DGK Veröffentlichungen der DGK

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften

Reihe C | Dissertationen | Heft Nr. 942

Anna Malinovskaya

Statistical Process Monitoring of Networks

München 2024

Bayerische Akademie der Wissenschaften

ISSN 0065-5325 ISBN 978-3-7696-5354-0

Diese Arbeit ist gleichzeitig veröffentlicht in:
Wissenschaftliche Arbeiten der Fachrichtung Geodäsie und Geoinformatik der Leibniz Universität Hannover
ISSN 0174-1454, Nr. 399, Hannover 2024

DGK Veröffentlichungen der DGK

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften

Reihe C Dissertationen Heft Nr. 942

Statistical Process Monitoring of Networks

Von der Fakultät für Bauingenieurwesen und Geodäsie
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation

von

Anna Malinovskaya, B. Sc.

München 2024

Bayerische Akademie der Wissenschaften

ISSN 0065-5325 ISBN 978-3-7696-5354-0

Diese Arbeit ist gleichzeitig veröffentlicht in:
Wissenschaftliche Arbeiten der Fachrichtung Geodäsie und Geoinformatik der Leibniz Universität Hannover
ISSN 0174-1454, Nr. 399, Hannover 2024

Adresse der DGK:

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften (DGK)

Alfons-Goppel-Straße 11 ● D – 80 539 München

Telefon +49 - 331 - 6264 1685 ● E-Mail post@dgk.badw.de

http://www.dgk.badw.de

Prüfungskommission:

Vorsitzender: Prof. Dr.-Ing. Ingo Neumann

Referent: Prof. Dr. Philipp Otto

Korreferenten: apl. Prof. Dr. techn. Franz Rottensteiner
Prof. Dr. Christian H. Weiß (Helmut-Schmidt-Universität Hamburg)

Tag der mündlichen Prüfung: 11.04.2024

ISSN 0065-5325 ISBN 978-3-7696-5354-0

Abstract

The digital information revolution offers rich opportunities for scientific progress; however, the amount and variety of data available require new analysis techniques in order to adequately address the growing complexity of processes. These requirements have influenced the development of networks and integrated their application in various disciplines.

This thesis addresses the detection of changes in networks, combining network theory and statistical process monitoring to create improved techniques for *network monitoring*. Considering networks as *graph-structured data* with either fixed or dynamic nodes and edges or as a *model based on artificial intelligence*, three directions of network monitoring are identified, namely, *random network monitoring* where the networks represent random variables, *fixed network monitoring* where the networks are assumed to be fixed structures, and *monitoring of artificial neural networks*. The idea of using different modelling techniques and control charts to monitor network-related processes connects contributions in this thesis to the outlined monitoring directions.

The first developed approach shows how multivariate control charts can be used to detect changes in dynamic networks of various types generated by the temporal exponential random graph model in an online manner. This monitoring procedure allows for many applications in different disciplines which are interested in analysing networks of medium size.

Next, the monitoring framework to detect anomalies in the network with a given structure but a random process on it by combining the generalised network autoregressive model with node-specific time series exogenous variables and the cumulative sum control chart based on residuals is presented. This approach can be of particular interest for guaranteeing the safety of the infrastructure but also for foreseeing possible accidents.

The third contribution is dedicated to the development of a monitoring procedure for artificial neural network applications that applies a non-parametric multivariate control chart based on ranks and data depths. The core idea is to monitor a low-dimensional representation of input data called *embeddings* that are generated by artificial neural networks to detect nonstationarity in a processed data stream.

In addition to the development of three monitoring approaches, a fourth contribution, namely the extension from the pure detection of the change point to the identification of its cause is presented. The investigation includes a proposal for an automated inspection procedure, bringing together a control chart for quantile function values and a graph convolutional network.

Keywords: Artificial Neural Networks, Change Point Detection, Control Charts, Machine Learning on Graphs, Processes on Networks, Statistical Network Modelling, Statistical Network Monitoring, Statistical Process Monitoring.

Kurzfassung

Die digitale Informationsrevolution bietet reichhaltige Möglichkeiten für den wissenschaftlichen Fortschritt; die Menge und Vielfalt der verfügbaren Daten erfordert jedoch neue Analysetechniken, um die wachsende Komplexität der Prozesse angemessen darstellen zu können. Diese Anforderungen haben die Entwicklung von Netzwerken beeinflusst und ihre Anwendung in verschiedene Disziplinen integriert.

Diese Arbeit befasst sich mit der Erkennung von Änderungen in Netzwerken, wobei die Netzwerktheorie und die statistische Prozessüberwachung kombiniert werden, um verbesserte Techniken für die *Netzwerküberwachung* zu schaffen. Betrachtet man Netzwerke als g*raphenstrukturierte Daten* mit entweder festen oder dynamischen Knoten und Kanten oder als *ein auf künstlicher Intelligenz basierendes Modell*, können *drei Richtungen der Netzwerküberwachung* identifiziert werden, nämlich die Netzwerküberwachung, bei der die Netzwerke Zufallsvariablen darstellen, die Netzwerküberwachung, bei der die Netzwerke als feste Strukturen angenommen sind, und die Überwachung künstlicher neuronaler Netzwerke. Die Idee, verschiedene Modellierungstechniken und Kontrollkarten aus der statistischen Prozessüberwachung zur Überwachung der netzbezogenen Prozesse zu verwenden, verbindet die Beiträge zu den skizzierten Überwachungsrichtungen.

Der erste entwickelte Ansatz zeigt, wie multivariate Kontrollkarten verwendet werden können, um Veränderungen in dynamischen Netzwerken verschiedener Arten, die durch das zeitliche exponentielle Zufallsgraphenmodell erzeugt werden, in Echtzeit zu erkennen. Dieses Überwachungsverfahren ermöglicht zahlreiche Anwendungen in verschiedenen Disziplinen, die an der Analyse von Netzwerken mittlerer Größe interessiert sind.

Als nächstes wird der Überwachungsansatz zur Erkennung von Anomalien in einem Netzwerk mit einer festen Struktur und einem Zufallsprozess auf seinen Kanten durch die Kombination des verallgemeinerten autoregressiven Netzwerkmodells mit knotenspezifischen exogenen Zeitreihenvariablen und der auf Residuen basierenden kumulativen Summenkontrollkarte vorgestellt. Dieses Verfahren kann von besonderem Interesse für die Gewährleistung der Infrastruktursicherheit, aber auch für die Vorhersage möglicher Ausfälle sein.

Der dritte Beitrag widmet sich der Entwicklung eines Überwachungsverfahrens für künstliche neuronale Netze, das eine nichtparametrische multivariate Kontrollkarte auf der Grundlage von Rängen und Datentiefen anwendet. Die Kernidee besteht darin, die niedrigdimensionale Repräsentation der Eingabedaten, sogenannte *embeddings*, zu überwachen, die von künstlichen neuronalen Netzen erzeugt werden, um Nicht-Stationarität in einem Datenfluss zu erkennen.

Neben der Entwicklung von drei Überwachungsansätzen wird als vierter Beitrag die Erweiterung von der reinen Detektion des Änderungspunktes zur Identifikation seiner Ursache vorgestellt. Die Untersuchung umfasst einen Vorschlag für ein automatisiertes Inspektionsverfahren, das eine Kontrollkarte für Quantilsfunktionswerte und ein künstliches neuronales Netz für Graphen zusammenführt.

Schlagworte: Künstliche neuronale Netze, Erkennung von Änderungspunkten, Kontrollkarten, Maschinelles Lernen auf Graphen, Prozesse auf Netzen, Statistische Netzmodellierung, Statistische Netzüberwachung, Statistische Prozessüberwachung.

Contents

List of Abbreviations

A	
ACF	Autocorrelation Function
AI	Artificial Intelligence
ANN	Artificial Neural Network
ARL	Average Run Length

C	
CBPF	Cross-Border Physical Flow
CDR	Correct Detection Rate
CED	Conditional Expected Delay
CL	Central Line
CNN	Convolutional Neural Network
CUSUM	Cumulative Sum

E	
ENTSO-E	European Network of Transmission System Operators for Electricity
ERGM	Exponential Random Graph Model
EWMA	Exponentially Weighted Moving Average

F	
FAR	False Alarm Rate
FNN	Feedforward Neural Network

G	
GCN	Graph Convolutional Network
GNAR	Generalised Network Autoregressive Model
GNARX	GNAR model with node-specific time series exogenous variables
GRL	Graph Representation Learning
GWESP	Geometrically-Weighted Edgewise Shared Partnerships

H	
HD	Halfspace Depth

I	
iForest	Isolation Forest

K	
KDEOS	Kernel Density Estimation Outlier Score

L

LCL	Lower Control Limit
LOF	Local Outlier Factor
LSTM	Long Short-Term Memory

M

MCMC	Markov Chain Monte Carlo
MCUSUM	Multivariate Cumulative Sum
MD	Mahalanobis Depth
MDis	Mahalanobis Distance
MEWMA	Multivariate Exponentially Weighted Moving Average
ML	Machine Learning
MPLE	Maximum Pseudolikelihood Estimation
MR	Moving Range

N

NAR	Network Autoregressive Model
NLP	Natural Language Processing
NOF	Natural Outlier Factor

P

PD	Projection Depth

R

Radviz	Radial Coordinate Visualisation
ReLU	Rectified Linear Unit

S

SBM	Stochastic Block Model
SD	Simplicial Depth
SR	Signal Rate
SPM	Statistical Process Monitoring
SVM	Support Vector Machine

T

TEN	Temporal Edge Network
TERGM	Temporal Exponential Random Graph Model

U

UCL	Upper Control Limit

V

VAR	Vector Autoregressive Model

1 Introduction

Network representation is fascinating. It conveys complexity by introducing a relational structure between objects, facilitating the integration of diverse information. Its origins trace back to the 18th century when Leonhard Euler solved the Königsberg bridge problem. Since that milestone, network science has developed into a substantial field of research. The broad interest of the statistical community in graph-based data analysis arose in the last century with the development of theory on random graphs and respective models such as the Erdős-Rényi model (cf. Erdős and Rényi, 1959) introducing a probabilistic view of the problem.

Another significant factor in the growing popularity of networks is the availability of extensive data sources. The present era of big data provides a unique opportunity to gain remarkable insight into molecular, social, economic, and many other systems (cf. O'Malley and Marsden, 2008; Schweitzer et al., 2009; Grennan et al., 2014). To be precise, the focus lies in analysing a relational structure between entities, *e.g.* an economic network with entities being banks, a communication network consisting of users or a chemical reaction network having chemical elements as entities (cf. De Masi and Gallegati, 2007; Khrabrov and Cybenko, 2010; McDermott et al., 2021). An example of a relation between two entities could be an occurrence of a transaction or a fixed, physically defined connection.

Besides the rise of big data, the rapid advancement of Artificial Intelligence (AI) has encouraged practitioners in various scientific fields to examine the benefits and challenges of Machine Learning (ML) algorithms in their research. For instance, in astronomy, the separation of astrophysical objects such as stars and galaxies can be performed by applying decision trees (Ball et al., 2006); reliable forecasts of energy consumption using support vector machines are extensively studied in civil engineering (Gao et al., 2019); in biology, Artificial Neural Networks (ANNs) are applied for predicting molecular traits (Angermueller et al., 2016). Together with the emergence of big data and the increased capability of computers, the development of ANNs has received particularly broad attention (cf. Chiroma et al., 2018). Current ANNs, consisting of many hidden layers and being called deep learning models, have achieved impressive results in many areas, such as finance, linguistics, or photogrammetry (cf. Aldridge and Avellaneda, 2019; Otter et al., 2020; Heipke and Rottensteiner, 2020).

From the highlighted trends related to big data and AI at least two perspectives on defining a network can be derived: *network* being either a *graph-structured data* or an *AI-based model*. The additional third stream can be identified in current research trends, namely graphical models, where the network structure is used to represent conditional dependencies between random variables (Schweinberger et al., 2021). In this thesis, the focus is on the first two points of view, being *network as graph-structured data* and *network as AI-based model*, which are summarised in Figure 1.1.

An important problem in studying network data is the detection of anomalous behaviour. Also, a frequent deployment of models based on ANNs reveals the necessity of the models' control and monitoring, as currently the attention to the occurrence of critical flaws is often diminished. Considering the first point of view on networks, *i.e.* graph-structured or network data, one can have networks with either a fixed structure (for example, an infrastructural network such as an

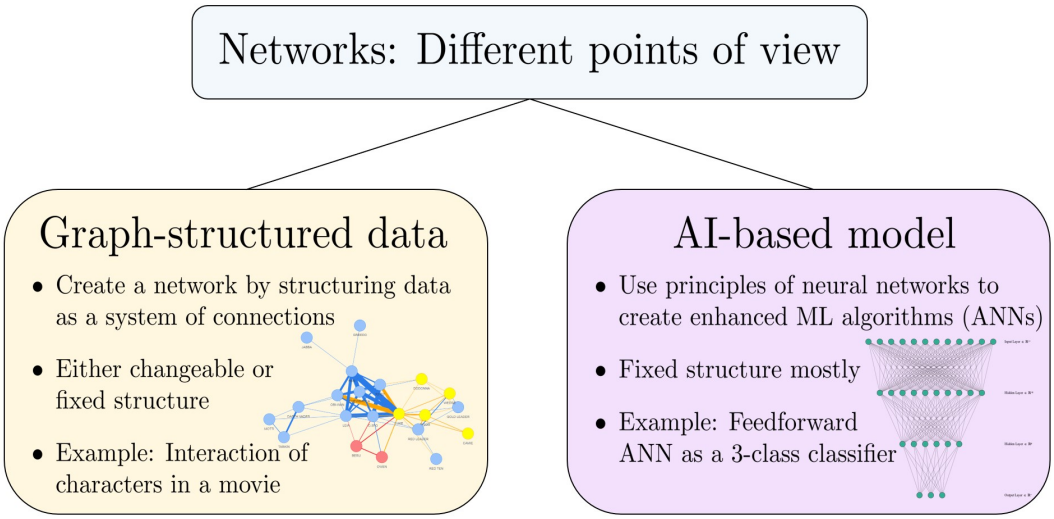


Figure 1.1: Network perspectives explored in this thesis.

electrical transmission system or a road network) or a changeable, random structure (an e-mail network within an institution). Hence, the methods for detecting the deviations in the networks would differ due to the differences in their origin. Considering a network to be an AI-based model, its application also requires an alternative approach to perform monitoring for detecting possible changes.

Thinking about change detection from a statistical angle, there is a well-established concept known as *Statistical Process Monitoring* (SPM) whose original aim was to monitor and control industrial processes, ensuring their operation within specified requirements and compliance with quality standards (Stoumbos et al., 2000). However, among the new streams in expanding the usage of SPM tools, the field of *statistical process monitoring of networks* or in short *network monitoring* has arisen (Woodall and Montgomery, 2014). Statistical network monitoring is defined as a form of surveillance procedure to detect deviations from a so-called in-control state of a network, *i.e.* the state when no unaccountable variation of the network is present (Stevens et al., 2021b).

Currently, one usually distinguishes between two types of network monitoring in the statistical community, differentiating the treatment of nodes and links, namely *random network monitoring* and *fixed network monitoring* (cf. Jeske et al., 2018; Leitch et al., 2019). However, in this thesis, a framework for monitoring applications based on ANNs is additionally proposed, separating it into a further monitoring direction called *monitoring of ANNs*. Thus, in Figure 1.2 three monitoring directions discussed in this thesis are shortly described, with a corresponding indication of a chapter where the contribution to a particular area is presented.

Classical SPM methods can be remarkably powerful for the surveillance of networks. However, due to the complex structure and potentially large size of the network data or also of the data stream processed by ANNs, traditional tools for multivariate process monitoring cannot directly be applied, as the data complexity must be reduced first. Moreover, the central assumption about independent and identically distributed observations or consideration of a specific distribution is also violated when one works with networks (cf. Stevens et al., 2021a). However, there are solutions to overcome these limitations. For instance, this can be done by statistical modelling of the network data or the data stream passed through ANNs. The choice of the model is crucial as

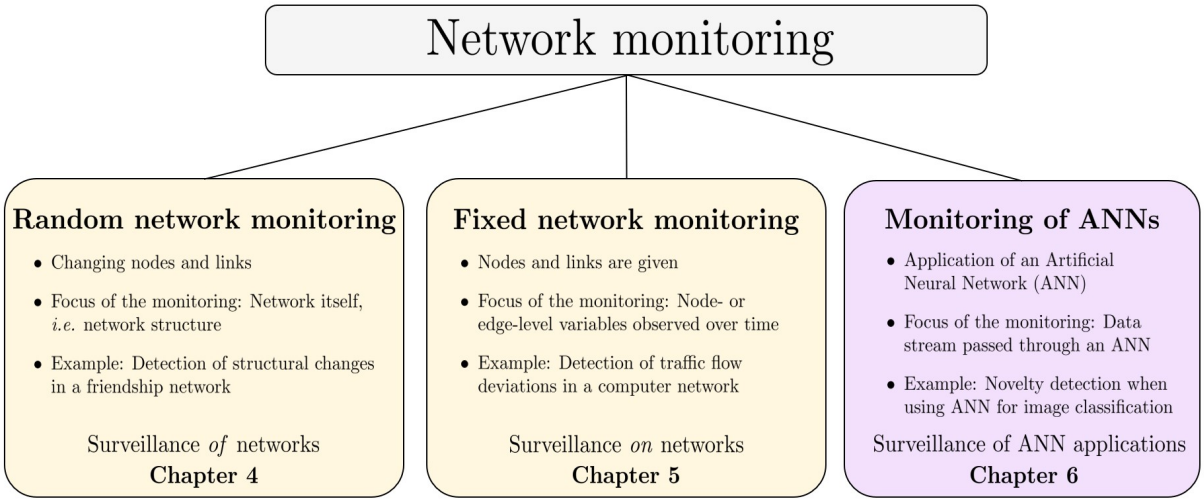


Figure 1.2: Network monitoring directions discussed in this thesis.

it determines the constraints and simplifications of the original process which later influence the types of changes one is able to detect.

Although there are substantial differences in viewing the three types of network monitoring illustrated in Figure 1.2, the main stages in the developed monitoring concepts remain consistent. In each of the proposed frameworks, the same technique from the ensemble of SPM tools is applied, namely *control charts*. Their universality and efficiency in monitoring the process online, meaning in real time, offer technical advantages as well as enable a clear-cut interpretation.

The purpose of applying control charts in network monitoring is to detect the time point when a significant change from the target state of a network has occurred. However, it is often the case that a monitoring statistic is aggregated from several observations which were collected within a specific time frame. That means, if the change point is identified, it is possible that only a few samples were anomalous and the rest not. To be precise about which data points truly deviate from the target state, one would need to inspect the entire batch. Still, the question of a possible cause that has led to a change remains, often leading to a manual inspection. Consequently, it becomes important to retrieve the *control* part being originally the central component of the SPM (cf. Montgomery, 2009) into statistical network monitoring. Hence, this thesis also raises the topic of auxiliary procedures that automatically identify potential causes and anomalous observations after a control chart has identified a change in a network process.

The outline of the thesis is as follows: It begins with the revision of central theoretical foundations in Chapter 2 and proceeds with the presentation of the related research as well as the identification of scientific lacuna in Chapter 3. Afterwards, the contribution to *random network monitoring* in Chapter 4 is presented, being published in Malinovskaya and Otto (2021). It is followed by the method related to *fixed network monitoring* in Chapter 5 which is issued in Malinovskaya et al. (2023a). Then, the focus is changed from networks as graph-structured data to networks being a type of AI-based models, so that the thesis proceeds with presenting a framework for *monitoring of ANNs* in Chapter 6 which is published in Malinovskaya et al. (2023b). The last technical Chapter 7 considers the proposal of an auxiliary procedure after a change point was detected during network monitoring which is based on the publication Malinovskaya et al. (2022). Chapter 8 summarises scientific directions related to the achieved outcomes that need future investigation.

2 Theoretical Foundations

According to Figure 1.1, one encounters networks from two major perspectives in this thesis: Graph-structured data and AI-based models. Thus, in the subsequent Sections 2.1.1 and 2.1.2, the basics of both kinds of networks are introduced. Afterwards, the foundations of network monitoring, especially the definitions of change point detection and control charts are provided in Sections 2.2.1 and 2.2.2, respectively.

2.1 Representation of Networks

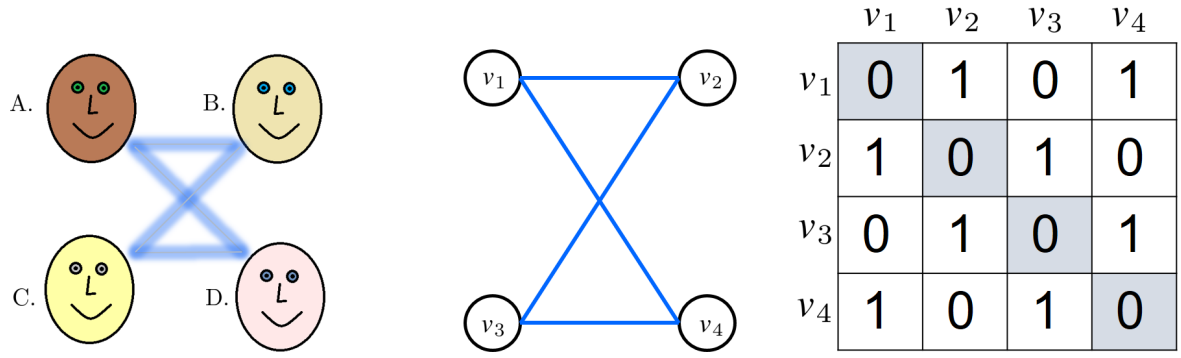
In this section, the mathematical notation to describe networks is accompanied by the preliminary theory of their modelling, separating networks with random structure from networks with fixed structure. Next, the basics of artificial neural networks are given. In general, to distinguish between a network as graph-structured data and the neural network approach, a full name or the abbreviation for the latter is used, *e.g. artificial neural network* or *ANN*.

2.1.1 Networks as Graph-Structured Data

Regarding networks as a structure of data, such representation is strongly shaped by the discipline of discrete mathematics called *Graph Theory* (cf. Diestel, 2017). A *graph* (interchangeably called *network*) $G = (V,E)$ is defined by a set of *nodes* (also known as *vertices*) $v_i \in V$, where $i = 1,\dots,|V|$ with $|V|$ representing the total number of nodes, and a set of *edges* $e_{i,j} \in E$ with $e_{i,j}$ being an edge (also called *link* or *tie*) between vertices v_i and v_j, $j \neq i$. Usually, the network is defined by a binary or weighted *adjacency matrix* $\boldsymbol{Y} \in \mathbb{R}^{|V|\times|V|}$. Two vertices are adjacent if they are connected by an edge. If we consider a binary adjacency matrix, then $Y_{ij} = 1$, otherwise, $Y_{ij} = 0$. In the case of an undirected network, $\boldsymbol{Y}$ is symmetric.

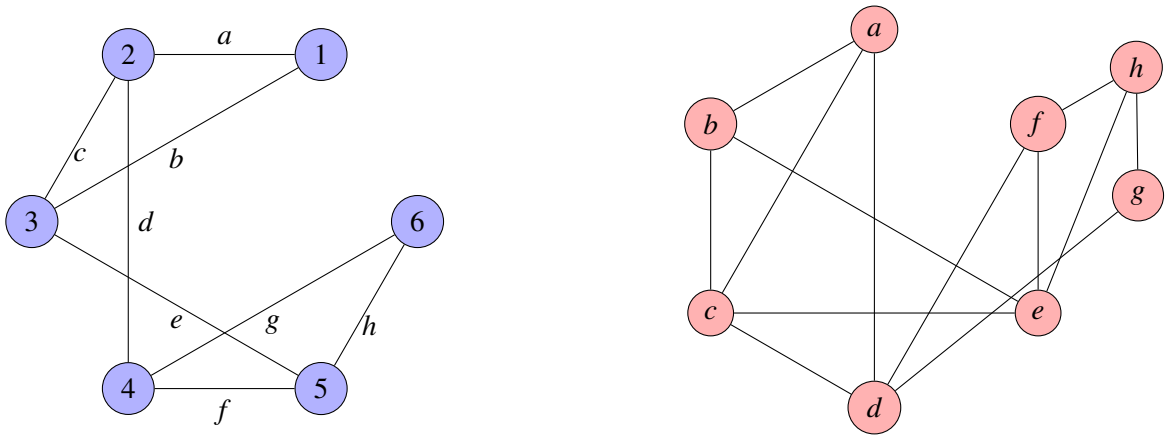
To illustrate these definitions, a small social network consisting of four colleagues is designed in Figure 2.1, left side. Consequently, if the graph representation is considered to display the interactions within this group, one would obtain a network positioned in the centre of Figure 2.1. In this case, edges represent connections between two colleagues if they work on the same project. The same network can be expressed in the form of an adjacency matrix as shown in Figure 2.1, right side. Additionally, nodal or edge attributes can be assigned. For example, if a graph is weighted, weights can be incorporated as one of the edge attributes. To create a weighted network of the case presented in Figure 2.1, the straightforward extension of that graph could be to include the number of projects the colleagues work on jointly as the edge weight.

In some applications, alternative representation techniques from Graph Theory can be more suitable. For instance, in Chapter 5 the focus lies on the network edge processes, where edges become nodes by using a line graph definition (also known as edge-to-vertex dual graph) $L(G)$ of G. The line graph $L(G)$ is constructed on E, where $(i,j) \in E$ are adjacent as nodes if and only if they are adjacent, *i.e.* share a common node, as edges in G (Diestel, 2017). In Figure 2.2 an example of a graph and its transformation into a line graph is displayed.



	v_1	v_2	v_3	v_4
v_1	0	1	0	1
v_2	1	0	1	0
v_3	0	1	0	1
v_4	1	0	1	0

Figure 2.1: An example of a social network that consists of four colleagues (left side). In the centre, the graph represents the relations between the colleagues, where an edge defines working on the same project. Here, the vertex set consists of $V = \{v_1, v_2, v_3, v_4\}$, meaning v_1 is the colleague A up to v_4 being the colleague D. Consequently, the edge set is defined as $E = \{e_{1,2}, e_{1,4}, e_{2,3}, e_{3,4}\}$. On the right side is the representation of the social network as a binary adjacency matrix. The illustration is retrieved from Malinovskaya et al. (2022).



(a) Original graph *(b)* Line graph

Figure 2.2: Example of a line graph.

To learn more about a particular network, several descriptive statistics, such as degree distribution, *i.e.* number of connections every node has to other nodes in the network, clustering coefficient, and average path length are widely applied to characterise network topology (cf. Kolaczyk and Csárdi, 2014). However, as soon as the interest in monitoring effectively a particular network arises, the computation of descriptive statistics to summarise network properties concisely but also thoroughly might not be sufficient anymore. Thus, mathematical modelling of networks becomes necessary. Hence, in the next section, the general concept of network modelling is described and famous frameworks for random networks are introduced. Afterwards, the modelling of multivariate time series is briefly discussed, being a foundation of the approach developed in Chapter 5 for monitoring fixed networks.

2.1.1.1 Random Graph Models

In the literature, there are usually two ways of regarding network models: either subdividing them into *mathematical models* and *statistical models* (cf. Kolaczyk and Csárdi, 2014) or describing them jointly under one category called *random graph models* (cf. Avrachenkov and Dreveton, 2022). The former subdivision emphasises the existence of more suitable modelling approaches for statistical inference, which are capable of realistically representing complex data, remaining a member of a later, broader category of random graph models. Nevertheless, mathematical random graph models, *e.g.* the Erdős-Rényi model, serve as the origin of the network modelling area, remaining equally important for statistical analysis of networks.

Formally, a network model is defined as a collection

$$\{\mathbb{P}_{\boldsymbol{\theta}}(G), G \in \mathscr{G} : \boldsymbol{\theta} \in \Theta\},$$

where $\mathbb{P}_{\boldsymbol{\theta}}$ is a probability distribution on the ensemble $\mathscr{G}$ and $\boldsymbol{\theta}$ is a vector of parameters, ranging over possible values in a subset of p-dimensional Euclidean space $\Theta \subseteq \mathbb{R}^p$ with $p \in \mathbb{N}$ (Kolaczyk, 2009b). In the case of a directed graph, where the edges have a direction assigned to them, this stochastic mechanism determines which of the $|V|(|V|-1)$ edges are present, *i.e.* it assigns probabilities to each of the $2^{|V|(|V|-1)}$ graphs (cf. Cannings and Penman, 2003). For the undirected case, the number of possible edges decreases to $\frac{|V|(|V|-1)}{2}$.

Starting with a classical random graph model from a mathematical domain, the Erdős-Rényi model $G(|V|, \mathrm{p})$ generates edges between nodes independently with probability p (Erdős and Rényi, 1959). Hence,

$$\mathrm{P}(Y_{ij}) = \begin{cases} \mathrm{p} & \text{for } Y_{ij} = 1 \\ 1-\mathrm{p} & \text{for } Y_{ij} = 0 \end{cases},$$

and can be rewritten for the associated adjacency matrix $\boldsymbol{Y}$ as

$$\mathrm{P}(\boldsymbol{Y}) = \prod_{i<j} \mathrm{p}^{Y_{ij}}(1-\mathrm{p})^{1-Y_{ij}} = (1-\mathrm{p})^{\frac{|V|(|V|-1)}{2}} \left(\frac{\mathrm{p}}{1-\mathrm{p}}\right)^{|E|}.$$

The result follows by obtaining number of edges as $|E| = \sum_{i<j} Y_{ij}$ (Avrachenkov and Dreveton, 2022). Moreover, the restriction $i < j$ is necessary because of the symmetry of the adjacency matrix. As one can observe, Y_{ij} are independent and identically distributed (*i.i.d.*) Bernoulli random variables with parameter p. Additionally, $Y_{ii} = 0$ for all i (no self-loops are allowed in this model). Although a useful theoretical abstraction, Erdős-Rényi graphs produce an oversimplified structure for modelling real graphs that have a heavy-tailed degree distribution and not a binomial one (cf. Barabási and Albert, 1999). Still, as shown in Section 5.4, in some applications the $G(|V|, \mathrm{p})$ model provides high usefulness.

Coming to the direct extension of Erdős-Rényi model, the Stochastic Block Model (SBM), initially introduced by Holland et al. (1983), is discussed. It enriches the previously described model by allowing the edge probability to depend on the node's membership to a particular group. Hence, by defining SBM, one considers a symmetric matrix $\boldsymbol{B} \in \mathbb{R}^{k\times k}$, for $k \ll |V|$, together with a map $C : \{1,\ldots,|V|\} \to \{1,\ldots,k\}$, such that $\mathrm{P}(Y_{ij}) = B_{C(i),C(j)}$. That means, $|V|$ nodes are grouped into k blocks or communities $Q_1,\ldots,Q_k$, with labels provided by the map C which assigns to each

node a community label. Consequently, the distribution of the adjacency matrix $\boldsymbol{Y}$ with $C(i) = r$ and $C(j) = d$ is as follows

$$\mathrm{P}(\boldsymbol{Y}) = \prod_{\substack{r,d \in \\ \{1,\ldots,k\}}} \prod_{i \in Q_r} \prod_{\substack{j \in Q_d \\ j \neq i}} B_{r,d}^{Y_{ij}} (1 - B_{r,d})^{1 - Y_{ij}},$$

where the Erdős-Rényi model $G(|V|, \mathrm{p})$ can be recovered by setting the values of $B_{r,d}$ to p.

The SBM offers flexibility in representing various community structures such as assortative and disassortative patterns within a network (Yan et al., 2014). However, SBM assumes nodes within the same community to have the same expected number of connections, contrary to the inhomogeneous degrees observed in many real-world networks (Zhao et al., 2018). When fitting an SBM to such networks, it becomes necessary to separate one community into two in case it includes members with both high and low degrees. This solution, however, introduces distortions to the original community structure. Consequently, an extension of the SBM has been developed to accommodate heterogeneous degrees, namely degree-corrected SBM (cf. Karrer and Newman, 2011). Lee and Wilkinson (2019) comprehensively review further extensions of the SBM. Nevertheless, for some applications as presented in Section 5.4, the initial version of SBM remains useful.

In this thesis, the particular interest lies in the modelling and monitoring of directed graphs (or digraphs), as proposed in Chapter 4. Initially, a framework for digraphs defined as the p_1 model was proposed by Holland and Leinhardt (1981), for the first time accounting for mutual and asymmetric relationships between nodes. Relaxing the central assumption about the independence of dyads (pair of nodes), Frank and Strauss (1986) introduce Markov Graphs applying the Hammersley-Clifford theorem (cf. Besag, 1974). This progress led to the generalisation of the approach by Frank (1991) and Wasserman and Pattison (1996), who developed the p^* model. Later this model obtained a more common name as the *Exponential Random Graph Model* (ERGM) (cf. Wasserman and Pattison, 1996; Handcock, 2003) as it belongs to the exponential family of distributions.

The functional representation of ERGM is given by

$$\mathrm{P}_{\boldsymbol{\theta}}(\boldsymbol{Y}) = \frac{\exp[\boldsymbol{\theta}' \boldsymbol{s}(\boldsymbol{Y})]}{c(\boldsymbol{\theta})}, \tag{2.1}$$

where $\boldsymbol{Y} \in \mathscr{Y}$ is the adjacency matrix of an observed graph, belonging to the ensemble of possible network states $\mathscr{Y}$, with $\boldsymbol{s} : \mathscr{Y} \to \mathbb{R}^p$ being a p-dimensional statistic describing the essential properties of a network based on $\boldsymbol{Y}$ (cf. Frank, 1991; Wasserman and Pattison, 1996). The network terms $\boldsymbol{s}(\boldsymbol{Y})$ are sufficient statistics that summarise the network structure by counting the existence of different network terms so that the inference about the model parameters $\boldsymbol{\theta}$ depends on the graph data $\boldsymbol{Y}$ only through the values of $\boldsymbol{s}(\boldsymbol{Y})$. There are several types of network terms, including dyadic dependent terms, for example, a statistic capturing transitivity, and dyadic independent terms, for instance, a term describing graph density (Morris et al., 2008). A selection of network terms that could occur in a directed network is presented in Figure 2.3. It is also possible to include nodal and edge attributes in the statistics, whose variety depends on the type of network, *i.e.* either directed or undirected graph structure.

The model parameters $\boldsymbol{\theta}$ can be defined as respective coefficients of $\boldsymbol{s}(\boldsymbol{Y})$ which are of considerable interest in understanding the structural properties of a network. They reflect, on the network level, the tendency of a graph to exhibit certain sub-structures relative to what would be expected from a model by chance, or, on the tie level, the probability of observing a specific edge, given

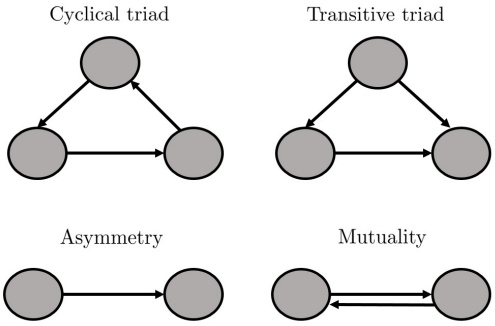


Figure 2.3: Example of triangular network terms as well as terms that describe asymmetric and mutual behaviour.

the rest of the graph (Block et al., 2018). The last interpretation follows from the representation of the problem as a log-odds ratio. The normalising constant in the denominator ensures that the sum of probabilities is equal to one, meaning it includes all possible network configurations $c(\boldsymbol{\theta}) = \sum_{\boldsymbol{Y} \in \mathscr{Y}} [\exp \boldsymbol{\theta}' \boldsymbol{s}(\boldsymbol{Y})]$ in the ensemble $\mathscr{Y}$. The application of the model's temporal extension called TERGM is a major part of the monitoring framework proposed in Chapter 4, where it is introduced in detail in Section 4.3.1.

2.1.1.2 Modelling of Multivariate Time Series

Although one remains in the field of network analysis, as soon as a fixed network structure with a random process on it is considered, alternative modelling methods are required. They differ from the previously described random graph models and originate from the classical time series modelling approaches.

Consider a stochastic process $\{X_t,\ t \in \mathbb{Z}\}$ to be a univariate time series. In the case of a network, *e.g.* for analysing a nodal attribute, one would have a separate time series $X_{i,t}$ for each node i. In terms of a whole graph, one obtains $|V|$ time series variables to be $X_{1,t}, \ldots, X_{|V|,t}$. A multivariate time series is defined as $\boldsymbol{X}$ of size $|V| \times T$, where T is the number of time points. That is, the i-th row of $\boldsymbol{X}$ is $X_{i,t}$, and for any time t, $\boldsymbol{X}_t = (X_{1,t}, \ldots, X_{|V|,t})'$.

A prominent model to represent such data is to apply the Vector Autoregressive (VAR) model of order q. Based on the description in Lütkepohl (2005), with the assumption about the mean values of nodes $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_{|V|})'$ to be $\boldsymbol{\mu} = \boldsymbol{0}$, VAR($q$) is specified as

$$\boldsymbol{X}_t = \begin{bmatrix} X_{1,t} \\ X_{2,t} \\ \vdots \\ X_{|V|,t} \end{bmatrix} = \sum_{\kappa=1}^{q} \begin{bmatrix} \phi_{1,1,\kappa} & \phi_{1,2,\kappa} & \cdots & \phi_{1,|V|,\kappa} \\ \phi_{2,1,\kappa} & \phi_{2,2,\kappa} & \cdots & \phi_{2,|V|,\kappa} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{|V|,1,\kappa} & \phi_{|V|,2,\kappa} & \cdots & \phi_{|V|,|V|,\kappa} \end{bmatrix} \begin{bmatrix} X_{1,t-\kappa} \\ X_{2,t-\kappa} \\ \vdots \\ X_{|V|,t-\kappa} \end{bmatrix} + \begin{bmatrix} u_{1,t} \\ u_{2,t} \\ \vdots \\ u_{|V|,t} \end{bmatrix},$$

or in a compact notation as

$$\boldsymbol{X}_t = \sum_{\kappa=1}^{q} \Phi_\kappa \boldsymbol{X}_{t-\kappa} + \boldsymbol{u}_t, \tag{2.2}$$

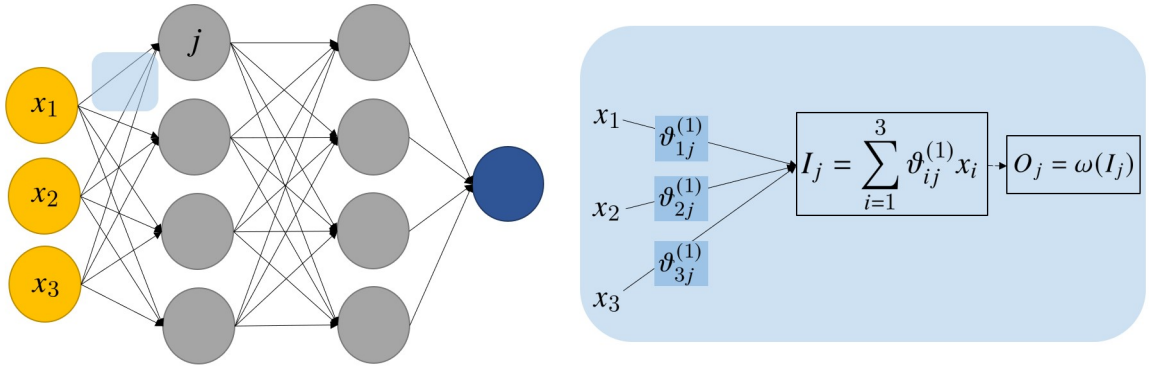


Figure 2.4: A feedforward neural network consisting of four fully connected layers (left side) and the computation of a value for the neuron j in the first hidden layer (right side). For simplicity, the bias term is not included. The illustration is retrieved from Malinovskaya et al. (2022).

where Φ_κ are fixed ($|V| \times |V|$) coefficient matrices, $\boldsymbol{u}_t = (u_{1,t}, \ldots, u_{|V|,t})'$ is a $|V|$-dimensional *white noise* or *innovation process*. That is, $\mathbb{E}(u_t) = 0$, $\mathbb{E}(u_t u_{t'}') = \Sigma_u$, for $t \neq t'$. The covariance matrix Σ_u is assumed to be non-singular if not otherwise stated.

However, here each nodal attribute would depend on the recording of the same attribute of all other nodes in the network, distorting the actual dependency structure reflected in the adjacency matrix $\boldsymbol{Y}$. Moreover, it makes the calculation mathematically and computationally intricate due to a large number of parameters, requiring a high number of T available observations for a precise estimation that restricts the application of the VAR model substantially. Hence, in Chapter 5 an alternative modelling framework is introduced that accounts for network properties and which possesses computational advantages, serving as a suitable candidate for designing an efficient monitoring procedure.

2.1.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) represent a large class of advanced computational models whose (often multilayer) structure was initially inspired by the biological brain, however, nowadays developed beyond this neuroscientific perspective. ANNs serve as prominent examples of deep learning algorithms, being a group of advanced machine learning methods that can learn gradually a large number of parameters in an architecture composed of multiple non-linear transformations.

In Figure 2.4 an example of a simple feedforward neural network consisting of four fully connected layers is displayed. Each layer is composed of separate processing elements that are known as *neurons*. Here, each neuron in one layer is connected to every neuron in the subsequent layer. Looking at the left side of Figure 2.4, the input layer (yellow) has three neurons, the two consecutive hidden layers (grey) have four neurons, and the output layer (blue) consists of one neuron. The goal of a neural network is to process the incoming data that is entered as the input layer up to the output layer, where a corresponding result known as target (*e.g.* a class label) is returned. Meanwhile, the hidden layers combine learned data representations from previous layers, abstracting irrelevant details and extracting useful features for generating the output.

The right side of Figure 2.4 illustrates calculations for obtaining a value of a particular neuron in a hidden layer. The input I_j of the neuron j corresponds to the weighted sum (applying the

parameters $\vartheta_{ij}^{(1)}$, where the (1) subscript refers to the layer being calculated, i defines the input neuron and j describes the hidden layer neuron) of values from neurons in the previous layer. Next, a non-linear function $\omega(\cdot)$ also known as the *activation function* is used, with the final value of the neuron j being $O_j = \omega(I_j)$. To minimise the error between the desired and computed outputs in the final layer of a neural network, the parameters (in the case of Figure 2.4, they are the weights displayed on the arrows connecting respective neurons) are estimated during the training phase. Thus, the main goal of training ANNs is to estimate the parameters ϑ of a highly nonlinear function $f(\cdot,\vartheta) : \mathbb{R}^l \to \mathbb{R}^u$, mapping a l-dimensional input (*i.e.* observed data) to a u-dimensional output (*i.e.* class labels).

Some changes in the number of nodes, layers, and types of connections lead to new ANN architectures. To handle high-dimensional data efficiently, specific types of ANN models have been developed. For instance, convolutional ANNs enable image data processing, while recurrent ANNs are suitable for working with temporal sequences (cf. Shrestha and Mahmood, 2019). It is worth noting that designing ANNs usually follows the trial-and-error principle (Emambocus et al., 2023). Nevertheless, some strategies are recommended by the experts, *e.g.* how to start choosing hyperparameter values (cf. Smith, 2018). Goodfellow et al. (2016) and Calin (2020) provide a broader overview of how to design and train ANNs.

2.2 Preliminaries for Statistical Network Monitoring

The goal that one usually pursues when conducting network monitoring is to identify the time point when a network (including the perspective of a network-related process) becomes spurious. To be precise, the aim is to test over time the null hypothesis

$$H_{0,t} : \text{The network observed at time point } t \text{ is in its target state}$$

against the alternative

$$H_{1,t} : \text{The network observed at time point } t \text{ deviates from its target state.}$$

The time point when an abrupt change in the process happens is also known as a *change point*. Thus, in the next section, the definitions of change point detection as well as of control charts being a prominent tool for performing it are presented.

2.2.1 Change Point Detection

Looking at network monitoring from the change point detection perspective, one is particularly interested in identifying those points in time when a significant change or shift occurs in the statistical properties of a process (Aminikhanghahi and Cook, 2017). For instance, it could result in a sudden increase or decrease in the mean or variance of a process.

Consider $\boldsymbol{s}_t = (s_1(G_t),\ldots,s_p(G_t))'$ as a collection of p network statistics which is derived from a graph G at time point t. Following, let F_0 be the in-control or target distribution and F_τ the out-of-control distribution. One refers to τ as a change point for a stochastic process $\boldsymbol{s}_t$, if

$$\boldsymbol{s}_t \quad \sim \quad \begin{cases} F_0 & \text{if } t < \tau \\ F_\tau & \text{if } t \geq \tau. \end{cases}$$

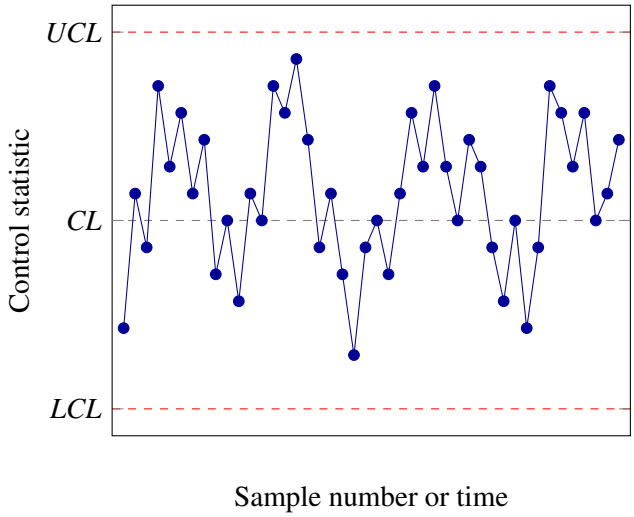


Figure 2.5: A typical control chart as presented in Montgomery (2009).

However, as indicated in each of the chapters, the definition of a change point varies from application to application due to differences in the form of how one regards the network. Consistent in each context is the endeavour to detect τ as soon as possible.

In the field of change detection, according to Basseville and Nikiforov (1993), there are three classes of problems: online (real-time) detection of a change, offline hypotheses testing and offline estimation of the change time. The methods developed in this thesis refer to the first class, meaning that the change point should be detected as soon as possible after the change has occurred. In this case, real-time monitoring of complex structures becomes necessary: for instance, if the network is observed every minute, the monitoring procedure should be faster than one minute.

To perform online surveillance, *i.e.* a real-time change detection, the efficient way is to use tools from the SPM that were originally developed for industrial purposes to achieve process stability and variability reduction (cf. Montgomery, 2009). A technique which particularly fits the requirement of real-time change point detection and belongs to the leading SPM methods is a control chart (*e.g.* Kan, 2003; Celano et al., 2013; Psarakis, 2015; Perdikis and Psarakis, 2019). Thus, the theory behind control charts is provided subsequently.

2.2.2 Control Charts

The purpose of control charts is to identify occurrences of unusual deviation of the observed process from a prespecified target (or in-control) process, distinguishing common from special causes of variation (cf. Johnson and Wichern, 2007). They graphically display how a corresponding process variable changes over time. Figure 2.5 illustrates a typical control chart, which includes a Central Line (*CL*) that defines the average of the process variable and two horizontal red lines – the Upper Control Limit (*UCL*) and the Lower Control Limit (*LCL*) (Montgomery, 2009). When the process is in control, the control or test statistic is plotted close to the *CL* within the area defined by the control limits. As soon as unusual variability occurs, the observations start appearing on or outside the control limits. This practice is known as *signalling*, meaning that the control chart signals or detects an out-of-control state, informing about the need for an investigation of the process.

There is a strong connection between control charts and hypothesis testing, as it repeatedly tests at different points of time t the null hypothesis $H_{0,t}$ against the alternative $H_{1,t}$. Taking the example from the previous section, if one is interested in monitoring the deviation of the network statistics $\boldsymbol{s}_t$ from its expected value $\boldsymbol{s}_0$, then one specifies

$$H_{0,t} : \mathbb{E}(\boldsymbol{s}_t) = \boldsymbol{s}_0 \qquad \text{against} \qquad H_{1,t} : \mathbb{E}(\boldsymbol{s}_t) \neq \boldsymbol{s}_0 \,.$$

A hypothesis $H_{0,t}$ is rejected if the control statistic is equal to or exceeds the value of one of the control limits.

Usually, the are two phases involved in the implementation of control charts. Phase I corresponds to the exploration and calibration period, where the control chart parameters are estimated (Jones-Farmer et al., 2014). It is based on the assumption that the process during this phase is in control or in its target state, *i.e.* stable, predictable and repeatable (Vining, 2009). In Phase II, the monitoring of a system starts which is assumed to stay in control, and the functionality of the control chart with respect to detected anomalies is examined, *i.e.* to out-of-control states that correspond to unusual variation of a process, and to false alarms – when no abnormality is presented but the chart signals a change. The more precise the estimation of the parameters in Phase I, the more reliable the performance of the control chart in Phase II. The desired operation of control charts during Phase II is a quick detection of the process when it experiences an out-of-control state. At the same time, one strives for a long-running scheme without false alarms occurring, meaning the case when the process actually remains in control but the scheme signals a change. This translates to a balance between small control limits for fast detection and large control limits for small false alarm rates.

There are several ways to classify control charts. In terms of the number of variables, there are univariate and multivariate control charts. An example of multivariate control charts can be found in Chapter 4 and univariate ones in Chapter 5. Another grouping is related to monitoring either the process average (mean) or the process variability (variance). There are also control charts, where the monitoring of both quantities is possible, highlighting the non-parametric control charts presented in Chapter 6. Consequently, the opposite type relates to parametric control charts applied in the remaining technical chapters. Further differentiation happens by accounting for the batch size, where there are alternative control charts in case more than one sample is observed in time stamp t. An application of such a control chart is provided in Section 6.5.4.2.

After revising the main theoretical concepts that serve as a foundation for the presented contributions in Chapters 4-7, the overview of the related research together with the identification of the scientific lacuna is provided in the next chapter.

3 Related Research and Identified Scientific Lacuna

Although many points have been demonstrated and researched in the area of statistical network monitoring, there are still open questions, some of which are the focus of this thesis. Covering more than one type of network monitoring (see Figure 1.2), the literature review as well as identification of the knowledge gaps are subdivided into several sections, following the thematic order of the subsequent Chapters 4-7.

3.1 Statistical Analysis *of* Networks

Since graphs are powerful abstractions, there are numerous applications of them, including semantic, transportation, document citation, protein-protein interactions networks and many others (cf. Chen et al., 2015; Brevier et al., 2007). Consequently, the focus of the statistical analysis of random networks varies from obtaining descriptive properties of a graph up to implementing inferential modelling (cf. Kolaczyk and Csárdi, 2014). Thus, in this section, solely the topic of random network monitoring is discussed.

According to Woodall et al. (2017), one can classify statistical monitoring methods of networks into the following four categories: 1) Bayesian methods; 2) Scan methods; 3) Time series models; 4) Control chart and hypothesis testing methods. The Bayesian methods are based on two stages: The first stage involves the application of Bayesian models for discrete time counting processes on observed connections between nodes, assessing the normality of their behaviour; in the second stage, standard network inference techniques are used on a substantially reduced subset of potentially anomalous nodes (Heard et al., 2010). The scan-based monitoring is known in the engineering literature as *moving window analysis*; Priebe et al. (2005) adapt the concept of scanning a particular region of data to digraphs, calculating a standardised statistic for specified metrics for each time window. Pincombe (2005) illustrates the idea of modelling graph topology distance measures with conventional time series models, comparing the graph for a given period with the graph from previous periods; Then the detection of changes happens by using residuals and a decision threshold.

The fourth category, being the focus of this thesis, is control chart and hypothesis testing methods. Existing in various forms in terms of the number of variables, data type and statistics being of interest, there are multiple application variations of these methods. As Multivariate Cumulative Sum (MCUSUM) and Multivariate Exponentially Weighted Moving Average (MEWMA) charts are used to propose a new monitoring framework in Chapter 4, the summary of research based on a similar choice of control charts is provided below.

The monitoring of network topology statistics by applying the Cumulative Sum (CUSUM) chart and illustrating its effectiveness in the analysis of military networks was presented by McCulloh and Carley (2011). Wilson et al. (2019) use the degree-corrected stochastic block model to generate networks and then perform surveillance over the maximum likelihood estimates using the Shewhart and Exponentially Weighted Moving Average (EWMA) charts. The combination of the Exponential Random Graph Model (ERGM) in the form of a Markov Graph together with EWMA and Hotelling's T^2 charts was proposed by Sadinejad et al. (2020). Farahani et al. (2017)

evaluate the application of MEWMA and MCUSUM together with the Poisson regression model for monitoring social networks. Hosseini and Noorossana (2018) apply EWMA and CUSUM to degree measures for detecting outbreaks in a weighted undirected network. A comprehensive study of different network characteristics, which are monitored with EWMA control charts for finding various types of changes in a network, is provided by Flossdorf and Jentsch (2021). The distribution-free MCUSUM is introduced by Liu et al. (2019) to analyse longitudinal networks. Salmasnia et al. (2019) present a comparative study of univariate and multivariate EWMA for social network monitoring. An overview of further control chart-based studies is provided by Noorossana et al. (2018).

Regarding other hypothesis testing methods, Azarnoush et al. (2016) propose monitoring the underlying network edge-formation mechanism via attributes, applying logistic regression and a likelihood-ratio test. The incorporation of vertex attributes that lead to better detection performance is also presented in Miller et al. (2013).

The combination of statistical network models and control charts still offers many research opportunities and has not reached its depth yet. Especially, the development of approaches suitable for various types of networks in terms of the edge direction and topology, enabling the integration of covariates and temporal dependence has not been explored substantially yet. Thus, in Chapter 4 a monitoring framework that covers the stated requirements is proposed.

3.2 Statistical Analysis of Processes *on* Networks

It is worth noting that compared to an established research field of statistical analysis and especially monitoring *of* networks, the analysis of processes *on* networks is currently an emerging research direction. The reasons could be twofold: The processes on networks, where networks are often physically or technically arranged fixed structures, *i.e.* between server and client computers such as data traffic in internet networks, have a strong technical scope that requires precise knowledge of constraints and rules posed by the system (*e.g.* Thottan and Ji, 2003; Kim et al., 2004; Zhang, 2009; Li et al., 2021b). On the contrary, the goal of analysing processes on networks from a statistical perspective is to model and monitor any time series with an underlying relational component, without imposing any specific constraints. Thus, although the term *fixed network monitoring* has been introduced by Stevens et al. (2021a), it has not been widely exploited yet. Thus, this section covers general methods for modelling and analysing time series where an underlying network dependency structure is taken into account.

Zhu et al. (2017) introduce a Network Autoregressive (NAR) model that remedies the deficiencies of the VAR model with a considerable number of parameters to be estimated, assuming that each node's response at a given time point is a linear combination of its previous value, the average of its connected neighbours, a set of node-specific covariates and an independent noise. As a real-world application, they model social network data with a fixed followee-follower relationship, choosing the node's response to be the length of a published post. Knight et al. (2020) propose a Generalised Network Autoregressive (GNAR) model, enriching the previous model definition by enabling the consideration of a higher-stage neighbourhood and weighted edges, although neglecting node-specific covariates. Consequently, Nason and Wei (2022) extend GNAR to GNARX, accounting for node-specific time series exogenous variables. As an empirical example, Nason and Wei (2022) model and forecast the Purchasing Managers' Indices, incorporating COVID-19 mitigation stringency indices and COVID-19 death rates as nodal covariates.

Although the developed models offer high utility, the common application is based on time series belonging to nodes. However, for some processes, the focus on the response coming from edges is considered to be more relevant or the only one recorded. Thus, in Chapter 5, a monitoring framework suitable for processes on networks, where the time series belong to edges, is developed.

3.3 Nonstationarity Detection in Machine Learning Applications

Considering supervised learning tasks such as regression or classification, data with a known relationship between the input and the output is required. During training, the algorithm minimises the error between the model's output and the target. After the parameter estimation and the testing phase, the model is deployed on new data, promising a stable performance. However, if the distribution of input data changes, it violates the stationarity assumption and adversely impacts the predictive performance (cf. Huang et al., 2011).

In computer science[1], *concept drift* refers to the problem of changes in the data distribution that render the current prediction model inaccurate or obsolete (Demšar and Bosnić, 2018). In statistics, *nonstationarity* describes time series with changing statistical properties. Additionally, *novelty* can occur, *e.g.* when the data stream introduces instances of a new class, being not present during the training and testing periods (cf. Masud et al., 2009; Garcia et al., 2019). Model revision is necessary to address these issues, ranging from retraining with new data to algorithm replacement. Strategies handling nonstationarity without explicit detection exist (cf. Gama et al., 2014), but current research highlights several benefits of employing drift detection methods during the model deployment (cf. Piano et al., 2022; Zhang et al., 2023). This minimises redundant adjustments and maintains prediction accuracy.

Detecting nonstationarity can be explored from different perspectives. In statistics, one would usually look at change point detection methods (cf. Ali et al., 2016), while in computer science, these techniques are rather known as concept drift, anomaly, or out-of-distribution detection (cf. Žliobaitė et al., 2016; Fang et al., 2022; Yang et al., 2022b). In general, approaches to detect nonstationarity in ML applications can be subdivided into two groups: performance-based methods and distribution-based methods (Hu et al., 2020). The first group relies on labelled instances, monitoring the classification error rate or other performance metrics (cf. Klinkenberg and Renz, 1998; Klinkenberg and Joachims, 2000; Nishida and Yamauchi, 2007), for example, by conducting sequential hypothesis tests based on the ideas of control charts (cf. Gama et al., 2004; Baena-Garcıa et al., 2006; Kuncheva, 2009; Mejri et al., 2017). Mejri et al. (2021) propose a two-stage time-adjusting control chart for monitoring misclassification rates, which updates the control limits in the first stage, and validates the detected changes in the second stage. However, in practice, labelled data is usually not available when ANNs are applied. Thus, the idea of adapting performance-based approaches to a confidence-related output produced by a model was introduced (Haque et al., 2016; Kim and Park, 2017). If a classifier processes the anomalous data, it is expected that the model's confidence about the affinity of a data point to a certain class would change so that the unusual behaviour could be detected (cf. Hendrycks and Gimpel, 2016).

Anomaly detection in distribution-based methods involves the analysis of metrics related to the distribution. A considerable number of techniques applies a sliding window approach on either a one-dimensional data stream or on several features individually, comparing the data distribution of the current window to the reference sample obtained from the training dataset (cf. Bifet and

[1] The overview is retrieved from Malinovskaya et al. (2023b).

Gavalda, 2007; Bifet et al., 2018; Gemaque et al., 2020). However, the underlying reason related to the notable performance of ANNs is their generalisation ability in complex high-dimensional AI tasks such as object or speech recognition (Goodfellow et al., 2016), meaning that the detection of nonstationarity from the original data would not be appropriate due to the excessive number of features. Considering novelty detection, it can be based on parametric density estimates (*e.g.* using Gaussian mixture models (Roberts and Tarassenko, 1994) or hidden Markov models (Yeung and Ding, 2003)), and nonparametric estimates (*e.g.* based on k-nearest neighbours (Guttormsson et al., 1999), kernel density estimates (Yeung and Chow, 2002) or string matching (Forrest et al., 1994)). These methods usually require heuristically chosen thresholds to decide about the novelty. A more detailed review is provided by Markou and Singh (2003a,b).

By considering *embeddings* – a latent representation of the original data stream generated by ANNs, outlier and anomaly detection methods based on distance metrics and nearest neighbour approaches were shown to be suitable and efficient in detecting nonstationary samples (cf. Lee et al., 2018; Sun et al., 2022). Particularly beneficial is their capability of providing an overall outlying score that would consider all data features together, leading to a more explicit decision about the observed abnormalities. Alternatively, other ML algorithms for drift detection such as Support Vector Machine (SVM) (Krawczyk and Woźniak, 2015) or autoencoders as specific types of ANNs (Pidhorskyi et al., 2018) can be applied for change detection. Another suggestion is to use ensemble models consisting of, for instance, several decision trees with a majority voting mechanism (Li et al., 2015). In particular cases, the detection methods can be enhanced through large-scale pre-training techniques, leading to remarkably informative embeddings (Fort et al., 2021).

Although the introduction of SPM tools to monitor ML applications is an existing research direction, *e.g.* the reviewed performance-based methods that implement control charts, there is no competitive usage of control charts for monitoring applications based on advanced ML algorithms such as ANNs. Thus, in Chapter 6 a monitoring framework to supervise the quality of ANN applications in a realistic setting is developed. That means one relaxes any assumptions on the availability of additional data for applying any pre-training techniques or specific model architectures that pose constraints on the distribution of the embeddings. Moreover, one accounts for the limited availability of labels, namely the ground truth exists only during the model training and test phases but not during its deployment. These considerations play a vital role in the monitoring approach being used by practitioners, however, there is only a limited number of suitable methods that fulfil these criteria, creating a research field that offers new perspectives for SPM.

3.4 Machine Learning Methods in Statistical Process Monitoring

Practically, four directions for complementing SPM instruments with ML algorithms and creating enhanced hybrid monitoring approaches can be identified in the current research: 1) Dimensionality reduction and feature selection; 2) Inductive learning for improving the detection precision; 3) Pattern detection of the control chart statistic as an early warning system; 4) Bypass of statistical assumptions, *e.g.* in case of the autocorrelation (cf. Kang and Park, 2000; Fountoulaki et al., 2011; Psarakis, 2011). As presented in comparative studies, ML-based SPM approaches often outperform traditional control charts in predicting when the process becomes out of control (Psarakis, 2011; Khoza and Grobler, 2019). A currently popular direction is to enhance control charts by deep learning models (cf. Chen and Yu, 2019; Zan et al., 2020; Li et al., 2021a; Kim and Ha, 2022). A comprehensive methodological review of how SPM has been enriched by the available statistical learning methods is provided by Weese et al. (2016). For a specialised literature review

either on the application of support vector machines or kernel methods in SPM, it is recommended to refer to Apsemidis and Psarakis (2020) or Apsemidis et al. (2020), respectively.

Overall, the main scientific goal in integrating ML methods into SPM procedures is the improvement of detection precision in the nowadays challenging data environment. However, there is no introduction to this area of research from the perspective of statistical network monitoring. Moreover, there is a lack of contribution to the area of augmenting SPM with automated post-monitoring procedures based on ML methods for identifying the cause of the change point. Diren et al. (2020) propose to integrate ML techniques for classifying the out-of-control signals based on fault types, facilitating the undertaking of effective corrective measures, instead of solely detecting the variables that caused out-of-control signals (cf. Murphy, 1987; Mason et al., 1995; Kourti and MacGregor, 1996; Li et al., 2008). Thus, the idea of enhancing the SPM of networks by combining it with advanced ML algorithms for simplifying the inspection step after a change point was detected has not been an explored concept yet. The procedure described in Chapter 7 tends to close this literature gap.

4 Monitoring of Networks with Changeable Structure

This chapter[1] concentrates on modelling and monitoring of networks with randomly generated edges across time, developing a surveillance method that accounts for temporal dependence in the network structure and enables the detection of significant changes in real time. In other words, this chapter is dedicated to the field of *Random Network Monitoring*.

4.1 Main Intent

To monitor the network data effectively, it is important to account for its complex structure and possibly high computational costs. The proposed approach for mitigating these issues and simultaneously reflecting the stochastic and dynamic nature of network models is to model graphs by applying a temporal random graph model. Afterwards, one proceeds with a "compressed" version of the graph, using it in the change point detection part. A general class of Exponential Random Graph Models (ERGM) (cf. Frank and Strauss, 1986; Robins et al., 2007; Schweinberger et al., 2020) is considered, which was originally designed for modelling cross-sectional networks and briefly described in Section 2.1.1.1. This class includes many prominent random network configurations such as dyadic independence models and Markov random graphs, enabling the ERGM to be generally applicable to many types of complex networks. Furthermore, if a network has many covariates (also called *attributes*), *i.e.* variables that provide additional information about the graph's edges and nodes, it can be both computationally and theoretically challenging to recognise meaningful patterns. In this case, it is beneficial to apply ERGM which facilitates data reduction by summarising the network in the form of sufficient statistics that capture relevant features of a network. Knowing the observed values of sufficient statistics, one can derive the model parameters from the network data and make inferences.

Hanneke et al. (2010) propose a dynamic extension based on ERGM which is known as the Temporal Exponential Random Graph Model (TERGM). This model contains the overall functionality of the ERGM, while additionally enabling time-dependent covariates. To perform real-time detection of changes, multivariate control charts are chosen, being an efficient and versatile tool for this task. It relies on exponential smoothing and cumulative sums to track network behaviour over time, which is generated using TERGM. Subsequently, one starts defining the framework with the definition of the change point and the respective monitoring statistic.

4.2 Definition of the Change Point

Consider a network presented by its adjacency matrix $\boldsymbol{Y} := (Y_{ij})_{i,j=1,\dots,|V|}$, where $|V|$ represents the total number of nodes. In a dynamic setting, a random sequence of $\boldsymbol{Y}_t$ for $t = 1,2,\dots$ with $\boldsymbol{Y}_t \in \mathscr{Y}$ defines a stochastic process for all t, where $\mathscr{Y}$ denotes the ensemble of possible network

[1] This chapter is based on the publication *Malinovskaya, A., Otto, P. Online network monitoring. Statistical Methods & Applications, Special Issue on Statistical Analysis of Networks, 30(5), 1337-1364, (2021).* Available online: `https://link.springer.com/article/10.1007/s10260-021-00589-z` . Published under Creative Commons licence CC BY.

states. The overall concept presented in this chapter is valid for both directed and undirected graph types, however, from now on directed graphs are explicitly considered.

Although the monitoring procedure can be constructed by supervising $\boldsymbol{Y}_t$ directly, this approach is likely to become computationally intricate as it depends on the order of a graph, leading to the curse of dimensionality. In the case of network models considered in this chapter, there are two reasonable choices for network monitoring, namely, it can be performed either in terms of the (normalised) network statistics or the model parameters whose dimension remains independent from the network evolvement.

To obtain a time series of the corresponding estimates, the application of the moving window approach with the window size z is proposed. More precisely, one takes into account the past z observations of the network $\{\boldsymbol{Y}_{t-z+1},\ldots,\boldsymbol{Y}_t\}$ to estimate the respective quantities at time point t.

Let $\boldsymbol{\theta}$ be the true model parameters and $\hat{\boldsymbol{\theta}}_t$ their estimates at time point t based on the last z network states. Similarly, the expected value of the network statistics $\mathbb{E}_{\boldsymbol{\theta}}(\boldsymbol{s}(\boldsymbol{Y}))$ can be estimated as

$$\hat{\boldsymbol{s}}_t = \frac{1}{z}\sum_{n=0}^{z-1}\boldsymbol{s}(\boldsymbol{Y}_{t-n}). \tag{4.1}$$

Concerning the choice of monitoring the network statistics or the model parameters, it is worth noting that there is a one-to-one relationship between $\boldsymbol{\theta}$ and $\mathbb{E}_{\boldsymbol{\theta}}(\boldsymbol{s}(\boldsymbol{Y}))$. That is, for every $\boldsymbol{\theta}$, there is only one expectation of $\boldsymbol{s}(\boldsymbol{Y})$. Hence, one can monitor the network based on the estimates of either $\boldsymbol{\theta}$ or $\mathbb{E}_{\boldsymbol{\theta}}(\boldsymbol{s}(\boldsymbol{Y}))$. Since the monitoring procedure is identical for $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{s}}_t$, a new notation $\hat{\boldsymbol{c}}_t$ for the estimates of the network characteristics is introduced. Consequently, $\boldsymbol{c}$ corresponds either to $\boldsymbol{\theta}$ or to $\mathbb{E}_{\boldsymbol{\theta}}(\boldsymbol{s}(\boldsymbol{Y}))$.

Let p be the number of network terms, which describe the in-control state and can reflect the deviations in the case of an out-of-control state. Thus, at time point t there is a p-dimensional vector $\hat{\boldsymbol{c}}_t = (\hat{c}_{1t},\ldots,\hat{c}_{pt})'$ that estimates the network characteristics $\boldsymbol{c}$. Moreover, let $F_{\boldsymbol{c}_0,\boldsymbol{\Sigma}}$ be the target distribution of these estimates with $\boldsymbol{c}_0 = \mathbb{E}_0(\hat{c}_1,\ldots,\hat{c}_p)'$ being the expected value and $\boldsymbol{\Sigma}$ the respective $p \times p$ variance-covariance matrix of the network characteristics (Montgomery, 2009). Thus,

$$\hat{\boldsymbol{c}}_t \quad \sim \quad \begin{cases} F_{\boldsymbol{c}_0,\boldsymbol{\Sigma}} & \text{if } t < \tau \\ F_{\boldsymbol{c}_\tau,\boldsymbol{\Sigma}} & \text{if } t \geq \tau, \end{cases} \tag{4.2}$$

where τ denotes a change point to be detected and $\boldsymbol{c}_\tau \neq \boldsymbol{c}_0$. If $\tau = \infty$ or $t < \tau$ the network is said to be in control, whereas it is out of control in the case of $\tau \leq t < \infty$. Furthermore, one assumes that the estimation precision of the parameters does not change across t, *i.e.* $\boldsymbol{\Sigma}$ is constant for the in-control and out-of-control state. Hence, the monitoring procedure is based on the expected values of $\hat{\boldsymbol{c}}_t$. In fact, one can specify the corresponding hypothesis as follows

$$H_{0,t} : \mathbb{E}(\hat{\boldsymbol{c}}_t) = \boldsymbol{c}_0 \qquad \text{against} \qquad H_{1,t} : \mathbb{E}(\hat{\boldsymbol{c}}_t) \neq \boldsymbol{c}_0\,.$$

4.3 Monitoring Framework

The proposed monitoring procedure for random network monitoring consists of two steps: 1) Network modelling using the TERGM that is described in Section 4.3.1; 2) Process monitoring using a multivariate control chart presented in Section 4.3.2. To determine the most satisfactory

monitoring setting in terms of performance, two kinds of control charts are compared, namely cumulative sum and exponentially weighted moving average control charts.

4.3.1 Temporal Exponential Random Graph Model

To conduct surveillance over $\boldsymbol{Y}_t$, one considers only the dynamically estimated characteristics of a graph in order to reduce computational complexity and allow for real-time monitoring. In most cases, the dynamic network models serve as an extension of well-known static models. Similarly, the discrete temporal expansion of the ERGM is known as TERGM (cf. Hanneke et al., 2010) and can be seen as a further advancement of a family of network models proposed by Robins and Pattison (2001).

The TERGM defines the probability of a network at the discrete time point t both as a function of subgraph counts in t and by including the network terms based on the previous graph observations until the particular time point $t-v$. That is

$$\mathrm{P}_{\boldsymbol{\theta}}(\boldsymbol{Y}_t|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_{t-v},\boldsymbol{\theta}) = \frac{\exp[\boldsymbol{\theta}'\boldsymbol{s}(\boldsymbol{Y}_t,\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_{t-v})]}{c(\boldsymbol{\theta},\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_{t-v})}, \tag{4.3}$$

where v represents the maximum temporal lag, capturing the networks which are incorporated into the $\boldsymbol{\theta}$ estimation at t, hence, defining the complete temporal dependence of $\boldsymbol{Y}_t$ that corresponds to the Markov structure of order $v \in \mathbb{N}$ (Hanneke et al., 2010). In Sections 4.4 and 4.5, $v = 1$ is assumed, leading to $(\boldsymbol{Y}_t \perp\!\!\!\perp \{\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_{t-2}\}|\boldsymbol{Y}_{t-1})$, where $\perp\!\!\!\perp$ defines conditional independence.

To model the joint probability of z networks between the time stamps $v+1$ and $v+z$, one defines $\mathrm{P}_{\boldsymbol{\theta}}$ based on the conditional independence assumption as

$$\mathrm{P}_{\boldsymbol{\theta}}(\boldsymbol{Y}_{v+1},\ldots,\boldsymbol{Y}_{v+z}|\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_v,\boldsymbol{\theta}) = \prod_{t=v+1}^{v+z} \mathrm{P}_{\boldsymbol{\theta}}(\boldsymbol{Y}_t|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_{t-v},\boldsymbol{\theta}). \tag{4.4}$$

Regarding the network statistics in the TERGM, $\boldsymbol{s}(\cdot)$ includes *memory terms* such as dyadic stability or reciprocity (Leifeld et al., 2018). To distinguish the processes leading to the dissolution and formation of links, Krivitsky and Handcock (2014) presented Seperable TERGM (STERGM). To be precise, the STERGM is a subclass of the TERGM class, which can reproduce any transition process captured by the parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}^+,\boldsymbol{\theta}^-)$ and the network terms $\boldsymbol{s} = (\boldsymbol{s}^+,\boldsymbol{s}^-)$, where $\boldsymbol{\theta}^+$ and $\boldsymbol{s}^+$ belong to the formation model, $\boldsymbol{\theta}^-$ and $\boldsymbol{s}^-$ to the dissolution model.

The careful selection of the network statistics is relevant from several points of view. First of all, under the maximum likelihood estimation, the expected value of the network statistics is equal to the observed value, *i.e.* $\mathbb{E}_{\boldsymbol{\theta}}(\boldsymbol{s}(\boldsymbol{Y})) = \boldsymbol{s}(\boldsymbol{Y}_{obs})$ (cf. Van Duijn et al., 2009). To be precise, on average, the observed network $\boldsymbol{Y}_{obs}$ is reproduced in terms of sufficient statistics $\boldsymbol{s}(\boldsymbol{Y})$. Second, the selected network statistics determine the understanding of the network formation, combining the available knowledge about the important terms to recover the graph structure with the interest of including additional statistics for monitoring. The dimension of the sufficient statistics can differ over time, however, one assumes that in each time stamp t the same configuration $\boldsymbol{s}(\cdot)$ is given. In general, the selection of terms extensively depends on the field and context, although the statistical modelling standards such as avoidance of linear dependencies among the terms should be also considered (Morris et al., 2008). It is also helpful to perform goodness of fit tests, which enable one to find a compromise between the model's complexity and its explanatory power.

An improper selection of the network terms can often lead to a near-degenerate model, resulting in algorithmic issues and lack of fit (cf. Handcock, 2003; Schweinberger, 2011). In this case, apart from the fine-tuning of how statistics are configured, one can modify some settings which design the estimation procedure of the model parameters. Considering the Markov Chain Monte Carlo (MCMC) maximum likelihood estimation, for example, the run time, the sample size or the step length could be adjusted (Morris et al., 2008). Another possible improvement would be to add some stable statistics such as Geometrically-Weighted Edgewise Shared Partnerships (GWESP) (Snijders et al., 2006). However, the TERGM is less prone to degeneracy issues compared to the ERGM as ascertained by Hanneke et al. (2010) and Leifeld and Cranmer (2019). Overall, one assumes that most of the network surveillance studies can reliably estimate beforehand the type of anomalies which are possible to occur. This assumption guides the choice of terms in the models throughout the chapter.

4.3.2 Multivariate Cumulative Sum and Exponentially Weighted Moving Average Control Charts

The strength of the multivariate control chart over the univariate control chart is the ability to monitor several interrelated process variables at once. It implies that the corresponding test statistic should take into account the correlation of the data and be dimensionless as well as scale-invariant, as the process variables can considerably differ from each other. The squared Mahalanobis distance, which represents the general form of the control statistic, fulfils these criteria and is defined as

$$D_t^{(1)} = (\hat{\boldsymbol{c}}_t - \boldsymbol{c}_0)' \boldsymbol{\Sigma}^{-1} (\hat{\boldsymbol{c}}_t - \boldsymbol{c}_0), \tag{4.5}$$

being part of the respective *data depth* expression – Mahalanobis depth that measures a deviation from an in-control distribution (cf. Liu, 1995). Hence, $D_t^{(1)}$ maps the p-dimensional characteristic quantity $\hat{\boldsymbol{c}}_t$ to a one-dimensional measure. It is important to note that the characteristic quantity at time point t is usually the mean of several samples at t, but in the discussed case, only one network is observed at each instant of time. Thus, the characteristic quantity $\hat{\boldsymbol{c}}_t$ is the value of the obtained estimates and not the average of several samples.

In Sections 4.4 and 4.5, two control chart types are applied and compared in their performance to monitor networks. Firstly, multivariate CUSUM (MCUSUM) charts (cf. Woodall and Ncube, 1985; Joseph et al., 1990; Ngai and Zhang, 2001) are discussed. One of the widely used versions was proposed by Crosier (1988) and is defined as follows

$$C_t = \left[(\boldsymbol{r}_{t-1} + \hat{\boldsymbol{c}}_t - \boldsymbol{c}_0)' \boldsymbol{\Sigma}^{-1} (\boldsymbol{r}_{t-1} + \hat{\boldsymbol{c}}_t - \boldsymbol{c}_0) \right]^{1/2}, \tag{4.6}$$

where

$$\boldsymbol{r}_t = \begin{cases} \mathbf{0} & \text{if } C_t \leq k, \\ (\boldsymbol{r}_{t-1} + \hat{\boldsymbol{c}}_t - \boldsymbol{c}_0)(1 - k/C_t) & \text{if } C_t > k, \end{cases}$$

given that $\boldsymbol{r}_0 = \mathbf{0}$ and $k > 0$. The respective chart statistic is

$$D_t^{(2)} = \boldsymbol{r}_t' \boldsymbol{\Sigma}^{-1} \boldsymbol{r}_t, \tag{4.7}$$

and it signals if $\sqrt{D_t^{(2)}}$ is greater than or equals the upper control limit UCL. The lower control limit $LCL = 0$ because the chart statistic obtains non-negative values so only the UCL has to be determined. Certainly, the values k and UCL considerably influence the performance of the chart. The parameter k, also known as reference value or allowance, reflects variation tolerance, taking into consideration δ – the deviation from the mean one aims to detect, which is measured in the

standard deviation units. According to Page (1954) and Crosier (1988), the chart is approximately optimal if $k = \delta/2$.

Besides, multivariate charts based on exponential smoothing (EWMA) are considered. Lowry et al. (1992) propose a multivariate extension of the EWMA control chart (MEWMA), which is defined as follows

$$\boldsymbol{l}_t = \lambda(\hat{\boldsymbol{c}}_t - \boldsymbol{c}_0) + (1-\lambda)\boldsymbol{l}_{t-1} \tag{4.8}$$

with $0 < \lambda \leq 1$ and $\boldsymbol{l}_0 = \boldsymbol{0}$ (cf. Montgomery, 2009). The corresponding chart statistic is

$$D_t^{(3)} = \boldsymbol{l}_t' \boldsymbol{\Sigma}_{\boldsymbol{l}_t}^{-1} \boldsymbol{l}_t, \tag{4.9}$$

where the covariance matrix is defined as

$$\boldsymbol{\Sigma}_{\boldsymbol{l}_t} = \frac{\lambda}{2-\lambda}\left[1-(1-\lambda)^{2t}\right]\boldsymbol{\Sigma}. \tag{4.10}$$

Here, the control chart signals if $D_t^{(3)} \geq UCL$. Together with the MCUSUM, the MEWMA is an advisable approach for detecting relatively small but persistent changes. However, the detection of large shifts is also possible by setting the reference parameter k or the smoothing parameter λ high. For instance, in the case of the MEWMA with $\lambda = 1$, the chart statistic coincides with $D_t^{(1)}$. Thus, it is equivalent to Hotelling's T^2 control procedure, which is suitable for the detection of substantial deviations. It is worth mentioning that the discussed methods are directionally invariant, therefore, the investigation of the data at the signal time point is necessary if the change direction is of particular interest.

4.3.2.1 Estimation of In-Control Parameters

In practice, the in-control parameters $\boldsymbol{c}_0$ and $\boldsymbol{\Sigma}$ are usually unknown and therefore have to be estimated. Thus, one subdivides the sequence of network observations into Phase I and Phase II. In Phase I, the process must coincide with the in-control state so that the true in-control parameters $\boldsymbol{c}_0$ and $\boldsymbol{\Sigma}$ can be estimated by the sample mean vector $\bar{\boldsymbol{c}}$ and the sample covariance matrix $\boldsymbol{S}$ from $\hat{\boldsymbol{c}}_t$.

It is important that Phase I replicates the natural behaviour of a network so that possible dynamics related to its growth or changes in its topological structure are considered. Similarly, if the network is prone to remain constant, this fact should be captured in Phase I for reliable estimation and later network surveillance. After the estimates of $\boldsymbol{c}_0$, $\boldsymbol{\Sigma}$ and the UCL are obtained, the calibrated control chart can be applied to the actual data in Phase II.

4.3.2.2 Computation of Control Limits

If $D_t^{(2)}$ or $D_t^{(3)}$ is equal to or exceeds the UCL, it means that the charts signal a change. To determine the UCL values, one typically assumes that the chart has a predefined (low) probability of false alarms, *i.e.* signals when the process is in control, or a prescribed in-control Average Run Length denoted as ARL_0 that represents the number of expected time stamps until the first signal. To compute the UCL values corresponding to ARL_0 theoretically, a prevalent number of multivariate control charts require a normally distributed target process (cf. Johnson and Wichern, 2007; Porzio and Ragozini, 2008; Montgomery, 2009). Here, this assumption would need to be valid for the estimates of the model parameters/the network statistics. However, while there are some

studies on the distributions of particular network statistics $\boldsymbol{s}(\boldsymbol{Y})$ (cf. Yan and Xu, 2013; Yan et al., 2016; Sambale and Sinulis, 2020), only a few results are obtained about the parameter estimates of $\boldsymbol{\theta}$. Primarily, the difficulty in determining the distribution is that the assumption of independent and identically distributed data is violated in the ERGM case. In addition, the parameters depend on the choice of the model terms and network size (He and Zheng, 2015). Kolaczyk and Krivitsky (2015) proved asymptotic normality for the maximum likelihood estimators in a simplified context of the ERGM, pointing out the necessity to establish a deeper understanding of the distributional properties of parameter estimators.

In case the normality assumption is violated to a slight or moderate degree, the control charts still will remain robust (Montgomery, 2009). The most crucial assumption that needs to be satisfied is the independence of the observations at different time points (Qiu, 2014). If the data is autocorrelated, the theoretically derived *UCL* values become invalid, so their implementation would lead to inaccurate results. However, here networks which are dependent over time are considered. Moreover, networks used for the estimation of the characteristics $\hat{\boldsymbol{c}}_t$ are overlapping due to the application of the moving window approach. As shown in Section 4.4.2, the characteristics that are based on the averaged network statistics $\hat{\boldsymbol{s}}_t$ can violate this assumption substantially. Regarding the estimates $\hat{\boldsymbol{\theta}}_t$, if their computation does not involve overlapping of the networks by the sliding window approach of size z, *i.e.* each graph is involved only once in the estimation of $\boldsymbol{\theta}$, and the size of z is enough for recovering the temporal dependence completely, then the estimates become uncorrelated. Nonetheless, as one designs an online monitoring procedure, one supports the idea of computing $\hat{\boldsymbol{c}}_t$ immediately as soon as a new data point is available. In this case, one needs to account for the correlation between the estimated characteristics $\hat{\boldsymbol{c}}_t$.

There are several works which apply control charts in the presence of autocorrelation, advising either using the residuals of the time series models as observations, calculating theoretical control limits under autocorrelation or designing a simulation study to determine the control limits corresponding to the desired ARL_0 (cf. Montgomery and Mastrangelo, 1991; Alwan, 1992; Runger and Willemain, 1995; Schmid and Schöne, 1997; Zhang, 1997; Lu and Reynolds Jr, 1999, 2001; Sheu and Lu, 2009). It is worth noting that the residual charts have different properties from the traditional charts, which are considered in this chapter. Hence, the *UCL* values are determined via Monte Carlo simulations described in Section 4.4.2.

4.4 Simulation Study

To verify the applicability and effectiveness of the proposed approach and also to determine the *UCL* values, a simulation study is designed, followed by the surveillance of real-world data in Section 4.5 with the goal of obtaining some insights into its temporal development.

4.4.1 Generation of Network Time Series

To compute $\bar{\boldsymbol{c}}$ and $\boldsymbol{S}$ a certain number of in-control networks is required. For this purpose, 2500 temporal graphs of desired length $T < \tau$ are generated, where each graph consists of $|V| = 100$ nodes, meaning that the simulation is repeated 2500 times to estimate the respective parameters and control limits in Section 4.4.2. The simulation of synthetic networks is based on the Markov chain principle: The network observation in time point $\boldsymbol{Y}_t$ is simulated from its previous state $\boldsymbol{Y}_{t-1}$ by selecting randomly a fraction ϕ of elements of the adjacency matrix and setting them to either 1 or 0, according to a specified transition matrix $\boldsymbol{M}$. This setting allows for the inclusion of the memory term during the estimation of the TERGM that reflects the stability of both edges and

non-edges between the previous and the current network observation. The in-control values are $\phi_0 = 0.01$ and

$$\boldsymbol{M}_0 = \begin{pmatrix} m_{00,0} & m_{01,0} \\ m_{10,0} & m_{11,0} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.1 \\ 0.4 & 0.6 \end{pmatrix},$$

where $m_{ij,0}$ denotes the probability of a transition from i to j in the in-control state.

At the beginning of each sequence, a directed network which is called the *base network* is simulated by applying an ERGM with predefined network terms and corresponding coefficients so that it is possible to control the network creation indirectly. This procedure helps to guarantee that the temporal networks have a stochastic but analogous initialisation. Hence, three network statistics are selected, namely an edge term, a triangle term and a parameter that defines asymmetric dyads. These terms are used later for estimating network characteristics. Subsequently, a new graph is produced by applying the in-control fraction ϕ and the transition matrix $\boldsymbol{M}$.

Next, one needs to confirm that the generated samples of networks behave according to the requirements of Phase I, capturing only the usual variation of the target process. For this purpose, one can exploit Markov chain properties and calculate its steady state equilibrium vector $\boldsymbol{\pi}$, as it follows that the expected number of non-edges and edges is given by $\boldsymbol{\pi}$. Using eigenvector decomposition, the steady state is found to be $\boldsymbol{\pi} = (0.8, 0.2)'$. Consequently, the expected number of edges in the graph in its steady state is 1980. However, the network density is only one of the aspects to define the in-control process, as the temporal development and the topology are also involved in the network creation. Hence, a suitable start of the considered network sequence is identified by computing the network statistics $\boldsymbol{s}(\boldsymbol{Y}_t)$ over multiple network time series. By plotting the behaviour, one determines that all four terms become stable by $t = 1000$. Thus, 1000 network observations are simulated in a burn-in period so that the in-control sequence of network states starts at $t = 1001$. In total, $T = 1600$, where the length of Phase II amounts to 600 time points.

4.4.2 Calibration of the Charts in Phase I

After the generation of temporal networks, one computes $\hat{\boldsymbol{\theta}}_t$ by fitting the TERGM and $\hat{\boldsymbol{s}}_t$ by applying Equation (4.1) with a certain window size z using the four network terms, namely edge term, a triangle term, a term that defines asymmetric dyads and a memory term which describes the stability of both edges and non-edges over time with the temporal lag $v = 1$. Currently, there are two widely used approaches to estimate the TERGM: Maximum Pseudolikelihood Estimation (MPLE) with bootstrapped confidence intervals and MCMC maximum likelihood estimation (Leifeld et al., 2018). The chosen estimation method to derive $\hat{\boldsymbol{\theta}}_t$ is the bootstrap MPLE which is appropriate to handle a relatively large number of nodes and time points (Leifeld et al., 2018). Next, the in-control parameters $\bar{\boldsymbol{c}}$ and $\boldsymbol{S}$ are calculated for both monitoring cases. Finally, different control charts are calibrated by obtaining the *UCL* values with respect to the predefined ARL_0 via the bisection method. For two window sizes $z = \{7, 14\}$, Tables 4.1 and 4.3 summarise the obtained results for surveillance of $\boldsymbol{\theta}$, and Tables 4.2 and 4.4 for surveillance of $\boldsymbol{s}(\boldsymbol{Y})$ with the MEWMA and MCUSUM charts respectively. If one wishes to apply the TERGM with the same network terms and a similar window size, the presented *UCL* values can be used directly. Otherwise, it is necessary to conduct different Monte Carlo simulations that address the specific settings of the TERGM.

As both network characteristics describe the same process, one would expect the *UCL* results to be similar. However, in Figure 4.1 the analysis of the autocorrelation functions applied to the

z	ARL_0/λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
7	50	39.32	35.76	31.04	26.52	22.58	19.22	16.46	14.10	12.14	10.46
	75	45.76	41.03	35.20	29.81	25.27	21.61	18.53	15.86	13.66	11.72
	100	50.52	45.30	38.58	32.56	27.62	23.48	20.00	17.13	14.71	12.66
14	50	55.14	43.44	33.80	26.96	21.98	18.16	15.16	12.76	10.76	9.15
	75	65.63	50.94	39.29	31.23	25.26	20.68	17.24	14.50	12.20	10.32
	100	73.85	56.20	42.97	33.97	27.44	22.52	18.72	15.69	13.21	11.15

Table 4.1: Upper control limits for the MEWMA chart based on the estimates $\hat{\boldsymbol{\theta}}_t$ and $ARL_0 \in \{50,75,100\}$ for two different windows sizes $z = 7$ and $z = 14$.

z	ARL_0/λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
7	50	65.03	43.79	32.23	25.29	20.36	16.72	13.91	11.66	9.85	8.31
	75	82.40	52.41	38.13	29.52	23.59	19.23	15.88	13.23	11.13	9.42
	100	96.23	59.39	42.92	32.96	26.19	21.24	17.58	14.69	12.32	10.35
14	50	71.09	45.47	32.66	24.81	19.51	15.80	12.95	10.74	8.96	7.53
	75	89.03	55.26	38.71	29.06	22.85	18.40	15.07	12.46	10.35	8.66
	100	103.00	62.73	43.73	32.56	25.40	20.40	16.65	13.73	11.43	9.57

Table 4.2: Upper control limits for the MEWMA chart based on the estimates $\hat{\boldsymbol{s}}_t$ and $ARL_0 \in \{50,75,100\}$ for two different windows sizes $z = 7$ and $z = 14$.

z	ARL_0/k	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
7	50	21.01	19.36	17.75	16.21	14.83	13.52	12.25	11.10	9.99	9.00	8.03
	75	25.19	22.90	20.82	18.95	17.33	15.91	14.47	13.19	11.94	10.74	9.61
	100	28.25	25.59	23.31	21.18	19.38	17.67	16.12	14.67	13.27	11.96	10.73
14	50	30.10	27.84	25.64	23.67	21.69	19.83	17.92	16.17	14.44	12.75	11.06
	75	37.35	34.60	31.91	29.25	26.86	24.53	22.37	20.20	18.14	16.19	14.32
	100	43.06	39.52	36.20	33.15	30.45	27.84	25.43	23.16	20.97	18.81	16.77

Table 4.3: Upper control limits for the MCUSUM chart based on the estimates $\hat{\boldsymbol{\theta}}_t$ and $ARL_0 \in \{50,75,100\}$ for two different windows sizes $z = 7$ and $z = 14$.

z	ARL_0/k	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
7	50	51.85	47.41	42.96	38.27	33.87	29.71	25.94	22.51	19.01	15.79	13.06
	75	75.93	69.26	62.97	56.83	50.79	44.82	39.02	33.73	29.05	24.95	20.95
	100	97.58	89.46	81.68	73.51	65.78	59.01	52.43	45.96	39.96	34.40	29.23
14	50	55.72	51.27	46.63	41.90	37.54	33.29	29.18	25.46	21.69	18.21	15.36
	75	80.28	73.70	67.32	61.13	54.75	48.86	43.15	37.76	32.81	28.26	24.20
	100	102.34	94.25	85.88	78.05	70.90	63.96	57.03	50.65	44.31	38.71	33.39

Table 4.4: Upper control limits for the MCUSUM chart based on the estimates $\hat{\boldsymbol{s}}_t$ and $ARL_0 \in \{50,75,100\}$ for two different windows sizes $z = 7$ and $z = 14$.

Anomaly Type	Description	Case	
Type A	Change in the transition matrix $\boldsymbol{M}$	A.1	$m_{00,1} = 0.89$ ($m_{00,0} = 0.9$) $m_{01,1} = 0.11$ ($m_{01,0} = 0.1$)
		A.2	$m_{10,1} = 0.6$ ($m_{10,0} = 0.4$) $m_{11,1} = 0.4$ ($m_{11,0} = 0.6$)
		A.3	$m_{00,1} = 0.5$ ($m_{00,0} = 0.9$) $m_{11,1} = 0.5$ ($m_{11,0} = 0.6$)
Type B	Change of the fraction $\phi_0 = 0.01$	B.1	$\phi_1 = 0.009$
		B.2	$\phi_1 = 0.015$
		B.3	$\phi_1 = 0.02$
Type C	Increase of the proportion of mutual edges by ζ	C.1	$\zeta = 0.005$
		C.2	$\zeta = 0.01$
		C.3	$\zeta = 0.05$

Table 4.5: Anomaly cases.

estimates of one of the generated network time series shows that the dependence structures of $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{s}}_t$ considerably differ. While the elimination of the overlap in the calculation procedure removes correlation in the case of the parameter estimates $\hat{\boldsymbol{\theta}}_t$, there is only a slight improvement regarding the averaged network statistics $\hat{\boldsymbol{s}}_t$. Thus, the *UCL* values are different for both cases.

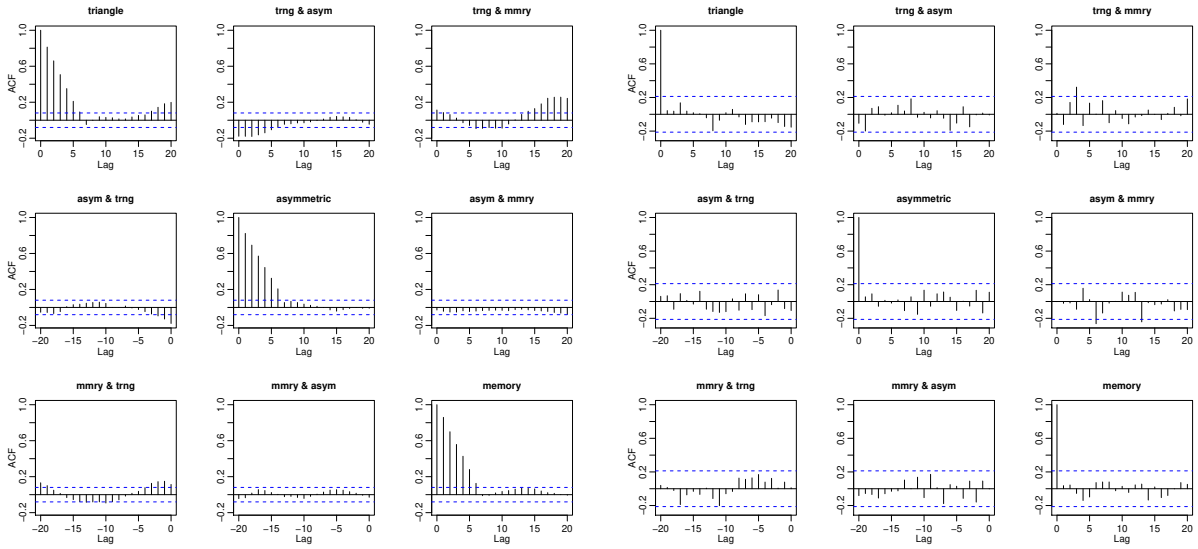
4.4.3 Design of the Anomalous Behaviour

To test how well the proposed control charts can detect the changes in the networks' development, it is necessary to compose different anomalous cases and generate samples from Phase II. Since the focus is on the detection of shifts in the process mean, an anomalous change can occur either in the proportion of the asymmetric edges, in the fraction of the randomly selected adjacency matrix entries ϕ or in the transition matrix $\boldsymbol{M}$. Thus, these scenarios are subdivided into three different anomaly types which are briefly described in the flow chart presented in Figure 4.2.

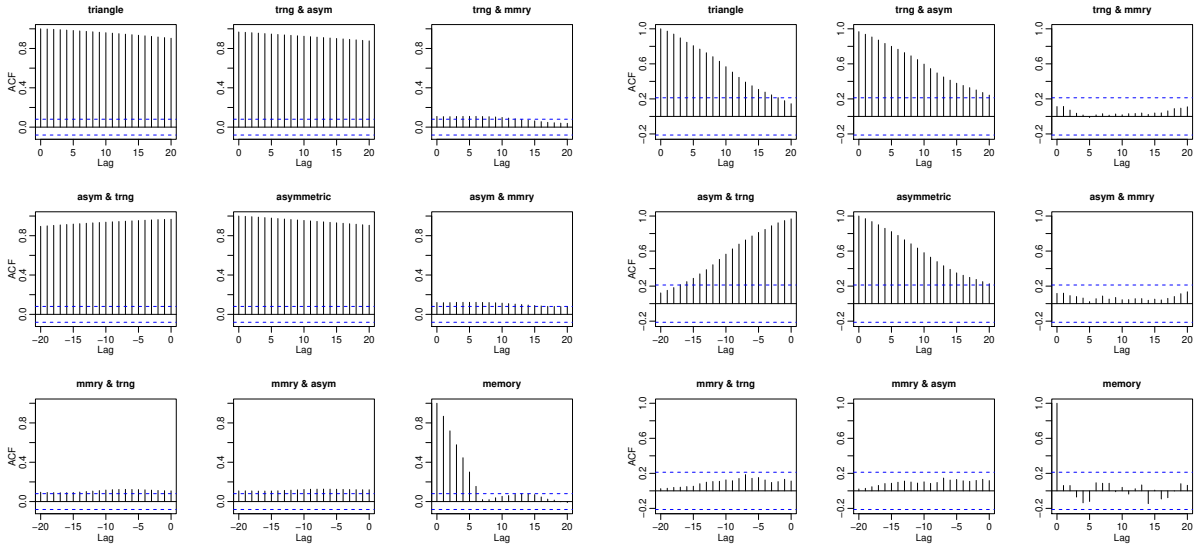
One defines a Type A anomaly as a change in the values of $\boldsymbol{M}$. That is, there is a transition matrix $\boldsymbol{M}_1 \neq \boldsymbol{M}_0$ when $t \geq \tau$. Similar to Type A, Type B anomalies are created by introducing a new fraction value ϕ_1 in the generation process when $t \geq \tau$. Both types are instances of a persistent change (also known as simply a *change*), where the abnormal development continues for all $t \geq \tau$ (Ranshous et al., 2015). Anomalies of Type C differ from the previous two types as they represent a *point change* (also referred to as an *event*) – the abnormal behaviour occurs only at a single point of time τ but its outcome may also affect subsequent network states in the considered case due to the Markov property. One recreates this type of anomaly by converting a fraction ζ of asymmetric edges into mutual links. This process happens at time point τ only. Afterwards, the new network states are created similar to Phase I by applying $\boldsymbol{M}_0$ and ϕ_0 up until the anomaly is detected. The considered cases are summarised in Table 4.5.

4.4.4 Performance of the Charts in Phase II

In the next step, the performance of the proposed charts is analysed in terms of their detection speed. As a performance measure, the Conditional Expected Delay (*CED*) of detection is computed, conditional on a false signal not having occurred before the (unknown) time of change τ (Kenett and Pollak, 2012). For this simulation, $\tau = 101$ is chosen. Using 250 simulations, the



(a) ACF of the estimates $\hat{\boldsymbol{\theta}}_t$ on the example of triangle, asymmetric and memory network terms.



(b) ACF of the estimates $\hat{\boldsymbol{s}}_t$ on the example of triangle, asymmetric and memory network terms.

Figure 4.1: Comparison of the Autocorrelation Function (ACF) values when the network characteristics are estimated with a sliding window approach of size $z = 7$ containing (left) and not containing (right) overlapping network states.

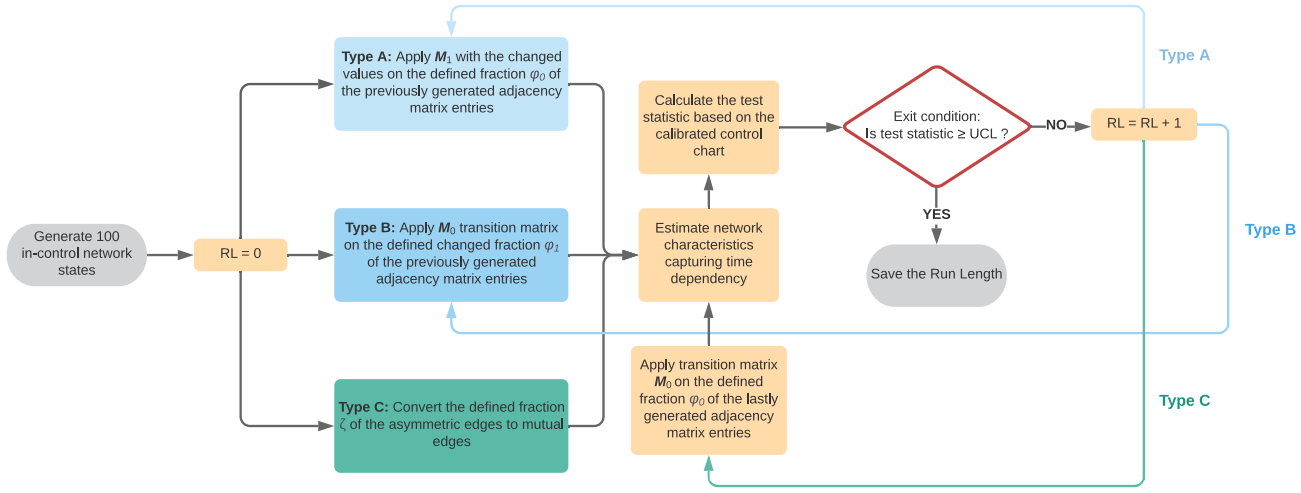


Figure 4.2: Anomaly types for the generation of observations in Phase II and calculation of the associated run length.

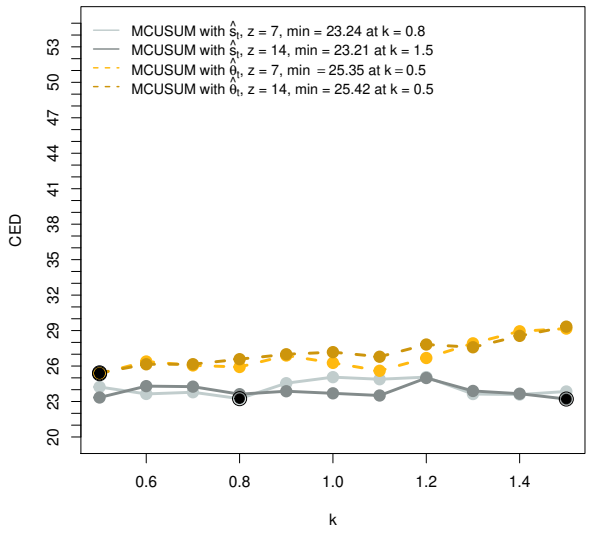
CED is estimated based on the *UCL* values with $ARL_0 = 50$ for each setting. That means one would expect $CED = 50$ if no change happened and it should be considerably smaller in the case of an anomaly. Figures 4.3, 4.4 and 4.5 present the results of the simulation for anomalies of Type A, B and C, respectively.

There are several aspects to assess fully the obtained results. First of all, the comparison of performance between the MCUSUM and the MEWMA control charts. In most of the cases, the *CED* of the MEWMA chart is smaller compared to the corresponding MCUSUM chart. However, for the best choice of the reference parameter k or the smoothing parameter λ, both charts are competitive. The respective values are indicated by the large dots indicating the minimum on the *CED* curve. For instance, the weakest change of Type A.1 (Figure 4.3, (a)) is detected quicker by the MCUSUM chart with the low parameters k. In contrast, the MEWMA charts perform better for bigger changes such as in Cases 2 and 3.

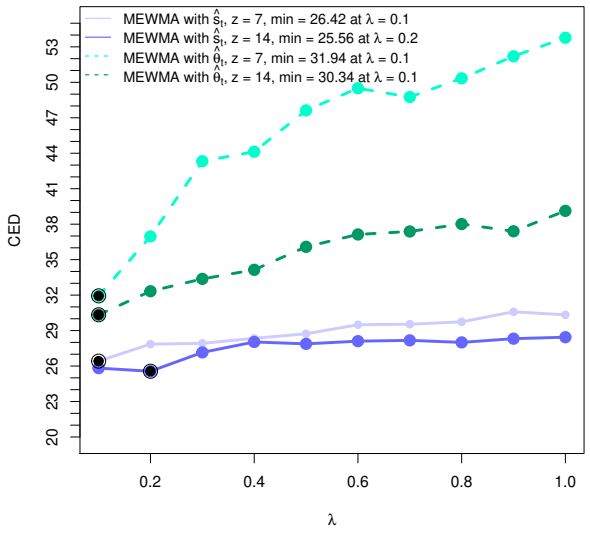
Generally speaking, one observes that the *CED* is decreasing if the shift size or the intensity of the change is increasing. Moreover, if the reference parameter k or the smoothing parameter λ is smaller, less intense anomalies can be detected. If in practical implementation the detection of larger changes is required, these parameters should also be higher. It is worth reminding that the MEWMA chart coincides with Hotelling's T^2 chart if $\lambda = 1$, *i.e.* the control statistic depends only on the current value.

The disadvantage of both approaches is that small and persistent changes are not detected quickly when the parameters k or λ are not optimally chosen. For example, considering Case A.1 in Figure 4.3 (b), one can notice that at the high values of the parameter λ the *CED* slightly exceeds the ARL_0 reflecting the poor performance. However, a careful selection of the parameters can overcome this problem. Also, the choice of the window size plays a significant role in detecting the anomalies reliably, being a trade-off between a precise description of the process and the ability to reflect the sudden changes in its behaviour.

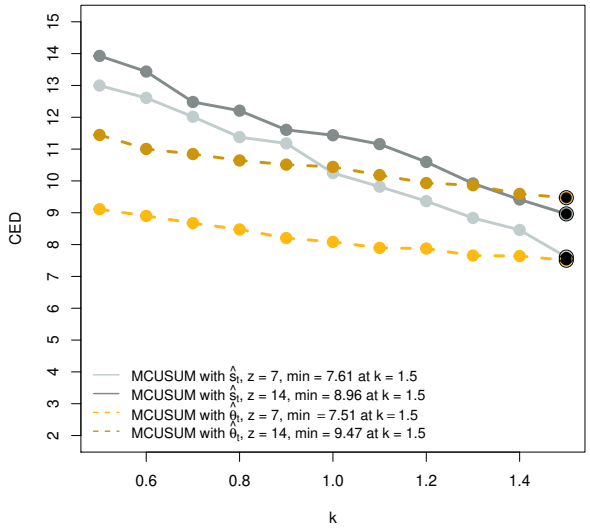
Regarding the differences in results with respect to the quantities $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{s}}_t$, a similar performance is noticed in Anomaly Types A and B. It is interesting that in most of the cases, the MEWMA control charts work better for $\hat{\boldsymbol{s}}_t$ and the CUSUM control charts for $\hat{\boldsymbol{\theta}}_t$. However, looking at the detection of anomaly Type C.2, one notes a considerable advantage of applying $\hat{\boldsymbol{\theta}}_t$ rather than $\hat{\boldsymbol{s}}_t$.



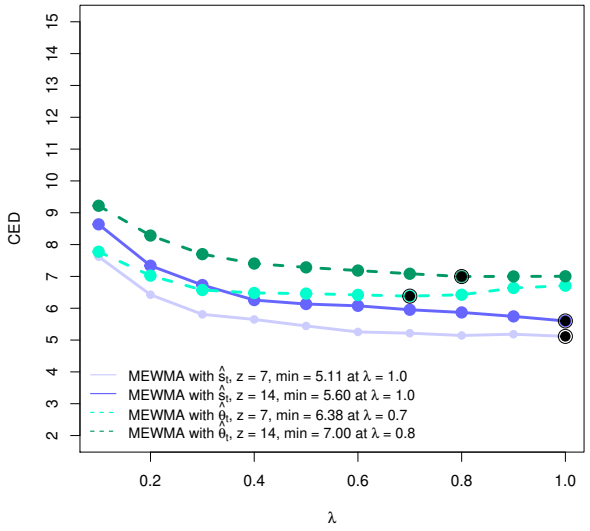
(a) MCUSUM, Case A.1



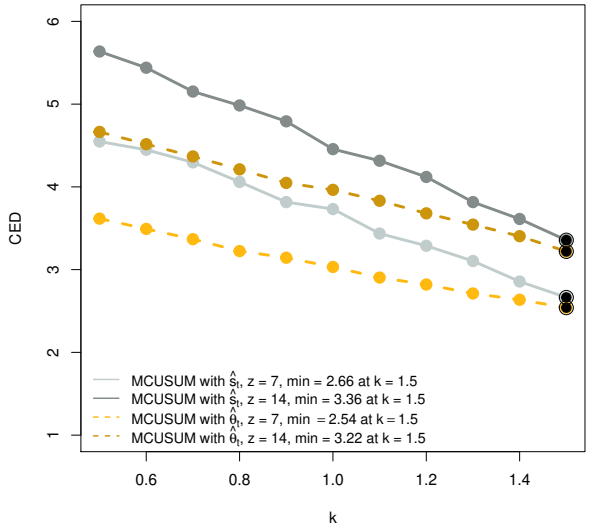
(b) MEWMA, Case A.1



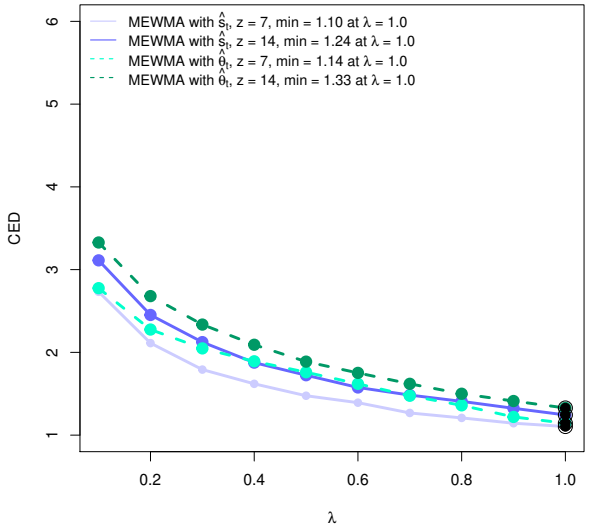
(c) MCUSUM, Case A.2



(d) MEWMA, Case A.2

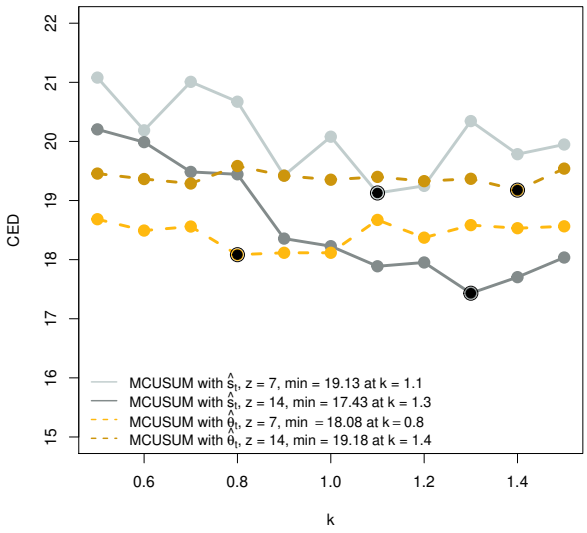


(e) MCUSUM, Case A.3

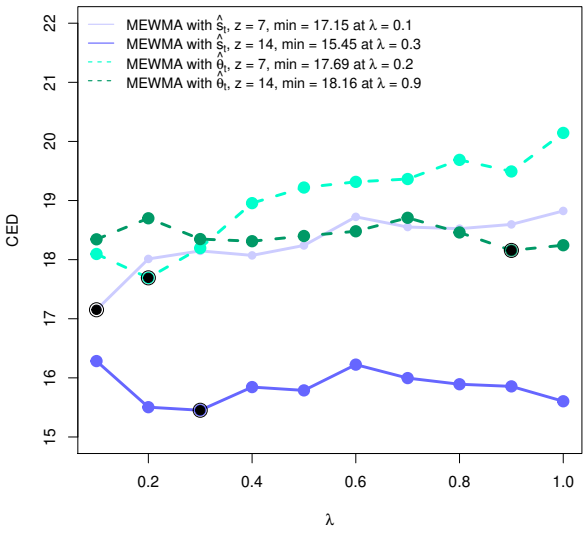


(f) MEWMA, Case A.3

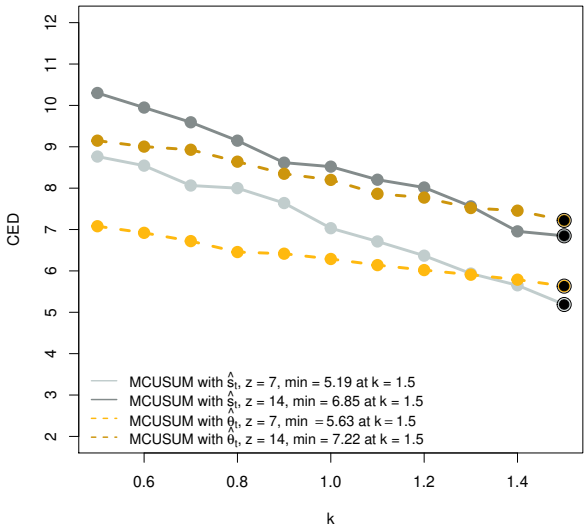
Figure 4.3: Conditional expected delays for anomalies of Type A for MCUSUM (left) and MEWMA (right) together with the different choices of the reference parameter k and the smoothing parameter λ, the window sizes $z = 7$ and $z = 14$, and the network estimates $\hat{\mathbf{s}}_t$ (solid lines) and $\hat{\boldsymbol{\theta}}_t$ (dashed lines). Black points indicate the minimum CED for each setting.



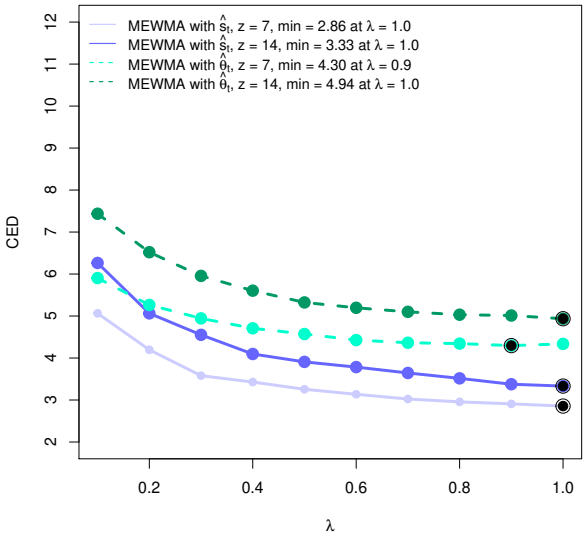
(a) MCUSUM, Case B.1



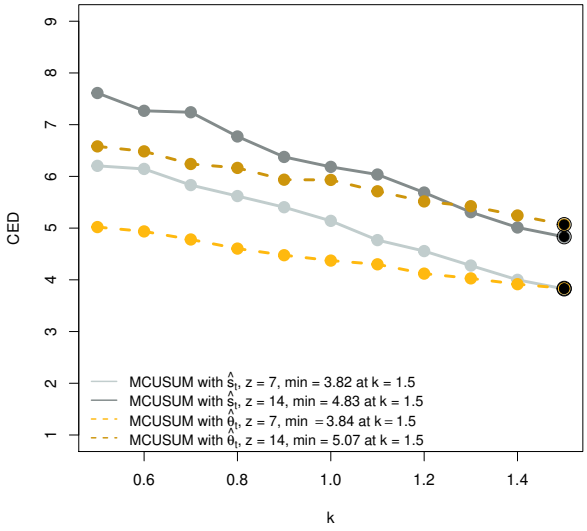
(b) MEWMA, Case B.1



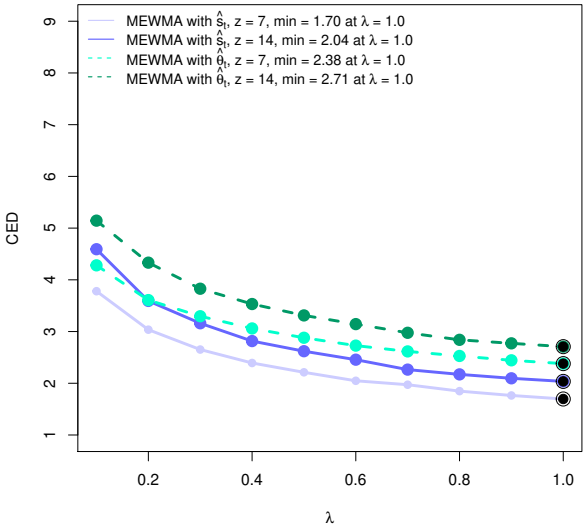
(c) MCUSUM, Case B.2



(d) MEWMA, Case B.2

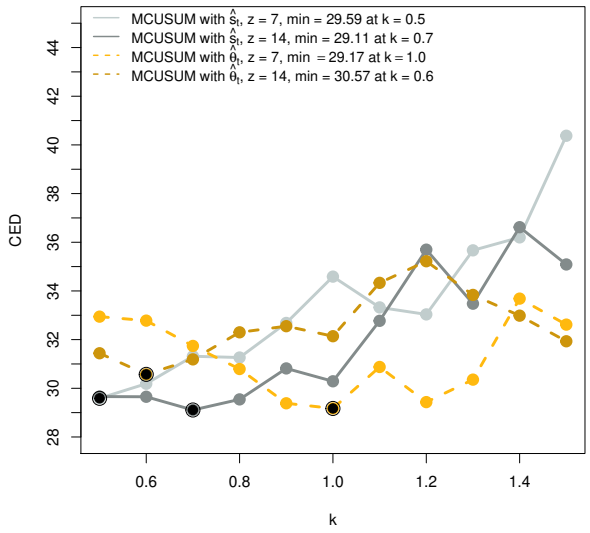


(e) MCUSUM, Case B.3

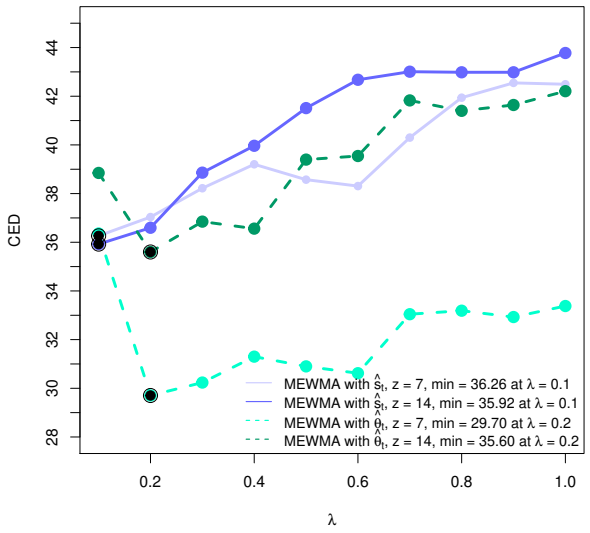


(f) MEWMA, Case B.3

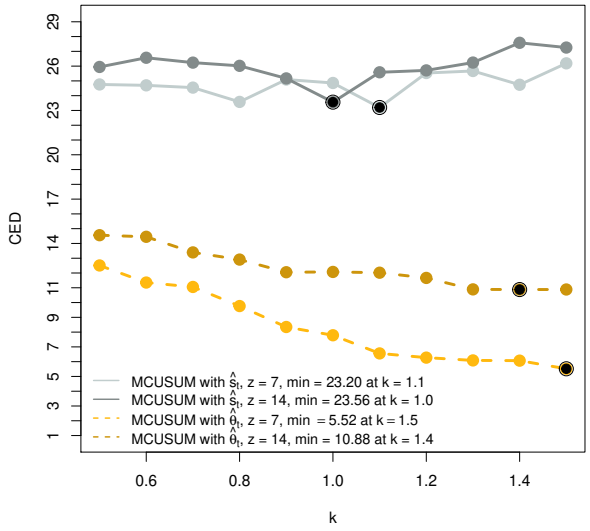
Figure 4.4: Conditional expected delays for anomalies of Type B for MCUSUM (left) and MEWMA (right) together with the different choices of the reference parameter k and the smoothing parameter λ, the window sizes $z = 7$ and $z = 14$, and the network estimates $\hat{\boldsymbol{s}}_t$ (solid lines) and $\hat{\boldsymbol{\theta}}_t$ (dashed lines). Black points indicate the minimum CED for each setting.



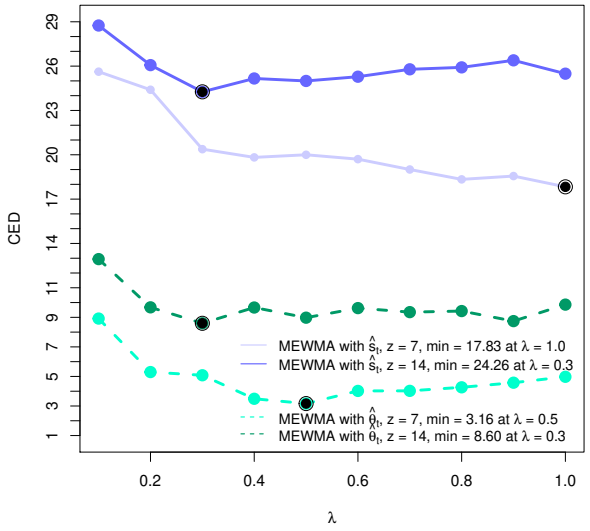
(a) MCUSUM, Case C.1



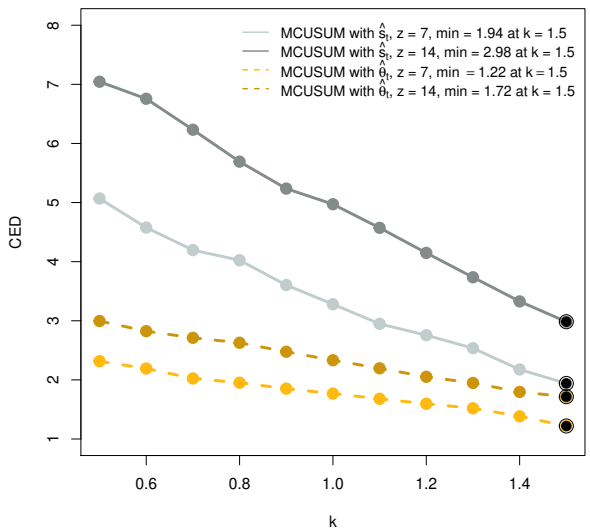
(b) MEWMA, Case C.1



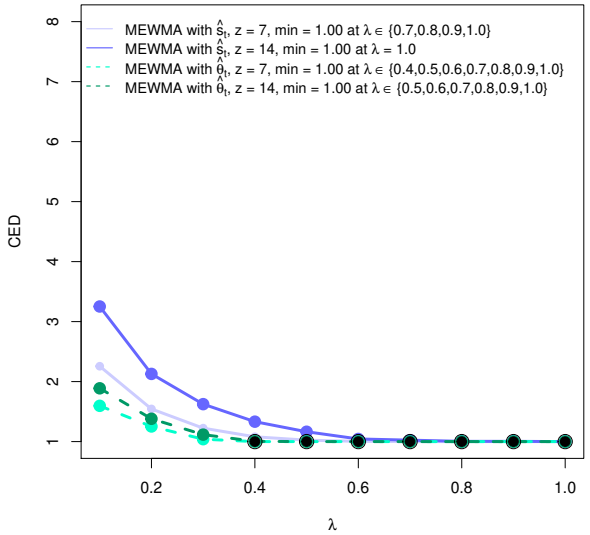
(c) MCUSUM, Case C.2



(d) MEWMA, Case C.2



(e) MCUSUM, Case C.3



(f) MEWMA, Case C.3

Figure 4.5: Conditional expected delays for anomalies of Type C for MCUSUM (left) and MEWMA (right) together with the different choices of the reference parameter k and the smoothing parameter λ, the window sizes $z = 7$ and $z = 14$, and the network estimates $\hat{\boldsymbol{s}}_t$ (solid lines) and $\hat{\boldsymbol{\theta}}_t$ (dashed lines). Black points indicate the minimum CED for each setting.

	CED	$\hat{\boldsymbol{s}}_t$				$\hat{\boldsymbol{\theta}}_t$			
Case		C.1	C.2		C.3	C.1	C.2		C.3
Parameter ζ		**0.005**	**0.01**	**0.02**	**0.05**	**0.005**	**0.01**	**0.02**	**0.05**
MEWMA with $z=7$	Min.	36.26	17.83	1.85	<u>1.00</u>	<u>29.70</u>	<u>3.16</u>	<u>1.00</u>	<u>1.00</u>
	λ_{min}	0.1	1.0	1.0	0.7	0.2	0.5	0.9	0.4
	Max.	42.55	25.62	7.16	2.26	36.36	8.92	2.56	1.60
	λ_{max}	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1
MEWMA with $z=14$	Min.	35.92	24.26	3.57	<u>1.00</u>	35.60	8.60	<u>1.00</u>	<u>1.00</u>
	λ_{min}	0.1	0.3	0.9	1.0	0.2	0.3	1.0	0.5
	Max.	43.78	28.75	9.90	3.25	42.21	12.94	3.49	1.89
	λ_{max}	1.0	0.1	0.1	0.1	1.0	0.1	0.1	0.1
MCUSUM with $z=7$	Min.	29.59	23.20	6.66	1.94	29.17	<u>5.52</u>	<u>2.10</u>	<u>1.22</u>
	k_{min}	0.5	1.1	1.5	1.5	1.0	1.5	1.5	1.5
	Max.	40.38	26.19	18.70	5.07	33.68	12.51	3.91	2.32
	k_{max}	1.5	1.5	0.5	0.5	1.4	0.5	0.5	0.5
MCUSUM with $z=14$	Min.	<u>29.11</u>	23.56	9.45	2.98	30.57	10.88	3.14	1.72
	k_{min}	0.7	1.0	1.5	1.5	0.6	1.4	1.5	1.5
	Max.	36.62	27.58	18.86	7.04	35.22	14.56	6.19	3.00
	k_{max}	1.4	1.4	0.5	0.5	1.2	0.5	0.5	0.5

Table 4.6: Summary of the CED results to detect anomalies of Type C with the additional test case $\zeta = 0.02$. The corresponding smoothing and reference parameters λ and k are provided under the respective CED. The minimum CED for each case and the control chart group are underlined. The maximum CED represents the "worst-case" scenario. In case several values of the parameter λ correspond to the CED result, only the smallest value is reported.

To confirm that this behaviour is supported by another example, an additional test case with $\zeta = 0.02$ is created. These results as well as the others from Type C anomalies are summarised in Table 4.6. As one can observe, if the change is too small, then both groups of control charts created on the basis of $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{s}}_t$ need relatively long to detect it. In the case when $\zeta = 0.05$, representing a substantial anomaly, the change is identified quickly by both options. However, when the change is of a moderate degree, for example, $\zeta = 0.02$, then the control charts based on $\hat{\boldsymbol{\theta}}_t$ signal the anomalous behaviour considerably quicker. Whether the main reason for such difference is the particular type of anomaly, namely it is an example of a point change, cannot be said generally as additional variations of such anomalies should be examined. However, from the obtained evidence, the hypothesis is that the estimates $\hat{\boldsymbol{\theta}}_t$ might be more suitable for general network monitoring when it is assumed that a point as well as a persistent change can occur. Nevertheless, the comparison between the performance of $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{s}}_t$ is worth further investigation.

To summarise, the effectiveness of the presented charts in detecting structural changes depends significantly on the accurate estimation of the anomaly size one aims to detect. Thus, to ensure that no anomalies are missed, it can be effective to apply paired charts and benefit from the strengths of each of them to detect varying types and sizes of anomalies if the information on the possible change is not available or not reliable.

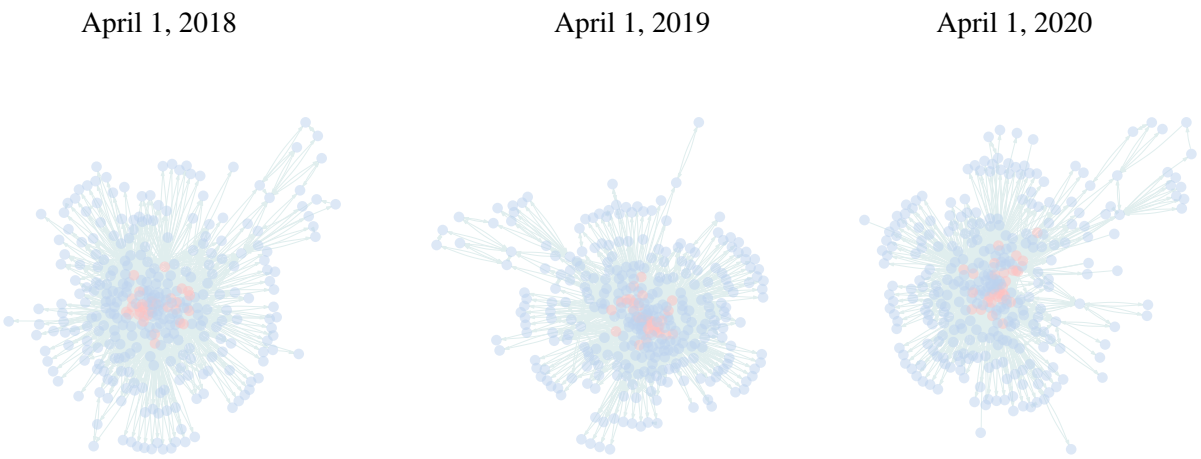


Figure 4.6: Illustration of the flight network on April 1 of each year excluding isolated vertices. It can be seen that the topology of the network has changed. The red coloured nodes represent the 30 busiest airports.

4.5 Empirical Illustration

To demonstrate the applicability of the described method, the daily flight data of the United States (US) published by the US Bureau of Transportation Statistics is monitored using the parameter estimates $\hat{\boldsymbol{\theta}}_t$. Each day can be represented as a directed network, where nodes are airports and directed edges define flights between airports. In Figure 4.6, examples of flight network data in 2018, 2019 and 2020 are presented.

Flexibility in choosing the network terms, according to the type of anomalies one would like to detect, enables different perspectives on the same network data. In this section, one aims to identify considerable changes in network development. The intuition of how the flight network usually operates guides the choice of its terms. Due to the Coronavirus disease (COVID-19) pandemic in the year 2020, some regions have paused the operation of transport systems with the aim of reducing the number of new infections. However, the providers enabled mobility by establishing connections through territories which allow travelling. That means, instead of having a direct journey from one geographical point to another, the route passes through several locations, which can be interpreted as nodes. Thus, the topology of the graph has changed: instead of directed mutual links, the number of intransitive triads and asymmetric links starts to increase significantly. One can incorporate both terms, together with the edge term and a memory term ($v = 1$), and expect the estimates of the respective coefficients belonging to the first two statistics to be close to zero or strongly negative in the in-control case.

Initially, one needs to decide which data is suitable to define observations coming from Phase I, *i.e.* the in-control state. There were no considerable events which would seriously affect the US flight network in the year 2018, therefore, this year is chosen to characterise the in-control state. Consequently, the years 2019 and 2020 (until the end of April) represent Phase II. To capture the weekly patterns, a time window of size $z = 7$ was chosen, so that the first instant of time when the monitoring begins represents January 8, 2018. In this case, Phase I consists of 358 observations and Phase II of 486 observations. To guarantee that only factual flight data are considered, cases when a flight was cancelled are removed. Additionally, multiple edges are eliminated. The main descriptive statistics for Phase I and II are reported in Table 4.7. There are no obvious changes

		2018	2019	2020
Phase		I	II	II
Number of nodes		358	360	354
Density	Min.	0.031	0.033	0.022
	Median	0.037	0.038	0.038
	Max.	0.039	0.040	0.041
Reciprocity	Min.	0.97	0.96	0.89
	Median	0.99	0.99	0.99
	Max.	1.00	1.00	1.00
Transitivity	Min.	0.315	0.322	0.263
	Median	0.339	0.339	0.326
	Max.	0.357	0.354	0.345

Table 4.7: Descriptive statistics of the flight network data. Density is calculated on networks without multiple edges.

when considering the descriptive statistics. Hence, control charts, which are only based on such characteristics, could fail to detect the possible changes in 2019 and 2020. When considering the estimates $\hat{\boldsymbol{\theta}}_t$ of the TERGM described by a series of boxplots in Figure 4.7, one can observe substantial changes in the values.

Before proceeding with the analysis, it is important to evaluate whether a TERGM fits the data well (Hunter et al., 2008). For each of the years, one period of the length z is randomly selected and 500 networks are simulated based on the parameter estimates from each of the corresponding networks. Figure 4.8 depicts the results for the time frame April 3-9, 2019, where the grey boxplots of each of the statistics represent the simulations, and the solid black lines connect the median values of the observed networks. Despite the relatively simple definition of the model, some typical network characteristics such as the distributions of edge-wise shared partners, the vertex degrees, various triadic configurations (triad census) and geodesic distances (the value of infinity replicates the existence of isolated nodes) match the observed distributions of the same statistics satisfactory.

To select appropriate control charts, one needs to take into consideration the specifications of the flight network data. Firstly, it is common to have 3-4 travel peaks per year around holidays, which are not explicitly modelled, so that one can detect these changes as verifiable anomalous patterns. It is worth noting that one could account for such seasonality by including nodal or edge covariates. Secondly, as one aims to detect considerable deviations from the in-control state, one is more interested in sequences of signals. Thus, $k = 1.5$ is chosen for MCUSUM and $\lambda = 0.9$ for the MEWMA chart. The target ARL_0 is set to 100 days, therefore, one can expect roughly 3.65 in-control signals per year by the construction of the charts.

Figure 4.9 depicts the results of both charts for monitoring the US flight network data. In Phase I there are slightly more in-control signals than expected, which are left without investigation as they occur as single instances. Considering Phase II, there are several anomalous behaviours which were detected. The first series of signals in summer 2019 is due to a particularly increased demand for flights during the holidays. The second sequence of signals corresponds to the development of the COVID-19 pandemic. On March 19, the State Department issued a Level 4 “do not travel” advisory, recommending that United States citizens avoid any global travel. Although this security measure emphasises international flights, it also influences domestic aerial connections. The continuous sequence of the signals in the case of the MEWMA begins on March 21, 2020.

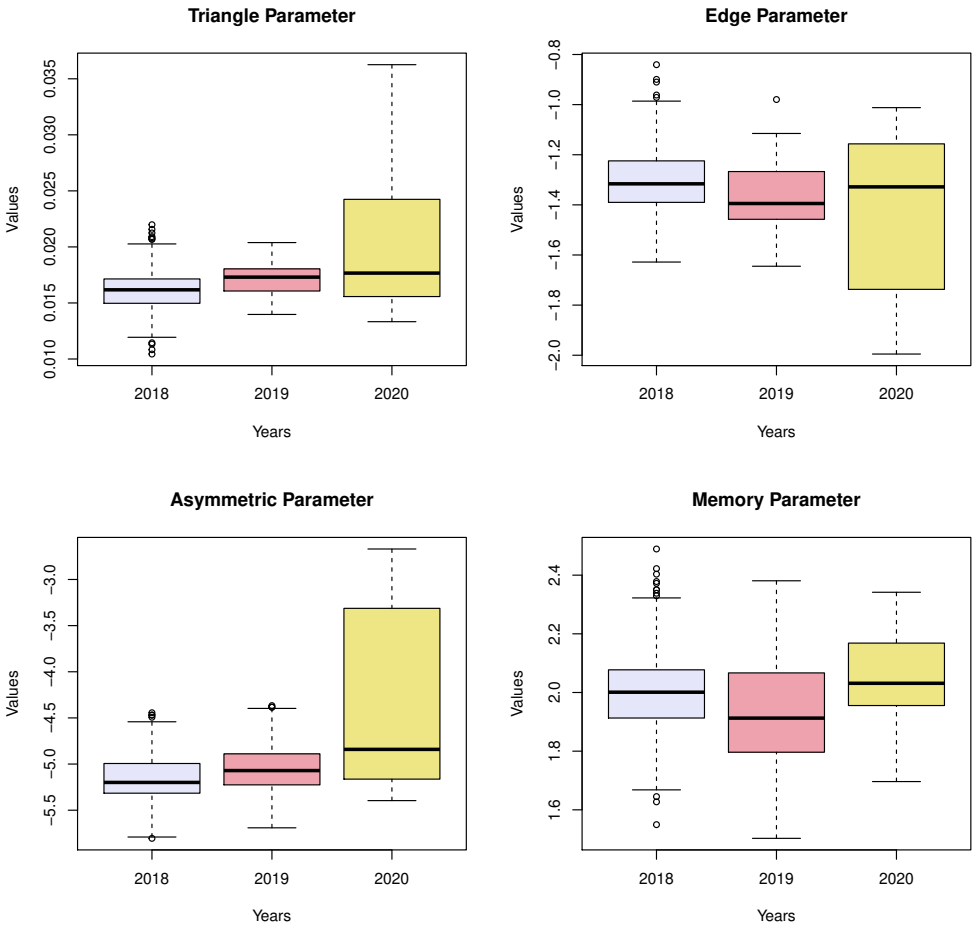


Figure 4.7: Distribution of the estimated coefficients $\hat{\boldsymbol{\theta}}_t$ in 2018, 2019 and 2020.

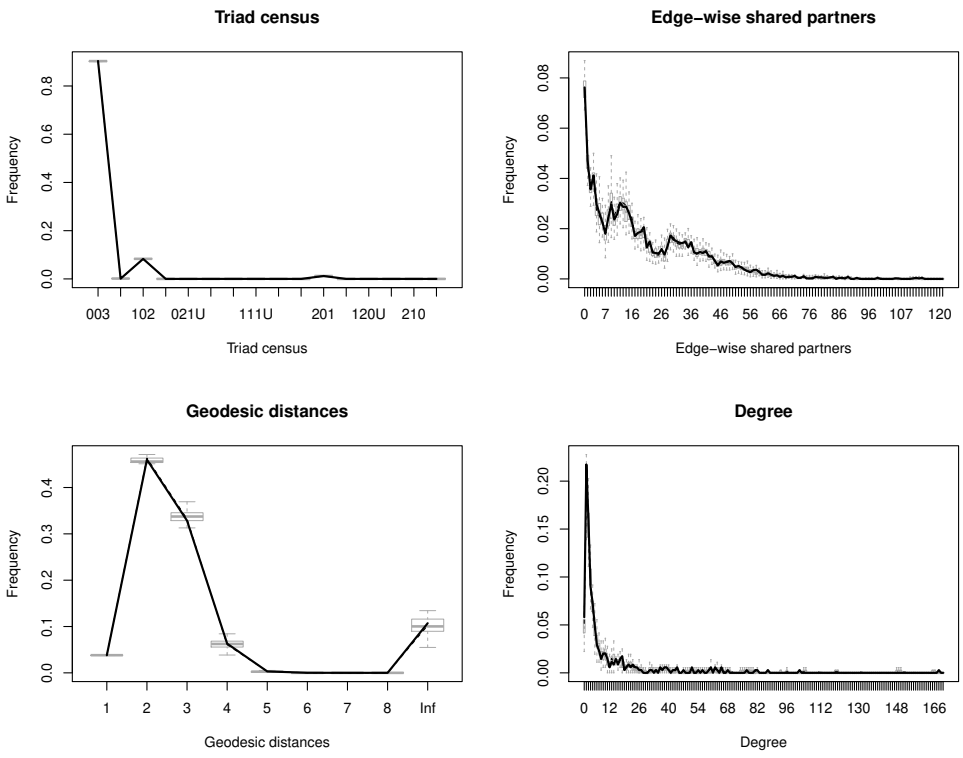


Figure 4.8: Illustration of the goodness of fit assessment for the TERGM. The considered networks belong to the period April 3-9, 2019.

In the case of the MCUSUM, the start is on March 24. Although in both cases the control statistic resets to zero after each signalling, the repeated violation of the upper control limit is a clear indicator of this shift in network behaviour.

To identify smaller and more specific changes in the daily flight data of the US, one could also integrate nodal and edge covariates, which would refer to further aspects of the network. Additionally, control charts with smaller k and λ can be applied.

4.6 Discussion

In this chapter, the approach of how multivariate control charts can be used to detect changes in dynamic networks of various types generated by the TERGM in an online manner is developed. This monitoring procedure allows for many applications in different disciplines which are interested in analysing networks of medium sizes, such as sociology, political science, engineering, economics and psychology (cf. Carrington et al., 2005; Ward et al., 2011; Das et al., 2013; Jackson, 2015; Fonseca-Pedrero, 2018).

Despite the benefits of the TERGM, such as the incorporation of the temporal dimension and representation of the network in terms of its sufficient statistics, there are several considerable drawbacks. Other than the difficulty in determining a suitable combination of the network terms, the model is not applicable to networks of large size (Block et al., 2018). Furthermore, the temporal dependency statistics in the TERGM depend on the selected temporal lag and the size of the time window over which the data is modelled (Leifeld and Cranmer, 2019). Thus, the accurate modelling of the network strongly relies on the analyst's knowledge about its nature.

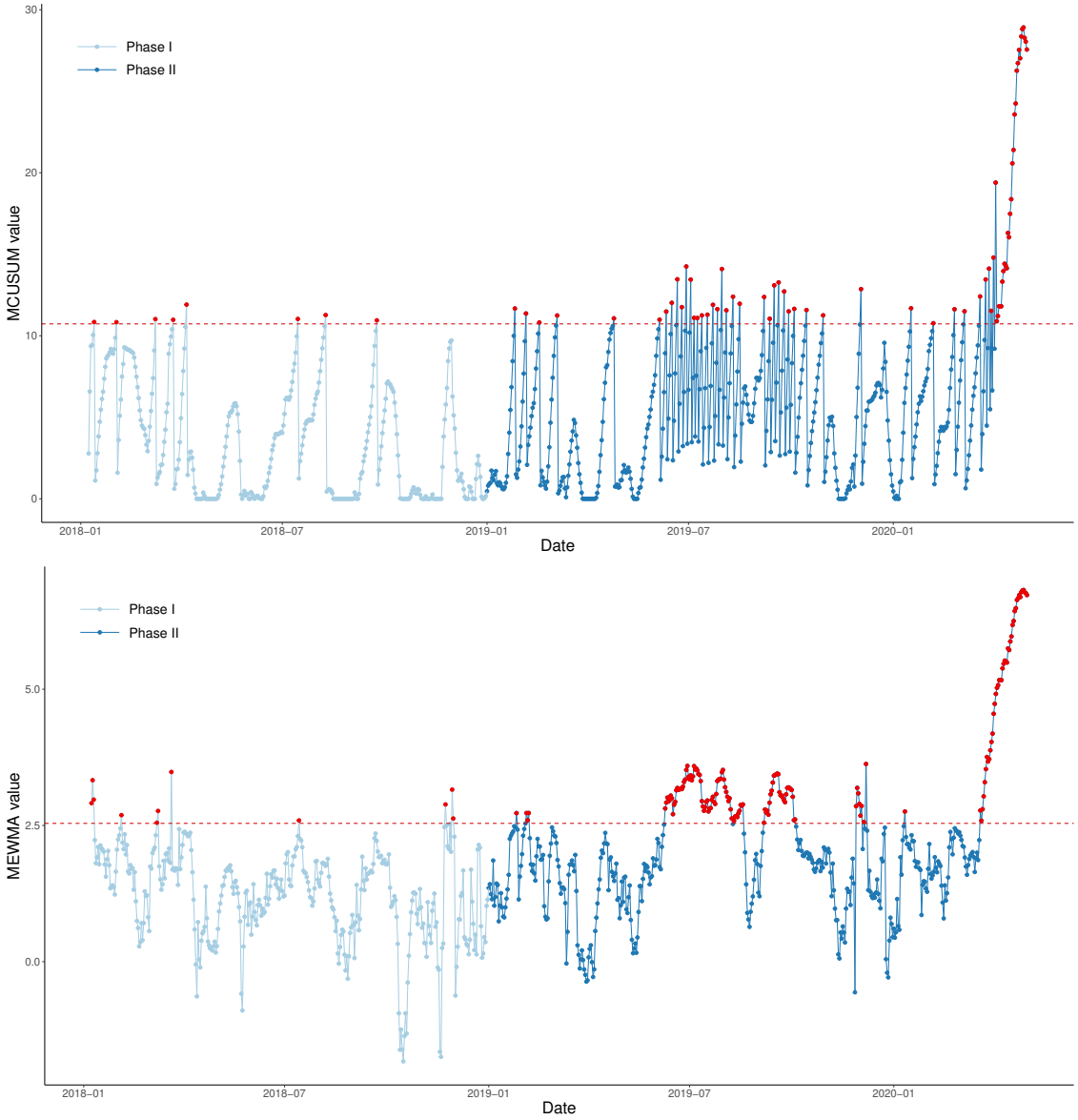


Figure 4.9: The MCUSUM control chart (above) and the logarithmic MEWMA control chart (below). The horizontal red line corresponds to the upper control limit and the red points to the occurred signals.

A helpful extension of the approach would be the implementation of the STERGM. In this case, it could be possible to subdivide the network monitoring into two distinct streams, so that the interpretation of changes in the network would become clearer. Another topic that demands consideration is the determination of cases when it is reliable to use the averaged network statistics $\hat{\boldsymbol{s}}_t$ to construct the monitoring procedure and not the parameter estimates $\hat{\boldsymbol{\theta}}_t$, as their calculation is more complex than of $\hat{\boldsymbol{s}}_t$. Also, it could be beneficial to consider other estimators to compute $\hat{\boldsymbol{s}}_t$ and compare their effectiveness in detecting anomalies.

5 Monitoring of Networks with Fixed Structure

This chapter[1] is devoted to the modelling and monitoring of networks with fixed structures across time. A surveillance method is developed for processes that happen on a network to detect significant changes related to the network itself. In other words, this chapter is dedicated to the contribution in the field of *fixed network monitoring*.

5.1 Main Intent

Currently, the modelling and monitoring of networks are usually focused on detecting changes and anomalies reflected in the geometric properties of a graph, falling under the type of *random network monitoring*. Illustrative studies include the monitoring of e-mail communication (Perry, 2020) or the daily flights within a country presented in the previous chapter, Section 4.5 for the sake of detecting unusual connectivity patterns. However, some networks might also reach a point of saturation. That means, the development of connections or inclusion of new nodes is either no longer possible (for instance, due to physical or geopolitical constraints) or would not be of considerable monitoring interest anymore due to the context or regulations that hinder its further expansion or interconnectivity. Thus, what becomes of interest and great importance are processes that happen within or on the network, basically the information along the connection. This is denoted as an *attributed network*. The monitoring of a process happening on a network might reveal whether the network remains in control, *i.e.* in the state that corresponds to a known standard, or whether it experiences abnormality.

The type of network considered in this chapter is an attributed network where the monitored process takes place on fixed edges and can be seen as an edge covariate. This type of monitoring is considered an emerging research direction (Jeske et al., 2018) and offers advancement in the field of *fixed network monitoring* for such cases as threat detection in information technology networks (cf. Stevens et al., 2021b). Due to a particular concentration on the variables attributed to the network edges which are observed over time, from now on the reference to those networks is Temporal Edge Network (TEN) processes.

In general, the integration of nodal or edge attributes for improving the modelling of the underlying network formation mechanism and in turn the network monitoring itself is not a new concept (Azarnoush et al., 2016; Shaghaghi and Saghaei, 2020). However, in existing research, the attributes are considered to regulate the presence or absence of an edge, discovering a different angle of how a network structure arises. In other words, the contextual information available either on vertices or edges is viewed as an extra dimension to the graph helping to explain the likelihood of the observed links (Miller et al., 2013). In contrast, the process happening on edges is considered as the primary or only source of information about the network state.

[1] This chapter is based on the preprint *Malinovskaya, A., Killick, R., Leeming, K., Otto, P. Statistical monitoring of European cross-border physical electricity flows using novel temporal edge network processes. arXiv preprint: 2312.16357, (2023).* Available online: `https://doi.org/10.48550/arXiv.2312.16357`.

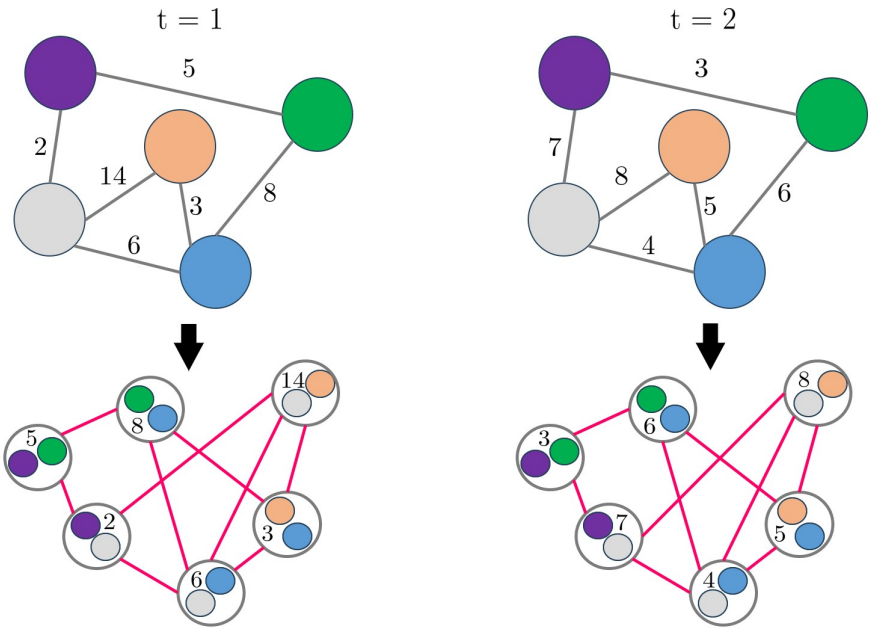


Figure 5.1: Visual illustration of a TEN process (top) for two time points with an adapted representation (bottom) described in Section 5.3.1.1.

For network monitoring, a model and a monitoring procedure should be defined. For modelling, as described in Section 3.2, there are recently developed models for modelling time series with an underlying network dependency structure, however, their primary focus lies on nodal response variables. Thus, to use the Generalised Network Autoregressive model with time-dependent exogenous covariates (GNARX) as introduced in Section 5.3.1, one needs to adapt the representation of TEN processes (see Section 5.3.1.1). For monitoring, it is vital to determine the particular framework in which it is going to be performed. One possibility is to monitor the estimates of the model parameters although this is computationally intensive as discussed in Section 4.6. A more efficient alternative is to perform monitoring based on the residuals. For instance, Miller et al. (2013) compute graph residuals as the difference between the observed graph and its expected value. According to Alexopoulos et al. (2004), who offer a detailed introduction to SPM and especially to the forecast-based monitoring methods, when the model or prediction is accurate, the prediction errors are uncorrelated. Thus, one can apply traditional SPM techniques such as control charts to these forecast errors. A technique that is particularly suitable for the monitoring part of the proposed framework is a residual-based control chart described in Section 5.3.2.

5.2 Definition of the Change Point

Before proceeding with the definition of the change point in the considered setting, the notation of the temporal edge network is specified. Consider a network $G = (V,E)$, where the elements of V represent vertices (or nodes) and E – edges (or links). Further, a fixed structure of G over time is assumed which is described by an adjacency matrix $\boldsymbol{Y} := (Y_{ij})_{i,j=1,\ldots,|V|}$, with $|V|$ being the number of nodes. To each existing edge $e \in E$ a time series $\{x_{e,t}\}$, where $t = 1,\ldots,T$, being the attributed process of G, is related. The complete representation $\boldsymbol{X} = (X_{e,t})_{e=1,\ldots,|E|,t=1,\ldots,T}$, where $|E|$ denotes the number of edges, and the graph G form a Temporal Edge Network (TEN) process illustrated in Figure 5.1 (top) for two time stamps.

It is beneficial to distinguish general random processes one regards on the fixed network structures from specific areas of analysis of processes on networks. The first well-established perspective is the analysis of spatial point patterns on networks (cf. Baddeley et al., 2021), where the accurate location of the object on the physical network is of importance. The second area concerns the analysis of network flows (cf. Ahuja et al., 1993; Kolaczyk, 2009a), where the physical constraints of a network structure and the flow itself play a vital role. In this work, however, one could think of the analysed process as being a collection of flows with no associated constraints unless they are imposed explicitly.

As the monitoring part is based on the residuals obtained by finding the difference between the actual process and the predictor at each flow, it is vital to obtain an accurate model before the monitoring starts. The objective is then to test

$$H_{0,t} : \text{The observed TEN process coincides with the fitted GNARX model}$$

against the alternative

$$H_{1,t} : \text{The observed TEN process does not coincide with the fitted GNARX model}$$

for each t. Now the question arises which variable to use for the test statistic that could provide the information about when H_0 is violated; therefore, the change point τ is defined as

$$\boldsymbol{x}_t \quad \sim \quad \begin{cases} F(\boldsymbol{\mu}, \Sigma) & \text{if } t < \tau \\ F_\tau(\boldsymbol{\mu}_\tau, \Sigma_\tau) & \text{if } t \geq \tau, \end{cases}$$

where $\boldsymbol{x}_t = (x_{1,t}, \ldots, x_{|E|,t})'$, $\boldsymbol{\mu}$ and Σ define the mean vector and variance-covariance matrix of the network flow distribution F, respectively. The change in mean or/and in variance of the raw data would lead to the respective changes in the forecast errors (Grundy, 2021). Hence, by constructing a test statistic based on $\boldsymbol{u}_t = (u_{1,t}, \ldots, u_{|E|,t})'$, one can accordingly determine the time point τ when the change has occurred. Following, the change point detection framework specified by Grundy (2021) is introduced and the respective test statistic for applying residual-based Cumulative Sum (CUSUM) control chart is discussed.

5.3 Monitoring Framework

After the introduction of the original GNARX model in Section 5.3.1, its extension from nodal time series to time series on network edges is presented in Section 5.3.1.1. Subsequently, the residual-based CUSUM control chart as well as the monitoring scope are specified in Section 5.3.2.

5.3.1 Generalised Network Autoregressive Model with Time-Dependent Exogenous Variables

In this section, consider $\{x_{i,t}\}$ to be a time series related to each node i. The GNARX model $(p, \boldsymbol{s}, \boldsymbol{q})$ with H exogenous regressors $\{z_{h,i,t:h=1,\ldots,H,i\in V,t=1,\ldots,T}\}$ and the autoregressive order p, where $(p, \boldsymbol{s}, \boldsymbol{q}) \in \mathbb{N} \times \mathbb{N}_0^p \times \mathbb{N}_0^H$ holding for all vertices $i \in V$, is specified as

$$x_{i,t} = \sum_{l=1}^{p} \left(\alpha_{i,l} x_{i,t-l} + \sum_{r=1}^{s_l} \beta_{l,r} \sum_{j \in N^{(r)}(i)} \omega_{i,j} x_{j,t-l} \right) + \sum_{h=1}^{H} \sum_{q=0}^{q_h} \gamma_{h,q} z_{h,i,t-q} + \varepsilon_{i,t}. \tag{5.1}$$

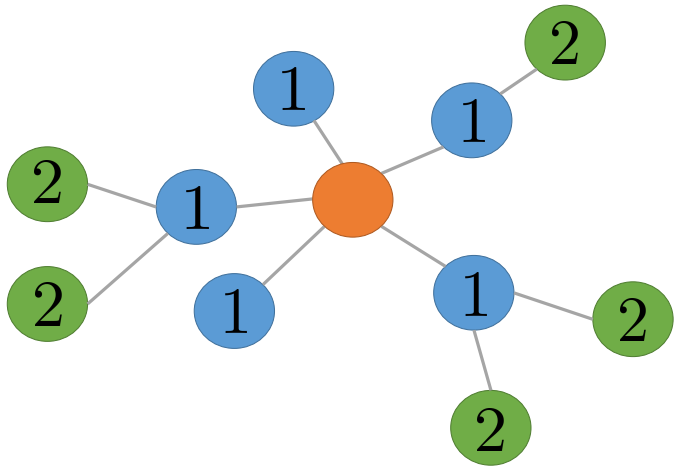


Figure 5.2: Stage-1 (blue) and stage-2 (green) neighbourhood of the orange node.

The order p also determines the maximum order of neighbour time lags, *i.e.* $\boldsymbol{s} = (s_1, \ldots, s_p)$ with s_l being the maximum stage of neighbour dependence for time lag l. For example, $s_1 = 2$ means that nodes depend on their first and second-stage neighbours in G in the first time lag. Similarly, the maximum time lag of the h-th exogenous regressor $\{z_{h,i,t}\}$ is defined as q_h and collectively as $\boldsymbol{q} = (q_1, \ldots, q_H)$. In case $q_h = 0$, the current value of the exogenous variable is considered at time point t.

The noise is denoted by $\varepsilon_{i,t}$ and is assumed to be independent and identically distributed at each vertex i with mean zero and variance σ_i^2. The parameters $\alpha_{i,l}$, $\beta_{l,r}$, $\gamma_{h,q} \in \mathbb{R}$ define autoregressive influence, neighbouring influence and external influence from regressors, respectively. It is possible to estimate a global-α model, where $\alpha_{i,l} = \alpha_l$, assuming the same autoregressive process for all nodes.

The set $N^{(r)}(i)$ denotes the r-th stage neighbourhood set of node $i \in G$. For instance, the stage-1 neighbours of a node $i \in V$ are the adjacent nodes $j \in V$, connected by an edge. Further, the stage-2 neighbourhood set of a node $i \in V$ is the stage-1 neighbours of the adjacent nodes $j \in V$ as can be seen in Figure 5.2. Moreover, there are weights $\omega \in [0,1]$ associated with every pair of nodes that, in this case, depend on the size of the neighbour set and are usually set as the inverse of some prior notion of distance between vertices as explained by Knight et al. (2020).

By fitting the GNARX model, one obtains the estimates of the parameters $\alpha_{i,l}$, $\beta_{l,r}$, $\gamma_{h,q}$ that can be used for predicting the $\hat{x}_{i,t+1}$ value from $\{x_{i,t}\}$. The one-step-ahead forecast errors are then determined as

$$u_{i,t+1} = x_{i,t+1} - \hat{x}_{i,t+1} \tag{5.2}$$

which can be utilised in the proposed monitoring framework. However, the GNARX model is defined for time series related to nodes. Thus, one needs to extend the representation of TEN processes to enable a correct application of the GNARX model for performing the monitoring.

5.3.1.1 Extension to TEN Processes

There are different ways of thinking about why an alternative representation is required. First, the main focus lies in discovering changes happening between and within the streams, *i.e.* a process

captured on the edges. Thus, one actually indirectly considers it to be a network not of single nodes anymore but of pairs of nodes. Second, the influence from node to node and the influence from one stream to its neighbouring stream portray two distinct standpoints. In the case of dependency between two flows, more than two nodes are involved so that other underlying mechanisms determine the exchange and its strength. Therefore, one needs a different adjacency matrix to describe this novel dependency structure.

For distinguishing between G and $\boldsymbol{Y}$, and the novel representation of the TEN process as G', the notation for the new adjacency matrix is introduced as $\boldsymbol{Y}'$. Now, as displayed in Figure 5.1 (below), the edges $e \in E$ from the original graph G become vertices $\iota \in V'$ with the number of nodes being $|V'| = |E|$ so that the time series $\{x_{e,t}\}$ placed on edges $e \in E$ are now placed on the nodes, becoming $\{x_{\iota,t}\}$. Consequently, one would also have new edges $\xi \in E'$. Considering the connectivity structure captured by the adjacency matrix, without any expert knowledge, the natural choice for creating $\boldsymbol{Y}'$ would be to use a model for generating random graphs. A potential candidate is Erdős-Rényi model, where all graphs on a fixed vertex set with a fixed number of edges are equally likely. Another possibility is to consider sampling from a stochastic block model which produces graphs with a community structure. For example, links within a community may be more probable than between communities. Both of those options are tested in Section 5.4.

There is, however, an additional approach in choosing a new connectivity structure by referring to other types of graphs from graph theory. In this case, a deterministic approach to construct $\boldsymbol{Y}'$ is used. One relevant representation technique is the line (also known as edge-to-vertex dual) graph $L(G)$ of G. As introduced in Section 2.1.1, $L(G)$ is constructed on E, where $e \in E$ are adjacent as nodes $\iota \in V'$ if and only if they share a common node as edges in G (Diestel, 2017). In Figure 5.1 (bottom) this connectivity structure is adapted to the original TEN process shown above. Such formation can suit well real-world applications as it offers a more logical structure in case of limited knowledge about any existing communities or insufficiency of a connectivity structure generated from a random graph model.

After redefining the representation of TEN processes, one can proceed with their modelling by applying the GNARX model and monitoring using a residual-based control chart introduced in the next section.

5.3.2 Residual-based Cumulative Sum Control Chart

In real-world applications, it is usually unknown which properties, *e.g.* mean or variance, of a process may change. Thus, an important criterion for selecting a suitable control chart is its ability to track many types of change simultaneously. Another point considers the decision about what exactly to monitor – the data itself or some process related to it. As discussed by Grundy (2021), the monitoring of the data directly might deteriorate in case of temporal dependency, complex trends or seasonality effects within its structure. However, monitoring forecast residuals of the process omits these issues and is capable of reflecting changes in either the process mean or the process variance.

As under the real settings it is usually unknown whether the change occurs in a mean and/or variance of the process, a more general type of Page's CUSUM detector is utilised, which is based on the (centred) squared data and is able to detect a combination of the mean and/or variance change in the original data. For monitoring forecast errors $u_{\iota,t}$ obtained for each flow ι, Grundy

(2021) adapts Page's CUSUM test statistic (cf. Page 1954) to the centred squared forecast errors as follows

$$Q_\iota(m,k) = \sum_{t=m+1}^{m+k} (u_{\iota,t} - \hat{b})^2 - \frac{k}{m}\sum_{t=1}^{m}(u_{\iota,t} - \hat{b}), \tag{5.3}$$

where m corresponds to the length of Phase I, k is the current time point in Phase II and $\hat{b}$ is the mean estimate of the forecast errors computed from Phase I. From that, according to Grundy (2021), the control chart statistic is calculated as

$$D_\iota(m,k) = \max_{0\leq a\leq k} |Q(m,k) - Q(m,a)|, \tag{5.4}$$

and the corresponding upper control limit is given by

$$UCL = \hat{\sigma}_\iota \zeta_\delta g(m,k,\nu), \tag{5.5}$$

with ζ_α being the critical value of the distribution with the significance level of $\delta = 0.05$ in this chapter and $\hat{\sigma}_\iota$ the estimate of the standard deviation of the centred squared forecast errors belonging to the flow ι. The part $g(m,k,\nu)$ defines the weighting function with the tuning parameter ν as

$$g(m,k,\nu) = \sqrt{m}\left(1 + \frac{k}{m}\right)\left(\frac{k}{m+k}\right)^{\nu}, \tag{5.6}$$

which is described in detail in Horváth et al. (2004) and Grundy (2021). The tuning parameter ν enables practitioners to adjust the stopping time based on the anticipated change. For instance, if the change is expected early in the process, increasing ν will accelerate detection. In this chapter, one considers $\nu = 0$. The current time point $k = \tau$, *i.e.* the control chart detects a change point if $D_\iota(m,k) \leq UCL$. It is worth noting that no lower control limit is defined as the control statistic, obtaining non-negative values only, has its natural lower control limit which is zero.

5.3.2.1 Monitoring Scope and Performance Function

The considered CUSUM chart belongs to the univariate control charts, meaning that only one process variable is monitored by the scheme. In this case, it means that one simultaneously implements $n = |V'|$ control charts to monitor the TEN process completely. Consequently, it is also possible to perform monitoring locally and control the behaviour of only one essential flow from a complete TEN process.

Here, however, the interest lies in proposing a monitoring procedure suitable for tracking the complete TEN process. Thus, as a performance metric, a cumulative intensity function of a change is introduced as

$$I_{\mathbf{X}}(T) = \sum_{k=1}^{T} \frac{\sum_{\iota=1}^{n} \mathbf{1}_{[UCL,\infty)}[D_\iota(m,k)]}{n}, \tag{5.7}$$

presenting a cumulative sum of flows which trigger a change at a time point t by exceeding a specified threshold W. When in one flow ι a change was detected, the monitoring of this flow stops, resulting in $\max(I) = 1$. Thinking about how a signal is produced by computing $I(T)$, one needs to define a suitable threshold W based on expert knowledge or system requirements.

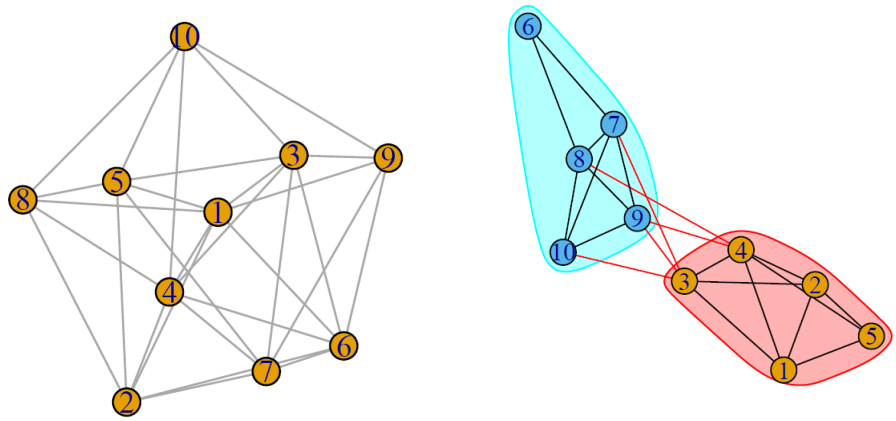


Figure 5.3: Representation of two different connectivity types: A graph structure generated by an Erdős-Rényi model (left) vs. by a SBM (right).

In the subsequent section, a simulation study that handles different cases of changes as well as compares monitoring results with respect to the selected model for constructing the adjacency matrix $\boldsymbol{Y}'$ is presented.

5.4 Simulation Study

The reason for conducting a simulation study is twofold: First, one aims to quantify the change detection speed in the effects captured by the parameters of the GNARX model. Second, there are different possibilities to construct the adjacency matrix $\boldsymbol{Y}'$ as introduced in Section 5.3.1.1. Here, two distinct random graph models are examined that cover both the availability and absence of expert knowledge about a suitable sampling technique. If a TEN process is well understood, the application of a Stochastic Block Model (SBM), where the flows should be first subdivided into clusters, is a good choice. Alternatively, it is possible to run a clustering algorithm. If the cluster structure is not suitable or no specific knowledge about the TEN process is available, the structure of G' can be sampled by applying the Erdős-Rényi model. An example of both structure types is presented in Figure 5.3.

In the next part, the design of the conducted simulation study is explained along with the presentation of the results as well as their comparison with regard to the generated graph structure.

5.4.1 Experimental Setting

To perform a simulation study, one has to decide on the design of a TEN process, the initial parameters and the test cases. To keep the study simple for reproducing it but comprehensive for designing different anomaly types as well as testing two distinct $\boldsymbol{Y}'$ structures, a TEN process with 10 flows and a medium connectivity strength is created. That means, for sampling $\boldsymbol{Y}'$ from the Erdős-Rényi model, the number of nodes is set to be 10 and the number of connections to be 30, considering that everything else is random. For sampling from an SBM, one separates the flows

Procedure 1 Simulation steps for one iteration for the test case $\beta + 0.3$

1. **Setting:** Choose the lengths of the burn-in period (300), model fitting period (600), Phase I (200) and Phase II (100) subdivided into in-control (50) and out-of-control (50) parts. Set the data-generating parameters of the GNARX model: $\alpha_{\iota,l} = \alpha = 0.2$, $\beta_{l,r} = \beta = 0.3$ with $p = 1$ and $\boldsymbol{s} = (1)$, $\gamma_{h,q} = \gamma_1 = 2$ and $\gamma_2 = 3$ with $\boldsymbol{q} = (0,0)$. Select the change strength: $\beta_\tau = \beta + 0.3$.
2. **TEN Generation:**
   a) Generate the adjacency matrix $\boldsymbol{Y}' := (Y'_{\iota\upsilon})_{\iota,\upsilon=1,\ldots,|V'|}$ either by sampling from the Erdős-Rényi model with $|V'| = 10$ and $|E'| = 30$ or the SBM with $P_{c_1c_2} = P_{c_2c_1} = 0.2$ and $P_{c_1c_1} = P_{c_2c_2} = 0.8$.
   b) For each $t = 1,\ldots,1200$ generate the GNARX process:
      i. Sample error values $\varepsilon_{\iota,t}$, where $\varepsilon_{\iota,t} \sim \mathcal{N}(0,1)$.
      ii. Sample covariate values $z_{1,\iota,t} \sim \mathcal{N}(0,1)$ and $z_{2,\iota,t} \sim \mathcal{N}(0,1)$.
      iii. For $\iota = 1$ to $\iota = |V'|$
         A. If $t \leq 1150$, generate the data with α,β,γ_1 and γ_2 parameters:
$$x_{\iota,t} = \alpha x_{\iota,t-1} + \beta \sum_{\upsilon \in N^{(1)}(\iota)} \omega_{\iota,\upsilon} x_{\upsilon,t-1} + \gamma_1 z_{1,\iota,t-1} + \gamma_2 z_{2,\iota,t-1} + \varepsilon_{\iota,t}.$$
         B. If $t \geq 1151$, generate the data with $\alpha,\beta_\tau,\gamma_1,\gamma_2$ parameters having an effect either on all flows or only on flows from c_1, adjusting the equation in A.
3. **TEN Modelling:**
   a) Fit the GNARX model with $p = 1$ and $\boldsymbol{s} = (1)$ using the first 600 generated observations after discarding the burn-in period $t = 1,\ldots,300$.
   b) Save the estimated parameters $\tilde{\alpha},\tilde{\beta},\tilde{\gamma}_1$ and $\tilde{\gamma}_2$.
4. **TEN Monitoring:**
   a) Phase I: Estimate the target process by using $\tilde{\alpha},\tilde{\beta},\tilde{\gamma}_1$ and $\tilde{\gamma}_2$ parameters to compute forecast residuals $u_{\iota,t}$ and calibrate the CUSUM control chart described in Section 5.3.2.
   b) Phase II: Run monitoring using the calibrated CUSUM control chart and compute the cumulative intensity function provided in Equation 5.7. Save the time stamp t when a change is detected.

into two clusters c_1 and c_2 of equal size. Additionally, one needs to define probabilities P that the flows are connected within a cluster or between the clusters.

Regarding the settings for sampling from a GNARX model, a process with a global autoregressive effect of order $p = 1$ is designed. Also, the stage-1 neighbourhood is considered together with two exogenous regressors and corresponding time lags $\boldsymbol{q} = (0,0)$. To be precise, one generates separate data for estimating the GNARX model (600 observations) before designing Phase I with 200 observations and Phase II with 100 observations which is subdivided into 50 in-control and 50 out-of-control samples. The detailed description of each simulation step is presented in Procedure 1. In total, 500 iterations for each test case are conducted, whose outcomes are presented subsequently.

5.4.2 Results and Comparison of Connectivity Structures

As explained in Section 5.3.2.1, the goal is to detect changes in the whole TEN process, *i.e.* the anomalous network states. Therefore, the cumulative change intensity given in Equation 5.7 is computed. For summarising the performance in the simulation study, for each time point in Phase II the values of the cumulative change intensity function are averaged, so that a mean cumulative change intensity is obtained. It is worth noting that no threshold W is defined here, letting the focus

to be on the general detection capability of the approach as well as fluctuation in the performance. It is important to note that as soon as the control chart signals for a particular flow, the monitoring of that flow stops. It means, that no multiple change points for the same flow are possible.

For each parameter α,β and γ_1 there are three test cases of changes whose values gradually increase. Comparing the different types of sampling the adjacency matrix $\boldsymbol{Y}'$, the initially estimated parameters are exchanged with anomalous either for all flows or only for flows being part of c_1.

In the following Figures 5.4, 5.5 and 5.6 the monitoring results in Phase II (PII) are displayed. The blue area defines the in-control and the red the out-of-control part. First, in all plots, one can notice the same pattern: The curve that shows the simulation with $\boldsymbol{Y}'$ sampled from the SBM stays below another curve that corresponds to the setting with an adjacency matrix generated from the Erdős-Rényi model. That corresponds to the way the change in the effects was implemented: While in the case of the SBM structure, only one community (half of the flows) was affected by anomalous parameters, the simulation that involves the Erdős-Rényi structure experiences the change in the complete network, *i.e.* all flows were affected. Thus, the cumulative intensity change reaches a value of only 0.5 which is the normalised size of a community with 5 flows that exhibit anomalous behaviour when using the SBM. Second, one can clearly recognise that the most difficult type of anomaly for the proposed approach is the change in the neighbourhood effects captured by the parameter β. It can be seen from the wider uncertainty span and longer run length to detect the implemented change in β. Overall, the simulation study reveals that the monitoring approach is highly effective, meaning that no signals occur when no actual change has been introduced but is only efficient in detecting quickly the change point if the underlying structure of G' is known, or the aim is to detect medium or large anomalies in a process.

In case one specific flow reflects the state of the whole system or is particularly relevant for the flawless functionality of a network, one could focus solely on its monitoring. For that, the CUSUM control chart would be directly applied, determining the change point. Visually, a possible outcome could look as displayed in Figure 5.7.

In the following section, as a real-world illustration, the monitoring of cross-border physical electricity flows across Europe is discussed.

5.5 Empirical Illustration

Cross-country analysis of electricity trade, especially in the context of renewable energy, is a relevant field of study that can contribute to the evaluation of transmission infrastructure policies (cf. Abrell and Rausch, 2016). It is worth noting that the cross-border physical flow network indicates the actual flow of electricity that corresponds to the laws of physics and could differ from the scheduled commercial exchanges, which reflect the economic relations between the market parties. Thus, such network process is of a particular technical importance, allowing for the flawless electricity trade across Europe. As there are no studies related to the change point detection in European Cross-Border Physical Flows (CBPFs), the monitoring of this process is described below.

5.5.1 Data Description

The European Network of Transmission System Operators for Electricity (ENTSO-E) provides data about the cross-border physical flow and scheduled commercial electricity exchanges of more

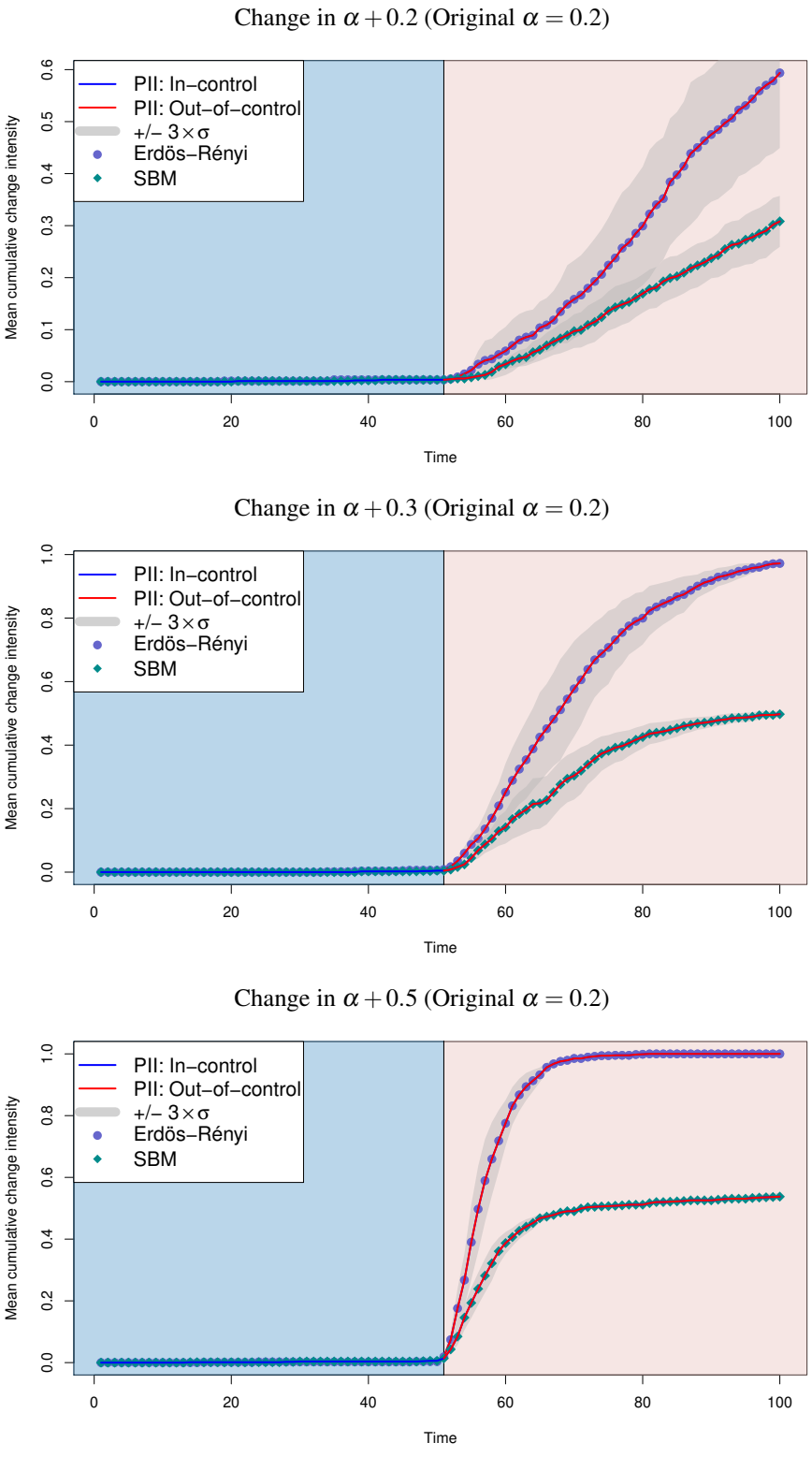


Figure 5.4: Simulation study: Visualisation of the mean of cumulative change intensity functions $I(T)$ over 500 iterations with differences in α.

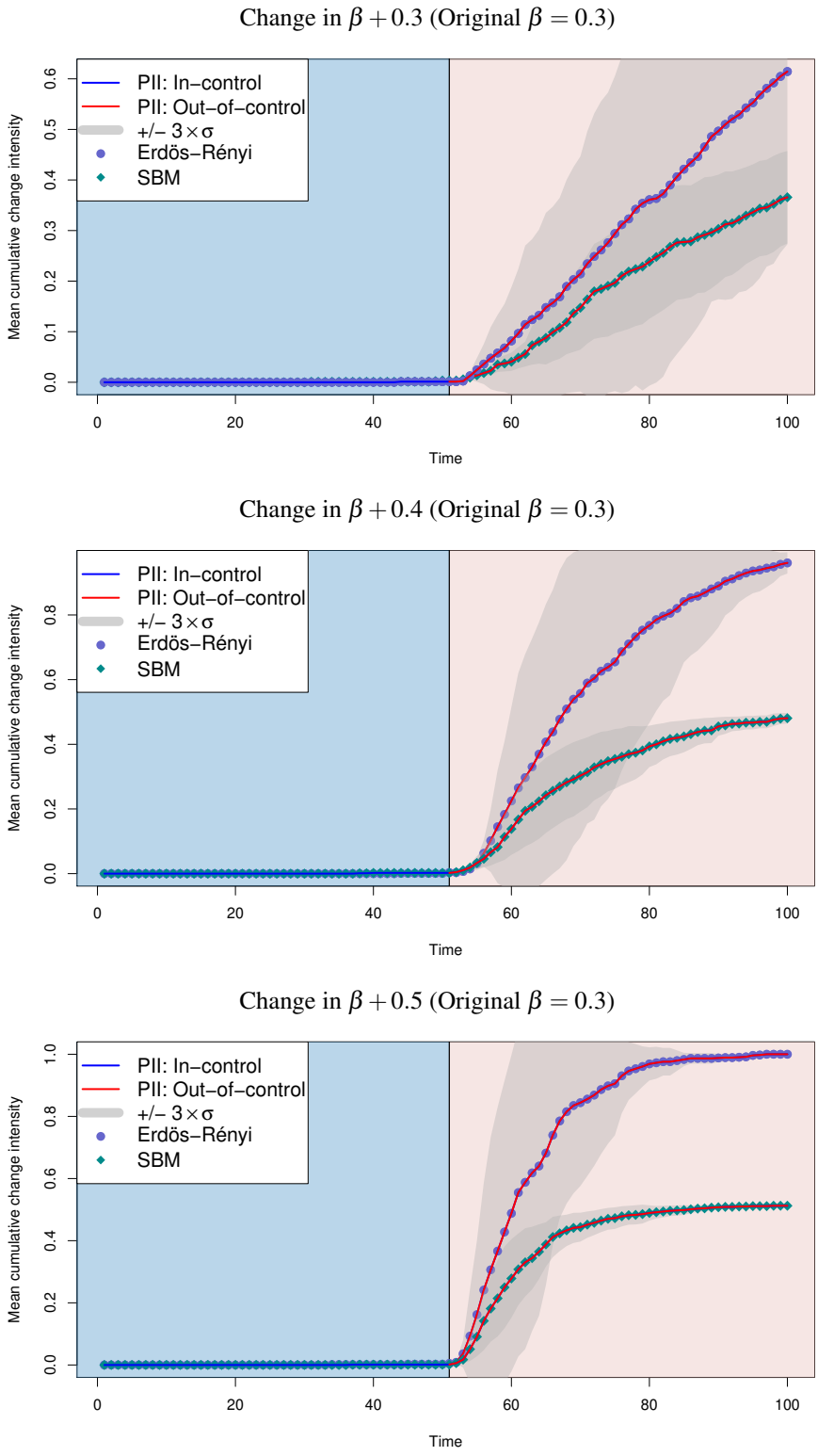


Figure 5.5: Simulation study: Visualisation of the mean of cumulative change intensity functions $I(T)$ over 500 iterations with differences in β.

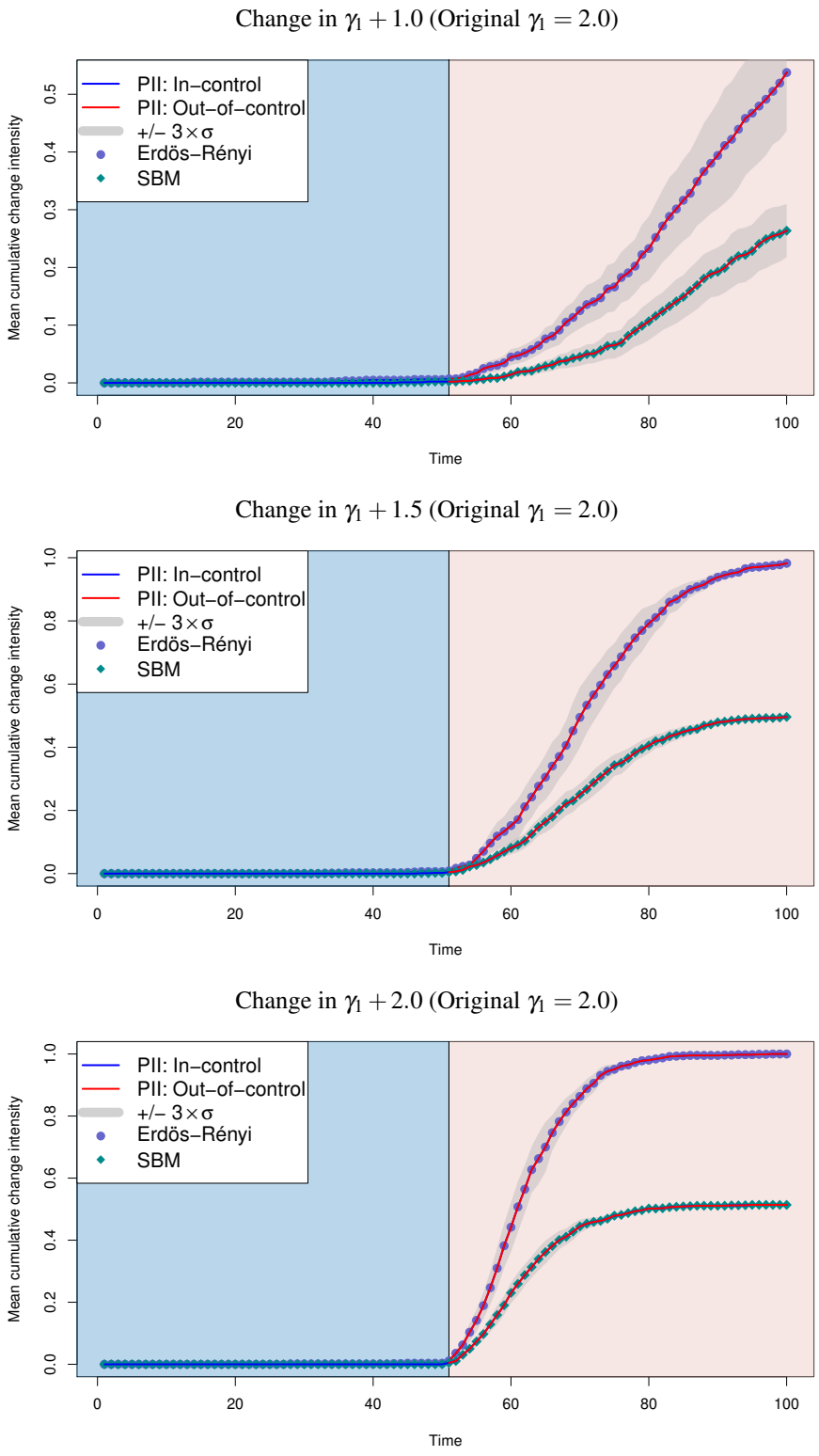


Figure 5.6: Simulation study: Visualisation of the mean of cumulative change intensity functions $I(T)$ over 500 iterations with differences in γ_1.

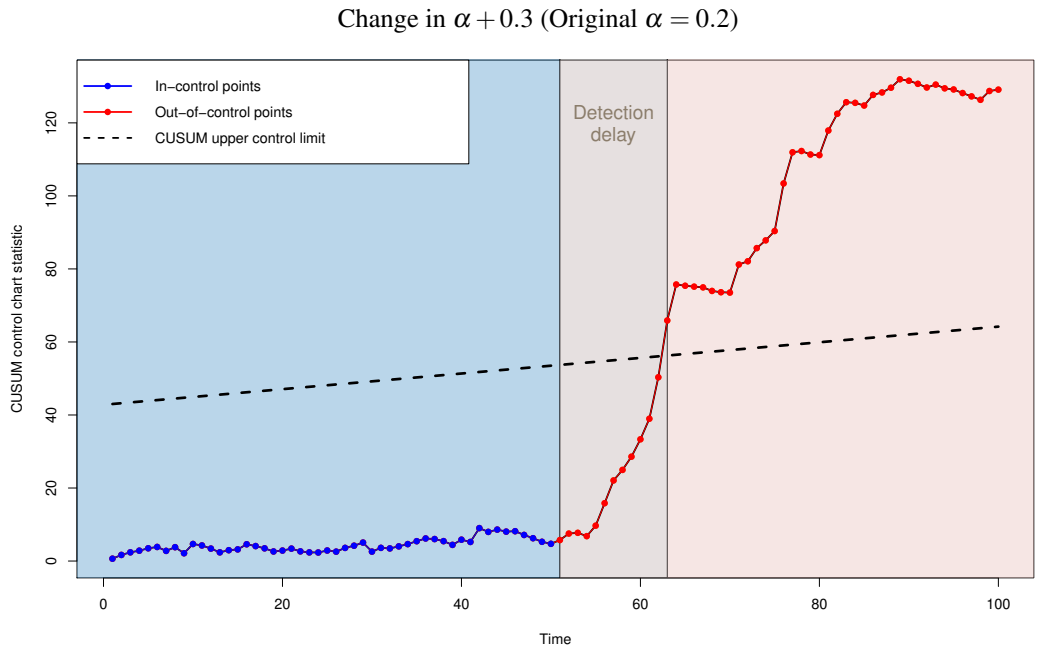


Figure 5.7: Simulation study: Example of monitoring one specific flow from a TEN process.

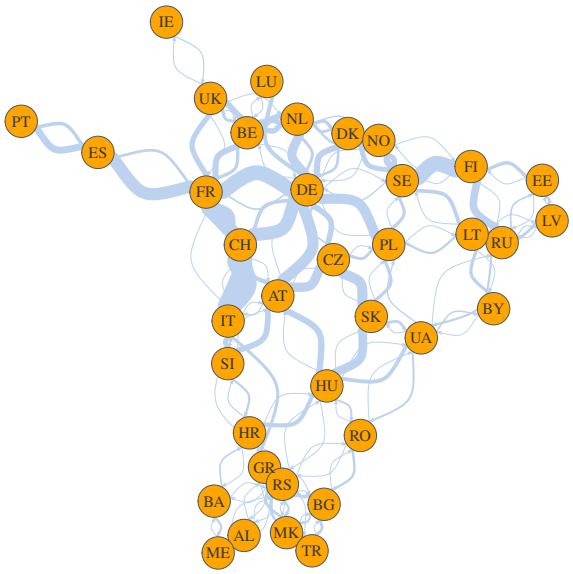
than 40 countries[2]. It offers the data for each hour for years starting from 2014. However, after careful investigation of the amount of missing data, the decision is to concentrate on the monitoring period from January 1, 2018, to November 27, 2022, and to use a weekly aggregation of the electricity values given in megawatts.

The graph in Figure 5.8 (a) represents the aggregated quantity of electricity which was transmitted in the year 2019 so that the thickness of the edges reflects the total amount of power which was passed between the two countries. The higher this value is, the thicker the edge between two nodes. However, what one can also notice is the existence of parallel edges, *i.e.* one flow f_1 coming from France (FR) to Spain (ES) and another flow f_2 back. If one decides to directly apply a new representation as introduced in Section 5.3.1.1, one would obtain a network that would be considerably bigger than the original graph. Hence, the aggregation of both flows f_1 and f_2 is required, so that only one vertex can be taken to represent a country pair. There are different possibilities for doing it, in this chapter three of them are discussed subsequently.

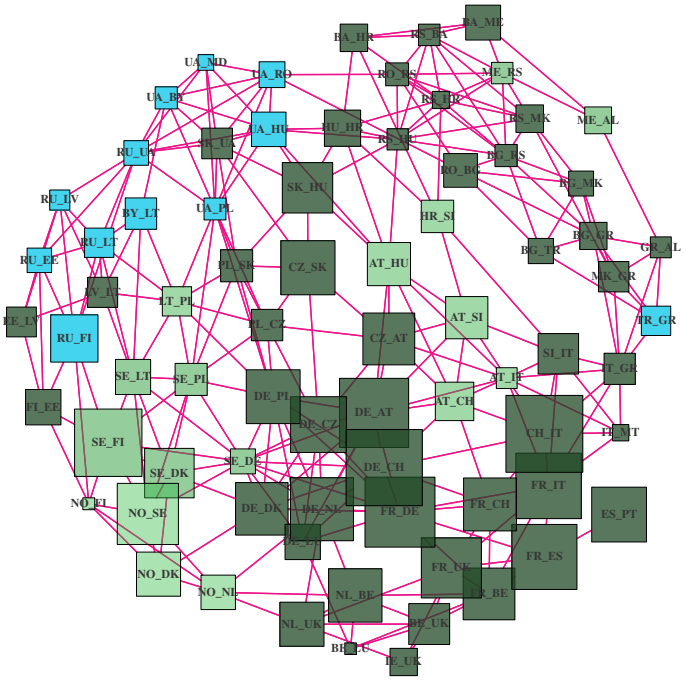
5.5.2 Phase I Modelling

To avoid a considerable expansion in the new representation of the CBPFs (see Section 5.3.1.1) but still consider the distinctive behaviour of in- and outgoing edge directions, three different aggregation strategies of both flows f_1 and f_2 are introduced, leading to three models being fitted and compared, respectively. The first statistic is the Box-Cox transformation of the sum of both flows $\mathscr{M}_1 = ln(f_1 + f_2 + 1)$ with $\lambda_1 = 0$ and $\lambda_2 = 1$ (cf. Box and Cox, 1964) that reflects the overall strength of the exchange. The second statistic measures the asymmetry in both flows as the difference of the Box-Cox transformed variables f_1 and f_2: $\mathscr{M}_2 = \ln(f_1 + 1) - \ln(f_2 + 1)$. It implies whether it is more common for one country to import or export electricity. The third statistic cap-

[2]Central collection and publication of electricity generation, transportation and consumption data and information for the pan-European market. ENTSO-E Transparency platform (2023). `https://transparency.entsoe.eu`



(a) The edge thickness implies the strength of the flow.



(b) The size of the nodes implies the strength of the electricity exchange and the colour its proportion of electricity generated with renewable energy sources (light green corresponds to the higher proportion, blue corresponds to the missing information).

Figure 5.8: Conventional representation of the cumulated cross-border physical electricity flow in the year 2019 (a) and new representation (b) following the description in Section 5.3.1.1.

	Model $\mathscr{M}_1$	Model $\mathscr{M}_2$	Model $\mathscr{M}_3$
$\tilde{\alpha}$	$\mathbf{0.896}^{***}$	$\mathbf{0.904}^{***}$	$\mathbf{0.910}^{***}$
	(0.005)	(0.005)	(0.005)
$\tilde{\beta}$	$\mathbf{0.094}^{***}$	$\mathbf{0.019}^{\cdot}$	$\mathbf{0.020}^{\cdot}$
	(0.006)	(0.011)	(0.011)
$\tilde{\gamma}_1$	$\mathbf{0.003}^{\cdot}$	$\mathbf{-0.034}^{***}$	$\mathbf{-0.003}^{***}$
	(0.002)	(0.006)	(0.001)
$\tilde{\gamma}_2$	$\mathbf{0.005}^{\cdot}$	$\mathbf{0.032}^{***}$	$\mathbf{0.003}^{***}$
	(0.002)	(0.006)	(0.001)
$\bar{\mathscr{M}}$	10.84	0.14	0.03
$\tilde{\sigma}_{\mathscr{M}}$	2.00	6.81	0.84
RMSE	0.89	2.75	0.34

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$, $^{\cdot}p \leq 0.1$

Table 5.1: Phase I modelling.

tures proportionally differences in both flows, being $\mathscr{M}_3 = \frac{f_1 - f_2}{f_1 + f_2}$. No additional transformation is required in this case as $\mathscr{M}_3 \in [-1,1] \setminus \{0\}$. Figure 5.8 (b) illustrates a new representation of CBPFs, where the new adjacency structure is constructed using the rules of a line graph described in Sections 2.1.1 and 5.3.1.1.

As no exceptional events are known in year 2018 and 2019 that could considerably disturb the CBPF network, the decision is to select these two years as Phase I. Testing different model settings, the final set-up involves a global autoregressive parameter $\tilde{\alpha}$ with $p = 1$, 1-stage neighbourhood $\tilde{\beta}$ and two covariates that correspond to the Box-Cox transformed total amount of electricity generated by renewable energy sources from both countries with the effects $\tilde{\gamma}_1$ and $\tilde{\gamma}_2$, having $\boldsymbol{q} = (0,0)$.

Table 5.1 presents the results of estimating the parameters for each choice of the aggregation statistic $\mathscr{M}$ using 7828 observations ($76 \times (52 \times 2 - p)$), where 76 defines the number of bilateral exchanges. As one can observe, the GNARX model fits the data in Phase I well and confirms the relevance of accounting for the network structure as well as external effects when modelling TEN processes. Using the displayed coefficients, one can proceed with the monitoring of Phase II which consists of the years 2020–2022.

5.5.3 Phase II Monitoring

Figures 5.9, 5.10 and 5.11 display the monitoring results during Phase II. Out of 76 country pairs, between 27 and 39 pairs have had a change point according to the implemented CUSUM control chart based on either $\mathscr{M}_1$, $\mathscr{M}_2$ or $\mathscr{M}_3$ statistics. Two periods during Phase II are highlighted: The period related to severe situations due to the COVID-19 pandemic and the period starting in February 2022 where several crises were expected or happened due to the Russian-Ukrainian war. Considering the threshold W, it is selected as $W = 0.2$, *i.e.* when in at least 16 bilateral exchanges a change point has been detected.

Comparing three different aggregation methods, similarities in detecting changes during the pandemic are noticed. However, starting from June 2021 the behaviour of the control chart significantly differs. Inspecting the results with $\mathscr{M}_1$, one notices that the time between two defined periods remains relatively stable compared to two other outcomes with $\mathscr{M}_2$ and $\mathscr{M}_3$. Then, the

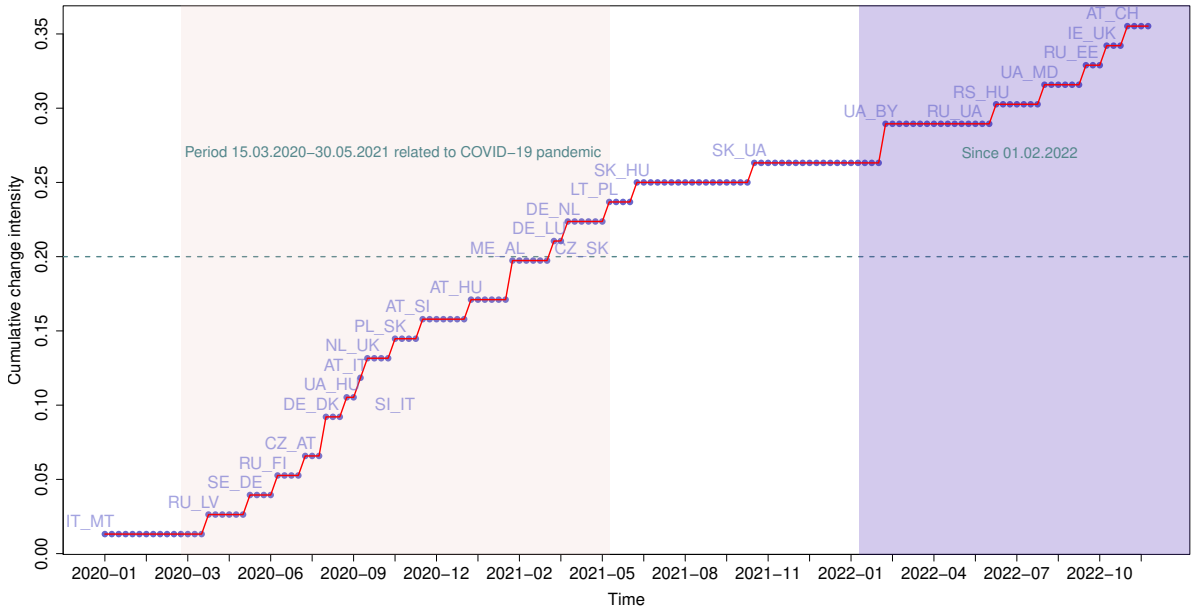


Figure 5.9: Results of monitoring $\mathscr{M}_1$, obtaining in total 27 change points.

first detection on March 6, 2022 (aggregating the days February 28 – March 6) considers the beginning of the war between Russia and Ukraine. Surprisingly, another country pair experiences a change in the same week, namely Belarus and Ukraine. Later, another change detection occurs at the end of May in the pair Russia and Estonia, reflecting the political agreement of the Baltic states to stop any energy trade with Russia. The same date and reason are also given in the control chart with $\mathscr{M}_3$ in the pair Russia and Lithuania[3]. In terms of the signal in the pair Ukraine and Moldova (control chart with $\mathscr{M}_2$), it could be related to the launch of the planned commercial exchanges announced at the end of June 2022 and further being increased at the beginning of autumn[4].

Overall, one can notice that without expert knowledge of the CBPF network, it is challenging to reflect a particular reason for the change point, however, one can also see a reliable performance of the proposed framework in detecting major events such as the pandemic and the energy crisis related to the begin of the Russian-Ukrainian war. The caveat to be aware of is the interpretation of the chosen aggregation statistic $\mathscr{M}_1$,$\mathscr{M}_2$,$\mathscr{M}_3$ as in some of the cases the signalled anomalies considerably differ.

5.6 Discussion

The network with a given structure but a random process on its edges that is defined as a Temporal Edge Network (TEN) can be of particular interest for guaranteeing the safety of the infrastructure but also for foreseeing possible accidents. Beyond the application to the bilateral electricity flows across Europe, the proposed framework could potentially be applied for monitoring traffic flows on roads or rail systems to optimise transportation infrastructure as well as within the computer

[3]https://enmin.lrv.lt/en/news/no-more-russian-oil-gas-and-electricity-imports-in-lithuania-from-sunday

[4]https://www.entsoe.eu/news/2022/09/04/transmission-system-operators-of-continental-europe-decide-to-further-increase-trade-capacity-with-the-ukraine-moldova-power-system/

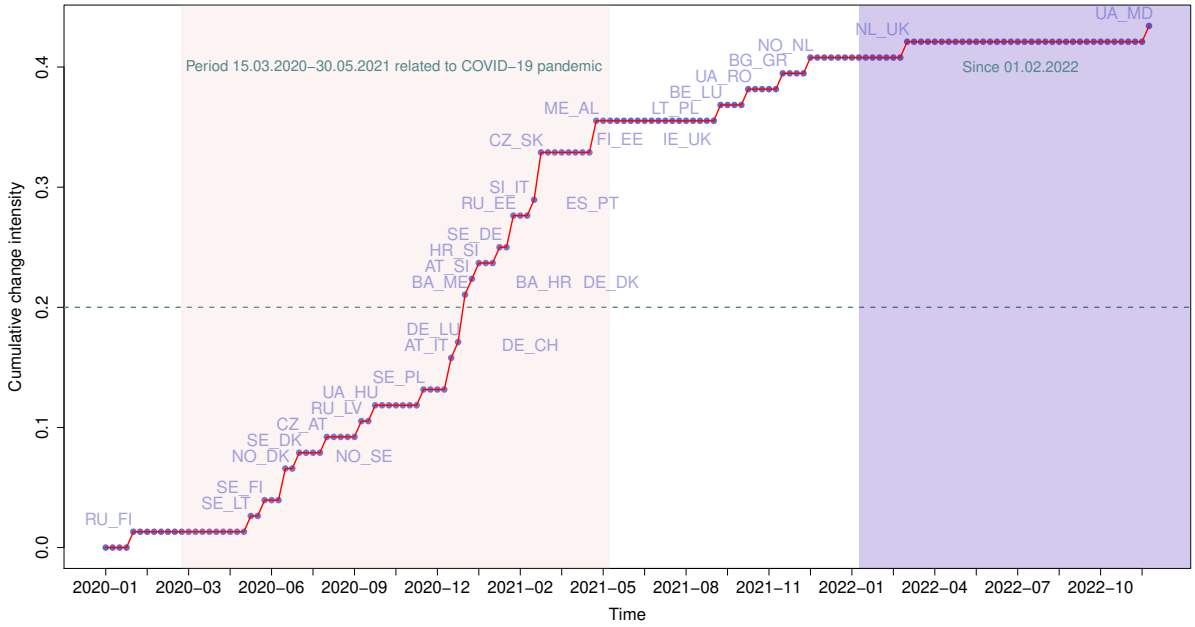


Figure 5.10: Results of monitoring $\mathscr{M}_2$, obtaining in total 33 change points.

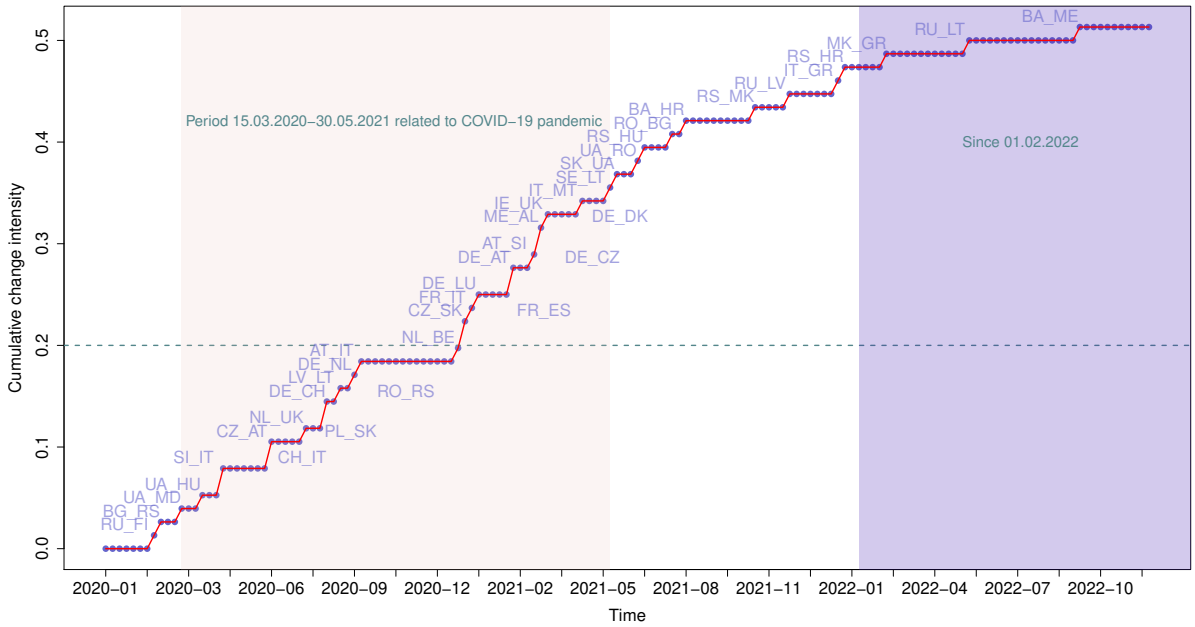


Figure 5.11: Results of monitoring $\mathscr{M}_3$, obtaining in total 39 change points.

communication domain, aiding in cybersecurity. In an environmental context, TENs may be useful in evaluating water transfers between areas or companies, contributing to sustainable water resource management.

There might still exist the question of why monitoring networks with a fixed structure needs special treatment. For example, why not try to "randomise" the connections? The explanation is straightforward: It could be possible to select a flow threshold for creating dynamics in the graph by deleting or adding the links, however, the anomalies which would be detected in this case are similar to those which are the focus of the random network monitoring. Here, the interest lies rather in anomalies occurring in the edge process, *e.g.* changes in the flow's strength or some other temporal deviations. Hence, the representation of TEN processes where edges become the main focus of both modelling and monitoring parts should be considered in this case.

6 Monitoring of Artificial Neural Networks

In this chapter[1], networks are considered to be not a structure for representing a dataset but a model based on artificial intelligence and inspired by the nervous system. In particular, a monitoring procedure for Artificial Neural Networks (ANNs) that are trained in a supervised manner for performing classification tasks is designed.

6.1 Main Intent

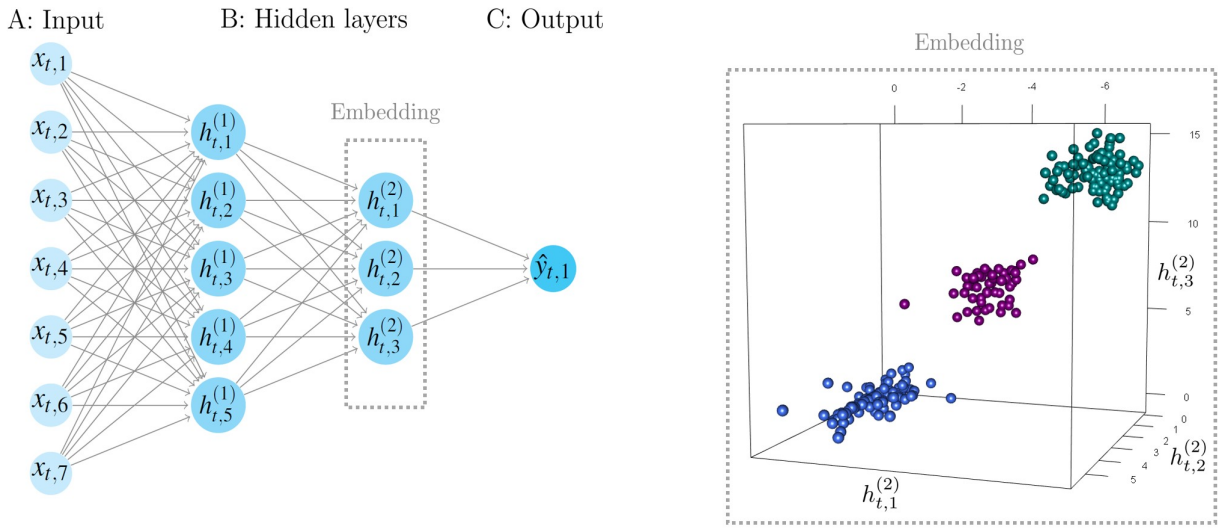
In an ANN, data flows through hidden layers, each containing neurons that perform nonlinear transformations, ultimately producing predictions in the output layer (Hermans and Schrauwen, 2013). Neurons within the ANN, as demonstrated by Wang et al. (2019), possess two crucial properties: stability and distinctiveness. For smooth classification surfaces or prediction functions, nearby samples should activate similar neurons, leading to similar output values. However, in the presence of nonstationary data, such as from an unknown class, neuron values may deviate beyond the typical activation range. In this case, its interim representation generated by ANN before the output layer would be different compared to the interim representation of the data that correctly belongs to a predicted class. Commonly represented by a low-dimensional vector, the learned latent feature representation, also called *embedding*, comprises a dense summary of the incoming sample. Alternatively, sliced-inverse regression can reduce dimensionality without specifying a regression model (Li, 1991). It projects predictors onto a subspace while preserving information about the conditional distribution of the response variable(s) given the predictors. Various methods have been proposed that utilise the inverse relationship between the variables (Cook and Ni, 2005; Wang and Xia, 2008; Wu, 2008), or sparse techniques facilitating the interpretation (Li and Nachtsheim, 2006; Li, 2007; Lin et al., 2019). Both ways of reducing the dimensionality and compressing the knowledge about a data sample are suitable for SPM.

The majority of multivariate SPM methods are based on the assumption that the observed process follows a specific distribution. In the general case of ANNs, however, the distribution of embeddings is unknown. Although the network could be trained in a way such that the embeddings follow a certain distribution, this concept is too restrictive in practice. Thus, the focus is on developing a nonparametric SPM approach using a data depth-based control chart, making no assumptions about the ANN type and the distribution of embeddings.

6.2 Definition of the Change Point

Although the whole chapter is presented from the classification scope, the proposed monitoring technique can also be applied to ANNs which solve regression tasks by grouping the predicted values into a set of classes. Thus, consider $y_t \in \{1, \ldots, v\}$ to be class labels, where the v-dimensional

[1] This chapter is based on the publication *Malinovskaya, A., Mozharovskyi, P., Otto, P. Statistical Process Monitoring of Artificial Neural Networks. Technometrics, (2023).* Copyright *American Statistical Association*. Available online: `https://www.tandfonline.com/doi/full/10.1080/00401706.2023.2239886`.



(a) The FNN with two hidden layers in the toy example

(b) Produced embeddings in the toy example

Figure 6.1: The FNN architecture displayed in (a) and embeddings from B: Hidden layers visualised in (b). Grey-framed nodes represent the last hidden layer. Blue and green points denote training data references, while magenta points represent out-of-control data embeddings. For more details, see Section 6.4.3.

output of the algorithm contains the discriminant scores for each of the given classes. In supervised learning, one assumes that true class labels are known for the training period $t = 1,\ldots,T$, which is used for estimating the parameters ϑ. Thus, $\hat{y}_t = \arg\max f(\boldsymbol{x}_t, \hat{\vartheta})$ for a set of input variables $\boldsymbol{x}_t$ is the prediction of ANNs, *i.e.* the most probable class, where $\hat{\vartheta}$ defines the learned parameters during the training, such that $\hat{y}_t$ coincides with y_t in most cases for all $t = 1,\ldots,T$.

Consider a toy example of a feedforward ANN (FNN) which defines the basic family of ANN models as presented in Figure 6.1 (a). The network consists of four layers with two hidden layers, where each circle represents a neuron or node that stores a scalar value. Consequently, a layer is a k_i-dimensional vector with k_i being the number of nodes contained in the i-th layer. Here, the input layer is a k_1-dimensional vector $\boldsymbol{x}_t \in \mathbb{R}^7$ (first layer), and its first neuron is defined as $x_{t,1}$.

For monitoring ANN applications, the usage of embeddings is proposed that are illustrated in Figure 6.1 (b). Usually, they have a vector form which is denoted by $\boldsymbol{m}_t \in \mathbb{R}^k$ with k being the dimension of the hidden layer that produces the embeddings (see Figure 6.1 (a)). This representation is observed every time ANNs are applied, *i.e.* for historical (training) and new data. Thus, embeddings implicitly depend on $\boldsymbol{x}_t$, but also on the fitted parameters ϑ, so that changes associated with ANN's data quality and performance can be detected. The proposed monitoring technique can also be applied to ANNs which solve regression tasks by grouping the predicted values into a set of classes.

In succeeding parts, one refers to the set $\{\boldsymbol{m}_t : t = 1,\ldots,T\}$ as historical data with correctly known labels $\{y_t : t = 1,\ldots,T\}$ and to the set $\{\boldsymbol{m}_i : i = T+1, T+2,\ldots\}$ as incoming data instances with the predicted class label $\hat{y}_i$ obtained from $f(\boldsymbol{m}_i, \hat{\vartheta})$. Moreover, one considers that the true label of y_i is not available and historical data are stationary. Additionally, one assumes that $\boldsymbol{m}_t$ follows a certain distribution $F_{\boldsymbol{m}_t|y_t=c}$ depending on the true class y_t. These conditional distributions $F_{\boldsymbol{m}_t|y_t=c}$ are denoted by Ξ_c for all classes $c \in \{1,\ldots,v\}$. Based on historical data, it is possible to estimate the distribution Ξ_c empirically that can be used to determine whether a possible change in the data

stream occurred, *i.e.* whether the current observation $\boldsymbol{m}_i$ is generated from a different distribution than Ξ_{y_i}. Hence, in this chapter, one defines a change point τ as

$$\boldsymbol{m}_i \quad \sim \quad \begin{cases} \Xi_{y_i} & \text{if } i < \tau \\ \Xi_{\tau} & \text{if } i \geq \tau. \end{cases}$$

In other words, a change point τ is the time stamp when the analysis of an embedding generated from the incoming data indicates that the data sample belongs to a different unknown distribution Ξ_{τ}.

Consequently, one can regard this situation as a change in the class definition and formulate a sequential hypothesis test of each sample i for detecting changes in the ANN application as follows

$$\begin{aligned} H_{0,i}: &\quad \Xi_{y_i} = \Xi_{\hat{y}_i} \qquad \text{against} \\ H_{1,i}: &\quad \text{there is a location shift and/or a scale increase in } \Xi_{y_i}. \end{aligned}$$

The alternative hypothesis could be true due to (1) misclassification by the model, *i.e.* $\hat{y}_i \neq y_i = a$, where $a \in \{1,\ldots,v\}$, leading to $\Xi_{\hat{y}_i} \neq \Xi_a$, or (2) nonstationarity of the data stream, *i.e.* $\hat{y}_i \neq y_i = b$ with $b \notin \{1,\ldots,v\}$ and $\Xi_{\hat{y}_i} \neq \Xi_b$ because $i \geq \tau$. The accurate distinction between those cases is crucial for reliable change point detection. Labelled data in Phase II helps achieving this, but a practical solution without labelled data is discussed in Section 6.5.1.3.

To test repeatedly whether the process is in control over time, *i.e.* whether the data assigned to a particular class does not deviate from the rest in this class, one can apply a multivariate control chart. Since the class distributions $\{\Xi_c : c = 1,\ldots,v\}$ are unknown and can be estimated only empirically, a nonparametric monitoring technique that relaxes any distributional assumptions is necessary. Alternatively, the nonparametric kernel density estimates can be used, namely Parzen window estimators (Parzen, 1962; Breiman et al., 1977). Moreover, kernel density estimates also have been proven beneficial for classification tasks (*e.g.* Ghosh et al., 2006).

Several multivariate nonparametric (distribution-free) charts are based on rank-based approaches. These approaches can be divided into two categories: control charts using longitudinal ranking and those employing cross-component ranking (Qiu, 2014). The first group includes componentwise longitudinal ranking (Boone and Chakraborti, 2012), spatial longitudinal ranking (Zou et al., 2012), and longitudinal ranking by data depth (Liu and Singh, 1993). While the first two subgroups are moment-dependent, control charts based on data depth offer flexibility by not imposing moment requirements. For instance, geometric and combinatorial depth functions, such as Simplicial depth and Halfspace depth, are considered and discussed in Section 6.3.1.

The depth-based control charts allow simultaneous monitoring for location shifts and scale increases in the process. The depth functions discussed in this chapter are affine invariant and satisfy important axioms such as monotonicity, convexity (except for Simplicial depth), and continuity (Mosler and Mozharovskyi, 2022). These characteristics make them suitable for the considered monitoring problem, leading to the usage of nonparametric control charts based on data depth for online monitoring.

6.3 Monitoring Framework

In terms of effective monitoring, ANNs input space is usually too complex to identify distributional changes directly from the input data (layer A, Figure 6.1 (a)), whereas output does not

always contain enough information for this purpose as demonstrated in Section 6.4 (layer C, Figure 6.1 (a)). A sufficient but not excessive amount of information can be obtained by intercepting the input propagation on an intermediate layer of the neural network (layer B, Figure 6.1 (a)), *e.g.* to estimate prediction uncertainty which requires rarely accessible supervised training data (see Corbière et al., 2019). While several layers can be considered to account for more complex dependencies, one would normally take those which are closer to the output, achieving the highest dimensionality reduction (see, *e.g.* Parekh et al., 2021). Outputs of the intermediate layers constitute an Euclidean space, where, in the unsupervised setting, anomaly-detection techniques can be applied. These can constitute a neural network themselves (*e.g.* autoencoder), belong to (statistical) ML like a Local Outlier Factor (Breunig et al., 2000), one-class Support Vector Machine (SVM) (Schölkopf et al., 2001), isolation forest (Liu et al., 2008), or be based on data depth as proposed in this chapter.

The application of data depth in quality control was originally introduced by Liu and Singh (1993), resulting in the design of Shewhart-type multivariate nonparametric control charts based on the Simplicial depth (Liu, 1990, 1995). According to recent publications on data depth-based control charts (cf. Cascos and López-Díaz, 2018; Barale and Shirke, 2019; Pandolfo et al., 2021), the careful choice of data depth notion is crucial for a satisfactory monitoring performance. Thus, several notions of data depth are compared, and their effectiveness is discussed in Section 6.4, looking at computational costs in Section 6.5.5.

6.3.1 Notion of Data Depth

A data depth is a concept for measuring the centrality of a multivariate observation $\boldsymbol{m}_i$ (cf. Zuo and Serfling, 2000; Liu et al., 2006; Mosler and Mozharovskyi, 2022) with respect to a given reference sample $R_c = \{m_t := 1, \ldots, |R|, y_t = c\}$, where $|R|$ defines its size which is assumed to be the same for all considered classes. In other words, it creates a center-outward ordering of points in the Euclidean space of any dimension. There are various notions of data depth, each of them providing a distinctive center-outward ordering of sample points in a multidimensional space. In this chapter, four data depth notions are considered: Halfspace, Mahalanobis, Projection, and Simplicial depths.

First, the Halfspace depth (originally known as *Tukey depth*) introduced by Tukey (1975) and further developed by Donoho and Gasko (1992) is defined as the smallest number of data points in any closed halfspace with boundary hyperplane through $\boldsymbol{m}_i$ (Struyf and Rousseeuw, 1999). That is,

$$D_H^c(\boldsymbol{m}_i, R_c) = \frac{1}{|R|} \min_{\|\boldsymbol{p}\|=1} |\{b : \langle \boldsymbol{p}, \boldsymbol{m}_b \rangle \geq \langle \boldsymbol{p}, \boldsymbol{m}_i \rangle\}|,$$

where $|\cdot|$ denotes the cardinality of the set B with $\boldsymbol{m}_b \in R_c$, $\boldsymbol{p}$ are all possible directions with $\|\boldsymbol{p}\| = \sqrt{\langle p, p \rangle}$ being the Euclidean norm and $\langle \cdot, \cdot \rangle$ the inner product. Here, its robust version $\mathbf{HD}_r$ is considered which is proposed by Ivanovs and Mozharovskyi (2021) and calculated approximately, offering some advantages in being strictly positive and continuous beyond the convex hull of the observed samples.

Second, the Mahalanobis depth is presented which is based on the Mahalanobis distance (cf. Mahalanobis, 1936). It is derived as

$$D_M^c(\boldsymbol{m}_i, R_c) = \frac{1}{1 + (\boldsymbol{m}_i - \boldsymbol{\mu}_{\boldsymbol{m}})' \boldsymbol{\Sigma}^{-1} (\boldsymbol{m}_i - \boldsymbol{\mu}_{\boldsymbol{m}})},$$

where $\boldsymbol{\mu}_{\boldsymbol{m}}$ is the mean vector of the embeddings in the reference sample and $\boldsymbol{\Sigma}^{-1}$ is the covariance matrix, estimated by the sample mean and the sample covariance matrix, respectively.

Third, the Projection depth proposed by Zuo and Serfling (2000) is specified as

$$D_P^c(\boldsymbol{m}_i, R_c) = \left(1 + \sup_{\|\boldsymbol{p}\|=1} \frac{|\langle \boldsymbol{p}, \boldsymbol{m}_i \rangle - \text{med}(\langle \boldsymbol{p}, R_c \rangle)|}{\text{MAD}(\langle \boldsymbol{p}, R_c \rangle)}\right)^{-1}$$

with $\langle \boldsymbol{p}, \boldsymbol{m}_i \rangle$ denoting the inner product and the projection of $\boldsymbol{m}_i$ to $\boldsymbol{p}$ if $\|\boldsymbol{p}\| = 1$. The notation $\text{med}(E)$ defines the median of a univariate random variable E and $\text{MAD}(E) = \text{med}(|E - \text{med}(E)|)$ is the median absolute deviation from the median. As the exact computation of Projection depth is possible only at very high computational costs (cf. Mosler and Mozharovskyi, 2022), the algorithms that enable its calculation approximately are used. Dyckerhoff et al. (2021) provide the implementation and comparison of various algorithms. In this chapter, one studies the performance of control charts based on three different algorithms to compute the Projection depth of both symmetric and asymmetric types. In particular, coordinate descent ($\mathbf{PD}_1$), Nelder-Mead ($\mathbf{PD}_2$), and refined random search ($\mathbf{PD}_3$) for the symmetric type; for the asymmetric type, $\mathbf{PD}_1^a$, $\mathbf{PD}_2^a$, and $\mathbf{PD}_3^a$ are considered.

Fourth, the Simplicial depth (Liu, 1990) is calculated as

$$D_S^c(\boldsymbol{m}_i, R_c) = \binom{|R|}{k+1}^{-1} \sum_{\diamond} I_{S(\boldsymbol{m}_t \,|\, t \in R_c,\, y_t = c)}(\boldsymbol{m}_i),$$

where $S(\boldsymbol{m}_t | t \in R_c, y_t = c)$ defines the open simplex consisting of vertices $\{\boldsymbol{m}_{t,1}, \ldots, \boldsymbol{m}_{t,k+1}\}$ from all observations t in the reference sample R_c. The $\diamond$ notation means that one validates all possible combinations to construct an open simplex with $(k+1)$ vertices. One specifies $I_A(x)$ as the indicator function on a set A returning 1 if $x \in A$ and 0 otherwise. Both Simplicial (**SD**) and Mahalanobis (**MD**) depths are computed with algorithms provided by Pokotylo et al. (2019).

Related to the classification problem of multivariate data, there exist depth-based classifiers (cf. Vencálek, 2017) such as the depth-vs-depth plot (*DD*-plot) designed by Li et al. (2012) or the *DD*-alpha procedure proposed by Lange et al. (2014). Also, the field of outlier or anomaly detection is a widespread area for data depth usage (cf. Dang and Serfling, 2010; Baranowski et al., 2021). Combining these two perspectives, a data depth-based Shewhart-type r control chart for single observations (see Section 6.3.2) and a batch-wise Q control chart (see Section 6.5.4.2) developed by Liu (1995) are applied for detecting nonstationarity in a data stream.

6.3.2 The r Control Chart

As stated in Section 2.2.2, the data in Phase I does not have to coincide with the full training data of the ANN but could rather be its subset. That is, the sets $R_c \subseteq \{\boldsymbol{m}_t : t = 1, \ldots, T,\ y_t = c\}$ of size $|R|$ create the Phase I data where it is essential to consider only correctly classified data samples. Successively, in Phase II, the control chart statistic is plotted for each embedding $\boldsymbol{m}_i$ with $i > T$. Figure 6.2 displays the introduced periods and sets.

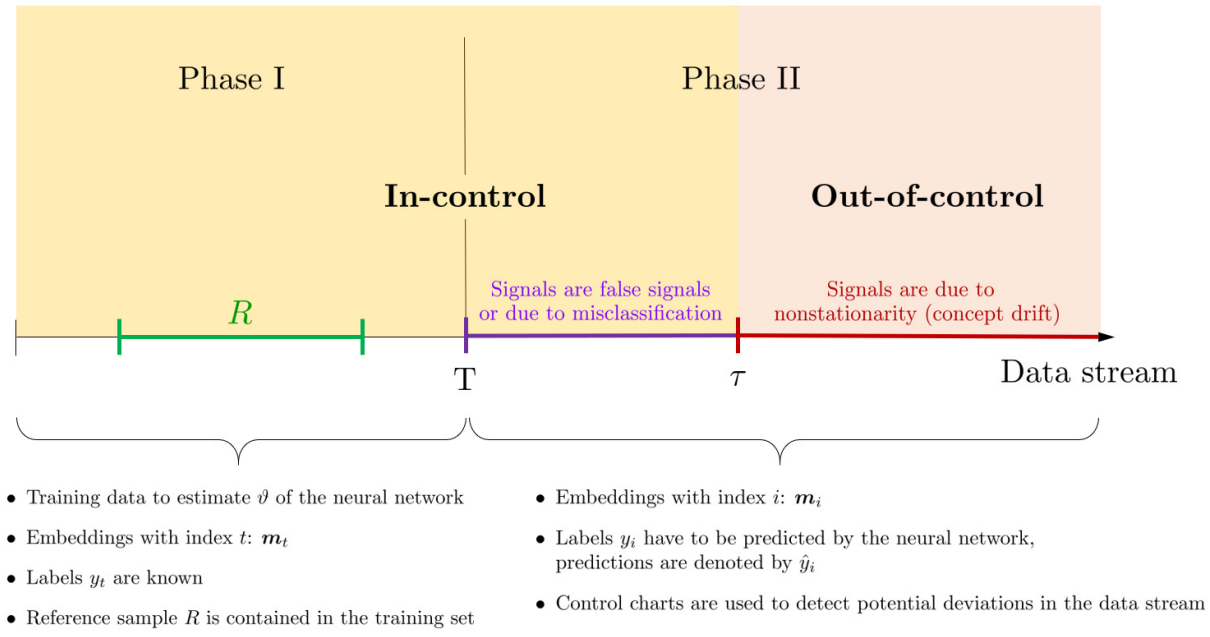


Figure 6.2: Summary of the introduced notation and data subdivision.

Considering the *r* control chart proposed by Liu (1995), the scheme is based on ranks of multivariate observations, which are obtained by computing data depth. To determine whether $\boldsymbol{m}_i$ belongs to Ξ_c, the following control chart statistic is used

$$r_{\cdot}^{c}(\boldsymbol{m}_i) = \frac{\left|\{D_{\cdot}^{c}(\boldsymbol{m}_t) \leq D_{\cdot}^{c}(\boldsymbol{m}_i) : t \in R_c, y_t = c\}\right|}{\left|\{t \in R_c : y_t = c\}\right|}$$

that defines the rank of the observed depth related to the observations in the reference sample with a class c. Thus, the *r* control chart monitors the values of $r_{\cdot}^{c}$ over time. Considering the interpretation of ranks, one can state that $r_{\cdot}^{c}(\boldsymbol{m}_i)$ reflects how outlying $\boldsymbol{m}_i$ is with respect to the reference sample. If $r_{\cdot}^{c}(\boldsymbol{m}_i)$ is high, then there is a considerable proportion of data in the reference sample that is more outlying compared to $\boldsymbol{m}_i$ (Liu, 1995).

Regarding the control limits, there is no need to introduce the *UCL* as $r_{\cdot}^{c}$ belongs to the continuous interval $[0,1]$. Considering the *LCL*, it coincides with the significance level of the hypothesis test, here defined as α. Thus, the process is considered to be out of control if $r_{\cdot}^{c}(\boldsymbol{m}_i) \leq \alpha$. The choice of α depends on the specification of Average Run Length (*ARL*) – the expected number of monitored data points required for the control chart to produce a signal (Stoumbos et al., 2001). In the case of the Shewhart-type control charts, the reciprocal of *ARL* corresponds to the False Alarm Rate (*FAR*) in the in-control state of a process. Technically speaking, since the *r* control chart is a Shewhart control chart, $FAR = \alpha$, where α is interpreted as the probability of a false alarm in Phase I (Stoumbos et al., 2001).

According to Liu (1995), the *r* control chart can be applied with affine-invariant notions of data depth, explicitly mentioning Simplicial depth, Mahalanobis depth, and Halfspace depth. Since Projection depth is also affine-invariant (cf. Mosler, 2013), both *r* and *Q* control charts can be combined with each of the data depth functions introduced in Section 6.3.1.

6.4 Simulation Study

In the following section, the effectiveness of the proposed monitoring framework is analysed by designing a simulation study in the form of a toy example. The discussion begins with a description of the considered benchmark methods, followed by the introduction of the selected performance measures in Section 6.4.2. Further, the performance of different approaches is compared in Section 6.4.3.

6.4.1 Benchmark Methods

Due to the independent development of comparable approaches (cf. Mozharovskyi, 2022; Yang et al., 2022b) and their focus on different scenarios/perspectives, there is no unified benchmark. Hence, to select a benchmark for ANN monitoring, one needs to consider the available options. Three possibilities arise: inspecting the initial input, the model's embeddings, or the final output (here softmax score) represented by layers A, B, and C in Figure 6.1 (a). Monitoring the initial data quickly becomes limited due to the complexity of typical datasets analyzed by ANNs. On the contrary, using intermediate layers reduces data dimensionality and storage requirements since only the embeddings from the training phase need to be saved. Therefore, only B and C options are considered as monitoring options. Furthermore, as a benchmark, the focus is on methods that can operate within the introduced framework. Specifically, one seeks methods capable of detecting nonstationarity in (1) individual samples (batch size of one), (2) without the need for labels in Phase II, and (3) working with or without available time stamps. To ensure comparability, the same SPM framework of the r control chart is utilised.

Based on the recent reviews (cf. Goldstein and Uchida, 2016; Villa-Pérez et al., 2021; Yang et al., 2022b), a Kernel Density Estimation Outlier Score (**KDEOS**) defined by Schubert et al. (2014), a distance-based Local Outlier Factor (**LOF**) developed by Breunig et al. (2000), and an ensemble-based outlier detection method such as isolation Forest (**iForest**) proposed by Liu et al. (2008) are chosen as a common benchmark coming from distribution- and distance-based methods. Both **LOF** and **KDEOS** compare the densities within local neighbourhoods. However, while **LOF** is based on the reachability distance of the point to its neighbourhood for density estimation, **KDEOS** uses classic kernel density estimates, *e.g.* based on Gaussian or Epanechnikov kernels. The **iForest** represents an ensemble of binary decision trees, where the points placed deeper in the trees are less likely to be outliers as they require more splits of space to isolate them. On the contrary, the samples which are allocated in shorter branches would rather be anomalous.

When considering option C in Figure 6.1 (a), the proposed approach is compared with monitoring the softmax scores. They are normalised between 0 and 1, and their length is equal to the number of neurons in the final layer. The neuron that has the maximum score corresponds to the predicted class. It is important to note that the softmax output is widely considered as a measure of the model's confidence (cf. Gawlikowski et al., 2023; Moon et al., 2020); however, there is substantial research into the area of alleviating overconfident prediction issue (Gawlikowski et al., 2023), *e.g.* by redesigning a loss function that leads to more trustful confidence estimates (Moon et al., 2020). Nevertheless, directly using the softmax output for nonstationarity detection is considered to perform reasonably well (Pearce et al., 2021). Thus, as benchmark techniques, Mahalanobis distance (**MDis**) which is well-known for detecting concept drift in similar settings (cf. Lee et al., 2018; Yang et al., 2022b) and a Natural Outlier Factor (**NOF**) based on the Natural Neighbour principle, where calculation of the factor is parameterless (Huang et al., 2016), are selected.

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	HD_r	PD_1^a	PD_2^a	PD_3^a	PD_1	PD_2	PD_3
Toy example $\boldsymbol{m}_i \in \mathbb{R}^3$	100	I	In-control	*FAR*	0.05	0.03	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	100	II	In-control	*SR*	0.08	0.00	0.10	0.08	0.04	0.06	0.06	0.08	0.10
	100	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	0.10	0.06	0.14

Table 6.1: Performance of r control charts ($\alpha = 0.05$) in the toy example with reference samples R being predicted classes. The underlined numbers indicate the suggested method for an entire monitoring period, based on the trade-off between SR and CDR.

6.4.2 Performance Measures

A well-operating control chart has a low false alarm rate (when it signals a change incorrectly) and a high rate of correctly detected out-of-control points. The performance of control charts is typically assessed by *ARL*. It measures the time until a false alarm when the process is in control and the speed at which the chart detects an actual change when the process is out of control. Alternatively, the False Alarm Rate (*FAR*) evaluates performance in Phase I, while the Signal Rate (*SR*) and Correct Detection Rate (*CDR*) assess performance in Phase II. All three metrics range between 0 and 1, providing the relative number of false or correct signals.

Regarding the *SR* value, a proportion of false alarms given the total length of the considered in-control part is calculated. In the case of the *CDR* value, it is computed as a proportion of correctly detected out-of-control data points given the total length of the designed out-of-control part. To account for a possible discrepancy between the class proportions of the predicted data in Phase II, the weighted mean of the occurred signals is computed, accounting for the number of data points in each predicted class within the observed period. In the case of *FAR*, one uses the sample mean because the reference sample sizes are identical.

If a tested control chart operates as desired, then *FAR* of Phase I equals the chosen probability of a false alarm α. In Phase II, ideally, one would expect *SR* to be similar to *FAR* for the in-control samples (neglecting the potential misclassification effect), while the *CDR* should be as large as possible for the out-of-control samples. If the *CDR* is low, *i.e.* close to 0, one would conclude that the control chart does not accomplish its primary purpose – to detect nonstationarity in a data stream.

6.4.3 Toy Example

To present the idea in a controllable environment, a toy example visualised in Figure 6.1 is created. Assuming one has a classification problem with two classes, an ANN with a one-dimensional output layer (last layer) that provides a score for the input to belong to Class 2 is constructed. When the output exceeds a threshold (often 0.5), the input is predicted as Class 2, otherwise Class 1.

As data stream, two 7-dimensional Gaussian random variables $\boldsymbol{x}_t^{(1)}$ and $\boldsymbol{x}_t^{(2)}$ with $\boldsymbol{\mu}_1 = \mathbf{0}$ and $\boldsymbol{\mu}_2 = 10 \cdot \mathbf{1}_7$, where $t = 1,\ldots,100$ and $\mathbf{1}_n$ is the n-dimensional vector of ones are simulated. Both variance-covariance matrices $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ have $\sigma_{ii} = 1$ but with $\sigma_{i-1,j} = \sigma_{i,j-1} = 0.3$ in the case of $\boldsymbol{\Sigma}_1$, and $\sigma_{i-1,j} = \sigma_{i,j-1} = -0.3$ in case of $\boldsymbol{\Sigma}_2$ for all $i,j = 1,\ldots,7$ (in respective cases $i,j > 1$), where the remaining entries are zero. Considering the out-of-control data, one samples from a new multivariate Gaussian distribution with $\boldsymbol{\mu}_\tau = 5 \cdot \mathbf{1}_7$ and $\boldsymbol{\Sigma}_1$.

A reference sample of size $|R| = 100$ for each of the classes is used, *i.e.* the entire training data because there were no misclassified data points. In Phase II, the in-control data corresponds to 50 new observations with the same distribution as used for training, while the out-of-control 50

Evaluation	Size $\lvert R\rvert$	Phase	Observed process	Metric	$\boldsymbol{m}_i \in \mathbb{R}$		$\boldsymbol{m}_i \in \mathbb{R}^{16}$			
					MDis	**NOF**	**KDEOS**	**LOF**	**iForest**	$\mathbf{PD}_2^{(a)}$
Toy example	100	I	In-control	*FAR*	0.05	<u>0.05</u>	0.05	<u>0.05</u>	0.05	<u>0.05</u>
	100	II	In-control	*SR*	0.08	<u>0.04</u>	0.00	<u>0.04</u>	0.12	<u>0.04</u>
	100	II	Out-of-control	*CDR*	1.00	<u>1.00</u>	1.00	<u>1.00</u>	1.00	<u>1.00</u>

*Table 6.2: Comparison study: Performance of r control charts ($\alpha = 0.05$) in the toy example. In the case of **MDis** and **NOF**, the data points represent the model's softmax output ($\boldsymbol{m}_i \in \mathbb{R}$). The underlined numbers indicate the suggested method based on the trade-off between SR and CDR.*

Experiment	1	2	3
Complexity	High	Medium	Low
Data type	Image	Sentence	Signal in [0,1]
Type of ANN	CNN	LSTM	FNN
Number of classes	10	4	2
Phase I, in-control (Training data)	50000	2800	170
Phase II, in-control (Test data)	10000	600	38
Phase II, out-of-control (Nonstationary data)	400	60	30
Results	Section 6.5.1	Section 6.5.2	Section 6.5.3

Table 6.3: Summary of experiments: Data size indicates the total number of samples across all classes.

data points correspond to the out-of-control distribution. The embedding layer consists of three neurons, *i.e.* $\boldsymbol{m}_i \in \mathbb{R}^3$. The visualisation of the embeddings that correspond to both reference samples and out-of-control data is displayed in Figure 6.1 (b).

Table 6.1 summarises the results from depth-based control charts. As one can observe, all versions apart from symmetric Projection and Simplicial depths can be successfully applied in this setting. The reason why symmetric Projection depths fail is the asymmetric distribution of the processed data (see Figure 6.1 (b)). Regarding the Simplicial depth, there are 24% of data points in the reference sample of Class 2 with $\mathbf{SD}(\boldsymbol{m}_t) = 0$. That can be explained by Simplicial depth assigning zero to every point in the space outside the sample's convex hull (Francisci et al., 2019). Moreover, because all out-of-control samples received the predictions of Class 2, the control chart based on **SD** could not detect the out-of-control samples. Comparing the best result from Table 6.1 which is $\mathbf{PD}_2^a$ to the benchmark in Table 6.2, one notices that the **LOF** and **NOF** achieved similar results, while the **KDEOS** and **iForest** did not hold their size of $\alpha = 0.05$ in Phase II.

6.5 Empirical Illustration

In this section, the applicability of the suggested monitoring approach is tested to real data. In total, three experiments were conducted. In decreasing complexity, the first experiment is about a ten-class classification of images, applying Convolutional ANN (CNN), followed by a four-class classification of questions, using ANN with a Long Short-Term Memory layer (LSTM) and finished with a binary classification of sonar data performed with an FNN. Table 6.3 provides a summary of the conducted experiments. The models' training aims to maximise overall classification accuracy, respectively tuning the hyperparameters and the ANN architectures. Additionally, it is worth noting that balanced datasets are used for training the ANN, ensuring an equal representation of samples from each class. In addition, the construction of reference samples (see Sections 6.5.1.1 and 6.5.1.2) as well as the misclassification effect in Section 6.5.1.3 are examined.

Figure 6.3: Image examples and class labels of the CIFAR-10 dataset.

6.5.1 Experiment 1: Multiclass Classification of Images

In this experiment, one works with the CIFAR-10 dataset[2], containing colour images of the size 32×32 pixels (Krizhevsky and Hinton, 2009), which is often applied for testing new out-of-distribution detection methods (cf. Yang et al., 2022a). In total, there are 60,000 images which correspond to 6000 pictures per class. Figure 6.3 shows examples of each category. For the out-of-control samples, the CIFAR-100 dataset[2] is considered. It contains 100 image groups, and from them four distinctive classes are selected, namely *Kangaroo*, *Butterfly*, *Train* and *Rocket*. From each category 100 images were randomly chosen, having in total 400 samples for the out-of-control part in Phase II.

To construct a classifier for predicting to which of the ten groups an input image belongs, a CNN is trained with a deep layer aggregation structure as proposed by Yu et al. (2018). A specification of such architecture is a tree-structured hierarchy of operations to aggregate the extracted features from different model stages. A detailed introduction to CNNs is given in O'Shea and Nash (2015). The embedding layer has 16 neurons, so one obtains a monitoring task of $\boldsymbol{m}_i \in \mathbb{R}^{16}$. The performance metrics such as validation loss and accuracy are used to determine the number of epochs, *i.e.* training cycles in which the model learns from the data and updates the parameters. Following that, the CNN model was trained for 88 epochs, achieving 90.43% accuracy on the test images.

As this dataset has a sufficient number of samples to perform some advanced experiments such as presented in Section 6.5.1.2 and is considered to be the most complex dataset out of three, it is used for conducting the same comparison study as for the toy example.

6.5.1.1 Choice of Reference Samples and Monitoring Results: Confidence-related Formation

Below, the effect of different reference samples is investigated, particularly concentrating on their size. To guarantee a well-chosen reference sample for each class, it is constructed by choosing $|R|$ data points that obtained the highest softmax scores. The analysis of applying randomly created reference samples is provided in the subsequent section. To illustrate the application of the r control chart in Figure 6.4, $\mathbf{PD}_2$ with $|R| = 4000$ is depicted. While there are 3 signals in Phase I (green points), where $FAR = 5\%$, a considerably larger number of signals is observed in Phase II

[2] `https://www.cs.toronto.edu/~kriz/cifar.html`

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	$\mathbf{HD}_r$	$\mathbf{PD}_1^a$	$\mathbf{PD}_2^a$	$\mathbf{PD}_3^a$	$\mathbf{PD}_1$	$\mathbf{PD}_2$	$\mathbf{PD}_3$
Experiment 1 $\boldsymbol{m}_i \in \mathbb{R}^{16}$	2000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	2000	II	In-control	*SR*	0.51	/	0.54	0.49	0.49	0.50	0.48	0.47	0.48
				SR|M	0.97	/	0.98	0.96	0.97	0.97	0.96	0.96	0.96
				SR|C	0.46	/	0.49	0.44	0.44	0.45	0.43	0.42	0.42
	2000	II	Out-of-control	*CDR*	0.92	/	0.93	0.92	0.92	0.92	0.91	0.91	0.89
	3000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	3000	II	In-control	*SR*	0.43	/	0.46	0.40	0.41	0.41	0.41	0.39	0.39
				SR|M	0.95	/	0.95	0.92	0.92	0.93	0.93	0.92	0.92
				SR|C	0.38	/	0.40	0.35	0.35	0.35	0.35	0.33	0.33
	3000	II	Out-of-control	*CDR*	0.88	/	0.90	0.86	0.87	0.86	0.87	0.86	0.84
	4000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	4000	II	In-control	*SR*	0.34	/	0.37	0.32	0.31	0.31	0.33	0.31	0.30
				SR|M	0.88	/	0.91	0.83	0.84	0.83	0.86	0.83	0.83
				SR|C	0.29	/	0.32	0.26	0.26	0.26	0.27	0.25	0.25
	4000	II	Out-of-control	*CDR*	0.79	/	0.83	0.78	0.79	0.78	0.78	0.78	0.73

*Table 6.4: Performance of r control charts ($\alpha = 0.05$) in Experiment 1 with reference samples R being predicted classes. The underlined numbers indicate the suggested method for an entire monitoring period, based on the trade-off between SR and CDR. The rows indicating SR|M and SR|C metrics are related to misclassification diagnostic and are discussed in Section 6.5.1.3. The calculation of **SD** does not happen due to the computational complexity being $O(n^{k+1})$.*

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	$\boldsymbol{m}_i \in \mathbb{R}$		$\boldsymbol{m}_i \in \mathbb{R}^{16}$			
					MDis	**NOF**	**KDEOS**	**LOF**	**iForest**	$\mathbf{PD}_2^{(a)}$
Experiment 1	2000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	0.05
	2000	II	In-control	*SR*	0.57	0.60	0.07	0.56	0.53	0.47
	2000	II	Out-of-control	*CDR*	0.92	0.92	0.05	0.93	0.93	0.91
	3000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	0.05
	3000	II	In-control	*SR*	0.44	0.47	0.07	0.48	0.46	0.39
	3000	II	Out-of-control	*CDR*	0.87	0.86	0.07	0.87	0.89	0.86
	4000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	0.05
	4000	II	In-control	*SR*	0.33	0.36	0.07	0.39	0.38	0.31
	4000	II	Out-of-control	*CDR*	0.79	0.78	0.10	0.77	0.83	0.79

*Table 6.5: Comparison study: Performance of r control charts ($\alpha = 0.05$) in Experiment 1. In the case of **MDis** and **NOF**, the data points represent the model's softmax output ($\boldsymbol{m}_i \in \mathbb{R}$). The underlined numbers indicate the suggested method based on the trade-off between SR and CDR.*

without novelty (*i.e.* in-control, purple points). A substantial part of these signals occurred on the misclassified samples, marked by the asterisks. In the out-of-control part in Phase II (red points), one notices the highest number of signals, signifying correctly detected nonstationarity.

Looking at the results shown in Table 6.4, one can recognise the following patterns: First, with increasing reference sample size, the number of false alarms in Phase II decreases. For instance, **MD** and $\mathbf{HD}_r$ lead to *SR* being over 50% when $|R| = 2000$, but the outcome has improved by over 15% when the size of the reference sample was increased. Second, the larger the reference sample, the less precise becomes the out-of-control detection. Here, the *CDR* values remain moderately high while doubling the size of the reference samples. Hence, it is beneficial to agree on such a reference sample size that slightly decreases *CDR* and, at the same time, improves the performance of the control chart during the in-control state. With this strategy, both $\mathbf{PD}_2^a$ and $\mathbf{PD}_2$ suit the entire monitoring period well.

Similar behaviour can be noticed for the benchmark methods presented in Table 6.5. Comparing the best depth-based monitoring result of $\mathbf{PD}_2^{(a)}$ with the benchmark, one recognises that it outperforms all other algorithms. Nevertheless, the *SR* values remain generally high, which could be due to misclassified observations being flagged as anomalous samples or because the dispersion

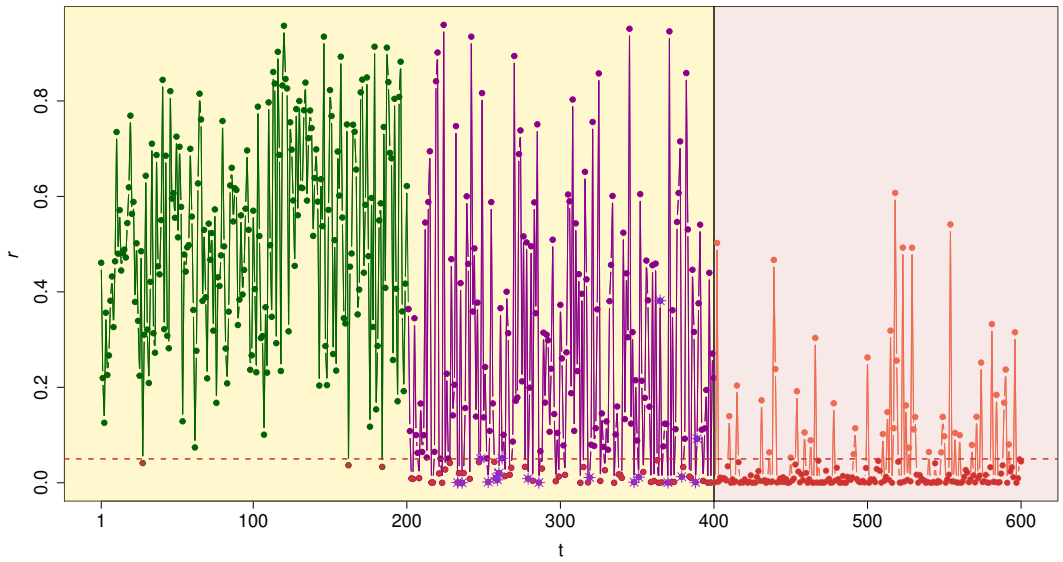


Figure 6.4: An example of the r control chart based on $\mathbf{PD}_2$ *using the data from Experiment 1. Colours correspond to the phases illustrated in Figure 6.2. The dashed line represents the control limit* $\alpha = 0.05$*, the signals are shown in dark red, and the misclassified samples in Phase II (in-control) are indicated with an asterisk.*

of the test data is large compared to the reference samples. To better understand these issues, the data from Phase II is analysed more closely in Section 6.5.1.3.

6.5.1.2 Choice of Reference Samples and Monitoring Results: Monte Carlo Study

To examine the influence of how the reference samples are formed, a Monte Carlo study is conducted. Here, reference samples for each class are created by randomly picking data points from correctly classified training data without considering the confidence-related outcomes of the ANN.

The obtained results based on 10 runs are summarised in Table 6.6. Due to the extensive computational resources involved, the study for only one type of Projection depth is conducted, namely for the symmetric case. For each choice of the data depth notion, the standard deviation does not exceed 0.01, meaning that the small number of iterations is sufficient for the investigation. The control charts based on **HD** achieve the highest *CDR* among all proposed in-control charts. Furthermore, the control charts based on $\mathbf{PD}_2$ and $\mathbf{PD}_3$ are more reliable during Phase II (In-control), resulting in *SR* of 0.18.

In contrast to the results where the reference samples are selected according to the softmax scores, one observes low fluctuation in performance with the changing size $|R|$ and lower *CDR* values. Thus, it is appropriate to choose the reference sample based on the intended purpose of the monitoring. When accepting higher signal rates in the in-control phase (potentially due to misclassification), reference samples should be chosen based on the softmax scores. In turn, this leads to more sensitive detection of out-of-control samples.

6.5.1.3 Misclassification and Data Diagnostic

To investigate the effect of misclassification, one refers to the wrongly classified images from the test data (Phase II, in-control), which constitute 958 data points. By calculating the signal rates conditional on misclassified $SR|M$ and correctly classified samples $SR|C$ for each depth notion and

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	HD_r	PD_1	PD_2	PD_3
Experiment 1 $\boldsymbol{m}_i \in \mathbb{R}^{16}$	2000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05
	2000	II	In-control	*SR*	0.21	/	0.23	0.19	0.18	0.18
	2000	II	Out-of-control	*CDR*	0.62	/	0.64	0.58	0.59	0.59
	3000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05
	3000	II	In-control	*SR*	0.21	/	0.23	0.20	0.18	0.18
	3000	II	Out-of-control	*CDR*	0.62	/	0.64	0.59	0.59	0.59
	4000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05
	4000	II	In-control	*SR*	0.21	/	0.23	0.20	0.18	0.18
	4000	II	Out-of-control	*CDR*	0.62	/	0.65	0.60	0.60	0.59

Table 6.6: Monte Carlo study: Performance of r control charts ($\alpha = 0.05$) in Experiment 1 with reference samples R being predicted classes that were randomly constructed. The underlined numbers indicate the suggested method based on the trade-off between SR and CDR. The calculation of ***SD*** *does not happen due to the computational complexity being $O(n^{k+1})$.*

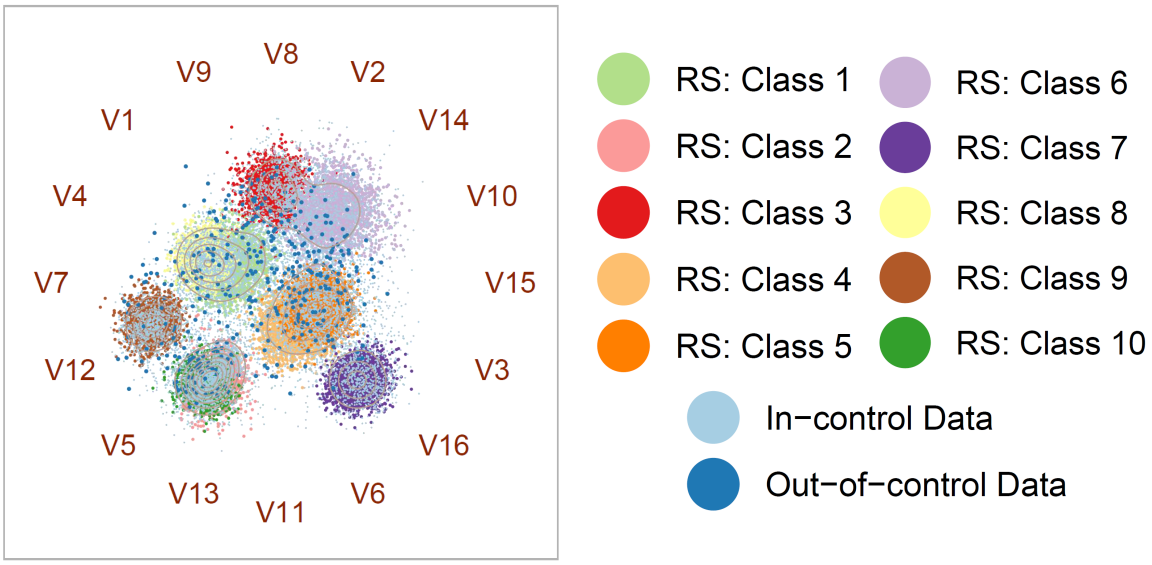


Figure 6.5: Visualisation of the data from Experiment 1 with $|R| = 2000$. $V_1, \ldots, V_{16}$ define anchors which correspond to neurons that produced embeddings $\boldsymbol{m}_i \in \mathbb{R}^{16}$. Density contour plots outline respective classes.

reference sample size in Table 6.4, one finds that the average $SR|M$ is 91.58%. At the same time, $SR|C$ values are considerably lower and approach 25% for bigger reference samples, compared to the original *SR* results.

To investigate another reason for high *SR* values, the in-control data in Figure 6.5 is visualised. One uses Radial Coordinate Visualisation (Radviz), where the variables are referred to as anchors, being evenly distributed around a unit circle (cf. Hoffman et al., 1999; Caro et al., 2010; Abraham et al., 2017). Their order is optimised to place highly correlated variables next to each other. Correspondingly, data points are projected to positions close to the variables that have a higher influence on them. As one can see in Figure 6.5, there is a comparably large section opposite the anchor V_{15} where the test data does not overlap with any of the reference samples. At the same time, a fraction of the out-of-control samples is located within reference samples, leading to more challenging detection. Hence, this analysis and the evaluation of the misclassification effect could facilitate the understanding of $SR > FAR$ and provide insight into *CDR*.

A possible solution for mitigating the misclassification effect and challenges in choosing a representative reference sample for each class (especially with a sparse availability of labels) could

Class	Example
Numeric values	What is the size of Argentina?
Description and abstract concepts	What is artificial intelligence?
Entities	What is the tallest piece on a chessboard?
Human beings	Who invented basketball?

Table 6.7: Question examples and respective four categories of the TREC dataset used for training the ANN in Experiment 2.

be the creation of a merged reference sample. In other words, one could neglect the condition of having class-specified reference samples and perform the computation of depths with respect to a grouped reference sample only. However, according to the results presented in Section 6.5.4.1, such construction of reference samples could eliminate misclassification and sample selection issues but work only for less complex cases. Moreover, the computation time would increase rapidly for high-dimensional problems (see Section 6.5.5), meaning that the proposed framework of using individual reference samples retrieved from each class would also be more suitable from this perspective.

6.5.2 Experiment 2: Multiclass Classification of Questions

For the second experiment, the Text Retrieval Conference (TREC) dataset is used which consists of fact-based questions divided into six broad semantic categories[3] (cf. Voorhees and Harman, 2000). The model was trained with the four classes: *Numeric values*, *Description and abstract concepts*, *Entities* and *Human beings*. Examples of such questions can be found in Table 6.7. The classification task is to assign an incoming question to one of four categories.

The trained neural network contains three hidden layers: a word embedding layer, a Long Short-Term Memory (LSTM) layer, and a fully connected layer which is used as the embedding generator of the size 1×8. After that, the output layer returns softmax vector 1×4, where the maximum value corresponds to the label of the predicted category. It is worth noting that here the word embedding layer is not a part of the proposed monitoring approach but a Natural Language Processing (NLP) technique that enables the model to associate a numerical vector to every word so that the distance between any two vectors is related to the semantic meaning of the encrypted words (cf. Yin and Shen, 2018). To obtain a comprehensive introduction to neural networks for NLP tasks, publications such as Goldberg (2016) and Nammous and Saeed (2019) are recommended.

In total, 700 data points of each category were used as the training data. After 25 training epochs, the achieved accuracy is 81.17% on the test dataset that contains 150 unseen samples of each class. The 60 out-of-control samples were taken from two other semantic categories that were not used for training, namely *Abbreviations* and *Locations*.

6.5.2.1 Monitoring Results

Considering the results of Experiment 2 in Table 6.8, a decreasing *SR* is given when the size of reference samples increases. Nevertheless, the *SR* values remain substantially high for considered control charts. Although the further increase of reference samples might improve monitoring

[3] https://cogcomp.seas.upenn.edu/Data/QA/QC/

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	$\mathbf{HD}_r$	$\mathbf{PD}_1^a$	$\mathbf{PD}_2^a$	$\mathbf{PD}_3^a$	$\mathbf{PD}_1$	$\mathbf{PD}_2$	$\mathbf{PD}_3$
Experiment 2 $\boldsymbol{m}_i \in \mathbb{R}^8$	400	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	400	II	In-control	*SR*	0.68	/	0.69	0.64	0.62	0.64	0.65	0.65	0.66
	400	II	Out-of-control	*CDR*	0.58	/	0.67	0.48	0.50	0.50	0.52	0.53	0.55
	500	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	500	II	In-control	*SR*	0.45	/	0.60	0.45	0.43	0.44	0.44	0.45	0.46
	500	II	Out-of-control	*CDR*	0.45	/	0.58	0.43	0.37	0.37	0.40	0.38	0.45
	600	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	600	II	In-control	*SR*	0.35	/	0.37	0.34	0.34	0.33	0.32	0.32	0.33
	600	II	Out-of-control	*CDR*	0.30	/	0.42	0.32	0.30	0.30	0.30	0.30	0.30

*Table 6.8: Performance of r control charts ($\alpha = 0.05$) in Experiment 2 with R being the predicted class. The underlined numbers indicate the suggested method based on the trade-off between SR and CDR. The calculation of **SD** does not happen due to the computational complexity being $O(n^{k+1})$.*

Evaluation	Size $\|R\|$	Metric	MD	SD	$\mathbf{HD}_r$	$\mathbf{PD}_1^a$	$\mathbf{PD}_2^a$	$\mathbf{PD}_3^a$	$\mathbf{PD}_1$	$\mathbf{PD}_2$	$\mathbf{PD}_3$
Experiment 2 $\boldsymbol{m}_i \in \mathbb{R}^8$	400	*SR*|*M*	0.86	/	0.87	0.81	0.82	0.84	0.83	0.86	0.86
	400	*SR*|*C*	0.63	/	0.65	0.60	0.57	0.60	0.61	0.60	0.61
	500	*SR*|*M*	0.80	/	0.84	0.79	0.76	0.76	0.76	0.78	0.79
	500	*SR*|*C*	0.37	/	0.55	0.37	0.35	0.36	0.37	0.37	0.38
	600	*SR*|*M*	0.69	/	0.72	0.64	0.65	0.61	0.65	0.64	0.64
	600	*SR*|*C*	0.28	/	0.29	0.26	0.26	0.26	0.25	0.25	0.26

*Table 6.9: Summary of signal rates under the condition that the samples were misclassified (SR|M) or correctly classified (SR|C) in Phase II. The calculation of **SD** does not happen due to the computational complexity being $O(n^{k+1})$.*

during the in-control state, it negatively affects the detection of anomalous data. As one can observe in the case of $\mathbf{HD}_r$, the *CDR* is reduced by 25%, changing from $|R| = 400$ to $|R| = 600$.

Overall, one notices that either a paired control chart or another procedure to decide confidently when the data points are in- or out-of-control is required to improve the performance in Experiment 2. However, to understand what could be the underlying reasons for unsatisfactory results, the misclassification effect is inspected as well as the in- and out-of-control data are visually compared.

6.5.2.2 Misclassification and Data Diagnostic

There are 113 samples from the test data that were misclassified, and the softmax output serves as a model's confidence about its prediction. Looking at the density plots in Figure 6.6, an evident difference in distributions between correctly classified and misclassified samples in Phase II (in-control) can be noticed. Remarkably, the scores of out-of-control samples are rather high and resemble the scores of the in-control samples from Phase II. This behaviour can be explained by the similarity of the out-of-control data and the reference samples. More precisely, the network is trained to distinguish the phrases based on features that are identical for both the in-control and out-of-control samples.

The Radviz shown in Figure 6.7 reveals that this issue is present in this case: the out-of-control data often overlaps with the reference samples. Moreover, most of the in-control (test data) regions are not covered by the reference samples. However, to answer the question of whether the high signal rate is partially due to the misclassified samples, conditional sample rates on misclassification ($SR|M$) and correct prediction ($SR|C$) are computed, respectively.

As shown in Table 6.9, the additional information about misclassified samples could considerably improve the *SR* results (cf. the outcome for $|R| = 600$ of $SR|C$). Moreover, one notices that the majority of the misclassified samples would be flagged as anomalous, with the signal rates declin-

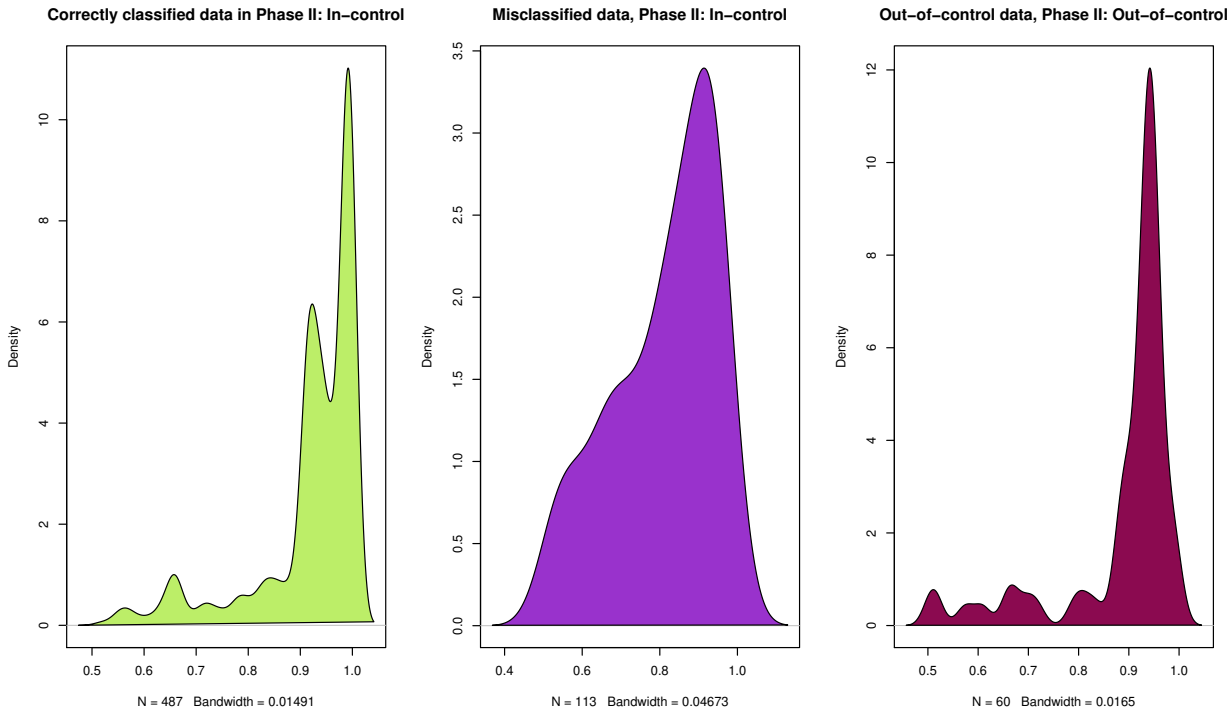


Figure 6.6: Kernel density estimates of the score distributions in Phase II for the in-control (test data) and out-of-control samples.

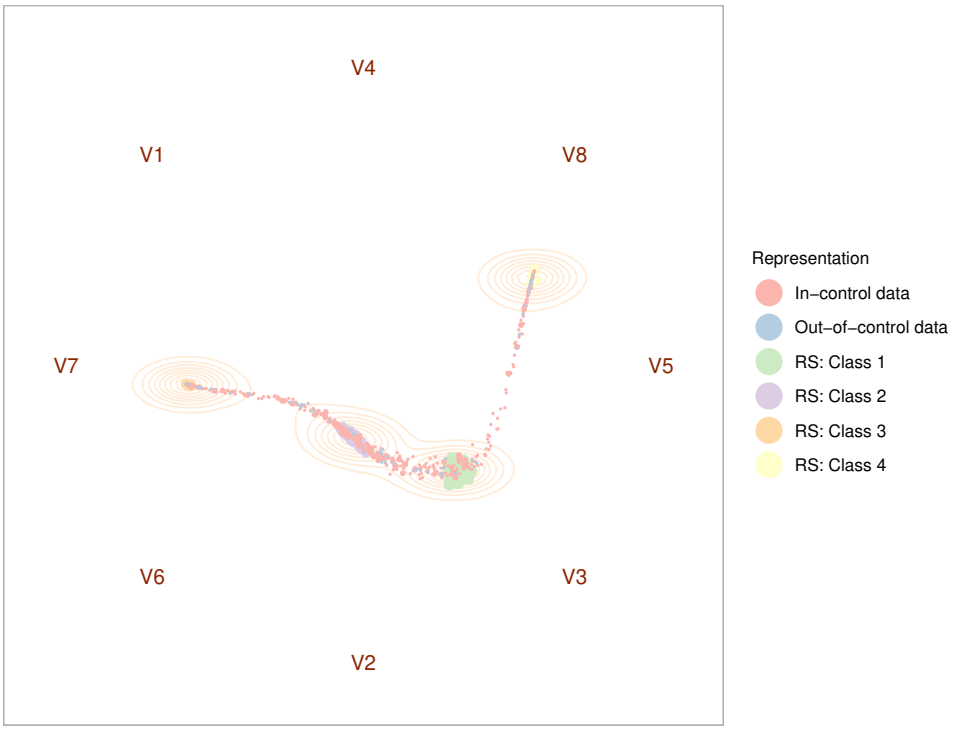


Figure 6.7: Visualisation of the Reference Samples (RS) $\{R_1,\ldots,R_4\}$ with $|R| = 400$ and the data from Phase II, in-control part (test data) from Experiment 2. $V_1,\ldots,V_8$ define anchors which correspond to neurons that produced embeddings $\boldsymbol{m}_i \in \mathbb{R}^8$. Density contour plots outline respective classes.

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	HD_r	PD_1^a	PD_2^a	PD_3^a	PD_1	PD_2	PD_3
Experiment 3 $\boldsymbol{m}_i \in \mathbb{R}^3$	50	I	In-control	*FAR*	0.04	0.00	0.04	0.04	0.04	0.04	0.04	0.04	0.04
	50	II	In-control	*SR*	0.26	0.00	0.61	0.16	0.16	0.05	0.00	0.05	0.05
	50	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	60	I	In-control	*FAR*	0.05	0.00	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	60	II	In-control	*SR*	0.16	0.00	0.50	0.26	0.24	0.16	0.00	0.00	0.00
	60	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	70	I	In-control	*FAR*	0.04	0.00	0.04	0.04	0.04	0.04	0.04	0.04	0.04
	70	II	In-control	*SR*	0.05	0.00	0.37	0.00	0.05	0.03	0.11	0.11	0.11
	70	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 6.10: Performance of r control charts ($\alpha = 0.05$) in Experiment 3 with R being the predicted class. The underlined numbers indicate the suggested method based on the trade-off between SR and CDR.

ing when the reference sample size grows. Hence, the combination of the proposed monitoring approach with an additional misclassification detection technique could lead to a more reliable nonstationarity detection.

6.5.3 Experiment 3: Binary Classification of Sonar Signals

In the third experiment, a binary classification problem of sonar data (Dua and Graff, 2019) is considered. This dataset summarises sonar signals collected from metal cylinders and cylindrically shaped rocks (Gorman and Sejnowski, 1988). There are 208 samples in total, comprising 111 metal cylinders and 97 rock returns. Each sample consists of a series of 60 numbers ranging from 0.0 to 1.0, representing a normalised spectral envelope. The task of the classifier is to distinguish which samples are from scanning a rock and which are from a metal cylinder.

The chosen model is an FNN with the architecture $60 \to 30 \to 15 \to 3 \to 1$ that comprises four fully connected layers reducing the complexity 1×60 of the input data by first processing it through the hidden layers that have 30 and 15 neurons. Afterwards, the compressed data representation enters the layer with 3 neurons whose output is also used for creating embeddings of size 1×3 for the monitoring procedure. Then, to obtain the class prediction, the interim output is transformed from size 1×3 to 1×1. As a binary classification problem is given, only one neuron in the output layer together with the sigmoid activation function that is centred around 0.5 is used, returning the probability of the processed sample belonging to Class 2. Thus, if the result of the output layer is 0.5 or higher, one concludes that the processed sample is a part of Class 2 (rock) and of Class 1 (metal cylinder) otherwise. Due to the small size of the dataset, only 38 samples (24 of Class 1 and 14 of Class 2) are allocated to the testing stage which is later used in Phase II as the in-control data. Consequently, the remaining 85 metal cylinders and 85 rock examples are taken for training the FNN. Regarding the training, the FNN model was trained for 39 epochs and achieved 81.58% accuracy on the test data.

To create out-of-control samples, after flattening the input, one estimates the parameters of a beta distribution for each class (*i.e.* $\tilde{\alpha}_1$, β_1 of Class 1, and $\tilde{\alpha}_2$, β_2 of Class 2). Afterwards, one randomly samples from a beta distribution with parameters $\tilde{\alpha}_\tau = \tilde{\alpha}_1/\tilde{\alpha}_2$ and $\beta_\tau = \beta_1/\beta_2$, obtaining 30 out-of-control observations.

6.5.3.1 Monitoring Results

Studying the outcomes of Phase I in Experiment 3 (Table 6.10), one recognises that although $\alpha = 0.05$ is chosen, *FAR* sometimes equals 0.04. This happens due to rounding, thus, $FAR = 0.04$ for $|R| = 50$ and $|R| = 70$ coincides with the expected value. Consequently, the expected

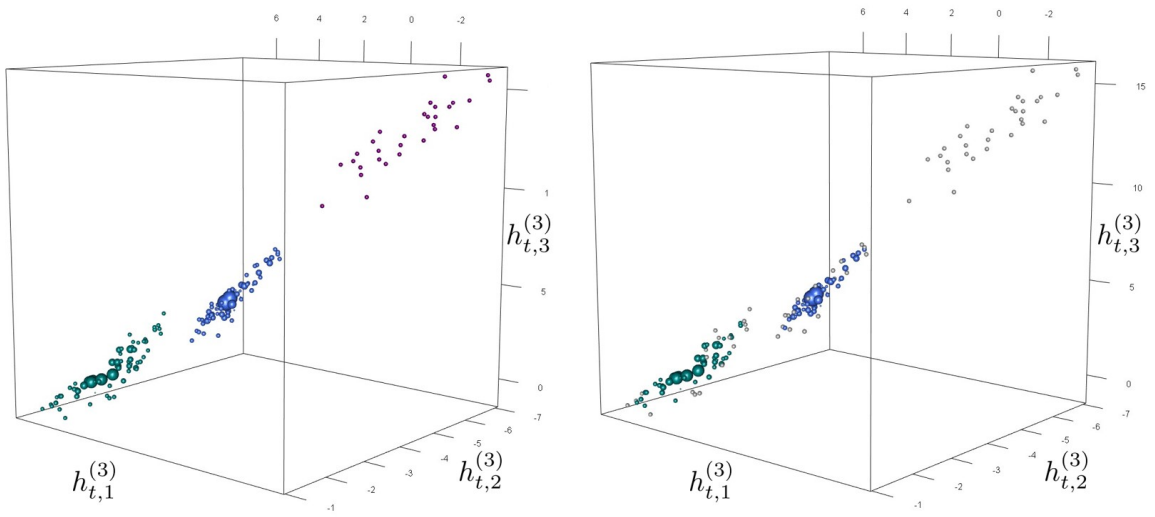


Figure 6.8: Reference samples ($|R| = 70$) and out-of-control embeddings from Experiment 3. Blue-coloured points belong to Class 1, green-coloured to Class 2, and magenta-coloured are out-of-control samples. The size of the points is proportional to the value of Simplicial depth and the grey-coloured points highlight the data samples that received $\boldsymbol{SD}(\boldsymbol{m}_t) = 0$.

value of *FAR* is reached for each notion of data depth except Simplicial depth. The reason for that is the large number of data points in the reference sample with $\mathbf{SD}(\boldsymbol{m}_t) = 0$. That can be explained by Simplicial depth assigning zero to every point in the space that lies outside the convex hull of the sample (Afshani et al., 2016; Francisci et al., 2019), similar to the results in the toy example, Section 6.4.3. In Figure 6.8, the visualisation of the dispersion of the data and how many samples obtained $\mathbf{SD}(\boldsymbol{m}_t) = 0$ is displayed. All out-of-control samples received the predictions of Class 1, meaning that their depth is determined with respect to the blue point cloud. Overall, the control charts operate well with $|R| = 50$ and the symmetric Projection depth, especially with $\mathbf{PD}_2$ and $\mathbf{PD}_3$. Alternatively, one can choose **MD** with $|R| = 70$ because of correctly reached *SR* and similarly high *CDR*.

6.5.4 Additional Examination

In this section, additional results that have not outrun the performance in already discussed settings but are still relevant for the completeness of the analysis are presented. First, one tests how the monitoring approach behaves when constructing reference samples in the form of merged classes, and whether there are significant changes in performance by applying an alternative control chart.

6.5.4.1 Reference Sample in the Form of Merged Classes

In this analytical part, the data in each of the three experiments is monitored by creating the reference samples without conditioning on the (predicted) class c. That is, the Phase II samples are compared to a joint embedding distribution from Phase I, being independent of the class labels. The benefit of this approach is that no predictions are needed, meaning that if the ANN model provides an incorrect class label, the possible negative effect of misclassification is excluded.

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	MD	SD	$\mathbf{HD}_r$	$\mathbf{PD}_1^a$	$\mathbf{PD}_2^a$	$\mathbf{PD}_3^a$	$\mathbf{PD}_1$	$\mathbf{PD}_2$	$\mathbf{PD}_3$
Experiment 1	30000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
$\boldsymbol{m}_i \in \mathbb{R}^{16}$	30000	II	In-control	*SR*	0.15	/	0.20	0.10	0.09	0.08	0.09	0.09	0.09
	30000	II	Out-of-control	*CDR*	0.19	/	0.24	0.02	0.02	0.01	0.02	0.02	0.02
Experiment 2	1600	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
$\boldsymbol{m}_i \in \mathbb{R}^{8}$	1600	II	In-control	*SR*	0.66	/	0.08	0.02	0.00	0.00	0.02	0.08	0.00
	1600	II	Out-of-control	*CDR*	0.63	/	0.05	0.00	0.00	0.00	0.02	0.02	0.02
Experiment 3	140	I	In-control	*FAR*	0.05	0.00	0.05	0.05	0.05	0.05	0.05	0.05	0.05
$\boldsymbol{m}_i \in \mathbb{R}^{3}$	140	II	In-control	*SR*	0.08	0.00	0.00	0.03	0.05	0.05	0.08	0.08	0.08
	140	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

*Table 6.11: Performance of r control charts ($\alpha = 0.05$) in the presented experiments with R being merged classes. The underlined numbers indicate the suggested method based on the trade-off between SR and CDR. The calculation of **SD** occurs for $\mathbb{R}^3$ only, as the computational complexity is $O(n^{k+1})$.*

Additionally, the application of merged reference samples implies that if a data point is flagged as out of control, it would remain out of control compared to the entire reference data.

Reporting one case for each experiment in Table 6.11, one can observe satisfactory performance in Experiment 3. In Experiments 1 and 2, the results are less convincing during the out-of-control part. The reason is that the data depth values of reference sample points in a merged case are considerably lower than in a case of individual classes, leading to a less sensitive detection of nonstationary samples. On the contrary, the *SR* values are reduced compared to the case when the reference sample relates to the predicted class only.

Although the calculation of the data depth with respect to the merged reference sample eliminates the misclassification problem, for high-dimensional problems in the conducted empirical study, the detection results of spurious data by using predicted classes on their own are substantially better. For low-dimensional cases such as Experiment 3, it is recommendable first to examine the performance of the monitoring based on the merged reference sample of different sizes, and then, if it is not operating acceptably, to apply the method with the reference samples of predicted classes.

6.5.4.2 The Q Control Chart (Batch Size > 1)

Similarly to the r control chart, the Q control chart proposed by Liu (1995) is based on ranks of multivariate observations which are obtained by computing data depth. The test statistic of the Q control chart is the average of consecutive subsets of $r_{\cdot}^{c}(\boldsymbol{m}_i)$, being

$$Q_{\cdot}^{c}(\boldsymbol{m}_i) = \frac{1}{b}\sum_{j=1}^{b} r_{\cdot}^{c}(\boldsymbol{m}_{ij})$$

with the batch size b (Liu, 1995). The interpretation of the ranks in the Q control chart is similar to the interpretation in the case of the r control chart.

To compute the control limit, the equation

$$LCL = \frac{(b!\alpha)^{1/b}}{b}$$

is used, given that $\alpha \leq \frac{1}{b!}$ (Stoumbos et al., 2001). However, if $\alpha > \frac{1}{b!}$, the LCL has to be computed numerically by solving the polynomial equation provided by Liu (1995). In general, the process is considered to be out of control if $Q_{\cdot}^{c}(\boldsymbol{m}_i) \leq LCL$.

Evaluation	Batch size	Phase	Observed process	Metric	MD	SD	$\mathbf{HD}_r$	$\mathbf{PD}_1^a$	$\mathbf{PD}_2^a$	$\mathbf{PD}_3^a$	$\mathbf{PD}_1$	$\mathbf{PD}_2$	$\mathbf{PD}_3$
Experiment 1 $\boldsymbol{m}_i \in \mathbb{R}^{16}$ $\lvert R\rvert = 3000$	3	I	In-control	*FAR*	0.08	/	0.09	0.07	0.08	0.08	0.08	0.08	0.08
	3	II	In-control	*SR*	0.58	/	0.59	0.55	0.56	0.56	0.55	0.54	0.55
	3	II	Out-of-control	*CDR*	0.98	/	0.98	0.98	0.97	0.97	0.98	0.96	0.97
	5	I	In-control	*FAR*	0.10	/	0.13	0.10	0.10	0.11	0.10	0.10	0.11
	5	II	In-control	*SR*	0.75	/	0.76	0.72	0.73	0.73	0.71	0.71	0.71
	5	II	Out-of-control	*CDR*	1.00	/	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Experiment 2 $\boldsymbol{m}_i \in \mathbb{R}^{8}$ $\lvert R\rvert = 400$	3	I	In-control	*FAR*	0.15	/	0.16	0.13	0.11	0.11	0.15	0.14	0.14
	3	II	In-control	*SR*	0.73	/	0.75	0.72	0.71	0.72	0.73	0.72	0.72
	3	II	Out-of-control	*CDR*	0.67	/	0.72	0.62	0.57	0.62	0.67	0.57	0.72
	5	I	In-control	*FAR*	0.19	/	0.19	0.17	0.15	0.16	0.20	0.18	0.18
	5	II	In-control	*SR*	0.74	/	0.77	0.74	0.74	0.74	0.74	0.75	0.74
	5	II	Out-of-control	*CDR*	0.80	/	0.90	0.80	0.70	0.80	0.80	0.80	0.80
Experiment 3 $\boldsymbol{m}_i \in \mathbb{R}^{3}$ $\lvert R\rvert = 70$	3	I	In-control	*FAR*	0.07	0.00	0.11	0.02	0.02	0.02	0.04	0.04	0.04
	3	II	In-control	*SR*	0.36	0.00	0.34	0.18	0.18	0.18	0.36	0.36	0.27
	3	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

*Table 6.12: Performance of Q control charts ($LCL = 0.22$ for $b = 3$ and $LCL = 0.29$ for $b = 5$) in the presented experiments with R being the predicted class. The underlined numbers indicate the suggested method based on the trade-off between SR and CDR. The calculation of **SD** occurs for $\mathbb{R}^3$ only, as the computational complexity is $O(n^{k+1})$.*

To evaluate the performance of Q control charts, those $|R|$ were chosen which achieved the most satisfactory (trade-off) performance with r control charts. Due to the small size of the dataset in Experiment 3, The Q control charts are computed with the batch size $b = 5$ for Experiments 1 and 2 only.

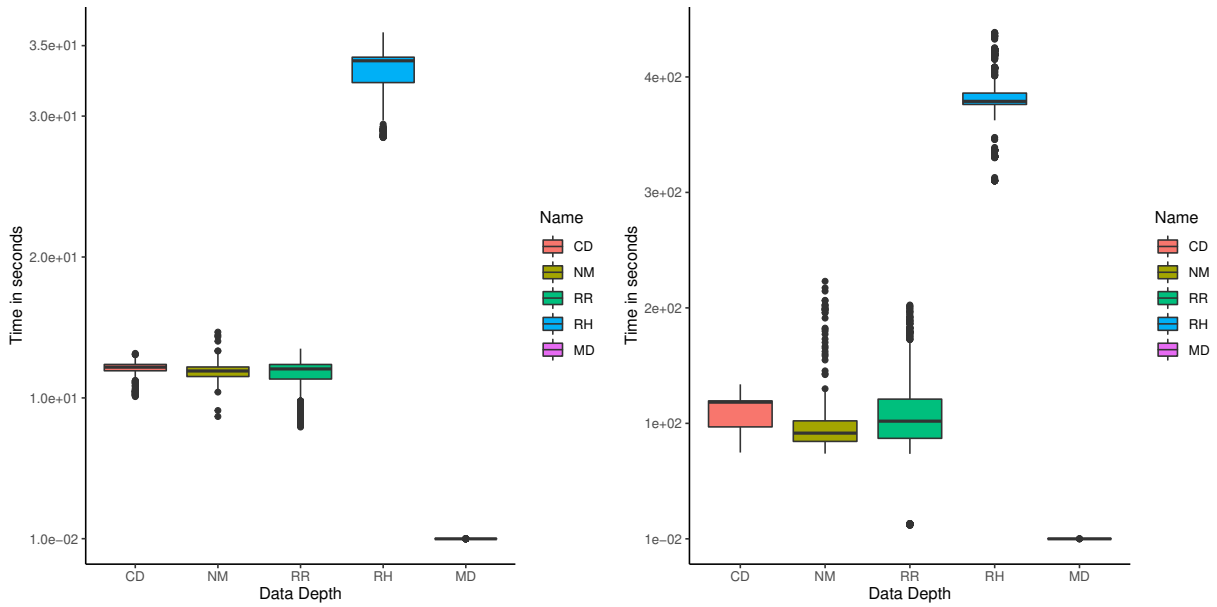
Considering the results in Table 6.12, one notes that (apart from **SD**) the *SR* values are excessively high. The increase can be explained by a substantial change in control limits. Referring to the previous description, for $b = 3$, $LCL = 0.22$ is obtained, and for $b = 5$, $LCL = 0.29$. At the same time, in all possible consolidations, the Q control chart achieves notable results during the out-of-control period, considerably improving the *CDR* values in Experiment 2 for Projection depth. Thus, if a supplementary procedure can be developed for detecting and filtering signals successfully when the process remains in control, the Q control chart would outperform the r control chart.

6.5.5 Computation Time

For performing online surveillance, the computation time of the monitoring statistic is of particular importance. As the most time-consuming part of the developed approach is the derivation of data depth values, the execution time of different algorithms to obtain the depth of one data point is studied. To provide a concise summary, one compares running times for the middle sizes of reference samples, namely $|R| = \{3000, 500, 60\}$, and for the cases displayed in Table 6.11. In the case of Projection depth, the results of the symmetric type with three applied algorithms are presented.

Regarding these algorithms to approximate Projection depth, the running times remain similar when comparing them against each other in every experiment (Figures 6.9, 6.10 and 6.11). In Figure 6.11, one can see that Simplicial depth requires considerably longer to be computed than other notions of data depth. Despite the increased complexity of experiments, Mahalanobis depth is characterised by a stable and low running time. On the contrary, the running time of $\mathbf{HD}_r$ increases noticeably with the growing size of reference samples as well as additional dimensions.

To summarise, if the performance of **MD** is excluded, the computation of data depth in $\mathbb{R}^{16}$ for one point with $|R| = 3000$ would usually take more than 10 seconds. In statistics, such results seem to be acceptable. However, taking into account the current applications of ANNs, for example, the



(a) Single class reference samples of size $|R| = 3000$ *(b)* Merged reference samples of size $|R| = 30000$

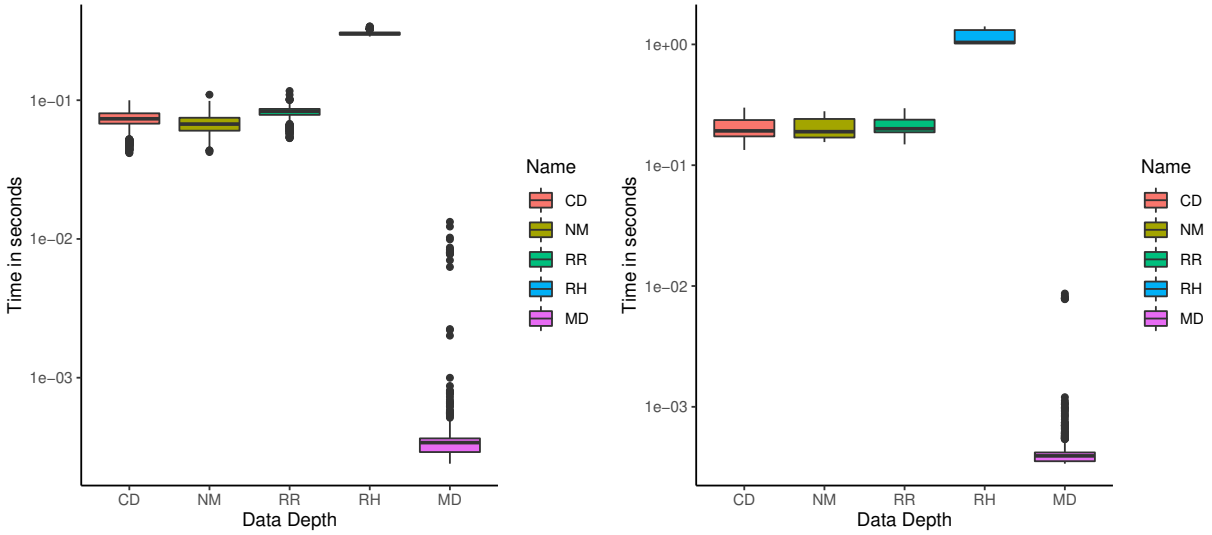
*Figure 6.9: Distribution of computation time for different data depths in Experiment 1. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace and **M**ahalanobis **D**epths.*

image classification applying a CNN, the time required to process one image is under 0.1 second (cf. Shi et al., 2022). Thus, one should critically consider the running times for the computation of data depth, striving for their improvement and guaranteeing the applicability of the proposed framework to monitor state-of-the-art models based on AI by improving the software for data depth computation.

6.6 Discussion

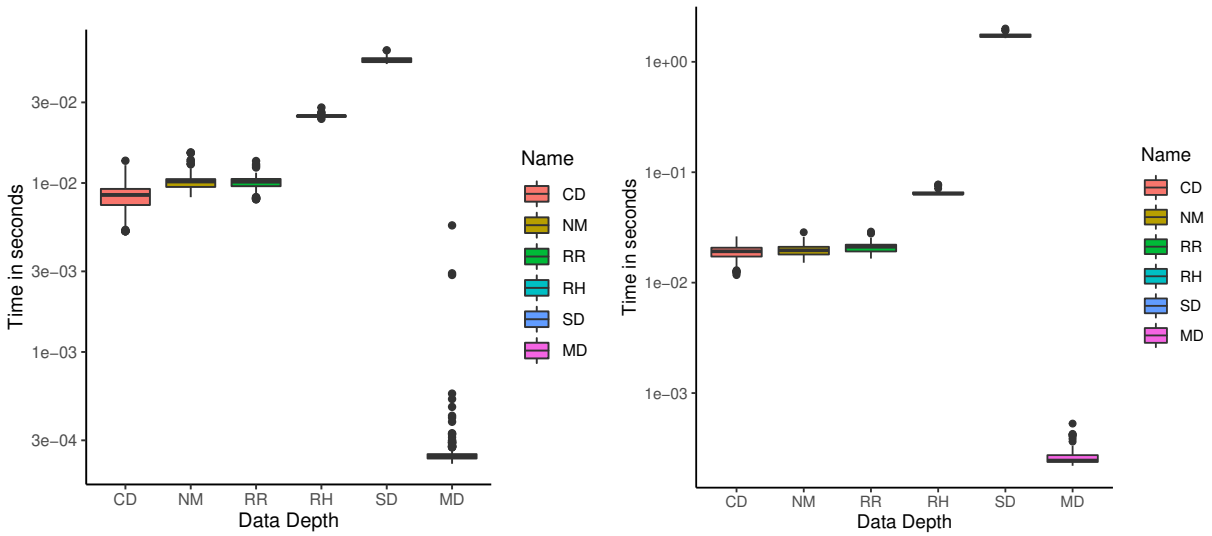
This chapter proposes a monitoring procedure designed for ANN applications that applies non-parametric multivariate control charts based on ranks and data depths. The core idea is to monitor the low-dimensional representation of input data called *embeddings* that are generated by ANNs. The proposed monitoring methodology has great potential and often outperforms benchmark methods in realistic experiments.

Overall, one notices that the asymmetric Projection depth works most reliably among the examined depths. The reason for that is twofold: First, it considers the geometry of the points, *i.e.* their asymmetric positioning. Second, as one usually aims to diagnose the outlyingness of the points that are outside of the convex hull, one needs to be able to order them. By obtaining positive depth values outside the convex hull, one can better recognise which points are anomalous. On the contrary, the Simplicial or symmetric Projection depths underperform, if many points are placed outside the convex hull (an issue for **SD**) or the data is asymmetrically spread (an issue for **PD**). Hence, under the trade-off between computation time and monitoring effectiveness, it is advisable to use the asymmetric Projection depth and compute it with the Nelder-Mead algorithm. In case the embeddings are scattered symmetrically, symmetric Projection depth can be used instead.



(a) Single class reference samples of size $|R| = 500$ *(b)* Merged reference samples of size $|R| = 1600$

*Figure 6.10: Distribution of computation time for different data depths in Experiment 2 on a logarithmic scale. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace and **M**ahalanobis **D**epths.*



(a) Single class reference samples of size $|R| = 60$ *(b)* Merged reference samples of size $|R| = 140$

*Figure 6.11: Distribution of computation time for different data depths in Experiment 3 on a logarithmic scale. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace, **S**implicial and **M**ahalanobis **D**epths.*

As soon as a change has been detected, various actions could be implemented. Lu et al. (2018) introduce the idea of *Concept Drift Understanding* before its adaptation. They stress that it is vital to answer how severe and in which data region the concept drift occurred before implementing further actions. Afterwards, the model can be either adjusted or rebuilt, resulting in a new cycle of training and validation.

7 Enhanced Network Monitoring with an Automated Inspection Procedure

In the previous three chapters, different types of network monitoring, proposing various approaches for effective identification of the time points when a significant change took place in a network (process) were intensively discussed. In this chapter[1], the focus is on the stage coming after a change has been detected, namely the inspection phase. By combining control charts and advanced machine learning algorithms, an automated and complete network monitoring procedure is offered in this chapter.

7.1 Main Intent

Many powerful techniques based on statistical inference exist to perform network monitoring. However, when a change is detected, *i.e.* the time point when the control chart has identified a considerable deviation of the process to its target state, the investigation of a possible reason happens manually. If one considers the surveillance of graph-structured data that can be particularly voluminous and challenging to process in its raw format, the task of identifying the cause of the signal and, if applicable, resolving the issue may become extremely time-consuming. Another problem arises due to the aggregation of the observations into one sample so that prior to identifying the reason for a change, one needs to determine which of the observations were truly anomalous. To improve the actions in the post-monitoring phase, an enhanced application of the control charts is proposed. In the case of a signal, it is followed by a graph machine learning algorithm that can operate on graphs to classify the cases and identify the reasons which led to the out-of-control state.

Depending on a particular application, the inspection happens either on the aggregated data, *e.g.* daily observations accumulated to a weekly sample, or on a single observation directly used in the monitoring. As the former type of data used for test statistics prevails, by starting an inspection to identify the cause of the change, one first needs to determine which single observations exhibit anomalous behaviour. Thus, before coming to the methodological part, one begins with a general discussion of anomaly detection in terms of graphs.

7.2 Anomaly Detection

It is worth mentioning that there is no unique definition of the problem *anomaly detection*. Akoglu et al. (2015) use *change point detection* as a synonym for the anomaly detection problem for dynamic graphs. On the contrary, Ranshous et al. (2015), who also provide an extensive methodological overview, introduce change point detection as a subcategory of anomaly detection problems. The reason for the considerably different points of view is that a meaningful definition can only be established after a context and particular application are specified, otherwise, the interpretation

[1] This chapter is based on the publication *Malinovskaya, A., Otto, P., Peters, T. Statistical Learning for Change Point and Anomaly Detection in Graphs. In: Steland, A., Tsui, KL. (eds) Artificial Intelligence, Big Data and Data Science in Statistics. Springer, Cham, (2022)*. Reproduced with permission from *Springer Nature Switzerland AG*. Available online: `https://link.springer.com/chapter/10.1007/978-3-031-07155-3_4`.

is ambiguous. Here, under *anomaly* one understands an abnormal activity being a sudden and significant change in the interaction patterns of a network Zhao et al. (2018). Consequently, *anomaly detection* defines the task to find those network states that significantly differ from the majority of the reference states, and, if applicable, to ascertain the type of an anomalous behaviour.

In contrast to the introduced definition of anomalous observation in the form of a whole graph, one can also define the anomaly detection problem in terms of edges or vertices. In other words, the aim is to find a subset of nodes/edges such that every element in this subset presents an uncommon evolution compared to other nodes/edges in the network. Another possible task is to identify anomalous subgraphs.

Considering recent advancements in the area of machine learning on graphs to detect anomalies, impressive results were achieved by applying the Graph Convolutional Network (GCN) framework (cf. Zheng et al., 2019; Kumagai et al., 2021; Wang et al., 2021). However, to extract necessary information and provide substantial results, the neural network needs the graph data to be constructed as a set of low-dimensional learned continuous vectors (called *embeddings*) without neglecting the relational structure and corresponding attributes. This task can be fulfilled by the graph representation learning techniques, which are briefly discussed together with the GCNs in the following section.

7.3 Graph Representation Learning

Undeniably, hand-engineered graph statistics are useful in analysing graph-structured data in terms of interpretation and computational costs. However, the manual selection of which features should be incorporated into the metrics, and further determination of statistics can be a time-consuming process. Moreover, this approach is restrictive because neither the selection of features nor metrics can be adapted through a learning process, which crucially constrains the effectiveness of machine learning-based algorithms. An alternative that encodes the network structure compactly and without losing any relevant information is Graph Representation Learning (GRL).

In contrast to conventional methods, where the selection and design of graph statistics are seen as a preprocessing step, GRL techniques regard the problem of learning embeddings as a machine learning task. To be more precise, the goal is to learn and optimise a mapping that embeds vertices, edges or entire (sub)graphs as points in a low-dimensional vector space $\mathbb{R}^d$ such that geometric relationships in this latent space reflect the structure of the initial graph (Hamilton et al., 2017). If one considers node embedding, the main purpose is to find a projection

$$f_\Theta : v_i \rightarrow \boldsymbol{z}_i \in \mathbb{R}^d,$$

where $d \ll |V|$, meaning new dimension d is much smaller than the number of vertices in the graph, $\boldsymbol{z}_i = \{z_1, z_2, \ldots, z_d\}$ represents the embedded vector that captures the graph position of node v_i and the structure of its local graph neighbourhood; f_Θ is a mapping function parametrised by Θ. Depending on the embedding method, the incorporation of edge and nodal attributes into the latent representation $\boldsymbol{z}_i$ of the node v_i is also possible. Afterwards, the learned representation can be used as input for the main machine learning task, for example, classification.

Related examples are shallow embedding approaches that define the *encoder* mapping function f as an embedding lookup. In this case, the set of trainable parameters is optimised directly, meaning that $\Theta = \boldsymbol{Z}$, with $\boldsymbol{Z} \in \mathbb{R}^{d \times |V|}$ being a matrix, where each column defines node embeddings $\boldsymbol{z}_i$ for each vertex v_i. The best-known techniques are either based on matrix factorisation (*e.g.* Laplacian

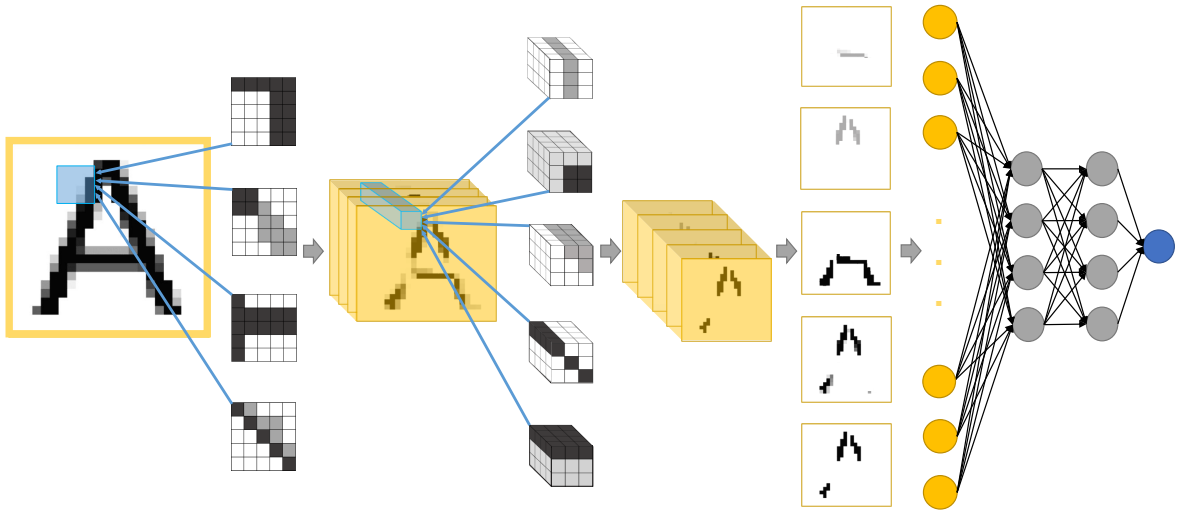


Figure 7.1: An example of a convolutional neural network solving a binary classification task. The last four layers coincide with the feedforward neural network architecture presented in Figure 2.4. The input to this part of the network is a long array, which is obtained by flattening the tensor consisting of final feature maps.

eigenmaps) or random-walk statistics (*e.g.* DeepWalk and node2vec) (Hamilton et al., 2017). However, shallow embedding approaches have some considerable limitations. The first issue is that there is a unique embedding for each node in the graph, meaning that no parameters are shared across vertices resulting in the absence of generalisation. Another problem is the ability to generate embeddings only for nodes that were present during the learning process. That means the graph structure should remain unchanged for the method to work correctly, which is highly unrealistic in many applications. To overcome these limitations, an alternative framework was proposed, namely *graph convolutional networks* that is presented in the subsequent section.

Further encoding techniques which do not only focus on node representation together with the discussion of recent challenges in the GRL can be found in Hamilton et al. (2017), Cai et al. (2018), Chen et al. (2020), Gogoglou et al. (2020) and Hamilton (2020).

7.4 Graph Convolutional Networks

To facilitate the understanding of GCNs, a detailed explanation of how a conventional Convolutional Neural Network (CNN) works, using the most popular data format they are applied on, namely images, is provided below.

7.4.1 Convolutional Neural Networks

Consider a binary classification of the letters into vowels and consonants. Figure 7.1 provides a schematic illustration of a CNN for this machine learning task. First of all, the input is an image displaying the letter A. To start extracting the necessary features for learning whether this letter is a vowel or a consonant, one applies a convolutional layer that is represented by four filters (also known as *feature detectors* or *kernels*) with different weights of size 5×5. These filters are applied sequentially to the image, sliding over all pixels and performing convolution of the

filter weights and respective neighbouring pixels. After the introduction of nonlinearity by using a suitable activation function which is applied after each layer in the neural network, the interim output consists of four feature maps combined into a single four-dimensional representation. As one can notice, the width and height of the initial image do not coincide with the size of the new output. The reason is the standard application of a pooling operation after the convolutional layer that reduces the size of the produced feature maps, accelerating the data processing and affirming some of the detected features. Next, another convolutional layer comes, this time with five filters of size $3 \times 3 \times 4$. The convolution is performed in the same way as in the previous layer, taking into consideration that the kernel size must be adjusted to the size of the previously produced interim output. As a result, one obtains five feature maps connected to a single tensor. Subsequently, the convolutional operations are followed by the usage of an activation function and pooling operation (these steps are not represented in Figure 7.1 due to the space limitation).

Now feature extraction is completed, and the classification step comes. To proceed, the final feature maps should have a one-dimensional form due to the subsequent application of fully connected layers. To obtain the desired dimension, one applies the operation called *flatten*, reshaping the obtained tensor to an array (the yellow neurons in Figure 7.1). The motivation for applying exactly this structure for determining the class label is the global consideration of the extracted features: The information flow between the fully connected neurons enables the mixing of signals. In contrast, the convolutional layers are particularly useful for data preprocessing due to their focus on the regions of adjacent pixels. Reaching the output layer of the CNN in Figure 7.1, as one has a binary classification, only one neuron (coloured blue) is provided, which would return a class label, meaning the letter *A* is either a vowel or a consonant.

Coming back to the network structure as an input, one needs to exchange the image with a graph. Consequently, instead of pixels, one considers nodes and their local neighbourhoods. However, a graph is an example of non-Euclidean data (Asif et al., 2021), therefore, the CNN cannot be directly applied to network data. Thus, the field *geometric deep learning* has emerged, whose aim is to develop deep learning models for irregular data structures (Bronstein et al., 2017). The pioneer framework is known as *Graph Neural Networks* (GNNs) which establishes the idea of including the neighbourhood information of a node into its latent representation, applying neural message passing form (Scarselli et al., 2008). The earliest GNN variations were limited in covering edge features and were also restricted in the choice of trainable parameters. Consequently, many advanced models arose, one of the examples being GCNs (Kipf and Welling, 2017).

Both GNNs and GCNs belong to a more general category which is message passing neural networks (Gilmer et al., 2017). For reading about other neural network architectures which operate on graphs, Wu et al. (2020) is suggested as a helpful reference.

7.4.2 Application Phases of Graph Convolutional Networks

The GCN framework generalises the concept of CNNs which is especially popular in image processing, as shown in the previous section. The application of GCNs is structured into two main phases: the message passing phase and the readout phase (Zhou et al., 2020). The goal of the first phase is to propagate the information across nodes in order to create a new representation of the whole graph. In the readout phase, the obtained graph representation is used to solve a particular task such as classification.

In the first phase, consider $k = 0, \ldots, K$ to be the number of message passing iterations. In fact, K equals the number of graph convolutional layers in the neural network. Next, one defines a

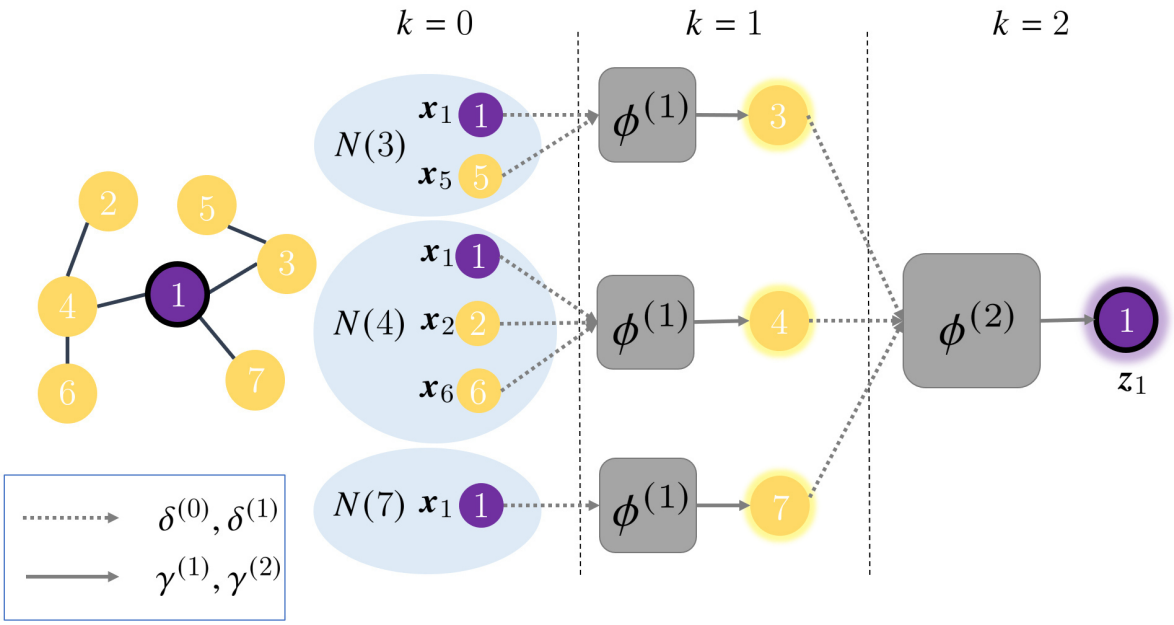


Figure 7.2: An example of the procedure to generate the embedding $\boldsymbol{z}_1$ of the vertex v_1 during the message passing phase. The attributes of node v_i are specified by $\boldsymbol{x}_i$. The sets $N(3)$, $N(4)$ and $N(7)$ define the neighbourhood of the nodes v_3, v_4 and v_7 respectively. The $\delta^{(k)}$ functions correspond to the message aggregation functions, the $\phi^{(k)}$ functions are the message creation functions and the $\gamma^{(k)}$ define the update functions.

set $N(i)$ to contain the neighbouring nodes of v_i. In contrast to an image input as presented in Figure 7.1 where each pixel has a constant number of neighbouring pixels (see the blue areas), the size of the sets $N(i)$ may vary for each node as shown in Figure 7.2. Similar to the feature map as a latent representation of an image, in the GCN one has a collection of feature vectors $\boldsymbol{H}^{(k)} \in \mathbb{R}^{d\times|V|}$, where $\boldsymbol{h}_i^{(k)}$ defines a d-dimensional hidden embedding of node v_i. At each iteration k, $\boldsymbol{h}_i^{(k)}$ incorporates the aggregated information (called *message*) from $N(i)$. As it can be seen in Figure 7.2, initially at $k=0$, $\boldsymbol{h}_i^{(0)} = \boldsymbol{x}_i$ that represents the input features of the node v_i. For example, considering node v_1 in Figure 7.2, one iterates the aggregation and updates process of the node embedding for $k=2$, so that the final learned representation is $\boldsymbol{z}_1 = \boldsymbol{h}_1^{(2)}$, which includes the information about the stage-2 neighbourhood.

To summarise, one has a step that creates a message for a vertex v_i based on the knowledge about $N(i)$

$$\boldsymbol{m}_i^k = \delta^k\big(\phi^k(\boldsymbol{h}_i^{k-1}, \boldsymbol{h}_j^{k-1}, \boldsymbol{l}_{i,j}) : j \in N(i)\big),$$

where δ^k defines a differentiable, permutation invariant function of the k-th convolutional layer, *e.g.* sum or average, and $\phi^{(k)}$ is a differentiable function that creates messages between the vertex i and the nodes in $N(i)$, incorporating the edge features $\boldsymbol{l}_{i,j}$. For instance, this could be a multi-layer perceptron or a complex filter function defined by Gaussian mixture models. In Figure 7.2, these functions are represented by grey boxes. Similar to a filter in the CNN displayed in Figure 7.1, the weights of the functions $\phi^{(k)}$ stay unchanged for the whole input in the layer k. The message passing is followed by the update step

$$\boldsymbol{h}_i^k = \gamma^k(\boldsymbol{h}_i^{k-1}, \boldsymbol{m}_i^k),$$

where γ specifies another differentiable function – the activation function such as Rectified Linear Unit (ReLU).

In the readout phase, one aggregates node features from the final iteration to obtain the entire graph representation $\boldsymbol{h}_G$

$$\boldsymbol{h}_G = \zeta(\boldsymbol{h}_i^K, v_i \in V),$$

where function ζ should satisfy the same conditions as δ, *e.g.* being invariant to graph isomorphism. An example could be a particular pooling operation. This representation is then used for the final task, for instance, graph classification.

Depending on the types of graph convolutions (the creation and propagation of messages), one can categorise GCN into spectral-based and spatial-based models. The GCNs defined in the spectral domain (*e.g.* the Chebyshev Spectral GCN) are based on the graph Fourier transform, starting with the construction of the frequency filtering, whereas the spatial domain methods are specified directly on the graph, operating on groups of spatially close neighbours (Zhang et al., 2019). In the following section, a spatial-based GCN with the Gaussian mixture model convolutional operator is applied.

7.5 Potential Application

As motivation of where the proposed approach could be particularly helpful, *system-relevant networks* are considered. The failure of those networks could lead to irreparable harm. An example could be a network, where particular nodes represent ambulance or fire and rescue stations. To guarantee proper functionality, these services are obliged to satisfy a strict policy regarding the time limit for arriving at the accident epicentre. For instance, in emergency medical cases, the ambulance must reach a patient without exceeding the legally prescribed response time.

The fundamental part of the maximum allowed response time is appointed to the travelling time and in some places is not allowed to exceed 12 minutes in 95% of cases. The monitoring of compliance with this rule can be performed by using the control chart for quantile function values (Grimshaw and Alt, 1997). One possibility is to create a test statistic based on the 0.95 and 0.97 quantiles so that the monitoring procedure corresponds to an early warning system and possible deviation towards the maximum limit is detected quicker. However, in the case of a signal, it remains unclear what led to its occurrence unless one inspects the network state. One of the supportive methods in this task would be a GCN which can classify the network states in predefined categories, providing the first insight into a possible issue. This motivation guides the following application demonstrated on the simulated graph-structured data.

7.5.1 Data Simulation

Consider a simplified road network shown in Figure 7.3 whose topology is based on an existing city map. It can be graphically represented by $|V| = 18$ vertices and $|E| = 25$ edges. Various nodal and edge attributes are specified which are described by $\boldsymbol{X}^V$ and $\boldsymbol{L}^E$, respectively. In particular, $\boldsymbol{x}_i$ defines attributes of node v_i and $\boldsymbol{l}_{i,j}$ contains attributes of edge $e_{i,j}$. To differentiate between an ambulance node and a patient region, the vertex attribute *Role* is introduced. One assumes that the ambulance station can serve only one patient at once and that two fixed vertices in total define available ambulance stations. Also, which patient needs help from an ambulance is decided randomly. Thus, another vertex attribute *Involvement in an accident* is included that describes whether an ambulance station provides help (in this case, the value is set to *1*) or is free (the value

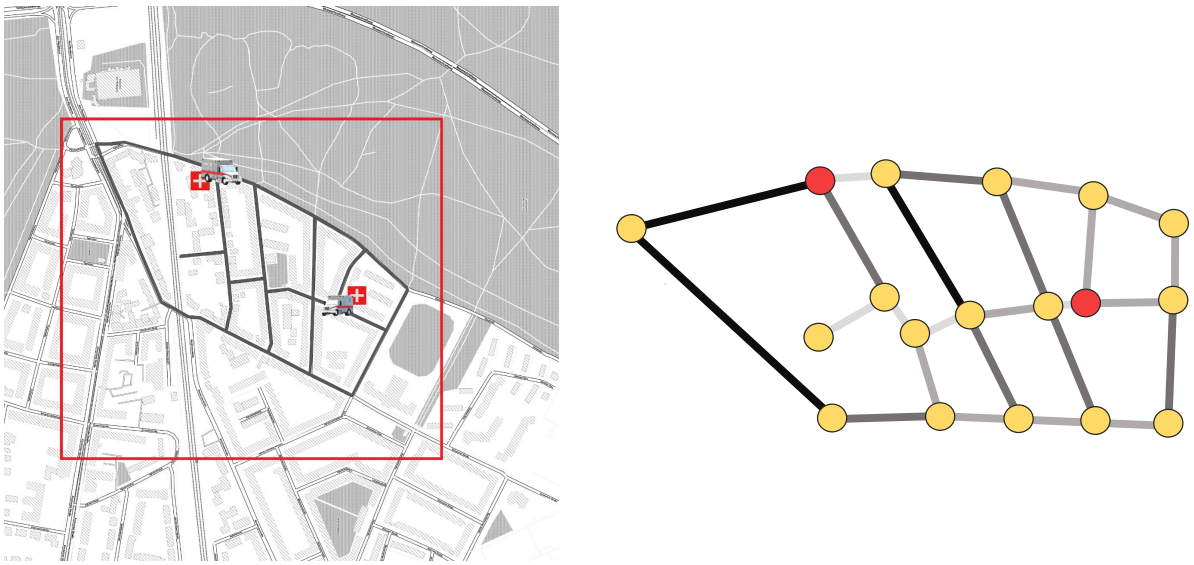


Figure 7.3: An example of a small road network which is derived from an existing city map available on Stamen Maps with two arbitrarily placed ambulance stations (left) and its undirected graph representation (right). Different colours of edges replicate the travelling time along the road, where a darker colour means longer travelling time. The red nodes define ambulance stations.

equals *0*). Considering the patient nodes, as soon as a patient is involved in an accident, the value is set to *1*. It remains *0* if no help is needed or obtained from the ambulance service. Both nodal attributes are contained in $\boldsymbol{X}^V \in \mathbb{R}^2$ and can be found in Table 7.1.

Regarding the edges, one models two characteristics $\boldsymbol{L}^E \in \mathbb{R}^2$ that reflect distinct types of roads. The first continuous attribute defines travelling time in minutes L_1^E (1, 2, 3 and 5 minutes are selected to be the expected values for passing respective roads), which is generated by applying lognormal distribution with different μ and σ. The selected values of μ and σ parameters displayed in Table 7.1 reflect the target state of the network. The second attribute L_2^E defines the level of construction works on the roads. Here, the in-control state is dominated by the attribute values *0* or *1*, which means no or minor roadworks are observed.

7.5.1.1 Generation of the Response Time Data

For monitoring the travelling time from the ambulance station to the patients, some assumptions considering the simulation of the response time data are made. Daily, as soon as there is a patient call with the need for help, the travelling time of the ambulance which is closer to a patient is registered together with the current network situation. Sometimes, the number of accidents can be higher than one at the same time, so the network situation is captured once in this case. However, if the network is in control, the maximum number of simultaneous accidents equals two, otherwise, there exists a personnel shortage. One assumes that the ambulance follows the most efficient route, being the shortest path in terms of travelling time between the ambulance station and the patient. For its calculation, Dijkstra's algorithm is applied.

By the end of the day, the recorded response times are collected so that the 95 % and 97 % quantiles can be derived for defining the test statistic. If the test statistic exceeds the control limit, the collected network data isprovided to the trained GCN that classifies the scenes into four different groups: a stable condition of the road network (label 0), an unstable condition due to the

Type	Attribute	Value
Edge	Travelling time (in minutes)	$\mathbb{E}(1) \sim \mathscr{F}(0.1,0.05^2)$ $\mathbb{E}(2) \sim \mathscr{F}(0.7,0.05^2)$ $\mathbb{E}(3) \sim \mathscr{F}(1.1,0.05^2)$ $\mathbb{E}(5) \sim \mathscr{F}(1.6,0.05^2)$ with $\mathscr{F}(\cdot) = \text{Lognormal}(\mu,\sigma^2)$
	Level of road blocking due to construction work	Free: 0 Low: 1 Middle: 2 High: 3
Node	Role	0: Patient 1: Ambulance
	Involvement in an accident	0: No involvement 1: Help is provided, obtained or needed

Table 7.1: Edge and nodal attributes.

manpower shortage (label 1), an unstable condition due to the construction works (label 2) and an unstable condition due to the traffic jams (label 3). It is important to include the class label 0 graphs which dominate in the definition of the in-control state for the identification of possible false alarms. To proceed with the application of the control chart itself, first, the explanation about how different label groups are designed is given.

7.5.1.2 Road conditions

To discuss the four condition classes, one should concentrate on the problem from the angle of what the neural network should learn in terms of the main differences between the specified scenarios. Considering the reliable state, the neural network needs to distinguish between the problems on the roads, which affect the travelling time so that it potentially leads to the out-of-control case, and which do not, as the patient was still reached on time. It means, despite the obstacles on the roads that can be modelled by increased values of edge attributes, each time the patient was reached by ambulance in or under 12 minutes (simply because the route to the patient was not affected considerably), the graph obtains the label 0.

The class with the label 1 defines the problem of manpower shortage, meaning the reason for longer travelling times is due to the imbalance in the capacity of the ambulance service and the number of patients who need help. In this case, a higher number of patients is generated, *i.e.* more nodes with *Role = 0* are involved in an accident ($X_2^V = 1$). As soon as both ambulance stations provide help and some further patients are not treated yet, the travelling time to these nodes is calculated as $\mathbb{E}(L_1^E) \cdot 2 \cdot 2$, where $\mathbb{E}(L_1^E)$ defines the expected value of the edge feature *Travelling time* multiplied with the number of roads to pass on average and the need to travel first to the ambulance station and then to the patient. In this case, the network is out of control due to the considerably increased travelling time to the third and further patients.

Creating some unreliable situations on roads due to some construction works (label 2), a particular group of roads which come from one or several travelling time distributions is selected and higher values of the second edge attribute L_2^E are assigned to these roads. Thus, the μ and σ parameters are changed accordingly, so that the higher attribute value corresponds to a longer travelling time along the road.

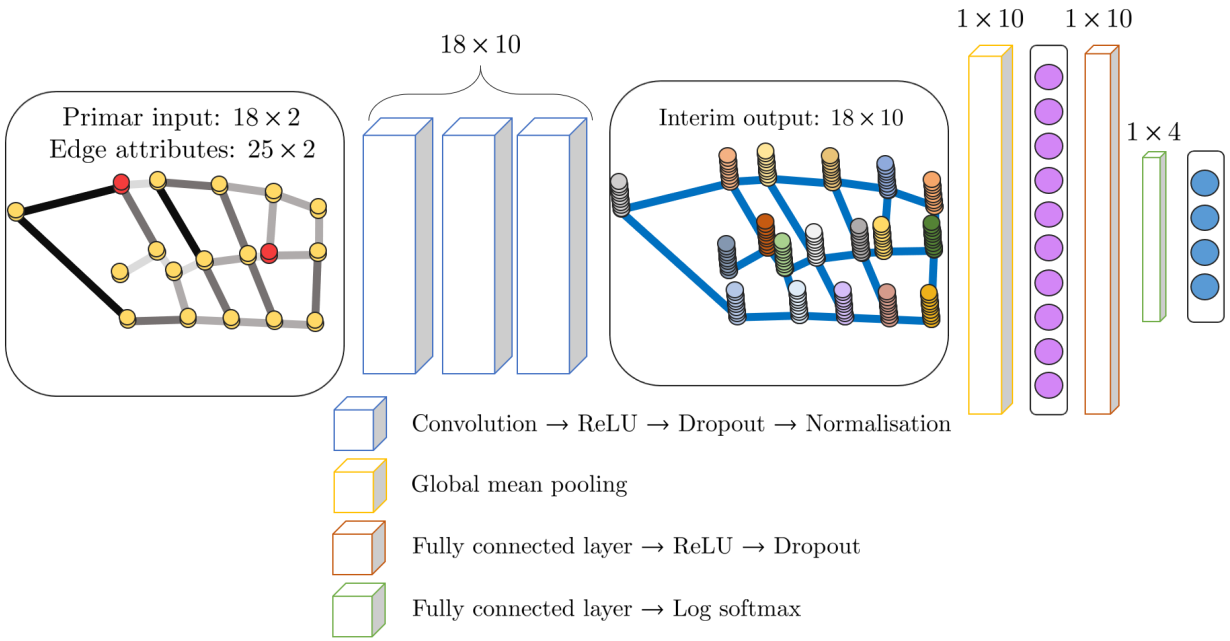


Figure 7.4: The schematic architecture of the applied GCN. Each block represents a single layer where the first stage (blue blocks) contains graph convolutional layers with layer normalisations for learning the feature representation and the second stage (yellow, orange and green blocks) consists of dense layers for classification.

For modelling the traffic jam (label 3), L_2^E is set low (*0* or *1*), and some values describing the travelling time of specific roads are generated from a different distribution that implies their increase. The examples from groups with labels 2 and 3 do not necessarily lead to an out-of-control state if the patients are still reached under the critical time prescription.

After defining the network composition and specifying possible in- and out-of-control scenarios, one can collect the quantile observations for the calibration of the control chart and the graph representations of different classes in order to train the neural network.

7.5.2 Training of the Neural Network

In this simulation study, one is interested in the classification of collected graphs that belong to a change point. The goal is to assign a given graph to one of the predefined categories by learning the feature representation from the provided training data which contains class labels. Consequently, one has to define the GCN architecture so that it can solve the specified task. Also, the graph convolutional operator should be capable of integrating the node, as well as edge attributes into the message passing process because they encompass valuable information about the network's condition.

Figure 7.4 presents the architecture of the applied GCN. The first three graph convolutional layers, each encoding the input in a feature vector of size 18×10, perform three propagation steps and effectively convolve the stage-3 neighbourhood of every node. The Gaussian mixture model convolutional operator described in Monti et al. (2017) has been chosen which is implemented in the programming framework provided by Fey and Lenssen (2019). Each convolutional layer is followed by the ReLU activation function. Afterwards, the dropout operation is applied, which

randomly sets the processed input units to 0 with a specified frequency ξ (here $\xi = 25\%$) during the training time, preventing the model from overfitting, *i.e.* learning from the training dataset without its generalisation. Before the convolution begins, the inputs across the features are normalised; this technique is known as *layer normalisation* (cf. Ba et al., 2016).

After the message passing phase, a readout layer that is defined by a global mean pooling operation transforms the latent vertex representations to a graph representation as a fixed-size vector. Here, the interim output is averaged across each hidden node dimension so that the graph-level output size is 1×10. Next, two fully connected layers are attached to increase the ability to learn a complex function and solve the classification task. Consequently, the second layer predicts the final class probability distribution of size 1×4 followed by the softmax activation function. The ReLU activation function cannot be applied here as it provides continuous output in the range $[0;\infty]$. In the final stage, the output needs to be in the finite range $[0;1]$ for interpreting its results as probabilities, with the highest value corresponding to the predicted class.

After defining the architecture of the GCN, one can start with training or fitting the neural network. This procedure involves the usage of a training dataset to update the model parameters (weights and biases) so that one obtains a reliable mapping between the input (a graph) and the output (a class label). For the training dataset, 2500 graphs are generated. It is important to avoid class imbalance during the training process, therefore, each label is represented by the same number of examples. Another vital part of the training process is the loss function. It calculates the difference between the computed output from the input data (this process is known as *forward pass*) and the value provided as ground truth. Here, the negative log-likelihood loss is chosen, which is appropriate for a multiclass classification problem. It defines the objective function that is minimised while updating the model parameters.

The results that are provided by the loss function are applied in the optimisation step of the parameters, which are based on gradient computation (known as *backpropagation* or *backward pass*). The negative log-likelihood is minimised using the Adaptive Moment Estimation (known as *Adam*) function with a learning rate of 10^{-3} (Kingma and Ba, 2014).

The execution of the backward and forward pass together defines one iteration. During one iteration, usually, a subset of the dataset known as a *mini-batch* is passed. In case one decides to pass all data at once, it is called *batch*. Here, one trains the neural network using the mini-batch with size 16, *i.e.* in every iteration, 16 graphs are processed together. As soon as the entire dataset is passed, one epoch is completed.

As a performance metric that supports the selection of the best model, one computes the weighted F-score after each epoch. Figure 7.5 (left side) illustrates the training and validation history of the applied GCN. To test how well the network generalises to unseen data, one applies the holdout validation method. The validation set, which contains more complex samples, *i.e.* new examples which belong to the classes but are not included in the training dataset, was designed with a size of 800 graphs.

In order not to overtrain the network, one uses early stopping after the 100-th epoch is reached with respect to the F-score improvement of the validation dataset, which terminates the training process if the value has not increased within ten epochs. The optimal model was obtained at epoch 101 with 93.4% and 87.5% being the weighted F-score of the training and the validation dataset, respectively. However, to see whether the model functions correctly, one needs to test it on a new dataset coming from the monitoring procedure.

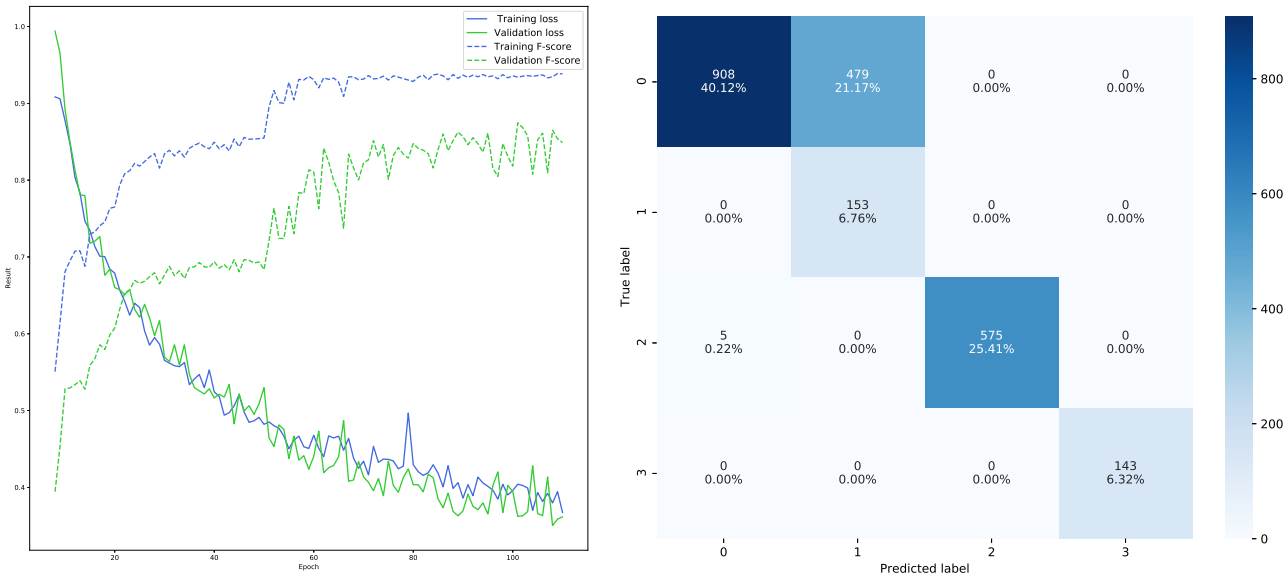


Figure 7.5: The training progress (left) shown on the training (blue curves) and validation sets (green curves). The confusion matrix (right) presents the performance of the trained GCN in Phase II. The numbers on the diagonal define the proportions of correctly classified examples (compared to the size of the complete dataset from Phase II), and the off-diagonal entries correspond to the proportions of the misclassified graphs.

7.5.3 Monitoring and Inspection

In this section, one combines the implementation of a multivariate control chart together with testing the trained GCN. In Phase I, the parameters of the control chart are computed, and in Phase II the monitoring as well as inspection of the detected changes in the road network are performed.

7.5.3.1 Control Chart for Quantile Function Values

In the simulation study, a multivariate control chart for quantile function values is applied. To be precise, the control chart introduced by Grimshaw and Alt (1997) is considered. Other examples of control charts that involve the application of quantile function and recommendations about the chart's calibration can be found in Kanji and Arif (2000), Ning and Wu (2011), Park et al. (2020) and Hwang (2021). To estimate quantile function values of a random variable at the time point t, one needs to obtain a sample of respective observations. In other words, the choice of a control chart for quantile function values indicates that outcomes from a specified period of the random variable are aggregated at each time stamp. In this case, one is interested in the response time of ambulance service, meaning one derives sample quantile function values exactly from this quantity. Although only one process characteristic is employed, the control chart is a multivariate chart due to the specification of two quantile values (*i.e.* $c = 2$) for the calculation of the test statistic in this application.

To calculate daily 95% and 97% quantile values, one randomly simulates between 10 and 100 accidents which are repeatedly assigned to different patients. Next, the shortest path between an available ambulance station and the selected patient is found and the travelling time is saved.

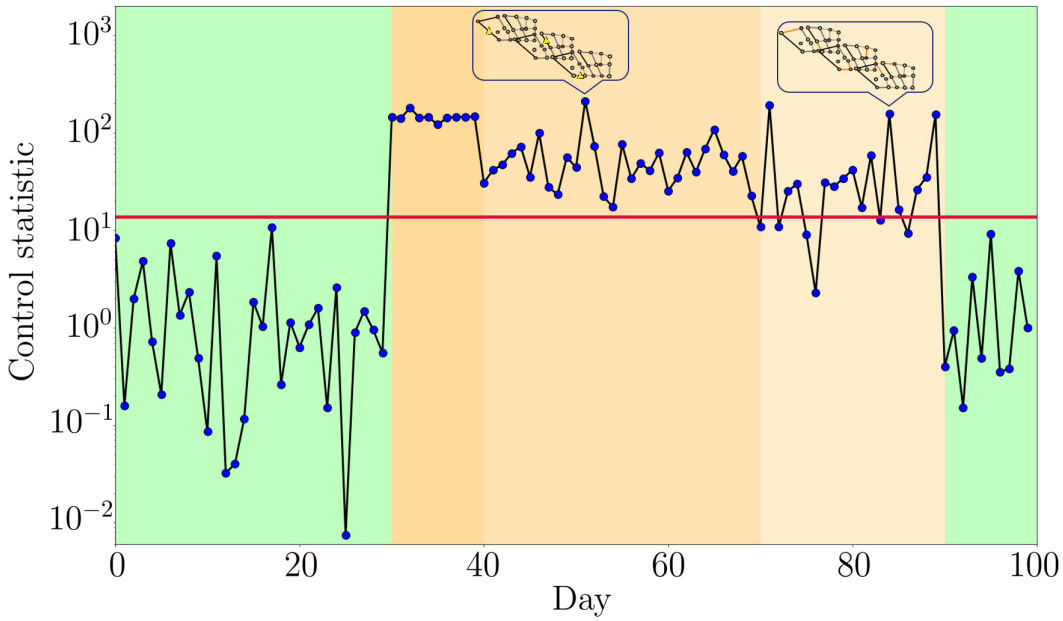


Figure 7.6: The control chart for quantile function values on a logarithmic scale. The horizontal red line corresponds to the control limit. The green areas are designed by the cases with the label 0, the dark orange by the label 1, followed by the labels 2 and 3. The incorporation of graphs with different properties such as construction works (triangular symbols) or traffic jams (orange-coloured edges) defines the availability of additional information to understand the reason for the detected change point.

Using the control chart presented in Grimshaw and Alt (1997), the test statistic is defined as follows

$$a_t = (\hat{\boldsymbol{Q}}_t - \boldsymbol{Q}_0)' \boldsymbol{\Sigma}_0^{-1} (\hat{\boldsymbol{Q}}_t - \boldsymbol{Q}_0),$$

where $\hat{\boldsymbol{Q}}_t = \left(\hat{Q}_{0.95,t}, \hat{Q}_{0.97,t}\right)'$ and the length of $\hat{\boldsymbol{Q}}_t$ is denoted by c. In Phase I, the expected value $\boldsymbol{Q}_0$ is estimated by the empirical mean and $\boldsymbol{\Sigma}_0$ by the sample covariance matrix with 2500 in-control samples. For a sufficiently large number of samples, a_t follows the χ^2 distribution with c degrees of freedom, if the sample at time point t corresponds to the specified in-control state. Hence, the control limit can be defined by $\chi^2_\alpha(c)$, selecting α with respect to the in-control ARL_0 using $\alpha = 1/ARL_0$. Here, ARL_0 = 1000 is chosen, therefore, $\chi^2_{0.001}(2) = 13.816$.

7.5.3.2 Results

For Phase II, one defines the length of the monitoring period to be 100 days, where the network in the first 30 and the last 10 days is considered to be in control. The out-of-control period is designed in the remaining days, where the process is exposed to the personnel shortage (10 days), excessive construction works (30 days) and an increase in traffic jams (20 days). After simulating the cases and calculating the quantiles, one obtains the control chart presented in Figure 7.6. In terms of potential false signals, there are several points that are close to the control limit. The possible reason may be the high variance in the in-control data. One can also notice that not all the test statistics show the out-of-control state in the period when the network was exposed to an increased number of traffic jams. However, they do not define missing signals; as it was mentioned in Section 7.5.1.2, if the ambulance services were still able to reach the patients within the allowed time, then no out-of-control state is given.

Normally, one would apply the neural network only in the out-of-control state to gain insight into the cause. Nevertheless, the primary aim here is to evaluate the performance of the trained GCN to classify provided graph observations in general. Hence, a test dataset using the data from Phase II displayed in Figure 7.6 is created, *i.e.* from the 100 generated days that include both in-control and out-of-control periods and examples from each of the four classes. As one can observe in Figure 7.5 (right side), the GCN can almost flawlessly identify classes 1, 2 and 3. However, class 0 seems to be not well learned, possibly due to a lack of clarity in its representation. Overall, the model achieves a weighted F-score of 82.9% being an encouraging result.

7.6 Discussion

This chapter uncovers the possibility of bringing together statistical process monitoring and deep learning algorithms to monitor efficiently and profoundly graph-structured data. It is a natural question to consider whether one could expand the use of algorithms such as GCNs to encompass the whole monitoring procedure, omitting control charts altogether. Although an appealing idea, the complexity of the model required for real-world data, combined with the amount of training time necessary, would severely limit the applicability of this approach. Applying a hybrid method allows taking advantage of the efficiency of classical techniques while using modern machine learning algorithms in order to specify more subtle network characteristics, which normally require human-lead scrutiny to determine.

Although this chapter was presented from the perspective of monitoring and inspecting graph-structured data, the proposed framework is equally applicable for monitoring ANN applications described in Chapter 6. Samples from the data stream that were determined as being anomalous could be passed through a suitable neural network model designed to classify the cause of the detected change.

8 Conclusion and Outlook

The development of monitoring methods that operate on networks is only a stepping stone on the path toward a significant expansion in gaining insight from available big data and in understanding better our environment. Besides the topic of how to unify statistics and network science for offering progressive monitoring frameworks, there are many other fundamental open questions in this area from the statistical perspective. How to represent the graph data and convolve the information in a unified form, how to identify a suitable modelling approach to a specific network problem, when to yield machine learning methods for assisting the change point detection: These and many other challenges are yet to be conquered.

This thesis is concerned with bringing together network theory and statistical process monitoring to create enhanced frameworks for network surveillance. Before anything else, it is vital to gain an initial sense of the context in which the network of interest arises, so that an appropriate statistical foundation for its analysis could emerge. Here, all technical chapters are unified by the idea of first compactly representing the considered network-related process either by modelling or applying other dimensionality reduction techniques and then proceeding with its monitoring using control charts. Overall, choosing a particular control chart depends on the specific problem, requiring a compromise between the number of signals in Phase II (in-control part) and correctly detected out-of-control samples. As soon as one concludes what is important, *e.g.* computation time or robustness, the proposed monitoring approaches can be customised to satisfactorily support any network-related applications.

With respect to monitoring networks as graph-structured data, there are several future research directions that are relevant for both fixed and random types of network monitoring. First of all, the discovery of scalable monitoring methods or ways to scale recently proposed approaches is vital. As it can be derived from the empirical illustrations in Chapters 4 and 5, the currently suitable sizes of networks in presented monitoring frameworks vary from small to medium. It would be challenging to perform real-time surveillance of larger networks. Another crucial point is the time scale: In this thesis, only the observations corresponding to discrete time stamps are considered. However, a realistic network system evolves continuously, so that the inspection of when this aspect begins playing a significant role in the quality of monitoring is decisive.

Regarding the design of control charts, a possible extension is to create a monitoring process when the values of the variance-covariance matrix can vary between the in-control and out-of-control states during the application of multivariate parametric control charts as used in Chapter 4. Whether this factor would beneficially enrich the surveillance remains open for future research. Moreover, further development of adaptive control charts as presented in Section 4.4 for both random and fixed network monitoring is interesting as they could remarkably improve the performance of the change detection (cf. Sparks and Wilson, 2019).

Having a network where the number of vertices differs over time, the current TERGM framework would model it by either removing or incorporating particular nodes as structural zeroes. However, the development of alternative solutions to address this issue (for instance, Krivitsky et al. (2011) introduce an offset term for the ERGM) as well as the expansion of the presented approach for monitoring the node composition, is subject to future research.

As an empirical illustration in Section 5.5, monitoring of bilateral electricity flows across Europe is presented, where a static network structure that fits well with the considered process is selected. However, thinking of potential processes with a changing underlying structure over time, it is important to determine whether the monitoring framework would experience substantial changes in terms of re-estimating the parameters in Phase I when some nodes or edges disappear with time. Moreover, the effectiveness of the proposed monitoring procedure strongly relies on the assumption of the GNARX model being suitable to the considered data. Thus, in case the data cannot be well represented by the GNARX model, *i.e.* the count data are given, an alternative modelling approach or an extension to the currently existing model is required.

Another potential aspect that could benefit the monitoring field is the usage of advanced network representations. An alternative way of seeing a connection within a network demands extensive study of its benefit for monitoring complex, especially technical systems. For example, multi-dimensional networks, where a network is created by multiple layers with entities being inter- and intraconnected, or k-networks (an edge running not between 2 but between k entities) could serve as useful construction mechanisms. However, for developing reliable and efficient statistical monitoring approaches for technical systems, stronger cooperation and effective communication between different scientific communities is required. Currently, there is considerable potential for improvement in this regard (cf. Stevens et al., 2021a).

Coming to the monitoring of artificial neural networks, for some applications, which process autocorrelated data streams, an alternative definition of a (rank-based) test statistic with temporal dependency could be beneficial. Moreover, the examination of the reference samples, especially the influence of their size and the construction method, has shown that a bigger reference sample is not automatically related to a better detection capability. In particular, there are open questions about attaining a reliable reference sample. It is advisable to research in more detail how the statistical process monitoring techniques such as the multivariate mean-rank chart (Bell et al., 2014), could support the analysis of Phase I data. In addition, it is subject to future research when the reference sample should be updated or continually augmented with recent observations while preventing contamination with the out-of-control data.

It could be shown that the proposed approach in Section 6.3 would achieve a better performance if additional misclassification information was available. In general, understanding when a data point has obtained a wrong prediction is indispensable and requires an additional method to be developed that could be later combined with the designed monitoring procedure. Moreover, one could investigate whether subdividing the data stream in moving windows would enhance the monitoring performance.

In the presented experiments in Section 6.5, the data points of the training and the test datasets are chosen following a convention by randomly dividing the dataset. In the future, it is recommendable to examine whether an optimal splitting of data into training and testing which preserves distributional similarity improves the performance of models based on artificial intelligence as well as leads to more reliable monitoring (cf. Vakayil and Joseph, 2022). Additionally, the field of data splitting and data compression, *i.e.* how to find a trade-off between reliable but fast training and a well-chosen training set that uses only a fraction of the initial dataset, is relevant for future research.

During the entire Chapters 6 and 7 well-balanced classification problems are considered. However, class imbalance is a frequent challenge in training ANNs and needs to be considered in future research (cf. Ghazikhani et al., 2013). Furthermore, whether the proposed methodology could be applied to monitoring semi-supervised or unsupervised learning models remains open. Despite

the challenges in designing a universal monitoring scheme for AI-based approaches, extending the presented framework to other AI algorithms seems to be a promising field. Additionally, there are different types of concept drift or nonstationarity (cf. Hu et al., 2020), meaning that further development of methods and their comparison is of practical importance.

The extension introduced in Chapter 7 devotes attention to the development of a monitoring procedure that incorporates the stage of automatically detecting the truly anomalous observations in the aggregated data samples and the identification of a cause that led to a signal. Currently, these both stages are usually researched separately, but the scientific direction of developing a unified method for change identification that combines change detection and change inspection steps suggests it to be worth investigating as an emerging research area.

Concluding the thesis, it is worth making a statement related to the discussion about the coexistence of statistics and machine learning methods. This thesis is based on the strong belief that a better way to understand the relationship between both frameworks is that machine learning is the logical next step in response to the growing volume of data. Thus, it is beneficial to see the successes in artificial intelligence applications not as an attempt to replace the traditional statistical methods but as a direction towards their enhancement, making statistics even more powerful. As presented in Chapter 6, statistical process monitoring can also enhance artificial intelligence applications by guaranteeing the reliability of their results but in turn they offer an efficient inspection procedure as a supplement to the actual monitoring as introduced in Chapter 7. Hence, continuing to develop approaches that use the synergy effects from both frameworks seems to be the solution for creating meaningful and powerful innovations for statistical process monitoring of networks and beyond.

List of Figures

List of Tables

Bibliography

Abraham, Y., Gerrits, B., Ludwig, M.-G., Rebhan, M., Gubser Keller, C., 2017. Exploring Glucocorticoid Receptor Agonists Mechanism of Action Through Mass Cytometry and Radial Visualizations. *Cytometry Part B: Clinical Cytometry* 92 (1), pp. 42–56.

Abrell, J., Rausch, S., 2016. Cross-country electricity trade, renewable energy and European transmission infrastructure policy. *Journal of Environmental Economics and Management* 79, pp. 87–113.

Afshani, P., Sheehy, D. R., Stein, Y., 2016. Approximating the simplicial depth in high dimensions. In: *The European Workshop on Computational Geometry*.

Ahuja, R. K., Magnanti, T. L., Orlin, J. B., 1993. Network Flows: Theory, Algorithms, and Applications. Prentice Hall.

Akoglu, L., Tong, H., Koutra, D., 2015. Graph-based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29 (3), pp. 626–688.

Aldridge, I., Avellaneda, M., 2019. Neural Networks in Finance: Design and Performance. *The Journal of Financial Data Science* 1 (4), pp. 39–62.

Alexopoulos, C., Goldsman, D., Tsui, K.-L., Jiang, W., 2004. SPC monitoring and variance estimation. *Frontiers in Statistical Quality Control 7*, pp. 194–209.

Ali, S., Pievatolo, A., Göb, R., 2016. An Overview of Control Charts for High-quality Processes. *Quality and reliability engineering international* 32 (7), pp. 2171–2189.

Alwan, L. C., 1992. Effects of autocorrelation on control chart performance. *Communications in Statistics – Theory and Methods* 21 (4), pp. 1025–1049.

Aminikhanghahi, S., Cook, D. J., 2017. A survey of methods for time series change point detection. *Knowledge and information systems* 51 (2), pp. 339–367.

Angermueller, C., Pärnamaa, T., Parts, L., Stegle, O., 2016. Deep learning for computational biology. *Molecular systems biology* 12 (7).

Apsemidis, A., Psarakis, S., 2020. Support vector machines: A review and applications in statistical process monitoring. *Data Analysis and Applications 3: Computational, Classification, Financial, Statistical and Stochastic Methods* 5, pp. 123–144.

Apsemidis, A., Psarakis, S., Moguerza, J. M., 2020. A review of machine learning kernel methods in statistical process monitoring. *Computers & Industrial Engineering* 142.

Asif, N. A., Sarker, Y., Chakrabortty, R. K., Ryan, M. J., Ahamed, M. H., Saha, D. K., Badal, F. R., Das, S. K., Ali, M. F., Moyeen, S. I., Islam, M. R., 2021. Graph Neural Network: A Comprehensive Review on Non-Euclidean Space. *IEEE Access* 9, pp. 60588–60606.

Avrachenkov, K., Dreveton, M., 2022. Statistical Analysis of Networks. Now Publishers, Inc.

Azarnoush, B., Paynabar, K., Bekki, J., Runger, G., 2016. Monitoring Temporal Homogeneity in Attributed Network Streams. *Journal of Quality Technology* 48 (1), pp. 28–43.

Ba, J. L., Kiros, J. R., Hinton, G. E., 2016. Layer normalization. In: *arXiv preprint: 1607.06450*.

Baddeley, A., Nair, G., Rakshit, S., McSwiggan, G., Davies, T. M., 2021. Analysing point patterns on networks – A review. *Spatial Statistics* 42.

Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., Morales-Bueno, R., 2006. Early Drift Detection Method. In: *Fourth international workshop on knowledge discovery from data streams*. pp. 77–86.

Ball, N. M., Brunner, R. J., Myers, A. D., Tcheng, D., 2006. Robust Machine Learning Applied to Astronomical Data Sets. I. Star-Galaxy Classification of the Sloan Digital Sky Survey DR3 Using Decision Trees. *The Astrophysical Journal* 650 (1).

Barabási, A.-L., Albert, R., 1999. Emergence of scaling in random networks. *Science* 286 (5439), pp. 509–512.

Barale, M., Shirke, D., 2019. Nonparametric control charts based on data depth for location parameter. *Journal of Statistical Theory and Practice* 13 (3), pp. 1–19.

Baranowski, J., Dudek, A., Mularczyk, R., 2021. Transient Anomaly Detection Using Gaussian Process Depth Analysis. In: *The International Conference on Methods and Models in Automation and Robotics (MMAR)*. IEEE, pp. 221–226.

Basseville, M., Nikiforov, I. V., 1993. Detection of Abrupt Changes: Theory and Application. Prentice Hall Englewood Cliffs.

Bell, R. C., Jones-Farmer, L. A., Billor, N., 2014. A Distribution-Free Multivariate Phase I Location Control Chart for Subgrouped Data from Elliptical Distributions. *Technometrics* 56 (4), pp. 528–538.

Besag, J., 1974. Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society: Series B (Methodological)* 36 (2), pp. 192–225.

Bifet, A., Gavalda, R., 2007. Learning from time-changing data with adaptive windowing. In: *Proceedings of the 2007 SIAM international conference on data mining*. pp. 443–448.

Bifet, A., Gavalda, R., Holmes, G., Pfahringer, B., 2018. Machine Learning for Data Streams: with Practical Examples in MOA. MIT Press.

Block, P., Koskinen, J., Hollway, J., Steglich, C., Stadtfeld, C., 2018. Change we can believe in: Comparing longitudinal network models on consistency, interpretability and predictive power. *Social Networks* 52, pp. 180–191.

Boone, J., Chakraborti, S., 2012. Two simple Shewhart-type multivariate nonparametric control charts. *Applied Stochastic Models in Business and Industry* 28 (2), pp. 130–140.

Box, G. E., Cox, D. R., 1964. An analysis of transformations. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 26 (2), pp. 211–243.

Breiman, L., Meisel, W., Purcell, E., 1977. Variable Kernel Estimates of Multivariate Densities. *Technometrics* 19 (2), pp. 135–144.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., Sander, J., 2000. LOF: identifying density-based local outliers. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. pp. 93–104.

Brevier, G., Rizzi, R., Vialette, S., 2007. Pattern Matching in Protein-Protein Interaction Graphs. In: *International Symposium on Fundamentals of Computation Theory*. Springer.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P., 2017. Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine* 34 (4), pp. 18–42.

Cai, H., Zheng, V. W., Chang, K. C. C., 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Transactions on Knowledge and Data Engineering* 30 (9), pp. 1616–1637.

Calin, O., 2020. Deep Learning Architectures. Springer International Publishing.

Cannings, C., Penman, D., 2003. Models of random graphs and their applications. *Stochastic Processes: Modelling and Simulation* 21, pp. 51–91.

Caro, L. D., Frias-Martinez, V., Frias-Martinez, E., 2010. Analyzing the role of dimension arrangement for data visualization in radviz. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, pp. 125–132.

Carrington, P. J., Scott, J., Wasserman, S., 2005. Models and methods in social network analysis. Cambridge University Press.

Cascos, I., López-Díaz, M., 2018. Control charts based on parameter depths. *Applied Mathematical Modelling* 53, pp. 487–509.

Celano, G., Castagliola, P., Fichera, S., Nenes, G., 2013. Performance of t control charts in short runs with unknown shift sizes. *Computers & Industrial Engineering* 64 (1), pp. 56–68.

Chen, F., Wang, Y. C., Wang, B., Kuo, C. C. J., 2020. Graph representation learning: A Survey. *APSIPA Transactions on Signal and Information Processing* 9.

Chen, L., Yu, T., Liu, M., 2015. A Semantic Graph Model. In: *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*. Springer, pp. 378–396.

Chen, S., Yu, J., 2019. Deep recurrent neural network-based residual control chart for autocorrelated processes. *Quality and Reliability Engineering International* 35 (8), pp. 2687–2708.

Chiroma, H., Abdullahi, U. A., Alarood, A. A., Gabralla, L. A., Rana, N., Shuib, L., Hashem, I. A. T., Gbenga, D. E., Abubakar, A. I., Zeki, A. M., 2018. Progress on Artificial Neural Networks for Big Data Analytics: A Survey. *IEEE Access* 7, pp. 70535–70551.

Cook, R. D., Ni, L., 2005. Sufficient dimension reduction via inverse regression: A minimum discrepancy approach. *Journal of the American Statistical Association* 100 (470), pp. 410–428.

Corbière, C., Thome, N., Bar-Hen, A., Cord, M., Pérez, P., 2019. Addressing Failure Prediction by Learning Model Confidence. *Advances in Neural Information Processing Systems* 32.

Crosier, R. B., 1988. Multivariate Generalizations of Cumulative Sum Quality-Control Schemes. *Technometrics* 30 (3), pp. 291–303.

Dang, X., Serfling, R., 2010. Nonparametric depth-based multivariate outlier identifiers, and masking robustness properties. *Journal of Statistical Planning and Inference* 140 (1), pp. 198–213.

Das, H., Mishra, S. K., Roy, D. S., 2013. The Topological Structure of the Odisha Power Grid: a Complex Network Analysis. *IJMCA* 1 (1), pp. 12–16.

De Masi, G., Gallegati, M., 2007. Debt-credit economic networks of banks and firms: the Italian case. In: *Econophysics of Markets and Business Networks: Proceedings of the Econophys-Kolkata III*. Springer, pp. 159–171.

Demšar, J., Bosnić, Z., 2018. Detecting concept drift in data streams using model explanation. *Expert Systems with Applications* 92, pp. 546–559.

Diestel, R., 2017. Graph Theory. Springer.

Diren, D. D., Boran, S., Cil, I., 2020. Integration of machine learning techniques and control charts in multivariate processes. *Scientia Iranica* 27 (6), pp. 3233–3241.

Donoho, D. L., Gasko, M., 1992. Breakdown properties of location estimates based on halfspace depth and projected outlyingness. *The Annals of Statistics*, pp. 1803–1827.

Dua, D., Graff, C., 2019. UCI Machine Learning Repository. URL `http://archive.ics.uci.edu/ml`

Dyckerhoff, R., Mozharovskyi, P., Nagy, S., 2021. Approximate computation of projection depths. *Computational Statistics & Data Analysis* 157.

Emambocus, B. A. S., Jasser, M. B., Amphawan, A., 2023. A Survey on the Optimization of Artificial Neural Networks Using Swarm Intelligence Algorithms. *IEEE Access* 11, pp. 1280–1294.

Erdős, P., Rényi, A., 1959. On random graphs. *Publ. Math. Debrecen* 6, pp. 290–297.

Fang, Z., Li, Y., Lu, J., Dong, J., Han, B., Liu, F., 2022. Is Out-of-Distribution Detection Learnable? *Advances in Neural Information Processing Systems* 35, pp. 37199–37213.

Farahani, E. M., Baradaran Kazemzadeh, R., Noorossana, R., Rahimian, G., 2017. A statistical approach to social network monitoring. *Communications in Statistics-Theory and Methods* 46 (22), pp. 11272–11288.

Fey, M., Lenssen, J. E., 2019. Fast graph representation learning with PyTorch Geometric. *arXiv preprint: 1903.02428*.

Flossdorf, J., Jentsch, C., 2021. Change detection in dynamic networks using network characteristics. *IEEE Transactions on Signal and Information Processing over Networks* 7, pp. 451–464.

Fonseca-Pedrero, E., 2018. Network analysis in psychology. *Papeles del Psicólogo* 39 (1), pp. 1–12.

Forrest, S., Perelson, A. S., Allen, L., Cherukuri, R., 1994. Self-nonself discrimination in a computer. In: *Proceedings of 1994 IEEE computer society symposium on research in security and privacy*. IEEE, pp. 202–212.

Fort, S., Ren, J., Lakshminarayanan, B., 2021. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems* 34, pp. 7068–7081.

Fountoulaki, A., Karacapilidis, N., Manatakis, M., 2011. Augmenting statistical quality control with machine learning techniques: an overview. *International Journal of Business and Systems Research* 5 (6), pp. 610–626.

Francisci, G., Nieto-Reyes, A., Agostinelli, C., 2019. Generalization of the simplicial depth: no vanishment outside the convex hull of the distribution support. *arXiv preprint: 1909.02739*.

Frank, O., 1991. Statistical analysis of change in networks. *Statistica Neerlandica* 45 (3), pp. 283–293.

Frank, O., Strauss, D., 1986. Markov Graphs. *Journal of the American Statistical Association* 81 (395), pp. 832–842.

Gama, J., Medas, P., Castillo, G., Rodrigues, P., 2004. Learning with drift detection. In: *Brazilian symposium on artificial intelligence*. Springer, pp. 286–295.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A., 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46 (4), pp. 1–37.

Gao, X., Pishdad-Bozorgi, P., Shelden, D. R., Hu, Y., 2019. Machine learning applications in facility life-cycle cost analysis: A review. *Computing in Civil Engineering 2019: Smart Cities, Sustainability, and Resilience*, pp. 267–274.

Garcia, K. D., Poel, M., Kok, J. N., de Carvalho, A. C., 2019. Online clustering for novelty detection and concept drift in data streams. In: *EPIA Conference on Artificial Intelligence*. Springer, pp. 448–459.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., 2023. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review* 56 (Suppl 1), pp. 1513–1589.

Gemaque, R. N., Costa, A. F. J., Giusti, R., Dos Santos, E. M., 2020. An overview of unsupervised drift detection methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10 (6).

Ghazikhani, A., Monsefi, R., Yazdi, H. S., 2013. Ensemble of online neural networks for non-stationary and imbalanced data streams. *Neurocomputing* 122, pp. 535–544.

Ghosh, A. K., Chaudhuri, P., Sengupta, D., 2006. Classification Using Kernel Density Estimates: Multiscale Analysis and Visualization. *Technometrics* 48 (1), pp. 120–132.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., Dahl, G. E., 2017. Neural message passing for quantum chemistry. *International Conference on Machine Learning*.

Gogoglou, A., Bruss, C. B., Nguyen, B., Sarshogh, R., Hines, K. E., 2020. Quantifying Challenges in the Application of Graph Representation Learning. In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, pp. 1519–1526.

Goldberg, Y., 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research* 57, pp. 345–420.

Goldstein, M., Uchida, S., 2016. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one* 11 (4).

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep learning. MIT Press.

Gorman, R. P., Sejnowski, T. J., 1988. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural networks* 1 (1), pp. 75–89.

Grennan, K. S., Chen, C., Gershon, E. S., Liu, C., 2014. Molecular network analysis enhances understanding of the biology of mental disorders. *Bioessays* 36 (6), pp. 606–616.

Grimshaw, S. D., Alt, F. B., 1997. Control Charts for Quantile Function Values. *Journal of Quality Technology* 29 (1), pp. 1–7.

Grundy, T. D., 2021. On Aspects of Changepoint Analysis Motivated by Industrial Applications. Lancaster University (United Kingdom).

Guttormsson, S. E., Marks, R., El-Sharkawi, M., Kerszenbaum, I., 1999. Elliptical novelty grouping for on-line short-turn detection of excited running rotors. *IEEE Transactions on Energy Conversion* 14 (1), pp. 16–22.

Hamilton, W. L., 2020. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14 (3), pp. 1–159.

Hamilton, W. L., Ying, R., Leskovec, J., 2017. Representation learning on graphs: Methods and applications. *arXiv preprint: 1709.05584*.

Handcock, M. S., 2003. Assessing degeneracy in statistical models of social networks. Working Paper No. 39, Center for Statistics and the Social Sciences, University of Washington, Seattle.

Hanneke, S., Fu, W., Xing, E. P., 2010. Discrete temporal models of social networks. *Electronic Journal of Statistics* 4, pp. 585–605.

Haque, A., Khan, L., Baron, M., Thuraisingham, B., Aggarwal, C., 2016. Efficient handling of concept drift and concept evolution over stream data. In: *IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, pp. 481–492.

He, R., Zheng, T., 2015. GLMLE: graph-limit enabled fast computation for fitting exponential random graph models to large social networks. *Social Network Analysis and Mining* 5, pp. 1–19.

Heard, N. A., Weston, D. J., Platanioti, K., Hand, D. J., 2010. Bayesian anomaly detection methods for social networks. *The Annals of Applied Statistics* 4 (2), pp. 645–662.

Heipke, C., Rottensteiner, F., 2020. Deep learning for geometric and semantic tasks in photogrammetry and remote sensing. *Geo-spatial Information Science* 23 (1), pp. 10–19.

Hendrycks, D., Gimpel, K., 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint: 1610.02136*.

Hermans, M., Schrauwen, B., 2013. Training and analysing deep recurrent neural networks. *Advances in neural information processing systems* 26, pp. 190–198.

Hoffman, P., Grinstein, G., Pinkney, D., 1999. Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations. In: *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*. pp. 9–16.

Holland, P. W., Laskey, K. B., Leinhardt, S., 1983. Stochastic blockmodels: First steps. *Social networks* 5 (2), pp. 109–137.

Holland, P. W., Leinhardt, S., 1981. An Exponential Family of Probability Distributions for Directed Graphs: Rejoinder. *Journal of the American Statistical Association* 76 (373), pp. 62–65.

Horváth, L., Hušková, M., Kokoszka, P., Steinebach, J., 2004. Monitoring changes in linear models. *Journal of statistical Planning and Inference* 126 (1), pp. 225–251.

Hosseini, S. S., Noorossana, R., 2018. Performance evaluation of EWMA and CUSUM control charts to detect anomalies in social networks using average and standard deviation of degree measures. *Quality and Reliability Engineering International* 34 (4), pp. 477–500.

Hu, H., Kantardzic, M., Sethi, T. S., 2020. No Free Lunch Theorem for concept drift detection in streaming data classification: A review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10 (2).

Huang, J., Zhu, Q., Yang, L., Feng, J., 2016. A non-parameter outlier detection algorithm based on natural neighbor. *Knowledge-Based Systems* 92, pp. 71–77.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., Tygar, J. D., 2011. Adversarial machine learning. In: *Proceedings of the 4th ACM workshop on Security and artificial intelligence*. pp. 43–58.

Hunter, D. R., Goodreau, S. M., Handcock, M. S., 2008. Goodness of fit of social network models. *Journal of the American Statistical Association* 103 (481), pp. 248–258.

Hwang, W. Y., 2021. Quantile-based control charts for Poisson and gamma distributed data. *Journal of the Korean Statistical Society* 50 (4), pp. 1129–1146.

Ivanovs, J., Mozharovskyi, P., 2021. Distributionally robust halfspace depth. *arXiv preprint: 2101.00726*.

Jackson, M., 2015. The past and future of network analysis in economics. In: *The Oxford Handbook of the Economics of Networks*.

Jeske, D. R., Stevens, N. T., Tartakovsky, A. G., Wilson, J. D., 2018. Statistical methods for network surveillance. *Applied Stochastic Models in Business and Industry* 34 (4), pp. 425–445.

Johnson, R. A., Wichern, D. W., 2007. Applied Multivariate Statistical Analysis, 6. Edition. Prentice Hall Upper Saddle River, NJ.

Jones-Farmer, L. A., Woodall, W. H., Steiner, S. H., Champ, C. W., 2014. An Overview of Phase I Analysis for Process Improvement and Monitoring. *Journal of Quality Technology* 46 (3), pp. 265–280.

Joseph, J., Pignatiello, J., Runger, G. C., 1990. Comparisons of Multivariate CUSUM Charts. *Journal of Quality Technology* 22 (3), pp. 173–186.

Kan, S. H., 2003. Metrics and models in software quality engineering. Addison-Wesley Professional.

Kang, B.-S., Park, S.-C., 2000. Integrated machine learning approaches for complementing statistical process control procedures. *Decision Support Systems* 29 (1), pp. 59–72.

Kanji, G. K., Arif, O. H., 2000. Median rankit control chart by the quantile approach. *Journal of applied statistics* 27 (6), pp. 757–770.

Karrer, B., Newman, M. E., 2011. Stochastic blockmodels and community structure in networks. *Physical review E* 83 (1).

Kenett, R. S., Pollak, M., 2012. On assessing the performance of sequential procedures for detecting a change. *Quality and Reliability Engineering International* 28 (5), pp. 500–507.

Khoza, S. C., Grobler, J., 2019. Comparing machine learning and statistical process control for predicting manufacturing performance. In: *EPIA conference on artificial intelligence*. Springer, pp. 108–119.

Khrabrov, A., Cybenko, G., 2010. Discovering influence in communication networks using dynamic graph analysis. In: *2010 IEEE Second International Conference on Social Computing*. IEEE, pp. 288–294.

Kim, J.-M., Ha, I. D., 2022. Deep learning-based residual control chart for count data. *Quality Engineering* 34 (3), pp. 370–381.

Kim, M.-S., Kong, H.-J., Hong, S.-C., Chung, S.-H., Hong, J. W., 2004. A flow-based method for abnormal network traffic detection. In: *2004 IEEE/IFIP network operations and management symposium*. IEEE, pp. 599–612.

Kim, Y., Park, C. H., 2017. An efficient concept drift detection method for streaming data under limited labeling. *IEICE Transactions on Information and systems* 100 (10), pp. 2537–2546.

Kingma, D. P., Ba, J. L., 2014. Adam: A Method for Stochastic Optimization. In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Kipf, T. N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: *International Conference on Learning Representations (ICLR)*.

Klinkenberg, R., Joachims, T., 2000. Detecting concept drift with support vector machines. In: *International Conference on Machine Learning (ICML)*. pp. 487–494.

Klinkenberg, R., Renz, I., 1998. Adaptive information filtering: Learning in the presence of concept drifts. *Learning for Text Categorization*, pp. 33–40.

Knight, M., Leeming, K., Nason, G., Nunes, M., 2020. Generalized Network Autoregressive Processes and the GNAR Package. *Journal of Statistical Software* 96 (5), pp. 1–36.

Kolaczyk, E. D., 2009a. Analysis of Network Flow Data. *Statistical Analysis of Network Data: Methods and Models*, pp. 285–331.

Kolaczyk, E. D., 2009b. Models for Network Graphs. *Statistical Analysis of Network Data: Methods and Models*, pp. 153–196.

Kolaczyk, E. D., Csárdi, G., 2014. Statistical Analysis of Network Data with R. Springer.

Kolaczyk, E. D., Krivitsky, P. N., 2015. On the question of effective sample size in network modeling: An asymptotic inquiry. *Statistical Science: a Review Journal of the Institute of Mathematical Statistics* 30 (2), pp. 184.

Kourti, T., MacGregor, J. F., 1996. Multivariate SPC methods for process and product monitoring. *Journal of quality technology* 28 (4), pp. 409–428.

Krawczyk, B., Woźniak, M., 2015. One-class classifiers with incremental learning and forgetting for data streams with concept drift. *Soft Computing* 19 (12), pp. 3387–3400.

Krivitsky, P. N., Handcock, M. S., 2014. A separable model for dynamic networks. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76 (1), pp. 29–46.

Krivitsky, P. N., Handcock, M. S., Morris, M., 2011. Adjusting for network size and composition effects in exponential-family random graph models. *Statistical methodology* 8 (4), pp. 319–339.

Krizhevsky, A., Hinton, G., 2009. Learning multiple layers of features from tiny images. *Toronto, ON, Canada*.

Kumagai, A., Iwata, T., Fujiwara, Y., 2021. Semi-supervised anomaly detection on attributed graphs. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.

Kuncheva, L. I., 2009. Using control charts for detecting concept change in streaming data. *Bangor University*, pp. 48.

Lange, T., Mosler, K., Mozharovskyi, P., 2014. Fast nonparametric classification based on data depth. *Statistical Papers* 55 (1), pp. 49–69.

Lee, C., Wilkinson, D. J., 2019. A review of stochastic block models and extensions for graph clustering. *Applied Network Science* 4 (1), pp. 1–50.

Lee, K., Lee, K., Lee, H., Shin, J., 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems* 31.

Leifeld, P., Cranmer, S. J., 2019. A theoretical and empirical comparison of the temporal exponential random graph model and the stochastic actor-oriented model. *Network Science* 7 (1), pp. 20–51.

Leifeld, P., Cranmer, S. J., Desmarais, B. A., 2018. Temporal Exponential Random Graph Models with btergm: Estimation and Bootstrap Confidence Intervals. *Journal of Statistical Software* 83 (6).

Leitch, J., Alexander, K. A., Sengupta, S., 2019. Toward epidemic thresholds on temporal networks: a review and open questions. *Applied Network Science* 4 (1).

Li, H., Deng, J., Yuan, S., Feng, P., Arachchige, D. D., 2021a. Monitoring and identifying wind turbine generator bearing faults using deep belief network and EWMA control charts. *Frontiers in Energy Research* 9.

Li, J., Cuesta-Albertos, J. A., Liu, R. Y., 2012. *DD*-classifier: Nonparametric classification procedure based on *DD*-plot. *Journal of the American Statistical Association* 107 (498), pp. 737–753.

Li, J., Jin, J., Shi, J., 2008. Causation-based T^2 decomposition for multivariate process monitoring and diagnosis. *Journal of Quality Technology* 40 (1), pp. 46–58.

Li, K.-C., 1991. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association* 86 (414), pp. 316–327.

Li, L., 2007. Sparse sufficient dimension reduction. *Biometrika* 94 (3), pp. 603–613.

Li, L., Nachtsheim, C. J., 2006. Sparse Sliced Inverse Regression. *Technometrics* 48 (4), pp. 503–510.

Li, P., Wu, X., Hu, X., Wang, H., 2015. Learning concept-drifting data streams with random ensemble decision trees. *Neurocomputing* 166, pp. 68–83.

Li, S., Pandey, A., Hooi, B., Faloutsos, C., Pileggi, L., 2021b. Dynamic graph-based anomaly detection in the electrical grid. *IEEE Transactions on Power Systems* 37 (5), pp. 3408–3422.

Lin, Q., Zhao, Z., Liu, J. S., 2019. Sparse Sliced Inverse Regression Via Lasso. *Journal of the American Statistical Association* 114 (528), pp. 1726–1739.

Liu, F. T., Ting, K. M., Zhou, Z.-H., 2008. Isolation Forest. In: *2008 Eighth IEEE International Conference on Data Mining*. pp. 413–422.

Liu, R. Y., 1990. On a Notion of Data Depth Based on Random Simplices. *The Annals of Statistics* 18 (1), pp. 405–414.

Liu, R. Y., 1995. Control charts for multivariate processes. *Journal of the American Statistical Association* 90 (432), pp. 1380–1387.

Liu, R. Y., Serfling, R. J., Souvaine, D. L., 2006. Data Depth: Robust Multivariate Analysis, Computational Geometry, and Applications. American Mathematical Society.

Liu, R. Y., Singh, K., 1993. A Quality Index Based on Data Depth and Multivariate Rank Tests. *Journal of the American Statistical Association* 88 (421), pp. 252–260.

Liu, Y., Liu, L., Yan, Y., Feng, H., Ding, S., 2019. Analyzing Dynamic Change in Social Network Based on Distribution-Free Multivariate Process Control Method. *Computers, Materials & Continua* 60 (3), pp. 1123–1139.

Lowry, C. A., Woodall, W. H., Champ, C. W., Rigdon, S. E., 1992. A Multivariate Exponentially Weighted Moving Average Control Chart. *Technometrics* 34 (1), pp. 46–53.

Lu, C.-W., Reynolds Jr, M. R., 1999. Control Charts for Monitoring the Mean and Variance of Autocorrelated Processes. *Journal of Quality Technology* 31 (3), pp. 259–274.

Lu, C.-W., Reynolds Jr, M. R., 2001. Cusum Charts for Monitoring an Autocorrelated Process. *Journal of Quality Technology* 33 (3), pp. 316–334.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G., 2018. Learning under Concept Drift: A Review. *IEEE transactions on knowledge and data engineering* 31 (12), pp. 2346–2363.

Lütkepohl, H., 2005. New introduction to multiple time series analysis. Springer Science & Business Media.

Mahalanobis, P. C., 1936. On the Generalised Distance in Statistics. In: *Proceedings of the National Institute of Sciences of India*. pp. 49–55.

Malinovskaya, A., Killick, R., Leeming, K., Otto, P., 2023a. Statistical monitoring of European cross-border physical electricity flows using novel temporal edge network processes. *arXiv preprint: 2312.16357*.

Malinovskaya, A., Mozharovskyi, P., Otto, P., 2023b. Statistical process monitoring of artificial neural networks. *Technometrics*, pp. 1–14.

Malinovskaya, A., Otto, P., 2021. Online network monitoring. *Statistical Methods & Applications* 30 (5), pp. 1337–1364.

Malinovskaya, A., Otto, P., Peters, T., 2022. Statistical learning for change point and anomaly detection in graphs. In: *Artificial Intelligence, Big Data and Data Science in Statistics: Challenges and Solutions in Environmetrics, the Natural Sciences and Technology*. Springer, pp. 85–109.

Markou, M., Singh, S., 2003a. Novelty detection: a review–part 1: statistical approaches. *Signal Processing* 83 (12), pp. 2481–2497.

Markou, M., Singh, S., 2003b. Novelty detection: a review–part 2: neural network based approaches. *Signal Processing* 83 (12), pp. 2499–2521.

Mason, R. L., Tracy, N. D., Young, J. C., 1995. Decomposition of T^2 for Multivariate Control Chart Interpretation. *Journal of quality technology* 27 (2), pp. 99–108.

Masud, M. M., Gao, J., Khan, L., Han, J., Thuraisingham, B., 2009. Integrating Novel Class Detection with Classification for Concept-Drifting Data Streams. In: *Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 79–94.

McCulloh, I., Carley, K. M., 2011. Detecting Change in Longitudinal Social Networks. Tech. rep., Military Academy West Point NY Network Science Center (NSC).

McDermott, M. J., Dwaraknath, S. S., Persson, K. A., 2021. A graph-based network for predicting chemical reaction pathways in solid-state materials synthesis. *Nature communications* 12 (1).

Mejri, D., Limam, M., Weihs, C., 2017. Combination of Several Control Charts Based on Dynamic Ensemble Methods. *Mathematics and Statistics* 5 (3), pp. 117–129.

Mejri, D., Limam, M., Weihs, C., 2021. A new time adjusting control limits chart for concept drift detection. *IFAC Journal of Systems and Control* 17.

Miller, B. A., Arcolano, N., Bliss, N. T., 2013. Efficient anomaly detection in dynamic, attributed graphs: Emerging phenomena and big data. In: *2013 IEEE International Conference on Intelligence and Security Informatics*. pp. 179–184.

Montgomery, D. C., 2009. Introduction to Statistical Quality Control. John Wiley & Sons, Inc.

Montgomery, D. C., Mastrangelo, C. M., 1991. Some Statistical Process Control Methods for Autocorrelated Data. *Journal of Quality Technology* 23 (3), pp. 179–193.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M. M., 2017. Geometric deep learning on graphs and manifolds using mixture model CNNs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 5115–5124.

Moon, J., Kim, J., Shin, Y., Hwang, S., 2020. Confidence-Aware Learning for Deep Neural Networks. In: *Proceedings of the 37th International Conference on Machine Learning*. vol. 119. PMLR, pp. 7034–7044.

Morris, M., Handcock, M. S., Hunter, D. R., 2008. Specification of Exponential-Family Random Graph Models: Terms and Computational Aspects. *Journal of Statistical Software* 24 (4).

Mosler, K., 2013. Depth Statistics. *Robustness and Complex Data Structures: Festschrift in Honour of Ursula Gather*, pp. 17–34.

Mosler, K., Mozharovskyi, P., 2022. Choosing among notions of multivariate depth statistics. *Statistical Science* 37 (3), pp. 348–368.

Mozharovskyi, P., 2022. Anomaly detection using data depth: multivariate case. *arXiv preprint: 2210.02851*.

Murphy, B., 1987. Selecting Out of Control Variables with the T^2 Multivariate Quality Control Procedure. *Journal of the Royal Statistical Society Series D: The Statistician* 36 (5), pp. 571–581.

Nammous, M. K., Saeed, K., 2019. Natural language processing: speaker, language, and gender identification with LSTM. In: *Advanced Computing and Systems for Security*. Springer, pp. 143–156.

Nason, G. P., Wei, J. L., 2022. Quantifying the Economic Response to COVID-19 Mitigations and Death Rates Via Forecasting Purchasing Managers' Indices Using Generalised Network Autoregressive Models with Exogenous Variables. *Journal of the Royal Statistical Society Series A: Statistics in Society* 185 (4), pp. 1778–1792.

Ngai, H.-M., Zhang, J., 2001. Multivariate cumulative sum control charts based on projection pursuit. *Statistica Sinica*, pp. 747–766.

Ning, X., Wu, C., 2011. Improved design of quantile-based control charts. *Journal of the Chinese institute of industrial engineers* 28 (7), pp. 504–511.

Nishida, K., Yamauchi, K., 2007. Detecting concept drift using statistical testing. In: *Discovery Science*. vol. 4755. Springer, pp. 264–269.

Noorossana, R., Hosseini, S. S., Heydarzade, A., 2018. An overview of dynamic anomaly detection in social networks via control charts. *Quality and Reliability Engineering International* 34 (4), pp. 641–648.

O'Malley, A. J., Marsden, P. V., 2008. The Analysis of Social Networks. *Health Services and Outcomes Research Methodology* 8 (4), pp. 222–269.

O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. *arXiv preprint: 1511.08458*.

Otter, D. W., Medina, J. R., Kalita, J. K., 2020. A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems* 32 (2), pp. 604–624.

Page, E. S., 1954. Continuous inspection schemes. *Biometrika* 41 (1/2), pp. 100–115.

Pandolfo, G., Iorio, C., Staiano, M., Aria, M., Siciliano, R., 2021. Multivariate process control charts based on the L^p depth. *Applied Stochastic Models in Business and Industry* 37 (2), pp. 229–250.

Parekh, J., Mozharovskyi, P., d'Alché Buc, F., 2021. A framework to learn with interpretation. *Advances in Neural Information Processing Systems* 34, pp. 24273–24285.

Park, K., Jung, D., Kim, J. M., 2020. Control charts based on randomized quantile residuals. *Applied Stochastic Models in Business and Industry* 36 (4), pp. 716–729.

Parzen, E., 1962. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics* 33 (3), pp. 1065–1076.

Pearce, T., Brintrup, A., Zhu, J., 2021. Understanding Softmax Confidence and Uncertainty. *arXiv preprint: 2106.04972*.

Perdikis, T., Psarakis, S., 2019. A survey on multivariate adaptive control charts: Recent developments and extensions. *Quality and Reliability Engineering International* 35 (5), pp. 1342–1362.

Perry, M. B., 2020. An EWMA control chart for categorical processes with applications to social network monitoring. *Journal of Quality Technology* 52 (2), pp. 182–197.

Piano, L., Garcea, F., Gatteschi, V., Lamberti, F., Morra, L., 2022. Detecting Drift in Deep Learning: A Methodology Primer. *IT Professional* 24 (5), pp. 53–60.

Pidhorskyi, S., Almohsen, R., Doretto, G., 2018. Generative probabilistic novelty detection with adversarial autoencoders. *Advances in Neural Information Processing Systems* 31.

Pincombe, B., 2005. Anomaly detection in time series of graphs using arma processes. *Asor Bulletin* 24 (4).

Pokotylo, O., Mozharovskyi, P., Dyckerhoff, R., 2019. Depth and Depth-Based Classification with R Package ddalpha. *Journal of Statistical Software* 91 (5), pp. 1–46.

Porzio, G. C., Ragozini, G., 2008. Multivariate control charts from a data mining perspective. *Recent Advances in Data Mining of Enterprise Data: Algorithms and Applications* 6.

Priebe, C. E., Conroy, J. M., Marchette, D. J., Park, Y., 2005. Scan Statistics on Enron Graphs. *Computational & Mathematical Organization Theory* 11 (3), pp. 229–247.

Psarakis, S., 2011. The use of neural networks in statistical process control charts. *Quality and Reliability Engineering International* 27 (5), pp. 641–650.

Psarakis, S., 2015. Adaptive control charts: Recent developments and extensions. *Quality and Reliability Engineering International* 31 (7), pp. 1265–1280.

Qiu, P., 2014. Introduction to Statistical Process Control. CRC press.

Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C., Samatova, N. F., 2015. Anomaly detection in dynamic networks: a survey. *Wiley Interdisciplinary Reviews: Computational Statistics* 7 (3), pp. 223–247.

Roberts, S., Tarassenko, L., 1994. A probabilistic resource allocating network for novelty detection. *Neural Computation* 6 (2), pp. 270–284.

Robins, G., Pattison, P., 2001. Random graph models for temporal processes in social networks. *Journal of Mathematical Sociology* 25 (1), pp. 5–41.

Robins, G., Pattison, P., Kalish, Y., Lusher, D., 2007. An introduction to exponential random graph (p^*) models for social networks. *Social Networks* 29 (2), pp. 173–191.

Runger, G. C., Willemain, T. R., 1995. Model-based and model-free control of autocorrelated processes. *Journal of Quality Technology* 27 (4), pp. 283–292.

Sadinejad, S., Saghaei, A., Rajabi, F., 2020. Monitoring of Social Network and Change Detection by Applying Statistical Process: ERGM. *Journal of Optimization in Industrial Engineering* 13 (1), pp. 131–143.

Salmasnia, A., Mohabbati, M., Namdar, M., 2019. Change point detection in social networks using a multivariate exponentially weighted moving average chart. *Journal of Information Science*.

Sambale, H., Sinulis, A., 2020. Logarithmic Sobolev inequalities for finite spin systems and applications. *Bernoulli* 26 (3), pp. 1863 – 1890.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., Monfardini, G., 2008. The graph neural network model. *IEEE Transactions on Neural Networks* 20 (1), pp. 61–80.

Schmid, W., Schöne, A., 1997. Some properties of the EWMA control chart in the presence of autocorrelation. *Annals of Statistics* 25 (3), pp. 1277–1283.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R., 2001. Estimating the support of a high-dimensional distribution. *Neural Computation* 13 (7), pp. 1443–1471.

Schubert, E., Zimek, A., Kriegel, H.-P., 2014. Generalized outlier detection with flexible kernel density estimates. In: *Proceedings of the 2014 SIAM International Conference on Data Mining*. SIAM, pp. 542–550.

Schweinberger, M., 2011. Instability, sensitivity, and degeneracy of discrete exponential families. *Journal of the American Statistical Association* 106 (496), pp. 1361–1370.

Schweinberger, M., Krivitsky, P. N., Butts, C. T., Stewart, J., 2020. Exponential-Family Models of Random Graphs: Inference in Finite-, Super-, and Infinite Population Scenarios. *Statistical Science*.

Schweinberger, M., Stingo, F. C., Vitale, M. P., 2021. Special issue on statistical analysis of networks: Preface by the guest editors. *Statistical Methods & Applications* 30, pp. 1285–1288.

Schweitzer, F., Fagiolo, G., Sornette, D., Vega-Redondo, F., Vespignani, A., White, D. R., 2009. Economic networks: The new challenges. *Science* 325 (5939), pp. 422–425.

Shaghaghi, M., Saghaei, A., 2020. PCA likelihood ratio test approach for attributed social networks monitoring. *Communications in Statistics-Theory and Methods* 49 (12), pp. 2869–2886.

Sheu, S.-H., Lu, S.-L., 2009. Monitoring the mean of autocorrelated observations with one generally weighted moving average control chart. *Journal of Statistical Computation and Simulation* 79 (12), pp. 1393–1406.

Shi, C., Zhang, X., Sun, J., Wang, L., 2022. Remote sensing scene image classification based on self-compensating convolution neural network. *Remote Sensing* 14 (3).

Shrestha, A., Mahmood, A., 2019. Review of deep learning algorithms and architectures. *IEEE Access* 7, pp. 53040–53065.

Smith, L. N., 2018. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. *arXiv preprint: 1803.09820*.

Snijders, T. A. B., Pattison, P. E., Robins, G. L., Handcock, M. S., 2006. New Specifications for Exponential Random Graph Models. *Sociological Methodology* 36 (1), pp. 99–153.

Sparks, R., Wilson, J. D., 2019. Monitoring communication outbreaks among an unknown team of actors in dynamic networks. *Journal of Quality Technology* 51 (4), pp. 353–374.

Stevens, N. T., Wilson, J. D., Driscoll, A. R., McCulloh, I., Michailidis, G., Paris, C., Parker, P., Paynabar, K., Perry, M. B., Reisi-Gahrooei, M., Sparks, R., 2021a. Research in network monitoring: Connections with SPM and new directions. *Quality Engineering* 33 (4), pp. 736–748.

Stevens, N. T., Wilson, J. D., Driscoll, A. R., McCulloh, I., Michailidis, G., Paris, C., Paynabar, K., Perry, M. B., Reisi-Gahrooei, M., Sengupta, S., Sparks, R., 2021b. Foundations of network monitoring: Definitions and applications. *Quality Engineering* 33 (4), pp. 719–730.

Stoumbos, Z. G., Jones, L. A., Woodall, W. H., Reynolds, M. R., 2001. On nonparametric multivariate control charts based on data depth. In: *Frontiers in Statistical Quality Control 6*. Springer, pp. 207–227.

Stoumbos, Z. G., Reynolds Jr, M. R., Ryan, T. P., Woodall, W. H., 2000. The state of statistical process control as we proceed into the 21st century. *Journal of the American Statistical Association* 95 (451), pp. 992–998.

Struyf, A. J., Rousseeuw, P. J., 1999. Halfspace depth and regression depth characterize the empirical distribution. *Journal of Multivariate Analysis* 69 (1), pp. 135–153.

Sun, Y., Ming, Y., Zhu, X., Li, Y., 2022. Out-of-distribution detection with deep nearest neighbors. In: *International Conference on Machine Learning*. PMLR, pp. 20827–20840.

Thottan, M., Ji, C., 2003. Anomaly detection in IP networks. *IEEE Transactions on signal processing* 51 (8), pp. 2191–2204.

Tukey, J. W., 1975. Mathematics and the picturing of data. *Proceedings of the International Congress of Mathematicians, Vancouver* 2, pp. 523–531.

Vakayil, A., Joseph, V. R., 2022. Data twinning. *Statistical Analysis and Data Mining: The ASA Data Science Journal*.

Van Duijn, M. A., Gile, K. J., Handcock, M. S., 2009. A framework for the comparison of maximum pseudo-likelihood and maximum likelihood estimation of exponential family random graph models. *Social networks* 31 (1), pp. 52–62.

Vencálek, O., 2017. Depth-based classification for multivariate data. *Austrian Journal of Statistics* 46 (3-4), pp. 117–128.

Villa-Pérez, M. E., Alvarez-Carmona, M. A., Loyola-Gonzalez, O., Medina-Pérez, M. A., Velazco-Rossell, J. C., Choo, K.-K. R., 2021. Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. *Knowledge-Based Systems* 218.

Vining, G., 2009. Technical Advice: Phase I and Phase II Control Charts. *Quality Engineering* 21 (4), pp. 478–479.

Voorhees, E. M., Harman, D., 2000. Overview of the sixth text retrieval conference (TREC-6). *Information Processing & Management* 36 (1), pp. 3–35.

Wang, H., Xia, Y., 2008. Sliced regression for dimension reduction. *Journal of the American Statistical Association* 103 (482), pp. 811–821.

Wang, X., Jin, B., Du, Y., Cui, P., Tan, Y., Yang, Y., 2021. One-class graph neural networks for anomaly detection in attributed networks. *Neural computing and applications* 33, pp. 12073–12085.

Wang, X., Wang, Z., Shao, W., Jia, C., Li, X., 2019. Explaining concept drift of deep learning models. In: *International Symposium on Cyberspace Safety and Security*. Springer, pp. 524–534.

Ward, M. D., Stovel, K., Sacks, A., 2011. Network analysis and political science. *Annual Review of Political Science* 14, pp. 245–264.

Wasserman, S., Pattison, P., 1996. Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p^*. *Psychometrika* 61 (3), pp. 401–425.

Weese, M., Martinez, W., Megahed, F. M., Jones-Farmer, L. A., 2016. Statistical learning methods applied to process monitoring: An overview and perspective. *Journal of Quality Technology* 48 (1), pp. 4–24.

Wilson, J. D., Stevens, N. T., Woodall, W. H., 2019. Modeling and detecting change in temporal networks via the degree corrected stochastic block model. *Quality and Reliability Engineering International* 35 (5), pp. 1363–1378.

Woodall, W. H., Montgomery, D. C., 2014. Some current directions in the theory and application of statistical process monitoring. *Journal of Quality Technology* 46 (1), pp. 78–94.

Woodall, W. H., Ncube, M. M., 1985. Multivariate CUSUM Quality-Control Procedures. *Technometrics* 27 (3), pp. 285–292.

Woodall, W. H., Zhao, M. J., Paynabar, K., Sparks, R., Wilson, J. D., 2017. An overview and perspective on social network monitoring. *IISE Transactions* 49 (3), pp. 354–365.

Wu, H.-M., 2008. Kernel sliced inverse regression with applications to classification. *Journal of Computational and Graphical Statistics* 17 (3), pp. 590–610.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S. Y., 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32 (1), pp. 4–24.

Yan, T., Leng, C., Zhu, J., 2016. Asymptotics in directed exponential random graph models with an increasing bi-degree sequence. *The Annals of Statistics* 44 (1), pp. 31–57.

Yan, T., Xu, J., 2013. A central limit theorem in the β-model for undirected random graphs with a diverging number of vertices. *Biometrika* 100 (2), pp. 519–524.

Yan, X., Shalizi, C., Jensen, J. E., Krzakala, F., Moore, C., Zdeborová, L., Zhang, P., Zhu, Y., 2014. Model selection for degree-corrected block models. *Journal of Statistical Mechanics: Theory and Experiment* 2014 (5).

Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., Peng, W., Wang, H., Chen, G., Li, B., Sun, Y., 2022a. OpenOOD: Benchmarking generalized out-of-distribution detection. *Advances in Neural Information Processing Systems* 35, pp. 32598–32611.

Yang, J., Zhou, K., Li, Y., Liu, Z., 2022b. Generalized out-of-distribution detection: A survey. *arXiv preprint: 2110.11334v2*.

Yeung, D.-Y., Chow, C., 2002. Parzen-window network intrusion detectors. In: *International Conference on Pattern Recognition*. vol. 4. IEEE, pp. 385–388.

Yeung, D.-Y., Ding, Y., 2003. Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition* 36 (1), pp. 229–243.

Yin, Z., Shen, Y., 2018. On the dimensionality of word embedding. *Advances in Neural Information Processing Systems* 31.

Yu, F., Wang, D., Shelhamer, E., Darrell, T., 2018. Deep layer aggregation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 2403–2412.

Zan, T., Liu, Z., Su, Z., Wang, M., Gao, X., Chen, D., 2020. Statistical process control with intelligence based on the deep learning model. *Applied Sciences* 10 (1), pp. 308.

Zhang, H., 2009. Study on the TOPN Abnormal Detection Based on the NetFlow Data Set. *Computer and Information Science* 2 (3), pp. 103–108.

Zhang, K., Bui, A. T., Apley, D. W., 2023. Concept Drift Monitoring and Diagnostics of Supervised Learning Models via Score Vectors. *Technometrics* 65 (2), pp. 137–149.

Zhang, N. F., 1997. Detection capability of residual control chart for stationary process data. *Journal of Applied Statistics* 24 (4), pp. 475–492.

Zhang, S., Tong, H., Xu, J., Maciejewski, R., 2019. Graph convolutional networks: A comprehensive review. *Computational Social Networks* 6 (1), pp. 11.

Zhao, M. J., Driscoll, A. R., Sengupta, S., Stevens, N. T., Fricker Jr, R. D., Woodall, W. H., 2018. The effect of temporal aggregation level in social network monitoring. *PLOS one* 13 (12).

Zheng, L., Li, Z., Li, J., Li, Z., Gao, J., 2019. AddGraph: Anomaly Detection in Dynamic Graph Using Attention-based Temporal GCN. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. *AI Open* 1, pp. 57–81.

Zhu, X., Pan, R., Li, G., Liu, Y., Wang, H., 2017. Network Vector Autoregression. *The Annals of Statistics* 45 (3), pp. 1096–1123.

Žliobaitė, I., Pechenizkiy, M., Gama, J., 2016. An overview of concept drift applications. *Big Data Analysis: New Algorithms for a New Society*, pp. 91–114.

Zou, C., Wang, Z., Tsung, F., 2012. A spatial rank-based multivariate EWMA control chart. *Naval Research Logistics (NRL)* 59 (2), pp. 91–110.

Zuo, Y., Serfling, R., 2000. General Notions of Statistical Depth Function. *Annals of Statistics* 28 (2), pp. 461–482.

Acknowledgements

First and foremost I express my profound gratitude to my supervisor and mentor, *Doktorvater* Prof. Dr. *Philipp Otto*. You taught me that exceptions might indeed become the rule if you find those who believe in you and support you in all situations. Your expertise and vision have propelled me to a place I once could only dream of, thank you very much for it.

With equally high appreciation I thank Prof. Dr.-Ing. habil. *Monika Sester* and apl. Prof. Dr.-Ing. *Claus Brenner* for their unwavering guidance, support, and encouragement throughout my doctoral journey. Feeling like a statistical "exotic" at the Institute of Cartography and Geoinformatics (IKG), I am particularly grateful for the welcoming atmosphere I found here.

This nurturing environment is remarkably owed to the friendly relationship of both *current and former colleagues at IKG* to me. The intellectual and casual conversations shared every day made my office and our library a home for me. Thank you for the personal support in IT and geoinformatics – I valued it greatly.

Reflecting on the milestones of the past four years, I acknowledge explicitly the invaluable help and guidance of my main collaborators, Prof. *Rebecca Killick* and Prof. Dr. *Pavlo Mozharovskyi*. Visiting you was the pinnacle of my doctoral journey, an experience for which I am endlessly grateful.

My professional growth owes much to the opportunities provided by the *Graduate Academy*, where I found enriching support, professional training, and financial aid during my research stay in Lancaster and the final phase of my doctoral study.

Completing this thesis would not have been possible without the encouragement, faith and boundless optimism of those close to me – *family, friends and significant individuals* who have left an indelible mark on this stage of my life. To all who accompanied me on this path and ensured its completion, I extend my heartfelt gratitude.

My acknowledgements would not have been complete without thanking *Hannover*, the city whose scenery has been a constant backdrop behind my office window. Its green spaces and waterways provide respite from work-related thoughts, a sanctuary I deeply appreciate.

Anna Malinovskaya

Curriculum Vitae

Personal Data

Date and Place of Birth: 23.06.1996, Kustanai, Kasachstan

Education

05.2024 11.2019	**Doctoral Candidate** **Leibniz University Hannover, Germany** Institute of Cartography and Geoinformatics Thesis Topic: "Statistical Process Monitoring of Networks" Supervisor: Prof. Dr. Philipp Otto
05.2019 10.2015	**Study of International Business Administration (Bachelor of Science)** **European University Viadrina, Frankfurt (Oder), Germany** Specialisation in Statistics, Finance and Accounting Grade: 1.2 (with honours)
12.2017 09.2017	**Semester Abroad** **Manchester Metropolitan University, United Kingdom** Specialisation in Finance and Accounting

Scientific Career

11.2023 07.2019	**Research Associate** **Leibniz University Hannover, Germany** Institute of Cartography and Geoinformatics
Today 03.2022	**Adjunct Lecturer** **Teaching of Statistics III Course** Leibniz University of Applied Sciences, Germany
02.2023 01.2023	**Research Collaborator** **STOR-i Centre for Doctoral Training** Lancaster University, United Kingdom
02.2022	**Research Collaborator** **Team " Signal, Statistique et Apprentissage"** Information Processing and Communications Laboratory, Télécom Paris, France

Professional Experience

Today 12.2023	**Data Scientist** **Nala Earth GmbH** Remote/Berlin, Germany
02.2019 09.2018	**Internship** **Department Risk Instruments** Daimler Financial Services, Stuttgart, Germany
05.2017 02.2017	**Internship** **Department Audit & Assurance** Deloitte GmbH, Berlin, Germany

Publications

2023 Malinovskaya, A., Mozharovskyi, P., Otto, P. **Statistical Process Monitoring of Artificial Neural Networks**. *Technometrics*. https://doi.org/10.1080/00401706.2023.2239886

Malinovskaya, A., Killick, R., Leeming, K., Otto, P. **Statistical monitoring of European cross-border physical electricity flows using novel temporal edge network processes**. *arXiv preprint*. https://doi.org/10.48550/arXiv.2312.16357

2022 Malinovskaya, A., Otto, P., Peters, T. **Statistical Learning for Change Point and Anomaly Detection in Graphs**. In: Steland, A., Tsui, KL. (eds) *Artificial Intelligence, Big Data and Data Science in Statistics*, pp. 85–109. Springer, Cham. https://doi.org/10.1007/978-3-031-07155-3_4

2021 Malinovskaya, A., & Otto, P. **Online network monitoring**. *Statistical Methods & Applications*, 30(5), pp. 1337–1364. https://doi.org/10.1007/s10260-021-00589-z

Conference Talks

2023 **Statistical Week**, TU Dortmund University, "Statistical monitoring of electricity-related networks"

18. Doktorand:innentreffen der Stochastik, Ruprecht Karl University of Heidelberg, "Statistical monitoring of flow networks"

2022 **Statistical Week**, University of Münster, "Comparison of data depths for monitoring deep learning applications"

The 6th conference of the Deutsche Arbeitsgemeinschaft Statistik, Online, "Statistical monitoring of deep learning models"

2021 **Royal Statistical Society International Conference**, Manchester, "Online network monitoring: Combining temporal exponential random graph models and control charts"

Women in Statistics and Data Science Conference (the American Statistical Association), Online, "When do we need a new neural network? Searching for nonstationarity using data depth"

Statistical Week, Online, "Online Network Monitoring"

The 6th Canadian Conference in Applied Statistics, Online, "Statistical learning for change point and anomaly detection in graphs"

Awards

2023 **University Award (Hochschulpreis)**, Leibniz University Hannover, Germany