, otherwise $\dot{y}_i = 1$, where $\dot{\theta}$ refers to a draw from $N(\hat{\theta}, V(\hat{\theta}))$ with $\hat{\theta}$ the maximum likelihood function for θ and $V(\hat{\theta})$ the posterior variance of θ. Parameter θ is a column vector with the same number of components as X_{obs} to capture the logistic element in Equation 2.13 (see [59], p. 170).

For the first iteration of m this means that we obtain as parameter estimates

$$\hat{\beta} = \begin{bmatrix} -1706.75 & 1338.39 & -9.07 \end{bmatrix}$$

$$V^{1/2} = \begin{bmatrix} 8992283.91 & 0 & 0 \\ -6175643.15 & 323519.024 & 0 \\ 27587.58 & -7459.884 & 542.9341 \end{bmatrix}$$

$$\dot{z}_1 = \begin{bmatrix} 0.1449583 & 0.4383221 & 0.1531912 \end{bmatrix}$$

[1] Estimated in R using the mice.impute.logreg() function from MICE [96, 97] with set.seed(1121)

so that

$$\dot{\beta} = \begin{bmatrix} 1301799.4995 & -752066.8512 & 803.3239 \end{bmatrix}$$

and

$$\dot{p}_i \approx 0$$
$$u_i \approx 0.86$$

to observe $u_i > \dot{p}_i$ so that $\dot{y}_i = 0$. If we continue this procedure with $m = 50$, we would obtain $\dot{y}_i = 0$ for 39 times and $\dot{y}_i = 1$ for 11 times, suggesting that James is likely male.

2.2.5. RELEVANCE OF MULTIPLE IMPUTATION TO THIS THESIS

The above explanation is relevant to chapter 3, where we present a model to impute the characteristics of observations when these are related. In the above example, the persons we used for the imputation (Suzan, Luke etc.) were assumed to be independent and we used statistical models that make assumptions about the behaviour of estimands in the independent cases scenario. Now imagine they are somehow related (e.g. friends), then we require statistical models that can accommodate the dependence, and we use autocorrelation regression models with extra parameters to capture the dependence (autocorrelation) between cases. Also, in this example we used a different model with a logistic link function to allow the prediction of a dichotomous y. In chapter 3 we deal with categorical variables in a different way; by considering them as latent continuous variables, constructed by sampling from the left- (if y_{obs}) = 0 and (right $y_{obs} = 1$) truncated normal distributions [98]. Finally, we used a Bayesian statistics paradigm for our imputation, supported with one clear rationale; accuracy of the estimation procedure. This will be explained in the next section.

2.3. BAYESIAN STATISTICS

In Bayesian statistics we use sampling to acquire the posterior distribution, which (in this thesis) is the (weighted) average of the likelihood and the prior. We have specifically chosen the sampling (Bayesian) approach to estimate the parameters, as sampling optimization is known to provide particularly accurate results [99, 100], which is opportune in the imputation setting as inaccuracy in parameter estimation can influence follow-up analyses. Bayesian statistics formulates probability distributions to express uncertainty about unknown quantities. We use $p(\cdot|\cdot)$ to denote a conditional probability density with the arguments determined by the context, and $p(\cdot)$ to denote a marginal distribution. The marginal distribution is the probability distribution of (the values of) the variable(s) without reference to (the values of) the other variables. This contrasts with a conditional distribution, which gives the probabilities contingent upon the values of the other variables.

The Bayes Theorem states

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(\text{data}|\theta)p(\theta)}{p(\text{data})} \tag{2.14}$$

where we use θ to represent the unknown parameters we aim to estimate, and $p(\theta)$ holds the prior information we have about these parameters. We also have $p(y|\theta)$ which is the likelihood, and $p(y)$ is the marginal probability density obtained by summing or integrating out all dependence on θ [101].

Assume we have a data point (x_i) and we want to infer which component (z) it is likely to belong to. This means we want to infer the posterior distribution $p(z|x)$, we use Bayes rule

$$p(z|x) \propto p(z)p(x|z) \tag{2.15}$$

where $\propto$ means proportional to (or: "equal up to a constant"). In other words, we can evaluate $p(z)p(x|z)$ for all values of z, and then normalize the values so that the values sum to 1 [102]. We now explain the different parts of the Bayes rule formula, based on [101].

2.4. LIKELIHOOD

Model fitting can be done by parameter estimation where we vary the parameter values and hold the data constant. To this end, we select or design a likelihood function, depending on assumptions about the data generating process. We call $p(\text{data}|\theta)$ the likelihood because we vary the parameters and keep our data fixed, to calculate the probability density for different values of θ. The likelihood in itself is not a valid probability distribution, which is why we use a normalizing constant to obtain the posterior in Equation 2.14. A schematic overview of probability distributions used for likelihood functions can be found in ([101], p. 145).

The density function of a normal random variable (with mean μ and variance σ^2) was presented in Equation 2.2. The log-likelihood for a random sample $y_1, \ldots, y_n$ (adopted from [79], p. 111) is

$$\mathscr{L}(\mu|\sigma) \equiv -\frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_{j=1}^{n}(y_j-\mu)^2. \tag{2.16}$$

where the summation indicates that we calculate the joint probability density for the n individuals by taking the product of the individual densities and the likelihood is log() transformed to allow easy comparison (it then follows a χ^2 distribution).

If we would like to estimate the probability, θ, that a randomly chosen individual from our example data is a male (0) or female (1) without prior information, we could estimate the probability $Pr(X=0|\theta) = (1-\theta)$ and $Pr(X=1|\theta) = \theta$ which can be expressed as

$$Pr(X=\alpha|\theta) = \theta^{\alpha}(1-\theta)^{1-\alpha} \tag{2.17}$$

where $\alpha \in \{0,1\}$ indicates male of female status. If we would generalize this model to apply for a new sample from the population, to calculate the probability of obtaining Z females in a total sample size of N, we generalise the model so that the likelihood for a sample size of N becomes

$$Pr(Z=\beta|\theta) = \binom{N}{\beta}\theta^{\beta}(1-\theta)^{N-\beta} \tag{2.18}$$

known as the binomial probability distribution. Our example data included 5 males and 5 females, suggesting $\theta = 50\%$. Suppose we collect a new sample of 100 individuals and find that 40 persons are female. We use the above equation to calculate the probability of obtaining 40 or more females by

$$Pr(Z \geq 40|\theta = 0.5) = \sum_{Z=40}^{100} \binom{100}{Z} 0.5^Z (1-0.5)^{100-Z} = 0.98$$

indicating the probability of generating such a sample of 100 individuals with at least 40 females, using this likelihood model, is high.

2.4.1. PRIORS

The Bayesian rule prescribes to multiply the likelihood with the prior. The prior probability distribution ($p(\theta)$) represents our a priori distribution for a parameter's true value. For example, if we study a group of children with mental problems, we use psychological tests (the estimate the likelihood), but we can also ask clinical psychologists to give a prior disease probability distribution based on their expertise. This prior is then used to obtain a probability distribution which is multiplied with the likelihood. Hence, in Bayesian statistics we combine our prior beliefs with data to get to new beliefs

$$p(\theta|\text{data}) \propto \underbrace{p(\text{data}|\theta)}_{\text{likelihood}} \times \underbrace{p(\theta)}_{\text{prior}}.$$

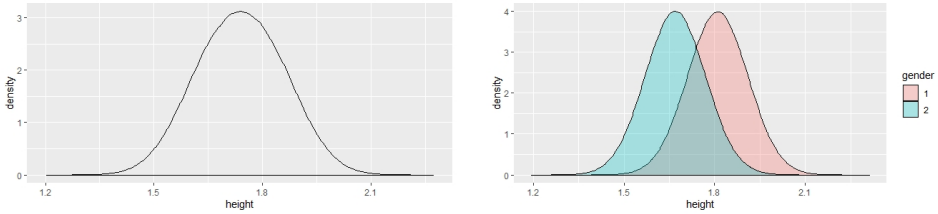
This also implies that when we add more data, the relative importance of the likelihood increases and the influence of the prior reduces [101].

2.4.2. DENOMINATOR

The denominator ($p(\text{data})$) is a normalizing factor, to ensure that the product of the Bayes rule (the posterior distribution) is a valid probability distribution. As can be seen in the Bayes formula, we do not include any effect of θ in the denominator. This implies that, in order to get the value of the denominator, we need to marginalise out all parameter dependence in the numerator. However, this is complicated and often intractable (due to the difficulty of integrating multidimensional probability distributions) [100]. This is why this thesis uses sampling techniques to sample from the posterior.

2.4.3. POSTERIOR

We use the above Bayes rule to combine the prior and likelihood to produce the posterior. The chosen likelihood model determines the influence of the data on the posterior, which is a probability distribution that usually peaks between the peaks of the likelihood and prior [101]. Point estimates are inferred from the posterior, most prominently the mean, median, and mode of the posterior distribution, see chapter 3. Around these points estimates lie credible intervals which describe the uncertainty in the parameter values; a 95% credible region satisfies the condition that 95% of the posterior probability lies in this parameter range.



(a) Height in the population

(b) Height for males and females

Figure 2.2: Example distribution of height in the population (a), and split for males and females (b). Both subfigures are based on the same data. If we neglect gender we observe a normal distribution of height in the population, but actually this normal distribution can be separated into a height-distribution for males and a height-distribution for females. The population distribution is therefore a mixture of the male and female distributions. In this example we are fortunate to have the information who is a male and who is a female. Sometimes these labels are absent from the data, and we use statistical modeling to recover the gender-labels from the population distribution. In other words: we decompose the population mixture distribution to obtain a probability for every person to belong to a gender-class. Hence, the purpose of mixture models is to generate or recover unobserved labels.

2.4.4. RELEVANCE OF BAYES TO THIS THESIS

Chapter 3 uses Bayesian statistics to estimate parameters in a statistical model where we estimate missing data in attributes of network nodes. We use sampling techniques to approximate the (posterior) distribution of different parameters. Specifically, we use Gibbs sampling (a variant of Markov chain Monte Carlo methods). With Monte Carlo we generate lots of (random) samples to make numerical estimations of unknown parameters.

2.5. MIXTURE MODELS

Up to this point, all the variables in the data were observed; we knew gender, height, weight, and friendship status for all individuals. This allowed to fit a range of statistical models where values are observed, or imputation methods when data are partially observed. However, there are many scenario's where we know or expect that there is a variable which we did not observe in the data. In those cases we estimate entire variables from the data by fitting mixture models [103, 104, 105].

Mixture models are statistical models with latent variables, who's values are estimated from the data. Figure 2.2 provides an example of a mixture model (from [106]); we know that height is strongly linked to gender (in our sample most males are longer than females). Because we observed gender, we can make a separate distribution of height for females and one for females. However, if we would not have observed the gender label, we would only have the distribution of height. With mixture models we can treat gender as latent variable and use the distribution of height to recover gender status.

Because height is normally distributed we use a Gaussian mixture model, with density

$$f(x,\vartheta) = \sum_{g=1}^{G} \pi_g \phi_p(x \mid \mu_g, \Sigma_g) \tag{2.19}$$

with $\pi > 0$, such that $\sum_{g=1}^{G} \pi_g = 1$, is the gth mixing proportion,

$$\phi(x \mid \mu_g, \Sigma_g) = \frac{1}{\sqrt{(2\pi)^p |\Sigma_g|}} \exp\{-\frac{1}{2}(x-\mu_g)' \Sigma_g^{-1}(x-\mu_g)\}$$

and $\vartheta = (\pi_1, \ldots, \pi_G, \theta_1, \ldots, \theta_G)$ is the vector of parameters. Hence, $\phi(x \mid \mu_g, \Sigma_g)$ resembles Equation 2.2 and is the density of a random variable X from a multivariate Gaussian distribution with mean μ_g and covariance matrix Σ_g (ϕ is $p-$dimensional)[107]. The component membership of every observation i is denoted by $z_i = (z_i, \ldots, z_{iG})$, so that $z_{ig} = 1$ if observation i belongs to component g and $z_{ig} = 0$ otherwise.

If we consider a clustering scenario with n p-dimensional data vectors $x_1, \ldots, x_n$ are observed and all are unlabelled, the Gaussian model-based clustering likelihood is

$$\mathscr{L}(\vartheta) = \prod_{i=1}^{n} \sum_{g=1}^{G} \pi_g \phi(x_i \mid \mu_g, \Sigma_g). \tag{2.20}$$

Because we have a latent variable, we cannot simply maximize the likelihood directly, but use Expectation Maximization [108]. This is a parameter estimation procedure that consists of two steps, the **E**xpectation step consists of calculating the expectation of the component assignments for each data point (e.g. height of Luke) given the model parameters. The second step is the **M**aximization step consisting of maximizing the expectations calculated in the expectation phase with respect to the model parameters. This step consists of updating the values.

Applying **EM** to the Guassian mixture model is done as follows: We have as complete data likelihood

$$\mathscr{L}_c(\vartheta) = \prod_{i=1}^{n} \prod_{g=1}^{G} [\pi_g \phi(x_i \mid \mu_g, \Sigma_g)]^{z_{ig}} \tag{2.21}$$

and log-likelihood

$$l_c(\vartheta) = \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} [\log \pi_g + \log \phi(x_i \mid \mu_g, \Sigma_g)]. \tag{2.22}$$

In the **E**xpectation, the expected value is updated by replacing z_{ig} by their expected values

$$\hat{z}_{ig} = \frac{\hat{\pi}_g \phi(x_i \mid \hat{\mu}_g, \hat{\Sigma}_g)}{\sum_{h=1}^{G} \hat{\pi}_h f(x_i \mid \hat{u}_h, \hat{\Sigma}_h)}, \tag{2.23}$$

for $i = 1, \ldots, n$ and $g = 1, \ldots, G$. The expected value of the complete-data log-likelihood is

$$Q(\vartheta) = \sum_{g=1}^{G} n_g \log \pi_g - \frac{np}{2} \log 2\pi - \sum_{g=1}^{G} \frac{n_g}{2} \log |\Sigma_g| - \sum_{g=1}^{G} \frac{n_g}{2} \text{tr}\{S_g \Sigma_g^{-1}\},$$

where $n_g = \sum_{i=1}^{n} \hat{z}_{ig}$ and $S_g = \frac{1}{n_g} \sum_{i=1}^{n} \hat{z}_{ig} (x_i - \mu_g)(x_i - \mu_g)'$.

In the **M**aximization step the aim is to maximize $Q(\vartheta)$ with respect to π_g, μ_g, and Σ_g;

$$\hat{\pi}_g = \frac{n_g}{n}, \quad \hat{\mu}_g = \frac{1}{n_g}\sum_{i=1}^{n} \hat{z}_{ig} x_i, \quad \text{and } \hat{\Sigma}_g = \frac{1}{n_g}\sum_{i=1}^{n} \hat{z}_{ig}(x_i - \hat{\mu}_g)(x_i - \hat{\mu}_g)'. \tag{2.24}$$

The model fitting procedure alternates between the E and M steps until convergence, which is when we obtain the EM-solution that has the best fit to the data given the number of mixture components (see an example in Figure 2.3).

One issue with mixture models is that the procedure cannot automatically infer from the data how many mixtures (the number of groups) are optimal given the data. We work around this problem by fitting mixture models for different groups ($g \in \{2,\ldots 4\}$) and we use a fit index (a parameter that expresses how good our model fits the data) to choose which optimized model best describes our data. In example Figure 2.4 we fit mixture models with 3 and 4 mixture components to the data and observe that the best EM-solution captures different clusters of people. These clusters are valid outcomes as they capture persons who are relatively similar in their height and weight. However, the goal is to find the most parsimonious model; the model with least parameters that best describes our data. In practice a fit index is used, such as the Bayesian Information Criterium, to select the model with the least misfit.

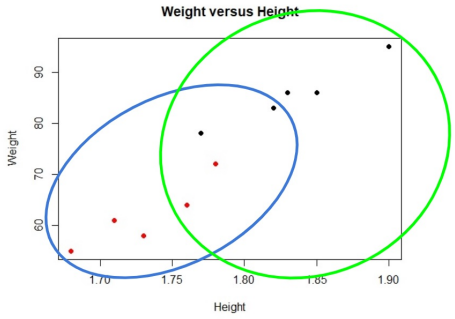
2.6. NETWORK MODELS

Network analyses in the context of this thesis means that we have a covariate that captures some kind of spatial relationship. This can be a number that resembles the distance between one observation and another one, so that all possible combinations of observations can be captured in a network, socio, or adjacency matrix. Below, we expand our above-mentioned example data by adding a network matrix that resembles friendship status. We illustrate the friendships in our sample in Figure 2.5. Using a statistical modelling approach, we capture the friendship relations in a connection-, adjacency-, or socio matrix:

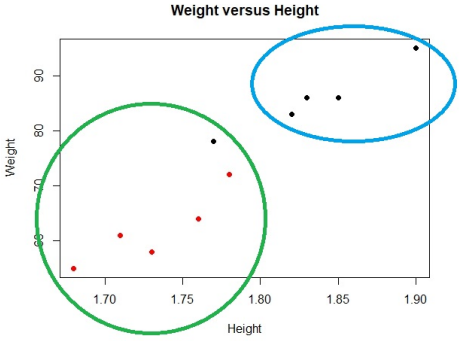
$$C = \begin{bmatrix} & Susan & Luke & \ldots & Yara \\ Susan & - & 1 & \ldots & 0 \\ Luke & 1 & - & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ Yara & 0 & 0 & \ldots & - \end{bmatrix}$$

where a 1 indicates a friendship or a 0 otherwise. Dots indicate that we skip fields in presenting the matrix, to be concise. The upper part of C is equal to the lower part, meaning that we assume a friendship is always mutual. The network consists of nodes (persons) and (connections).

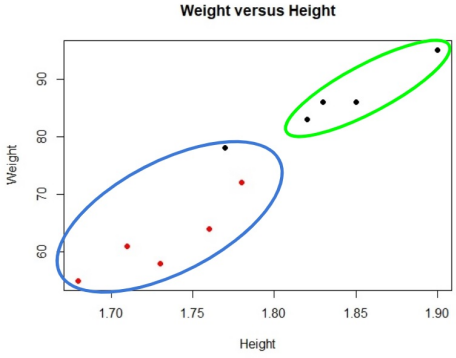
There are many ways to analyze network data; we can count the number of incoming (indegree) and outgoing (outdegree) friendships every person has, and calculate metrics (e.g. density, centrality, or assortativity) that inform about the importance of every individual in the group. For example, in our network we observe that Kevin, Mary, Jasmin, and Yara have 4 connections, Laura has 3 connections, Mark and Hank only have 1 connection. Hence, Kevin, Mary, Jasmin and Yara are well connected whereas Mark and Hank are worst connected.



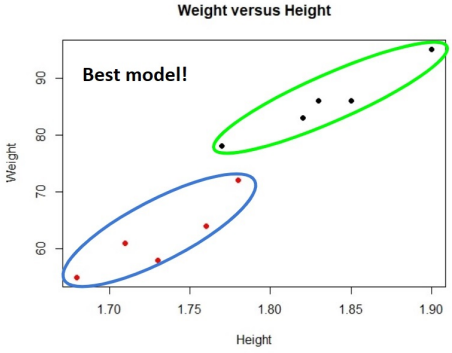
(a) Start model fitting



(b) Halfway model fitting

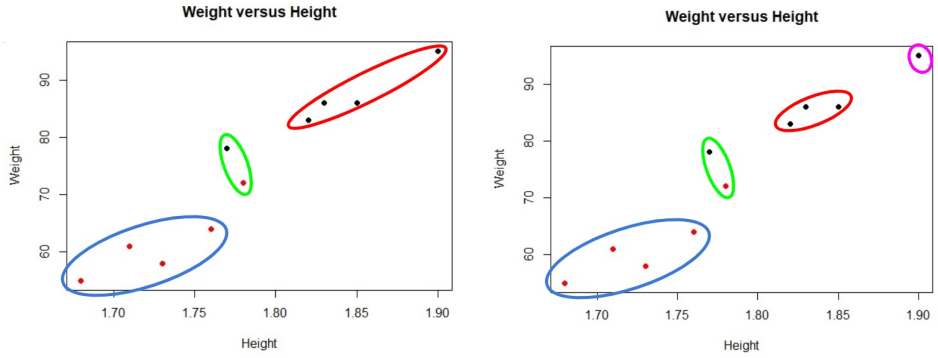


(c) Almost finished model fitting



(d) Finished model fitting

Figure 2.3: How Expectation Maximization works in mixture models: in this example the points are colour labelled to indicate gender, but we fit the mixture model to recover these labels to determine who is male and who is female, just based on height and weight. We use the variables height and weight to fit a model with 2 mixtures. When the EM-algorithm starts (a), it departs from starting values which do not provide a good fit to the data, as the algorithm continues (b) it starts capturing clusters in the height-weight space. When almost finished (c) there is one misclassification (black dot in the blue cluster), and when it is finished, the two mixtures have identified two clusters in the data, and it appears that males and females both end up on their own mixture (perfect recovery). The mixtures have been visualised as blue and green circles and these circles actually represent Gaussian distributions, which change in size and shape (mean and variance), this is typical for mixture models.



(a) Model with 3 mixtures

(b) Model with 4 mixtures

Figure 2.4: Extending the example in 2.3, where we now fit the data with 3 mixtures (a) and four mixtures (b). Both models capture subgroups of persons who are relatively similar in their height and weight.

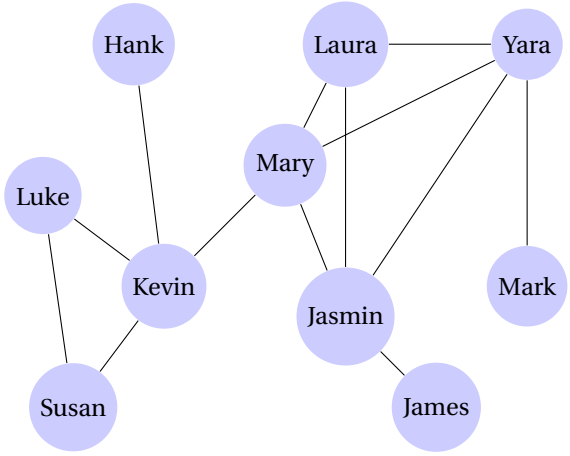


Figure 2.5: Example network of our sample. From left to right Luke and Susan are dating and both know Kevin. Kevin is also friends with Hank and Mary. Mary is friends with Kevin, Laura, Jasmin and Yara. Jasmin is friends with James. Laura and Jasmin are both friends with Yara. Mark is only friends with Yara.

In our example, the network also contains features or covariates, as presented in Table 1; for every node (person) we also have characteristics gender, height and weight. It is possible that the network structure affects the relationship between gender, height and weight. We know that most girls are shorter than boys and also see a cluster of girls (Mary, Jasmin, Laura, and Yara are densily connected). It is possible that body length correlates with network structure: persons who are shorter cluster together. In large networks these patterns are difficult to observe which has led to the development of statistical models for network analyses, such as Exponential Random Graph Models (explained below) and Autocorrelation Regression Models (ARM). The standard ARM is

$$Y = \rho W Y + X\beta + \epsilon, \qquad \epsilon \sim \mathcal{N}(0,1) \tag{2.25}$$

where ρ is a network parameter ($-1 \leq \rho \leq 1$) and W is a row-stochastic (meaning the rows sum to 1) version of our sociomatrix C. Equation 2.25 describes a normal linear regression model ($Y = X\beta + \epsilon$) to which we add a *special* part ($\rho W Y$) where we calculate ρ to correct the model with the network effect. Since W and y are readily observed in the data, ρ is a special parameter which must be estimated and indicates spatial dependence in the observations y. If the scalar parameter ρ takes a value of zero there is no spatial dependence and the model converges to a regular non-spatial regression model. How we calculate ρ is explained in chapter 3.

2.7. STOCHASTIC BLOCKMODELS

Detecting communities in graphs is a difficult and challenging task because we are trying to capture unobserved yet present structures in networks. These structures usually exist as communities or subgroups in graphs, meaning that nodes cluster together because they have an equal role in the network or are strongly connected. There are many different ways in which groups or clusters can exist in a network (see Figure 2.6), and the introduction explained that computer networks often display a disassortative structure (where nodes with a similar role in the network are less likely to be connected [49]). One approach that has been successful in capturing different types of relationships in networks is the Stochastic BlockModel (SBM).

The idea behind SBM is to move over the that are observed between two nodes $i \rightarrow j$, and $j \rightarrow i$ (if there is reciprocity), and find structurally equivalent nodes. If these structural equivalences exist, the network matrix can be partitioned into blocks of nodes with a similar role in the network. However, structural equivalence is rare in real network data so we estimate the stochastic equivalence (structurally equivalent nodes are stochastically equivalent but not vice versa) [109]. Here we describe how we estimate the stochastic blockmodel.

2.7.1. HISTORY AND DEFINITION OF SBM

Fifty years ago there was a growing application of data with some kind of social structure, but statistical models to analyse social network data were lacking. That gap resulted in the development of stochastic models using a family of probability distributions to analyze certain types of digraph data [110, 111, 112]. Those models allowed to estimate parameters that measure both the amount of reciprocation of directed edges between

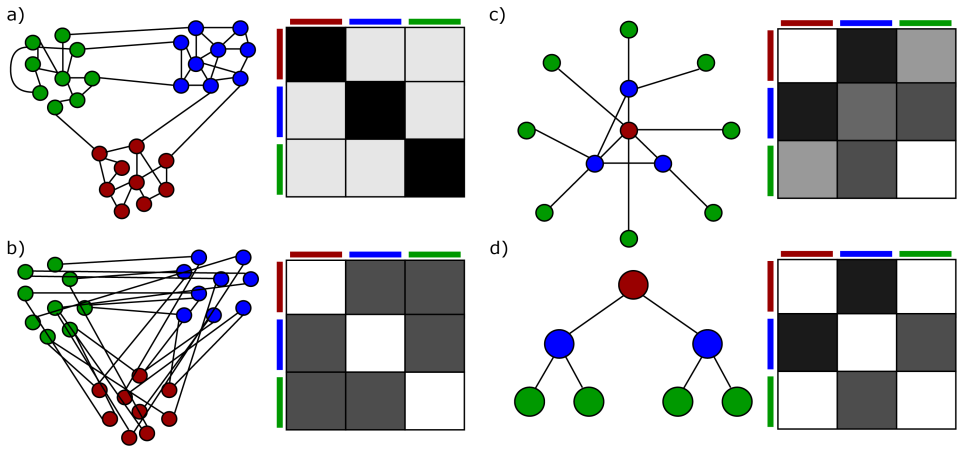


Figure 2.6: This Figure shows different types of relationships that can occur in a network; a) assortative structure, b) disassortative structure, c) coreperiphery, d) hierarchy. On the right of every network is the corresponding Stochastic BlockModel representation with probabilities in grey scale (from Funke & Becker, 2019).

nodes and the amount of differential attractiveness exhibited by each node. Today, SBMs represent a family of community detection methods, where the original SBM evolved to the degree corrected SBM [113] and SBMs where nodes are allowed to be members of multiple blocks (mixed-membership; [114]).

A formal definition is presented in [115][2]. We let G be a set of g nodes, and let $R(1),\ldots,R(m)$ be the m relations defined on the pairs of nodes. We write $iR(k)j$ to indicate that node i stands in relation $R(k)$ to node j. The adjacency matrix for the digraph of the single relation $R(k)$ is given by

$$x(k) = x_{ij}(k)), \qquad i,j = 1,\ldots,g, \tag{2.26}$$

where

$$x_{ij}(k) = \begin{cases} 1 & \text{if } iR(k)j \\ 0 & \text{otherwise} \end{cases}$$

with a zero on all diagonals of x; $x_{ii}(k) = 0$. With five nodes there are many possible configurations of edges (relations) but one example network with bidirectional relationships (called a digraph) is in Figure 2.7 and the corresponding adjacency matrix in Table 2.3. If X is a random adjacency matrix for g nodes and m relationships then the probability distribution of X denoted by $p(x) = Pr(X = x)$ is a stochastic multigraph. The latter is a graph where the nodes (vertices) are allowed to have multiple relationships (edges), for example; X_1 may refer to the amount of money send from node 1 to node 2, and X_2 may refer to the geographical distance between node 1 and node 2.

[2]The original definition [115] is based on a multigraph where there are many relationships, while I present a situation where there is a single relationship ($m = 1$) and x_{ij} is not a vector of all m relationships but a vector of a single relationship.

A SBM is a special case of a stochastic multigraph, where we partition the nodes into mutually exclusive and exhaustive subsets called blocks $\{B_1, \ldots, B_t\}$. We identify p(x) as a stochastic blockmodel with respect to the partition $\{B_1, \ldots, B_t\}$ if and only if:

1) the random vectors X are statistically independent; and

2) for any nodes $i \neq j$ and $i' \neq j'$, if i and i' are in the same node-block and j and j' are in the same node block, then the random vectors X_{ij} and $X_{i'j'}$ are identically distributed.

Point 2 implies that nodes in the same node block are stochastically equivalent, so if we have a node j in a given block B then the likelihood of any given pattern of edges with node j is the same for all nodes in the block B. Hence, node i and i' are stochastically equivalent if and only if the probability of any event about X is unchanged by interchanging nodes i and i'. Nodes are structurally equivalent if they have identical relational edges to and from all other nodes in a network [109]. For example, in Figure 2.6 part (a), all nodes that are coloured *blue* are in the same block, and therefore stochastically equivalent.

The pair of nodes i, j belongs to the pair block $B_r \times B_s$ if and only if i is in the node-block B_r and j is in the node-block B_s. If p(x) is the probability function for a SBM, X, then the pair-distribution for pair-block $B_r \times B_s$ is given by

$$p_{rs} = Pr(X_{ij} = z), \quad \text{for any } i \in B_r, j \in B_s, i \neq j \tag{2.27}$$

and

$$z = z(z(1), \ldots, z(m)), \quad z(k) = 0 \text{ or } 1. \tag{2.28}$$

Equation (2.27) indicates $p_{rs}(z)$ is the probability that z describes the pattern of relationships in the observed edges from a node in B_r to a node in B_s. This requires that the distribution of relationships between any pair of nodes in a given pair-block is the same as that of any other pair of nodes in the same pair-block, and is independent of edges between any other pairs of nodes.

There are different ways to calculate stochastic equivalence [116]. SBMs consist of a probability distribution for data and a function that maps stochastically equivalent nodes to positions. One problem is that we do not know the amount of Blocks that is optimal to map the stochastically equivalent nodes to, so [111, 117] extended the model presented in [110] for different amounts of blocks. Because the optimal amount of blocks is unobserved and has to be estimated from the data, the model has been redefined as a mixture model [118]. Now, the SBM procedure consists of; assignment of nodes to groups, estimation of the model parameters, and selection of the best fitting model to determine the optimal amount of blocks [51].

2.7.2. SBM PARAMETER ESTIMATION

We already explained the rationale behind MCMC sampling in the context of imputation. For SBM parameter fitting we use a different estimation paradigm. There is a clear division in the parameter estimation literature between sampling approximation or deterministic approximation [100]. When this thesis was written, different studies suggested that estimating parameters via sampling resulted in more accurate estimates; MCMC

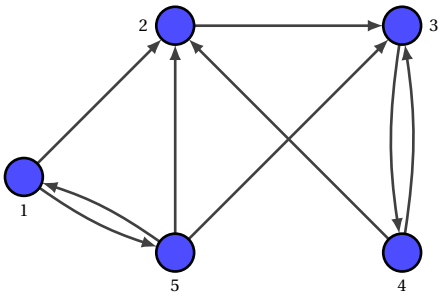


Figure 2.7: Digraph with five nodes and nine bidirectional edges (from Holland & Leinhardt [110]).

can generate exact results (given infinite computational resources) but sampling methods can be computationally demanding, often limiting their use to small-scale problems, limiting the scalability [119, 51]. Deterministic approximation is based on analytical approximations to the posterior distribution but can never generate exact results, opposing some of the characteristics of sampling methods. However, recent work shows the scalability of MCMC in SBM up to millions of [99] without the loss of accuracy, and that sampling can be faster than optimization [120].

In this thesis we use approximation via sampling for the imputation procedure, as we prefer accuracy in the imputation(s) over speed. In contrast, this thesis uses variational inference to allow approximation of the posterior distribution in highly dimensional and large datasets. Variational inference works by comparing candidate distributions against the data, where the parameters to create those distributions are varied (the variational parameters) [100]. We calculate the difference between the observed data distribution and the proposal distribution, and select the distribution that best approximates the data. A formulation of this procedure is given in chapter 4.

2.7.3. RELEVANCE OF MIXTURE MODELING AND SBM TO THIS THESIS

In chapters 4 and 5 we apply mixture models to detect communities of computers with converging behaviour in the context of malware detection. In our scenario, the computers are controlled by a botnet controller (botnets are computer networks with malware-infected machines) with the intent to use these computers for nefarious actions. Knowing which computers are infected in a computer network, based on their activity in a network, requires a statistical model to cluster this computer-activity in clusters. To as-

sign a cluster label to these *similar* computers we use mixture models on (versions of) the connection matrix. Stochastic blockmodels are used to retrieve the unobserved clusters on the networked data combined with different covariates.

2.8. CONCLUSION

This chapter presented a short overview of statistical methods developed and applied in this thesis. Normal distributions and likelihood models are used in all chapters. Regression modeling is applied in chapter 7, and Bayesian statistics is used in chapter 3. Variational inference and mixture modelling are used in chapters 4 and 5. Network modelling is used in chapters 3, 4, and 5.

Missing data mechanism	Related to	Not related to	Probability to be missing	Valid analyses
MCAR		Observed or missing data	Equality for every observation	Complete case analyses, single and multiple imputation
MAR	Oberved data	Missing data	Equal for data within groups	Multiple imputation
MNAR	Missing data		Uneqeual and unknown	Sensitivity analyses

Table 2.2: Summary of missing data mechanisms. Abbreviations; MAR: missing at random, MCAR: missing completely at random, MNOR; missing not at random. Adopted from [82].

	1	2	3	4	5	X_{i+}
1	0	1	0	0	1	2
2	0	0	1	0	0	1
3	0	0	0	1	0	1
4	0	1	1	0	0	2
5	0	1	1	0	0	3
X_{+j}	1	3	3	1	1	$9 = X_{++}$

Table 2.3: Adjacency (socio) matrix corresponding to the example network in Figure 2.7.

I

NETWORK IMPUTATION AND CLUSTERING

3

IMPUTATION OF ATTRIBUTES IN NETWORKED DATA USING BAYESIAN AUTOCORRELATION REGRESSION MODELS

Misspecification in network autocorrelation models poses a challenge for parameter estimation, which is amplified by missing data. Model misspecification has been a focus of recent work in the statistics literature and new robust procedures have been developed, in particular cutting feedback. This paper shows how this helps in a misspecified network autocorrelation model. Where model misspecification is mild and the traits are fully observed, Bayesian imputation is routine. In settings with high missingness, Bayesian inference can fail, but a closely related cut model is robust. We illustrate this on a data set of graduate students using a Facebook-like messaging app.

3.1. INTRODUCTION

In the cyber domain, large volumes of networked data are being collected, where links (or: edges) can indicate friendship (e.g. Facebook), following status (e.g. Twitter), sent and received emails, IP-traffic (e.g. botnets), financial transactions, and geolocation variables such as zip-code or country codes. Understanding network structure and topology has received considerable attention and is informative for statistical procedures such as clustering [121, 115]. Equally important is the analyses of network attributes (ie. node covariates), to understand node characteristics and the coordinating role of network topology.

Missing data is a feature in much automated or online data collection which can lead to biased estimates [86, 122]. Outside the network setting, imputation methods predom-

Published as Roeling, M.P., & Nicholls, G.K. (2020). *Social Networks (62)*, 24-32.

inantly focus on the prediction of missing values in relational datasets consisting of data from randomly selected individuals (e.g. [123, 124]). This has fed method-development for imputation of missing data from conditionally independent cases [58]. In a network setting, with observations linked through a network structure, complete-case analysis can exacerbate the problem since it is based on a fraction of the information available in the network. When well connected individuals are removed, entire clusters drop out with dramatic loss of information [125, 126]. In this setting, we may do better to reconnect the clusters by imputation of the missing values.

A previous meta-analysis of network effects [127] provided convincing evidence for homophily, the preference for associating with individuals/actors with similar attributes. If network structure is linked to actor covariate values, this may help in estimating the attribute of a node surrounded by nodes possessing data informing that attribute. On the other hand, if attribute values can be predicted from network edges, it is likely that the pattern of missing data is also affected by tie structure. The problem of incomplete tie structures has been described as part of the boundary specification problem, arising when the researcher has to decide which actors are relevant to include and which edges do not contribute (reviewed in [56]). Other factors are non-response effects in the data collection procedure (resulting in unobserved parts of the network) and fixed-choice effects (occurring when network actors are asked to nominate a fixed number of friends). Because missingness in networks is often reflected in tie structure, previous studies focusing on missing data have, for the most part, presented methods for predicting missing edges between actors [128, 57] and different strategies for (sub)network sampling [129, 130].

We are motivated by problems in which the network is observable, and there is important missing (or misleading) data in node attributes. Compared to edge-imputation, the computational work involved in imputation of node attributes is less demanding. From the perspective of parametric Bayesian inference, uncertainty in edges resembles model uncertainty, as interactions must be imputed. This leads to challenging problems resembling model averaging. Imputation of missing node attributes, conditioned on the network, is uncertainty in auxiliary variables and this leads to parameter estimation, typically more straightforward. There is also joint analysis, using parametric models for joint network and node attributes [131]. For Exponential Random Graph models (ERGMs; [132, 133, 134]) studies have proposed adaptive sampling mechanisms to acquire accurate posterior distributions [135, 128, 126, 136] under missing data. From these models, missing node attributes can be imputed. However, measuring the impact of misspecification, and treating it, is challenging due to the formal intractability of these models. Autocorrelation Regression Models (ARMs; [137, 138, 139]) provide a more straightforward setting for new statistical methods. In addition, for the probit ARM models (to accomodate categorical node attributes which we consider below), there must always be fields of missing latent continuous variables. Given these considerations, and the popularity of ARMs, we choose to illustrate misspecification-robust imputation using ARMs.

In a Bayesian setting, where missing data are treated as a latent variable, to be estimated or integrated in a joint posterior distribution along with other parameters of the model, the presence of missing data does not usually impose additional modelling.

The conditional distribution of the missing data is determined by the same observation model, model parameters and priors needed for the full-data analysis. We refer to imputation based on the joint posterior of the missing data and model parameters as a (standard) "full Bayes" analysis. In this paper we make two main points: where there is model misspecification, it may be the case that a two stage impute-and-fit approach may be preferred to full Bayes. Our second point is that the effect of model misspecification is reinforced where we have large network "fields" of coupled missing data. In this case the benefit of a two stage approach can be dramatic.

The methods we apply come from the recent literature on Bayesian theory and methods for inference from misspecified models. We demonstrate the usefulness of "cut models" [140, 141, 142]. Cut models are useful when the model is misspecified and we do not know the correct model, so we cannot immediately fix the problem by model elaboration [143]. Instead, we try to control the impact of the misspecification by cutting feedback from misspecified model elements. The designation "cut-model" is standard but misleading. Cut-model inference is a change in the inference procedure, not the model.

In the following, we define imputation with ARMs and explain how to carry out inference with a cut model. We compare the cut model approach to full Bayes in a misspecified setting, illustrating the differences. We use a publicly available social network dataset that is fully observed, and in which we introduce missing data according to different scenario's (snowball/MAR and random/MCAR sampling). Finally, in order to give a simple benchmark for comparison, and underline the robustness of our methods for network data, we compare with a simple "model-free" K-nearest-neighbour imputation procedure.

3.2. METHODS

3.2.1. DATA

Data were obtained via Dr. Tore Opsahl [https://toreopsahl.com/datasets/#online_social_network], who collected data from a Facebook-like messaging service from students at the University of California, Irvine, and was kind enough to share attribute data (gender and year of study) with us for this project. Data were available from 1899 persons (1118 females) who used the messenger application during 196 days covering the period from April to October 2004. The data included all users that sent or received at least one message during that period. These longitudinal data were collapsed into covariates; popularity (indegree + outdegree), the day somebody became an active user of the application, and the day a person reached 75% of the friends in his/her total network. 549 persons received a message but did not respond, and 37 send a message but did not receive a response. A full description of the data is presented in [144].

All data were fully observed, so we introduced missingness by removing values from the binary *gender* data $y \in \{0,1\}^n$. The data were a $n \times (p+1)$ matrix X with n rows corresponding to the $n = 1899$ people in the study, a column for the intercept, and $p = 7$ columns corresponding to covariates (indegree, five year of study levels, and day active) chosen to inform gender. A relationship network matrix C was constructed in the following way: for $i, j \in \{1, ..., n\}$ let $a_{i,j}$ denote the number of messages sent from i to j and let $C_{i,j} = \max(a_{i,j}, a_{j,i})$ be the overall network weight for edge $\langle i, j \rangle$.

In the model below, gender y will be the response, with some missing values. We use an ARM to model the relationship between y and the node covariates X in the context of network structure evidenced by C.

3.2.2. AUTOCORRELATION REGRESSION MODEL

In the Bayesian ARM for completely observed data [127, 62, 145, 146, 147], we let C be a general $n \times n$ matrix of network weights. Let $W_{i,j} = C_{i,j} / \sum_{k=1}^{n} C_{i,k}$ so that $W = [W_{i,j}]_{i=1,\ldots,n}^{j=1,\ldots,n}$ is a row-stochastic version of C. Let X be an $n \times (p+1)$ design matrix of covariates with first column corresponding to the intercept, let $\beta \in R^{p+1}$ be a vector of regression coefficients, I_n the $n \times n$ identity matrix, and $\epsilon \in R^n$ be a vector of n independent network variation differences $\epsilon \sim N(0, I_n \sigma^2)$, with $\sigma = 1$ as variance parameter which can be set equal to 1 in the probit setting of interest. Finally, $\rho \in R$ is the network autocorrelation parameter measuring the network influence. This is positive if attribute values of connected actors tend to converge and negative if those values diverge [127, 147].

The canonical spatial autoregressive model for a real response $z \in R^n$ is

$$z = \rho W z + X\beta + \epsilon \tag{3.1}$$

or equivalently

$$z = (I_n - \rho W)^{-1} X\beta + (I_n - \rho W)^{-1} \epsilon \tag{3.2}$$

Let $A_\rho = I_n - \rho W$ and $|A_\rho| = \det(A_\rho)$. The likelihood for β and ρ given fully observed z is

$$p(z|\beta, \rho) \propto |A_\rho| \exp\big(-\frac{1}{2}(A_\rho z - X\beta)^T (A_\rho z - X\beta)\big). \tag{3.3}$$

The log-determinant $\log(|A_\rho|)$ must be evaluated in order to infer ρ. This is non-trivial and we found some schemes were not numerically stable, albeit in rather extreme missingness cases. Of the three $\log(|A_\rho|)$-estimators implemented with [148] (the grid method of Pace and Barry [149], spline approximation using grid points, and a Chebyshev approximation [150]), the grid method proved most reliable, though we had agreement in all but the most extreme cases. The grid method, although robust, can be slow, and would not be used if other faster methods give adequate estimates.

BAYESIAN PROBIT ARMS FOR BINARY DATA

The probit-ARM models introduced below are parameter rich. In fact, the number of latent parameters is proportional to the number of response observations. In addition, we have missing data. In this setting some form of parameter regularisation is needed. Bayesian network model inference [61, 62] is a coherent regularisation framework. Bayesian implementations of ARM's (e.g. [145]) use Markov Chain Monte Carlo (MCMC) to summarise posterior distributions. In our setting maximum likelihood estimation can result in a downward bias of the network effect parameter ρ when cases are strongly connected [151, 152]. A number of factors contribute to this bias [153]; for example, a network effect can reduce the amount of information gained from each node.

In our data, the response variable $y_i, i = 1, \ldots, n$ is binary. Several studies have applied logit or probit Bayesian ARMs to discrete covariate data with, respectively, a dichotomous or multinomial/ordinal outcome [154, 146, 148]. In a probit ARM the binary

response y is modeled as a discretisation of an underlying continuous latent field z [98], itself following an ARM as above. The variance is fixed to unity so that the regression parameters β are identifiable [145]. For $i = 1,...,n$ we model $y_i = \mathbb{I}_{z_i>0}$, leaving us with parameters z, β and ρ, data y and a posterior distribution

$$\pi(z,\rho,\beta|y) \propto \pi(\rho,\beta)\, p(z|\rho,\beta)\mathbb{I}_{z\in\mathcal{Z}_y}, \tag{3.4}$$

where

$$\mathcal{Z}_y = \{z \in R^n : y_i = \mathbb{I}z_i > 0 \text{ for each } i = 1,...,n\}$$

and $p(z|\rho,\beta)$ is given in Equation 3.3. The prior for $z \in R^n$ is the ARM defined in Equation 3.3. This can alternatively be thought of as the observation model for the missing data z. We assume independent prior(s) for ρ and β with $\pi(\rho,\beta) = \pi_\rho(\rho)\pi_\beta(\beta)$. For the prior on $\rho \in [-1,1]$ we take a meta-analytic value based on 183 estimates of ρ [127] encountered in a wide variety of independent data sets. We summarise those data for ρ via a normal distribution with mean $\mu_\rho = 0.36$ and standard deviation $\sigma_\rho = 0.19$ truncated to the interval $[-1,1]$. Our priors for β are independent near flat normal priors with large variance ($\sigma_\beta = 10^{12}$), $\pi_\beta(\beta) = N(\beta;0,\sigma_\beta^2 I_{p+1})$.

For completely observed data, functions fitting models of this kind are incorporated in the Spatial Econometrics Toolbox for Matlab. There are some associated R packages (listed by [155]) such as *sarprobit* [148].

3.2.3. BAYESIAN INFERENCE FOR MISSING DATA

POSTERIOR PREDICTIVE DISTRIBUTION FOR MISSING DATA

We now consider imputation of missing data in the vector of responses, y. In our case y is a binary vector recording gender. In a Bayesian setting imputation of missing values is formally straightforward. The missing y-entries are unknown, and treated as parameters alongside z, β and ρ. We outline this "full Bayes" approach in this section. Our point below will be that the full Bayes approach fails where there is model misspecification combined with large amounts of missing data. However, recent developments in Bayes methods for misspecified models, and in particular the use of "cut models" [140], are robust tools for network model parameter inference and offer a way forward.

We assume the following setting: suppose we are given original data collectively $\tilde{X} = [y, X]$ which contains one column y with some missing entries. Let $y = (y_{obs}, y_{mis})^T$ with $y_{obs} = (y_1,...,y_{n-q})^T$ observed and $y_{mis} = (y_{n-q+1},...,y_n)^T$ missing, so that there are $q \in \{1,...,n\}$ missing entries in all. It is convenient to sort data in rows so that the missing data are in the last q rows. In order to impute y_{mis} we treat the full observed matrix X as a matrix of covariates and model the relation between y and X using the ARM given in Equation 3.1.

The posterior distribution conditions on the observed data only,

$$\pi(z,\rho,\beta|y_{obs}) \propto \pi(\rho,\beta)\, p(z|\rho,\beta)\mathbb{I}_{z\in\mathcal{Z}_{y_{obs}}}, \tag{3.5}$$

where

$$\mathcal{Z}_{y_{obs}} = \{z \in R^n : y_i = \mathbb{I}_{z_i>0} \text{ for each } i = 1,...,n-q\},$$

so that the sign condition on $z = (z_1, ..., z_n)$ applies only to those z_i matched with a y_i that is actually observed. The z-values matched with unobserved y-values are informed through their neighbours in the ARM. The posterior predictive distribution for y_{mis} is simulated by simulating $z|y_{obs}$ from the distribution above and setting $z_{mis} = (z_{n-q+1}, ..., z_n)$ and $y_{mis,i} = \mathbb{I}_{z_{mis,i}>0}$ for $i = 1, ..., q$. In terms of the posterior in Equation 3.5, the posterior predictive is

$$P(Y_{mis,i} = 1|y_{obs}) = P(Z_{mis,i} > 0|y_{obs}) \tag{3.6}$$

with

$$P(Z_{mis,i} > 0|y_{obs}) = \int_{z:z_{mis,i}>0} \pi(z, \rho, \beta|y_{obs}) dz d\beta d\rho.$$

We generate realisations from the marginal distribution $y_{mis}|y_{obs}$ by sampling the joint distribution $z, \rho, \beta \sim \pi(z, \rho, \beta|y_{obs})$ and setting $y_{mis} = \mathbb{I}_{z_{mis}>0}$.

In Bayesian inference for an ARM without missing data, parameter estimates are informed by the whole dataset. When there is missing data the investigator has the opportunity to control the flow of information from the imputed data back to parameters, and this leads to cut models, where parameters are estimated without feedback from imputed missing data. In a full Bayes analysis with missing data, parameters and missing data are coupled, and modelling decisions for missing data impact parameter estimates. If there is no or little model misspecification, the full Bayes approach is likely more effective compared to cut models as all the information available is reliable. Where there is model misspecification, cut models may be far more reliable.

POSTERIOR SIMULATION AND ESTIMATION FOR MISSING DATA

The full Bayes posterior $(z, \beta, \rho) \sim \pi(z, \rho, \beta|y_{obs})$ is simulated using MCMC as outlined in Algorithm 1. We run Algorithm 1 to generate $(z^{(t)}, \beta^{(t)}, \rho^{(t)})_{t=1,...,T}$ distributed asymptotically in T according to $\pi(z, \beta, \rho|y_{obs})$. For $i \in \{q+1, ..., n\}$ let

$$y_i^{(t)} = \mathbb{I}_{z_i^{(t)}>0}. \tag{3.7}$$

We estimate the missing data using the marginal posterior mode,

$$\hat{y}_i = \text{mode}(\{y_i^{(t)},\ t = 1, ...T\}). \tag{3.8}$$

To ensure an accurate posterior for ρ, a burn-in period of 1000 plus $T = 25000$ sweeps (where a sweep is one pass over all variables, equal to one loop of Algorithm 1) are used to simulate posterior distributions. The required number of sweeps was determined by targeting an effective sample size (see Table 3.1) in the thousands. We give the effective sample size (ESS) for the slowest mixing parameter, ρ, in the worst missing-data process (Snowball, with no edge matching). The ESS values of the β-parameters were similar or better. To ensure robustness of parameter estimation, we used *MultiESS* from the *mcmcse* package to estimate the effective sample and observed (see Table 3.1) that the number of missing observations influenced the effective sample size of our Markov Chain [156].

We use a mixture of Metropolis Hastings (ρ) and Gibbs (z and β) sampling [127, 62]. This is straightforward, but some details of the z-simulation in Algorithm 1 play a role in defining the cut model and it is helpful to be clear that y_{mis} plays no role in the MCMC itself.

Algorithm 1 Bayesian ARM parameter estimation.

MCMC targeting $\pi(z, \rho, \beta | y_{obs})$ in Equation 3.5.

Suppose at step $t \in \{0, 1, ..., T-1\}$ the current state of the Markov chain is $z^{(t)} = z, \beta^{(t)} = \beta$ and $\rho^{(t)} = \rho$. The state at step $t+1$ is determined in the following way. One update will be one cycle through each element of z, β and ρ.

1. Update $z|\beta, \rho, y_{obs}$: For $i = 1, ..., n$ let $W_{i,:}$ denote the i'th row of W; (A) simulate a new z-value using
$$z_i' \sim N(W_{i,:}z + X_{i,:}\beta, 1 | y_i = \mathbb{I}_{z_i' > 0})$$
if $i \leq n - q$ (note that $W_{i,i} = 0$ so the mean does not depend on z_i) and
$$z_i' \sim N(W_{i,:}z + X_{i,:}\beta, 1)$$
if $i > n - q$ and then (B) set $z_i \leftarrow z_i'$ (ie, before moving onto the next i). Denote by z' the updated z-vector.

2. Update $\beta | z', \rho, y_{obs}$: the conditional probability density of β is normal, so simulate
$$\begin{aligned}
\beta' &\sim N(\mu_\beta^*, \Sigma_\beta^*) \\
\mu_\beta^* &= (X^T X + \Sigma_\beta^{-1})(X^T A_\rho z' + \Sigma_\beta^{-1}\mu_\beta) \\
\Sigma_\beta^* &= (X^T X + \Sigma_\beta^{-1})^{-1} \\
A_\rho &= (I_n - \rho W)
\end{aligned}$$
In our case $\mu_\beta = 0$ and $\Sigma_\beta = \sigma_\beta^2 I_{p+1}$ with large σ_β^2, so these distributions simplify. Notice that μ_β^* is calculated using the new z'-values inherited from the z-update above. Denote by β' the updated β-vector.

3. Update $\rho | z', \beta', y_{obs}$: the conditional density of ρ depends on ρ through $|A_\rho|$, so Gibbs sampling is infeasible. We use Metropolis Hastings with a simple random walk proposal
$$\tilde{\rho} = \rho + uR, \quad R \sim N(0, 1) \tag{3.9}$$
where u is the tuning parameter, chosen by monitoring the acceptance rates for this step, and acceptance probability
$$\alpha(\tilde{\rho} | \rho) = \min \left\{ 1, \frac{\pi_\rho(\tilde{\rho}) p(z' | \beta', \tilde{\rho})}{\pi_\rho(\rho) p(z' | \beta', \rho)} \right\}.$$
With probability α set $\rho' = \rho$ and otherwise set $\rho' = \tilde{\rho}$.

The new state is $z^{(t+1)} = z', \beta^{(t+1)} = \beta'$ and $\rho^{(t+1)} = \rho'$.

Table 3.1: Effective sample size of ρ in the snowball/MAR sampling scenario.

%missing	Cut model	full Bayes
10%	21352.32	23008.53
25%	18658.36	24926.45
50%	26007.73	21691.17
75%	20346.38	24804.03

This Table presents the effective sample size estimates of network parameter ρ, where the ρ estimates from the fully observed model are compared with the ρ estimates from the missing data.

CUT MODEL

Cut models treat model misspecification by replacing full Bayesian inference (previous sections) with a form of multiple imputation. Suppose the entire ARM network model is misspecified (e.g. when important covariates are omitted, so that conditional independence does not hold). Estimates for parameters such as z_{obs}, which are tightly constrained by their data, may be relatively robust to model misspecification. However, z_{mis}-values are not tied to data and will settle at values consistent with each other, and the misspecified model. In a full Bayesian setting, these poorly located latent variables feedback to distort z_{obs}, β and ρ estimates. In a cut model, we cut interactions between poorly informed variables z_{mis} and the core parameters z_{obs}, β and ρ. We determine an imputation posterior distribution for the core parameters alone using otherwise standard Bayesian methods. We then use this imputation posterior distribution as "data" to estimate z_{mis}, again, using standard Bayesian methods. This means z_{mis} are informed by more reliable z_{obs}, β and ρ values.

Denote by $\pi_{cut}(z, \rho, \beta | y_{obs})$ the full distribution determined by the cut model. This will have the form

$$\pi_{cut}(z, \rho, \beta | y_{obs}) = p_{cut,mis}(z_{mis} | z_{obs}, \beta, \rho) \pi_{cut,obs}(z_{obs}, \rho, \beta | y_{obs}), \qquad z \in \mathcal{Z}_{y_{obs}}, \quad (3.10)$$

where $z = (z_{obs}, z_{mis})$ as above, and the distributions on the right hand side are defined below. In cut model MCMC, Algorithm 2, we use MCMC to simulate

$$(z_{obs}^{(t)}, \beta^{(t)}, \rho^{(t)}) \sim \pi_{cut,obs}(z_{obs}, \rho, \beta | y_{obs})$$

and then simulate a conditionally independent realisation of z_{mis},

$$z_{mis}^{(t)} \sim p_{cut,mis}(z_{mis}^{(t)} | z_{obs}^{(t)}, \beta^{(t)}, \rho^{(t)}),$$

setting $z^{(t)} = (z_{obs}^{(t)}, z_{mis}^{(t)})$, for $t = 1, ..., T$. Estimation of $\hat{y}_{mis,i}$ and further analysis is then unchanged from the full Bayes case in Section 3.2.3.

We now give details for $p_{cut,mis}(z_{mis} | z_{obs}, \beta, \rho)$ and $\pi_{cut,obs}(z_{obs}, \rho, \beta | y_{obs})$. Group the model elements according to the way they are linked to observed or missing data, dividing the ARM equations into blocks corresponding to connections between observed pairs of nodes, missing pairs of nodes, and missing and observed pairs of nodes. The $n \times n$ network weight matrix W is given in terms of its blocks as

$$W = \begin{bmatrix} W_{[obs,obs]} & W_{[obs,mis]} \\ W_{[mis,obs]} & W_{[mis,mis]} \end{bmatrix} \quad (3.11)$$

where the blocks have dimension

$$\dim W = \begin{bmatrix} (n-q)\times(n-q) & (n-q)\times q \\ q\times(n-q) & q\times q \end{bmatrix}.$$

Let $\mathbb{0}$ be an $(n-q)\times q$ matrix of zeros. Define a new cut matrix W_{cut} by removing feedback from missing to observed,

$$W^{cut} = \begin{bmatrix} W_{[obs,obs]} & \mathbb{0} \\ W_{[mis,obs]} & W_{[mis,mis]} \end{bmatrix} \tag{3.12}$$

We block covariates similarly. Let

$$\mathrm{X} = \begin{bmatrix} X_{obs} \\ X_{mis} \end{bmatrix} \qquad \text{with dimensions } \begin{bmatrix} (n-q)\times p \\ q\times p \end{bmatrix}, \tag{3.13}$$

Substituting W^{cut} for W in Equation 3.1 gives a new cut ARM,

$$z_{obs} = \rho W_{[obs,obs]} z_{obs} + X_{obs}\beta + \epsilon_{obs} \tag{3.14}$$
$$z_{mis} = \rho W_{[mis,obs]} z_{obs} + \rho W_{[mis,mis]} z_{mis} + X_{mis}\beta + \epsilon_{mis} \tag{3.15}$$

where $\beta = (\beta_1, ..., \beta_p)^T$, $\epsilon_{obs} \sim N(0, \sigma^2 I_{n-q})$ and $\epsilon_{mis} \sim N(0, \sigma^2 I_q)$. The cut distribution for the missing data is determined from Equation 3.4. Let $V = [X_{mis}, \rho W_{[mis,obs]}]$ (so V is a $q \times (n+1+p-q)$ matrix) and let $\theta = (\beta^T, z_{obs}^T)^T$ (θ is a $(n+1+p-q)\times 1$ vector). Let $A_\rho^{(mis)} = I_q - \rho W_{[mis,mis]}$. The cut prediction distribution $p_{cut,mis}$ in Equation 3.10 is

$$p_{cut,mis}(z_{mis}|z_{obs},\beta,\rho) \propto |A_\rho^{mis}| \exp\Big(-\frac{1}{2}(A_\rho^{mis} z_{obs} - V\theta)^T (A_\rho^{mis} z_{obs} - V\theta)\Big).$$

The cut posterior distribution $\pi_{cut,obs}$ on the RHS of Equation 3.10 is

$$\pi_{cut}(z_{obs},\rho,\beta|y_{obs}) \propto \pi(\rho,\beta)\, p_{cut,obs}(z_{obs}|\rho,\beta) \mathbb{I}_{z_{obs}\in Z_{y_{obs},obs}},$$

where

$$\mathcal{Z}_{y_{obs},obs} = \{z_{obs} \in R^{n-q} : y_i = \mathbb{I}_{z_i>0} \text{ for each } i = 1, ..., n-q\},$$

and likelihood from Equation 3.14,

$$p_{cut,obs}(z_{obs}|\rho,\beta) \propto |A_\rho^{obs}| \exp\Big(-\frac{1}{2}(A_\rho^{obs} z_{obs} - X_{obs}\beta)^T (A_\rho^{obs} z_{obs} - X_{obs}\beta)\Big),$$

where now $A_\rho^{obs} = I_{n-q} - \rho W_{[obs,obs]}$.

Cut models may be helpful with high missingness as even slight model misspecification can bias full Bayes estimates badly. Cut models can be characterised as Bayesian multiple imputation. Multiple imputation has two stages; an imputation stage, in which multiple copies of the missing data are imputed, followed by an analysis stage, in which a model is fit to the imputed and observed data and parameters estimated. In our setting

Algorithm 2 Cut model ARM parameter estimation.

MCMC targeting $\pi_{cut}(z,\rho,\beta|y_{obs})$ in Equation 3.10.

Suppose at step $t \in \{0,1,...,T-1\}$ the current state of the Markov chain is $z_{obs}^{(t)} = z_{obs}, z_{mis}^{(t)} = z_{mis}, \beta^{(t)} = \beta$ and $\rho^{(t)} = \rho$. The state at step $t+1$ is determined in the following way. One update will be one cycle through each element of z, β and ρ.

1. Update $z_{obs}|\beta,\rho,y_{obs}$: For $i = 1,...,n-q$ let $W_{i,:}^{cut}$ denote the i'th row of W^{cut}; (A) simulate a new z-value using

$$z'_{obs,i} \sim N(W_{i,:}^{cut} z_{obs} + X_{obs,i,:}\beta, 1|y_{obs,i} = \mathbb{I}_{z'_{obs,i}>0}).$$

   and then (B) set $z_{obs,i} \leftarrow z'_{obs,i}$ (ie, before moving onto the next i). Denote by z'_{obs} the updated z-vector.

2. Update $\beta|z'_{obs},\rho,y_{obs}$: simulate

$$\begin{aligned}
\beta' &\sim N(\mu_\beta^*, \Sigma_\beta^*) \\
\mu_\beta^* &= (X_{obs}^T X_{obs} + \Sigma_\beta^{-1})(X_{obs}^T A_\rho^{obs} z'_{obs} + \Sigma_\beta^{-1}\mu_\beta) \\
\Sigma_\beta^* &= (X_{obs}^T X_{obs} + \Sigma_\beta^{-1})^{-1} \\
A_\rho &= (I_{n-q} - \rho W_{[obs,obs]})
\end{aligned}$$

   Denote by β' the updated β-vector.

3. Update $\rho|z',\beta',y_{obs}$: We use Metropolis Hastings with a simple random walk proposal

$$\tilde{\rho} = \rho + uR, \quad R \sim N(0,1) \tag{3.16}$$

   where u is the tuning parameter, chosen by monitoring the acceptance rates for this step, and acceptance probability

$$\alpha(\tilde{\rho}|\rho) = \min\left\{ 1, \frac{\pi_\rho(\tilde{\rho}) p_{cut}(z'_{obs}|\beta',\tilde{\rho})}{\pi_\rho(\rho) p_{cut}(z'_{obs}|\beta',\rho)} \right\}.$$

   With probability α set $\rho' = \rho$ and otherwise set $\rho' = \tilde{\rho}$.

The new state is $z_{obs}^{(t+1)} = z'_{obs}, \beta^{(t+1)} = \beta'$ and $\rho^{(t+1)} = \rho'$.

4. Update $z_{mis}|z_{obs}^{(t+1)}\beta^{(t+1)},\rho^{(t+1)}$: Simulate $\epsilon_{mis}^{(t+1)} \sim N(0,\sigma^2 I_q)$ and set

$$z_{mis}^{(t+1)} = \rho W_{[mis,obs]} z_{obs}^{(t+1)} + \rho^{(t+1)} W_{[mis,mis]} z_{mis}^{(t)} + X_{mis}\beta^{(t+1)} + \epsilon_{mis}^{(t+1)}$$

   Note that since Steps 1-3 do not depend on $z_{mis}^{(t)}$, step 4 can be implemented in post-processing on the MCMC-output chain.

there is some flexibility in what we identify as missing data, and what we call a parameter. We use this flexibility to get robustness to model misspecification. Recall that y_{obs} is the observed data and β, ρ, z_{obs}, and z_{mis} and y_{mis}, are all unknown. Algorithm 2 does multiple imputation of "missing data" β, ρ, z_{obs} followed by estimation of the parameters z_{mis} and y_{mis}. When there is no model misspecification this cut model is consistent for β and ρ estimation, like full Bayes. However, in that (well-specified) case, cut models tend to give estimates with less precision, as (desirable) information spread through the network via missing data is lost.

EXPERIMENTS

From the original fully observed attribute data we created two types of missing data scenario's: Missing Completely at Random (MCAR) and Missing at Random (MAR). In MCAR, the missingness property is unrelated to the missing value itself or other attribute data. In the MAR data, the probability of being missing is the same only within groups defined by the observed data. In a network, missingness may be correlated by the network in the same way as any other node attribute. If there is a network effect on gender, there may well be a network effect on gender missingness.

In our experiments, four scenarios with 10%, 25%, 50% and 75% missing gender values (this is $q = 190, 475, 950, 1424$ missing node gender values out of $n = 1899$ in all) were created and compared with a baseline analysis of the fully observed data. For the MCAR setting, the individuals selected for imputation were selected uniformly at random. To mimic MAR, snowball sampling was used. We chose the percentage of missing node values in the snowball-sampling so that the number of "informative edges" in the snowball sampling matched the number of informative edges in the corresponding random/MCAR sampling. An edge was counted as "informative" if both the two gender node-values adjacent the edge were not missing. We refer to non-informative edges as "missing". We used the number of missing edges as a rough measure of the amount of network information in the data. Operationally, we selected m seed nodes and their direct neighbours (using *LSMI* from the *snowball* R-package) and removed their gender data. The number of nodes q removed in our MAR setup was determined by removing data at seed nodes and their neighbours until the target number of missing edges was reached. A smaller number of snowball-sampled nodes gives the same number of missing edges as a larger number of random/MCAR nodes (see Table 3.5). For example, removing data on 75% of nodes chosen completely at random gave the same number of missing edges as removing data on 37.8% of nodes chosen by snowball sampling (in one realisation of the missing-data snowball process). For further discussion of the snowball/MAR missing-data process see Section 3.5.2.

We analysed each of the four MCAR missing-data sets and each of the four MAR data sets twice, first using the (standard) full Bayes machinery of Section 3.2.3 and second using the cut model setup of Section 3.2.3. This led to four data/inference pairs of MCAR analyses and four pairs of MAR analyses. For reference, there is a single baseline Bayesian analysis made with no missing data and the same observation model and priors common to all analyses.

PERFORMANCE EVALUATION

Parameter estimates (always posterior mean unless indicated) obtained using a full Bayes analysis on the complete data can, for our purpose, be treated as the truth, since we are interested in methods which continue to recover the full-data parameter values and predict missing data well as the percentage of missing data increases. Different fitting procedures were evaluated by comparing parameter estimates and standard deviations. A method is successful (on this first criterion) if parameter estimates do not change significantly as we increase the proportion of missing data. We will see that the full Bayes analysis fails very badly on this score (due to model misspecification) but a cut model approach is much more reliable, out to even very large proportions of missing data.

Our second criterion is predictive performance on withheld data. Since we generated missing data by withholding completely random- and snowball- sampled data, the withheld data for performance evaluation was the missing data for the original analysis. We ran MCMC for each data/inference pair and used the sampled parameters to estimate parameters using the posterior mean for $\hat{\beta}$ and $\hat{\rho}$ and the posterior mode for $\hat{y}_{mis}$ (ie, Equations 3.7 and 3.8). We report the percentage of misclassified missing observations, $\sum_i \mathbb{I}(\hat{y}_{mis,i} \neq y_{true,i})/q$ not equal to its true withheld value, $y_{true,i}$ say, for each data/inference pair.

A good inference method should be well calibrated, that is $E(Y_{mis,i}|\hat{p}_{mis,i}) = \hat{p}_{mis,i}$, so predictions have the correct level of confidence. The Brier score is sensitive to calibration (and other things, see [157, 158]). The Brier Score B is given by

$$B = \frac{1}{q}\sum_{i=1}^{q}(\hat{p}_{mis,i} - y_{true,i})^2,$$

where $\hat{p}_{mis,i} = \sum_{t=1}^{T} y_{mis,i}^{(t)}/T$ is our Monte Carlo estimate of $P(Y_{mis,i} = 1|y_{obs})$ in Equation 3.6, and $y_{true,i}$ is the true value of the missing (in fact withheld) data. Smaller values of B indicate better-calibrated prediction. The misclassification rate and Brier score take values between 0 and 1. For reference, the ratio of males to females is approximately 6 : 4 in these data, so ignoring network data, taking $\hat{p}_{mis,i} \simeq 0.6$ and simply assigning values to missing data independently at random in these proportions gives (approximately) a misclassification rate of 0.48 and a Brier score of 0.24. This procedure is actually perfectly calibrated (but lacking in resolution) so 0.24 should be thought of as a reasonable score.

3.2.4. MODEL-FREE NETWORK-BASED PREDICTION METHOD

Given that model-misspecification is at the root of the difference between our cut model and full Bayes analysis, it is of interest to see how a straightforward model-free method performs. We tried a number of methods which we do not report as they gave poor performance. We report a K-nearest-neighbour scheme which is competitive.

For each node $i = n-q+1, ..., n$ with missing gender, we have covariates X_i. For each $j = 1, ...n-q$ corresponding to an observed node, let $D_{i,j} = |X_i - X_j|$ be the Euclidean covariate distance. We took the K-nearest neighbours of i in this covariate distance and predicted the value of $y_{mis,i}$ using the majority gender in this K-nearest-neighbour set. The value of K was chosen by applying the method to the fully observed part of the data and choosing K to minimise the misclassification rate on that data.

3.3. RESULTS

We illustrate our methods by recovering missing gender data (recall y_i is the binary gender variable). We begin with a brief summary of gender marked imbalance in the data. Table 3.2 describes covariates across gender. Females and males are equally popular, but females have a significantly higher in-degree (Mean = 12.47, SD = 16.15), indicating they receive more messages then males (Mean = 9.45, SD = 14.54). Males were more likely in a higher study year (Mean = 2.5, SD = 1.37) compared to females (Mean = 2.24, SD = 1.19). We selected the significant variables (indegree, year of study, and day active) as predictors for the imputation.

Table 3.2: Covariates descriptives for Males (N = 1118) and Females (N = 781).

Covariate	$mean_M$ (sd)	$mean_F$ (sd)	$t(df)$	p
Outdegree	10.92 (23.79)	10.36 (18.80)	.571 (1868.3)	.568
Indegree	9.45 (14.54)	12.47 (16.15)	-4.178 (1563.2)	<.001
Popularity	20.36 (37.01)	22.82 (33.47)	-1.509 (1776.4)	.131
Year of study	2.50 (1.37)	2.24 (1.19)	4.36 (1805.8)	<.001
Average characters	68.09 (88.16)	63.50 (75.72)	1.215 (1818.1)	.224
Day user became active	30.43 (29.22)	37.54 (35.99)	-4.568 (1449)	<.001
Day user contacted 75% of his/her friends	42.67 (33.74)	48.66 (38.21)	-3.526 (1541.7)	<.001

Descriptives for the data used in this study, where the mean values are compared between males and females. Columns are independent-sample T-test statistics (t) comparing means with p value < .05 indicating a significant effect, given a standard deviation (sd) and significance threshold depending on the stated degrees of freedom (df).

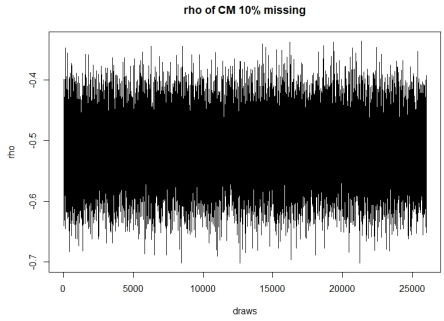
In Tables 3.3 and 3.4 we present the main results of our fitting data with random/M-CAR and snowball/MAR missingness respectively. MCMC convergence was checked and specimen traces, presented in the Supplement in Section 3.5.1, showed negligible burn-in and very good mixing. Effective sample sizes, reported in Section 3.2.3, are all over 10000. We replicated the parameter estimation results in two different missing-data subsets of the same size (see Supplementary Material Tables 3.7, 3.8) so we can be confident the results we present are representative, and not an artifact of one specific realisation of the missing data process.

To sum up the results briefly, parameter estimates in Tables 3.3 and 3.4 were far more stable (that is, they matched parameter values in the complete-data analysis) when we used the cut model. Parameter estimates remained approximately constant across rows of the cut model analysis (top half of each table) while they shrunk towards zero as we scanned across columns in the full Bayes analysis (bottom half of each table). This is what we would expect in a misspecified setting. The cut model protects parameter estimates from distortion due to model misspecification in the missing data. These improved parameter estimates then give better prediction when applied to the missing data. Interestingly, the key network parameter, ρ, was negative and significant: the residual network effect on gender, after accounting for our covariates, was anti-correlated. The significance of this effect was lost in the full Bayes analysis at high levels of missing-

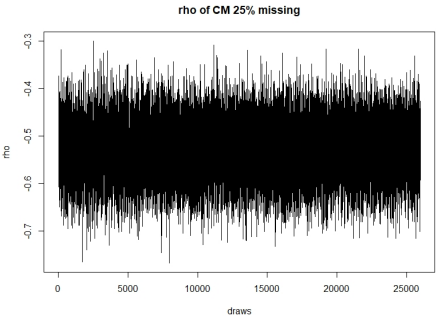
ness, but detected by the cut model.

Turning to our other criteria, the cut model had significantly smaller misclassification rate when the missingness was MCAR (Table 3.3) but only roughly equal misclassification rate when missingness was MAR (Table 3.4). The Brier scores were similar. Further investigation showed that at the highest levels of missingness, full Bayes was essentially predicting the missing binary gender data using the constant gender ratio, as there was little other information left in the data.

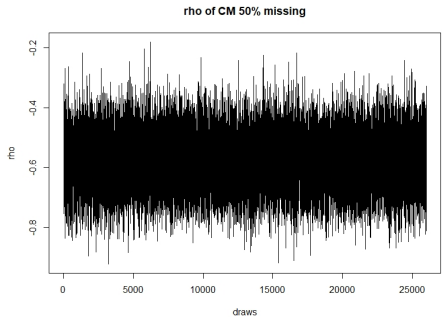
We repeated the analysis using snowball-sampling without edge-correction, which led to data with almost no informative edges. The levels of missingness are extreme, and we think network analysis is no-longer sensible. We present these results in Section 3.5.3 for completeness.



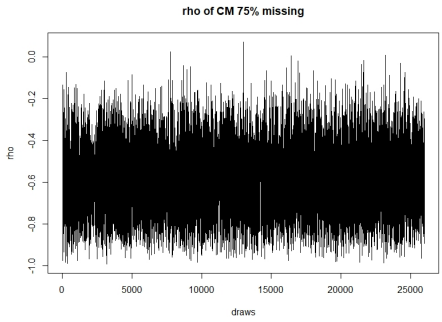
(a) 10% Missingness



(b) 25% Missingness



(c) 50% Missingness



(d) 75% Missingness

Figure 3.1: Network parameter MCMC traces from the random-missingness/MCAR sampling procedure in the cut model; burn-in of 1000 draws followed by 25000 draws.

Model-free K-nearest-neighbour analysis gave an optimal value of $K = 21$, so we assign each missing gender value by taking the modal gender of the 21 nearest gendered individuals. We tried this approach on the random/MCAR data with 10% missingness. The misclassification rate was 0.39, to be compared with the values 0.38 (cut) and 0.42 (Bayes) taken from Table 3.3.

Table 3.3: Comparison of posterior mean parameter estimates for *Gender* under a cut- and full Bayes imputation model with random missingness/MCAR-sampling.

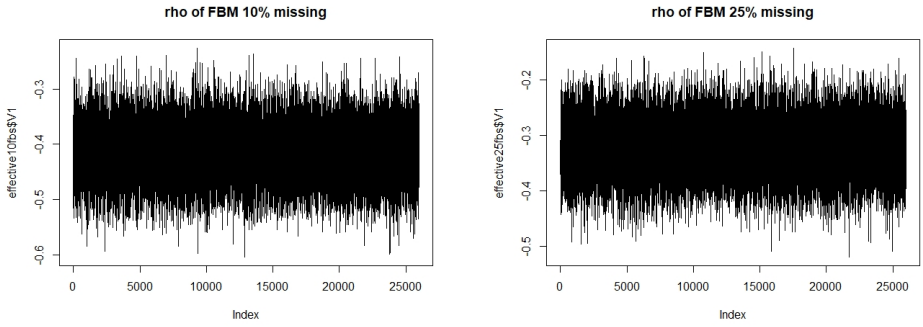
	0% missing	10% missing	25% missing	50% missing	75% missing
Parameters		CM	CM	CM	CM
Intercept	-.591(.073)	-.599(.075)	-.566(.082)	-.564(.098)	-.461(.133)
Indegree	.010(.002)	.011(.002)	.012(.002)	-.014(.003)	.013(.004)
Year of study 2	-.031(.079)	-.007(.082)	-.042(.090)	-.059(.111)	-.135(.157)
Year of study 3	-.043(.086)	-.021(.092)	-.095(.102)	-.050(.124)	.048(.170)
Year of study 4	-.087(.098)	-.044(.103)	-.098(.111)	-.077(.134)	-.152(.189)
Year of study 5	-.226(.155)	-.208(.160)	-.383(.188)	-.373(.229)	-.393(.296)
Year of study 6	-.754(.261)	-.730(.270)	-1.399(.443)	1.235(.461)	-3.737(2.611)
Day active	.005(.001)	-.005(.001)	.005(.001)	.006(.001)	.004(.002)
ρ	-.507(.045)	-.517(.050)	-.530(.059)	-.576(.090)	-.583(.143)
Misclassification rate	.331	.381	.332	.285	.220
Brier score		.267	.255	.262	.264
Parameters		FBM	FBM	FBM	FBM
Intercept		-.579(.072)	-.426(.069)	-.273(.066)	-.099(.064)
Indegree		.009(.002)	.006(.002)	.007(.002)	.003(.002)
Year of study 2		.040(.078)	-.040(.077)	-.084(.077)	-.066(.075)
Year of study 3		.035(.086)	-.078(.085)	-.058(.083)	.031(.081)
Year of study 4		.037(.098)	-.026(.095)	-.039(.093)	-.095(.092)
Year of study 5		-.148(.155)	-.265(.153)	-.405(.153)	-.075(.145)
Year of study 6		-.681(.252)	-.648(.236)	-.696(.227)	-.455(.209)
Day active		.005(.001)	.005(.001)	.004(.001)	.001(.001)
ρ		-.416(.047)	-.323(.047)	-.171(.047)	-.036(.046)
Misclassification rate		.416	.399	.418	.408
Brier score		.259	.243	.246	.247

Estimates based on 25000 MCMC steps with burn-in 1000, flat prior for β, CM = Cut Model, FBM = full Bayes Model and "Bayes" is simply the full analysis of all data. Covariate "Year of Study" is treated as a categorical variable with first-year baseline. Misclassification rate of 0% missing data is computed using leave-one-out prediction.

Table 3.4: Comparison of posterior mean parameter estimates for *Gender* under a cut- and full Bayes imputation model with snowball/MAR sampling based missingness, conditioned on the number of missing edges.

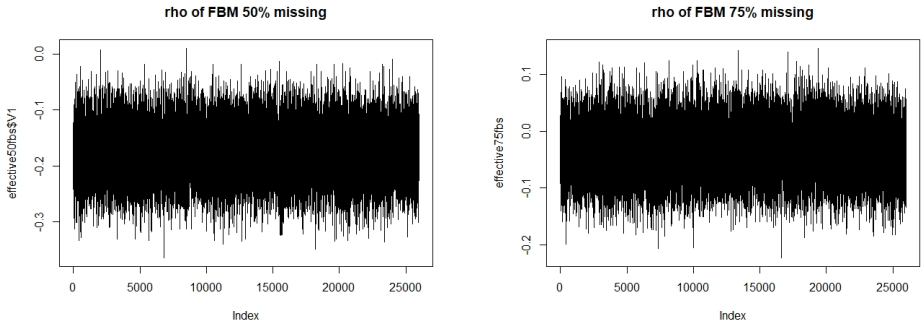
	0% missing	2% missing	9% missing	19% missing	38% missing
Parameters		CM	CM	CM	CM
Intercept	-.591(.073)	-.599(.073)	-.586(.073)	-.557(.076)	-.650(.088)
Indegree	.010(.002)	.010(.002)	.007(.003)	.011(.004)	.001(.006)
Year of study 2	-.031(.079)	-.023(.079)	-.041(.082)	-.049(.086)	-.051(.098)
Year of study 3	-.043(.086)	-.055(.088)	-.063(.091)	-.065(.094)	-.049(.109)
Year of study 4	-.087(.098)	-.127(.098)	-.142(.101)	-.146(.106)	-.066(.121)
Year of study 5	-.226(.155)	-.217(.157)	-.229(.158)	-.170(.171)	-.256(.189)
Year of study 6	-.754(.261)	-.898(.284)	-.952(.287)	-.991(.312)	-.905(.3970)
Day active	.005(.001)	.006(.001)	.005(.001)	.006(.001)	.007(.001)
ρ	-.507(.045)	-.515(.048)	-.493(.056)	-.367(.069)	-.178(.092)
misclassification rate	.331	.500	.567	.434	.528
Brier score		.309	.294	.265	.304
Parameters		FBM	FBM	FBM	FBM
Intercept		-.546(.071)	-.485(.069)	-.502(.067)	-.483(.066)
Indegree		.008(.002)	.007(.002)	.009(.002)	.008(.002)
Year of study 2		-.002(.078)	-.050(.077)	.014(.077)	-.072(.077)
Year of study 3		-.027(.086)	-.092(.085)	.017(.084)	-.059(.084)
Year of study 4		-.111(.098)	-.184(.097)	-.098(.095)	.003(.094)
Year of study 5		-.214(.155)	-.199(.152)	-.030(.150)	-.137(.150)
Year of study 6		-.692(.249)	-.743(.247)	-.535(.229)	-.266(.214)
Day active		.005(.001)	.005(.001)	.005(.001)	.005(.001)
ρ		-.443(.051)	-.306(.058)	-.183(.056)	-.097(.054)
misclassification rate		.529	.561	.439	.511
Brier score		.305	.283	.260	.268

Estimates based on 25000 draws and 1000 burn-in, m = 10, flat prior for β, CM = cut model, FBM = full Bayes model. Percentage missing nodes do not match random/MCAR levels as percentage missing edges were matched (see Table 3.5).



(a) 10% Missingness

(b) 25% Missingness



(c) 50% Missingness

(d) 75% Missingness

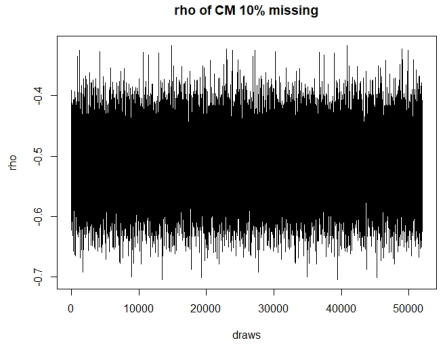
Figure 3.2: Network parameter MCMC traces from the random-missingness/MCAR sampling procedure in the full Bayes model; burn-in of 1000 draws followed by 25000 draws.

Our KNN method could be improved, for example by weighting covariates in the distance measure. However, our parametric models provide parameter estimates which are useful for interpretation, but absent in a model free approach. Our purpose here is to show that although the network based parametric model is misspecified, inference outcomes can be improved by changing the inference procedure, and not necessarily by improving the model.

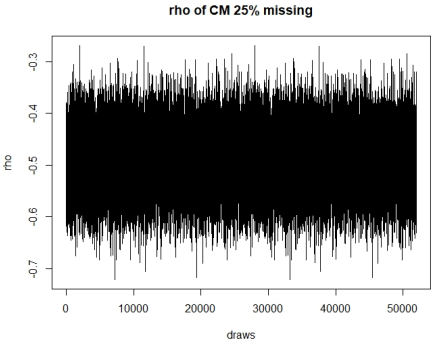
3.4. DISCUSSION

This paper gives a misspecification-robust imputation procedure for attributes in networked data via autocorrelation regression models. We used a cut model [140], where there is no feedback from the imputed data to parameter estimation, and a full Bayes approach, where feedback exists. These models were applied in different scenarios with increasing missingness.

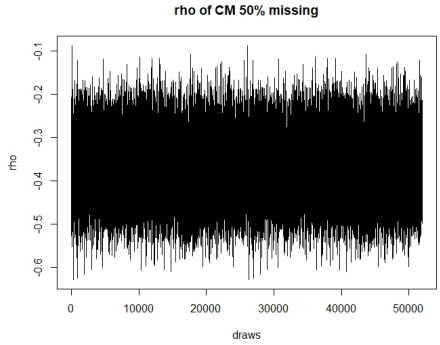
Model parameters were diversely impacted by different types of missingness. Network structure exacerbates the consequences of missing data, especially when entire clusters drop out. Most importantly, the combination of increased missing data and



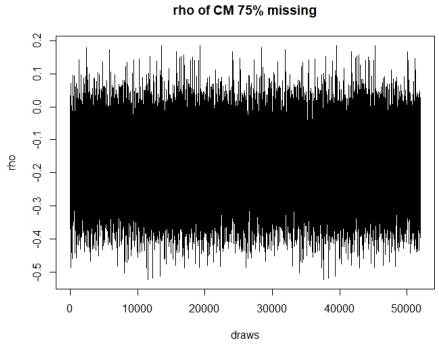
(a) 10% Missingness



(b) 25% Missingness



(c) 50% Missingness

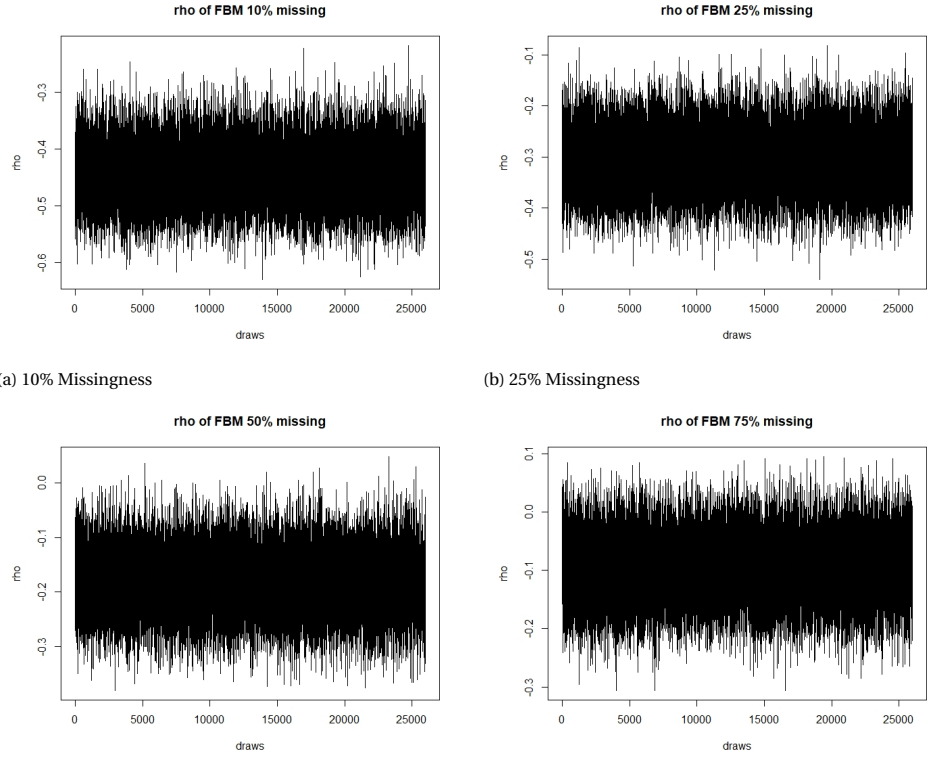


(d) 75% Missingness

Figure 3.3: Network parameter draws from the sampling procedure on the snowball/MAR sampled missingness, matching the number of missing edges, in the cut model; burn-in of 1000 draws followed by 25000 draws.

model misspecification deteriorated imputation performance. Both methods (i.e. cut model and full Bayes) fail at prediction when the missingness falls in clusters, as in the snowball-sampled case. However, cut model parameter estimates remain reliable even in this case, where full Bayes estimates shrink to zero and loose significance. Recent new methods similar in spirit to cut models, which employ "learning rates" [159] to control the contribution of misspecified model elements, seem to require more computational work than a cut model, but should improve on cut model performance.

If there is no model misspecification (for example, if we carried out this analysis on synthetic data, with parameters sampled from the prior, and data from the observation model) then straightforward Bayesian inference will be effective (in fact optimal). Moreover, because the cut model is discarding information by cutting feedback from missing data, it returns parameter estimates with greater associated variance. In this well-specified setting, straightforward Bayesian inference will give correctly calibrated estimates with greater precision. This lower-variance aspect is already visible in Tables 3.3



(a) 10% Missingness

(b) 25% Missingness

(c) 50% Missingness

(d) 75% Missingness

Figure 3.4: Network parameter draws from the sampling procedure on the snowball/MAR sampled missingness, matching the number of missing edges, in full Bayes; burn-in of 1000 draws followed by 25000 draws.

and 3.4, in the misspecified case, where we see the Bayes estimates have associated errors which are slightly, but uniformly, smaller than the corresponding cut model errors.

We briefly discussed a naive model-free alternative with a misclassification rate between the presented models. Given the quality of the spectral clustering literature, the outcome of our experiment suggests that machine learning methods may be competitive here. In some settings, recent work [160] shows dimensionality reduction and clustering methods outperform multiple imputation. This may be useful for networked data. One open problem in network based predictive mean matching is proper donor selection. More work is required to test different matching schemes in different scenarios, with different covariate types and network structure dependence, especially if small clusters are present.

This study assumed fully observed tie-structure in the network. Edge weights depended on collapsing data from 196 days, which seemed a solid solution to provide an indication whether a relationship existed (compared to cross-sectional designs) but does not completely rule out that other edges may come to existence (or break) in the future.

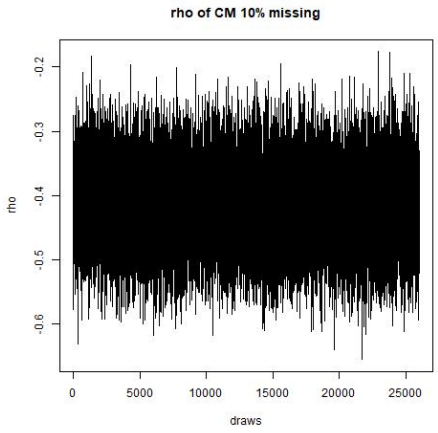
A potential misrepresentation of the network by missing edges could result in an over-or underestimation of ρ, depending on whether covariates diverse or converge with edges. Several studies have suggested imputation-like models to complete tie structure in networks (e.g. [57]), and one strategy in future studies could be to first complete the tie-structure, towards optimal estimation of ρ.

Another issue in Bayesian imputation of categorical y is imbalance in proportions in which categories are present in the observed and missing data. The gender variable used in this study was not severely imbalanced. Imbalance changes the shape of the distribution of the latent proxy z, as its distribution across the network depends on the distribution of y. When there is severe imblance, the infrequent outcome might be thought of as a kind of outlier. Our linear ARM cannot be expected to capture this variation. Cut-models are unlikely to help here and model elaboration may be needed. A non-linear Bayesian regression might help, but this is yet uncharted in the context of imputation with ARMs.

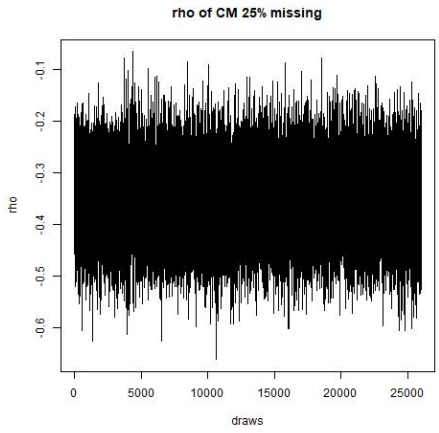
The main conclusion from our method comparison is as follows: researchers faced with missing data in Bayesian inference for network autocorrelation models are advised to try the cut model. If parameter estimates differ substantially from those estimated under full Bayesian inference then cut-model estimates are likely to be more reliable, especially when the pattern of missingness correlates with network edges and with high proportions of missing data $> 25\%$. The conclusion that cut models outperform full Bayes in a misspecified setting leads back to familiar inference schemes: inference with a cut-model is just a form of multiple imputation [96]. Data completeness and veracity are a major issue for any analyst, especially in the cyber domain. There are numerous cases where online identities are copied, faked, or profiles use false information to misguide other users (e.g. online grooming). By applying ARM imputation, attribute data from observations in a network can be completed.

3.5. SUPPLEMENTARY MATERIAL

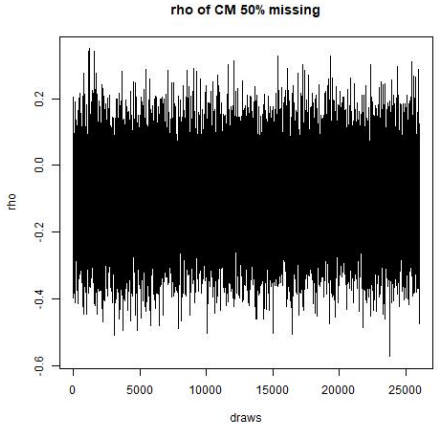
3.5.1. MCMC OUTPUT TRACES FOR SELECTED PARAMETERS



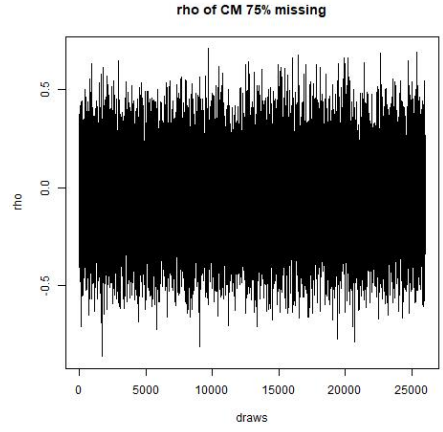
(a) 10% Missingness



(b) 25% Missingness



(c) 50% Missingness



(d) 75% Missingness

Supplementary Figure 3.1: Network parameter draws from the sampling procedure on the snowball/MAR sampled missingness, in the cut model; burn-in of 1000 draws followed by 25000 draws.

Table 3.5: Amount of remaining edges between different sampling techniques.

%nodes missing	random	snowball	edge matched snowball (%nodes)
10%	22220	13708	23072 (1.80)
25%	15002	4640	15296 (8.60)
50%	6946	590	6944 (19.2)
75%	1814	74	1802 (37.8)

This Table presents the number of remaining edges under different sampling methods. W starts complete with 27676 edges and we introduce 10, 25, 50, and 75% missingness by removing nodes. The last column presents the amount of remaining edges if matched on edges-amount, and the percentage of missing nodes that scenario corresponds to. For example, in the 50% nodes missing scenario, random sampling left 6946 edges, while if we use snowball sampling to prune a similar amount of edges, this results in 19.2% missing nodes (instead of 50%).

3.5.2. SNOWBALL SAMPLING WITH EDGE CONDITIONING

We give some further details of the Snowball/MAR sampling scheme, and in particular the edge-matching. The use of different sampling techniques to select nodes for the imputation analyses influenced model estimation. Snowball sampling tends to remove data from well-connected actors. This leads to large numbers of missing (ie non-informative) edges. When data are missing completely at random over the network, network information is retained even at very high levels of missingness, as much as 75%. A straightforward application of snowball sampling (next section of supplement) at a fixed percentage missing node values leads to extreme low levels of informative edges, so that little network information remains and there is little point in making a network analysis. The correspondence is shown in Table 3.5 and in Figure 3.3. In our Snowball/MAR missing data process we therefore match a fixed percentage missing edges. The data are missing in clusters in contrast to the random scatter generated by the random/MCAR missing-data process.

3.5.3. SNOWBALL SAMPLING WITHOUT MATCHING MISSING-EDGE COUNTS

In snowball sampling without matching missing-edges, we matched missingness levels of gender values in the MCAR setup approximately. We adjusted the number of seeds ($m = 20, 44, 159, 457$) to create four datasets with $q = 196, 401, 952, 1424$ persons with a missing value in y. This roughly matched the percentage missing nodes (10, 25, 50, 75) in the MCAR analysis. However, since data on better-connected individuals are more likely to be removed by snowball sampling than data on individuals with fewer connections, this quickly leads to a scenario with insufficient data for interesting or meaningful analysis, so in the main paper we report results for scenarios where we matched the number of edges with missing node data at both ends in MAR and MCAR.

For the unmatched snowball sampling procedure, model estimates are provided in Table 3.6. This Table can be thought of as an extension of the informative-edge-matched Table 3.4 adding columns on the right side of the Table at higher levels of missingness. Both methods are predicting very poorly as network based inference becomes irrelevant. The cut model picks up the significant negative network parameter $\rho < 0$ out to the

greatest levels of missingness.

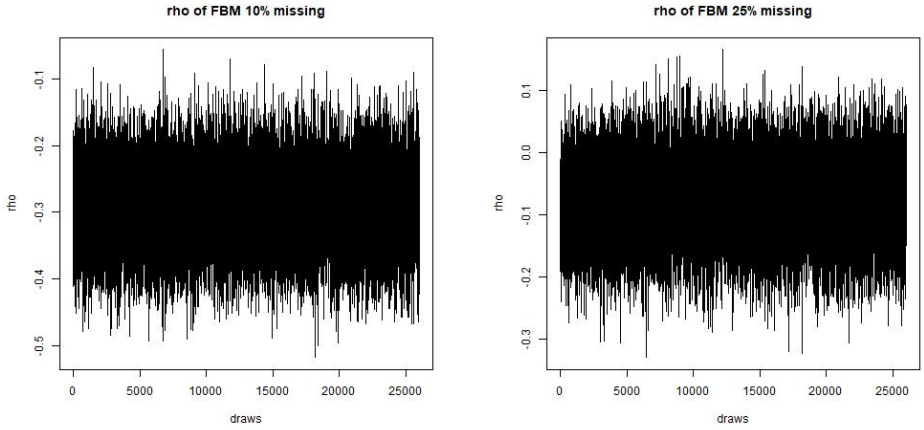
3.5.4. REPLICATION OUTCOMES

For random/MCAR missingness and snowball/MAR sampling we replicated the parameter estimation results by using different seeds as input for the sampling methods to select which nodes had their data removed (which nodes selected for imputation).

Table 3.6: Comparison of parameter estimates for *Gender* under a cut- and full Bayes imputation model with snowball/MAR sampling based missingness.

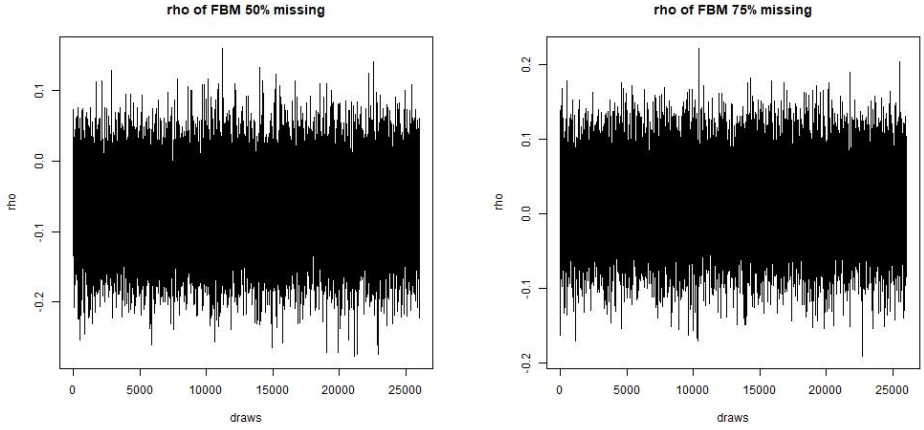
	0% missing	10% missing	25% missing	50% missing	75% missing
Parameters		CM	CM	CM	CM
Intercept	-.591(.073)	-.600(.074)	-.569(.077)	-.668(.098)	-1.022(.150)
Indegree	.010(.002)	.006(.003)	.009(.004)	.029(.011)	.176(.036)
Year of study 2	-.031(.079)	-.050(.083)	-.046(.087)	-.039(.112)	.090(.162)
Year of study 3	-.043(.086)	-.057(.091)	-.046(.096)	-.097(.121)	.011(.171)
Year of study 4	-.087(.098)	-.150(.102)	-.143(.109)	-.136(.133)	-.118(.188)
Year of study 5	-.226(.155)	-.215(.158)	-.143(.172)	-.266(.194)	-.142(.276)
Year of study 6	-.754(.261)	-.967(.299)	-.898(.309)	-.831(.418)	-13.585(8.140)
Day active	.005(.001)	.005(.001)	.006(.001)	.006(.001)	.007(.002)
ρ	-.507(.045)	-.489(.058)	-.354(.073)	-.097(.117)	-.037(.206)
Misclassification rate	.331	.622	.484	.428	.467
Brier score		.312	.268	.275	.369
Parameters		FBM	FBM	FBM	FBM
Intercept		-.507(.069)	-.482(.066)	-.378(.066)	-.314(.065)
Indegree		.005(.002)	.009(.002)	.010(.002)	.009(.002)
Year of study 2		-.047(.077)	-.045(.076)	-.004(.076)	.021(.076)
Year of study 3		-.016(.084)	-.009(.083)	-.100(.083)	.066(.082)
Year of study 4		-.150(.096)	-.145(.094)	-.131(.094)	.022(.091)
Year of study 5		-.184(.152)	-.057(.150)	-.207(.149)	-.126(.148)
Year of study 6		-.724(.246)	-.551(.227)	.015(.203)	-.492(.216)
Day active		.005(.001)	.005(.001)	.004(.001)	.003(.001)
ρ		-.305(.056)	-.076(.061)	-.068(.053)	.015(.049)
Misclassification rate		.617	.461	.441	.415
Brier score		.296	.260	.256	.254

Estimates based on 25000 draws and 1000 burn-in, flat prior for β, CM = cut model, FBM = full Bayes model.



(a) 10% Missingness

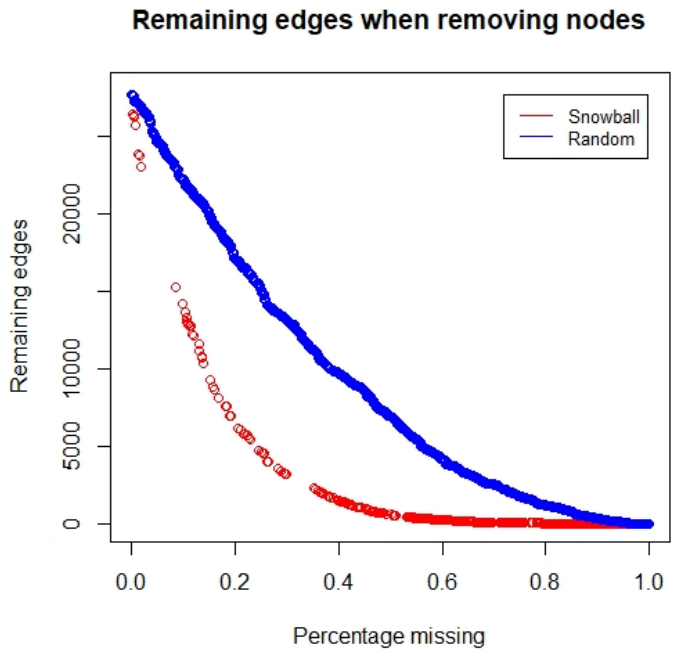
(b) 25% Missingness



(c) 50% Missingness

(d) 75% Missingness

Supplementary Figure 3.2: Network parameter draws from the sampling procedure on the snowball/MAR sampled missingness, in the full Bayes; burn-in of 1000 draws followed by 25000 draws.



Supplementary Figure 3.3: This Figure shows the number of informative edges when using different sampling methods to select nodes to remove attribute data. When selecting nodes randomly, the number of edges lost drops relatively gently, while snowball sampling drops more rapidly. Beyond about 60% node-missingness, the number of informative edges in snowball sampling is very small as only isolated nodes remain.

Table 3.7: Parameter estimates from replication with 10% random missingness.

	replication 1	replication 2
Parameters	CM	CM
Intercept	-.642(.074)	-.567(.074)
Indegree	.013(.002)	.009(.002)
Year of study 2	-.019(.084)	-.050(.082)
Year of study 3	-.039(.091)	-.024(.091)
Year of study 4	-.095(.104)	-.054(.101)
Year of study 5	-.178(.162)	-.192(.164)
Year of study 6	-.732(.264)	-.743(.267)
Day active	.006(.001)	.005(.001)
ρ	-.533(.050)	-.515(.051)
N missing	185	185
Parameters	FBM	FBM
Intercept	-.564(.071)	-.537(.070)
Indegree	.012(.002)	.010(.002)
Year of study 2	.008(.077)	-.029(.078)
Year of study 3	-.001(.086)	-.030(.086)
Year of study 4	-.125(.097)	-.038(.096)
Year of study 5	-.228(.155)	-.151(.153)
Year of study 6	-.872(.265)	-.679(.247)
Day active	.005(.001)	.005(.001)
ρ	-.439(.045)	-.450(.046)
N missing	185	185

This Table presents the parameter estimates from the two replication studies where gender of 10% of the observations was set to missing. Observations were selected via snowball/MAR sampling with a different seed. Estimates are based on 10000 draws with a burn-in of 500. The number of missing observations fluctuates due to the initial number of seed persons for the snowball/MAR sampling.

Table 3.8: Parameter estimates from replication with different snowball/MAR sampled subsets (10% missingness).

	replication 1	replication 2
Parameters	CM	CM
Intercept	-.593(.074)	-.483(.074)
Indegree	.011(.003)	.017(.003)
Year of study 2	-.038(.083)	-.052(.082)
Year of study 3	-.056(.091)	-.160(.089)
Year of study 4	-.092(.101)	-.174(.101)
Year of study 5	-.277(.161)	-.297(.161)
Year of study 6	-.985(.301)	-.886(.276)
Day active	.005(.001)	.005(.001)
ρ	-.452(.058)	-.483(.054)
N missing	234	178
Parameters	FBM	FBM
Intercept	-.505(.069)	-.390(.070)
Indegree	.007(.002)	.011(.002)
Year of study 2	.008(.077)	-.102(.077)
Year of study 3	-.005(.085)	-.220(.087)
Year of study 4	-.074(.096)	-.216(.097)
Year of study 5	-.223(.150)	-.318(.153)
Year of study 6	-.698(.236)	-.624(.231)
Day active	.004(.001)	.005(.001)
ρ	-.298(.061)	-.357(.054)
N missing	234	178

This Table presents the parameter estimates from the two replication studies where gender of 10% of the observations was set to missing. Observations were selected via snowball/MAR sampling with a different seed. Estimates are based on 10000 draws with a burn-in of 500. The number of missing observations fluctuates due to the initial number of seed persons for the snowball sampling ($N = 14$ for *set.seed(3030)* and $N = 20$ for *set.seed(6060)*).

4

STOCHASTIC BLOCKMODELS AS AN UNSUPERVISED APPROACH TO DETECT BOTNET INFECTED CLUSTERS IN NETWORKED DATA

Botnets consist of devices connected to the internet, supervised by a botnet owner, performing malicious tasks. The significant impact of botnets on corporate, governmental and civilian operations has resulted in a lot of attention from the machine learning community. However, most studies to date do not respect the linked structure of network data and rely heavily on the availability of a labelled dataset. This study applies Stochastic BlockModels (SBM) to botnet data with the aim of identifying infected clusters without the need for a labelled dataset. After providing a short review, replication, and simulation study, we apply SBMs to a publicly available dataset from the University of Victoria, including both neutral background data as well as a capture of the Zeus botnet. Our findings show that, although SBMs can be of merit in data that includes clusters of infected and uninfected traffic (and users), real world application is challenging due to the heterogeneity of the data, and the way currently available botnet samples have been collected and mapped. We discuss our findings in light of publicly available datasets and put forth suggestions for future research.

4.1. INTRODUCTION

Botnets usually consist of devices connected to the internet, supervised by a botnet owner, performing malicious tasks. Studies focusing on the taxonomy of botnets dis-

Parts of this chapter have been published as Roeling, M.P., & Nichols, G.K. (2018). Stochastic BlockModels as an unsupervised approach to detect botnet infected clusters in networked data. In N. Heard, N. Adams, P. Rubin-Delanchy, M. Turcotte (Eds.). Data Science for Cyber-Security. *Security Science and Technology, (3).*

tinguish Command and Control (C&C) and Peer to Peer (P2P) types[40, 161]. C&C botnets consist of infected computers (zombies or bots) that communicate with, and are controlled by, one centralized control centre or server. This follows the classical client-server network model. In contrast, P2P botnets rely on the infected bots to communicate commands to other infected bots. Botnets usually propagate through malware-based infection of computers, and the versatility and scalability (33% of worldwide Internet machines are infected by malware according to Panda Security) has made botnets attractive and useful tools for criminal purposes. Examples of attacks are Distributed Denial of Service (DDoS) and spam as well as fraud and theft of data or computational resources[162], illustrating the potential for botnet attacks to have a major impact on infrastructure and users in cyberspace.

Given the threat of botnets, a lot of work has focused on the detection of infected machines. Most of the detection methods rely either on malware analysis of infected machines or on differentiating normal versus malicious traffic with machine learning algorithms (reviewed elsewhere[38, 163, 39, 164, 165]). The latter approach relies on the availability of informative features (or covariates) which are generally created by researchers based on expert knowledge and data available in captured network flows. Features can be split into host-based (e.g. number of connections, ratio of source to destination ports) or flow-based (e.g. packet length, number of bytes). Features which are commonly reported to be informative are average payload packet length, average bits per second, the ratio between the number of incoming packets over the number of outgoing packets, and duration[166]. In the machine learning detection methods presented so far, it is convention to create and compare features from captures of network activity between bots and uninfected users using several techniques. These include decision trees [167, 166, 168, 169, 170, 171, 172, 173, 174, 175], distance based clustering [176, 168, 177, 174], support vector machines [169, 178, 179, 171, 174], perceptrons [168, 171], neural networks [164, 174, 180], bayesian methods [167, 170, 171, 181, 172, 174], and clustering based on local shrinking [182, 183]. Although (P2P) botnets are resilient [184], machine learning approaches have been markedly successful with reported detection rates ≥75% and occasionally ≥90%. However, a technical limitation to the currently used machine learning methods is the need for a labelled training set. Although numerous studies report an excellent performance of detection mechanisms on validation / test datasets, these validation sets often include the same botnets (only split in half randomly to create a training and test set) resulting in the evaluation against a model that is specifically tuned to the connections of the botnet in the training data. Arguably, this provides limited validity to the detection of abnormal network behaviour or other malicious traffic.

Another more relevant and immediate problem with most parametric machine learning methods is that they neglect the linked or networked structure of the data and assume conditional independence of the botnet / non-botnet status given node-based traffic summary statistics. To allow analyses, network data are typically collapsed into summary statistics for every node, indicating the node's position and properties. This assumption is clearly erroneous, as networked data are inherently dependent [185], due to unobserved latent factors acting locally on the network. Neglecting dependence implies throwing away important data and interesting interactions a priori, possibly biasing

the results of detection methods. For example, if a clustering technique identifies a small distance between two nodes in a network based on their covariates this could be (partially) explained by the distance of those nodes in the network, especially if calculation of covariate values (e.g. bytes transferred) between nodes directly depends on communication between those nodes. Another reason why neglecting the linked structure of the data may be a bad idea is that botnets are becoming increasingly advanced in mimicking normal traffic, decreasing the effectiveness of machine learning detection.

Some papers have already shown that analyses of the networked structure can be of added value by applying methods from graph theory. One study [186] calculated the degree distribution (number of connections in a given time window) to detect visited domains and found that C&C-domains receive an unexpectedly high amount of traffic. Other studies [187, 178] present clustering and connectivity techniques to provide more insight in converging patterns of communication, but do not present technical details. Clearly, network properties can influence classification accuracy, which makes the application and development of methods able to model networked data, and its covariates, particularly opportune.

This study aims to extend the application of Stochastic BlockModels (SBMs) to a cybersecurity setting, by fitting SBMs to a capture of network data including botnet infected machines. We think SBMs may assist in the detection of botnets, in a manner that is statistically sensible, without the requirement for a labelled dataset. SBMs are extensions of regular latent variable models to networked data, allowing the partitioning of vertices (nodes or addresses on the internet) of a graph into clusters that are more densely connected, and the cluster membership is inferred from the edge pattern [188]. The rationale behind using this approach is to apply an unsupervised method to discover blocks of nodes in the network given the connectivity pattern, with the aim to discover a latent class or multiple classes of malicious traffic as a subset of all classes that also include normal traffic (since some nodes in the network will likely display normal as well as abnormal behaviour).

4.2. METHODS

4.2.1. UNIVERSITY OF VICTORIA DATASET

Data were downloaded from the University of Victoria (https://www.uvic.ca/engineering/ece/isot/datasets/index.php) as made available by Saad and colleagues [189], and consisted of a collection of neutral / background data and 4 samples of botnet data.

NEUTRAL DATA

The neutral background data without infected machines were collected from the Traffic Lab at Ericsson Research in Hungary and from the Lawrence Berkeley National Lab (LBNL). The Ericsson Lab dataset contained a large number of general traffic from a variety of applications, including HTTP web browsing behaviour, gaming streams, and packets from popular bit-torrent clients such as Vuze (formerly Azureus). The LBNL is a research institute with a medium-sized enterprise network. The neutral data were collected over a three month period, from October 2004 to January 2005, include 22 subnets [189].

BOTNET DATA

A relatively small capture of Zeus botnet traffic was included, with C&C as well as P2P type traffic. Zeus is one of the biggest and most well known botnets running on Microsoft Windows, spreads through drive-by-downloads and phishing, and is estimated to have infected 3.6 million computers in the United States in 2009. It is able to secretly steal information from the infected machine, allowing the compromise of bank accounts, email and other personal files.

4.2.2. DESCRIPTIVES

Because the data were collected in separate environments the addresses of the botnet data have been mapped to match the addresses of the neutral data so that the connections seem to occur within the same network. The neutral data can be split into 6 blocks based on the time-stamp of the capture (see Table 4.1). Only one of the six captures (Table 4.1; Block 1) included IP addresses that overlapped with the Zeus botnet and that capture was used in subsequent analyses because otherwise, the botnet would form an isolated subnetwork, which is trivially identifiable. The used neutral were collected in a capture from 8/10/2007 to 10/10/2007 and included 2300385 connections. The combined Zeus botnet data included 2847 connections. First, we selected all unique pairs of connections (13609 in neutral data and 28 botnet data), and counted the number of unique nodes (9274 and 18 in neutral and botnet data, respectively) after which these matrices were merged (some nodes overlapped, e.g. IP address 172.16.2.12 was involved in both non-malicious and malicious activity). Second, these matrices were transformed into an adjacency matrix using *igraph* in R. The combined data with labels was visualized in Gephi (see Figure 4.2).

Table 4.1: Identified blocks of neutral traffic in the ISOT dataset.

Block	Date	Start time	End time	# Connections	# Addresses
1	08-10/10/2007	15:21	2:26	2300385	9274
2	04-05/10/2004	22:03	0:19	17694358	12035
3	15-16/12/2004	09:08	7:46	65255086	19560
4	16-17/12/2004	17:15	4:10	5023778	5122
5	06-07/01/2005	20:22	7:28	20511992	10501
6	07-08/01/2005	11:55	6:28	26394390	11198
Zeus C&C	17/01/2010	02:02	02.07	1632	14
Zeus P2P	26/02/2010	04:12	14.59	1215	3

Addresses = Unique Addresses; Block 1 = Traffic Lab at Ericsson Research in Hungary; Block 2-5 = Lawrence Berkeley National Lab.

4.2.3. REPLICATION OF FEATURES FROM PREVIOUS STUDIES

Previous studies have mentioned a number of features identified as valid predictors of botnet traffic. In an attempt to replicate previous studies and understand the validity of previously reported features in the current dataset we first present a naive analysis of basic data. We extracted previously reported features and statistically compared these features between non-malicious and malicious data. This is done by collapsing the infor-

mation available in the network for every node to covariates that are analysed, in replication of previous works, without considering any form of network structure.

EXTRACTION OF PREVIOUSLY REPORTED FEATURES

From the dataset, fourteen features could be extracted. These include: number of incoming and outgoing connections, length of the first packet, average packet length, the standard deviation (variability) of packet length, total number of packets exchanged, the ratio of the number of incoming and outgoing connections, the average number of packets per second, and the average duration of a connection (see Table 4.4). In most features, a distinction was made between incoming and outgoing connections. All features were analysed as continuous covariates and, in order to avoid distributional assumptions for non-malicious and malicious data, analyses were based on the non-parametric Wilcoxon rank test. This test was used to detect differences between non-malicious and botnet data.

REPLICATION OUTCOMES

The distributions of fourteen features were compared between the neutral and botnet data. Eleven of the 14 features significantly differed between the two groups (see Table 2). In neutral data, the average number of incoming packets was equal to the number of outgoing packets whereas in botnet data, these variables are unbalanced. Interestingly, the length of the first packets (incoming and outgoing) was also significantly smaller in botnet data, which is in line with earlier observations that botnets usually start with short connections followed by longer connections after establishing a solid connection [166]. Although the average length of the outgoing packets is higher in the botnet data (313.07 bytes) compared to non-malicious data (241.70), incoming connections tend to last longer (5.23 seconds in Zeus versus 1.29 seconds in non-malicious data). Standard deviations of incoming (neutral = 25.17, Zeus = 65.23) and outgoing connections (neutral = 136.9, Zeus = 150.42) do not significantly differ, suggesting that there is no increased variability in botnet data. Finally, incoming connections in non-malicious data tend to last significantly longer (134.83 seconds) compared to the botnet data (2.76 seconds), which could be the result of activity such as downloading torrents or gaming.

4.2.4. SBM MODEL

4.2.5. STOCHASTIC BLOCKMODELS

SBM stems from the merging of blockmodels and stochastic models [115]. Detailed definitions and derivations have been published elsewhere for directed [190] and undirected [191] graphs. Below follows a short outline of the model based on the work presented in Refs. [[115, 192]].

The botnet dataset consists of IP and DNS addresses (vertices) that exist on the internet, which are typically a person manning a computer or a server (hosting a website) connected (edges) in a pairwise manner. These connections can be represented by a directed binary digraph consisting of a set of g nodes. For a single relation between two addresses, the adjacency matrix is given by Y_{ij}.

The adjacency matrix was obtained (using *igraph*) after merging the edge-lists of the neutral data and Zeus data. Regardless of how many times identical pairs (e.g. a connection between IP1 and IP2) occurred over time, every ij pair (where $i \neq j$) was 1 if,

during the entire capture, at least one connection occurred between address i and address j, and 0 otherwise. The directed nature of the data made Y asymmetric as some addresses pairs had a connection in only 1 direction. The lower triangle of Y contains outgoing connection $i \rightarrow j$, and the upper triangle contains incoming connection $i \leftarrow j$. Addresses never connected to themselves so, in line with convention, $Y_{ij} = 0$, when i equals j.

This thesis considers the SBM as a general mixture model that describes the connections between nodes spread among a certain number of classes, and uses variational inference to estimate parameters. We used the model from [193] based on [51].

Consider the SBM taking as input a graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the node set of size $\mathcal{N} \in \{1, ..., n\}$, distributed over a set of $\mathcal{Q} \in \{1, ..., q\}$ latent variables capturing the communities of G. The unknown membership labels for every node are captured by a latent variable Z; $\{Z_i\}_i \overset{iid}{\sim} \mathcal{M}(1 : \alpha)$ with $\alpha = (\alpha_i, ..., \alpha_{\mathcal{Q}})$ and $\sum_q \alpha_q = 1$, indicating the Z_i are independent and identically distributed observations from a multinomial distribution. Conditionally on block membership, the edges are assumed independent, so the probability of an edge between any pair of nodes only depends on the block the two nodes belong to.

Every edge from node i to node j is associated to a random variable Y_{ij} that captures the strength of the relationship (the $\mathcal{N} \times \mathcal{N}$ adjacency matrix). SBMs are flexible enough to support different types of relationships (binary, count, continuous) by allowing different distributions Y_{ij} may take. In this thesis, Y_{ij} always takes 0 or 1 as values, where 1 indicates a connection and 0 otherwise. So that block q of node i and block l of node j, Y_{ij} has a probability distribution of Y_{ij} following a Bernoulli distribution:

$$Y_{ij}|Z_i, Z_j \overset{ind}{\sim} \mathcal{B}(\theta_{Z_i Z_j}), \qquad \forall i, j \in \mathcal{N}^2, \tag{4.1}$$

with $\mathcal{B}$ indicating the Bernoulli distribution, $\theta = (\theta_{ql})_{(q,l)\in\mathcal{Q}^2}$ is the $\mathcal{Q} \times \mathcal{Q}$ matrix with block connectivity probabilities, and $(Z_{iq})_{i\in\mathcal{N}, q\in\mathcal{Q}}$ is the $\mathcal{N} \times \mathcal{Q}$ membership matrix with the posterior probability that observation i belongs to block $\mathcal{Q}$. q indicates the group / class of node i, and l indicates the group of node j. In our model we aim to estimate $\gamma = (\alpha, \theta)$. In the undirected scenario $Y_{ij} = Y_{ji}$ and $\pi_{ql} = \pi_{lq}$ for all $(q, l) \in \mathcal{Q} \times \mathcal{Q}$, α are the mixture parameters.

We use Variational Inference (VI) to estimate model parameters. VI is classically presented as a fast parameter estimation procedure where the parameter estimates are obtained via optimization [100]. In short, a family of distributions over the hidden variables is compared (the selected functions have to come from the same distributional family as the data) and the candidate function's parameters are varied (the variational part). The model is split in global (mixture proportions, and means and variances of mixture proportions) and local (hidden cluster variable). The distance between the candidate distribution and the data distribution is quantified via the Kullback-Leibler (KL) divergence. In variational inference, the KL-divergence cannot be calculated so the method works by maximizing the Evidence Lower Bound (ELBo; [194]); a lower bound on the logarithm of the marginal probability of the observations $log p(x)$. The ELBO is optimized with coordinate ascent using the natural gradients of the ELBO. A disadvantage of Variational Inference is the requirement for multiple runs of the procedure with different starting

values, which in the SBM setting is usually done by first fitting k-means or hierarchical clustering [195, 192].

4.2.6. LIKELIHOOD

As explained, $\mathbf{Y}$ is the set of all edges and $\mathbf{Z}$ is the set of all indicator variables for nodes. The conditional independence of the edges knowing $\mathbf{Z}$ entails the decomposition of $\log\mathbb{P}(\mathbf{Z},\mathbf{Y}) = \log\mathbb{P}(\mathbf{Z}) + \log\mathbb{P}(\mathbf{Y}|\mathbf{Z})$, with log-likelihood

$$\log\mathbb{P}(\mathbf{Z},\mathbf{Y}) = \sum_i \sum_q Z_{iq}\log\alpha_q + \sum_{i\neq j}\sum_{q,l} Z_{iq}Z_{jl}\log f_{ql}(Y_{ij}), \tag{4.2}$$

obtained by summing over all possible $\mathbf{Z}$'s. Snijders and Nowicki [191] argue that models with $\mathcal{Q} > 2$ have a complexity that does not allow Maximum Likelihood estimation. Therefore SBM typically use some implementation of Expectation Maximization (EM). However, in the case of networked data we are faced with dependency between edges which make the EM procedure intractable. Therefore, a variational approach was proposed where a lower bound of the log-likelihood is maximized

$$\mathcal{J}(R_Y,\gamma) = \log\mathbb{P}(Y;\gamma) - KL(R_Y(\cdot),\mathbb{P}(\cdot|Y;\gamma)) \tag{4.3}$$

with KL refering to the Kullback-Leibler divergence and R_Y referring to some chosen distribution (in practice often from the exponential family) on $\mathbf{Z}$. This can be rewritten to

$$\mathcal{J}(R_Y,\gamma) = \mathcal{H}(R_Y) + \sum_Z R_Y(Z)\log\mathbb{P}(Y,Z;\gamma) \tag{4.4}$$

where $\mathcal{H}$ is the entropy of a distribution and the second part equals

$$\sum_Z R_Y(Z)\log\mathbb{P}(Y,Z;\gamma) = \sum_i\sum_q \mathbb{E}_{R_Y}(Z_{iq})\log\alpha_q + \sum_{i\neq j}\sum_{q,l}\mathbb{E}_{R_Y}(Z_{iq}Z_{jl})\log f_{ql}(Y_{ij}) \tag{4.5}$$

where $\mathbb{E}_{R_Y}$ denotes the expectation with respect to distribution R_Y, and requires the knowledge of $\mathbb{E}_{R_Y}(Z_{iq})$ and $\mathbb{E}_{R_Y}(Z_{iq}Z_{jl})$ for all i,j,q,l. In the Blockmodels package [192] this is implemented as a series of *for loops*:

```
{
 for(unsigned int i=0; i<lZ.n_rows; i++)
  for(unsigned int j=0; j<lZ.n_rows; j++)
   if(i!=j)
    for(unsigned int q=0; q<lZ.n_cols; q++)
     for(unsigned int l=0; l<lZ.n_cols; l++ )
      lZ(i,q) += membership.Z(j,l) * (
         logf(model,net,i,j,q,l)
          +
         logf(model,net,j,i,l,q)
      );
}.
```

To maximize $\mathscr{J}(R_Y, \gamma)$ some restrictions apply to R_Y, which means limiting the search to the class of completely factorized distributions

$$R_Y(Z) = \prod_i h(Z_i, \tau_i) \tag{4.6}$$

where h denotes the multinomial distribution and τ_i is a vector of probabilities, $\tau_i = (\tau_{i1}, \ldots, \tau_{iQ})$ summing to 1. Specifically, $\mathbb{E}_{R_Y}(Z_{iq}) = \tau_{iq}$ and $\mathbb{E}_{R_Y}(Z_{iq}Z_{jl}) = \tau_{iq}\tau_{jl}$. Finally the entropy is additove over the coordinates for the factorized (multinomial) distributions, so that equation 4.5 computationally boils down to

$$\begin{aligned}\mathscr{J}(R_Y, \gamma) = -\sum_i \sum_q \tau_{iq} \log \tau_{iq} + \sum_i \sum_q \tau_{iq} \log \alpha_q + \\ \sum_{i \neq j} \sum_{q,l} \tau_{iq}\tau_{jl} \log f_{ql}(Y_{ij}),\end{aligned} \tag{4.7}$$

with τ being the variational parameters of the functions that have to be optimized by minimizing the difference between $R_Y(Z)$ and $\mathbb{P}(Y, Z; \gamma)$.

There are different models that allow edge-weights $(i, j) \in \{1, ..., g\}$ but in this study edge weights are either 0 or 1, resulting in a Bernoulli model with $[0 \leq \pi_{ql} \leq 1]$ and $q, l \in \{1, ..., \mathscr{Q}\}^2$. We test models with $\mathscr{Q} \in \{2, \ldots, 10\}$ and determine the fit of each model with the Integrated Classification Likelihood [196]. The latter study showed that, in this setting, the ICL can be more robust than the Bayesian Information Criterion and performs well in partitioning mixture models. These algorithms have been recently implemented in the R library *Blockmodels*. [192]

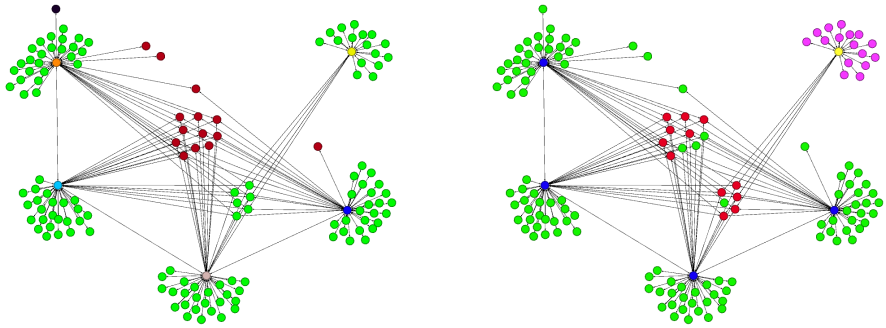
4.2.7. SIMULATION STUDY USING SBM ON SIMULATED NETWORK DATA

To illustrate the potential of SBMs in identifying clusters of network data in a botnet setting we first conducted a small simulation study. We simulated a network dataset with five users from which four are infected with a P2P botnet (see Figure 4.1a). The botnet controller directly connects only to user 1, who connects to user 2, who connects to user 3, who connects to user 4. All users visit between 14 and 30 non-malicious websites including 5 identical domains (e.g. `www.google.com`) and 25 unique domains. The four infected users also visit malicious domains as part of the botnet activity (9 visited by at least 2 users and 3 visited by only 1 user). Ultimately this dataset consisted of 121 non-malicious nodes (120 domains and 1 uninfected user), and 19 malicious nodes (1 connector, 14 malicious domains and 4 infected users)[1].

SIMULATION STUDY OUTCOMES

SBM model fitting on the simulated data revealed that a five class model ($Q = 5$) was optimal with the highest ICL estimate (see Figure 4.1b). This five class model was able to distinguish normal traffic, malicious traffic, and infected users (see Table 4.2 for the class assignment matrix). The first class captures 99 (81.8%) of the 121 non-malicious nodes and also includes 8 malicious nodes (7 domains and the botnet manager / connector). The second- and third class respectively capture the uninfected and infected users and

[1] All scripts, data and output from the simulation study can be downloaded from `https://github.com/mproeling/SBM`



(a) Simulated network with non-malicious activity from 4 infected users (orange, brown, light- and dark blue nodes) and one uninfected user (yellow) to non-malicious addresses (green nodes), and malicious activity to malicious addresses (red nodes). Nodes in the centre represent addresses visited by multiple users, both non-malicious (green) and malicious (red). The black dot in the left upper corner is the botnet controller communicating to infected user 1 (orange).

(b) Simulated network with SBM generated labels. Five classes capture non-malicious behaviour (green), predominantly malicious behaviour (red), infected users (dark blue), and the uninfected user (yellow) who also connects to unique non-malicious addresses (pink).

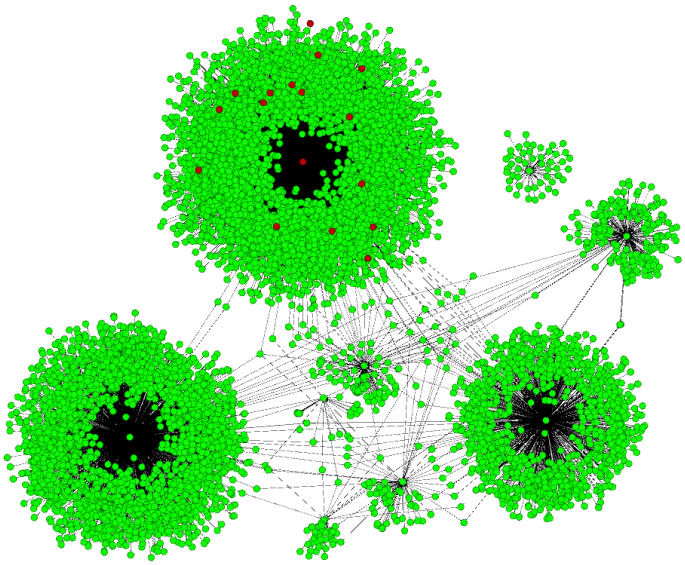
Figure 4.1: Network plots of SBM-recovery simulation study.

have perfect assignment. The fourth class captures the non-malicious domains only visited by the uninfected user. The fifth class includes the malicious and non-malicious domains visited by multiple users. If we consider classes 1, 2 and 4 as neutral and classes 3 and 5 as malicious, then we have 11 True Positives, 8 False Negatives, 5 False positives and 116 True Negatives. Hence, the accuracy of the simulation SBM is 90.7%, the sensitivity is 68.8% and the specificity is 93.5%.

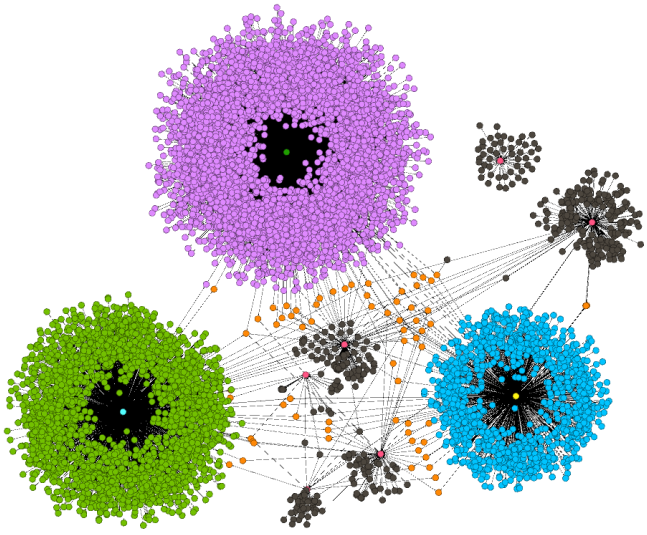
Table 4.2: Simulation study performance.

Node type	class #1	class #2	class #3	class #4	class #5
infected users	0	0	4	0	0
uninfected users	0	1	0	0	0
non-malicious	99	0	0	16	5
malicious	7	0	0	0	7
connector	1	0	0	0	0

This Table presents the frequencies of nodes as distributed over the different classes based on the SBM on the simulation data. Class 1 mostly captures non-malicious nodes and the botnet connector / controller, class 2 includes one uninfected user, class 3 captures all four infected users, class 4 includes non-malicious nodes and class 5 captures a subset of malicious and non-malicious nodes.



(a) Colour labels are based on the labels from the original dataset with non-malicious data (green) and the Zeus botnet infected nodes (red). Infections are all situated in the highly connected cluster in the top.



Colour labels are based on the SBM output with of the best fitting 9 class model, showing three highly connected clusters (purple, green, blue), one class moderately connected sub-networks (black), one class with nodes that connect different clusters (orange), and the remaining 4 classes capture the nodes that are typically in the centre of the clusters (light blue, yellow, dark green, and pink).

Figure 4.2: Network plots of the non-malicious background data and the Zeus botnet data with original and SBM labels

4.3. RESULTS FOR ISOT / ZEUS DATA

4.3.1. SBM OUTCOMES

The SBM model fitting procedure on the University of Victoria data yielded the highest ICL value in a model with $Q = 9$. From the membership matrix Z, we obtained, for every node g and for every class $q \in \{1,...,9\}$, a posterior probability, and estimate $\hat{p}_{i,q} = E(Z_{i,q}|X)$, which is an estimate of the posterior probability that node i is in group q. Table 4.3 shows the distribution of g over q. The infected nodes are distributed over two classes (3 and 5). In class 3, most of the connection (5150 / 5166 = 99.7%) is non-malicious. Yet, this class includes 94.1% of all botnet data. Class 5 is a separate class capturing only one node (IP = 172.16.2.12) involved in both non-malicious and malicious activity. Classes 1 and 7 both capture clusters with neutral data. Class 8 contains nodes that act as a hub and lie between clusters. Class 9 captures several small sub-networks. The other classes contain of a small number of central nodes from the neutral data.

Table 4.3: Distribution of malicious and non-malicious nodes across class membership.

class	#nodes	#neutral nodes	#Zeus nodes	%botnet nodes	Fig.1 class colour
1	1144	1144	0	-	blue
2	1	1	0	-	light blue
3	5166	5150	16	94.1%	purple
4	6	6	0	-	pink
5	2	1	1	5.9%	dark green
6	1	1	0	-	yellow
7	2425	2425	0	-	green
8	81	81	0	-	orange
9	465	465	0	-	black

This Table presents the frequencies of nodes as distributed over the different classes of the best fitting model, after assigning class membership to the class with the highest posterior probability.

4.4. DISCUSSION

This paper presents a first look at the use of the SBM framework for botnet discovery. The simulation study showed that SBMs can be of merit in networks with multiple infected users visiting the same (malicious and non-malicious) addresses. Interestingly, nodes that were wrongly assigned were all visited by multiple users, and botnet nodes that were wrongly identified as non-malicious were all visited by only subgroups of users (e.g. only user 1 and 2). Since botnets usually force a larger number of visitors to the same domains (e.g. DDoS attack), this could strongly increase the detection accuracy. In real world applications, SBM model performance was significantly lower: if the entire cluster that included botnet data would be treated as infected this would result in a very high false positive rate.

These outcomes should be interpreted in light of some limitations. First, the current model fitting procedure did not include covariates. Given aforementioned successes with machine learning features, the performance of our best fitting model could

increase if these features (e.g. bytes transferred or packets received) were included in the SBM. Second, modelling the temporal structure of the network data could significantly contribute to an improved detection rate [197], although such analysis would be artificial with current datasets since data are collected in separate environments, at different moments in time. Third, it is possible that the data used here does not provide an adequate capture of real botnet network activity. For example, botnet infections occurred only in one cluster, whereas in P2P botnet activity one would expect multiple clusters (or nodes across clusters) to be infected. Also, connections between clusters were unusually weak and almost always indirect, whereas in P2P traffic one would expect some form of direct link between infected nodes. Unlike the simulation data, where the simultaneous visiting of malicious websites (or the absence of visits) by different users in different clusters was a key marker for botnet detection, those patterns seemed absent in the real-world data analysed here. Finally, these data were collected in separate environments. The botnet data were collected with Virtual Machines (VM) and IP mapping was used to create one network. Most studies analyse botnet data collected in a VM [176, 166, 168, 177, 181, 170, 175] but this can be problematic, since there can be many (unobserved) factors that contribute to differences between botnet and background data. The SBM outcomes of this study do not show a strong bias of such difference as there was no immediate distinction between botnet and neutral data. Yet, merging data from different sources can be problematic and ideally one would ask a number of users to work on an infected VM during their normal internet / browsing activity, and then manually classify genuine from malicious traffic afterwards for comparison so that all activity is collected within the same setting. Another benefit of such a design would be that the neutral background data really is uninfected, which improves on current studies that assume that background data, from e.g. a University Campus, is completely free of malicious activity.

Table 4.4: Feature comparison between non-malicious and Zeus botnet data.

Feature	Mean (sd) neutral	Skewness neutral	Mean (sd) Zeus	Skewness Zeus	p
Number of packets IN	1.47 (25.00)	67.40	0.75 (.44)	-1.19	<.0001
Number of packets OUT	1.47 (61.05)	74.29	1 (0)	-	<.0001
First packet length IN	99.27 (32.57)	4.24	62.00 (0)	-	<.0001
First packet length OUT	119.82 (166.82)	2.83	57.50 (4.09)	.26	<.0001
Average length packets IN	104.89 (51.07)	11.58	96.51 (76.29)	2.39	.0026
Average length packets OUT	241.70 (240.45)	3.19	313.07 (449.61)	1.40	.0033
Sd length packets IN	25.17 (65.39)	5.61	65.23 (109.98)	1.61	.4678
Sd length packets OUT	136.39 (166.97)	1.94	150.42 (247.33)	1.38	.1090
Number packets exchanged	496.40 (18892.76)	65.78	177.94 (433.41)	2.41	<.0001
IN/OUT packets ratio	1.39 (.99)	4.85	0.69 (.46)	-.56	<.0001
Average packets time IN	1.29 (2.29)	18.33	5.23 (6.68)	1.19	.0027
Average packets time OUT	1.78 (4.61)	13.33	6.58 (12.05)	2.39	.1943
Average connection duration IN	134.83 (1744.93)	25.25	2.76 (2.73)	2.99	.0003
Average connection duration OUT	1.98 (10.76)	22.97	2.75 (2.38)	3.30	<.0001

Note. This Table presents the outcomes of the group comparison of features between neutral data and the Zeus botnet data. Degrees of freedom are rounded. **Abbreviations.** sd = standard deviation, IN = incoming connection, OUT = outgoing connection.

5

HYBRID CONNECTION AND HOST CLUSTERING FOR COMMUNITY DETECTION IN SPATIAL-TEMPORAL NETWORK DATA

This study investigates a novel combination of two sequential similarity methods (Dynamic Time Warping and N-grams with Cosine distances), with two state-of-the-art unsupervised network clustering algorithms (Hierarchical Density-based Clustering and Stochastic Block Models). A popular way to combine such methods is to first cluster the sequential network data, resulting in connection types. The hosts in the network can then be clustered conditioned on these types. In contrast, our approach clusters nodes and edges in one go, i.e., without giving the output of a first clustering step as input for a second step. We achieve this by implementing sequential distances as covariates for host clustering. While being fully unsupervised, our method outperforms many existing approaches. To the best of our knowledge, the only approaches with comparable performance require manual filtering of connections and feature engineering steps. In contrast, our method is applied to raw network traffic. We apply our pipeline to the problem of detecting infected hosts (network nodes) from logs of unlabelled network traffic (sequential data). We show that our method perfectly detects peripheral, benign, and malicious hosts in different clusters, and replicate our results in another botnet dataset with comparable performance: conjointly, 99.97% of nodes were categorized correctly.

Roeling, M.P., & Nadeem, A., Verwer, S. (in press). *Proceedings of the Workshop on Machine Learning for Cybersecurity (x)*, xx-xx.

5.1. INTRODUCTION

Spatial-temporal network data have a spatial structure, where observations are linked via single or multiple features, and a temporal structure, meaning multiple time-points are (partly) available. The analyses of the spatial element is usually performed via network clustering, which is a large field of research where a graph ($\mathscr{G}$), consisting of nodes ($\mathscr{V}$) and edges ($\mathscr{E}$), is represented by one or more pairwise distance matrices subject to an algorithm to group observations with, relatively speaking, small distances [198, 199, 200, 201, 202]. There are roughly two kinds of clustering methods: those that cluster edges (e.g. spectral-, density-, or centroid based clustering methods [203, 204]) and those that cluster nodes (e.g. community detection algorithms like Louvain clustering [121] or mixture clustering like the Stochastic Block Model [188]).

The analyses of the temporal aspect is equally complex. Apart from collapsing time-points by analyzing the mean of multiple events [171], some methods allow to analyse time-series as discrete windows. Examples of these methods are 1) creating windows and train models for each window so that state-changes over time can be identified [205]; 2) treating time as a latent variable in latent variable growth models [206]; 3) creating temporal graphs so that every pairwise interaction over time becomes a link [207]; 4) the analyses of network evolution with Stochastic Actor Based Models [208]; 5) Temporal Exponential Random Graph Models [209]; and 6) Time-contrastive learning [210]. Even more complex is the analyses of streaming data, where time cannot be treated as a strictly discrete variable either due to an arbitrary sequence in time where cutting windows is difficult, or a negative balance between the volume of time windows and the specificity (larger time windows equals lower specificity). Two common directions involve multilevel methods [211] and online-Expectation Maximization [212].

This paper focuses on unsupervised clustering of streaming spatial-temporal network data by combining node and edge clustering. We aim to present a reliable procedure to communities of nodes with converging behaviour, without the need for a labelled dataset and not requiring manual feature engineering or filtering steps. Our method computes pairwise edge distances based on the sequential behaviour of network connections using Dynamic Time Warping (distance measure for continuous sequences) and N-grams with Cosine distances (for nominal sequences), as implemented in the MalPaCA tool [213]. In order to include these distances in node clustering, the pairwise distances are aggregated via Principal Component Analysis into a small set of features. These features are added as co-variates to a node clustering algorithm based on Stochastic Block Models (SBMs), which is a well-known generative model for random graphs that produces graphs containing communities. Here, those subgroups represent hosts characterized by being connected with one another with particular edge densities [214]. Our SBM-definition is based on a recent review [215].

SBMs are attractive because they seek highly connected blocks in network connections while allowing the inclusion of features, in a statistically tractable way. This removes the need to first cluster the sequential data before analyzing the network structure or attributes as both are considered in one single node clustering algorithm. Our approach is complementary to earlier work [216] where hosts and connections were classified sequentially by first filtering P2P hosts and then categorizing P2P traffic. Using sequential features is beneficial since it reduces the required number of features as all

variation is (assumed to be) captured by the pairwise sequential distance [217, 213]. Our approach (shown graphically in Figure 5.1) does not require a priori (manual) host or sequence filtering and uses as input raw packet capture (.pcap) files.

We test our method in the setting of botnet-infected computers. Botnets are networks of computers that are infected with malware and are under the control of a botnet controller, able to use the computers for nefarious activities. Infection status is usually unknown to users or controllers and incomplete, meaning that in a large network not all computers are infected but only a relatively small number of machines can be part of a botnet. This motivates an unsupervised approach to cluster the hosts in a computer network, thereby uncovering yet unknown (latent) groups of similarly behaving hosts. The idea is that all infected hosts show different behaviour from the neutral hosts in a network and can thus be singled out, preferably in one or more dedicated clusters. We experiment with different packet thresholds to show which data-specific cutoffs are optimal (i.e. short but still informative). The reliability of our method is investigated by replicating the main result with another dataset containing different botnet captures.

Earlier in this dissertation we advocated the flexibility of the SBM allowing to capture different types of structure or clustering in networks. A recent review [195] illustrated how four types of clustering (assortative, disassortative, coreperiphery, and hierarchy) may be represented as a block structure (see Figure 2.6). This flexibility is important because not all clustering approaches work in these four types. Certainly in computer network data, as analysed in this chapter, it is suggested that nodes with similar behaviour are not necessarily linked directly, resulting in a disassortative structure [49]. As mentioned in the introduction of this thesis, two large fields of clustering-methods are spectral clustering and modularity based clustering. To test how our approach performed against one spectral clustering method (density-based clustering) and one modularity based clustering method (Louvain clustering) we applied these two clustering methods to the data and compare the results.

This paper presents the following contributions:

- We present a clustering method of network data that does not require manual filtering of observations.
- Clustering of nodes as well as edges in spatial-temporal network data is conducted in one procedure.
- We present a competitive performance in the setting of detecting malware infected computers (bots) and replicate our main result in different types of botnets.

5.2. RELATED WORK

To date, a common strategy is to collapse temporal data into aggregate values and neglect spatial structure [170, 218, 219, 220, 221, 222, 223, 224, 189, 225, 226, 227, 228, 229, 171, 181, 230, 172, 231]. This causes a loss of information as researchers remove streams of data that only occur once (e.g. because these connections are uninformative when calculating the variance of inter arrival time between packets in a sequence of connections) [167].

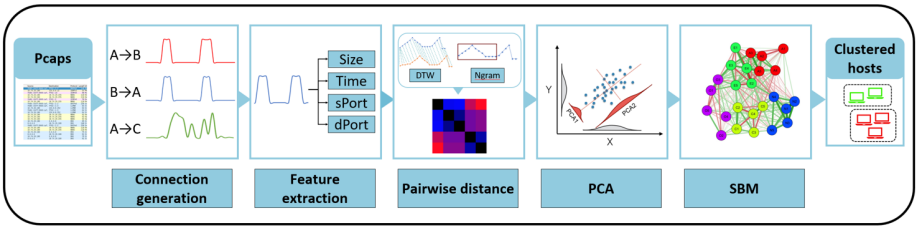


Figure 5.1: Schematic illustration of the proposed MalPaCA + SBM pipeline

Common methodological shortcomings in existing studies are (partly) neglecting the spatial-temporal nature of the (network) data [232, 170, 218, 219, 220, 221, 233, 222, 223, 224, 189, 225, 226, 227, 228, 229, 167, 171, 181, 230, 172, 164, 231, 216]. The temporal structure is usually lost through the analyses of features such as total bytes transferred with largest packets, variance of inter arrival time between packets, total bytes transferred, size of the largest packet in a flow, average size of packets in a flow, and ratio of largest packets in a flow. The main problem with these approaches is the loss of information and the gap between how the data are observed and how that situation is captured by the fitted model. Some researchers [167] remove streams of data that only occur once (e.g. a connection between computers that send only one packet), because these connections are uninformative when calculating the variance of inter arrival time between packets in a flow (sequence of connections). Apart from some studies using time-windows [234], collapsing or removing temporal information by collapsing streaming data can, at least in an unsupervised clustering setting, complicate botnet classification [235].

Apart from some studies using time-windows [234], removing temporal information by collapsing streaming data complicates botnet classification [235]. Neglecting spatial structure in botnet detection is equally problematic because this structure is informative for infection status [236]: the members of a botnet are more likely to have mutual contacts with each other than with benign hosts.

Another issue is that many studies apply some kind of manual filtering prior to analysis (e.g. removing approved DNS addresses via white-listing based on Alexa [172, 231] or other rule based exclusion criteria (e.g. [237, 238, 216]). It is unclear whether the obtained results are due to the analysis or filtering steps. Manual feature engineering may also bias the results of these experiments [239], especially when combined with sparsely reported procedures and outcomes (e.g. [228, 240]). Finally, only a few studies apply methods that do not require a labelled dataset (unsupervised learning: [224, 219]). Especially in the botnet setting where computers are *zombies* per definition, the dependence on a labelled dataset is an important shortcoming for operational usefulness.

5.3. METHODS

5.3.1. CONNECTION FEATURES

We build on a sequential feature paradigm presented recently in MalPaCA [213]: a behaviour discovery framework for network traffic which uses Hierarchical Density-Based

Spatial Clustering of Applications with Noise (HDBScan) [241], providing clusters of connection sequences.

From the original packet capture (.pcap file), we define dataframe C which is a matrix with $t \times p$ dimensions, with t rows (one row for every packet) and p features on the columns. C was made to include unidirectional connections, defined as an uninterrupted list of all packets sent from a source IP to destination IP. MalPaCA proposed to include four sequential features: packet size (bytes), time interval (gaps), source port (sport), and destination port (dport).

From every column of C we created the symmetric distance matrices D_{bytes}, D_{gaps}, D_{sport}, and D_{dport}. All distance matrices had $n_c \times n_c$ dimensions, with n_c unique unidirectional connections, and zero diagonals. For D_{bytes} and D_{gaps} the pairwise distance over time (t) was calculated via Dynamic Time Warping (DTW). For each pair of hosts we had time series $X \in \{1,...,N\}$ and $Y \in \{1,...,M\}$ and the average accumulated difference between X and Y is

$$d_\phi(X,Y) = \sum_{k=1}^{T} \frac{d(\phi_x(k),\phi_y(k))m_\phi(k)}{M_\phi} \tag{5.1}$$

with warping functions: $\phi(k) = (\phi_x(k),\phi_y(k))$, $\phi_x(k) \in \{1...N\}$, $\phi_y(k) \in \{1...M\}$, which shape the warping curve $\phi(k)$; $k \in \{1,...,T\}$. $m_\phi(k)$ is a weighting coefficient and M_ϕ is the corresponding normalization constant, which ensures that the accumulated differences in time series are comparable along different paths [242]. DTW optimises by finding the minimum the difference: dtw(X,Y) = $\arg\min_\phi d_\phi(X,Y)$ and we normalized the DTW estimates to range [0-1] with

$$\hat{x}_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{5.2}$$

where x = $[dtw(X_1,Y_1), dtw(X_1,Y_2), ..., dtw(X_{n_c},Y_{n_c})]$.

For source and destination port, the pairwise distances were calculated with the cosine similarity

$$cos(X,Y) = \frac{\sum_{k=1}^{T}(X_k * Y_k)}{\sqrt{(\sum_{k=1}^{T}(X_k^2))}\sqrt{(\sum_{k=1}^{T}(Y_k^2))}} \tag{5.3}$$

which were normalized as described to form D_{sport} and D_{dport}.

5.3.2. HOST FEATURES

The Stochastic Block Model (SBM) required to transform the connection distance matrices (D_{bytes}, D_{gaps}, D_{sport} and D_{dport}) to host distance matrices, which was achieved via Principal Component Analyses (PCA). The PCA works by calculating the singular value decomposition of the distance matrices so that by maximizing the variation captured per component a small number of components (ideally) captures a major proportion of the variation. We input the distance matrices so the aim was to acquire a number of dimensions less than the number of unique connections, accomplished by selecting the m components explaining at least 40% cumulative variation. For each of the 4 features, the PCA thus resulted in a matrix W with n_c rows and m columns, so that for each unique

D	ab	ac	bc	ca
ab	0	689	1262	512
ac	689	0	1169	680
bc	1262	1169	0	1062
ca	512	680	1062	0

W	m_1
ab	-3.18
ac	-2.96
bc	-4.60
ca	-2.92

Y_{m_1}	a	b	c
a	0	-3.18	-2.96
b	0	0	-4.60
c	-2.92	0	0

Table 5.1: A fictional example of a distance matrix D_{bytes}, PCA component weights matrix W_{bytes}, and corresponding SBM covariate matrix Y_{bytes,m_1}.

$a \rightarrow b$ connection m, component weights were available. We used W to create m host-host SBM covariates. Since every row of W referred to a unique $a \rightarrow b$ connection, the connection source (a) and destination (b) are used to indicate the rows and columns for each SBM covariate matrix Y_m with dimensions $n_h \times n_h$ where n_h is the unique number of hosts. Hence, the values in Y_{bytes,m_1}, the SBM covariate matrix for the first component of *bytes*, were inherited from m_1 of W_{bytes} (see Table 5.1).

5.3.3. STOCHASTIC BLOCK MODEL

The SBM model is explained in 4.2.4 and this paragraph only provides a short description for completeness. The SBM took as input a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ was the node set of size $n_h := |\mathcal{V}|$, and $\mathcal{E}$ was the edge list of size $M := |\mathcal{E}|$. The corresponding $n_h \times n_h$ adjacency matrix was denoted by Y, where $Y_{ab} = 1$ if there was a connection between hosts a and b and 0 otherwise. The main input graph was an undirected binary node matrix Y_{class} which held a 1 if there was any connection between nodes a and b; $Y_{class,ab} = 1$ or zero otherwise. The generated SBM covariate matrices are added to the model as covariates

$$SBM(Y_{class,ab}, List(Y_{packetSize,m}, Y_{gapsDist,m}, Y_{sourcePort,m}, Y_{destPort,m}))$$

Since group (g) membership is unknown, the membership labels for every host are captured by a latent variable Z_a, which elements are all 0, except exactly one that takes the value 1 and represents the group host a belongs to. This Z_a is assumed to be independent of Z_b for $a \neq b$. Finally, SBM outputs a $n \times g$ matrix $Z := (Z_1, ..., Z_n)^T$, such that $Z_{a,i}$ is the i^{th} element of Z_a. Graph generation and likelihood are explained elsewhere [215]. The lower and upper bound of fitted SBM models were 2 and 10. Model fit was evaluated with the Integrated Classification Likelihood (ICL), via a variational expectation maximization approach implemented in R [192].

5.3.4. OTHER CLUSTERING METHODS

As spectral clustering method we chose density based clustering (dbscan), which calculates for every point in a given space a neighbourhood through a radius [204]. Depending on how many points fall in this neighbourhood, observations are identified as core points, reachable points, outliers, or noise. This allows to group points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions (whose nearest neighbors are too far away). As modularity-based clustering method we applied Louvain clustering [121]. Modularity is a scaled

value that lies between -0.5 and 1 measuring the relative density of edges inside communities with respect to edges outside communities. Every node starts as his own community and these communities are merged if they survive a loss function. Both clustering methods were applied directly (and only) to the [0-1] network matrix.

5.3.5. EXPERIMENTAL SETUP

This study used data from the Malware Capture Facility Project, which is a sister project of the Stratosphere IPS Project: an initiative to obtain malware and neutral data. From all the published samples, a dataset was selected which included both neutral ($N_b = 12$) and infected ($N_i = 10$) hosts and included the entire network. The malicious hosts were infected with the Conficker botnet. The data were downloaded from https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-91/ as a .pcap file consisting of 198818 lines (packets), capturing 1011 unique ($a \rightarrow b$) connections. There were 3 isolated clusters which were removed, leaving 917 unique connections. The correlation between covariates was low (see S1) so instead of combining the distance matrices they were included in the SBM as individual predictors.

Not all observed connections are necessarily informative, so we experimented with a minimum number of packets-threshold (P_t) to ensure that the remaining connections represented sufficient information for effective behavioural modeling. The thresholds tested were $P_t \in \{5, 10, 15, 20\}$, respectively pruning to 631 (62.4%), 565 (55.9%), 523 (51.7%), and 483 (47.8%) connections (see Table 5.2). From analyses we determined that for this dataset a packet threshold of 10 is desirable, balancing the number of connections, nodes, MalPaCA and SBM clusters (see Supplementary Material). Higher thresholds resulted in too much pruning of the network structure, hindering accurate classification in this dataset.

Table 5.2: Descriptives of the Stratosphere CTU-91 data with different behavioural thresholds

Covariate	N_{seq}	N_{ip}	$Q_{MalPaCA}$	$outliers$	Q_{SBM}
5 packets	631	205	10	120	4
10 packets	565	182	9	154	4
15 packets	523	165	7	40	4
20 packets	483	148	6	38	5

This Table presents the number of unique $a \rightarrow b$ sequences (N_{seq}), unique hosts (N_{ip}), the optimal number of clusters ($Q_{MalPaCA}$) and $outliers$ determined by MalPaCA, and optimal SBM-cluster solution (Q_{SBM}).

5.3.6. REPLICATION SAMPLE

For replication of our main finding we used the ISOT dataset from the University of Victoria (https://www.uvic.ca/engineering/ece/isot/datasets) as presented in [189], which included of a collection of neutral / background data and 4 samples (Waledac, Storm, Zeus) of botnet data. Storm, Waledac, and Zeus are Windows targeting botnets predominantly used in spamming campaigns which peaked in 2007-2008. They can all be managed via a Command and Control as well as Peer to Peer communication. From the neutral data we selected the data from the Traffic Lab at Ericsson Research in

Hungary [243]. The latter contained a large number of general traffic from a variety of applications, including HTTP web browsing behaviour, World of Warcraft gaming packets, and packets from popular bittorrent clients. ISOT documentation states IP addresses of infected machines were mapped to the background traffic and all trace file were replaced to homogenize network behaviour. The infected data contained 747264 packets with 25308 unique connections and the Ericsson lab data included 2300385 packets from 12778 unique connections. These two sets were combined so that MalPaCA features could be extracted.

Table 5.3: MalPaCA clusters and infection status in the CTU-91 data. Connections in -1 are unclustered. $srcip_p, srcip_n, scrip_i$ are connections where the source host was peripheral, neutral, or infected (respectively). The same for destination ports $dstip$.

Cluster	$srcip_p$	$srcip_n$	$srcip_i$	$dstip_p$	$dstip_n$	$dstip_i$
-1	8	6	23	17	10	10
1	0	0	14	14	0	0
2	10	0	0	0	0	10
3	119	0	0	0	0	119
4	62	0	0	0	0	62
5	0	0	125	125	0	0
6	0	12	78	73	10	7
7	0	4	4	0	0	8
8	0	0	8	0	8	0
9	0	10	0	0	0	10

5.4. RESULTS

5.4.1. STRATOSPHERE DATA

MALPACA DIRECTLY

Applying MalPaCA directly to the data assigned the connections to 9 dense clusters (see Table 5.3). Visual inspection of the nodes belonging to the connections classified as outliers revealed that these were mostly peripheral, supporting the notion that nodes on the edges of the network, with negligible activity, are more likely to fall outside a MalPaCA cluster.

Different subsets of connections were identified. Cluster 1 captured all traffic from 192.168.0.118 to peripheral hosts. Cluster 3 included bidirectional traffic between neutral and infected hosts as well as connections from neutral to neutral, infected, and peripheral hosts. Clusters 4 and 5 included connections from neutral and infected to peripheral hosts (opposite to cluster 2: peripheral to infected and neutral), but apparently specific clusters were required to capture specific connections from peripheral to infected (clusters 6 and 7) and infected to peripheral hosts (clusters 8 and 9), illustrating the heterogeneity in connections from and to infected nodes. Relating the connections to their respective nodes, we identified 11 true negatives (cluster 1), 11 false positives (clusters 2:5), and 389 true positives, yielding an accuracy of 97.32%, sensitivity of 100% and specificity of 50%.

SBM DIRECTLY

Fitting the SBM directly on the network matrix, ignoring the MalPaCA features, resulted in a 6-class solution. This solution was incapable of distinguishing neutral and peripheral nodes (as described earlier in [235]). Class 1 and 3 captured 11 peripheral and 2 neutral hosts, class 2 and 5 respectively captured 2 and 3 infected hosts, class 4 included 3 neutral and 5 infected hosts, and class 6 only included 148 peripheral hosts. Hence, there are 10 true positives, 3 false positives (class 4), and 312 true negatives, resulting in a performance of: accuracy = 99.08%, sensitivity = 100% and specificity = 99.05%

OUR APPROACH

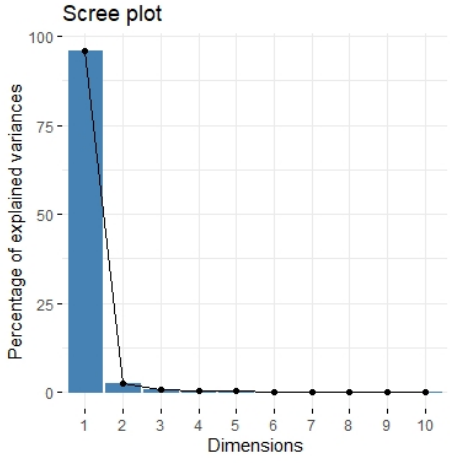
Applying MalPaCA to obtain the distance matrices, representing the distances between connections for the four features, resulted in 565 surviving connections. The average connection length was 348.48, with a minimum of 10 packets ($P_t = 10$) and a maximum of 5333.The PCA solution on the MalPaCA distance matrices commended a 1 (bytesDist), 3 (destPort), 1 (gapsDist), and 3 (sourcePort) component solution that cumulatively explained > 40% of the variation (see Figure 5.2). This result was P_t invariant; including more packets per connection does not change the amount of variation explained by the components.

Fitting the SBM on the PCA derived covariates favoured a 4-class solution (see Figure 5.3). The network with original- and cluster labels is visualized in Figure 5.4 and the performance matrix for the 10 threshold solution is provided in Table 5.5. After obtaining the cluster solution we used straightforward descriptive analyses and visualization to interpret the clusters (see Supplementary Material and [213]). We found that all malicious hosts were assigned to one cluster with a posterior probability of > .998. Most of the peripheral hosts were captured by one cluster, indicating behavioural similarity, with a class assignment posterior probability of .9982. The non-infected / neutral hosts were divided over two clusters, that also included peripheral hosts. Only one neutral host had a posterior probability < .95, which was host 192.168.1.6 with .82, with the remaining probability belonging to the other *neutral/mixed* class. If we consider all peripheral hosts (136+9+1) and neutral hosts (4+3) to be true negatives, and the correctly clustered infected hosts as true positives, the classification is perfect. These findings are consistent for all four tested packet thresholds (P_t).

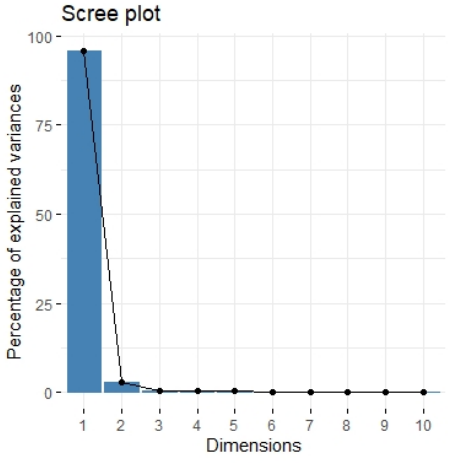
Table 5.4: Performance comparison with other studies using ISOT data

method	*accuracy*	*sensitivity*	*specificity*	*study*
BClus	.5	.4	.5	[244]
CAMNEP	.5	0	.9	[244]
BotHunter	.4	.01	.9	[244]
BotGM	.91	.83	n.p.	[217]
Decision tree	.99	.98	n.p.	[225]
Decision tree	.75	.99	n.p.	[166]

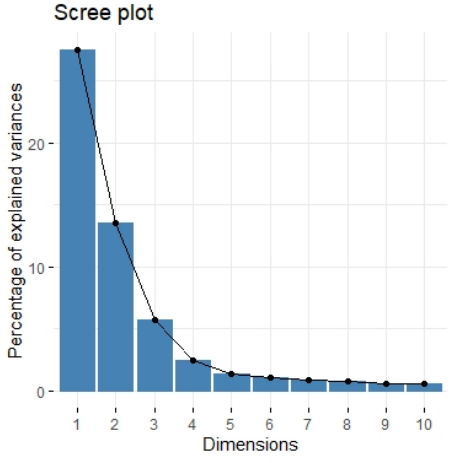
n.p. = not provided



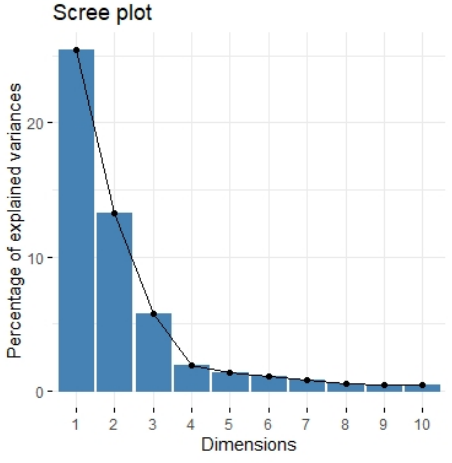
(a) Bytes distance



(b) Gaps distance

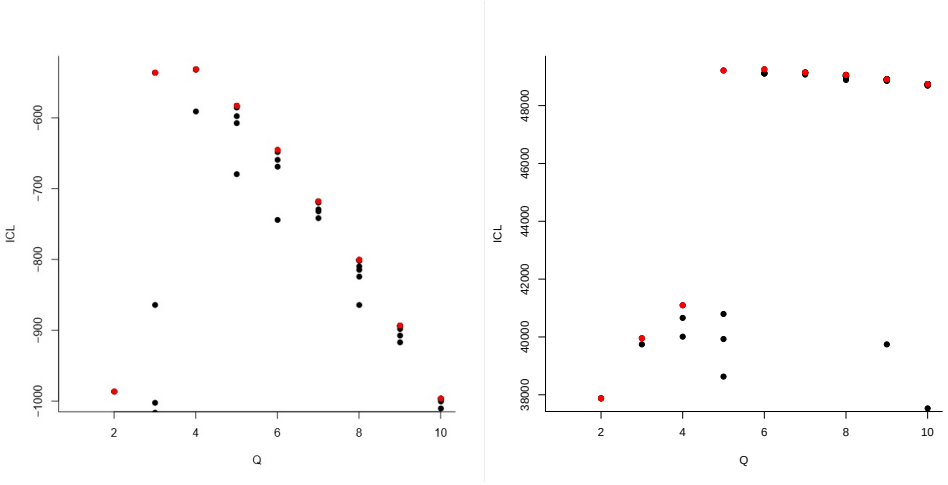


(c) Destination port distance



(d) Source port distance

Figure 5.2: CTU-91 data: Explained variance of components from the Principal Component Analysis on the four distance matrices, where the packet threshold was 10 packets. The connection distances in the bytes and gaps matrices were captured by one component approximately explaining 90% of the variance, whereas 3 components were required to capture > 40% of the variance in the destination and source port distances.



(a) ICL fit in the CTU-91 data.

(b) ICL fit in the ISOT data.

Figure 5.3: Plots of the ICL fit evaluation statistic in the ISOT data. The subtle peak at $Q = 4$ (ICL) and $Q = 5$ (ISOT) indicates that the optimal SBM clustering solution is reached at a 4- and 5-cluster solution, and model fit decays when Q increases.

5

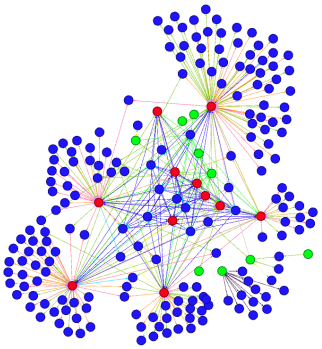
5.4.2. DENSITY-BASED AND AND LOUVAIN CLUSTERING

We applied dbscan and Louvain clustering to the CTU-91 data. The network plots with node colours are in Figure 5.4 and the node-classification versus ground truth in Table 5.5. Dbscan produced comparable results to our approach (MalPaCa and SBM), with the first and third cluster including one peripheral, one neutral, and half (5) of the infected nodes, the second cluster mostly peripheral nodes and neutral nodes. The fourth cluster with half (5) of the infected nodes and the fifth cluster with only peripheral nodes. Classification performance is lower but near to the MalPaCa + SBM procedure. Louvain clustering produced 6 clusters: all but one clusters have at least 1 infected node and only cluster 1, 4 and 6 have neutral nodes. The community structure looks very different from the other two clustering solutions, as we visibly identify clusters with highly connected nodes at the different hubs of the network and one cluster in the center.

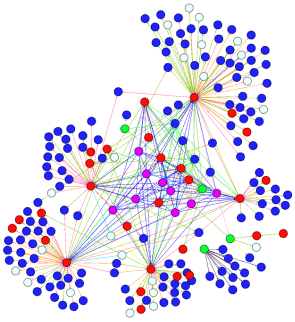
5.4.3. ISOT DATA

Previous studies have used the ISOT data for botnet identification purposes and Table 5.4 presents a selection of the performance reported in related works. As mentioned before, most of these methods require manual feature engineering and connection filtering to be applied, while others operate in a supervised setting. We compare our unsupervised clustering method to these results.

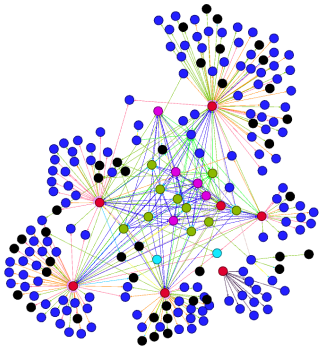
Creating the distance matrices with MalPaCA pruned the network (see Figure 5.5a) to 7683 surviving connections with $P_t = 20$. Average connection length was 365.95, with a minimum of 20 and a maximum of 525256. This amounted to 3847 nodes. There was one isolated sub-network of hosts connected to 172.16.2.3, of which only the connection between 172.16.2.3 and 193.88.8.59 survived the packet threshold of 20. Isolation sup-



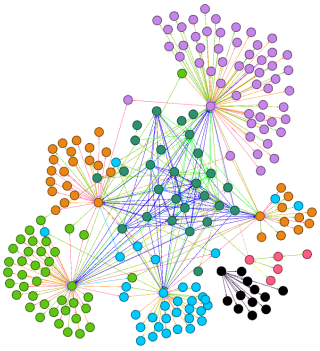
(a) Original labels



(b) Our method; MalPaCa + SBM



(c) Density-based clustering (dbscan)



(d) Louvain (modularity based) clustering

Figure 5.4: Network plots of a subset of the CTU-91 network (including hosts with a packet threshold $P_t = 10$) with nodes colour based on labels from different clustering methods. The subfigures show the network with the (a) original host labels, used in this analyses as ground truth (blue = peripheral, red = infected, green = neutral); (b) MalPaCA connection label colours and SBM host labels (blue = peripheral, red = infected, green & turquoise = neutral & peripheral); (c) labels from density based clustering resulting showing a 5 cluster solution (blue & black = peripheral & neutral, red & purple = mostly infected, turquoise = peripheral); (d) labels from Louvain clustering, which identified 6 highly-connected clusters. Clustering with (c) and (d) is only based on the network matrix, without covariates.

Table 5.5: Performance matrix of different clustering methods in the CTU-91 data

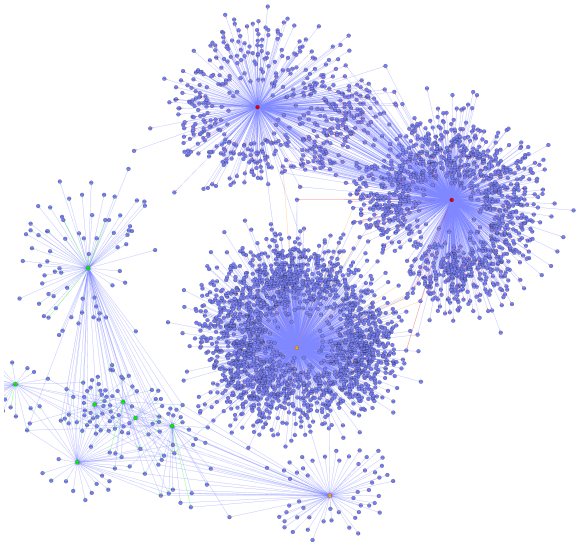
	Cluster	*peripheral*	*neutral*	*infected*
MalPaCa + SBM	1	-	-	10
	2	136	4	-
	3	9	-	-
	4	1	3	-
DBscan	1	1	1	5
	2	134	5	-
	3	9	1	-
	4	0	-	5
	5	2	-	-
Louvain	1	17	7	5
	2	47	-	1
	3	57	-	1
	4	145	1	2
	5	34	-	1
	6	5	2	-

ported their removal from subsequent clustering analyses, leaving 3845 nodes (running the analyses with these two nodes included yielded similar results in the optimal SBM solution; both were allocated to the cluster with infected nodes).

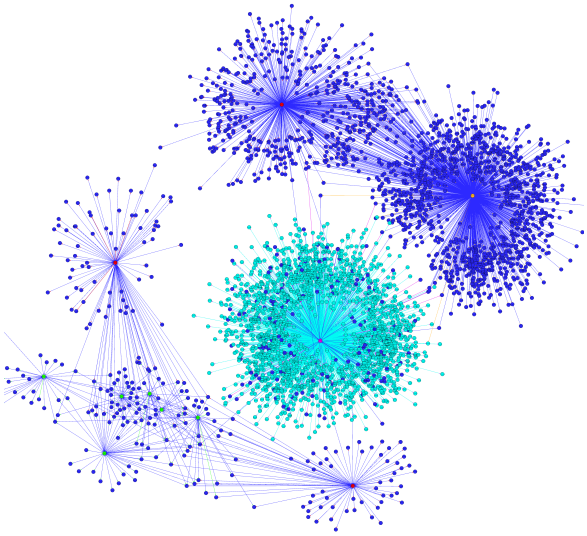
Identical to the Stratosphere data, a PCA fitting resulted 1, 1, 3, 3 components for respectively bytes, gaps, dport and sport to explain > 40% of the variation. The SBM model fitted on the binary adjacency matrix, with the PCA features resulted in an optimal 5 class solution (see Figure 5.5b and Table 5.6). Of these 5 clusters, clusters 1 and 2 captured the peripheral nodes, where the peripheral nodes in cluster 1 were all linked to host 172.16.2.11 (Storm + non-malicious) which was the only host allocated to cluster 3. Cluster 4 consisted of the Waledac and Storm hosts, confirming the comparability of Waledac and Storm activity. Cluster 5 captures eight hosts, of which seven are non-malicious: 172.16.2.2, 172.16.2.13-14, 172.16.2.111-114, and one host in cluster 5 (172.16.2.12) had combined (non-malicious & malicious) traffic. If we consider 1734 and 2100 peripheral nodes (cluster 1 and 2) and 7 non-malicious nodes (cluster 5) as true negatives, the Waledac and Storm nodes in cluster 3 and 4 as true positives, and the combined traffic node in cluster 5 as a false negative, the accuracy and sensitivity = 99.97 % and the specificity = 100%. This performance is similar to other work on supervised learning using decision trees [217, 225] and nearest neighbours [227] on manually curated collapsed data. We outperform the methods listed in [244].

5.4.4. SCALABILITY OF THE MODEL FITTING

In the previous section we fitted MalPaCa and SBM on the ISOT dataset with $P_t = 20$, resulting in 7683 surviving connections and 3847 nodes. Selecting a packet threshold of 20 was also a practical decision, because a major limitation of the model fitting procedure so far is the speed of Variational Inference when fitting a SBM with covariates to large datasets (> 2500 nodes). The runtime of our discovery (CTU-91) sample was about



(a) Network plots of a subset of the ISOT network for $P_t = 20$. Network with original host labels, used here as ground truth (blue = peripheral, red = malicious, orange = malicious + non-malicious, green = non-malicious.



(b) Network with labels assigned by our method: Turquoise (cluster 1) & blue (cluster 4) = peripheral, red (cluster 2) = malicious + missclasification, orange (cluster 3) and purple (cluster 5) = Waledac, and green (cluster 6) = non-malicious.

Figure 5.5: Network plot of a part of the ISOT data with original and recovered labels.

Table 5.6: Performance matrix from the SBM node-based clustering in the ISOT replication data

Cluster	*peripheral*	*neutral*	*neutral+infected*	*infected*
1	1734	-	-	-
2	2100	-	-	-
3	-	-	1	-
4	-	-	-	2
5	-	7	1	-

2.5 hours on a Windows 10 (i7-7700K CPU, 4.2 GHZ, 8-core, 16GB ram) machine. The much larger replication sample required the use of the High Performance Cluster from the TU Delft, with a runtime of approximately 5 days (multithreading with 8 cores on one node). We therefore sought a more scalable implementation of SBM, which we found in Stochastic Variational Inference (SVI).

Variational Inference has evolved from coordinate ascent Variational Inference (VI) to faster and more flexible versions such as Stochastic Variational Inference [245] and Black-Box Variational Inference [246, 247]. Since VI has local and global parameters, the estimation of local parameters does not require the full dataset. SVI makes use of this characteristic by allowing local parameter estimation for each individual observation, after which the estimates are combined via a weighted mean. This divide-and-update approach thus allows to split the data into mini-batches. SVI initially required using noisy estimates of the natural gradient of the ELBO [245], which are optimized via a Robbins-Monro algorithm. The use of noisy natural gradients has been surpassed by the less biased and computationally less demanding; smoothed gradients [248].

SBMs have benefited from these developments, recently presented as Stochastic Variational Inference in Mixed-Membership SBMs ([249]) on data from 3.7 million US patents, 575000 physics articles, and 875000 Web pages. This model has been implemented in an open source tool (SVINET: `https://github.com/premgopalan/svinet`) written in c++. We applied SVINET to our data, fitting a mixed membership SBM to the ISOT data. The speed of SVINET allowed to use the same packet threshold in the ISOT data as used in the CTU-91 discovery set (namely $P_t = 10$). With this less stringent threshold, the MalPaCa procedure filtered out 18325 connections between 9249 nodes. Applying SVINET in a Ubuntu linux virtual machine required a runtime of < 20 seconds to obtain a converging model, with only 16 iterations covering 92 mini batches. Model fitting resulting in a 12-cluster solution, this is higher compared to the single membership SBM, where nodes are not allowed to be part of multiple clusters. Most nodes ($n = 8742; 95\%$) were part of 1 cluster, 105 nodes were placed in two clusters, 9 nodes in 3 clusters, 4 nodes part of 4 clusters, 1 node was involved in 6 clusters, and 388 nodes were excluded. It is unclear whether the nodes in multiple clusters (the mixed members) are the infected machines; interpretation of the output is difficult since SVINET recodes the node labels which hinders merging and comparison, and changing SVINET is challenging as its code is uncommented and there is no documentation how to interpret output. SVINET does generate an .gml object which was used to plot the network in Gephi (see Figure 5.6).

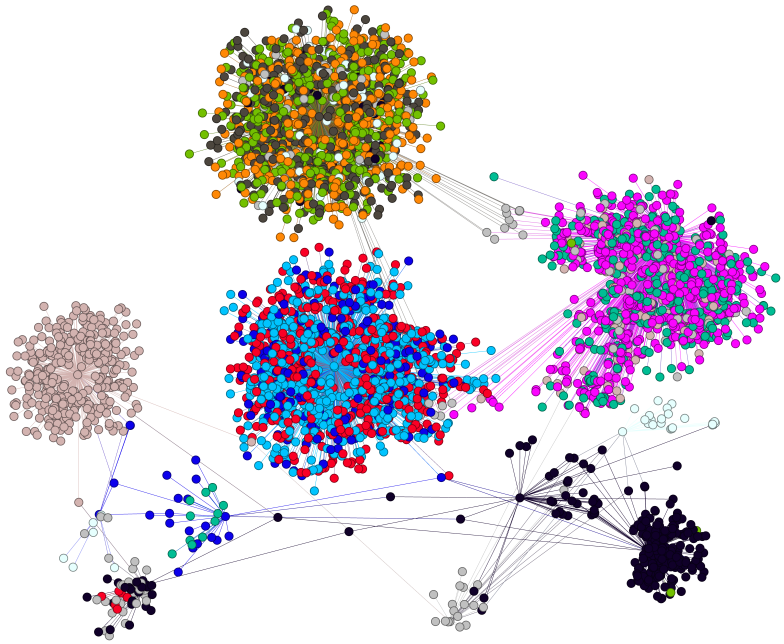


Figure 5.6: Network plots of a subset of the ISOT network for $P_t = 10$ (9249 nodes and 18325 connections) with labels from the mixed-membership SBM fitted in SVINET. The analyses removed 388 nodes as outliers, the colours indicate the different clusters for the remaining nodes. 8742 nodes (95%) are member of a single cluster, the remaining 119 (1.29%) nodes are in multiple clusters.

5.5. DISCUSSION

Here, we combined two unsupervised methods to solve the problem of analysing spatio-temporal data so that botnet infected computers can be identified via connection- and host clustering. In our discovery sample (CTU-91) we identified all infected machines and classification was perfect. The infected machines were all allocated to one cluster, indicating marked similarities between infected machines infected with the Conficker botnet. In the replication sample (ISOT), one host with malicious and non-malicious traffic was allocated to a cluster of non-malicious nodes, yielding one false negative with an overall accuracy of 99.97%. This procedure outperforms other botnet detection studies using the ISOT dataset [166, 217, 250, 189, 251] and has comparable performance to [227, 216]. Compared to the studies that report similar classification performance, our method does not require any type of filtering [216] or manual feature selection [227], and is therefore less sensitive to external factors. In the discovery sample, the neutral and peripheral hosts were allocated together in a cluster, whereas in the replication data, the peripheral hosts formed a separate cluster. This may be due to the mapping procedure used in the ISOT dataset, where botnet data were collected in a VM and mapped a posteriori, so that the differences in the ISOT data may be captured by our model, underlining

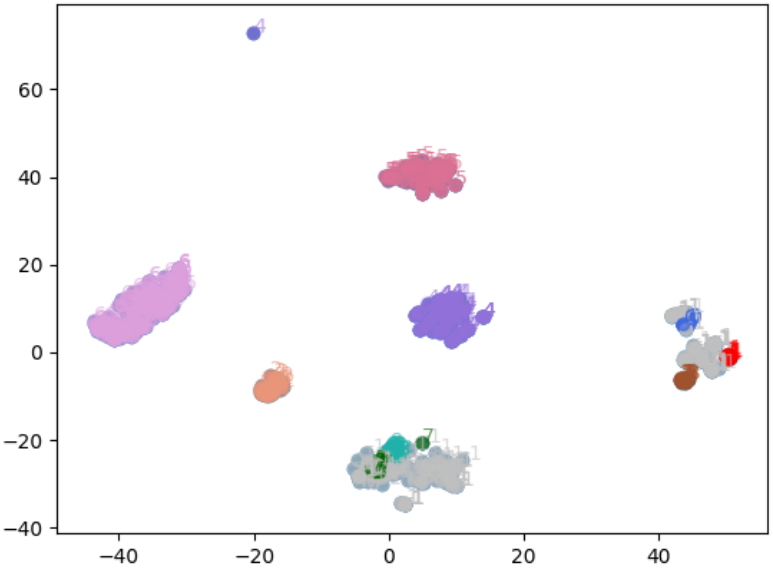
the sensitivity of our approach. Furthermore, although not explicitly illustrated, the output of MalPaCA has been found to be informative to identify malware families or other specifically tuned categories of traffic [213], and other similar connection profile based approaches exist [216].

A potential limitation of this study is the relatively short time window in which the data were collected. Ideally one would capture the temporal structure of the network traffic in more specific analyses. A prominent example of such analyses is creating snapshots [252], which facilitates network clustering within snapshots, so that state changes (nodes hopping to another cluster) between snapshots can be analysed [205]. However, given the length of the CTU-91 capture (roughly 20 minutes, compared to for example one year of data from mobile devices in [205]) we argue there is little sense in making 5-minute snapshots, since this would result in many, difficult to compare, local network clusters. Again, these packet thresholds are data specific, and shorter or other snapshots may be applicable in other types of network data (e.g. social network data where snapshots represent school-years). Although our approach does not require manual curation, understanding the effects of sample specific factors is a focus of future research.

5.5.1. SCALABLE MCMC

One year after the above-mentioned comparative study from Harenberg and colleagues the development of stochastic gradients MCMC (sg-MCMC), originating from Langevin Monte Carlo, outperformed Stochastic Variational Inference in Mixed-Membership SBMs [253] on 5 datasets (ranging from 75-5200 nodes). This was subsequently improved in 2016 with a more optimized parallel version of sg-MCMC [254]. Speeding up MCMC while retaining accuracy is topic of ongoing research and has reached the point where, in the SBM setting, the analyses of 100 million edges (equivalent to a complete adjacency matrix of 10000 nodes) is now achievable [99, 255].

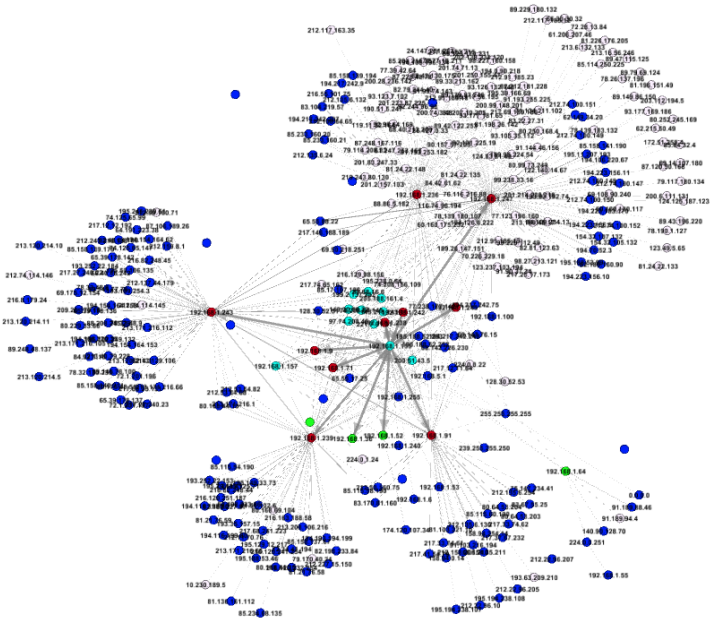
5.6. SUPPLEMENTARY MATERIAL



Supplementary Figure 5.1: This Figure shows the connections clustered with MalPaCA on the CTU-91 data. The grey dots indicate connections labeled as outliers by HDBScan. For this plot, the multidimensional sample space was reduced to two axes by TSNE, resulting in the ability to visually identify 7 clusters, of which the top cluster belongs to the middle cluster (letter 4), the right cluster decomposes into 3 sub-clusters (blue, red, brown) and outliers, and the bottom cluster consist of 2 sub-clusters (magenta, darkgreen) and outliers. Hence, 9 clusters are displayed.

5.6.1. HOST CLUSTERING CTU-91 DATASET

Node assignment to a cluster does not immediately inform which cluster(s) contain the infected nodes. Descriptive analyses are typically used to interpret the cluster output. For example, when comparing cluster 1 (10 hosts) with cluster 2 (140 hosts), we observed an almost 3-fold increase of packets send (93100 versus 33917), a higher occurrence of bigger packets send ($Mean_{c1} = 138.22 (SD = 180.51), Mean_{c2} = 118.97 (SD = 135.63), t = 1.9547, p = .051$) and received ($Mean_{c1} = 167.26 (SD = 226.31), Mean_{c2} = 142.92 (SD = 194.23), t = 1.6614, p = .09703$), and higher frequencies of HTTPS, UDP, and SMTP/IMF protocol traffic, whereas SMTP, TCP, NBNS, and BROWSER protocol traffic was significantly higher in cluster 2. This behaviour of nodes (more connections via specific protocols) is coherent for botnets. Further visualisation (not provided) resulted in the identification of cluster 1 as likely malicious (and verified with the original labels). All of the malicious hosts (192.168.1.238, 192.168.1.239, 192.168.1.236, 192.168.1.91, 192.168.1.71, 192.168.1.9, 192.168.1.243, 192.168.1.242, 192.168.1.247, 192.168.1.245) were assigned to

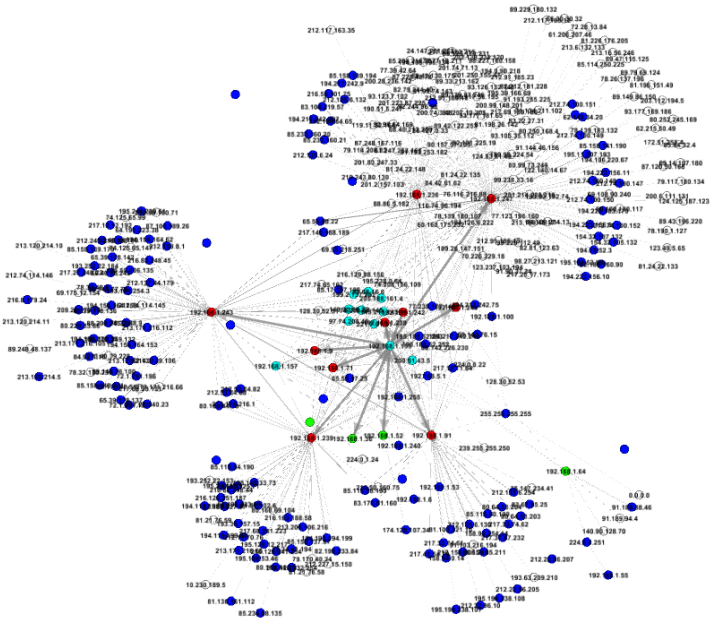


Supplementary Figure 5.2: This Figure shows the full network with the nodes coloured according to the labels from the optimal 4-class SBM solution. This plot is based on the analyses of 631 connections and 205 hosts (nodes) with packet threshold = 5. Nodes are coloured blue (neutral), green (neutral), turquoise (neutral), red (infected), or white (outliers).

one cluster with a posterior probability of > .998.

Our observation that mean differences between clusters (as exampled above) show a trend but are not significant, illustrates that just comparing mean differences to detect groups, with a straightforward anomaly detection approach, would be less successful in this particular setting.

Most of the peripheral hosts were captured by one cluster, indicating behavioural similarity, with a class assignment posterior probability of .9982. The non-infected / neutral hosts (192.168.1.155, 192.168.1.52, 192.168.1.157, 192.168.1.36, 192.168.1.6, 192.168.1.53, 192.168.1.64) were divided over two clusters, that also included peripheral hosts. Only one neutral host had a posterior probability < .95, which was host 192.168.1.6 with .82, with the remaining probability belonging to the other *neutral/mixed* class.



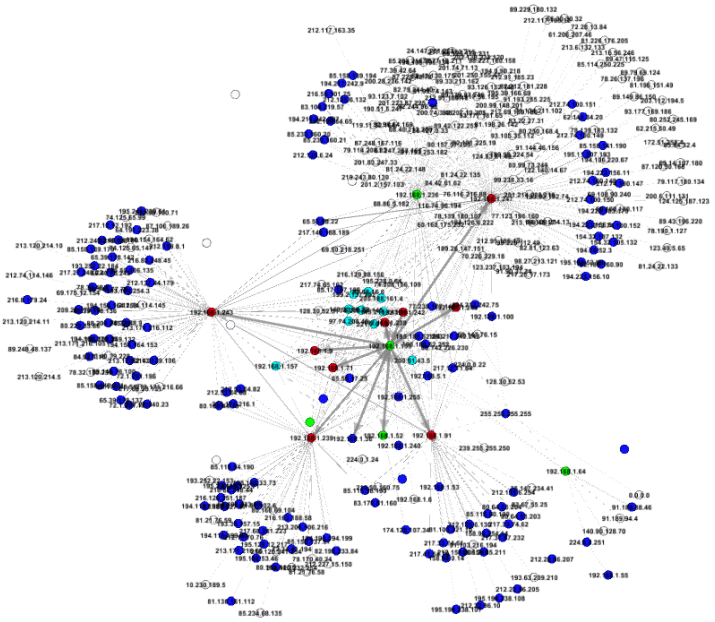
Supplementary Figure 5.3: This Figure shows the full network with the nodes coloured according to the labels from the optimal 4-class SBM solution. This plot is based on the analyses of 565 connections and 182 hosts (nodes) with packet threshold = 10. Nodes are coloured blue (neutral), green (neutral), turquoise (neutral), red (infected), or white (outliers).

Table S1: Correlation between distance matrices in the CTU-91 data

	bytes	gaps	dport	sport
bytes	-			
gaps	.04	-		
dport	.13	.09	-	
sport	.05	-.03	-.04	-

Table S2: Performance matrix from the SBM node-based clustering when packet threshold = 5

Cluster	*peripheral*	*neutral*	*infected*
1	9	0	0
2	0	0	10
3	1	4	0
4	158	3	0



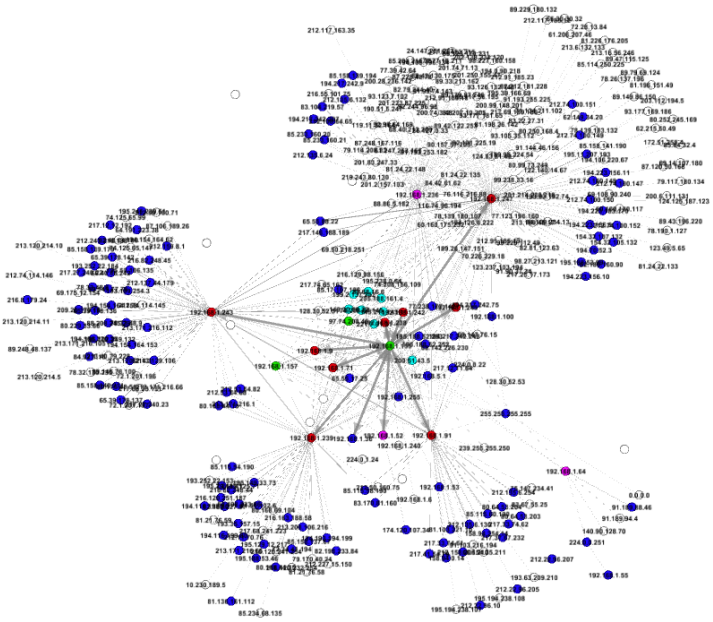
Supplementary Figure 5.4: This Figure shows the full network with the nodes coloured according to the labels from the optimal 4-class SBM solution. This plot is based on the analyses of 523 connections and 165 hosts (nodes) with packet threshold = 15. Nodes are coloured blue (neutral), green (neutral), turquoise (neutral), red (infected), or white (outliers).

Table S3: Performance matrix from the SBM node-based clustering when packet threshold = 15

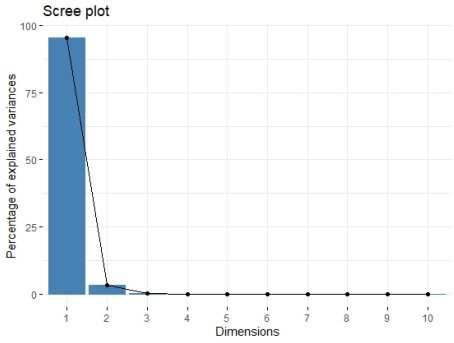
Cluster	*peripheral*	*neutral*	*infected*
1	133	4	0
2	3	1	10
3	0	1	0

Table S4: Performance matrix from the SBM node-based clustering when packet threshold = 20

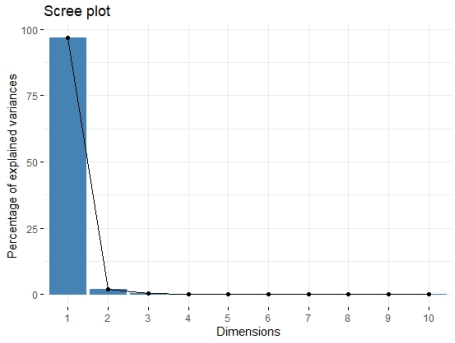
Cluster	*peripheral*	*neutral*	*infected*
1	123	5	0
2	0	0	6
3	0	0	4
4	2	1	0



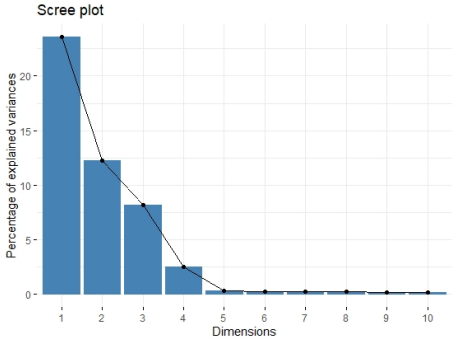
Supplementary Figure 5.5: This Figure shows the full network with the nodes coloured according to the labels from the optimal 4-class SBM solution. This plot is based on the analyses of 483 connections and 148 hosts (nodes) with packet threshold = 20. Nodes are coloured blue (neutral), green (neutral), turquoise (neutral), red (infected), or white (outliers).



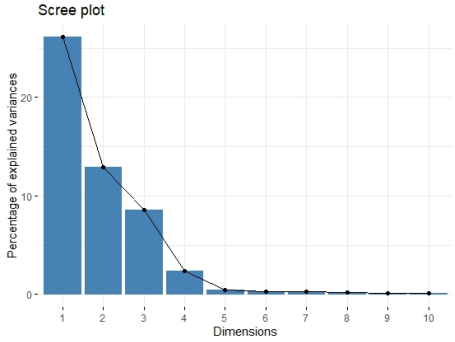
(a) Bytes distance



(b) Gaps distance

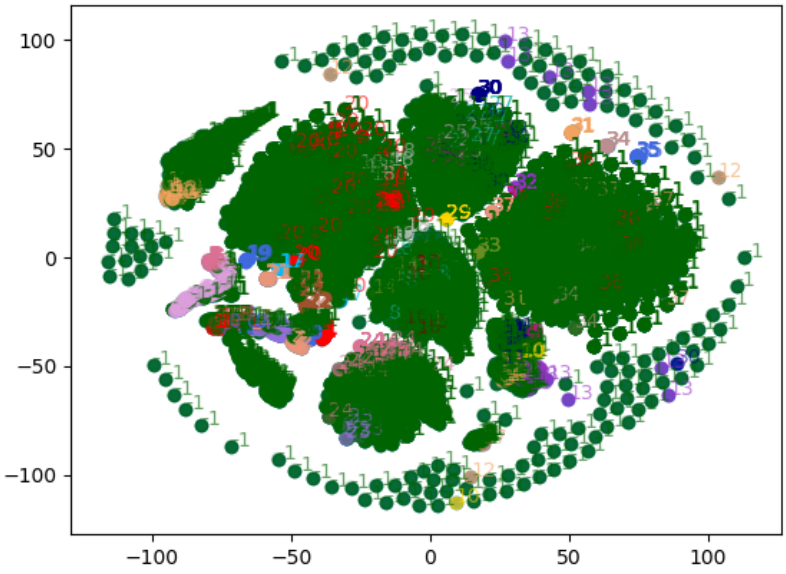


(c) Destination port distance



(d) Source port distance

Supplementary Figure 5.6: ISOT data: Explained variance of components from the Principal Component Analysis on the four distance matrices, where the packet threshold was 5 packets. The connection distances in the bytes and gaps matrices were captured by one component approximately explaining 90% of the variance, whereas 3 components were required to capture > 40% of the variance in the destination and source port distances.



Supplementary Figure 5.7: This Figure shows the connections clustered with MalPaCA on the ISOT data. The green dots indicate connections labeled as outliers by HDBScan. For this plot, the multidimensional sample space was reduced to two axes by TSNE. By colour we different clusters (e.g. orange and purple). Compared to the CTU-91 dataset we see the connections occupy a larger sample space, indicating more variance in the ISOT replication data.

Table S5: MalPaCA clusters and infection status in the ISOT data

Cluster	$srcip_n$	$srcip_i$	$dstip_n$	$dstip_i$
-1	1948	3415	1703	3216
1	0	0	9	10
2	12	12	0	0
3	21	0	0	0
4	0	0	24	0
5	22	0	0	0
6	0	0	20	6
7	0	0	90	17
8	92	10	0	0
9	0	0	0	10
10	0	16	0	0
11	0	9	0	0
12	0	43	0	0
13	0	48	0	0
14	0	38	0	0
15, 20 & 22	0	0	0	10
16	0	0	0	11
17	0	0	0	22
18 & 19	0	0	0	8
21	0	0	0	49
23	0	0	0	27
24	0	0	0	7
25	0	0	0	40
26 & 27	0	7	0	0
28	0	11	0	0
29	0	7	0	0
30	0	4	0	4
31	0	27	0	0
32	11	11	0	0
33, 34	8	8	0	0
35 & 36	11	11	0	0
37	8	8	0	0
38	15	15	0	0

Interpretation of rows and columns equal to Table 5.3. Clusters 1, 6, and 7 contain connections from peripheral hosts to neutral and infected hosts. Clusters 2, 8, 32-38 contain connections from both infected and neutral host to peripheral nodes. Clusters 3 and 5 both include connections from a neutral source ip to a peripheral nodes. Cluster 9 includes connections from peripheral nodes to infected destination hosts. Clusters 10-14, 26-29, and 31 comprise of connections from infected source hosts to peripheral hosts. Cluster 30 includes connections from infected source IPs to infected destination IPs.

II

DATA VERACITY

6

FALSE DATA INJECTION IN KALMAN FILTERS IN AN AEROSPACE SETTING

6.1. INTRODUCTION

Kalman Filters (KF) are recursive state estimation algorithms capable of combining and weighting different variables to estimate the real latent state of a system [64]. In this context, recursive reflects the property that not all previous data has to be kept in storage but every iteration incorporates information from previous observations and predictions [65]. This made the KF widely applicable resulting in its implementation across various settings, including aerospace, submarines, and the estimation of missile trajectories [256]. Given the importance and KFs across settings and systems there is growing interest from security researchers to understand the robustness of KFs under different adversarial models.

6.2. RELATED WORK

In the broader context of machine learning, taxonomies have been proposed to categorize attacks on learning algorithms [257]. According to those taxonomies, false data injection can be classified as a causative attack where the attackers aim to influence the learning process by affecting the training data. Several studies have addressed the effects of false data injection, mainly in the context of cyber physical systems such as power systems (see [67, 68]), network coordinate systems [258], and spam filters [259, 260]. Although estimating effective attack vectors for the measurement * state matrix is computationally intensive, brute force attacks are still feasible. However, one option is to increase the resilience of the system by relaxing the constraints on brute force attacks, and installing redundant measurement sensors [261]. Also, if the attacker knows the input data and the system, this could allow him to add a vector to the original measurement $z_a = z + x$ instead of true measurement z. Resulting in the attack vector to become a linear combination of the vectors of the (column vectors of the) measurement * state matrix, letting the L_2 norm of the measurement residual of z_a equal z, passing the detec-

tion [262]. Other suggestions that help preventing false data injection including schemes to protect measurements [263] and detect the attack [264, 265], as well as algorithms to select the optimal subset of measurements to protect (e.g. through encryption [266]).

A typical method to investigate the robustness of state estimation systems and the capacity of detection methods is the Frog Boiling method. This method words by gradually and episodically injecting data to attack the system, in order not to be detected. A study by Chan-Tin and colleagues [258] showed that in network coordinate systems, the frog boiling attack was just as effective as a random attack, leading to the assumption that KFs will not be effective outlier detectors. This assumption was tested by Mo and Sinopoli in 2010 [261] to provide proof that the KF estimates could indeed be destabilized with false data injection, despite several failure detectors.

False data injection in KFs has been studied in the context of SCADA systems [67] and secure estimation methods on simulated UAV data [69]. Yang and colleagues [67] investigated the robustness of state estimates by evaluating an innovation factor in five attack models; maximum magnitude-based, wave-based, positive deviation, negative deviation, and mixed. In the maximum magnitude-based attack the adversary tries to achieve the maximum deviation of original measurements that equals to the maximum magnitude of the attack vector. In the wave-based attack, the malicious measurements are the reverse direction of injected attack data. In the positive and negative deviation attack, the adversary tends to achieve the maximum deviation of original measurements along with the direction of increase. Finally, the mixed attack can be a combination of the latter four attack models in consecutive time points (e.g. positive deviation at $t+1$, wave-based at $t+2$). Chang et al. [69] combined the KF with secure estimation and showed that applying the KF after data were run through a secure estimation algorithm yielded more secure output than applying the algorithm or filter alone. Finally, one of the reasons why the KF itself was not robust against attacks was that the manipulated data violated the KF assumption of Gaussian distributed noise. This observation adds to the overall impression that KFs are inherently insecure and vulnerable to data manipulation, since KF produces reasonable estimates even when assumptions are violated making it difficult to detect malicious input.

Aircraft position estimation has historically relied on the availability and interpretation of radar data. Recently, a new system has been developed called Original Automatic Dependent Surveillance - Broadcast (ADS-B) that replaced primary and secondary surveillance radar technologies in 2017. ADS-B is based on the Global Navigation Satellite System (GNSS) and relies on on-board navigation systems that retrieve GPS data, determine the aircraft position, and forward these data to ground stations[1]. Researchers [70] and hackers [267] have already identified several vulnerabilities in the ADS-B infrastructure. The main problem is the absence of encryption of ADS-B message content, resulting in the possibility that adversaries can eavesdrop on messages sent out by aircraft. Other vulnerabilities include the injection of ADS-B messages to create ghost aircraft, jam the signal to make aircraft disappear, or replace aircraft by replacing the identifier of the ADS-B message with modified data. While the technical details of the on-board aircraft position estimation in the ADS-B infrastructure are difficult to come by, integration and combination of raw satellite data to derive an accurate GPS position

[1] https://en.wikipedia.org/wiki/Automatic_dependent_surveillance_-_broadcast

is likely based on the Kalman Filter. Also Kalman Filters can be used for the combination of ADS-B with radar data [268]. Given the outlined vulnerabilities of both the Kalman Filter algorithm and the ADS-B infrastructure, this study aims to investigate the sensitivity of the Kalman Filter to the effects of false data injection under different adversarial scenario's, by replicating the study of [67] in the ADS-B context.

6.3. METHODS

6.3.1. OPENSKY ADS-B DATA

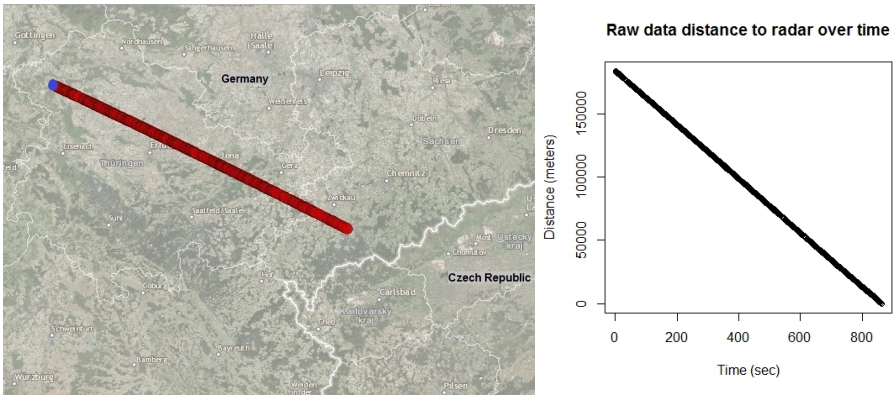
Original Automatic Dependent Surveillance - Broadcast (ADS-B) data were obtained through the OpenSky platform [269]. In short, OpenSky consists of various off-the-shelf sensors distributed over Europe capturing more than 40% of Europe's commercial air traffic. Aircraft use on-board satellite navigation systems (e.g. GPS) to retrieve their own position and velocity, which is broadcasted twice per second to ATC stations on the ground and other aircraft. Exploration of the data from a subset of flights revealed that the data were already filtered upon collection as most commercial GPS systems have built in Kalman Filtering [270]. Filtering ADS-B output again is unlikely to be beneficial, because it violates the time-independence assumption of KFs: GPS data are time dependent as the filter bases its current estimate on the recursive estimates of all previous measurements. For this project, a representative subset of data from flight OHY925 from Antalya (Turkey) to Amsterdam (The Netherlands; see Figure 1) were used.

Figure 1: Flightpath of flight OHY925 based on the ADS-B GPS data

6.3.2. LINEAR UNIDIMENSIONAL MODEL

TRANSFORMATION OF ADS-B TO RADAR DATA

Latitude and longitude were included in the ADS-D data. Latitude of a point on the Earth's surface is the angle between the equatorial plane and the straight line that passes through that point and through (or close to) the centre of the Earth. Longitude of a point on the Earth's surface is the angle east or west from a reference meridian to another meridian that passes through that point[2]. For the linear model, I decided to transform the GPS data to position data, allowing simulation of radar distance signals. For every timestep in the ADS-B data, I calculated the great-circle distance, which is the shortest distance between two points on the surface of a sphere, with the Vincenty method [271] as implemented in the R package Geopshere [272]. Because the flight occurred above central Europe, reference measures were used from the European Terrestrial Reference System 1989 (ETRS89). ETRS89 is an earth-centered, earth-fixed geodetic Cartesian reference frame, in which the Eurasian Plate as a whole is static. The equatorial axis of ellipsoid is 6378137, the polar axis of ellipsoid is 6356752.31414, and the inverse flattening of ellipsoid = 298.257222101.



(a) Linear plot mimicking typical radar distance estimates

(b) Linear distance × time relationship

Figure 2: Subpart of the flightpath without a deviation in direction

The standard Kalman Filter can only be applied to linear states. To simulate a linear radar system, a subset of the first 500 ADS-B coordinates was selected that consisted of the flightpath between east Aue (Germany) and Flinsberg (Germany; Figure 2a). The last point of the ADS-B flight data was set as radar station (longitude = 10.24911, latitude = 51.31821). With the Vincenty method, the metric distance was calculated for every ABS-B data point and the virtual radar station. The largest distance (beginning of the flightpath) to the radar station was 184092.35 meters (m), the smallest distance (end of the flightpath) to the radar station was 0.28 m. Normally, ADS-B is forwarded twice a second, but the distance between forwards revealed that, assuming a constant velocity in flight, the forwarding occurred with large instability. Also, 46.8% of the forwards contained identical GPS information to the previous forward. Given that ADS-B data is

[2]Wikipedia, Geographic Coordinate System https://en.wikipedia.org/wiki/Geographic_coordinate_system

Table 1: Parameter definition

$\bar{u}_t$ = Predicted state estimation
A_t = Matrix of n*n that describes how the state evolves from t-1 to t
B_t = Matrix of n*l that describes how the control u_t changes the state
$\overline{\Sigma}_t$ = Predicted process covariance matrix
Q_t = Process noise covariance matrix
K_t = Kalman gain
C_t = m-dimensional measurement matrix
R_t = Measurement noise
u_t = Current state estimation
y_t = real noisy measurement
y_{noise} = observation errors in mechanism (eg. electronic delays)
z_t = Imported measurement
Σ_t = Updated process covariance matrix
I = Identity Matrix
w = Gaussian white noise

forwarded from the aircraft, filtering of the raw satellite data has already been conducted and ADS-B data is very smooth. This supports the decision to simulate additional error. Noise in radar systems can vary from 5 to 300 m. Given the speed of the aircraft in flight (250 m/s at 900 km per hour) noise was simulated by creating a random normal distribution (N = 100.000) with mean zero and a standard deviation of 250 m. At every iteration of the model, one sample was independently drawn from this distribution and added to the raw radar distance..

If the aircraft has a constant velocity (which is to be expected in this part of the flight), the variance around the distances between updates should be minimal. To verify this, I calculated how much steps were identical between every step in GPS coordinates and divided the metric distance between GPS coordinates by the amount of identical steps. As a result, the differences will average out and every update includes a distance that is standardized for time (one second), yielding a linear model in which the aircraft approached the virtual radar every timestep (Figure 2b). Verification showed substantial variability of the distance between updates, following a normal distribution with mean -213.56 m/s and standard deviation (SD) 101.47 m. These distances were used as the velocity of meters/second to model acceleration in the dynamical model. The noise simulation was identical to the linear model.

KALMAN FILTER MODEL

The Kalman Filter for the linear model followed the structure as outlined in Welch & Bischop (2006) with parameter definition in Table 1, with the prediction step defined as

$$\bar{u}_t = A_t \bar{u}_{t-1} + B_t u_{t-1} + w_t \tag{6.1a}$$

$$z_t = C_t y_t + y_{noise} \tag{6.1b}$$

$$\overline{\Sigma} = A_t \Sigma_{t-1} A_t^T + Q_t \tag{6.2}$$

And the filtering step defined as:

$$K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + (R_t)^{-1} \tag{6.3}$$

$$u_t = \bar{u}_t + K_t[z_t - C_t \bar{u}_t] \tag{6.4}$$

$$\Sigma_t = (I - K_t C_t)\overline{\Sigma}_t \tag{6.5}$$

The dynamical model, or state transition function, followed a standard radar-aircraft model with the innovation function $\left(\begin{smallmatrix}1 & \triangle T\\ 0 & 1\end{smallmatrix}\right)$ with $\triangle T$ being the timestep (1 second) and the state change estimator $\left(\begin{smallmatrix}\frac{1}{2}\triangle T^2\\ \triangle T\end{smallmatrix}\right)$ $[acceleration]$ [273]. Labbe [270] presents the explanation for the innovation function, with state space matrix $\left(\begin{smallmatrix}0 & 1\\ 0 & 0\end{smallmatrix}\right)\left(\begin{smallmatrix}x\\ \dot{x}\end{smallmatrix}\right)$ where x is the position and $\dot{x}$ is the velocity. The F matrix is $\left(\begin{smallmatrix}0 & 1\\ 0 & 0\end{smallmatrix}\right)$ and the following Taylor series expansion linearises the equation at t:

$$\Phi(t) = e^{F_t} = I + F_t + \frac{(F_t)^2}{2!} + \frac{(F_t)^3}{3!} + ... + \frac{(F_t)^n}{n!} \tag{6.6}$$

resulting in $F^2 = \left(\begin{smallmatrix}0 & 0\\ 0 & 0\end{smallmatrix}\right)$, so with all higher powers of F equal 0: $\Phi(t) = I + F_t + 0 = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right) + \left(\begin{smallmatrix}0 & 0\\ 1 & 0\end{smallmatrix}\right) t = \left(\begin{smallmatrix}1 & t\\ 1 & 0\end{smallmatrix}\right)$. Acceleration was defined as the velocity for every $\triangle T$, which could increase and decrease and the value in the model was updated with the velocity. Starting values of the model were taken from the data, position equal to the largest distance from the aircraft to radar (184092.35 m) and velocity equal to the mean of the velocity in the data (-213.56 m/s). Other starting parameters were uncertainty in measurement (500 m), process covariance matrix (position 1km, velocity 300 m/s), and observation error (position 1km, velocity 250 m/s).

There are different methods to formulate the process covariance matrix [274] varying from a face value definition to the Autocovariance Least-Squares technique [275]. I decided to use a process noise covariance model that assumes that the acceleration is constant for the duration of each time period (in line with the standardization), but differs for each time period (in line with the variance in distance), and each of these are uncorrelated between time periods (time independence), outlined in [270], shortly defined as $Q_t = \left(\begin{smallmatrix}\frac{\triangle T^4}{4} & \frac{\triangle T^3}{2}\\ \frac{\triangle T^3}{2} & \triangle T^2\end{smallmatrix}\right)$ $[\sigma_v^2]$ where $[\sigma_v^2]$ is the variance of the velocity.

6.3.3. NON-LINEAR MULTIDIMENSIONAL MODEL

ADDING NOISE TO ADS-B DATA

For the non-linear model, a subset of the ADS-B GPS data with altitude was selected that consisted of the flightpath between east Flinsberg (Germany) and Schiphol (the Netherlands, Figure 3). Given that ADS-B data are already filtered, noise was added to the data (Figure 4). First, for every GPS coordinate, the longitude estimate was incremented with 0.000001, while keeping latitude identical. The Vincenty Method was used to calculate the pairwise distance on every iteration, which stopped if the distance reached 100 meters. Typically, satellite estimates are accurate, especially within Europe where continental drift effects are minimal. Given the speed of the aircraft (900 km/h = 250 m/s) and the observation that ADS-B normally forwards twice a second, I used 100 meters as noise threshold. Then, the same procedure was repeated, decreasing the longitude until the

distance again reached 100 m. Finally, this iterative process was repeated for latitudes in both directions, while keeping the longitude identical. This resulted in a range for longitude and a range for latitude (Figure 5), that was used to create a list of all possible intermediate GPS values (resolution = 0.00001 ≈ 1.9 km) from which a random GPS coordinate was sampled. This noisy data was used for all subsequent analyses.

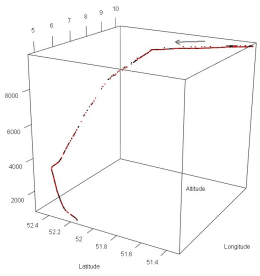


Figure 3: Flightpath (black) and noise (red) starting at the upper right corner. Axes are Longitude, Latitude, and Altitude.

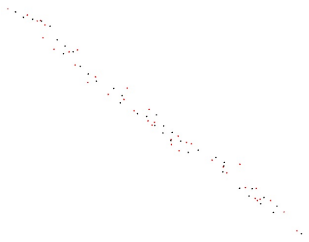


Figure 4: Zoomed-in part of the flightpath, raw data in black, simulated noise in red.

Figure 5: One GPS position (middle) with four 100 m deviations (up, down, left, right) and the yellow noise window.

Figure 3 shows the flightpath following a non-linear model. As the aircraft slowed down in the descent towards the airport, the mean of the velocity (-122.424 m/s) and with a larger variability (SD = 371.073 m) than the linear flightpath (SD = 101.47 m). These distances were used as the velocity of meters per second.

EXTENDED KALMAN FILTER MODEL

The Extended Kalman Filter (EKF) is the most commonly used state estimation algorithm for non-linear processes [276]. Following [67] I used the computation method described from [277], which is largely identical to the non linear KF as it still is defined as a linear model but uses local linearisation to approximate the slope at the point of measurement. This local linearisation occurs in the estimation of the dynamical model, so that the estimated state is system function (f) that takes three parameters $\bar{u}_t = f(\bar{u}_{t-1}, u_{t-1}, w_t)$ with output function $z_t = h(\bar{u}_t, v_k)$, where $\bar{u}_t$ and z_k are the state variable vector and measurable output at time t, respectively. Parameter u_t is the measurable input, w_t is the process noise (White Gaussian), and v_k is the measurement noise (White Gaussian). Calculation of the Jacobians was conducted with the package numDeriv [278].

The dynamical model that has to be linearised was identical to the model used in the linear model but extended to six variables.

Giving the following innovation function $\begin{pmatrix} 1&0&0&\triangle T&0&0\\ 0&1&0&0&\triangle T&0\\ 0&0&1&0&0&\triangle T\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{pmatrix} \begin{pmatrix} longitude\\ latitude\\ altitude\\ velocity_{longitude}\\ velocity_{latitude}\\ velocity_{altitude} \end{pmatrix}$

and state change estimator $\begin{pmatrix} \frac{1}{2}\triangle T^2&0&0\\ 0&\frac{1}{2}\triangle T^2&0\\ 0&0&\frac{1}{2}\triangle T^2\\ \triangle T&0&0\\ 0&\triangle T&0\\ 0&0&\triangle T \end{pmatrix} \begin{pmatrix} accelaration_{longitude}\\ accelaration_{latitude}\\ accelaration_{altitude} \end{pmatrix}$.

The EKF formulation is as follows:
The prediction step:

$$\bar{u}_t = f(\bar{u}_{t-1}, u_{t-1}, w_t) \tag{6.7a}$$
$$z_t = h(y_t, u_{t-1}, y_{noise}) \tag{6.7b}$$

$$\overline{\Sigma} = A_t \Sigma_{t-1} A_t^T + L_{t-1} Q_t L_{t-1}^T \tag{6.8}$$

Where A_t = and L_{t-1} = and the filtering step is defined as:

$$K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + M_t R_t M_t^T)^{-1} \tag{6.9}$$

$$u_t = \bar{u}_t + K_t[z_t - h(\bar{u}_t, 0)] \tag{6.10}$$

$$\Sigma_t = (I - K_t C_t)\overline{\Sigma}_t \tag{6.11}$$

Where $C_t = \frac{\partial h}{\partial x}(\bar{u}_t, 0)$ and $M_t = \frac{\partial h}{\partial v}(\bar{u}_t, 0)$.

Starting parameters were uncertainty in measurement (1 km, equal to a bidirectional deviation of 0.0015 longitude and 0.0010 latitude), process covariance matrix (500 m with start error of 300 m/s), and observation error (position 1 km, velocity 250 m/s).

6.3.4. ATTACK MODELS

ANOMALY DETECTION IN KALMAN FILTER

This method follows [67]. After each prediction step the innovation factor (v_t) is calculated, which is equal to the difference between the prediction and the actual measurement:

$$v_t = z_t - C_t \bar{u}_t \tag{6.12}$$

With z_t being the original measurement and $C_t \bar{u}_t$ being the predicted state. The innovation factor can be approximated by a white Gaussian process. To enhance interpretation of the innovation factor, it is standardized:

$$\lambda_t = v_t / \rho_t \tag{6.13}$$

$$\rho_t = \sqrt{(C_t \overline{\Sigma}_t C_t^T + R_t)} \tag{6.14}$$

Where C_t is a m-dimensional measurement matrix, $\overline{\Sigma}_t$ is the updated process covariance matrix, and R_t is the measurement noise. A detailed description of the steps involved in anomaly detection is published elsewhere [67]. In short, anomalies are detected by comparing the absolute value of the standardized innovation factor $|\lambda_t|$ against a predefined threshold λ_{max}. Given the two tailed distribution of the standardized innovation factor.

Sophisticated data attacks use an effective non-zero attack vector c_t, in the anomaly detection algorithm: $\frac{z_t - C_t \bar{u}_t}{\rho_t} \leq \lambda_{max}$ so that the range of z_t van be obtained by

$$C_t \bar{u}_t + \lambda_{max} \rho_t \geq z_t \geq C_t \bar{u}_t - \lambda_{max} \rho_t \tag{6.15}$$

In other words, the malicious measurement z_t should be a value that is derived from the boundaries depending on the measured state, the predicted state, and the (standardized) innovation factor threshold, in order not to be detected by the anomaly detection

threshold. The attack vector c_t can be obtained by subtracting y_t (observed noisy estimate) from z_t (predicted measurement). I assume the attacker knows the anomaly detection algorithm and the predefined threshold λ_{max}. Other parameters ρ_t and $C_t\bar{u}_t$ can be derived between $t-1$ and t. Since the state prediction is conducted at the very beginning of the KF procedure, and adopts the value of the previous state estimation after the first iteration, z_t can be derived as soon as the previous iteration is completed, which is before t.

MAXIMUM MAGNITUDE-BASED ATTACK

In this attack, the adversary tries to achieve the maximum deviation of the original measure. That is, the maximum deviation that is allowed within the anomaly detection threshold, by estimating the maximum attack vector $|c_t|$ that achieves the maximum manipulation of the received measurement z_t from the original measurement y_t by inserting false data. The adversary acquires the parameters at time $t-1$, computes the predicted measurement h_i, λ_{max}, and ρ_t. For the next timestep $(t+1)$, the original measurement y_t is retrieved and the innovation vector v_t is calculated. Depending on the evaluation of the innovation vector, $\lambda_{max}\rho_t$ is added to $(v_t < 0)$ or subtracted from $(v_t \geq 0)$ h_i. This attack can be expressed as follows:

$$\text{if } v_t \geq 0 : h(\bar{u}_t, 0) - \lambda_{max}\rho_t \tag{6.16a}$$
$$\text{if } v_t < 0 : h(\bar{u}_t, 0) + \lambda_{max}\rho_t \tag{6.16b}$$

With attack vector c_t:

$$\text{if } v_t \geq 0 : c_t = z_t - y_t = -v_t - \lambda_{max}\rho_t \tag{6.17a}$$
$$\text{if } v_t < 0 : c_t = z_t - y_t = -v_t + \lambda_{max}\rho_t \tag{6.17b}$$

Giving:

$$|c_t| = |z_t - y_t| = |v_t| + \lambda_{max}\rho_t \tag{6.18}$$

WAVE-BASED ATTACK

This attack is computationally identical to the maximum magnitude-based attack, but the injected attack data will be in the opposite direction of the estimated state. This translated into the formulas below, with opposite conditions on v_t:

$$\text{if } v_t < 0 : h(\bar{u}_t, 0) - \lambda_{max}\rho_t \tag{6.19a}$$
$$\text{if } v_t \geq 0 : h(\bar{u}_t, 0) + \lambda_{max}\rho_t \tag{6.19b}$$

With attack vector c_t:

$$\text{if } v_t < 0 : c_t = z_t - y_t = -v_t - \lambda_{max}\rho_t \tag{6.20a}$$
$$\text{if } v_t \geq 0 : c_t = z_t - y_t = -v_t + \lambda_{max}\rho_t \tag{6.20b}$$

Giving:

$$|c_t| = |z_t - y_t| = \lambda_{max}\rho_t - |v_t| \tag{6.21}$$

POSITIVE DEVIATION ATTACK

In aim of this attack is to achieve the maximum deviation (maximum value of z_k) of original measurements along with the direction of increase, independent of the direction of the innovation factor. This attack can be formulated as:

$$z_k = h(\bar{u}_t, 0) + \lambda_{max}\rho_t \tag{6.22}$$

With attack vector c_t:

$$c_t = z_t - y_t = -v_t + \lambda_{max}\rho_t \tag{6.23}$$

Giving:

$$|c_t| = |z_t - y_t| = \lambda_{max}\rho_t - v_t \tag{6.24}$$

NEGATIVE DEVIATION ATTACK

The negative deviation attack is identical to the positive deviation attack, but here, z_t is always the minimum of the range of its possible value:

$$z_k = h(\bar{u}_t, 0) - \lambda_{max}\rho_t \tag{6.25}$$

With attack vector c_t:

$$c_t = z_t - y_t = -v_t - \lambda_{max}\rho_t \tag{6.26}$$

Giving:

$$|c_t| = |z_t - y_t| = \lambda_{max}\rho_t + v_t \tag{6.27}$$

6.3.5. STATE DEVIATION UNDER ATTACK

LINEAR MODEL

This section explains how and where in the KF procedure, data are injected. This study assumed that attacks had full knowledge about the system, the incoming data, the anomaly detection algorithm, and the implementation of the state estimation algorithm(s). The KF is defined below:

The linear model:

$$\bar{u}_t = A_t\bar{u}_{t-1} + B_t u_{t-1} + w_t \tag{6.28}$$

$$\overline{\Sigma} = A_t\Sigma_{t-1}A_t^T + Q_t \tag{6.29}$$

And the filtering step defined as:

$$K_t = \overline{\Sigma}_t C_t^T (C_t\overline{\Sigma}_t C_t^T + (R_t)^{-1} \tag{6.30}$$

$$u_t = \bar{u}_t + K_t[z_t - C_t\bar{u}_t] \tag{6.31}$$

$$\Sigma_t = (I - K_t C_t)\overline{\Sigma}_t \tag{6.32}$$

The attack vector was defined previously as $c_t = z_t - y_t$, and the attack vector errors are obtained by multiplying them with the Kalman Gain:

$$a_t = K_t c_t \tag{6.33}$$

Typically, the attack vector and its errors are matrices $m * 1$ matrices, with m being the number of variables or dimensions in the model, necessitating the definition of an attack

parameter c_t for every dimension (the errors a_t can be derived from the Kalman gain and attack vector). These attack parameters are added to the state estimation model parameter $\bar{u}_t$. Hence, both the observed measurement y_t and the predicted state $\bar{u}_t$ are respectively manipulated by the attack vector c_t and a_t:

$$z_{t+1} = y_{t+1} + c_{t+1} \tag{6.34a}$$

$$\bar{u}_t^+ = \bar{u}_t + a_t \tag{6.34b}$$

The attack vectors c_t and a_t are injected in the state estimation formula, resulting in a manipulated state estimation:

$$u_{t+1}^+ = [A_t(\bar{u}_{t-1} + a_t) + B_t u_{t-1} + w_t] + K_{t+1}[(y_{t+1} + c_t) - C_t[A_t(\bar{u}_{t-1} + a_t) + B_t u_{t-1} + w_t]] \tag{6.35}$$

Equal to:

$$u_{t+1}^+ = \bar{u}_t^+ K_{t+1}[z_{t+1} - C_t \bar{u}_t^+] \tag{6.36}$$

Where u_t^+ is the state estimation after the attack. Since the state estimation is defined or acknowledged to be the moment t, the attack occurs between $t-1$ and t.

NON-LINEAR MODEL

Data injection is largely identical in the non-linear model, with a_t identical to formula 33. Consequently, the state estimation model is as follows:

$$u_{t+1}^+ = f(\bar{u}_t^+, 0) + K_{t+1}[z_{t+1} - h(f(u_t^+, 0), 0)] \tag{6.37}$$

Where z_{t+1} is the received measure at $t+1$ and $z_{t+1} = y_{t+1} + c_{t+1}$.

6.3.6. EVALUATING KF PERFORMANCE

There are numerous ways to investigate the performance of the KF. In general, relying on visual inspection of the plotted KF estimation can be intuitive but is not always valid. Labbe [270] describes how to use check the KF residuals and compare these residuals against 95% confidence intervals. To understand the accuracy of the parameters in the KF model, there is a widely used performance index (J_t [67] that evaluates the ratio of [estimated measurement - true vector of measurements] versus [real (noisy) measurement - true vector of measurement]. Ideally, the ratio approaches unity, as an indication of optimal performance:

$$J_t = \frac{\Sigma|u_t - \bar{u}_t|}{\Sigma|y_t - \bar{u}_t|} \tag{6.38}$$

6.4. RESULTS

6.4.1. LINEAR MODEL

The normal (not attacked) linear model is visually presented in Figure 6, which shows a very good fit of the KF to the data.

Evidence for good fit of the data can also be inferred from the residuals, where 94.76% of the position estimates (Figure 7a) and 94.64% of the velocity estimates (Figure 7b) fall within the 95% confidence interval.

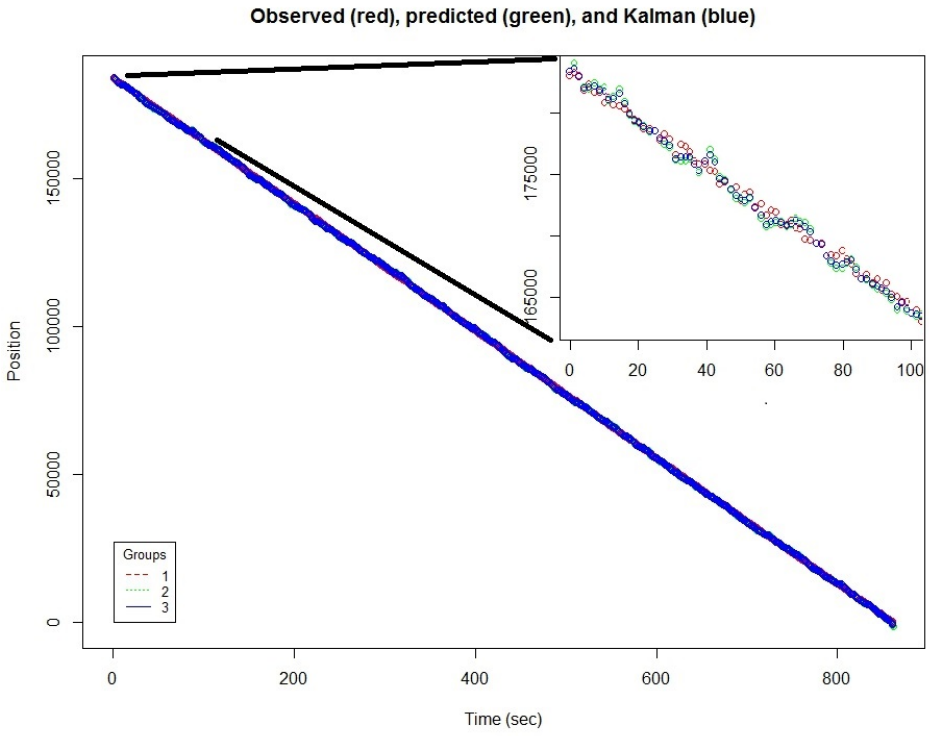
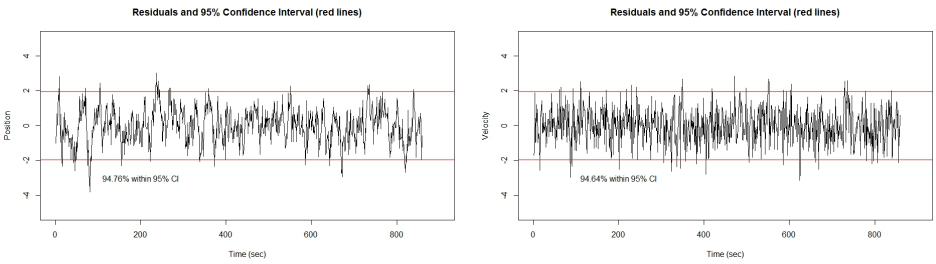


Figure 6: Linear model noisy data (red), the predicted state (green) and the current state Kalman Filter (blue).

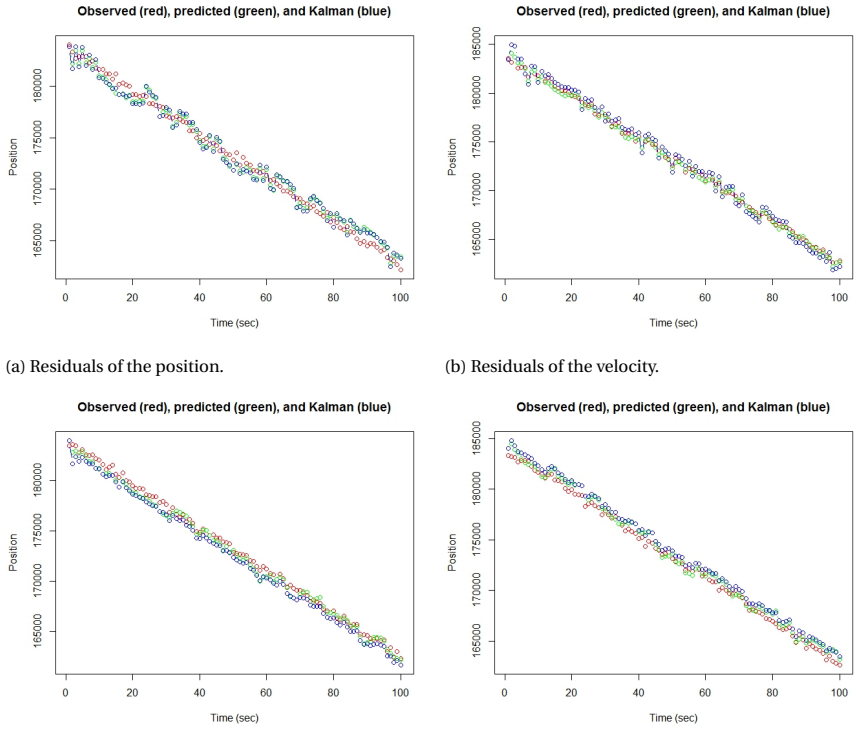


(a) Residuals of the position.

(b) Residuals of the velocity.

Figure 7: Residuals of linear model on position and velocity

Figure 8 presents (a part of) the trajectory plots for the four attack models. The maximum magnitude based attack shows considerable deviation from the trajectory, especially when compared to the wave based (opposite direction of the prediction), where deviation of the predicted state and current state (Kalman) look consistently smaller, owing to the fact that they are in the opposite direction, and therefore could level out strong deviations in the real (noisy) state. The positive and negative deviation attacks show similar patterns, with the positive deviation yielding a uniform deviation to lower positions; pulling the Kalman estimate of position closer to the radar. In contrast, the negative deviation attack provides a persistent overestimation of the position of the aircraft.



(a) Residuals of the position.

(b) Residuals of the velocity.

(c) Residuals of the position.

(d) Residuals of the velocity.

Figure 8: Scatterplots of the noisy data (red), predicted state (green) and Kalman state estimate (blue) under different attack models. The four images represent the maximum magnitude (upper left), wave-based (upper right), positive deviation (lower left) and negative deviation (lower right) attacks.

Figure 9 displays the performance estimates of the different models and reveals, although the effect is small, that the positive deviation attack and maximum magnitude data injection provide the worst performance from the KF in a linear model. Whereas the negative deviation and wave based attack model have less impact on model performance.

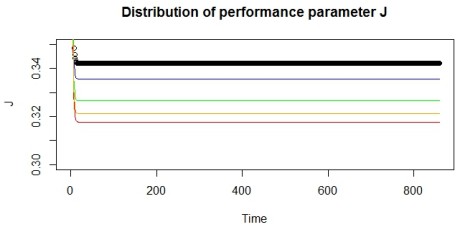


Figure 9: Plot of the performance index J for all timepoints, for the model without attack (black), maximum magnitude attack (orange), wave based attack (blue), positive deviation (red), and negative deviation (green) attack.

6.4.2. NON-LINEAR MODEL

The baseline (not attacked) non-linear multidimensional model is visually presented in Figure 10, which shows the trajectory of the aircraft, coming in at an altitude of 10 km, slowly decreasing in altitude for landing. During it's descent, there are several manoeuvres, mainly to the left (please note that the ADS-B data used here is not forwarded twice a second resulting in a pattern that is not really time-scaled). We can clearly see that in stable flight, there is a very good convergence of the predicted state and current (Kalman) estimates. When the aircraft decreases in altitude (and velocity; not in model) the predicted estimates slightly diverge, but the Kalman state estimation remains relatively close to the measured position. Especially during sudden manoeuvres, the residuals (Figure 11) show slight model divergence, which is resolved after a 20-50 seconds, illustrating typical KF behaviour. In normal flight, the residuals remain well within the 95% Confidence Interval boundaries, but the signal crosses the boundaries during fast and unpredicted changes of longitude, latitude, and altitude of the aircraft. These kinds of deviations in multidimensional systems are well known [270].

The trajectory plots for the flight under data injection are presented in Figure 12. Again, the maximum magnitude based attack (Figure 12a) showed considerable deviation from the trajectory, exceeding the deviation of the wave based attack (Figure 12b).

The performance indices for the different attack models are presented in Figures 13a and 13b. Which both show strong differences between the different attack models. In short, the positive deviation attack and maximum magnitude data injection result in the worst deviation from the baseline (no attack) model. The negative deviation and wave based attack model have less impact on model performance.

6.5. DISCUSSION

The overarching aim of this study was to test the effects of false data injection on state estimation with Kalman Filters in an aerospace setting. I replicated earlier findings [67], confirming that the wave-based attack had the least impact. This finding may be influenced by the variability in the data, as the wave based attack is an injection of signal in the opposite direction of the innovation factor. Although the innovation factor follows a normal distribution with zero mean, strong deviations from the expected result could

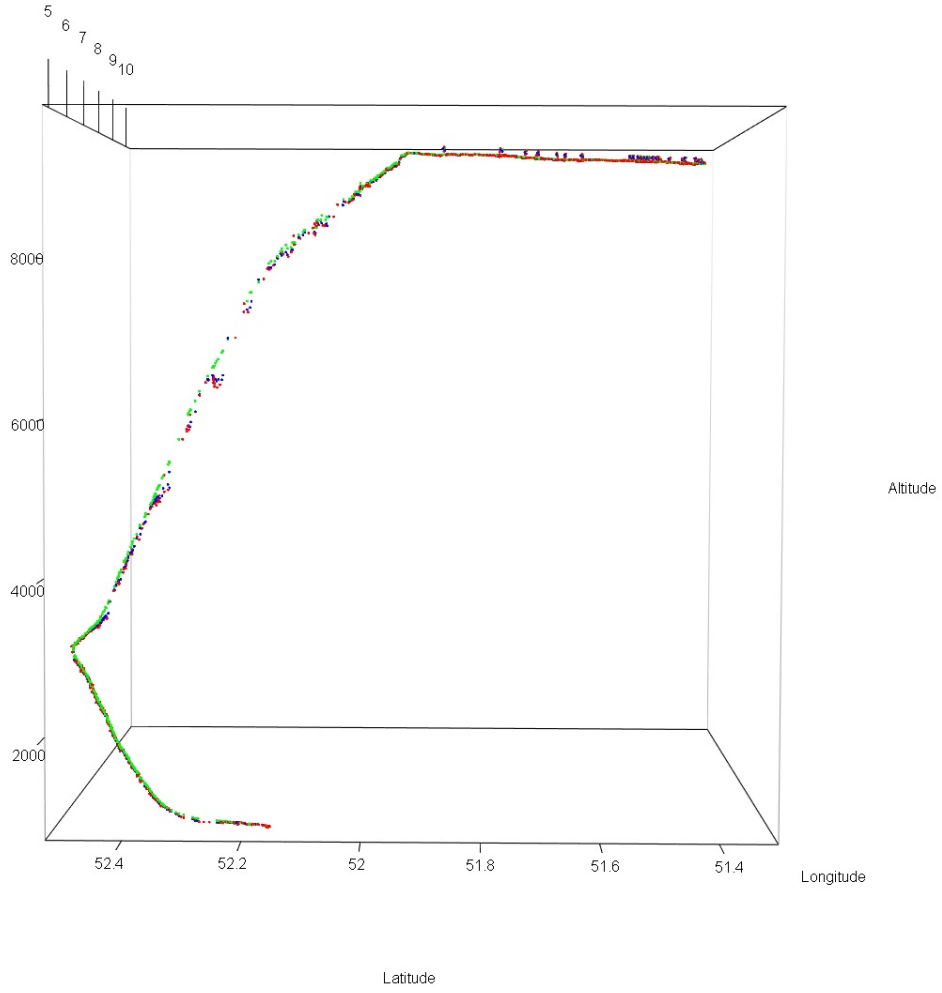
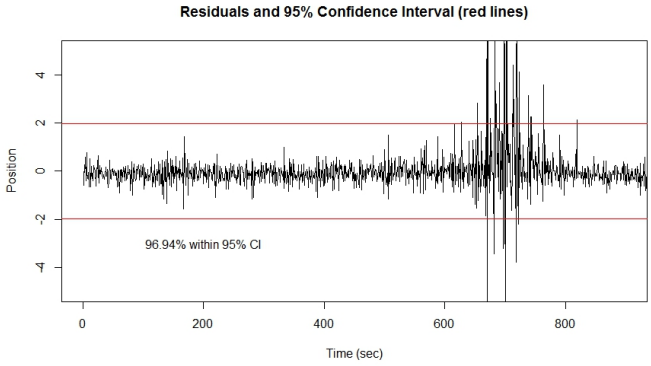
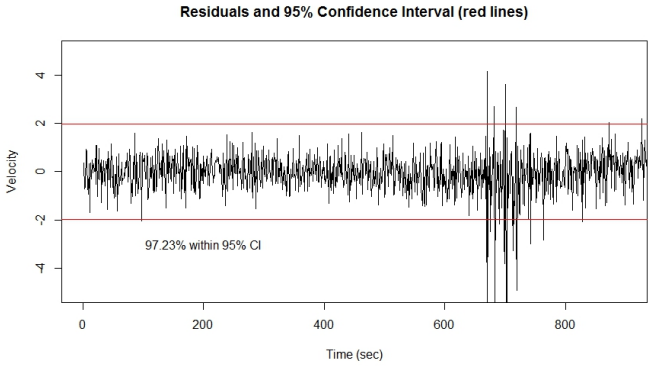


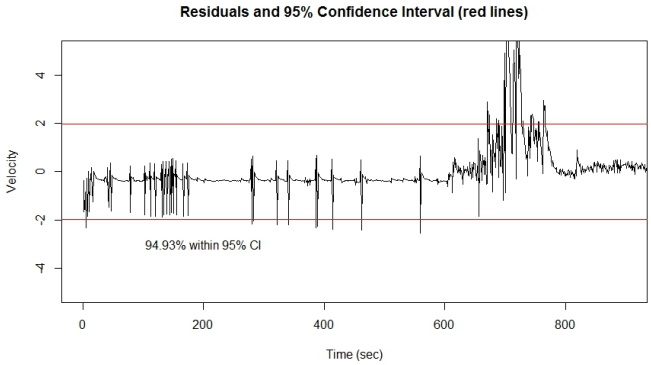
Figure 10: 3d plot of the flight trajectory without data injection, showing the aircraft in landing. Axes are longitude, latitude, and altitude. Colors are noisy (real) state in green, predicted state in red, and Kalman estimates in blue.



(a) Residuals of longitude.

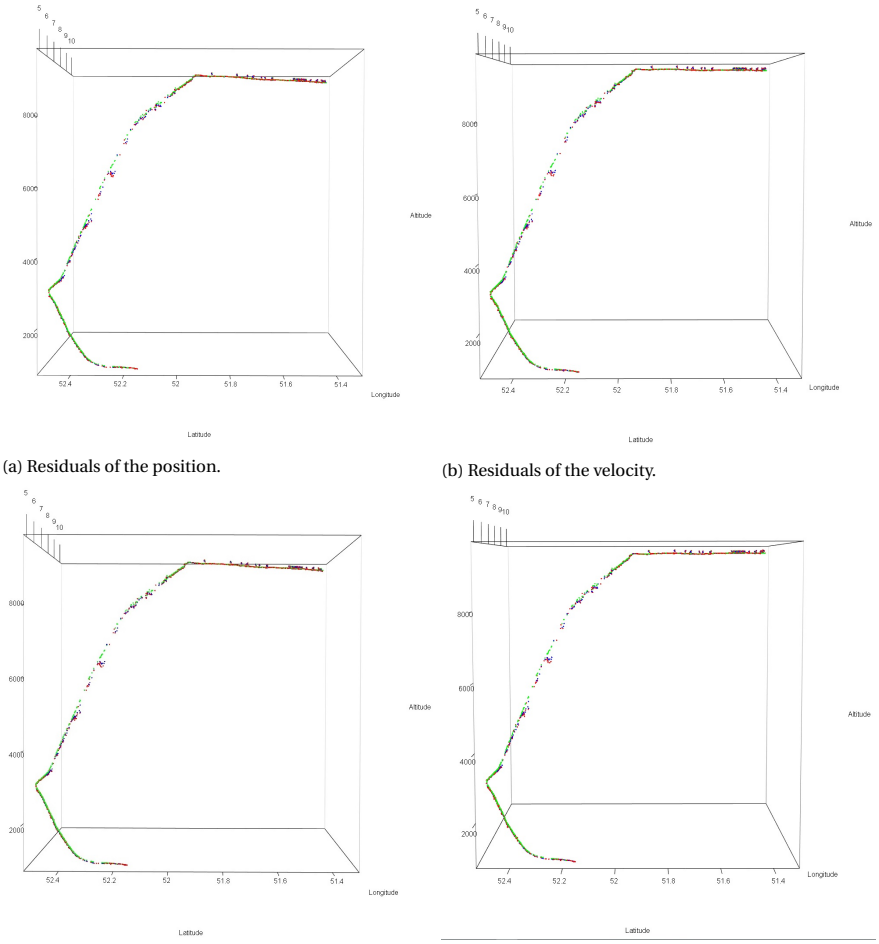


(b) Residuals of latitude.



(c) Residuals of altitude.

Figure 11: Residuals of different parameters from the non-linear model. We can see the effect of the data injection around 600 seconds after which the KF estimate diverges and the residuals increase to fall outside the 95% Confidence Intervals.



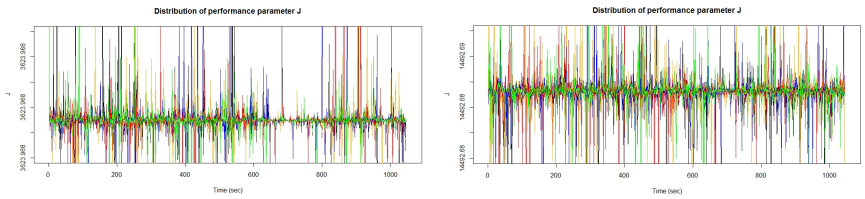
(a) Residuals of the position.

(b) Residuals of the velocity.

(c) Residuals of the position.

(d) Residuals of the velocity.

Figure 12: 3d plots of the noisy data (green), predicted state (red) and Kalman state estimate (blue) under different attack models. The four images represent the maximum magnitude (upper left), wave-based (upper right), positive deviation (lower left) and negative deviation (lower right) attacks. Axes are longitude, latitude, and altitude.



(a) Residuals of the position.

(b) Residuals of the velocity.

Figure 13: Plot of the performance parameter J for longitude and latitude under the various attack models (normal = black, maximum magnitude = orange, wave based = blue, positive deviation = red, and negative deviation = green.

be counterbalanced by data injection in the wave based attack. In contrast, attacks that manipulate the signal to the furthest possible (within the anomaly detection threshold λ_{max}) extremes have more impact on state estimation, especially when the injection is in the same direction as the predicted state. Kalman filter behaviour was in line with descriptions and reports from others [279, 273, 280, 270]. Arguably, one of the reasons why the KF and EKF models fitted provided a good fit to the data was that the data, even in the non linear model, did not contain very large non-linear patterns. The fit of the multidimensional model could be improved by weighting the individual velocity estimates in the dynamical model for the speed estimate forwarded from the aircraft, which was not done in this study because of the apparent complexity of modelling speed in miles per hour and longitude/latitude estimates. Indeed, in sudden manoeuvres the EKF diverged, only to return to normal after a few seconds. This EKF divergence problem could intensify if data would be highly non-linear [281, 282]. I am aware of the advantages for model convergence and computational stability of other filters like the unscented KF or the enhanced EKF, but current evidence does not show that these versions of the KF algorithm are more stable or secure when attacked with false data [67].

The addition of noise to the ADS-B GPS coordinates may have provided an overestimation of the variation in ADS-B data, causing the innovation factor distribution to be too wide. In data with small variability, the anomaly detection threshold could be more stringent because large deviations are not expected, allowing early detection of deviations (indicating attacks). However, even without noise added to the data, the anomaly detection window will likely to be large because it has to allow manoeuvres from the aircraft. Although velocity and altitude are major predictors for the probability that an aircraft makes a rapid manoeuvre. The anomaly detection threshold is assumed to be static, meaning it does not depend on these variables. Therefore, λ_{max} has to present an equilibrium that allows manoeuvring of the aircraft within its flight envelope, but stringent enough to detect anomalies. Consequently, it is likely that the frog boiling attack would still have been successful, even without simulated noise, because λ_{max} allows significant deviation from the predicted signal.

This work also confirms that the frog boiling method can be successfully used to attack state estimation systems, even with anomaly detection. It is unclear whether state estimation systems are currently equipped with anomaly detection algorithms. However, the severity of the attack (as modelled here) is limited by the innovation factor v_t

calculated from the measured state z_t and the predicted state $\bar{u}_t$. As the measured state is updated on every iteration of the model and is not infinite in its error (given the requirement of Gaussian noise), the effect of the frog boiling attack could be smaller than observed by others [258], since its impact is bounded by the anomaly detection system. This could also explain why our model deviated from its original state, but did not diverge as a result of the attack, which contrast other results [261]. Moreover, the innovation vector v_t is estimated for every variable that contributes to the state estimation. This study used the same standardized innovation vector threshold λ_{max} across variables, but variable-specific thresholds could easily be implemented to make the system more robust against attacks. Nevertheless, frog boiling remains successful, even with an anomaly detection system implemented.

6.5.1. COUNTERMEASURES

Following the apparent vulnerabilities of the KF, studies have proposed several countermeasures to mitigate or reduce the impact of false data attacks. Yang, Chang, and Yu [67] proposed to multiply the measurement noise with an exponential, which leads to a decrease of the Kalman Gain so that it favours prediction over measurement. They also proposed temporal-based detection, using the nonparametric cumulative sum (CUSUM) algorithms to detect change in the observations as early as possible. Another proposed countermeasure resembles more general machine learning anomaly detection and involves comparing state estimates to distributions based on historical data [68].

6.5.2. ATTACK MODELS

One of the limitations of the models used in this project (and others) could be the assumptions made about the attacker. This study assumed that attackers had full knowledge about the system, the incoming data, the anomaly detection algorithm, and the implementation of the state estimation algorithm(s). It is unclear to what extend these assumptions are valid. Maybe these assumptions stem from cryptography, a field where it is common practice to assume the attacker has knowledge about the technical and computational details of the crypto protocols [283]. It is difficult to prove these assumptions are wrong, but it could be a good idea to formulate general rules and best practice guidelines that can be used to formulate attack model assumptions. This could aid generalization of theoretical problems. Also, however worrisome the effects of false data injection are, the exact implications of false data injection attacks are unknown. In industrial systems, training data are often not available to attackers and the data-driven thresholds used in detection system (e.g., weights of words in spam filters) are not continuously updated with every new email (observation) but based on large amounts of historical data that have been screened intensively. Also, most learning processes are inherently robust against direct data injection attacks. Given the amount of data that is typically used in industrial data driven detection algorithms it is almost impossible for a single attacker to immediately change the underlying distribution of a detection algorithm. To prevent rejection of the injection as an outlier, one has to model the data injection careful to allow subtle deviation of the original signal. Hence, the outcomes of this project favour the use of data-driven security thresholds in state estimation systems. With data-driven thresholds, the best achievable scenario could be one where the vari-

ation of the distribution increases, broadening the boundaries for attack messages that will ultimately fall within the distribution.

6.6. CONCLUSION

This study reports the effects of false data injection on ADS-B derived position estimates of aircraft position. Data were injected in a linear model (Kalman Filter), investigating the change of radar-distance position, and in a non-linear model (Extended Kalman Filter) with the ADS-B GPS coordinates with simulated noise. For both models, the positive deviation attack and maximum magnitude data injection provide the worst performance of the filtered state estimation model. Whereas the negative deviation and wave based attack model have less impact on model performance.

7

INVESTIGATING RESIDUALS AS A MEASURE OF SURPRISE IN 219.810 CONSUMER CREDIT APPLICATIONS

This paper presents a proof of concept where we explored the usefulness of residuals to create a measure of surprise, as an indicator for unexpected credit-application responses, in the context of fraud detection. In a non-fraud training set we fitted generalized linear models to a subset of variables, to obtain coefficients from which predicted values could be estimated in the test set. The distributions of these residuals were then tested for significance against the fraud label. The significant residuals were combined to form a measure of surprise, which was experimentally added to the fraud detection model. We illustrate the potency of this approach on a classic open source Australian dataset from credit card applications, and then field test this procedure on a large representative database including 219810 consumer credit applications from a large financial institution (ING bank Netherlands). Both datasets included a subsantial number of covariates with a residual-distribution that was fraud-variant, and adding residuals to the ING fraud detection model increased explained variance but did not improve classification, implying the same classification is conducted with more certainty. The outcomes evidence that in fraud cases the reported responses in credit applications more often deviate from expected values and evaluation of residuals in application data can be an important step in understanding the trustworthiness of provided information and the detection of fraud.

Parts of this chapter have been presented at the Data Leaders Summit Europe Conference in Barcelona (2016). Title: Introducing Machine Learning at the foundation of your data sets: Ascertaining the validity of consumer data to increase the accuracy of fraud detection.

7.1. INTRODUCTION

Fraud is a crime that is often well-considered, organized, concealed, time-evolving and can take many forms [284]. Despite the low frequency of fraud events, the large number of transactions, financial products and the variety in which fraud can be perpetrated makes it a major problem to financial institutions (annual losses predicted to be 73 billion pounds in the UK alone). Fraud detection has benefited greatly from the statistical analyses of suspicious behaviour captured in data [75, 34, 35, 285]. Typical machine learning methods reported useful in fraud detection are regression analyses, neural networks, Bayesian networks, decision trees, and clustering via distance measuring.

A major complexity in fraud detection is that fraudsters are inherently dishonest about their intention. Especially when client-input is required, it is not unusual for fraudsters to purposely provide false or manipulated data to "blend in", mislead or corrupt the detection pipeline by presenting false, incomplete, or inaccurate data [76]. In banking, one obvious fraud context is credit applications, where clients apply for a credit online and provide information on different topics spanning income, marital status, purpose of the credit, etc. In our experience, there are two types of fraudulent clients in credit applications; those who provide falsified information, such as using a non-existent identity, or stolen information with the purpose of getting money quickly without paying back (hit and run), and those who provide falsified information with the purpose of getting (a higher) loan, but with the idea of paying it back (forgery). An example of the first group is a client who lies about marital status to prevent a partner with a bad credit history from being evaluated in the application. An example of the second type is a student who applies for a student-loan and lies about the year of study (credit height depends on student seniority). Both types are an obvious problem to any financial organization.

Although fraud detection algorithms aim to identify general trends of suspicious applications and transactions [35], the validity and integrity of the data underlying these models, apart from cleaning, has received little attention. Models merely trained on a fraud / not-fraud label can be useful, but when the features depend information given by the clients themselves the feature-weights may underestimate the risk posed by clients who provide false-information. Here, we explore a procedure based on the combination of residuals to detect abnormal responses in individuals who apply for credit a financial setting. On data from non-fraud cases, we use the covariance between features to train generalized linear models for each feature, where we use a significant subset of the other features as predictors. The remaining data (fraud + non fraud) are used as a test set in which we calculate the predicted value and residuals for each observation on every feature. We then investigate which features have residuals with a significantly different distribution in fraud vs. non fraud cases (ie. are informative for fraud status). We combine the significant studentized residuals as a measure of surprise. By training only on non-fraud cases we circumvent the complexity of unbalanced fraud / non-fraud categories. We first test the principle on a classic open source Australian dataset from credit card applications, and then field test this procedure on a large representative database including 219810 consumer credit applications from a large financial institution (ING bank Netherlands) and investigate the merits for the detection of fraud.

7.2. METHODS

7.2.1. AUSTRALIAN DATA

We used the Australian Credit Approval data as provided by dr. Ross Quinlan and retrieved online via http://archive.ics.uci.edu/ml/datasets/Statlog+(Australian+Credit+Approval), and includes data from 690 observations why apply for a credit card. Although the attribute names are not provided and values have been recoded to be meaningless, the data are useful since they hold a combination of categorical and continuous variables with a realistic covariance structure. There were 14 covariates and 1 dichotomous class label. Eight covariates were categorical, but item A5 and A6 were analysed as continuous (given the number of categories). Variables A4 and A12 had 3 categories and were analyzed as dummy variables (if independent) or multinomial (dependent). Variables 13 was log-transformed to a normal distribution. Variable 14 did not correlate with other covariates and was excluded from further analyses.

7.2.2. ING DATA

Data from January 2013 to July 2016 were extracted from two sources; an existing customer database (number of individuals (N) = 568341, number of variables (N_p) = 707) and application information (N = 3.887.625, N_p = 74). Customer database data included variables as age, income, address, customer group (e.g. young potentials), owning insurances, amount of credit and debit cards owned, etc. This information is usually available from the moment the client starts an account in the bank. From several clients, multiple records were available because they updated their data (e.g. because they moved to another geographical location) and we selected only the most recent information per client, resulting in data from 332435 clients with 707 variables. Identical information was available for 486 (474 unique) clients who committed credit fraud.

Application data were provided by the client at the moment of application and are typically product specific. This involves variables as current income, marital status (and legal construction), lending purpose, children, and whether one's income stems from a permanent or temporary contract. From the application dataset we selected 22 informative variables (eg, marital status, income, source of income, number of children). From the customer dataset, 28 variables were selected (number of applications, student or not, financial capital). As part of the non disclosure agreement, we do not report specific variable names. A flowchart of the data processing is included in Figure 6.

7.2.3. DATA CLEANING

CUSTOMER DATABASE

First, duplicate variables were removed (#p = 14) along with variables with zero variation (#p = 160), complete missingness (#p = 34). Dates were split into year and month which replaced the original date variables. This resulted in 332435 participants and 467 variables, including 247 continuous and 220 categorical variables.

For continuous variables, we observed considerable skewness (see Figure 1). For every variable, the skewness was calculated and compared against the skewness of it's log transformed values (for negative variables we added a constant $minimumvalue + 1$, see 3). In 194 of 220 variables the log transformation improved the distribution by low-

ering the skewness (see Figure 2). For several variables we observed a large number of 0 responses (e.g. clients who never deposited money on a savings account had a zero on that indicator variable). Zero was recoded to missing and we verified whether a zero (indicating never using that product or service or using it somewhere else) was related to fraud (see Figure 3).

Algorithm 3 Clean continuous variables

for i in 1:$N_{continuous}$ **do**
  if min($variable_i < 0$) **then**
    constant = |min($variable_i$)| + 1
    if |skewness(log($variable_i$ + constant))| < |skewness($variable_i$)| **then**
      $variable_i$ = log($variable_i$ + constant)
    end if
  else
    if |skewness(log($variable_i$))| < |skewness($variable_i$)| **then**
      $variable_i$ = log($variable_i$)
    end if
  end if
end for

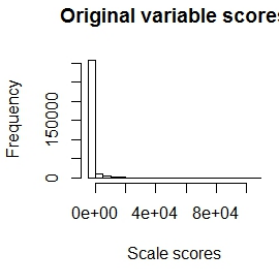


Figure 1: Original variable with raw values.

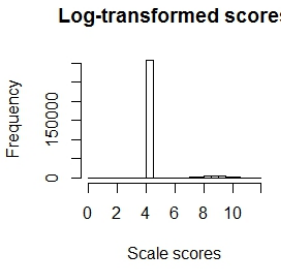


Figure 2: Variable with log-transformed values.

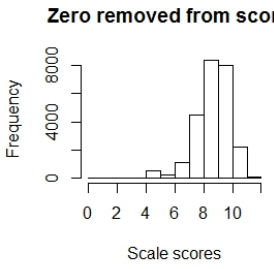


Figure 3: Variable with log-transformed values and zero category set to missing.

For categorical variables, we calculated the skewness by evaluating the number of individuals in 1 category. If one category included 95% of the observations the variable was dichotomized (collapsing all the other categories). All categorical variables were recoded to have 0 as lowest category.

Variable missingness was calculated by taking the proportion of missing values per variable (see Figure 4) removing 182 variables with ≥ 75% missing values. Uninformative variables prone to produce biased estimates were also removed (e.g. given the relatively small number of fraud cases, the range of House Numbers is smaller in the fraud sample because large numbers are decreasingly likely and only occur in large samples, resulting in a (spurious) mean difference between fraud and non-fraud). Individual missingness

was calculated by counting the number of missing values for all items for every individual (see Figure 5). Only individuals with ≥ 90% (N = 279702) observed data were included. The removed group included 74 fraud cases (0.14% of 474) which was an equal proportion of the fraud cases in the included group (400 of 279702), indicating no bias in individual missingness. Hence, data cleaning of the customer database resulted in a sample of 279702 individuals and 309 variables, including 55 variables with missingness < 25%.

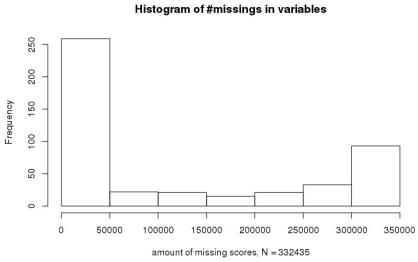


Figure 4: Frequency of variables given the amount of missing observations.

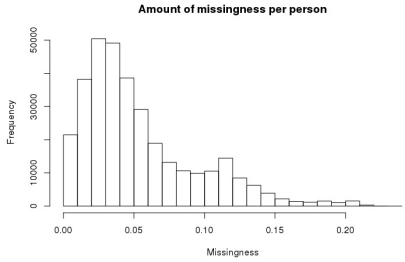


Figure 5: Percentage of missingness per person.

APPLICATION DATA

During the inclusion period, 3887625 credit applications were received from N = 332251 unique clients. The application data included $\#p$ = 74 variables and 2178 applications labelled as fraudulent from 298 unique clients. This dataset included many duplicate applications from clients who applied for the same credit product on multiple occasions (e.g. through internet) or clients who reapplied to increase their credit. For analyses, one value was selected per person. First, the accepted (N = 207990) applications (if multiple existed only the most recent was selected) were included, followed by rejected applications (N = 123944). Data cleaning of the application data resulted in a sample of 219595 individuals and 56 variables. Merging these data resulted in a combined sample of 219810 individuals and 337 variables.

7.2.4. MULTIPLE IMPUTATION

Imputation of missingness was only conducted for variables from the customer database because these data are not directly derived or influenced by the customer, and often rely on multiple checks (e.g. home address can be checked regularly through municipal services). The data were partitioned into fraud (N = 400) and non-fraud (N = 219367). The non-fraud cases were randomly split into a training N = 50000 and test N = 169302 sample. Splitting the non-fraud cases into a training and test dataset was important to prevent over-fitting and to lower the computational burden. Because we partitioned the data, the number of missing values per variable also decreased resulting in 10 variables becoming completely observed. Highly correlating variables ($\#p$ = 4) and the fraud label of the application data table ($\#p$ = 1) were removed. For 2 variables, one or more levels were collapsed because partitioning the data in training and test samples resulted in zero

endorsement of 1 or more levels. Thus, the final dataset for imputation included 50.000 individuals and 332 variables, of which 39 variables were imputed (39 continuous and 1 multinomial).

Missing data were imputed with Multiple Imputation (MI) using *mice* in the R programming language. MI is known to be very robust as it prevents crude manipulations of the distribution of data (e.g. in mean imputation) or underestimation of the variation (e.g. in regression imputation). The MI procedure consists of three steps: imputation, analysis, and pooling [286]. From the training sample, five $m = 5$ complete datasets were created with imputed values. The missing values are replaced by plausible values that are drawn from a distribution specifically modeled for each missing entry (see below). The imputed data remain identical for the already observed values but differ in the imputed values. After imputation, analyses are conducted on the different imputed datasets and results are pooled. For every variable we specified the imputation method, which was a Bayesian linear regression for numeric continuous scales ($\#p = 100$), a multinomial logit model for nominal scale variables ($\#p = 47$), and a logistic regression for binary scale variables ($\#p = 168$) and an ordered logit model for ordinal data ($\#p = 17$).

TYPE OF MISSINGNESS

Because subsequent analyses will involve all variables related to all other variables, we assumed Missingness At Random (MAR). Defining the type of missingness related to fraud is difficult because prima facie evidence may suggest MCAR (e.g. missingness in 'amount of children' or 'having an insurance product') but later analyses could show that fraudsters are men who do not seek strong affiliation with a company (typically not consuming many products). In that context, the missingness does become informative, so that the probability of being missing depends on some parameters ψ and the observed information $Pr(R = 0|Y_{obs}, Y_{mis}, \psi) = Pr(R = 0|Y_{obs}, \psi)$ as presented in [286], page 31.

PREDICTION

Ideally, MI uses as much information in the dataset as possible to reduce bias and reach maximal certainty [96]. However, given the number of variables in this dataset, it was (computationally) unfeasible to include all variables as predictors and only the best 25 predictors were selected based on correlations. Variables that did not contribute (e.g. house number, being deceased, postal code, relation number, birth year (we already had age), etc.) were excluded as predictors from the prediction matrix. Evaluation of MI performance showed that imputation was not successful in 1 variable since imputed values did not follow the observed data distribution. This variable was not included as predictor or covariate in subsequent analyses.

7.2.5. PENALIZED REGRESSION ILLUSTRATION

There was a risk for perfect separation if zero observations in one field of the 2 × 2 table, as possible in the analyses of two dichotomous items. An example was the variable deceased / not deceased. None of the clients who committed fraud had died, whereas in the non-fraud group 29 clients had. This problem addressed by using a bias-reduction method based on Firth (1993), implemented in the *brglm* package [287] that calculates β with a Newton-Raphson algorithm. For practical reasons one polytomous variable (level of education) which was dichotomized into low and high education (see 7.2.5).

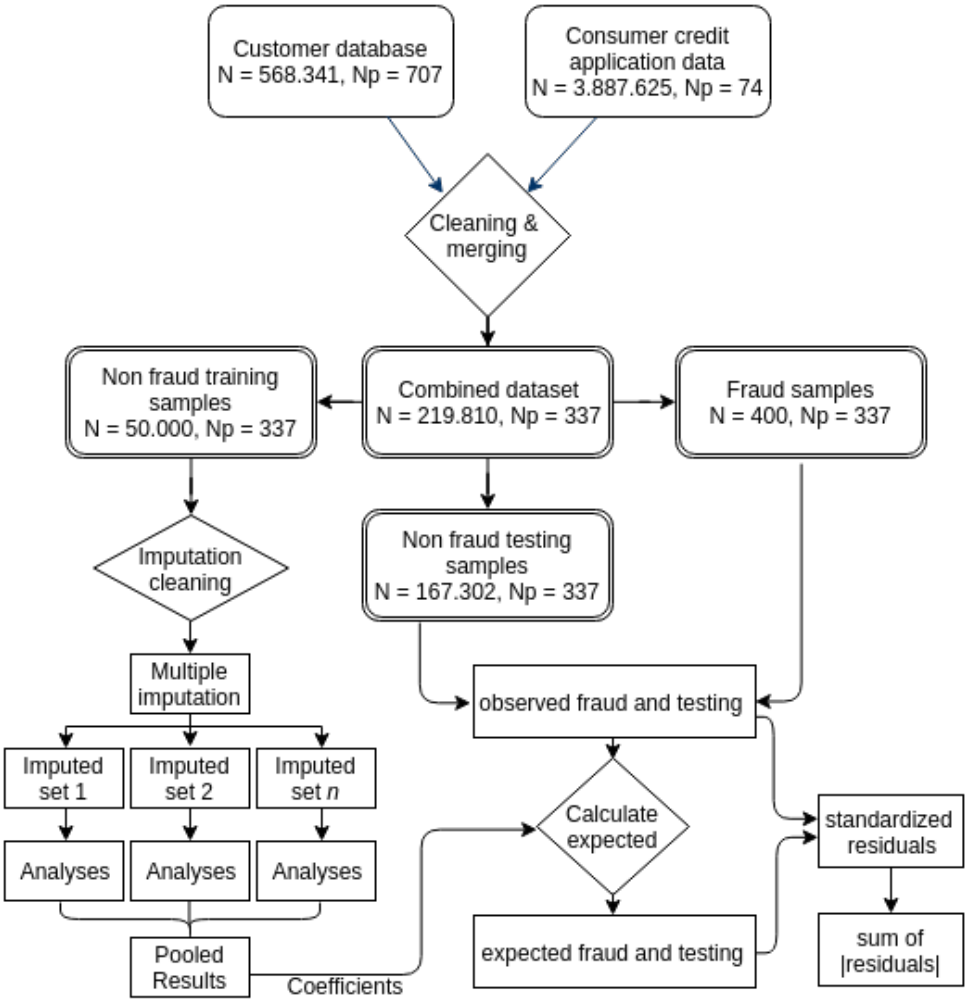


Figure 6: Flowchart of the data cleaning, multiple imputation, and analyses

As an example of the substantial bias that can occur in regression coefficients from a logistic regression, this illustration shows the difference between βs from a standard GLM logistic model with one random dichotomous dependent variable and the intercept + 30 random independent variables (IV). In total, this sub analysis included 8 continuous and 22 categorical variables. For most IVs, the regression coefficients are fairly similar between both methods with the difference ranging between 0 and 1. For 11 (categorical) variables + the intercept, the difference was larger than 1. In 4 of these 11 categorical variables, one group or category had all the observations (that is, the proportion of observation in 1 category was equal to 1). in the other 7 variables, this problem was less extreme as there was at least 1 observation in all categories and the differences be-

came smaller as the amount of observations in the rarest category increased.

MODELING

We start with a dataset X, with N observations on the rows and p covariates on the columns. The Australian data were randomly split in a train- ($X_{tr} : N = 150$) and test ($X_{te} : N = 538(C_1 = 302, C_2 = 236)$) set. In the ING data, data cleaning (see 7.2.3) and merging resulted in an overall sample of 219767 individuals including 400 fraud cases and 332 variables. In X is a class variable (credit approved / fraud) which we define as Y. We randomly split X into a train- (X_{tr}) and test (X_{te}) set, where X_{tr} consisted of 50000 non-fraud cases and X_{te} included 169367 non-fraud and 400 fraud cases. Before analyses, missing data in 39 variables in X_{tr} was imputed via multiple imputation as implemented in *mice* (see 7.2.4). From the 332 covariates in $X_{tr} : j \in \{1, ..., p\}$ we selected 50 variables to become dependent variables (sequentially) in a generalized linear model (glm);

$$X_{tr_j} = \beta_0 + X_{tr\_j}\beta_{tr\_j} + \epsilon \tag{7.1}$$

where X_{tr_j} is the credit application variable selected to be the dependent variable, X_{tr_j} are the other variables that significantly predict X_j, β_{tr_j} are the regression coefficients (with β_0 as intercept) and ϵ is the model error. If X_{tr_j} was continuous this would amount to fitting a standard linear model, if X_{tr_j} was categorical the link function of the glm would be Logit in a binomial or generalized Logit in a multinomial response. Poisson regression models were applied to count variables. Covariates that were polytomous ($i = 5$) were analyzed with multinomial logistic regression using R package *nnet*. In short, this procedure requires to specify a baseline group and the analysis consists of $(g-1)$ distinct logistic regression functions with the same p explanatory variables which are computed for the g categories. Then, the logistic regression functions are combined into one multinomial equation that includes the intercepts from the $g-1$ logistic regression functions plus the regression coefficients for the p predictors in each regression function.

Modelling involved evaluating multicollinearity, dimensionality reduction with Principal Component Analyses to remove highly correlating covariates and removing covariates with high missingness in the test-sample.

We used β_{tr_j} from X_{tr} to calculate $\hat{y}_{te_j}$ (the predicted value for X_{te_j}) based on the predictors in X_{te} (as determined in Equation 1) so that for every observation in the test set we could calculate the residual. For continuous and poisson variables, residuals were calculated as $e_{te_j} = Y_{te_j} - \hat{y}_{te_j}$. Studentized residuals were calculated by dividing e_{te_j} by the standard deviation of the residuals of non-fraud (nf) samples in the test dataset: $r_{te_j} = \frac{Y_{te_j} - \hat{y}_{te_j}}{sd(Y^{nf}_{te_j} - \hat{y}^{nf}_{te_j})}$.

For logistic responses, the output represents the probability of a 1 for every individual given the data: $\hat{y}_{te_j}(1|X_{te_j}) = 1$ so as residual we defined

$$r_{te_j} = \begin{cases} \hat{y}_{te_j}, & \text{if } Y_{te_j} = 0 \\ 1 - \hat{y}_{te_j}, & \text{otherwise} \end{cases} \tag{7.2}$$

To investigate differences in residuals between fraud and non fraud samples the cumulative density function of residuals was evaluated with the Kolgomorov-Smirnov test for all 50 variables. This ultimately results in a subset of residuals that have a distribution that is significantly different between fraud and non-fraud cases, which can be summed to create one composite measure of surprise: X_{sur}.

7.2.6. FRAUD MODEL

We created one fraud detection model to predict fraud status in each dataset by fitting a glm to the class label (Australian data) or fraud status (ING data). The model was trained on 60% of all the data (including the fraud label) and tested on the remaining observations. We investigated the merit of the residuals to the fraud status by comparing the percentage of fraud cases in different percentiles of $X_{te,sur}$. We also added the composite variable to the null model (regressing fraud status on the original variable) to investigate explained variance.

7.3. RESULTS

7.3.1. AUSTRALIAN DATA

For each of the 13 covariates, residuals were calculated in the test dataset based on the coefficients from the glms on the training data. Variables 3, 4, 5, 6, 7, 8, 9, 10, and 13 had residuals that were significantly different ($p < .05$) between fraud and non-fraud cases, which were summed to create $X_{te,sur}$ (see Figure 7). Adding $X_{te,sur}$ to the fraud detection model did not result in a significant increase in explained variance: McFadden's $R^2 = .56$ in both models. Notably, $\beta_{te,sur}$ was not significant due to multicollinearity with other predictors.

7.3.2. ING DATA

In total, 50 variables from the application data were investigated and 18 variables had residuals that were significantly different ($p < .05$) between fraud and non-fraud cases and were summed to create one $X_{te,sur}$ (see Figure 8). Adding $X_{te,sur}$ to the fraud detection model significantly increased explained variance: McFadden's $m_0 : R^2 = .41$ and $m_0 : R^2 = .59$. Classification performance was quantified with the Area Under the Curve (AUC: see Figure 9, which was .89 in the baseline model and .84 in the baseline + residuals model, which is not significantly different and indicates excellent accuracy. Finally, checking the percentage of fraud in the 90th percentile of $X_{te,sur}$ resulted in $P_{90} = 0.15\%$ which was almost twice the global percentage in the fraud-detection model testset of .08%, indicating that on and beyond the 90th percentile, the saturation of fraud cases was twice as high compared to the overall sample.

7.4. DISCUSSION AND CONCLUSION

This paper presented a proof of concept where we explored the usefulness of residuals to create a measure of surprise, as an indicator for unexpected credit-application responses, in the context of fraud detection. By building a model on a non-fraud training-set we circumvented the complexity of unbalanced data classification due to low fre-

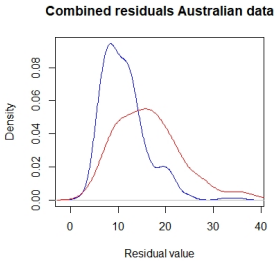


Figure 7: Distribution of composite residual variable (blue = non-fraud, red = fraud)

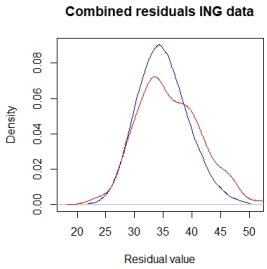


Figure 8: Distribution of composite residual variable (blue = non-fraud, red = fraud)

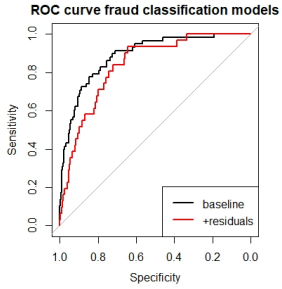


Figure 9: ROC curve (black = baseline, red = baseline+residuals)

quency fraud status. In both datasets we calculated residuals and detected a significant number of covariates with a residual-distribution that was fraud-variant. We observed that creating a composite residual variable could increase the explained variance but did not improve classification, which implies that the fraud-classification is only made with more certainty. This could be expected since the residual distribution indirectly stems from (parts of) the raw data, and when a covariate is only predicted by one or two independent variables this increases the correlation between the composite residual and the original data (although this problem was less pronounced in the ING data, given the heterogeneity of the sample and the amount of predictors per model).

The major shortcoming of this paper is that we were unable to validate the measure of surprise by checking whether the top-observations had indeed falsified their input data or responses. Apart from validation being a tedious case-by-case process there was limited access to the data or INGs infrastructure. The findings do show that in a real-life setting there is evidence that in fraud cases the reported responses in credit applications more often deviate from expected values and evaluation of residuals in application data can be an important step in understanding the trustworthiness of provided information and the detection of fraud. Although significant, most differences were subtle and bidirectional, suggesting the existence of substantial heterogeneity in fraudulent applications compared to legitimate applications across the full range of the distribution. For example, the amount of years in a job is likely to be lied about since it is easy to deduce that longer and more stable job status enhances credit worthiness. However, the data did not support the idea that fraudsters exaggerate their job-years. Instead, the reported amount of years in a job was observed to be lower in fraudulent cases compared to legitimate applications (which is normal from a risk perspective but somewhat counter-intuitive if one assumes exaggeration by fraudsters). The observed heterogeneity argues in favour of a matched design where fraudulent applications or persons are matched based on a small sample of descriptive covariates, or partitioning the data into subsets with a desired distribution [288].

Current studies have been criticized for being too focused on complex methods (neural networks, fuzzy logic, support vector machines) while easier and straightforward methods can often produced similar results [35]. In other fields there is evidence that

models allowing non-linear discriminants outperform linear models (e.g. k-nearest neighbour [289, 290, 74]), but for fraud analyses there is a strong necessity to understand performance of different models with different types of data. Residuals evaluated in this study stem from straightforward regression analyses which seemed sensitive enough to find new and informative features. Despite the volume of complex techniques, many studies are not very explicit about data cleaning and processing (likely due to the sensitivity of the provided data). Especially in large companies, where data is often collected from different products in different databases with varying architectures and formats, good description of data cleaning procedures could clearly improve the generalizability of proposed methods.

8

CONCLUSION, REFLECTION, AND FUTURE WORK

This chapter presents the main contributions of this thesis and lessons learned. A number of limitations are discussed and suggestions for future work are put forth.

8.1. CONCLUSION

This thesis addressed a number of statistical problems that occur when one is faced with data collected in the cyber-domain, such as in cyber physical systems, large numbers of credit applications, social network data (typically collected online), and captures of computer network traffic. There is no reason to assume raw data collected in settings where humans are involved are immediately suitable for statistical analyses, which is why we systematically investigated the values people reported with the values we expected, in a context where the respondents can directly benefit financially from the outcome (will I receive money or not?). Nor is there any reason to assume collected data are complete. In fact, missing data is a common problem in many settings and this thesis addressed that issue by furthering the work on imputation of attributes in a setting where observations are linked. Once data veracity and completeness are safeguarded, one may start analyzing the data. We illustrated how the combination of clustering of edges and nodes can help in understanding hidden subgroups in data with a spatial (network) and temporal (collected over time) structure.

8.1.1. IMPUTATION OF ATTRIBUTES IN NETWORKS

Imputation of attributes (or covariates) has traditionally focused on contexts where cases were independent samples from the population [291, 292]. This follows most methodological research in behavioural sciences where statistical tests often assume independent draws from the population for robust (co)variance calculations [293, 294]. The problem of missing data in networks had already received attention in the framework of generative graph models [126, 128] and link prediction [295], but in chapter 3 we

extended Autocorrelation Regression Models (ARMs) to deal with missing data in attributes. We showed that ARMs are a straightforward and tractable platform to impute attribute values of persons in a social network, who connected through a messaging app. In an experimental setting where we gradually introduced different sorts of missing data, the best performance was achieved with a model where there was no feedback from the imputed data to estimated parameters. We specifically chose the sampling (Bayesian) approach to estimate the parameters, as sampling optimization is known to provide particularly accurate results [99, 100], and our imputation results supported that decision.

8.1.2. CLUSTERING OF SPATIO-TEMPORAL NETWORK DATA

Chapters 4 and 5 dealt with the problem of detecting infected computers in capture(s) of network data, when no labels are available. When this PhD project started (late 2015) all botnet (networks with infected computers) detection studies make use of a labelled dataset where the computer can use these labels (botnet / no botnet) to learn what an infected machine looks like. We showed the usefulness and applicability of unsupervised learning to separate infected machines from non-infected machines. In the first attempt this was only successful in a simulation study (chapter 4), followed-up with an approach where the temporal structure of the raw data was captured in features from an innovative spectral clustering approach to connection clustering called MalPaCA (chapter 5), resulting in excellent classification results.

8.1.3. DATA INJECTION IN STATE ESTIMATORS

Chapter 6 reported the effects of false data injection on ADS-B derived position estimates of an aircraft's position. Data were injected in a linear model (Kalman Filter), investigating the change of radar-distance position, and in a non-linear model (Extended Kalman Filter) with the ADS-B GPS coordinates with simulated noise. For both models, the positive deviation attack and maximum magnitude data injection provided the worst performance of the filtered state estimation model. In contrast, the negative deviation and wave based attack model had less impact. One of the limitations of the models used in this project (and others) is the assumption that attackers had full knowledge about the system, the incoming data, the anomaly detection algorithm, and the implementation of the state estimation algorithm(s). This broad assumption is often found in cryptography, a field where it is common practice to assume the attacker has knowledge about the technical and computational details of the crypto protocols [283]. Future research could benefit from general rules and best practice guidelines that can be used to formulate attack model assumptions. Also, however worrisome the effects of false data injection are, the exact implications of false data injection attacks are unknown. In industrial systems, training data are often not available to attackers and the data-driven thresholds used in detection system (e.g. weights of words in spam filters) are not continuously updated with every new email (observation) but based on large amounts of historical data that have been screened intensively. Also, most learning processes are inherently robust against direct data injection attacks. Given the amount of data that are typically used in industrial data driven detection methods it seems unlikely that a single attacker can immediately change the underlying distribution of a detection algorithm. Recent developments in adversarial learning [296] describe how an adversary can poison a system

where the training data or trained model are manipulated to cause specific misclassifications. Although mostly theoretical, the infection process is gradual and always involved multiple datapoints.

8.1.4. RESIDUALS INDICATIVE FOR FRAUD STATUS

Chapter 7 analysed data from ING bank, collected when clients who applied for a consumer credit also answered a number of questions regarding their personal and financial situation. Generalized linear models showed that some of these questions are provided with answers that are unlikely given the response to other questions. We showed that residuals (= observed - expected), for a selection of questions, correlated with fraud status. This suggests that people who commit consumer credit fraud are more likely to respond with surprising (unexpected) answers to some of the application questions. The definition of fraud includes the element *well-considered*, and there is some literature suggesting that fraudsters are more likely to be dishonest [76]. This prompts the question whether our observation is evidence that people who committed fraud with consumer credits are more likely to lie on their applications, or provide false / deceiving responses. We were the first to show, in this setting, the merit of a *measure of surprise*, based on residuals, to predicting fraud status.

8.2. REFLECTION

Cyber security is a relatively young field, and the birth of the (sub-)field cyber statistics was only a few years before the start of this project. As a result, there are many area's where the field can be improved and progress can be made. This thesis provides a small and selected illustration where those improvements can be made, but there is good reason to assume some of our research findings are limited by the immaturity of the cyber security research field. A general impression of possible (study-design) limitations has been presented in a famous essay from John Ioannidis [239], outlining several factors contributing to incorrect conclusions. A selection of these factors applies to the botnet literature:

1. No null findings reported: there are no botnet detection papers presenting an analyses where the attempt to detect infected computers was unsuccessful, for example because of heterogeneity in computer "behaviour", noise in the data, or insufficient sample sizes.

2. Important results have never been replicated (e.g. by testing the exact procedure on another dataset). There are some studies that use cross-validation [216] or divide the data in a train and test-sample [167, 171] to prevent over-fitting [297], which makes the analyses more robust. However, the botnet captures are often (very) small relative to the background data to which they are compared, introducing the problem of analyzing unbalanced categories. When the classification method becomes specifically tuned towards predicting the highest category (uninfected cases) we know accuracy decreases to be a valid performance metric [298].

3. Isolated studies: there has now been 15 years of literature on botnet detection and although most studies refer to previous literature, there are only a few contribu-

tions that actually build on previous insights. Most reported findings are presented in isolated studies by single teams, and this arguably results in limited coherence in the literature. Many different algorithms have been applied to the same problem, but lack a clear rationale that explains why those methods are selected or why one method outperforms the other.

4. Bias: as mentioned many studies apply some kind of manual filtering prior to statistical analyses and do not present the outcomes without filtering (e.g. removing approved DNS addresses via whitelisting based on Alexa [172, 231] or other rule based exclusion criteria [220, 234, 299, 222, 237, 238, 230]), nor test the appropriateness of thresholds by comparing output under different settings. Another example is the investigation of only synthetic data [300] without any extension to captures obtained *in the wild*.

5. Sparsely reported results: a minority of studies [228, 301, 181, 240] present (very) limited information regarding methods and results, leaving several important procedural (e.g. data cleaning and interpretation) and design questions unanswered.

These limitations are inherent to a young scientific field where small steps can represent significant advances. For example, in the field of genetic epidemiology, genome-wide-association studies became available in 2005. This allowed to scan large numbers of places in the (human) genome where variation is known to occur. Acknowledging the burden of early-day false-positives, that field has adopted standards to correct for multiple testing and replicate important findings [302]. Ideally cyber statistics will slowly evolve so that the quality of reported findings increases.

8.2.1. HOW THIS THESIS DEALT WITH COMMON SHORTCOMINGS

The studies in this thesis attempted to overcome and avoid some of the aforementioned pitfalls. First, we reported an unsuccessful clustering attempt in chapter 4 where the simulation study suggested that the SBM approach as sensible but fitting the model to the real data appeared more challenging. Second, we replicated the main finding either by using different sampling seeds (chapter 3), or by applying the same procedure on a different dataset (chapter 5). In line with the winner's-curse phenomenon [303], the replication outcomes were not as pronounced as those in the discovery sample, but still confirmed the main result. Third, we provided a clear rationale for the analytical approaches presented in our work, instead of pumping the data with an unmotivated selection of classification methods. Fourth, we aimed to limit the bias by removing the need for prior information (e.g. flat priors in chapter 3.1) and filtering (chapter 5). Finally, we also openly discussed our methods and results in all papers, for example by publishing Supplementary Material and making the code publicly available.

8.3. OVERARCHING CONTRIBUTION

The overarching contribution of this thesis is the presentation of new ways to gain a better understanding of data in the cyber-context. This is important as more and more systems process or generate data, sometimes with increasing complexity due to volume, data quality issues, or unobserved structures. Especially the (social) network literature

is filled with examples where observations are, to some extend, linked through a physical (geo), social (cultural, economical), or direct (communication) structure. This linkage bounds the estimation of parameters under the burden of missing data to more restrictions and complexities, and in chapter 3 we have shown how to deal with missing data in an accurate and efficient way. This hopefully allows better estimation of statistical models on network attributes so that different phenomena may be better understood. Now that online social networks are becoming a structural part of our lives there is also increased mixture between the online and offline environment. Common examples unions, clubs, or other peer-groups who have some kind of online presence where people can meet each other or share content. Therefore, we deemed the development of the ARM-based imputation models particularly opportune.

Another complexity in the cyber context is the processing of spatio-temporal data, due to the dependence in the observations and volume (real-time streams of packets or large networks), and accurate methods to describe and summarize these data by clustering, labelling or classification are scarce. With chapters 4 and 5 we add to the field how mixture models can provide a scalable and (very) accurate way to help in that process, removing the need to manually curate the data (e.g. filter domains, ports or hosts) while respecting the sequential structure of the data and the dependence between observations. This helps in allowing the analyses of streaming data with less interference (and potential bias) from users or analysts, hopefully increasing the overall quality.

Finally, we showed why it is important to consider data veracity when data are (partly) processed or collected automatically. Given that fraud is a conscious act, previous studies already suggested that people who aim to commit fraud actively try to conceal their activities by being dishonest [76]. We quantified these patterns of behaviour by investigating which residuals were predictive for fraud status, and the main messages of our proof of concept study with ING bank Netherlands (chapter 7) is that the responses from clients in online (credit) application systems should not be taken for granted, as there is clear evidence that people who committed fraud with these products more often report strange and unexpected responses. Using a substantial dataset our contribution is a plain showcase why straightforward application of fraud-detection models on client responses without the proper handling of data veracity issues is credulous.

Along these lines, chapter 6 showed how (input) data can be manipulated or influenced (e.g. by suspicious reporting) and how problematic this can be for state optimization or fraud detection. As mentioned, there have been several examples of data-injection attacks to cyber infrastructure and the popularity of Kalman Filters requires some understanding of how these state estimators can be maliciously influenced, so that these can be implemented with more (in)sensitivity to attacks and the security of state estimation can be increased.

8.4. FUTURE RESEARCH

8.4.1. IMPUTATION IN NETWORKS

Simultaneous with the publication of chapter 3, volume 62 of *Social Networks* also included a review on imputation strategies in networked data [304] where actor non- response occurred if all outgoing tie variables of an actor are missing. That study ob-

served a performance difference in link reconstruction between small and large datasets. In small datasets, simple ERGM performed well, while missing data was more accurately imputed in larger data sets with multiple imputation by complex Bayesian ERGMs (BERGMs). For the imputation analyses, this thesis did not simulate data, but used real-world data, which explains why we did not change with sample size. It is likely that BERGMs can also be used to impute node attributes [128]. It is unclear whether identical effects will be observed when we impute attributes with ARM, since ARM is not a generative network model. However, it would be interesting to investigate the imputation accuracy of cutting feedback and full Bayes, as with very large sample sizes, the effect of multiple draws diminishes and perhaps there is a plateau where sufficient observations and variation is reached so that any model will perform with high accuracy. Other applications where this model may be useful is in data with some (geo)spatial structure; e.g. recovery of the region of New Orleans in the aftermath of hurricane Katrina [305] or disease patterns in Soutch Korea [305].

Furthermore, our observation that a cut model outperforms full Bayes is convenient since most imputation impede feedback from imputed observations to parameters [94]. In full Bayes, the missing data are replaced with a starting value which is updated at the end of every MCMC step, meaning these values can flow to the estimated parameters. There is hesitance in the imputation literature to this feedback principle: in the context of multivariate imputation, feedback loops where imputed values in Y_1 are used to impute Y_2 are warranted (see [58], paragraph 4.5.4), especially if these distributions are not compatible [306] (even if Gibbs provides seemingly reasonable estimates). Testing or proving compatibility between (bivariate) distributions is not trivial; recently presented examples [307] are not readily applicable to real-world data. I have emailed Dr. Indranil Ghosh to inquire the latest developments in compatibility testing and the availability of source code, to which he responded: *"... the concept I had invoked is very new and in fact we are still working on the development of associated results to the multivariate case. After this we will focus on developing some code in R/Matlab/Python or so. To the best of my knowledge, I have not seen any available programming in any of the environment such as R/Matlab/Python or so."*. Understanding compatibility can important in multivariate imputation models that implement feedback loops and developing the capability to perform such analyses would be a welcome contribution.

8.4.2. CLUSTERING

There is a large body of research on analysing social networks [131], and we confirmed (chapter 5) that different clustering methods applied to the same data do not necessarily identify/restore the same clusters. This could depend on the type of network structure [195, 49], but comparison between the computer network data and the social network data in this thesis also revealed differences in the frequency and amount of connections. If the (repeated) occurrence of a connection is used as a weight in the network matrix, the type of network (computer versus social) determines the shape of the distribution of the network matrix, which likely influences the suitability of different network clustering methods, since their optimization often relies on the distribution of edge-weights between nodes in the network (e.g. [308, 188]). When these edges are sparse (because links have a low frequency), the model may not converge or become very sensitive to

starting values (especially with Variational Inference). These effects are not yet sufficiently described in the literature. Finally, chapter 1 mentioned the problem of volume and scalability of the analyses. Developments like stochastic- and black-box variational inference [245, 249], and efficient MCMC [99] aid the deployment of complex network clustering methods at scale [309]. The availability of open-source tools (e.g. SVINET) helps to apply these models, especially if they would include the option to include covariates and come with proper documentation.

8.4.3. DATA VERACITY

Despite major efforts in statistical fraud-detection and risk modelling, there is insufficient attention for the problem where persons can provide deceiving responses, to avoid detection by feeding false answers to a detection system. Perhaps this problem can be addressed in the context of adversarial machine learning [296], where adversarial input perturbations can subvert the predictions of a system. Perhaps, the popularity of adversarial machine learning will eventually absorb veracity-challenges as presented in this thesis.

BIBLIOGRAPHY

[1] Yaser S Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from data*, volume 4. AMLBook New York, NY, USA:, 2012.

[2] Nate Lord. What is cyber security? definition, best practices & more, 2019.

[3] Jan van den Berg. Grasping cybersecurity: A set of essential mental models. In *European Conference on Cyber Warfare and Security*, pages 534–XX. Academic Conferences International Limited, 2019.

[4] Jan van den Berg. Cybersecurity for everyone. In *Cybersecurity Best Practices*, pages 571–583. Springer, 2018.

[5] Paul Bischoff. Report: 267 million facebook users ids and phone numbers exposed online, 2019.

[6] Identity Theft Research Center. 2019 end-of-year data breach report, 2019.

[7] Wikipedia. Malware, 2019.

[8] Wikipedia. Social engineering (security), 2019.

[9] Su Sheng, Wang Yingkun, Long Yuyi, Li Yong, and Jiang Yu. Cyber attack impact on power system blackout. *IET Conference on Reliability of Transmission and Distribution Networks (RTDN 2011)*, 2011.

[10] Krishna Sampigethaya and Radha Poovendran. Aviation cyber–physical systems: Foundations for future aircraft and air transport. *Proceedings of the IEEE*, 101(8):1834–1855, 2013.

[11] Clemens Scott Kruse, Benjamin Frederick, Taylor Jacobson, and D Kyle Monticone. Cybersecurity in healthcare: A systematic review of modern threats and trends. *Technology and Health Care*, 25(1):1–10, 2017.

[12] Chee-Wooi Ten, Chen-Ching Liu, and Govindarasu Manimaran. Vulnerability assessment of cybersecurity for scada systems. *IEEE Transactions on Power Systems*, 23(4):1836–1846, 2008.

[13] veiliginternetten.nl. Phishing: laat je niet vangen!, 2020.

[14] Betaalvereniging Nederland. Feiten en cijfes, 2020.

[15] Betaalvereniging Nederland. 3,81 miljoen euro schade door phishing bij internetbankieren in 2018, 2019.

[16] Kelly Bissell, Ryan M Lasalle, and Paolo D Cin. Ninth annual cost of cybercrime study, 2019.

[17] RiskBased Security. 2019 midyear data breach quickview report, 2019.

[18] Steve Morgan. Global cybersecurity spending predicted to exceed $1 trillion from 2017-2021, 2019.

[19] Per Håkon Meland, Karin Bernsmed, Christian Frøystad, Jingyue Li, and Guttorm Sindre. An experimental evaluation of bow-tie analysis for cybersecurity requirements. In *Computer Security*, pages 173–191. Springer, 2018.

[20] Mike Fisk. Data-driven decision making for cyber-security. *Data Science For Cybersecurity*, 3:267, 2018.

[21] Nina Gerber, Paul Gerber, and Melanie Volkamer. Explaining the privacy paradox: A systematic review of literature investigating privacy attitude and behavior. *Computers & Security*, 77:226–261, 2018.

[22] Helen Thackray, John McAlaney, Huseyin Dogan, Jacqui Taylor, and Christopher Richardson. Social psychology: An under-used tool in cybersecurity. In *HCI '16: Proceedings of the 30th International BCS Human Computer Interaction Conference: Companion Volume*. BCS: Chartered Institute for IT, 2016.

[23] Dennis Jackson, Cas Cremers, Katriel Cohn-Gordon, and Ralf Sasse. Seems legit: Automated analysis of subtle attacks on protocols that use signatures. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2165–2180, 2019.

[24] Suzana Andova, Cas Cremers, Kristian Gjøsteen, Sjouke Mauw, Stig F Mjølsnes, and Saša Radomirović. A framework for compositional verification of security protocols. *Information and Computation*, 206(2-4):425–459, 2008.

[25] Thomas Rid. Cyber war will not take place. *Journal of strategic studies*, 35(1):5–32, 2012.

[26] Lucas Kello. *The virtual weapon and international order*. Yale University Press, 2017.

[27] Santiago Quintero-Bonilla and Angel Martín del Rey. Proposed models for advanced persistent threat detection: A review. In *International Symposium on Distributed Computing and Artificial Intelligence*, pages 141–148. Springer, 2019.

[28] Andrew Martin. The ten page introduction to trusted computing. Technical Report RR-08-11, OUCL, December 2008.

[29] Ekta Gandotra, Divya Bansal, and Sanjeev Sofat. Malware analysis and classification: A survey. *Journal of Information Security*, 5(02):56, 2014.

[30] Anastasia Skovoroda and Dennis Gamayunov. Review of the mobile malware detection approaches. In *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 600–603. IEEE, 2015.

[31] Manish Kumar Sahu, Manish Ahirwar, and A Hemlata. A review of malware detection based on pattern matching technique. *Int. J. of Computer Science and Information Technologies (IJCSIT)*, 5(1):944–947, 2014.

[32] V Suganya. A review on phishing attacks and various anti phishing techniques. *International Journal of Computer Applications*, 139(1):20–23, 2016.

[33] Joshua Neil, Curtis Hash, Alexander Brugh, Mike Fisk, and Curtis B Storlie. Scan statistics for the online detection of locally anomalous subgraphs. *Technometrics*, 55(4):403–414, 2013.

[34] Richard J Bolton and David J Hand. Statistical fraud detection: A review. *Statistical science*, pages 235–249, 2002.

[35] Clifton Phua, Vincent Lee, Kate Smith, and Ross Gayler. A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*, 2010.

[36] Imtithal A Saeed, Ali Selamat, and Ali MA Abuagoub. A survey on malware and malware detection systems. *International Journal of Computer Applications*, 67(16), 2013.

[37] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, 2015.

[38] Mohammed Jamil Elhalabi, Selvakumar Manickam, Loai Bani Melhim, Mohammed Anbar, and Huda Alhalabi. A review of peer-to-peer botnet detection techniques. *Journal of Computer Science*, 10(1):169, 2014.

[39] Zahian Ismail and Aman Jantan. A review of machine learning application in botnet detection system. *Sindh University Research Journal-SURJ (Science Series)*, 48(4D), 2016.

[40] Ahmad Karim, Rosli Bin Salleh, Muhammad Shiraz, Syed Adeel Ali Shah, Irfan Awan, and Nor Badrul Anuar. Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University SCIENCE C*, 15(11):943–983, 2014.

[41] Nick Heard, Niall Adams, Patrick Rubin-Delanchy, and Melissa Turcotte. Preface. *Data Science For Cyber-security*, 3:161, 2018.

[42] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[43] Jessemyn Modini, Timothy Lynar, Elena Sitnikova, and Keith Joiner. Applications of epidemiology to cybersecurity. In *European Conference on Cyber Warfare and Security*, pages 483–490. Academic Conferences International Limited, 2020.

[44] Jessemyn Modini, Timothy Lynar, Elena Sitnikova, Keith Joiner, et al. The application of epidemiology for categorising dns cyber risk factors. *Journal of Computer and Communications*, 8(12):12, 2020.

[45] Damjan Fujs, Anže Mihelič, and Simon LR Vrhovec. The power of interpretation: Qualitative methods in cybersecurity research. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pages 1–10, 2019.

[46] Brian Wansink. New techniques to generate key marketing insights. *Marketing Research (Summer 2000)*, pages 28–36, 2000.

[47] Qianjia Huang, Vivek Kumar Singh, and Pradeep Kumar Atrey. Cyber bullying detection using social and textual analysis. In *Proceedings of the 3rd International Workshop on Socially-Aware Multimedia*, pages 3–6. ACM, 2014.

[48] Saeed Nari and Ali A Ghorbani. Automated malware classification based on network behavior. In *2013 International Conference on Computing, Networking and Communications (ICNC)*, pages 642–647. IEEE, 2013.

[49] Patrick Rubin-Delanchy, Niall M Adams, and Nicholas A Heard. Disassortativity of computer networks. In *2016 IEEE conference on intelligence and security informatics (ISI)*, pages 243–247. IEEE, 2016.

[50] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

[51] Mahendra Mariadassou, Stéphane Robin, and Corinne Vacher. Uncovering latent structure in valued graphs: a variational approach. *The Annals of Applied Statistics*, pages 715–742, 2010.

[52] Mohammed Basil Albayati and Ahmad Mousa Altamimi. Identifying fake facebook profiles using data mining techniques. *Journal of ICT Research and Applications*, 13(2):107–117, 2019.

[53] Ronald S Burt. A note on missing network data in the general social survey. *Social Networks*, 9(1):63–73, 1987.

[54] AC Ghani, CA Donnelly, and GP Garnett. Sampling biases and missing data in explorations of sexual partner networks for the spread of sexually transmitted diseases. *Statistics in medicine*, 17(18):2079–2097, 1998.

[55] Stephen P Borgatti and José Luis Molina. Ethical and strategic issues in organizational social network analysis. *The Journal of Applied Behavioral Science*, 39(3):337–349, 2003.

[56] G Kossinets. Effects of missing data in social networks. *Social Networks*, 28(3):247–268, 2006.

[57] Mark Huisman. Imputation of missing network data: some simple procedures. *Journal of Social Structure*, 10(1):1–29, 2009.

[58] Stef Van Buuren. *Flexible imputation of missing data*. CRC press, 2012.

[59] Donald B Rubin. *Multiple imputation for nonresponse in surveys*. John Wiley & Sons, 1987.

[60] Stef Van Buuren, Jaap PL Brand, Catharina GM Groothuis-Oudshoorn, and Donald B Rubin. Fully conditional specification in multivariate imputation. *Journal of Statistical Computation and Simulation*, 76(12):1049–1064, 2006.

[61] Leslie W Hepple. Bayesian techniques in spatial and network econometrics: 2. computational methods and algorithms. *Environment and Planning A*, 27(4):615–644, 1995.

[62] James P LeSage. Bayesian estimation of spatial autoregressive models. *International Regional Science Review*, 20(1-2):113–129, 1997.

[63] Wikipedia. Botnet, 2020.

[64] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82:35–45, 1960.

[65] Peter S Maybeck. *Stochastic models, estimation, and control*. Academic press, 1982.

[66] Simon J Julier and Jeffrey K Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[67] Qingyu Yang, Jie Yang, Wei Yu, Dou An, Nan Zhang, and Wei Zhao. On false data-injection attacks against power system state estimation: Modeling and countermeasures. *IEEE Transactions on Parallel and Distributed Systems*, 25(3):717–729, 2013.

[68] Qingyu Yang, Liguo Chang, and Wei Yu. On false data injection attacks against kalman filtering in power system dynamic state estimation. *Security and Communication Networks*, 9(9):833–849, 2016.

[69] Young Hwan Chang, Qie Hu, and Claire J Tomlin. Secure estimation based kalman filter for cyber–physical systems against sensor attacks. *Automatica*, 95:399–412, 2018.

[70] Martin Strohmeier, Vincent Lenders, and Ivan Martinovic. Security of ads- b: State of the art and beyond, 2013.

[71] Pieter Johan Diederick Drenth and Klaas Sijtsma. *Testtheorie: Inleiding in de theorie van de psychologische test en zijn toepassingen*. Bohn Stafleu Van Loghum, 2005.

[72] ING. Persoonlijke lening bij ing, 2020.

[73] America's Debt Help Organization. Consumer fraud, 2020.

[74] A Blanco, R Rodriguez, I Olabarrieta, I Perez, and I Martinez-Maranon. Comparison of different linear and non-linear techniques in the gender determination of mackerel (scomber scombrus) by colorimetry. *International Journal of Food and Biosystems Engineering*, page 25, 2015.

[75] Richard Wheeler and Stuart Aitken. Multiple algorithms for fraud detection. In *Applications and Innovations in Intelligent Systems VII*, pages 219–231. Springer, 2000.

[76] Bart Baesens, Veronique Van Vlasselaer, and Wouter Verbeke. *Fraud analytics using descriptive, predictive, and social network techniques: a guide to data science for fraud detection*. John Wiley & Sons, 2015.

[77] Keith B Anderson, Erik Durbin, and Michael A Salinger. Identity theft. *Journal of Economic Perspectives*, 22(2):171–192, 2008.

[78] Bimal Parmar. Protecting against spear-phishing. *Computer Fraud & Security*, 2012(1):8–11, 2012.

[79] Anthony Christopher Davison. *Statistical models*, volume 11. Cambridge university press, 2003.

[80] Jean Jacod and Philip Protter. Gaussian random variables (the normal and the multivariate normal distributions). In *Probability Essentials*, pages 125–139. Springer, 2004.

[81] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[82] Grigorios Papageorgiou, Stuart W Grant, Johanna JM Takkenberg, and Mostafa M Mokhles. Statistical primer: how to deal with missing data in scientific research? *Interactive cardiovascular and thoracic surgery*, 27(2):153–158, 2018.

[83] John Barnard and Xiao-Li Meng. Applications of multiple imputation in medical studies: from aids to nhanes. *Statistical methods in medical research*, 8(1):17–36, 1999.

[84] Jason C Cole. How to deal with missing data. *Best practices in quantitative methods*, pages 214–238, 2008.

[85] Joseph L Schafer. Multiple imputation: a primer. *Statistical methods in medical research*, 8(1):3–15, 1999.

[86] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 2. John Wiley & Sons, 2014.

[87] Patricia A Patrician. Multiple imputation for missing data. *Research in nursing & health*, 25(1):76–84, 2002.

[88] Mark A Klebanoff and Stephen R Cole. Use of multiple imputation in the epidemiologic literature. *American journal of epidemiology*, 168(4):355–357, 2008.

[89] Paul D Allison. *Missing data*. Sage publications, 2001.

[90] Roderick JA Little and Donald B Rubin. The analysis of social science data with missing values. *Sociological Methods & Research*, 18(2-3):292–326, 1989.

[91] Donald B Rubin. Multiple imputation after 18+ years. *Journal of the American statistical Association*, 91(434):473–489, 1996.

[92] Jerzy Neyman. On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625, 1934.

[93] James M Robins and Naisyin Wang. Inference for imputation estimators. *Biometrika*, 87(1):113–124, 2000.

[94] Donald B Rubin. Statistical matching using file concatenation with adjusted weights and multiple imputations. *Journal of Business & Economic Statistics*, 4(1):87–94, 1986.

[95] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[96] S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

[97] Ian R White, Rhian Daniel, and Patrick Royston. Avoiding bias due to perfect prediction in multiple imputation of incomplete categorical variables. *Computational statistics & data analysis*, 54(10):2267–2275, 2010.

[98] James H Albert and Siddhartha Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

[99] Tiago P Peixoto. Bayesian stochastic blockmodeling. *Advances in network clustering and blockmodeling*, pages 289–332, 2019.

[100] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[101] Ben Lambert. *A student's guide to Bayesian statistics*. Sage, 2018.

[102] Roger Grosse and Nitish Srivastava. Lecture 16: Mixture models, September 2020.

[103] Bengt Muthen. Latent variable mixture modeling. *New developments and techniques in structural equation modeling*, 2:1–33, 2001.

[104] Gitta Lubke. Latent variable mixture models. *The reviewer's guide to quantitative methods in the social sciences*, page 209, 2010.

[105] David M Blei. Build, compute, critique, repeat: Data analysis with latent variable models. *Annual Review of Statistics and Its Application*, 2014.

[106] Daniel Oberski. Mixture models: Latent profile and latent class analysis. In *Modern statistical methods for HCI*, pages 275–287. Springer, 2016.

[107] Paul D McNicholas. *Mixture model-based classification*. CRC press, 2016.

[108] Chuong B Do and Serafim Batzoglou. What is the expectation maximization algorithm? *Nature biotechnology*, 26(8):897–899, 2008.

[109] Katherine Faust and Stanley Wasserman. Blockmodels: Interpretation and evaluation. *Social networks*, 14(1-2):5–61, 1992.

[110] PW Holland and S Leinhardt. An exponential family of probability densities for directed graphs. *Unpublished manuscript*, 1979.

[111] Stephen E Fienberg and Stanley S Wasserman. Categorical data analysis of single sociometric relations. *Sociological methodology*, 12:156–192, 1981.

[112] Harrison C White, Scott A Boorman, and Ronald L Breiger. Social structure from multiple networks. i. blockmodels of roles and positions. *American journal of sociology*, 81(4):730–780, 1976.

[113] Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical review E*, 83(1):016107, 2011.

[114] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 9(Sep):1981–2014, 2008.

[115] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[116] Carolyn J Anderson, Stanley Wasserman, and Katherine Faust. Building stochastic blockmodels. *Social networks*, 14(1-2):137–161, 1992.

[117] Stephen E Fienberg, Michael M Meyer, and Stanley S Wasserman. Statistical analysis of multiple sociometric relations. *Journal of the american Statistical association*, 80(389):51–67, 1985.

[118] Krzysztof Nowicki and Tom A B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association*, 96(455):1077–1087, 2001.

[119] J-J Daudin, Franck Picard, and Stéphane Robin. A mixture model for random graphs. *Statistics and computing*, 18(2):173–183, 2008.

[120] Yi-An Ma, Yuansi Chen, Chi Jin, Nicolas Flammarion, and Michael I Jordan. Sampling can be faster than optimization. *Proceedings of the National Academy of Sciences*, 116(42):20881–20885, 2019.

[121] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[122] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological Methods*, 7(2):147, 2002.

[123] Ofer Harel and Xiao-Hua Zhou. Multiple imputation: review of theory, implementation and software. *Statistics in medicine*, 26(16):3057–3077, 2007.

[124] Panteha Hayati Rezvan, Katherine J Lee, and Julie A Simpson. The rise of multiple imputation: a review of the reporting and implementation of the method in medical research. *BMC medical research methodology*, 15(1):30, 2015.

[125] Mark Huisman and Christian Steglich. Treatment of non-response in longitudinal network studies. *Social networks*, 30(4):297–308, 2008.

[126] MS Hancock and K Gile. Modeling social networks with sampled or missing data. *Seattle: Center for Statistics and the Social Sciences, University of Washington*, 2007.

[127] Dino Dittrich, Roger Th AJ Leenders, and Joris Mulder. Bayesian estimation of the network autocorrelation model. *Social Networks*, 48:213–236, 2017.

[128] Johan H Koskinen, Garry L Robins, Peng Wang, and Philippa E Pattison. Bayesian analysis for partially observed network data, missing ties, attributes and actors. *Social Networks*, 35(4):514–527, 2013.

[129] Kayla de la Haye, Joshua Embree, Marc Punkay, Dorothy L Espelage, Joan S Tucker, and Harold D Green Jr. Analytic strategies for longitudinal networks with missing data. *Social networks*, 50:17–25, 2017.

[130] John R Hipp, Cheng Wang, Carter T Butts, Rupa Jose, and Cynthia M Lakon. Research note: The consequences of different methods for handling missing network data in stochastic actor based models. *Social networks*, 41:56–71, 2015.

[131] Tom AB Snijders. Statistical models for social networks. *Annual Review of Sociology*, 37, 2011.

[132] Ove Frank and David Strauss. Markov graphs. *Journal of the American Statistical Association*, 81(395):832–842, 1986.

[133] Stanley Wasserman and Philippa Pattison. Logit models and logistic regressions for social networks: I. an introduction to markov graphs andp. *Psychometrika*, 61(3):401–425, 1996.

[134] Paul W Holland and Samuel Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association*, 76(373):33–50, 1981.

[135] Aaron Zimmerman, Tyler McCormick, Ali Shojaie, and Hedwig Lee. Improving attribute prediction through network-augmented attribute prediction, 2015.

[136] Krista J Gile and Mark S Handcock. Analysis of networks with missing data with application to the national longitudinal study of adolescent health. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 66(3):501–519, 2017.

[137] Keith Ord. Estimation methods for models of spatial interaction. *Journal of the American Statistical Association*, 70(349):120–126, 1975.

[138] Tony E Smith and James P LeSage. A bayesian probit model with spatial dependencies. In *Spatial and Spatiotemporal Econometrics*, pages 127–160. Emerald Group Publishing Limited, 2004.

[139] Luc Anselin. *Spatial Econometrics: methods and models*, volume 4. Springer Science & Business Media, 2013.

[140] Martyn Plummer. Cuts in bayesian graphical models. *Statistics and Computing*, 25(1):37–43, 2015.

[141] David Lunn, Nicky Best, David Spiegelhalter, Gordon Graham, and Beat Neuenschwander. Combining mcmc with 'sequential'pkpd modelling. *Journal of Pharmacokinetics and Pharmacodynamics*, 36(1):19, 2009.

[142] Pierre E Jacob, Lawrence M Murray, Chris C Holmes, and Christian P Robert. Better together? statistical learning in models made of modules. *arXiv preprint arXiv:1708.08719*, 2017.

[143] Andrew Gelman. Parameterization and bayesian modeling. *Journal of the American Statistical Association*, 99(466):537–545, 2004.

[144] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the Association for Information Science and Technology*, 60(5):911–932, 2009.

[145] James LeSage. Applied econometrics using matlab. *Manuscript, Dept. of Economics, University of Toronto*, pages 154–159, 1999.

[146] James P LeSage, R Kelley Pace, Nina Lam, Richard Campanella, and Xingjian Liu. New orleans business recovery in the aftermath of hurricane katrina. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 174(4):1007–1027, 2011.

[147] James P LeSage and R K Pace. *Introduction to Spacial Econometrics*. New York: Taylor & Francis Group: Chapman & Hall, 2009.

[148] Stefan Wilhelm and Miguel Godinho de Matos. Estimating spatial probit models in r. *R Journal*, 5(1), 2013.

[149] R Kelley Pace and Ronald Barry. Quick computation of spatial autoregressive estimators. *Geographical Analysis*, 29(3):232–247, 1997.

[150] R Kelley Pace and James P LeSage. Chebyshev approximation of log-determinants of spatial weight matrices. *Computational Statistics & Data Analysis*, 45(2):179–196, 2004.

[151] Mark S Mizruchi and Eric J Neuman. The effect of density on the level of bias in the network autocorrelation model. *Social Networks*, 30(3):190–200, 2008.

[152] Eric J Neuman and Mark S Mizruchi. Structure and bias in the network autocorrelation model. *Social Networks*, 32(4):290–300, 2010.

[153] Tony E Smith. Estimation bias in spatial models with strongly connected weight matrices. *Geographical Analysis*, 41(3):307–332, 2009.

[154] Garth Holloway, Bhavani Shankar, and Sanzidur Rahmanb. Bayesian spatial probit estimation: a primer and an application to hyv rice adoption. *Agricultural Economics*, 27(3):383–402, 2002.

[155] Davide Martinetti and Ghislain Geniaux. Probitspatial r package: Fast and accurate spatial probit estimations. In *22. International Conference on Computational Statistics (COMPSTAT)*, page np, 2016.

[156] Dootika Vats, James M Flegal, and Galin L Jones. Multivariate output analysis for markov chain monte carlo. *arXiv preprint arXiv:1512.07713*, 2015.

[157] Frederick Sanders. On subjective probability forecasting. *Journal of Applied Meteorology*, 2(2):191–201, 1963.

[158] David B Stephenson, Caio AS Coelho, and Ian T Jolliffe. Two extra components in the brier score decomposition. *Weather and Forecasting*, 23(4):752–757, 2008.

[159] Peter Grünwald, Thijs Van Ommen, et al. Inconsistency of bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4):1069–1103, 2017.

[160] Domonique W Hodge, Sandra E Safo, and Qi Long. Multiple imputation using dimension reduction techniques for high-dimensional data. *arXiv preprint arXiv:1905.05274*, 2019.

[161] Jignesh Vania, Arvind Meniya, and HB Jethva. A review on botnet and detection technique. *International Journal of Computer Trends and Technology*, 4(1):23–29, 2013.

[162] Wentao Chang, Aziz Mohaisen, An Wang, and Songqing Chen. Measuring botnets in the wild: Some new trends. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 645–650. ACM, 2015.

[163] Thomas S Hyslip and Jason M Pittman. A survey of botnet detection techniques by command and control infrastructure. *Journal of Digital Forensics, Security and Law*, 10(1):2, 2015.

[164] Ahmad Karim, Rosli Salleh, and Muhammad Khurram Khan. Smartbot: A behavioral analysis framework augmented with machine learning to identify mobile botnet applications. *PloS one*, 11(3):e0150077, 2016.

[165] Matija Stevanovic and Jens Myrup Pedersen. Machine learning for identifying botnet network traffic. *Journal of Cyber Security and Mobility*, 4(2):1–32, 2016.

[166] Elaheh Biglar Beigi, Hossein Hadian Jazi, Natalia Stakhanova, and Ali A Ghorbani. Towards effective feature selection in machine learning-based botnet detection approaches. In *Communications and Network Security (CNS), 2014 IEEE Conference on*, pages 247–255. IEEE, 2014.

[167] Pijush Barthakur, Manoj Dahal, and Mrinal Kanti Ghose. An efficient machine learning based classification scheme for detecting distributed command & control traffic of p2p botnets. *International Journal of Modern Education and Computer Science*, 5(10):9, 2013.

[168] Pavani Bharathula and N Mridula Menon. Equitable machine learning algorithms to probe over p2p botnets. In *Proceedings of the 4th International Conference on Frontiers in Intelligent Computing: Theory and Applications (FICTA) 2015*, pages 13–21. Springer, 2016.

[169] Leyla Bilge, Davide Balzarotti, William Robertson, Engin Kirda, and Christopher Kruegel. Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In *Proceedings of the 28th Annual Computer Security Applications Conference*, pages 129–138. ACM, 2012.

[170] Livadas Carl et al. Using machine learning techniques to identify botnet traffic. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on. IEEE*, 2006.

[171] Ali Feizollah, Nor Badrul Anuar, Rosli Salleh, Fairuz Amalina, Shahaboddin Shamshirband, et al. A study of machine learning classifiers for anomaly-based mobile botnet detection. *Malaysian Journal of Computer Science*, 26(4):251–265, 2013.

[172] Fariba Haddadi, Jillian Morgan, Eduardo Gomes Filho, and A Nur Zincir-Heywood. Botnet behaviour analysis using ip flows: with http filters using classifiers. In *Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on*, pages 7–12. IEEE, 2014.

[173] Fariba Haddadi and A Nur Zincir-Heywood. Botnet detection system analysis on the effect of botnet evolution and feature representation. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 893–900. ACM, 2015.

[174] Xin Meng and George Spanoudakis. Mbotcs: A mobile botnet detection system based on machine learning. In *International Conference on Risks and Security of Internet and Systems*, pages 274–291. Springer, 2015.

[175] Kamaldeep Singh, Sharath Chandra Guntuku, Abhishek Thakur, and Chittaranjan Hota. Big data analytics framework for peer-to-peer botnet detection using random forests. *Information Sciences*, 278:488–497, 2014.

[176] Sajjad Arshad, Maghsoud Abbaspour, Mehdi Kharrazi, and Hooman Sanatkar. An anomaly-based botnet detection approach for identifying stealthy botnets. In *Computer Applications and Industrial Electronics (ICCAIE), 2011 IEEE International Conference on*, pages 564–569. IEEE, 2011.

[177] Naveen Davis. Botnet detection using correlated anomalies. *Technical University of Denmark Informatics and Mathematical Modelling*, 2012.

[178] Pedro Camelo, Joao Moura, and Ludwig Krippahl. Condenser: A graph-based approachfor detecting botnets. *arXiv preprint arXiv:1410.8747*, 2014.

[179] Nhauo Davuth and K Sung-Ryul. Classification of malicious domain names using support vector machine and bi-gram method. *International Journal of Security and Its Applications*, 7(1):51–58, 2013.

[180] António Nogueira, Paulo Salvador, and Fábio Blessa. A botnet detection system based on neural networks. In *Digital Telecommunications (ICDT), 2010 Fifth International Conference on*, pages 57–62. IEEE, 2010.

[181] Sharath Chandra Guntuku, Pratik Narang, and Chittaranjan Hota. Real-time peer-to-peer botnet detection framework based on bayesian regularized neural network. *arXiv preprint arXiv:1307.7464*, 2013.

[182] Fatih Haltaş, Erkam Uzun, Necati Şişeci, Abdulkadir Poşul, and Bakır Emre. An automated bot detection system through honeypots for large-scale. In *Cyber Conflict (CyCon 2014), 2014 6th International Conference On*, pages 255–270. IEEE, 2014.

[183] Florian Tegeler, Xiaoming Fu, Giovanni Vigna, and Christopher Kruegel. Botfinder: Finding bots in network traffic without deep packet inspection. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, pages 349–360. ACM, 2012.

[184] Christian Rossow, Dennis Andriesse, Tillmann Werner, Brett Stone-Gross, Daniel Plohmann, Christian J Dietrich, and Herbert Bos. Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 97–111. IEEE, 2013.

[185] Christian Steglich, Tom AB Snijders, and Michael Pearson. Dynamic networks and behavior: Separating selection from influence. *Sociological methodology*, 40(1):329–393, 2010.

[186] Pieter Burghouwt, Marcel Spruit, and Henk Sips. Detection of botnet collusion by degree distribution of domains. In *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pages 1–8. IEEE, 2010.

[187] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. Big data behavioral analytics meet graph theory: On effective botnet takedowns. *IEEE Network*, 31(1):18–26, 2017.

[188] Emmanuel Abbe. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

[189] Sherif Saad, Issa Traore, Ali Ghorbani, Bassam Sayed, David Zhao, Wei Lu, John Felix, and Payman Hakimian. Detecting p2p botnets through network behavior analysis and machine learning. In *Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on*, pages 174–180. IEEE, 2011.

[190] Yuchung J Wang and George Y Wong. Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82(397):8–19, 1987.

[191] Tom AB Snijders and Krzysztof Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of classification*, 14(1):75–100, 1997.

[192] Jean-Benoist Leger. Blockmodels: A r-package for estimating in latent block model and stochastic block model, with various probability functions, with or without covariates. *arXiv preprint arXiv:1602.07587*, 2016.

[193] Timothée Tabouy, Pierre Barbillon, and Julien Chiquet. misssbm: An r package for handling missing values in the stochastic block model. *arXiv preprint arXiv:1906.12201*, 2019.

[194] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[195] Thorben Funke and Till Becker. Stochastic block models: A comparison of variants and inference methods. *PloS one*, 14(4):e0215296, 2019.

[196] Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE transactions on pattern analysis and machine intelligence*, 22(7):719–725, 2000.

[197] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[198] Amit Saxena, Mukesh Prasad, Akshansh Gupta, Neha Bharill, Om Prakash Patel, Aruna Tiwari, Meng Joo Er, Weiping Ding, and Chin-Teng Lin. A review of clustering techniques and developments. *Neurocomputing*, 267:664–681, 2017.

[199] Vinay Kumar, Sanjay B Dhok, Rajeev Tripathi, and Sudarshan Tiwari. A review study of hierarchical clustering algorithms for wireless sensor networks. *International Journal of Computer Science Issues (IJCSI)*, 11(3):92, 2014.

[200] Priyanka Tavse and Anil Khandelwal. A critical review on data clustering in wireless network. *International Journal of Advanced Computer Research*, 4(3):795, 2014.

[201] Rui Xu and Donald C Wunsch. Clustering algorithms in biomedical research: a review. *IEEE reviews in biomedical engineering*, 3:120–154, 2010.

[202] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[203] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[204] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[205] Naoki Masuda and Petter Holme. Detecting sequences of system states in temporal networks. *Scientific reports*, 9(1):1–11, 2019.

[206] Tony Jung and Kandauda AS Wickrama. An introduction to latent class growth analysis and growth mixture modeling. *Social and personality psychology compass*, 2(1):302–317, 2008.

[207] Vassilis Kostakos. Temporal graphs. *Physica A: Statistical Mechanics and its Applications*, 388(6):1007–1023, 2009.

[208] Tom AB Snijders. Stochastic actor-oriented models for network dynamics. *Annual Review of Statistics and Its Application*, 4:343–363, 2017.

[209] Mark S Handcock, Carter T Butts, David R Hunter, Steven M Goodreau, Skye Bender de Moll, Pavel N Krivitsky, and Martina Morris. Temporal exponential random graph models (tergms) for dynamic network modeling in statnet. *Sunbelt 2015*, 2015.

[210] Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In *Advances in Neural Information Processing Systems*, pages 3765–3773, 2016.

[211] Lianne Ippel, Maurits Clemens Kaptein, and Jeroen K Vermunt. Estimating random-intercept models on data streams. *Computational Statistics & Data Analysis*, 104:169–182, 2016.

[212] Zikuan Liu, Jalal Almhana, Vartan Choulakian, and Robert McGorman. Online em algorithm for mixture with application to internet traffic modeling. *Computational statistics & data analysis*, 50(4):1052–1071, 2006.

[213] Azqa Nadeem, Christian Hammerschmidt, Carlos H Gañán, and Sicco Verwer. Malpaca: Malware packet sequence clustering and analysis. *arXiv preprint arXiv:1904.01371*, 2019.

[214] Elchanan Mossel, Joe Neeman, and Allan Sly. Stochastic block models and reconstruction. *arXiv preprint arXiv:1202.1499*, 2012.

[215] Clement Lee and Darren J Wilkinson. A review of stochastic block models and extensions for graph clustering. *arXiv preprint arXiv:1903.00114*, 2019.

[216] Babak Rahbarinia, Roberto Perdisci, Andrea Lanzi, and Kang Li. Peerrush: Mining for unwanted p2p traffic. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 62–82. Springer, 2013.

[217] Sofiane Lagraa, Jérôme François, Abdelkader Lahmadi, Marine Miner, Christian Hammerschmidt, and Radu State. Botgm: Unsupervised graph mining to detect botnets in traffic flows. In *2017 1st Cyber Security in Networking Conference (CSNet)*, pages 1–8. IEEE, 2017.

[218] M Patrick Collins and Michael K Reiter. Hit-list worm detection and bot identification in large networks using protocol graphs. In *International Workshop on Recent Advances in Intrusion Detection*, pages 276–295. Springer, 2007.

[219] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *17th USENIX Security Symposium*, pages 139–154. USENIX, 2008.

[220] W Timothy Strayer, David Lapsely, Robert Walsh, and Carl Livadas. Botnet detection based on network behavior. In *Botnet detection*, pages 1–24. Springer, 2008.

[221] Feng Liu, Zhitang Li, and Qingbin Nie. A new method of p2p traffic identification based on support vector machine at the host level. In *2009 International Conference on Information Technology and Computer Science*, volume 2, pages 579–582. IEEE, 2009.

[222] Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov. Botgrep: Finding p2p bots with structured graph analysis. In *USENIX security symposium*, volume 10, pages 95–110, 2010.

[223] Wen-Hwa Liao and Chia-Ching Chang. Peer to peer botnet detection using data mining scheme. In *2010 International Conference on Internet Technology and Applications*, pages 1–4. IEEE, 2010.

[224] Junjie Zhang, Roberto Perdisci, Wenke Lee, Unum Sarfraz, and Xiapu Luo. Detecting stealthy p2p botnets using statistical traffic fingerprints. In *2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN)*, pages 121–132. IEEE, 2011.

[225] David Zhao, Issa Traore, Ali Ghorbani, Bassam Sayed, Sherif Saad, and Wei Lu. Peer to peer botnet detection based on flow intervals. In *IFIP International Information Security Conference*, pages 87–102. Springer, 2012.

[226] Pijush Barthakur, Manoj Dahal, and Mrinal Kanti Ghose. A framework for p2p botnet detection using svm. In *2012 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pages 195–200. IEEE, 2012.

[227] Shree Garg, Ankush K Singh, Anil K Sarje, and Sateesh K Peddoju. Behaviour analysis of machine learning algorithms for detecting p2p botnets. In *2013 15th International Conference on Advanced Computing Technologies (ICACT)*, pages 1–4. IEEE, 2013.

[228] Kazumasa Yamauchi, Yoshiaki Hori, and Kouichi Sakurai. Detecting http-based botnet based on characteristic of the c & c session using by svm. In *2013 Eigth Asia Joint Conference on Information Security*, pages 63–68. IEEE, 2013.

[229] Pratik Narang, Jagan Mohan Reddy, and Chittaranjan Hota. Feature selection for detection of peer-to-peer botnet traffic. In *Proceedings of the 6th ACM India computing convention*, pages 1–9, 2013.

[230] Hongling Jiang and Xiuli Shao. Detecting p2p botnets by discovering flow dependency in c&c traffic. *Peer-to-Peer Networking and Applications*, 7(4):320–331, 2014.

[231] Muhammad N Sakib and Chin-Tser Huang. Using anomaly detection based techniques to detect http-based botnet c&c traffic. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[232] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. Blinc: multilevel traffic classification in the dark. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 229–240, 2005.

[233] Mahbod Tavallaee, Wei Lu, and Ali A Ghorbani. Online classification of network flows. In *2009 Seventh Annual Communication Networks and Services Research Conference*, pages 78–85. IEEE, 2009.

[234] Guofei Gu, Junjie Zhang, and Wenke Lee. Botsniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium.*, pages 1–18. Wright, 2008.

[235] Mark Patrick Roeling and Geoff Nicholls. Stochastic block models as an unsupervised approach to detect botnet-infected clusters in networked data. *Data Science For Cyber-security*, 3:161, 2018.

[236] Baris Coskun, Sven Dietrich, and Nasir Memon. Friends of an enemy: identifying local members of peer-to-peer botnets using mutual contacts. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 131–140, 2010.

[237] Tao Cai and Futai Zou. Detecting http botnet with clustering network traffic. In *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–7. IEEE, 2012.

[238] Chun-Yu Wang, Chi-Lung Ou, Yu-En Zhang, Feng-Min Cho, Pin-Hao Chen, Jyh-Biau Chang, and Ce-Kuen Shieh. Botcluster: A session-based p2p botnet clustering system on netflow. *Computer Networks*, 145:175–189, 2018.

[239] John PA Ioannidis. Why most published research findings are false. *PLos med*, 2(8):e124, 2005.

[240] Pablo Torres, Carlos Catania, Sebastian Garcia, and Carlos Garcia Garino. An analysis of recurrent neural networks for botnet detection behavior. In *2016 IEEE biennial congress of Argentina (ARGENCON)*, pages 1–6. IEEE, 2016.

[241] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[242] Toni Giorgino et al. Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, 31(7):1–24, 2009.

[243] Géza Szabó, Dániel Orincsay, Szabolcs Malomsoky, and István Szabó. On the validation of traffic classification algorithms. In *International Conference on Passive and Active Network Measurement*, pages 72–81. Springer, 2008.

[244] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. An empirical comparison of botnet detection methods. *computers & security*, 45:100–123, 2014.

[245] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[246] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

[247] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

[248] Stephan Mandt and David Blei. Smoothed gradients for stochastic variational inference. In *Advances in Neural Information Processing Systems*, pages 2438–2446, 2014.

[249] Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

[250] Sudipta Chowdhury, Mojtaba Khanzadeh, Ravi Akula, Fangyan Zhang, Song Zhang, Hugh Medal, Mohammad Marufuzzaman, and Linkan Bian. Botnet detection using graph-based feature clustering. *Journal of Big Data*, 4(1):14, 2017.

[251] Jing Wang and Ioannis Ch Paschalidis. Botnet detection based on anomaly and community detection. *IEEE Transactions on Control of Network Systems*, 4(2):392–404, 2016.

[252] Orestis Kostakis, Nikolaj Tatti, and Aristides Gionis. Discovering recurring activity in temporal networks. *Data Mining and Knowledge Discovery*, 31(6):1840–1871, 2017.

[253] Wenzhe Li, Sungjin Ahn, and Max Welling. Scalable mcmc for mixed membership stochastic blockmodels. In *Artificial Intelligence and Statistics*, pages 723–731, 2016.

[254] Ismail El-Helw, Rutger Hofman, Wenzhe Li, Sungjin Ahn, Max Welling, and Henri Bal. Scalable overlapping community detection. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 1463–1472. IEEE, 2016.

[255] Soumyasundar Pal and Mark Coates. Scalable mcmc in degree corrected stochastic block model. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5461–5465. IEEE, 2019.

[256] Mohinder S Grewal and Angus P Andrews. Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]. *IEEE Control Systems Magazine*, 30(3):69–78, 2010.

[257] Marco Barreno, Blaine Nelson, Anthony D Joseph, and J Doug Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[258] Eric Chan-Tin, Daniel Feldman, Nicholas Hopper, and Yongdae Kim. The frog-boiling attack: Limitations of anomaly detection for secure network coordinate systems. In *International Conference on Security and Privacy in Communication Systems*, pages 448–458. Springer, 2009.

[259] Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 641–647, 2005.

[260] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58, 2011.

[261] Yilin Mo and Bruno Sinopoli. False data injection attacks in control systems. In *Preprints of the 1st workshop on Secure Control Systems*, pages 1–6, 2010.

[262] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[263] Rakesh B Bobba, Katherine M Rogers, Qiyan Wang, Himanshu Khurana, Klara Nahrstedt, and Thomas J Overbye. Detecting false data injection attacks on dc state estimation. In *Preprints of the First Workshop on Secure Control Systems, CPSWEEK*, volume 2010, 2010.

[264] Oliver Kosut, Liyan Jia, Robert J Thomas, and Lang Tong. Malicious data attacks on the smart grid. *IEEE Transactions on Smart Grid*, 2(4):645–658, 2011.

[265] Fabio Pasqualetti, Ruggero Carli, and Francesco Bullo. A distributed method for state estimation and false data detection in power networks. In *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 469–474. IEEE, 2011.

[266] Tung T Kim and H Vincent Poor. Strategic protection against data injection attacks on power grids. *IEEE Transactions on Smart Grid*, 2(2):326–333, 2011.

[267] Brad Hianes. Defcon 20: Hacker + airplanes = no good can come of this. https://www.youtube.com/watch?v=CXv1j3GbgLk, 2012. Online; accessed June 2016.

[268] Greg Dunstone. Ads-b in a radar environment. https://www.icao.int/APAC/Meetings/2014%20ADSBSITF13/SP06_AUS%20-%20ADSB%20in%20radar%20environments.pdf, 2014.

[269] Martin Strohmeier, Ivan Martinovic, Markus Fuchs, Matthias Schäfer, and Vincent Lenders. Opensky: A swiss army knife for air traffic security research. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 4A1–1. IEEE, 2015.

[270] Roger Labbe. Kalman and bayesian filters in python. *Chap*, 7:246, 2014.

[271] Thaddeus Vincenty. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey review*, 23(176):88–93, 1975.

[272] Robert J Hijmans, Ed Williams, Chris Vennes, and Maintainer Robert J Hijmans. Package 'geosphere'. *Spherical trigonometry*, 1:7, 2019.

[273] Jan Wendel, Christian Schlaile, and Gert F Trommer. Direct kalman filtering of gps/ins for aerospace applications. In *International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation (KIS2001)*, 2001.

[274] Nina PG Salau, Jorge O Trierweiler, Argimiro R Secchi, and Wolfgang Marquardt. A new process noise covariance matrix tuning algorithm for kalman based state estimators. *IFAC Proceedings Volumes*, 42(11):572–577, 2009.

[275] Murali R Rajamani. *Data-based techniques to improve state estimation in model predictive control*. PhD thesis, Citeseer, 2007.

[276] Sai Dheeraj Nadella. Use of extended kalman filter in estimation of attitude of a nano-satellite. *International Journal of Electronics and Electrical Engineering*, 3(1), 2015.

[277] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter, 1995.

[278] Paul Gilbert, Maintainer Paul Gilbert, and Ravi Varadhan. The numderiv package, 2006.

[279] Dah-Jing Jwo, Mu-Yen Chen, Chien-Hao Tseng, and Ta-Shun Cho. Adaptive and nonlinear kalman filtering for gps navigation processing. *Kalman Filter: Recent Advances and Applications*, 19, 2009.

[280] S Romaniuk and Z Gosiewski. Kalman filter realization for oriental and position estimation on dedicated processor. *Retrieved online 12th June from http://www. actawm. pb. edu. pl/volume/vol8no2/06_2014_004_ROMANIUK_GOSIEWSKI. pdf*, 2014.

[281] David F Bizup and Donald E Brown. The over extended kalman filter- don't use it! In *Proceedings of the Sixth International Conference of Information Fusion*, volume 1, pages 40–46. Citeseer, 2003.

[282] Rune Jansberg. Tracking of an airplane using ekf and spf. Master's thesis, University of Oslo, 2010.

[283] Bruce Schneier. *Applied cryptography: protocols, algorithms, and source code in C*. john wiley & sons, 2007.

[284] Véronique Van Vlasselaer, Tina Eliassi-Rad, Leman Akoglu, Monique Snoeck, and Bart Baesens. Gotcha! network-based fraud detection for social security fraud. *Management Science*, 63(9):3090–3110, 2016.

[285] Anuj Sharma and Prabin Kumar Panigrahi. A review of financial accounting fraud detection based on data mining techniques. *arXiv preprint arXiv:1309.3944*, 2013.

[286] Stef Van Buuren. *Flexible imputation of missing data*. Chapman and Hall/CRC, 2018.

[287] Ioannis Kosmidis. On iterative adjustment of responses for the reduction of bias in binary regression models, 2009.

[288] Philip K Chan, Wei Fan, Andreas L Prodromidis, and Salvatore J Stolfo. Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems and Their Applications*, 14(6):67–74, 1999.

[289] Kuldeep Kumar and Sukanto Bhattacharya. Artificial neural network vs linear discriminant analysis in credit ratings forecast: A comparative study of prediction performances. *Review of Accounting and Finance*, 5(3):216–227, 2006.

[290] Deon Garrett, David A Peterson, Charles W Anderson, and Michael H Thaut. Comparison of linear, nonlinear, and feature selection methods for eeg signal classification. *IEEE Transactions on neural systems and rehabilitation engineering*, 11(2):141–144, 2003.

[291] Therese D Pigott. A review of methods for missing data. *Educational research and evaluation*, 7(4):353–383, 2001.

[292] Jared S Murray et al. Multiple imputation: A review of practical and theoretical findings. *Statistical Science*, 33(2):142–159, 2018.

[293] Laurence G Grimm. *Statistical applications for the behavioral sciences*. Wiley, 1993.

[294] Riccardo Russo. *Statistics for the behavioural sciences: an introduction*. Psychology Press, 2004.

[295] Robert W Krause, Mark Huisman, Christian Steglich, and Tom AB Sniiders. Missing network data a comparison of different imputation methods. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 159–163. IEEE, 2018.

[296] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[297] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[298] Amalia Luque, Alejandro Carrasco, Alejandro Martín, and Ana de las Heras. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, 91:216–231, 2019.

[299] Hossein Rouhani Zeidanloo, Farhoud Hosseinpour, and Farhood Farid Etemad. New approach for detection of irc and p2pbotnets. *International Journal of Computer and Electrical Engineering*, 2(6):1029, 2010.

[300] Padmini Jaikumar and Avinash C Kak. A graph-theoretic framework for isolating botnets in a network. *Security and communication networks*, 8(16):2605–2623, 2015.

[301] Wernhuar Tarng, Li-Zhong Den, Kuo-Liang Ou, and Mingteh Chen. The analysis and identification of p2p botnet's traffic flows. *International Journal of Communication Networks and Information Security*, 3(2):138, 2011.

[302] Rita M Cantor, Kenneth Lange, and Janet S Sinsheimer. Prioritizing gwas results: a review of statistical methods and recommendations for their application. *The American Journal of Human Genetics*, 86(1):6–22, 2010.

[303] Johannes Moser. Hypothetical thinking and the winner's curse: An experimental investigation. *Theory and Decision*, 87(1):17–56, 2019.

[304] Robert W Krause, Mark Huisman, Christian Steglich, and Tom Snijders. Missing data in cross-sectional networks–an extensive comparison of missing data treatment methods. *Social Networks*, 62:99–112, 2020.

[305] Won Seob Oh, Cong Hieu Nguyen, Sang Min Kim, Jung Woo Sohn, and Joon Heo. Spatial autocorrelation of disease prevalence in south korea using 2012 community health survey data. *Journal of the Korean Society of Surveying, Geodesy, Photogrammetry and Cartography*, 34(3):253–262, 2016.

[306] Barry C Arnold, Enrique Castillo, and Jose-Maria Sarabia Alegria. *Conditionally specified distributions*, volume 73. Springer Science & Business Media, 2012.

[307] Indranil Ghosh and N Balakrishnan. On compatibility/incompatibility of two discrete probability distributions in the presence of incomplete specification. *arXiv preprint arXiv:1909.08447*, 2019.

[308] Masayuki Karasuyama and Hiroshi Mamitsuka. Adaptive edge weighting for graph-based learning algorithms. *Machine Learning*, 106(2):307–335, 2017.

[309] Steve Harenberg, Gonzalo Bello, La Gjeltema, Stephen Ranshous, Jitendra Harlalka, Ramona Seay, Kanchana Padmanabhan, and Nagiza Samatova. Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439, 2014.

SUMMARY

This thesis presents several methodological and statistical solutions to problems encountered in cyber security. We investigated the effects of compromised data veracity in state estimators and fraud detection systems, a model to impute missing data in attributes of linked observations, and an unsupervised approach to detect infected machines in a computer network.

The first part of this thesis deals with imputation of missing data and the clustering of nodes in a computer network. Chapter 3 considers the handling of incomplete data in the case where the observations are linked, such as in a network. We applied the framework of autocorrelation regression models to estimates missing values in covariates in data from a messaging app, where graduate students were able to send each other messages. Parameter estimation used Bayesian statistics, combining Gibbs sampling (regression coefficients) and Metropolis Hasting (network parameters) sampling. We compared a model where feedback from imputed observations was allowed against a model where this feedback was cut, to observe a higher imputation accuracy in the cut model.

Also, this thesis addressed the problem of unsupervised clustering of streaming data in chapters 4 and 5, represented by a capture of network activity from computer networks including a number of machines that were infected with malware. By modelling the time sequence with features that were able to capture the temporal element of the data (via dynamic time warping) and the distance between ports used in the connection were were able to recover the infection status of machines with excellent classification performance. The pipeline we proposed did not rely on filtering or manually manipulating the data prior to analyses.

Part two investigates the influence of data injection and the veracity of provided data. State estimation is important in many (cyber) physical system to control parameters that often direct mechanical processes. One prominent example is the Kalman Filter that predicts the real state of a dynamical model given a measurement, such as the position of an airplane in flight. In chapter 6 we showed how the prediction of a Kalman Filter can be manipulated by inserting fake data, under different scenario's. Even with boundaries that govern the amount of variation allowed in the measurements, we can still influence the filter with manipulated updates to diverge location estimates.

In chapter 7 we investigated unexpected responses given by bank-clients on a consumer credit application, and observed that in a selection of covariates (to which a response was given) the residual (distance between observed and expected responses) was informative to fraud status. This indicated there are some questions in the online application form that are more likely to receive an unexpected value, and that a combination of these unexpected values (residuals) can predict fraud status.

Hence, this thesis applied and developed techniques to understand fundamental problems often encountered when analysing data in the cyber security setting, and ex-

tend to many other scenario's. They have been implemented, evaluated, and applied to several real-world cases to demonstrate the effectiveness and applicability.

SAMENVATTING

Dit proefschrift presenteert verschillende methodologische en statistische oplossingen voor problemen die zich voordoen in het context van cyber veiligheid. Het onderzoek heeft zich gericht op het schatten van ontbrekende gegevens in attributen / variabelen in observaties die op een manier samenhangen (zoals in een netwerk), een model om zonder de aanwezigheid van labels geïnfecteerde machines in een computernetwerk te detecteren, en de gevolgen van het injecteren van verkeerde data in Kalman Filters en fraudedetectie systemen.

Het eerste deel van dit proefschrift gaat over imputatie van missing data en clustering van nodes in een computer netwerk. Hoofdstuk 3 betreft het omgaan met missende gegevens in het geval dat de waarnemingen aan elkaar zijn gekoppeld, zoals in een netwerk. Statistische methoden voor het schatten van missende gegevens zijn voornamelijk gericht op data waarin de observaties (bijvoorbeeld personen die een vragenlijst invullen) niet met elkaar samenhangen; een random steekproef zijn. In ons onderzoek hebben we regressiemodellen toegepast die om kunnen gaan met autocorrelatie, om ontbrekende waarden in variabelen te schatten, met data uit een berichten-app waar studenten elkaar berichten konden sturen. Voor het schatten van parameters werd gebruik gemaakt van Bayesiaanse statistiek; een combinatie van Gibbs-sampling (voor regressiecoëfficiënten) en Metropolis-Hasting (voor netwerkparameters). We vergeleken een model waarin feedback van geïmputeerde data was toegestaan met een model waarin deze feedback werd afgebroken, en observeerden een hogere nauwkeurigheid in het model zonder feedback.

Daarnaast behandelde dit proefschrift het probleem van het clusteren van streaming data zonder de aanwezigheid van een label wat door de computer kan worden gebruikt ter voorbeeld (hoofdstukken 4 en 5). Dit onderzoek werd uitgevoerd met een aantal captures van netwerkactiviteit van computernetwerken, met daarin een aantal machines die waren geïnfecteerd met malware. Het model transformeert de stream in een afstandsschatting (via dynamische time-warping) en de afstand tussen de poorten die in de verbinding werden gebruikt. De features die hierdoor ontstaan, in combinatie met de netwerk structuur, kunnen worden gebruikt om de infectiestatus van machines te voorspellen met uitstekende classificatieprestaties. De door ons voorgestelde pijplijn is niet afhankelijk van het filteren of handmatig manipuleren van de data voorafgaand aan de analyse.

Deel twee onderzoekt de gevolgen van data injectie en de waarheidsgetrouwheid van aangeleverde data. Kalman filters worden gebruik om de toestand / staat te schatten aan de hand van een aantal parameters die worden gemeten. Deze toestand-schatting is in veel (cyber) fysieke systemen belangrijk om veranderende mechanische processen aan te sturen. Zoals bijvoorbeeld het schatten van de locatie van een vliegtuig aan de hand van richting, snelheid, hoogte, en radar afstand, waarbij rekening wordt gehouden met de meetfout van de meet-instrumenten. In hoofdstuk 6 hebben we laten zien hoe de

voorspelling van een Kalman-filter kan worden gemanipuleerd door verkeerde data te injecteren tijdens en tussen meetmomenten, onder verschillende scenario's. Zelfs als er grenzen worden gebruikt die bepalen hoeveel variatie er in de metingen mag optreden, kunnen we het filter beïnvloeden gemanipuleerde updates om locatieschattingen te laten afwijken.

In hoofdstuk 7 onderzochten we onverwachte antwoorden van cliënten van een grote bank op een aanvraagformulier voor een consumptief krediet, en stelden vast dat in een selectie van variabelen (waarop een antwoord werd gegeven) het residu (afstand tussen het gegeven antwoord en het verwachte antwoord) informatief was voor de fraudestatus. Dit betekent dat er enkele vragen in het aanvraagformulier zijn waarbij er een grotere kans is op een onverwachte antwoord, en dat een combinatie van deze onverwachte antwoorden (residuen) kan worden gebruikt in het bepalen welke aanvragen waarschijnlijk frauduleus zijn.

Dit proefschrift heeft technieken toegepast en ontwikkeld om fundamentele problemen te begrijpen die vaak voorkomen bij het analyseren van data in de cybersecurity context. Ze zijn geïmplementeerd, geëvalueerd en toegepast op verschillende scenario's uit de praktijk om de effectiviteit en toepasbaarheid aan te tonen.

ACKNOWLEDGEMENTS

My academic work started during my undergraduate studies and I thank Alexander Waringa, Andreas Wismeijer, Paul Hodiamont, Ivan Nyklicek and Karel Oei for their support and supervision. I am forever thankful to Dorret Boomsma: thank you for facilitating my academic work and inspiring me to conduct my own research. I am thankful to the department of Biological Psychology for their friendship, continuing long after I left. I thank Marleen de Moor, Gonneke Willemsen, Gwen Dieleman, Tinca Polderman, and Henning Tiemeier for supervising and inspiring me during my graduate studies in Amsterdam and Rotterdam. I thank my immediate Erasmus MC colleagues Dr. Dekker, Dr. Duvekot, Prof. Dr. El Marroun, Dr. De la Bois, Dr. Schoemaker, Dr. Stapersma, stb. Dr. De Neve-Enthoven, stb. Dr. Gerritsen, and Dr. De Lijster for their ongoing kindness, understanding, and friendship. Most of you completed your PhD before me, willing me on, and although most of my Erasmus MC work will remain unacknowledged, working with you enriched my life.

I thank the Centre of Doctoral Training in Cybersecurity (CDT) for the opportunity to conduct DPhil-research at Oxford. Geoff, thank you for supervising my statistical work at Oxford, and supporting me when I went through a rough time. It was very inspiring to experience, even via Skype, how you tackled complex statistical procedures and instantly proposed improvements to make the work stronger. Credits to my CDT-friends Jacqueline, Jantje, Aaron, Dennis, John, Nick, Daniel, and Olu for the great times in Oxford, scientific discussions, and many lessons how to be British. Likewise, I thank my Delft-colleagues Azqa and Christian for the interesting discussions and their contributions to my work.

I thank Jan and Sicco for their supervision during my time in Delft. Jan, we had a short but very interesting time, where your encyclopedic knowledge of the field helped to put the work into perspective. Sicco, thanks for the daily supervision and making me feel part of your group. Every month we basically thought up a new study and that was very inspiring! Inald, thanks for your input in this project and help to take care of the more formal part of the project.

I thank ING Bank Netherlands for hosting a combined project; Wim, thanks for your supervision and feedback! Many thanks to the Netherlands Ministries of Defence and Finance for facilitating me to combine a full-time position with completing a PhD. For assisting me with the formalities and management of this project I am grateful to David Hobbs, Maureen York, Gesine Reinert, Sandra Wolff, Wolfson College Oxford, and the EEMCS graduate school.

To my parents and brother; thanks for your love and support during this academic “endeavour”. I conclude by thanking Evie and Tom for being the paranymphs who assisted with the final preparations and ceremonial aspects of the defence, as another token of their love, support, humour, and (statistical) input.

CURRICULUM VITÆ

Mark Patrick ROELING

1985	Born in Rhenen, the Netherlands.

EDUCATION

1997–2005	Willem van Oranje College, Waalwijk (1997-2002) Koninklijk Technisch Atheneum Maaseik (2002–2003) Koninklijk Atheneum Maaseik (2003–2005)
2010	BSc Psychology, Tilburg University
2011	MSc Behavior Genetics (Neuroscience), VU University Amsterdam
2013	MSc Genetic Epidemiology, Erasmus MC Rotterdam
2021	PhD Cybersecurity and statistics University of Oxford, MPLS, Department of Statistics Delft University of Technology, EEMCS, Intelligent Systems, Cybersecurity

PROFESSIONAL

2011-2014	Junior researcher Child and Adolescent Psychiatry, Erasmus MC Rotterdam Centre of Neurogenomics and Cognitive Research, VU University Amsterdam
2014-2015	Data Scientist, Capgemini (Netherlands)
2015-2017	Full time PhD-student, University of Oxford (United Kingdom)
2018-2020	Data Scientist, Ministry of Defence (Netherlands)
2020-2021	Senior Data Scientist, Ministry of Finance (Netherlands)

PORTFOLIO

PhD Training	Year
Centre of Doctoral Training in Cybersecurity, Oxford	
Core curriculum	
Security Architectures and Information Defence	2015
Cross Disciplinary Research Methods	2015
(Cyber) International Relations	2015

Human Factors in Cyber Security	2015
Ethics and Policies of Cyber Security	2015
Malware	2016
Cyber Risk	2016
High Integrity Systems	2016
Digital Forensics	2016
Criminology	2016
Skills courses	
Presentation Skills for Cybersecurity	2016
How to make a Conference poster?	2016
Short courses	2016
Offensive Computing	2016
Quantum Computing	2016
Data Visualisation	2016
Faculty of Electrical Engineering, Mathematics and Computer Science – EEMCS, TU Delft	
PhD Start-Up module A	2019

Teaching	Year
Supervising BSc and MSc students	
R. Claassen (Erasmus MC, MSc Medicine): *Heterogeneity within attention problems, hyperactivity/impulsivity symptoms and cognition in a childhood population*	2013
L. Sauer (VU Amsterdam, BSc Biomedical Sciences) *Autonomic nervous system and Hypothalamic-pituitary-adrenal axis functioning in the onset of the anxiety disorders in children.*	2013
N. Halve (VU Amsterdam, BSc Biomedical Sciences) *Hypothalamic-pituitary-adrenal-axis and autonomic nervous system functioning of childhood-onset anxiety disorders.*	2013
R. van der Lans (Leiden, MSc Child & Adolescent Psychology) *Examining the Influence of Children's and Teachers'Characteristics on the Teacher's Report Form*	2013
A. van Rijn (Leiden, MSc Child & Adolescent Psychology) *The association between autism spectrum disorders and depression*	2014
Other	
Lecture, SURFsara, Amsterdam Science Park, computational power in	

genomic analyses	2013
Lecture, Erasmus University Rotterdam; heritability and behavior genetics	2014
Lecture, VU Amsterdam: genetics of schizophrenia	2014
TA, Applied Statistics (SB1.1), BA in Mathematics, with Dr. N Laws	2016

List of Publications

- Roeling, M.P., & Nicholls, G.K. (2020). Imputation of attributes in networked data using Bayesian Autocorrelation Regression Models. *Social Networks (62)*, 24-32.
- Silomon, J.A.M., & Roeling, M.P. (2018). Assessing Opinions on Software as a Weapon in the Context of (Inter)national Security. *Transactions on Computational Science XXXII, Special Issue on Cybersecurity and Biometrics*, 43-56.
- Roeling, M.P., Willemsen, G., & Boomsma, D.I. (2016). Heritability of working in a creative profession: a Dutch twin siblings study. *Behavior Genetics, 47*, 298-304.
- Wind, A., Roeling, M.P., Heerink, J., Sixma, H., Presti, P., Lombardo, C., & van Harten, W. (2016). Piloting a Generic Cancer Consumer Quality Index in six European countries. *BMC Cancer, 16(1):711*.
- Roeling, M.P. (2016). False data injection in Kalman Filters in an aerospace setting; ADS-B data with simulated noise. *Oxford Research Archive; 2016-06-23*.
- Distel, M.A., Roeling, M.P., van Tielbeek, J., van Toor, D., Derom, C., Trull, T., & Boomsma, D.I. (2012). The covariation of trait anger and borderline personality: A bivariate twin-siblings study. *Journal of Abnormal Psychology, 121(2)*, 458-66.
- de Moor, M.H.M., Roeling, M.P., & Boomsma, D.I. (2011). Creativity and talent: Etiology of familial clustering. In Vartanian, A Bristol, & JC Kaufman (Eds.), *The Neuroscience of Creativity*. New York: Cambridge University Press.
- Roeling, M.P. (2010). Functioning is the cornerstone of life; diagnostic methods to assess the chronic impairment level of clinical disorders on social functioning. *Journal of European Psychology Students, Vol 2*.
- Roeling, M.P., Polderman, T.E.C., & Boomsma, D.I. (2010). Tweelingstudies en intelligentie. *Blind, 24*.

Conferences:

- Roeling, M.P., Nadeem, A., & Verwer, S.E. (2021). Hybrid connection and host clustering for community Detection in spatial-temporal network data. European Conference on Machine Learning, Ghent, Belgium. *Communications in Computer and Information Science (Vol. 1323)*.
- Roeling, M.P., & Nicholls, G.K. (2017). Stochastic Blockmodels as an unsupervised approach to detect botnet infected clusters in networked data. In N Heard, N Adams, P Rubin-Delanchy, M Turcotte (Eds.), *Proceedings of Data Science for Cyber-Security, London, United Kingdom*.

- Roeling, M.P., & Nicholls. G.K. (2016). Detection of abnormal data provided by fraudsters: a new approach to using residuals from regression models to improve fraud classification. *FINE annual conference: The Hague, the Netherlands.*
- Roeling, M.P., & Nicholls. G.K. (2016). Introducing Machine Learning at the foundation of your data sets: Ascertaining the validity of consumer data to increase the accuracy of fraud detection. *Worldwide Business Research: Data Insight Leaders Summit, Barcelona, Spain.*
- Roeling, M.P., Dieleman, G., de Leeuw, C., Goudriaan, A., Polderman, T.J.C., Verhage, M., Smit, A., Verheijen, M., Verhulst, F., & Posthuma, D. (2014). A functional genomics approach to understand genetic variation in (non)neuronal cell types underlying autism spectrum disorders. *World Congress of Psychiatric Genetics, Copenhagen, Denmark.*
- Roeling, M.P., Polderman, T.J.C., van der Ende, J., Lubke, G., Verhulst, F.C., Posthuma, D., & Dieleman, G. (2013). The use of hybrid statistical models to chart the heterogeneity of internalizing problem behavior in child- and adolescent psychiatric patients, Amsterdam, the Netherlands. *Tijdschrift voor kindergeneeskunde.*
- Roeling, M.P. (2013). Computational Power: closer to the truth. *Life Science e-Infrastructure Workshop, SURFsara, SciencePark Amsterdam, the Netherlands.*

Posters:

- Nadeem, A., Roeling, M.P., & Verwer, S.E. (2019). A hybrid approach to allow unsupervised clustering of both connections and hosts in networked data. *Annual international cybersecurity and AI conference, Prague.*
- Roeling, M.P., & Nicholls. G.K. (2016). Lie detection in big data: Comparing residuals from regression analyses to detect deceptive scores in 219.810 credit applications. *CDT Cybersecurity Oxford, Showcase.*

Acknowledgements:

- de Lijster, J., van den Dries, M.A., van der Ende, J., Utens, E.M.W.J., Jaddoe, V.W., Dieleman, G.C., ... Legerstee, J.S. (2019). Developmental trajectories of anxiety and depression symptoms from early to middle childhood: a population-based cohort study in the Netherlands. *Journal of Abnormal Child Psychology.*
- Silomon J. (2017). Attitudes towards software as a weapon. *European Security Conference: Cybersecurity Analytics*, Lisbon.
- van der Sluis, S., Polderman, T.E.C., Neale, M.C., Verhulst, F.C., Posthuma, D., Dieleman, G. (2016). Sex differences and gender-invariance of mother-reported childhood problem behavior. *International Journal of Methods in Psychiatric Research.*
- Polderman, T.E.C., Benyamin, B., de Leeuw, C., Sullivan, P.F., Bochoven, A., Visscher, P.M., Posthuma, D. (2015). Meta-analysis of the heritability of human traits based on fifty years of twin studies. *Nature Genetics.*