Ensuring Time Series Data Privacy via Complex Networks

Jaime Amaro Soares do Vale
Master in Information Security
Computer Science Department
Faculty of Sciences of the University of Porto
2024

Ensuring Time Series Data Privacy via Complex Networks

Jaime Amaro Soares do Vale

Dissertation carried out as part of the Information Security Masters

Computer Science Department

2024

Supervisor
Vanessa Alexandra Freitas da Silva, Assistant Professor, Faculty of Science, University of Porto

Co-supervisor
Fernando Manuel Augusto da Silva, Professor, Faculty of Science, University of Porto

Acknowledgements

First and foremost, I would like to extend my sincere and deepest appreciation to my thesis supervisors, Professor Vanessa Silva and Professor Fernando Silva, for their invaluable guidance, and to Professor Maria Eduarda Silva for her invaluable support. I am deeply grateful for the insightful feedback and continuous encouragement I received throughout the development of this work. Your expertise was fundamental and invaluable to my success.

To my family, especially to my parents and my partner, your endless patience and faith in me have been the foundation of my perseverance and success. I would not be able to achieve this milestone without your constant support, perseverance and guidance.

I would also like take the opportunity to extend my gratitude to my closest friends who, since the beginning, believed and encouraged me to pursue my goals.

Thank you all.

Resumo

A privacidade dos dados de séries temporais coloca desafios significativos em vários domínios, como as finanças, a saúde, e o setor energético, entre outros, onde a informação sensível, quando partilhada, deve ser protegida, enquanto mantém a utilidade dos dados. Os métodos tradicionais lutam frequentemente para equilibrar a privacidade e a utilidade, e por isso abordagens inovadoras são necessárias, como é o caso da geração de dados sintéticos que retêm as propriedades essenciais do conjunto de dados original sem comprometer informação sensível e privada.

Nesta dissertação, exploramos se o mapeamento de dados de séries temporais no domínio de redes complexas pode contribuir para a privacidade, preservando ao memso tempo as principais propriedades e utilidade dos dados. Especificamente, investigamos se os dados sintéticos de séries temporais gerados a partir de mapeamentos de e para redes complexas exibem propriedades semelhantes aos dados originais, garantindo a sua privacidade.

Propomos uma abordagem baseada num método de mapeamento de séries temporais duplo para gerar dados sintéticos de séries temporais, primeiro transformando os dados originais em redes utilizando um método de mapeamento baseado em quantis e, então, gerando dados sintéticos através do método de mapeamento inverso. Para validar a abordagem proposta, realizámos uma avaliação empírica abrangente utilizando uma variedade de modelos estatísticos de séries temporais, tanto lineares como não lineares. Analisamos as principais característica dos dados de séries temporais originais mantidas pelos dados de séries temporais sintéticos, analisando tando medidas estatísticas das séries temporais como medidas topológicas de redes complexas. Comparámos também, qualitativamente, a metodologia apresentada com a framework TimeGAN de estado da arte, visualizando as distribuições dos conjuntos de dados sintéticos e originais. Por fim, aplicámos métricas de privacidade amplamente utilizadas em privacidade de séries temporais para avaliar a privacidade e a utilidade dos dados sintéticos gerados pelo método de mapeamento inverso baseado em amostras de quantis.

Este estudo destaca o potencial da ciência de redes para enfrentar os desafios duplos da utilidade e da privacidade dos dados na análise de séries temporais, fornecendo uma solução promissora para a geração de dados sintéticos seguros.

Palavras-chave: Privacidade de Séries Temporais, Geração de Dados Sintéticos, Redes Complexas, Mapeamentos de Séries Temporais

Abstract

Privacy of time series data poses significant challenges in various fields such as finance, healthcare, and the energy sector, among others, where confidential information, when shared, must be protected while maintaining the utility of the data. Traditional methods often struggle to balance privacy and utility, requiring innovative approaches, such as the generation of synthetic data that retains the main properties of the original dataset without compromising sensitive and private information.

In this dissertation, we explore whether mapping time series data into the complex networks domain can contribute to privacy while preserving the main properties and utility of the data. Specifically, we investigate whether synthetic time series data generated via mappings to and from complex networks exhibit properties similar to the original data, ensuring data privacy.

We propose an approach based on a dual time series mapping method to generate synthetic time series data, first transforming the original data into networks using a quantile-based mapping method and then generating synthetic data through the inverse mapping method. To validate the proposed approach, we perform a comprehensive empirical evaluation using a variety of time series statistical models, both linear and nonlinear. We analyze the main characteristics of the original time series data maintained by the synthetic time series data by analyzing both statistical measures of the time series and topological measures of complex networks. We also compare, qualitatively, the proposed methodology with the state of the art TimeGAN framework by visualizing the distributions of synthetic and original data sets. Finally, we apply privacy metrics widely used in time series privacy to evaluate the privacy and utility of synthetic data generated by the quantile sample-based inverse mapping method.

This study highlights the potential of network science to address the dual challenges of data utility and privacy in time series analysis, providing a promising solution for generating secure synthetic data.

Keywords: Time Series Privacy, Synthetic Data Generation, Complex Networks, Time Series Mappings

Contents

List of Tables

List of Figures

Acronyms

ACF Autocorrelation Function

AR Autoregressive

ARFIMA Autoregressive Fractionally Integrated Moving Average

ARIMA Autoregressive Integrated Moving Average

CE Conditional Entropy

DP Differential Privacy

dpGAN Differentially Private Generative Adversarial Network

GAN Generative Adversarial Network

GARCH Generalized Autoregressive Conditional Heteroskedasticity

HE Homomorphic Encryption

HMM Hidden Markov Models

HVG Horizontal Visibility Graph

INAR Integer-Valued Autoregressive

InvQG Inverse Quantile Graph

IoT Internet of Things

MI Mutual Information

NAGC Next-Generation Access Control

NIST National Institute of Standards and Technology

NMI Normalized Mutual Information

NN Neural Network

NVG Natural Visibility Graph

OASIS Organization for the Advancement of Structured Information Standards

OCE Offline Conditional Entropy

PCA Principal Component Analysis

QG Quantile Graph

SAML Security Assertion Markup Language

SETAR Self-Exciting Threshold Autoregressive

t-SNE t-Distributed Stochastic Neighbor Embedding

TimeGAN Time-series Generative Adversarial Network

WN White Noise

XACML eXtensible Access Control Markup Language

Chapter 1

Introduction

A time series is a sequence of observations collected in chronological order [1] and a central piece in today's data ecosystem. Collected at large scale from industrial systems [2], smart home equipment [3] to individual wearables [4], they have become a key source for the success of multidisciplinary applications such as climate and sustainability studies, economic analysis, monitoring of transportation and health [5, 6]. Hence, temporal data has garnered considerable attention from researchers, companies, and the general public, serving various research purposes [7] — this significantly increased interest in sharing this data among organizations and individuals. However, time series data can be considered sensitive as it may contain identifiable information about individuals or organizations. A simple example is data in business settings, which can enclose information both about production lines, allowing the inference of internal strategies, and about clients, allowing their identification [8]. Another simple example is individual data, sourced from smart or medical devices and Internet of Things (IoT) devices installed in homes, which can enclose information about particular locations [9, 10], health status [11] or behavioral patterns [12]. Figure 1.1 shows an illustrative example of a power trace of an individual's day's activity where some sensitive patterns can be identified.

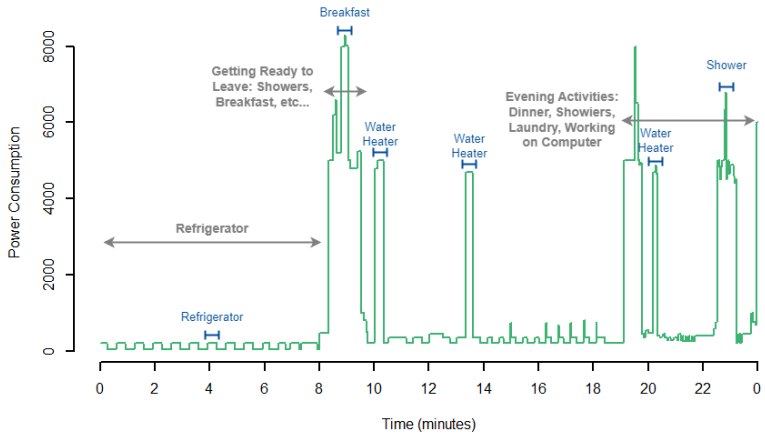


Figure 1.1: Example of a power trace from the day's activity. *Source:* Adapted from [12].

Processing and extracting useful information from time series, while maintaining the privacy and utility of the data is a current challenge. Anonymization, aggregation and encryption are traditional techniques [9, 13, 14] for privacy-preserving time series data. Anonymization and aggregation are widely adopted strategies that modify sensitive information but at a cost to the data utility [8, 15]. Homomorphic encryption allows extracting time series values using encryption methods, but the operations are limited [2]. A recent approach gaining importance is generating synthetic time series from the original data [16]. This involves creating fictitious datasets with similar dynamical and statistical properties to the original dataset, without revealing its sensitive information. Data perturbation methods, such as adding noise [17] and innovative approaches based on Generative Adversarial Network (GAN) techniques [16, 18, 19] are some of the most used techniques for this purpose. However, the high noise percentage reduces the data utility and GANs are naturally non-interpretable, making it difficult to explain and ascertain the reliability of the resulting synthetic data.

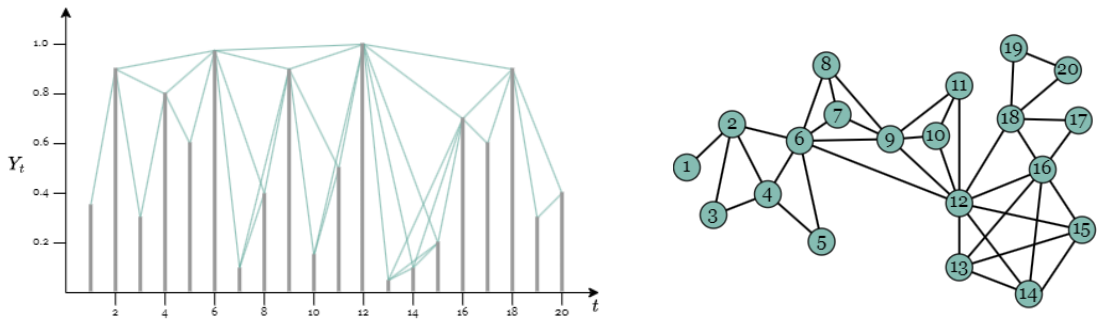
Analysing time series via complex networks involves mapping the data into one (or more) complex networks (or graphs). The time series is then represented by a set of nodes and edges: the nodes correspond to timestamps, symbols representing data characteristics, statistics or even individual time series while the edges, representing connections between the nodes, are established according to criteria such as visibility, transition or distance [20].

The resulting networks capture time series dynamics and introduce a novel perspective for interpreting original data, providing new insights. For example, random time series are mapped into random networks and periodic series into regular networks [21]. Representing time series as networks may unveil hidden patterns or data dependencies overlooked by traditional analyses.

1.1 Motivation

Given the issues mentioned concerning preserving privacy when dealing with time series data and the alternative approach to represent and analyse time series, a question that naturally arises is: can we harness time series analysis via complex networks for privacy-preserving in time series data?

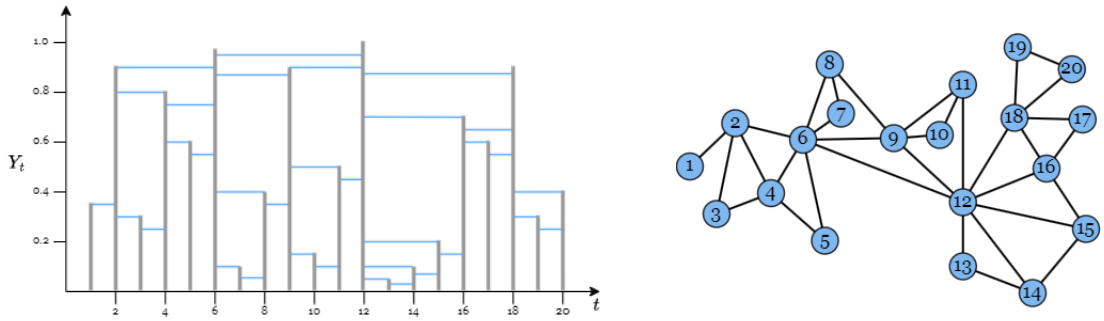
We start by introducing the most common time series mappings into networks. The first mappings, proposed by [21], are based on visibility concepts and transform a time series into a complex network by first representing each observation in the time series as a vertical bar with a height equal to the observation value. Observations are then placed sequentially within a landscape based on their timestamps. Each timestamp is represented as a node in the resulting graph. Nodes are then interlinked if bar tops have visibility to other bar tops [20]. Figure 1.2 shows a visual representation of the Natural Visibility Graph (NVG) mapping.



(a) Toy time series (b) Natural Visibility Graph

Figure 1.2: Illustration of Natural Visibility Graph mapping method. *Source:* Adapted from [20].

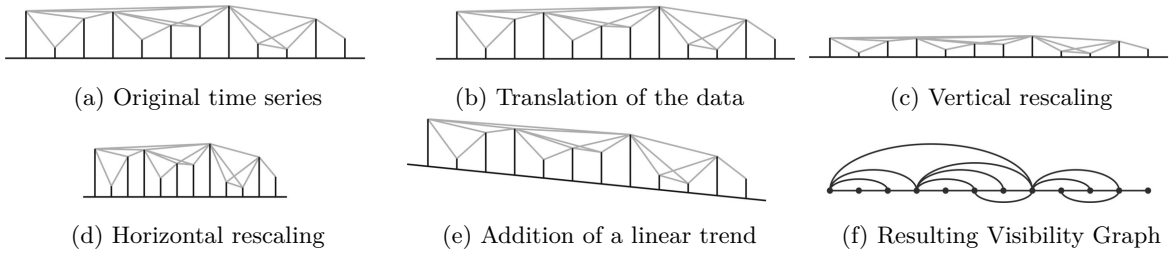
Another visibility mapping method is the Horizontal Visibility Graph (HVG). In HVG the process is similar to NVG with the exception of how links are created. In this case, nodes are interlinked if the bars can be connected to other bars using a straight and horizontal line without any obstructions. Compared to NVG, this approach improves the computational efficiency of the visibility mapping process [22] and results in a visibility graph that is always a subgraph of the corresponding NVG, with the same set of nodes but with a set of edges which is a subset of the NVG edge set. Figure 1.3 illustrates the HVG mapping.



(a) Toy time series (b) Horizontal Visibility Graph

Figure 1.3: Illustration of Horizontal Visibility Graph mapping method. *Source:* Adapted from [20].

Visibility mappings capture local and global properties of the time series [20]. However, reconstructing the original time series from the corresponding complex network is an infeasible task since different time series can be mapped into the same visibility graph [21, 22], as illustrated in Figure 1.4. These properties suggest that mapping a time series into visibility graph and analysing the data in the complex network space may be an approach to privacy preserving in time series analysis.



(a) Original time series (b) Translation of the data (c) Vertical rescaling

(d) Horizontal rescaling (e) Addition of a linear trend (f) Resulting Visibility Graph

Figure 1.4: Stability of Natural Visibility Graph under multiple affine transformations of time series data. *Source:* Extracted from [21]

Alternative mappings are based on transition probabilities techniques which analyse and aggregate information from transition patterns (dynamics) present in the original data and transform the resulting information into complex networks. One specific example is the quantile-based process that maps a time series into a directed weighted network. The mapping method starts by calculating the quantiles from the data distribution and then maps each quantile into a node in the resulting network. Two nodes are linked if the transition from one quantile to another is observed in sequential data points within the time series. To each link is then associated a direction and a weight, which is the probability of transitioning from a quantile to another in consecutive data points [23]. This method not only facilitates the extraction of insightful information from the quantile-based graph but also allows for the reverse mapping. This provides a novel approach to obtain new synthetic time series data based on the original time series as we analyzed in this work.

By enabling the synthesis of new time series, this method has the potential to contribute to data privacy field by allowing the use of synthetic versions instead of the original data. To the best of our knowledge such use has never been analysed and will be our main focus for the remainder of this thesis.

1.2 Goals and Contributions

This work aims to answer the following two main research questions:

1. By mapping a time series into the complex networks domain, can we ensure privacy yet preserving the main properties of the data and their usefulness?
2. Do synthetic time series data generated using mappings into and from complex networks have properties similar to the original data, guaranteeing simultaneously their privacy and their usefulness?

The main contributions of the work described in this thesis are the following:

- **Time series mappings as a means to protect sensible data.** Synthetic data generation methods offer robust privacy protection with high data utility. In this work we propose to use the methods for mapping time series into networks as a (non-parametric) technique to generate synthetic time series data. We first transform the original data into a graph (complex network domain) using the quantile-based mapping method. We then compute synthetic time series data from the quantile graphs generated using a reverse mapping method proposed by [23] (thus returning to the time series domain). As far as we know, no other existing work uses these time series mapping methods together for synthetic data generation with a view to data security. An advantage to highlight of this method is the possibility of generating many synthetic time series (statistically similar), given the probabilistic nature of the method.
- **A comprehensive empirical assessment of the synthetic time series.** Creating synthetic data that preserves the statistical and structural properties of the original data is crucial. To assess our proposal for generating synthetic time series, we resort to data simulated from several well-established time series models. The time series simulated from each model stand for the *original* data from which a synthetic counterpart is created. We then perform an empirical analysis of the synthetic time series both in the time series domain via statistical properties and in the complex networks domain via topological characteristics.
  - We selected a set of eight different interpretable and well-established statistics in the field of time series analysis and conducted a detailed and comprehensive analysis of these statistics on both the original time series and the synthetic time series. We analyzed which statistical properties are preserved in the synthetic data generation and analyze the correlation of the corresponding statistics via Principal Component Analysis (PCA). As far as we know, no other work provides a similar comprehensive analysis.
  - Topological features of complex networks refer to the properties and structural characteristics that describe the organization and connectivity within the network. Theses features are essential for understanding the dynamics and behavior of networks. To analyze the utility of synthetic time series data generated by reverse mapping within the scope of network science, we selected a set of fifteen topological features from the literature and applied them to complex networks generated from the original data (simulated data) and to complex networks generated from the corresponding synthetic data. We analyzed using PCA which topological properties are maintained in the synthetic data when compared to those of their respective original counterparts.

- **Evaluation of proposed methodology against the state of the art.** We evaluated the quantile-based reverse mapping method for generating synthetic time series data against a GAN-based strategy from the literature, Time-series Generative Adversarial Network (TimeGAN). We adapted the available TimeGAN framework for univariate time series datasets and qualitatively compared our approach by conducting t-Distributed Stochastic Neighbor Embedding (t-SNE) and PCA analyses to visualize how well the distributions of the generated synthetic data resemble those of the original data.
- **Evaluation of privacy metrics.** Generating synthetic data not only must have high utility, but also protect sensitive and private data. We studied and compiled some widely used privacy metrics applicable to time series data. We applied these privacy evaluation metrics to the synthetic datasets generated by the reverse quantile-based mapping and analyzed the utility and privacy of the generated data.

1.3 Organization

We organized this thesis into six major chapters. Below we provided a brief description of each one of them.

Chapter 1 - Introduction. Provides a brief introduction to the research area, the main motivation, goals and contributions, and the organization of the thesis.

Chapter 2 - Time Series Privacy Strategies. Presents a brief overview of the state of the art for traditional data-privacy strategies, with special focus in time series.

Chapter 3 - Mapping Time Series into Quantile Graphs. Provides a detailed explanation of the QG mapping method including a review of an existing implementation that converts a time series into a quantile graph. It also presents our implementation for the reverse procedure, which converts a quantile graph back into a times series.

Chapter 4 - Empirical Evaluation. Conducts a comprehensive empirical evaluation of the quantile-based mapping method's synthesizing capabilities. This involves synthesizing various time series models using QG mappings and measuring data utility.

Chapter 5 - Privacy Metrics and Application. Introduces and details statistical privacy metrics, illustrating the expected behaviour for each metric. Here we apply these metrics to the QG-generated data to effectively evaluate the privacy degree offered by the mapping method. Additionally we compare the synthesizing capabilities of QG to TimeGAN.

Chapter 6 - Conclusions. Summarizes all our findings and conclusions and gives for future work.

Chapter 2

Time Series Privacy Strategies

Privacy can be defined as the degree of uncertainty an attacker faces when attempting to disclosure original data [24]. Although time series data, defined as sequences of data points ordered in time, is naturally void of explicit sensitive information, such data includes properties such as amplitude, average, peaks, trend or periodicity that may include sensitive information, as is the case of biometric, geographic and behavioural characteristics. An attacker can use these elements to identify end-users or extract confidential personal or business information [6], for example. Therefore, time series data should be regarded as sensitive or confidential and be subject to the same level of protection as sensitive categorical data.

Preserving the privacy of time series data involves protecting organized data that evolves over time. This effort requires methodologies that enable data sharing while maintaining its statistical properties and temporal patterns, all while ensuring the confidentiality and integrity of sensitive or identifiable information [25].

Protecting sensitive or confidential information has long been a focus of study in information security and, more recently, in data science. Some well-known and widely used traditional privacy-preserving data methods include limiting and controlling access, encryption, anonymization and differential privacy methods [8]. Recently, the proliferation of data resulting from our current digital era has led to increased interest in this topic and new methodologies have begun to emerge. Using synthetic data, derived from the original data, is a recent methodology that has emerged as a novel approach to protect and preserve data privacy [5, 19, 26].

Another crucial aspect in time series analysis is data utility, which evaluates how effectively time series data successfully allow forecasting and analytical tasks. High data utility indicates that the data is relevant, accurate, and properly formatted for analysis, thereby enhancing the reliability of predictive models [27]. This involves ensuring the data is well-prepared, addressing factors such as stationarity, seasonality, and trends [28]. Balancing utility and privacy is essential when defining privacy strategies, as it involves assessing the impact of various methodologies while ensuring data usability and confidentiality [29]. As seen in Figure 2.1, the ideal solution would be to balance between the two.

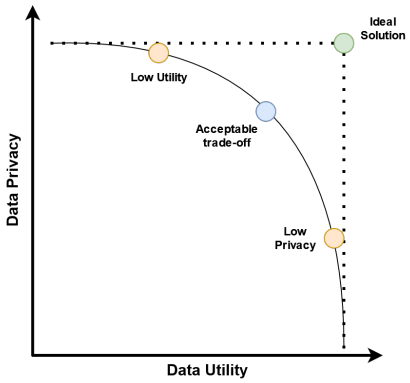


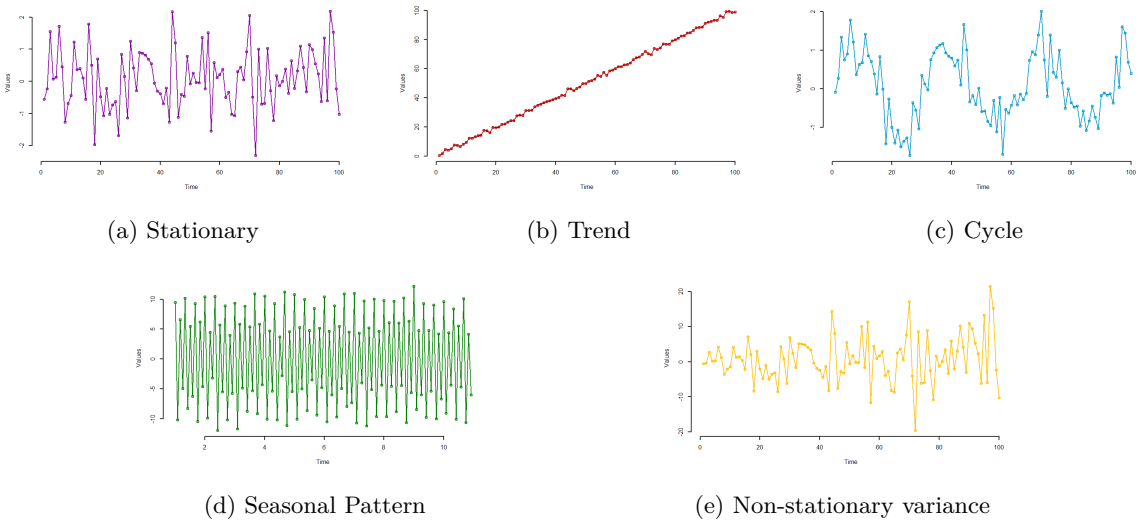
Figure 2.1: Graphical representation of the trade-off between privacy level and data utility. *Source:* Adapted from [29].

In this chapter, we start by briefly describing the concepts and necessary terminology of time series analysis, and we will briefly describe the main methods used to ensure time series data privacy and confidentiality such as access control, encryption, anonymization and synthetic data generation.

2.1 Time Series

A time series $\boldsymbol{Y} = (Y_1, \ldots, Y_T)$ is a sequence of data points typically consisting of successive measurements $\{Y_t\}_t$ taken indexed at a time t. This sequence can represent a stochastic process, where the points are random variables indexed by time, or a deterministic process that can be fully described by a mathematical equation. Time series are typically defined by an existing serial dependence between observations where data points in the series are related to its predecessors and successors. This sets time series apart from independent and identically distributed (i.i.d.) data, where each data point is independent of others. Due to this dependence, conventional statistical methods often do not apply and specialized techniques for time series analysis are needed [1].

A crucial concept in time series analysis is stationarity. A time series is stationary if its statistical properties, such as mean and variance, remain constant over time. Stationarity is vital because it allows for the meaningful computation of statistics like mean, variance, and autocorrelation from the data. There are different types of stationarity. Strict stationarity requires that the joint distribution of any collection of points remains the same regardless of shifts in time. Weak stationarity, on the other hand, imposes conditions only on the first two moments (mean and variance) of the series [20]. This property is essential in time series analysis, since most of conventional statistical models require time series to be stationary. However, most time series of interest (mainly, real-world series) are inherently non-stationary. This data exhibits properties such as trend, cycles, seasonal patterns, or other properties that make the data non-stationary. Figure 2.2 illustrated five toy time series with this statistical properties.



(a) Stationary (b) Trend (c) Cycle

(d) Seasonal Pattern (e) Non-stationary variance

Figure 2.2: A illustration of a stationary toy time series in (a), a toy time series with linear trend in (b), a toy time series with cycles in (c), a toy time series with a seasonal pattern in (d), and a non-stationary toy time series with constant mean and non-constant variance in (e).

2.2 Access Control

Access control is the cornerstone of information security and a widely adopted approach that safeguards information through the use of access control policies [30]. These policies define data access users and configure the terms of how these can process information. It reduces the risk of information disclosure by limiting the amount of data access to a required minimal and by allowing only trustworthy users to access the information [9].

In order to standardize the process of configuration and enforcing access control policies, the Organization for the Advancement of Structured Information Standards (OASIS) introduced the eXtensible Access Control Markup Language (XACML) and the Security Assertion Markup Language (SAML). XACML is a standard XML-based access control policy language that allows the definition of attribute-based policies and it is used to manage access control between multiple systems [31]. SAML offers a standard markup language and protocol to exchange attribute-based authorization and authentication information. It is designed to facilitate the recognition of user credentials and permissions across multiple systems [32]. Together, these two frameworks enable access control and identity verification across diverse platforms using a unified and standard approach.

However, in IoT systems, XACML and SAML face significant challenges. The ever-growing volume of information and the dynamic nature of IoT environments call for a more flexible and scalable security solution than XACML and SAML. In IoT, we observe high volatility in the changes of nodes, users and data structures, making it hard to predict who will use the system or how the data will be organized. This unpredictability makes it difficult to define

security policies. Moreover, IoT systems have now extended beyond the traditional single-closed environment and became part of a larger network involving multiple organizations comprised of several different information systems [30, 33]. For this reason, and aiming at more dynamic and volatile environments, the National Institute of Standards and Technology (NIST) proposed the Next-Generation Access Control (NAGC) which, unlike XACML and SAML, uses a relationship-based architecture rather than rule-based system. This allows for a more dynamic and flexible policy enforcement. NAGC manages accesses via configurations of relations like assignments, associations, prohibitions, and obligations, which can be modified in real-time to adapt to changes in the environment or user status. Making it an effective tool in more complex systems where security requirements are subject to more frequent changes, as is the case of IoT environments [33, 34].

Nonetheless, despite recent efforts, traditional challenges still arise when applying access control measures in time series data and IoT. For instance, defining fine-grained access controls leads to complex policy specifications and significant computational overhead. Moreover, when applied in semi-trusted or untrusted environments, applying access control measures can potentially expose sensitive information [9].

2.3 Cryptography

Cryptography is a critical field within information security and a standard approach for preserving data privacy [6, 9]. It involves converting plain data into an encrypted format, rendering it incomprehensible to observers. Cryptography can protect information while it exists at rest or in transit. It ensures user authentication, data confidentiality, data integrity, non-repudiation and access control [35, 36].

In traditional cryptography, data must be deciphered before any computations can be performed. This can potentially expose it to privacy threats [35–37]. Furthermore, cryptographic operations are computationally intensive, which poses a challenge for sensor-based systems typically equipped with a limited amount of processing power. In addition, it introduces the complexity of managing cryptographic keys. Consequently, traditional cryptography is not the optimal method for public data sharing [6, 8]. Faced with this problem, a novel cryptography-based approach called Homomorphic Encryption (HE) was proposed. It makes use of a special kind of encryption scheme that allows to extract insightful information directly from encrypted data without the need to decipher it in advance [9, 37, 38]. Figure 2.3 generally illustrates a simple example of client-server HE scenario.

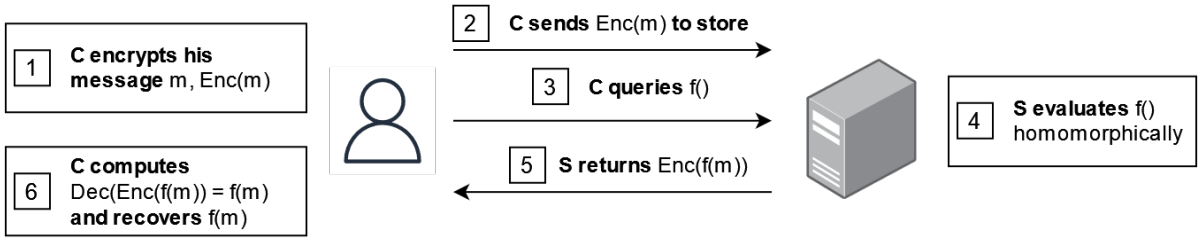


Figure 2.3: A simple illustration of client-server Homomorphic Encryption scenario. Variable C refers to client and S to server. *Source:* Adapted from [37].

HE schemes can be organized into three main categories as follows [37]:

- **Partially Homomorphic Encryption**: supports a single operation that can be executed for an unlimited number of times on a ciphertext.
- **Somewhat Homomorphic Encryption**: supports a few operations that can be executed for a limited amount of times per operation.
- **Fully Homomorphic Encryption**: supports an unlimited number of operations and executions.

Recently the use of HE for IoT systems and temporal data streams was proposed in [2]. Using a symmetric homomorphic encryption-based access control technique, the approach supports aggregated statistical queries such as *sum*, *mean*, *max*, *min*, and other operations such as linear regression or other analytic functions involving aggregated data. In addition, it allows for data storage and fine-grained control access for operations such as limiting the date range for the aggregation calculation.

Despite its potential, HE is computationally intensive and suffers from the ciphertext expansion problem, which refers to an unavoidable communication overhead caused by the large nature in size of ciphertext [39]. Improving querying efficiency and mitigating the ciphertext expansion problem is an open area of study in HE with promising advances being made [37, 39]. Furthermore, HE operations are performed via centralized server, which adds another layer of complexity to data sharing applications.

Other cryptographic solutions exist and are numerous [9]. However, for the sole purpose of this thesis and its goal of openly sharing private time series data, such options are considered to be out of scope.

2.4 Data Perturbation

Data perturbation techniques are the one of most widely used mechanisms in data privacy. These techniques consist in modifying original data observations in an attempt of masking sensitive information. Classic data perturbation methods inject noise from Gaussian or Laplace distributions and can vary in strategy. Such strategies can be *additive*, *multiplicative*, *geometric*, *nonlinear* and *transformation-based*. In the later, data is first converted into a lower-dimensional feature space where noise then is added [6, 8, 38].

More recently, new methodologies have been proposed under *Differential Privacy (DP)*, a term representing another privacy ensuring data-perturbation technique. Initially built for querying databases [40], for a method to be declared differential private it needs to be insensitive to the presence of data from a particular data-source [9]. Put simply, differential privacy ensures that modifying any individual record will have a negligible effect on the statistical outcome [6].

Both differential private and classical perturbation techniques inject noise into data with a cost to data utility [6, 10], and most work in this area revolves around optimizing the total amount of noise injection to strike a balance between data privacy and utility [38].

In time series, most differential privacy schemes ensure privacy by protecting either at a single timestamp (*event-level*), a single data-source (*user-level*) or per window (*w-event-level*), considering all timestamps equally significant. However, in a novel approach called *Landmark Privacy* [10], it is possible to differentiate significant timestamps (*landmarks*) from regular timestamps and thus decrease the amount of noise injected and increase data utility. Figure 2.4 exemplifies landmarks in geographical time series data.

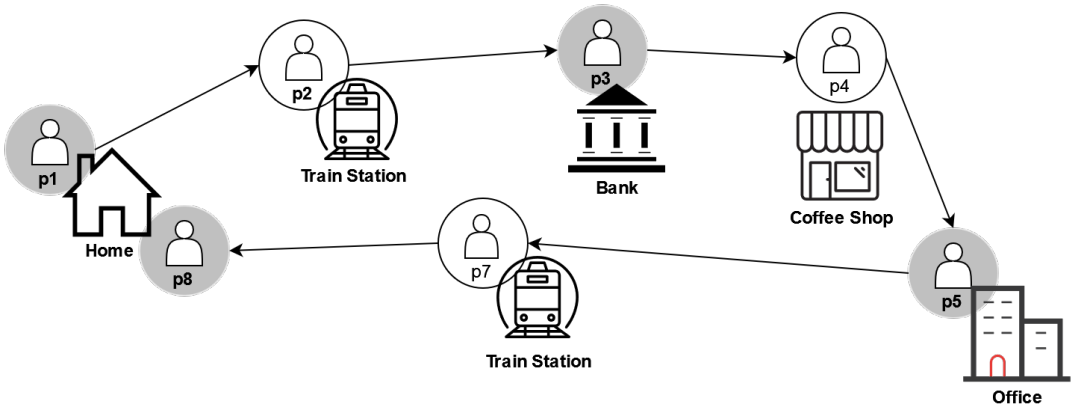


Figure 2.4: Illustration of Landmark Privacy on geographical time series data (with landmarks highlighted in gray). *Source:* Adapted from [10].

However, data-perturbation methods might not ensure total privacy. Initial work has shown that there are still chances of re-identification using distance analysis [41] and record linkage [42].

2.5 Synthetic Data: Generative Adversarial Network

Since their inception in 2014, Generative Adversarial Networks (GANs) have primarily been applied to computer vision. However, recent studies have demonstrated promising results in their applicability to time series data. In this domain, GANs are generating high-quality, diversified and differentially private time series data. GANs can not only augment smaller datasets by synthesizing new, unseen data, but can also denoise and impute missing or corrupted data [18, 43]. Figure 2.5 depicts the typical generative adversarial network architectural design.

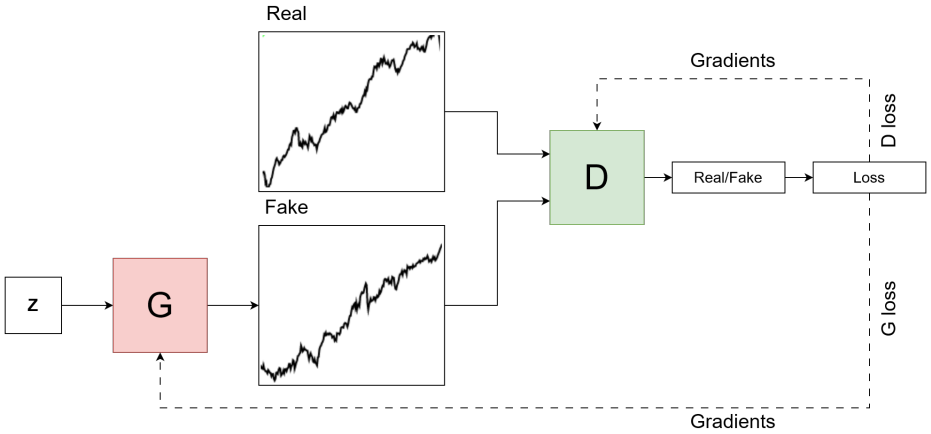


Figure 2.5: Diagram representing a typical generative adversarial network architecture design for time series analysis. *Source:* Adapted from [18].

A GAN typically consists of two main Neural Network (NN): a generator (represented by **G** in Figure 2.5) and a discriminator (represented by **D**). The generator aims to synthesize data that is indistinguishable from the original, the discriminator evaluates if the generated data is real or synthetic. The training occurs jointly and uses a zero-sum non-cooperative game to optimize the adversarial loss. In this game, the generator tries to maximize the failure rate of the discriminator, whereas the discriminator tries to minimize it. This allows the generator to improve over time, resulting in the generation of more realistic data. The input (**Z**) is often a random noise vector, serves as the input for the generator to create new data instances, ensuring variability and robustness in the generated outputs [18, 19, 43].

When applied to time series, traditional GANs struggle to preserve temporal dynamics. To overcome this limitation, the Time-series Generative Adversarial Network (TimeGAN) enhanced the traditional GAN technique by introducing a stepwise supervised loss function and an embedding network. The supervised loss function uses original data as supervisor and encourages the model to capture temporal dynamics such as conditional distributions within the data. Meanwhile, the embedding network, transforms high-dimensional data into a lower-dimensional latent space, reducing the complexity of the model's learning space. This promotes parameter efficiency (due to the lower complexity) and allows the generator to more effectively capture the temporal dynamics within the data [19].

Similar TimeGAN alternatives for improved synthesis of temporal data exist, such as *Doppel-GANger* [44] and *RCGAN* [45], however, some GAN implementations have a special focus not only in improving the synthesizing process of temporal data, but also in improving data privacy.

GAN techniques can still disclosure sensitive information. The adversarial procedure and the high complexity of deep neural networks both encourage a distribution concentrated around the training samples. By repeatedly sampling from that distribution, there is a considerable chance of recovering the original training samples. It also has been shown that an active inference attack model can reconstruct the training samples from the generated ones. To solve these security concerns the Differentially Private Generative Adversarial Network (dpGAN) has been proposed as a technique that is capable of generating high-quality synthetic data while ensuring differential privacy by carefully adding noise in the adversarial learning process [26].

However, GAN models come with several disadvantages. In one hand GANs can be unstable. This instability leads to training difficulties and creates problems such as non-convergence, diminishing/vanishing gradients and mode collapse. A non-converging model fails to stabilize and continuously oscillates, causing it to diverge. Diminishing gradients occur when the discriminator becomes overly effective, preventing the generator from learning. Mode collapse happens when the generator collapses, leading to the generation of uniform samples with little to non-existent variety [18].

Chapter 3

Mapping Time Series into Quantile Graphs

Time series analysis via network science is an approach that involves mapping time series data into one or more complex networks, represented by a set of nodes and edges. Nodes typically represent specific time points or temporal symbols/patterns and edges capture the temporal dependencies. Mapping algorithms for time series typically utilize concepts such as visibility, probability transition, proximity, time series models, and statistical principles [20]. Network science methodologies have been widely and effectively applied to time series analysis across various domains, addressing tasks such as forecasting [46], classification [47], clustering [48], and outlier detection [49]. Additionally, the topological properties of networks have proven to be useful and powerful in capturing the main characteristics of time series data [50].

Transition networks are constructed from continuous time series data by essentially performing two steps. First, a symbol encoding is performed. This entails mapping the continuous time series into a set of states, which will then be mapped into nodes in the network. Secondly, transition probabilities between defined states are calculated. These probabilities are computed as the relative frequency of states sequences, leading to the establishment of directed and weighted edges in the transition network [20].

Symbol encoding can be achieved using various methods, including time range or time series support partitioning, among others. Time range partitioning works at the temporal level and slices data at regular intervals, or meaningful points in time. To each temporal slice an unique symbol/state is assigned. Support partitioning involves analysing the amplitude or magnitude of data observations at distinct intervals and assigning them to symbol/state.

In this chapter, we begin by presenting the fundamental concepts and terminology of network science that are relevant to understanding the remainder of this dissertation. Next, we provide a detailed description of the time series mapping methods in/from complex networks used in this work. In particular, we describe the mapping method *Quantile Graph (QG)* proposed in [23] and which transforms time series data into a transition probability network. Then we describe the

reverse mapping method of the QG mapping [23], which we call *Inverse Quantile Graph (InvQG)*, that transforms the original time series data back from the networks created by the QG method. We also describe our implementation of quantile-based mapping methods to generate synthetic time series data from time series mappings.

3.1 Complex Networks

Complex networks are structures composed of a large number of interconnected elements, exhibiting non-trivial patterns of connection that frequently occur in natural and artificial systems [51]. Network science is an emerging interdisciplinary area that investigates these types of complex structures to understand the dynamics and topology underlying diverse real-world systems, such as social networks, communication networks, biological networks, among others [52].

Graph theory, a branch of discrete mathematics, provides the essential theoretical basis for the analysis and understanding of complex networks. A *graph* is the mathematical formulation of a complex network. It consists of a set of *nodes* (or *vertices*) connected by *edges* (or *links*). Graph analysis allows us to investigate fundamental structural properties of complex networks and understand how their topology influences the functionality of the systems they represent.

Graph A graph $G = (V, E)$ is a mathematical structure consisting of a set of nodes V and a set of edges E that connect pairs of nodes. If two nodes $v_i \in V$ and $v_j \in V$ are connected by an edges $(v_i, v_j) \in E$, the nodes are called neighbours or adjacent. Graphs can be classified as *directed* or *undirected*. In the former, the edges have an associated direction, while in the latter, the connections between the nodes have no direction. Graphs can also be classified as *weighted* and *unweighted*, depending on whether or not there is a weight (or cost), $w_{i,j}$, associated with the edges. Some graphs may also have *self-loops*, an edge that connects a node to itself.

Adjacency Matrix The adjacency matrix $\boldsymbol{A}$ of a graph with N nodes is an $N \times N$ matrix where each entry $A_{i,j}$ indicates the presence (or absence) of an edge between nodes v_i and v_j. For unweighted graphs, $\boldsymbol{A}$ is binary (i.e., $A_{i,j} = 1$ when $(v_i, v_j) \in E$ and is 0 otherwise), and for undirected graphs, $\boldsymbol{A}$ is symmetric (i.e., $A_{i,j} = A_{j,i}$).

Figure 3.1 illustrates a simple representation of a toy directed graph and the corresponding asymmetric adjacency matrix. Note that the illustrated toy graph have one self-loop (i.e., $(2, 2) \in E$).

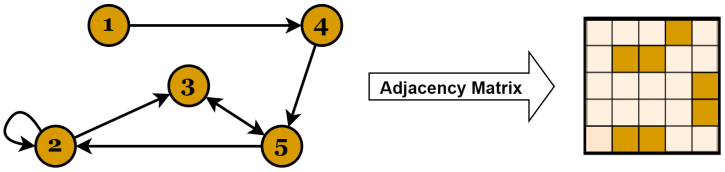


Figure 3.1: A representation of directed graph and the corresponding adjacency matrix.

To analyze the structure and behavior of complex networks, an arsenal of topological measures exists [53]. These measures allow to quantify important characteristics of graphs, such as centrality, connectivity, information transmission, communities, and network robustness. Below, we describe some of the main topological measures used in network science and analysed in this work.

Average Weighted Degree ($\bar{k}$) Is the arithmetic mean of the weighted degrees of all nodes in the graph, indicating the degree of connectivity between nodes. The weighted degree of a node is the sum of the weights of its connections (edges) in the graph, it is a node centrality measure. It is calculated as follows:

$$\bar{k} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} A_{i,j}$$

Average Path Length ($\bar{d}$) This measures the mean of the shortest paths (distances) between all nodes in the graph, reflecting the efficiency of information flow. The average path length is calculated as follows:

$$\bar{d} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} d_{i,j},$$

where $d_{i,j}$ is the length of the shortest paths (number of edges) between nodes v_i and v_j in the graph.

Clustering Coefficient (C) This measures the degree to which nodes tend to cluster together, indicating the likelihood of nodes forming groups. In this work, it is defined as the ratio between three times the number of triangles (set of three nodes where each pair of nodes is connected by an edge) in the graph and the number of triplets (set of three nodes, where at least two of them are connected by an edge) connected of nodes.

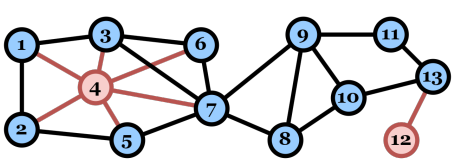
Number of Communities (S) This indicates the number of subgraphs within the network, measuring groups of nodes that are densely connected. In this work, an as in [50], we use the Walktrap community finding algorithm that calculates densely connected subgraphs via random walks, such that short random walks tend to stay in the same community.

Modularity (Q) This measures the degree of division of a network into communities and the quality of connections between these community structures, as follows:

$$Q = \frac{1}{2|E|} \sum_{i,j} \left(A_{i,j} - \frac{k_i k_j}{2|E|} \right) \delta\left(c_i, c_j\right),$$

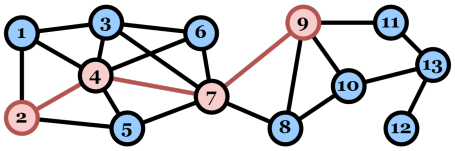
where $|E|$ is the number of edges in the graph, k_i and k_j are the degree of nodes v_i and v_j, c_i and c_j the communities of v_i and v_j, and $\delta\left(c_i, c_j\right)$ is the Kronecker delta function which is 1 if v_i and v_j are in the same community (i.e., $c_i = c_j$), and 0 otherwise.

Figure 3.2 illustrates the concepts of the topological measures mentioned above.



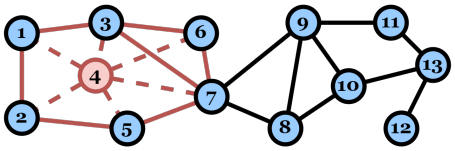
$k_4 = 6; k_{12} = 1; \bar{k} = 3.385$

(a) Node centrality



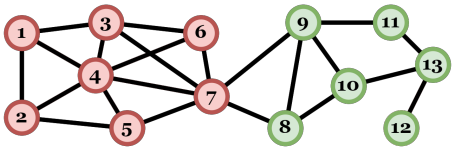
$d_{2,9} = 3; \bar{d} = 2.474$

(b) Shortest path



$C_4 = 0.467; C = 0.483$

(c) Clustering coefficient



$S = 2; Q = 0.372$

(d) Communities

Figure 3.2: A simple illustration of topological measure concepts in graphs. (a) illustrates node centrality (the node degree k_4 and k_{12} and the average degree $\bar{k}$), (b) a shortest path between two nodes ($d_{2,9}$) in the graph and the average path length $\bar{d}$, (c) the clustering coefficient (of the node C_4 and the whole graph C), and (d) illustrates two communities (S) in the graph and the corresponding modularity (Q).

3.2 Quantile Graph Algorithm

Campanharo et al. introduced in [23] an innovative transition-based method to map a time series into a complex network which relies on partitioning the time series support and defining transition probabilities, converting the time series data into a quantile-based complex network, called *Quantile Graph (QG)*.

Mapping an univariate time series $\boldsymbol{Y} \in \mathcal{T}$ into a directed weighted graph $G \in \mathcal{G}$ is defined by the function $\mathcal{M}_{\mathrm{QT}}$. This function splits the support of a time series into $\mathcal{Q}$ sample quantiles, $\{q_1, q_2, \ldots, q_{\mathcal{Q}}\}$. Each quantile, q_i will be mapped into a node $v_i \in V$ in the resulting graph. Two nodes v_i and v_j are linked via directed and weighted edge, $(v_i, v_j, w_{i,j}) \in E$, based on the transition frequency between the corresponding successive quantiles, q_i and q_j, in the time series. That is, $w_{i,j}$ computes the number of times a time series data point Y_t, belonging to the quantile q_i, is followed by an time series data point Y_{t+1}, belonging to the quantile q_j. The adjacency matrix of $\mathcal{G}$ is then normalized into a Markov transition matrix, $\boldsymbol{W}$, where $\sum_{j=1}^{\mathcal{Q}} w_{i,j} = 1$, for each $i = 1, \ldots, \mathcal{Q}$, where the weight $w_{i,j}$ represents the transition probability between the quantile ranges. Figure 3.3 illustrates this mapping method.

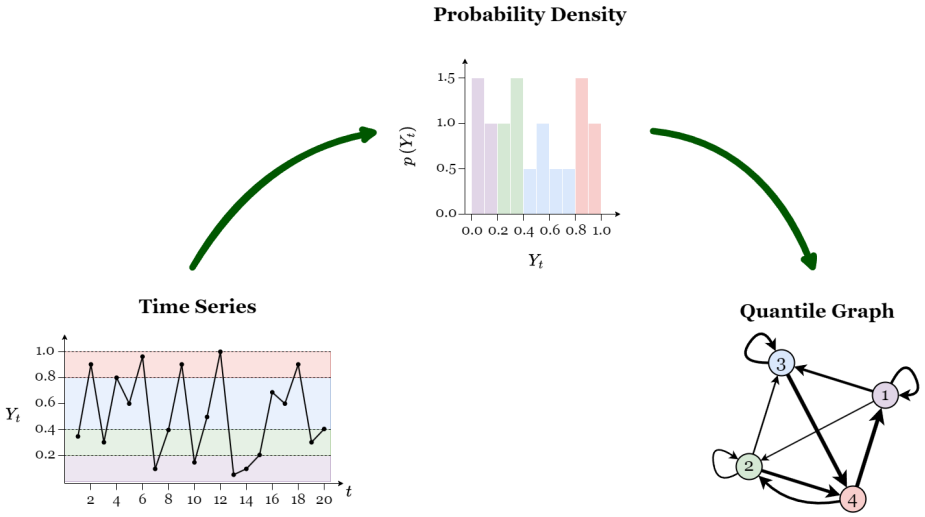


Figure 3.3: Illustration of Quantile Graph mapping method with a toy time series. *Source:* Adapted from [20, 23].

The created quantile graph has as many nodes as the number of defined quantiles, (i.e., $|V| = \mathcal{Q}$), which is usually much smaller than the length of the time series, T, reducing this dimensionality ($\mathcal{Q} \ll T$) [20]. The directed and weighted edge, $(v_i, v_j, w_{i,j})$, will have a larger weight if transitions from q_i to q_j are more often observed within time series data. Note that if there are many quantiles, i.e., $\mathcal{Q}$ is large, the resulting quantile graph might end up being disconnected, presenting isolated nodes (nodes that are not connected to any other node). On the other hand, when $\mathcal{Q}$ is small, the quantile graph might exhibit a notable loss of information, indicated by high weights in self-loops, for example.

The Algorithm 1 describes the procedure of mapping univariate time series data, $\boldsymbol{Y}$, into a quantile graph G. The source code used in this work to compute this QG mapping procedure M_{QT} was implemented by Silva et al. in [50]. The function `QG` enable the mapping of a input time series $\boldsymbol{Y}$ into a quantile graph G given by the output Markov transition matrix. The Algorithm 2 describes the used parameters to create G and obtain the sample quantiles of $\boldsymbol{Y}$.

Algorithm 1 Quantile Graph

Input: time series $\boldsymbol{Y}$ and number of quantiles $\mathcal{Q}$

Output: Markov transition matrix $\boldsymbol{W}$ of the resulting Quantile Graph and the sample quantiles *quantiles*

quantile calculates the range of quantiles from data values of a time series

index_of finds the quantile of a given value

procedure QG($\boldsymbol{Y}, \mathcal{Q}$)
  $\boldsymbol{W} \leftarrow$ Array($\mathcal{Q}, \mathcal{Q}$)
  $\boldsymbol{W} \leftarrow 0$
  $quantiles \leftarrow$ Array($\mathcal{Q}$)
  $q \leftarrow$ Array($\mathcal{Q}$)
  for $i \leftarrow 1$ to $\mathcal{Q}$ **do**
    $quantiles[i] \leftarrow \frac{i}{\mathcal{Q}}$
  end for
  $q \leftarrow$ quantile($\boldsymbol{Y}, quantiles$)
  for $i \leftarrow 1$ to size($\boldsymbol{Y}$) $- 1$ **do**
    $from\_v \leftarrow$ index_of($\boldsymbol{Y}[i], q$)
    $to\_v \leftarrow$ index_of($\boldsymbol{Y}[i+1], q$)
    $\boldsymbol{W}[from\_v][to\_v] \leftarrow \boldsymbol{W}[from\_v][to\_v] + 1$
  end for
  for $i \leftarrow 1$ to $\mathcal{Q}$ **do**
    $total \leftarrow 0$
    for $j \leftarrow 1$ to $\mathcal{Q}$ **do**
      $total \leftarrow total + W[i][j]$
    end for
    for $j \leftarrow 1$ to $\mathcal{Q}$ **do**
      if $\boldsymbol{W}[i][j] \neq 0$ **then**
        $\boldsymbol{W}[i][j] \leftarrow \boldsymbol{W}[i][j]/total$
      end if
    end for
  end for
  return $\boldsymbol{W}, quantiles$
end procedure

Algorithm 2 $M_{\mathrm{QT}} : \mathcal{T} \rightarrow \mathcal{G}$

graph_from_adjacency_matrix creates a graph structure from an adjacency matrix

$T \leftarrow$ size($\boldsymbol{Y}$)
$\mathcal{Q} \leftarrow \sqrt{T}$
$\boldsymbol{W}, quantiles \leftarrow$ QG($\boldsymbol{Y}, \mathcal{Q}$)
$G \leftarrow$ graph_from_adjacency_matrix($\boldsymbol{W}$, directed=*true*, weighted=*true*)

3.3 Inverse Quantile Graph Algorithm

Campanharo et al. in [23] additionally introduced a second mapping function, M_{QT}^{-1}, which is the reverse of M_{QT}, and allows to generate a new time series $\boldsymbol{Y} \in \mathcal{T}$ from an existing quantile graph $G \in \mathcal{G}$.

M_{QT}^{-1} uses the Markov transition matrix $\boldsymbol{W}$ associated to the adjacency matrix of a QG (G) in the trying process of re-reconstructing of the original time series ($\boldsymbol{Y}$). In this work we use and adapt the reverse process of the QG method to generate synthetic time series, statistically similar to the original series. We call the method *Inverse Quantile Graph* (*InvQG*), and Figure 3.4 illustrates this reverse mapping method that reconvert a quantile graph into a new time series similar in statistical terms to the original time series based on the adjacency matrix of the quantile graph.

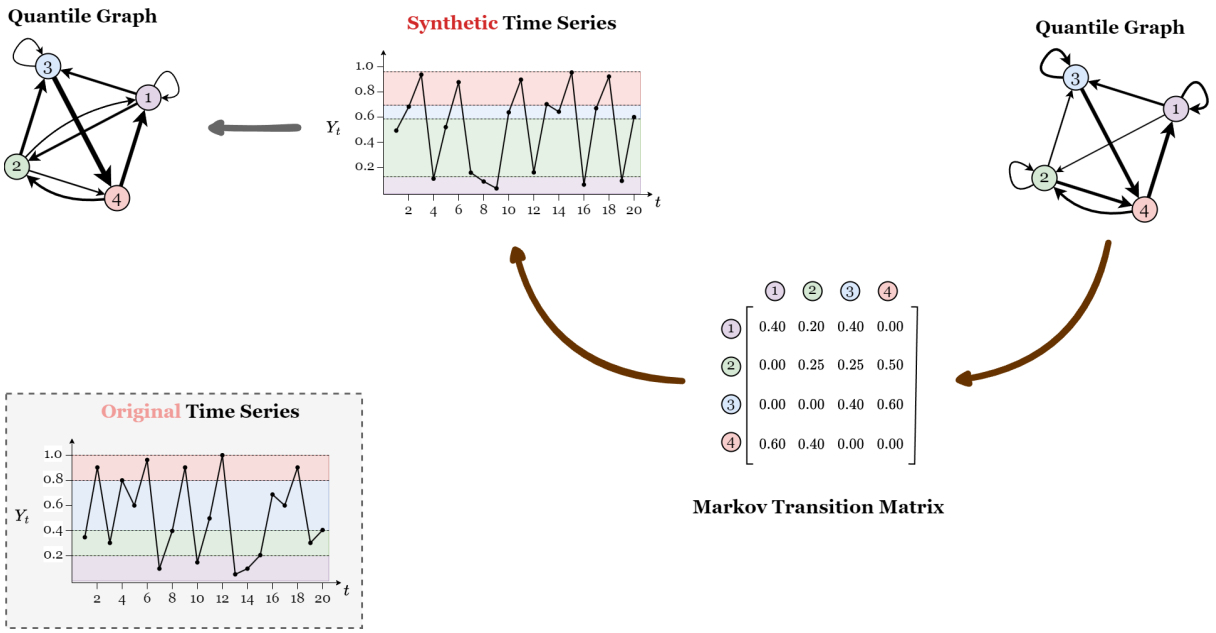


Figure 3.4: Illustration of Inverse Quantile Graph mapping method. The quantile graph resulting from the synthetic time series is also presented. *Source:* Adapted from [23].

Overall, the proposed method starts by randomly selecting the first quantile q_i and chooses a random number corresponding to the range of that quantile, then repeatedly moving from one node v_i to another v_j with a certain probability $w_{i,j}$ given by the transition matrix $\boldsymbol{W}$, again choosing a random number within the corresponding quantile range. The method is repeated until we obtain a new time series of the same length as the original.

In more detail, InvQG mapping involves the following steps (see Algorithm 3):

Step 1: Get the adjacency matrix of the time series graph generated by QG mapping, which is a Markov transition matrix (line 2).

Step 2: Get the sample quantiles of the time series graph generated by QG mapping (line 2).

Step 3: Randomly choose from the adjacency matrix the initial sample quantile, with non-zero probability, for the first value of the time series to be generated (line 5).

Step 4: Until obtain a time series of size T, do (for-loop in line 6):

Step 4.1: Get the range of values from the selected quantile sample (lines 7 and 8).

Step 4.2: Assign a value uniformly at random from the range between the lower and upper quantiles (line 9).

Step 4.3: Select the next sample quantile (to the next value in the time series to be generated), selecting the next node based on the non-zero transition probabilities of the adjacency matrix (line 10).

Step 5: Return the synthetic time series data (line 12).

Algorithm 3 Inverse Quantile Graph

Input: time series $\boldsymbol{Y}$ and number of quantiles $\mathcal{Q}$
Output: synthetic time series $\boldsymbol{X}$
init_sample select randomly sample initial quantile with non-zero probability
runif assign an uniform random value between quantile range
select_quantil_from_prob_matrix select next node based on transition probabilities

```
1:  procedure InvQG(Y, Q)
2:      W, quantiles ← QG(Y, Q)
3:      T ← size(Y)
4:      X ← Array(T)
5:      ni ← init_sample(W)
6:      for t ← 1 to T do
7:          min_range ← quantiles[ni]
8:          max_range ← quantiles[ni + 1]
9:          X[t] ← runif(min_range, max_range)
10:         ni ← select_quantil_from_prob_matrix(W[ni], Q)
11:     end for
12:     return X
13: end procedure
```

Note that the generated time series is not the same as the original but has the same statistical properties (as we will show in the next chapter), consequently the quantile graph corresponding to the generated time series may not be identical to that of the original data. Figure 3.4 shows that the generated synthetic time series is not the same and therefore can preserve sensitive information of individuals or organizations. However, the generated synthetic time series have essential properties that can be used for research purposes, for example, and can be safely shared with others. This is what we explore and analyze in the following chapters of this dissertation.

An important advantage of this synthetic time series generation method is the possibility of generating several different synthetic time series, but with properties and dynamics similar to the original data. This is possible given the probabilistic nature of Algorithm 3, both in the selection of the next sample quantile q_i to generate the value in time t of the time series (given by the Markov transition matrix), and by the random choice of the value of Y_t, which is a randomly assigned value between the minimum and maximum limits of the selected quantile. Such advantage not only allow the sharing of time series data securely (as we will analyze in the following chapters), but also allow us to increase the set of time series data that can improve the *analytics* performed on the data.

We implemented our own version of the InvQG mapping method. Out implementation starts by extracting all information regarding nodes and edges from the quantile graph G generated by QG mapping method from a time series. Then we convert this information into a Markov matrix $\boldsymbol{W}$. The matrix $\boldsymbol{W}$ and original quantiles information is then used as a parameter in Algorithm 3, which is responsible for generating the synthetic time series corresponding to the original time series.

Chapter 4

Empirical Evaluation

In this chapter, we evaluate the Inverse Quantile Graph (InvQG) mapping method ability to create high-utility synthetic data by analyzing how effectively the intrinsic properties of the original time series data can be preserved. For this purpose, in this chapter, we use a large and diverse set of time series data simulated from well-established statistical models. And we analyze whether the properties of the original data are preserved in the synthetic data generated (through the analysis of their properties) both in the domain of time series analysis (using statistical measures) and in the domain of complex networks (using topological network measures).

We start by describing the simulated time series dataset used in our empirical analysis and describing the methodology developed within the scope of this work. We detail the process of generating the time series statistical metrics using the *tsfeatures* package [54]. These metrics will produce insight information into fundamental dynamics and characteristics for each time series model. We then present our findings, including plotting Principal Component Analysis (PCA) charts, that demonstrate that quantile-based mapping method successfully generates high-utility synthetic data and is capable of mirroring key original data characteristics.

Subsequently, we convert original and respective synthetic time series data into complex networks and use topological metrics using the *NetF* package [50] to extract topological properties for each network. These properties will allow to understand similarity between original and synthetic time series in the complex networks domain, as well as showing that by using synthetic networks would allow to extract insightful information. We present these metrics and show that in similarity to time series data, resulting synthetic complex networks also maintain key global topological properties. This further validate the method's effectiveness in generating synthetic time series data.

Finally, we also compare the proposed InvQG method to generating synthetic time series data against to state of the art methods, in particular, the Time-series Generative Adversarial Network (TimeGAN) method. We perform a qualitative analysis using PCA and t-Distributed Stochastic Neighbor Embedding (t-SNE) to analyze the distributions of the original and synthetic data from the two approaches.

4.1 Data Set and Methodology

Time Series Models Data Set.

To evaluate the efficiency of the InvQG mapping method, we tested the synthesis process on several and different time series models. The statistical models used to simulate the time series dataset were selected from [50] with the propose of analyze particular statistical properties of the data. For our empirical analysis, we use 100 sample paths of length $T = 10000$ of each time series model, in a total of 1000 time series. The dataset consists of 100 iterations of time series data generated for each model, with minor variations in each iteration. Despite these variations, the fundamental dynamics of each model were consistently maintained. The time series statistical models analyzed are described as below (additional details can be found at [50]).

- *White Noise (WN)* process, ϵ_t, is a sequence of independent and identically distributed random variables with mean 0 and constant variance σ_ϵ^2. The process used in this work is generated by: $\epsilon_t \sim N(0, 1)$, and we will refer to this model only as **WN**.
- Linear models

  Autoregressive (AR) model that specifies a linear relationship between the current and past values, we study AR process of order 1, AR(1), generated by the following equation:

  $$Y_t \quad = \quad \phi_1 y_{t-1} + \epsilon_t,$$

  where $\phi_1 = \{-0.5, 0.5, 0.9\}$ is the autoregressive constant and ϵ_t is the white noise. We will refer to these three models as **AR -0.5**, **AR 0.5** and **AR 0.9**, respectively.

  Autoregressive Integrated Moving Average (ARIMA) model is a generalization of the ARMA model for modeling non-stationary time series. It combines three main components: autoregressive (AR) component which specifies a linear relationship between the current and past values, integrated (I) component which refers to the differentiation of observations to make the time series stationary (i.e., using the difference operator $\nabla^d Y_t = (1 - B)^d Y_t$, $d \in \mathbb{N}$), and a moving average (MA) component specifies a linear dependence between a value and past prediction errors. In this work we analyse ARIMA(1,1,0) process, with 1 autoregressive term, 1 integrated term and no moving average term, so the process was generated by the following equation:

  $$(1 - \phi_1 B)\,(1 - B)^d\, Y_t = \epsilon_t, \tag{4.1}$$

  where B represents the backshift operator, $BY_t = Y_{t-1}$, $d = 1$, $\phi_1 = 0.7$ and ϵ_t is the white noise. We will refer to this model only as **ARIMA**.

  Autoregressive Fractionally Integrated Moving Average (ARFIMA) model is a generalization of the ARIMA model that allows fractional differentiation, useful for modeling time series with long range dependence. We analyse ARFIMA(1,0.4,0) generated by the Equation 4.1 where $BY_t = Y_{t-1}$, $d = 0.4$, $\phi_1 = 0.9$ and ϵ_t is the white noise. We will refer to this this model as **ARFIMA**.

- Nonlinear models

  Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model specified by the conditional variance and developed mainly for financial time series. It is a function not only of the squares of past innovations, but also of their own past values. We study GARCH(1,1) generated by:

  $$\sigma_t^2 = \omega + \alpha_1 \epsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2,$$

  where $\omega = 10^{-6}$, $\alpha_1 = 0.1$ and $\beta_1 = 0.8$. We will refer to this model as **GARCH**.

  Integer-Valued Autoregressive (INAR) model that models integer-valued time series, in particular, correlated counts. We analyse INAR(1) that satisfies the following equation:

  $$Y_t = \alpha * Y_{t-1} + \epsilon_t,$$

  where the innovation sequence ϵ_t and the initial distribution are Poisson and $\alpha = 0.5$ and $\epsilon_t \sim Po(1)$ to generate integer valued data with autocorrelation decaying at a rate of 0.5. We will refer to this model only as **INAR**.

  Self-Exciting Threshold Autoregressive (SETAR) process that specifies the nonlinearity in the conditional mean. It models regime changes that approximate a nonlinear function by piece wise linear functions dependent on the regime. We study SETAR process of order 1, SETAR(1), generated by:

  $$Y_t = \begin{cases} \alpha Y_{t-1} + \epsilon_t & , \text{ if } Y_{t-1} \leq r \\ \beta Y_{t-1} + \gamma \epsilon_t, & \text{if } Y_{t-1} > r \end{cases},$$

  where $\alpha = 0.5$, $\beta = -1.8$, $\gamma = 2$ and $r = -1$. We will refer to this model only as **SETAR**.

  Poisson *Hidden Markov Models (HMM)* probabilistic model for the joint probability the random variables. For our study, Y_t is a discrete (or continuous) variable and X_t is a *hidden Markov chain* with a finite number of states. We use 2 states and the transition matrix: $\left[\begin{smallmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{smallmatrix}\right]$. The data are generated from a Poisson distribution with $\lambda = 10$ for the first regime and $\lambda = 15$ for the second. We refer to this model as **HMM**.

The ten time series models used in this work are summarized in Table 4.1, summarizing the time series models parameters, the main characteristic of data to be analyzed, and the notation used to refer to the respective model.

Table 4.1: Summary about the time series models. Parameters, main characteristic of the data sets and notation is included. See [50] for more details.

Process	**Parameters**	**Main Property**	**Notation**
White Noise	$\epsilon_t \sim N(0,1)$	Noise effect	**WN**
AR(1)	$\phi_1 = -0.5$	Smooth oscillation	**AR -0.5**
	$\phi_1 = 0.5$	Moderate persistence	**AR 0.5**
	$\phi_1 = 0.9$	High persistence	**AR 0.9**
ARIMA$(1,1,0)$	$\phi_1 = 0.7$	Stochastic trend	**ARIMA**
ARFIMA$(1,0.4,0)$	$\phi_1 = 0.9$	Long memory effect	**ARFIMA**
GARCH$(1,1)$	$\omega = 10^{-6}$, $\alpha_1 = 0.1$, $\beta_1 = 0.8$	Persistent periods of high or low volatility	**GARCH**
INAR(1)	$\alpha = 0.5$, $\epsilon_t \sim Po(1)$	Correlated counts	**INAR**
SETAR(1)	$\alpha = 0.5$, $\beta = -1.8$, $\gamma = 2$, $r = -1$	Regime-dependent autocorrelation	**SETAR**
Poisson-HMM	$N = 2$, $\left[\begin{smallmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{smallmatrix}\right]$ $\lambda \in \{10, 15\}$	State transitions	**HMM**

Proposed Methodology.

For each time series in the set of time series models, we applied the quantile-based processes to synthesize a corresponding time series data version for each iteration, first we applied the QG mapping (Algorithms 1 and 2) and then we applied the InvQG mapping method (Algorithm 3). We then extracted metrics from both the original and synthetic data using statistical metrics, for this we use the *tsfeatures* package [54]. These metrics allowed us to evaluate the topological characteristics and temporal dynamics of the time series data. Finally, we compare similarity between original and synthetic data analyzing the corresponding statistical features. For this, we calculated the mean and standard deviation of these metrics for all 100 time series models samples, once for the original and another for corresponding synthetic data, and determined the differences between the features of the original data and respective synthetic counterparts. This process can be observed in diagram illustrated in Figure 4.1.

Afterwards, for a empirical analysing in complex networks domain, we used original and synthesized data to generate respective complex networks using three different mapping methods: Natural Visibility Graph (NVG), Horizontal Visibility Graph (HVG) and Quantile Graph (QG). Then, for each network, we calculate a set of topological metrics using the *NetF* [50] that permit to examine global topology for the network. These metrics where then used to visually compare the similarity between the original data networks and the synthetic data networks. For visual comparison of similarity between original and synthetic time series and networks, we plotted a PCA for each time series model set (100 samples). Additionally we plotted all measurements into a single PCA plot. We performed this process for time series data and for complex networks, using both statistical and topological features. This process is also illustrated in Figure 4.1.

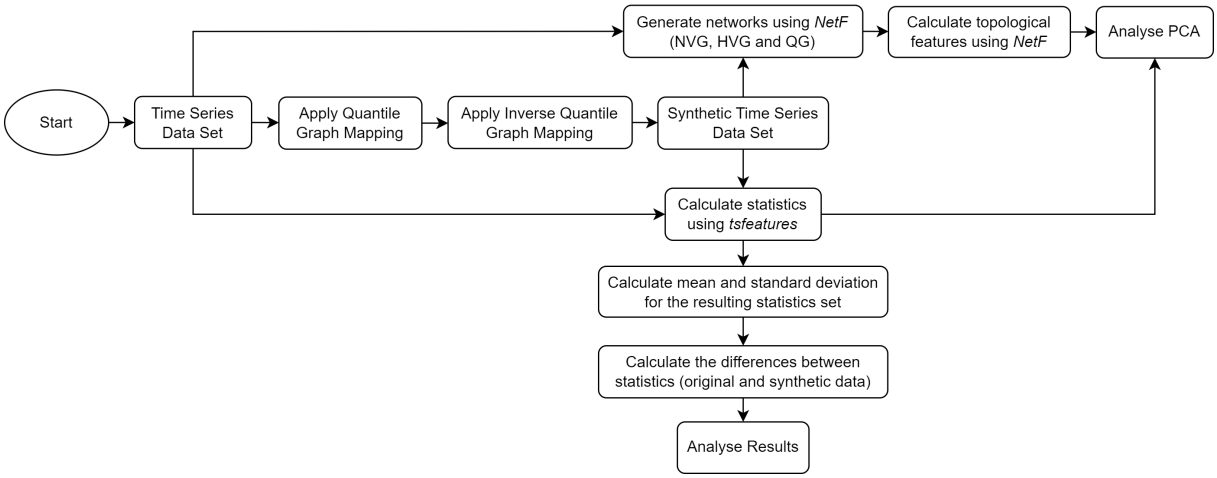


Figure 4.1: Diagram of the methodology implemented in this work.

Additionally, we compared the quantile-based process to TimeGAN. We adapted the code from [19] to handle univariate time series data. Subsequently, we synthesized a single time series for each model and compared the results using t-SNE and PCA, as provided by the authors.

This approach allowed us to assess the stability and reliability of the proposed method under slight data fluctuations.

4.2 *tsfeatures*: Statistical Features

In this section, we analyse the statistical features of time series resulting from calculating *tsfeatures* [54] for the original and synthetic time series data. The Tables 4.2 and 4.3 present the mean and standard deviation of each statistical measure across the original 100 time series iterations and respective synthetic counterparts, by time series model. All resulting measurements can also be seen in the Appendix A.1 summarized in the form of boxplots. Furthermore, and to analyse the similarities and differences between the original and synthetic data measurements, we also display this information in Figures 4.3 and 4.4 that show the differences between the original 100 time series iterations and respective synthetic counterparts, also by time series model.

We start analysing the results of the *trend*, *linearity* and *curvature* measurements, which are presented in Table 4.2.

The *trend* allows to examine the direction of a time series when progressing over time. Higher trend values indicate a stronger trend and lower values indicate a weaker or non-existent trend. Results show that for time series models where trend was initially low or non-existent, the method maintained this behaviour and synthesized trend-free time series data. However, for models where trends was initially high, the mapping method successfully captured and preserved the trend-like behaviour in respective synthetic counterparts. Therefore, trend-wise, the synthesizing successfully captures trend behaviour and embeds it into synthetic data, as we can see from the

boxplots presented in the first line of Figure 4.3 where we do not observe substantial differences between the original and synthetic data. In the Figure 4.2 we plot an example of original versus synthetic time series data for a single repetition of time series models **ARIMA** and **ARFIMA**.

Table 4.2: Statistical features (*trend*, *linearity* and *curvature*) for original time series data and and respective synthetic counterparts. The values correspond to the mean (and standard deviation) of the 100 instances of each time series models for each feature.

Model	Data	trend	linearity	curvature
WN	Original	0.00081 (0.00056)	0.10912 (1.09429)	−0.01800 (0.98653)
	Synthetic	0.00081 (0.00063)	−0.18415 (0.99645)	−0.04706 (0.94711)
AR 0.5	Original	0.00480 (0.00224)	0.03684 (1.72586)	0.07395 (1.77812)
	Synthetic	0.00470 (0.00198)	−0.07068 (1.59329)	−0.14457 (1.56811)
AR -0.5	Original	0.00013 (0.00012)	−0.05565 (0.56968)	−0.00868 (0.50985)
	Synthetic	0.00014 (0.00013)	0.14271 (0.58558)	−0.07492 (0.51484)
AR 0.9	Original	0.04061 (0.01193)	−1.12671 (4.10122)	−0.02913 (4.47904)
	Synthetic	0.03788 (0.01131)	−0.60999 (4.31005)	−0.11794 (3.91387)
ARIMA	Original	0.94570 (0.03549)	−1.59836 (63.30089)	−2.66558 (44.85890)
	Synthetic	0.91383 (0.07537)	−0.74836 (58.24916)	−1.81284 (42.76233)
ARFIMA	Original	0.37506 (0.09449)	0.40366 (25.15691)	−0.33583 (20.18378)
	Synthetic	0.41627 (0.07800)	1.00782 (15.72771)	0.45231 (15.73553)
GARCH	Original	0.00081 (0.00052)	−0.05508 (1.02774)	−0.13518 (1.10424)
	Synthetic	0.00072 (0.00045)	0.07019 (1.02521)	−0.11739 (1.01895)
INAR	Original	0.00102 (0.00089)	−0.11170 (1.30331)	−0.00125 (1.38863)
	Synthetic	0.00110 (0.00086)	−0.12017 (1.51568)	−0.18914 (1.24791)
SETAR	Original	0.00025 (0.00018)	0.12840 (0.74152)	0.01367 (0.61749)
	Synthetic	0.00017 (0.00013)	0.02709 (0.59507)	−0.04026 (0.50647)
HMM	Original	0.00769 (0.00248)	0.26983 (2.05605)	−0.29090 (1.83514)
	Synthetic	0.00260 (0.00114)	0.07511 (1.41590)	0.24758 (1.34251)

(a) **ARIMA** (b) **ARFIMA**

Figure 4.2: Original time series *vs* synthetic time series comparison for (a) **ARIMA** and (b) **ARFIMA** time series models.

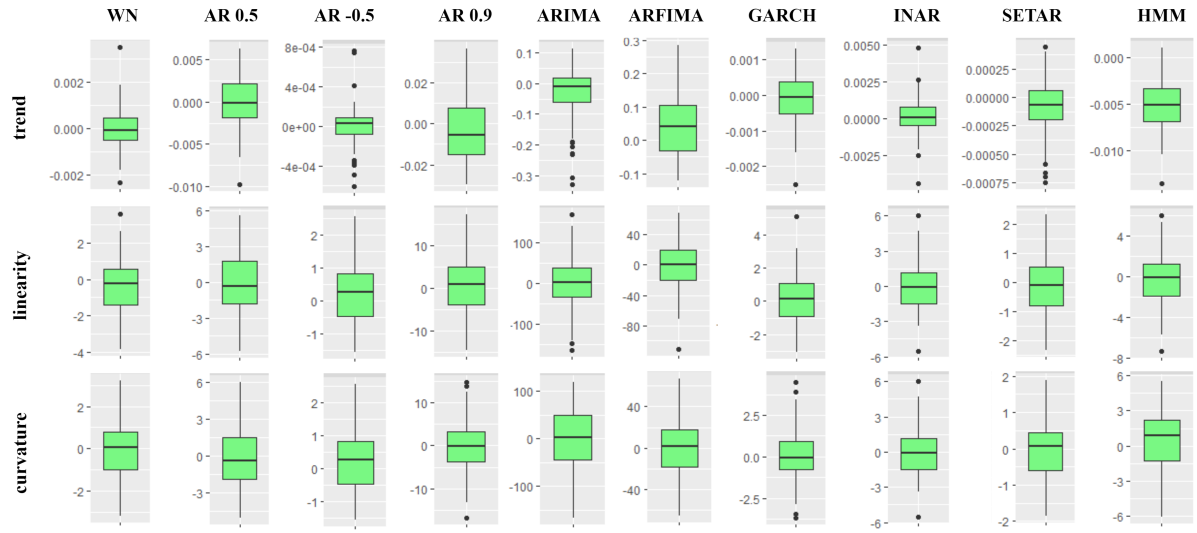


Figure 4.3: Boxplots with the differences of the statistical features (*trend*, *linearity* and *curvature*) between original time series and respective synthetic counterparts.

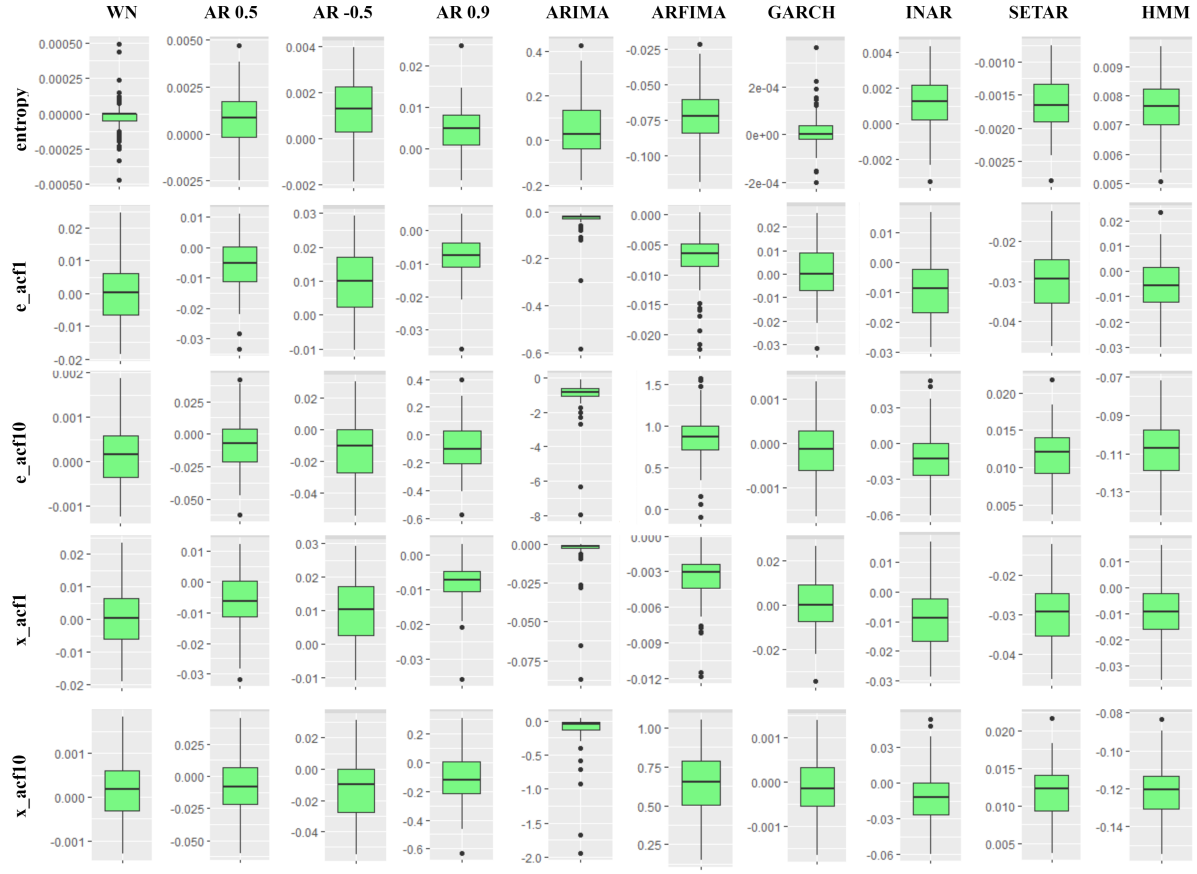
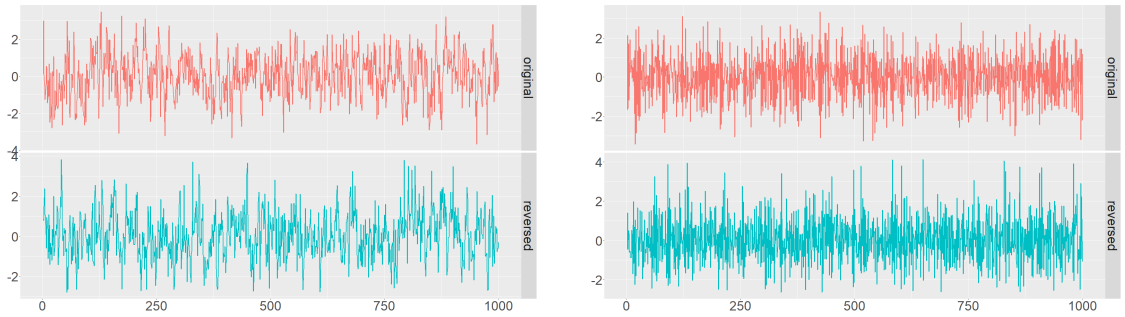


Figure 4.4: Boxplots with the differences of the statistical features (*entropy*, *e_acf1*, *e_acf10*, *x_acf1* and *x_acf10*) between original time series and respective synthetic counterparts.

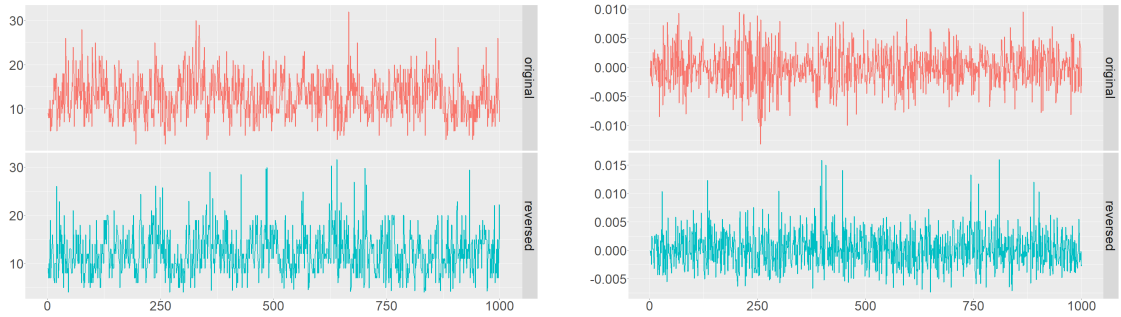
The *linearity* measures the extent to which a time series can be described by a linear relationship. Higher values indicate a stronger linear trend, while lower values suggest more complex, non-linear patterns. Results present a linearity decrease for models **WN**, **AR 0.5**, **INAR**, **SETAR** and **HMM**. For these models data was synthesized with a lowered temporal dynamic when compared to the original counterparts, suggesting more complex temporal structures. In the other hand, **AR -0.5**, **AR 0.9**, **ARIMA**, **ARFIMA** and **GARCH** present a increase in linearity. This indicates a simplification of original versions, and a decrease in complexity of temporal dynamics. Overall, the method's impact on the linearity varies depending on the time series model, as we can see in the second line of Figure 4.3. In the Figure 4.5 we plot an example of original versus synthetic time series data for a single repetition of time series models **AR 0.5** and **AR -0.5**.



(a) **AR 0.5** (b) **AR -0.5**

Figure 4.5: Original time series *vs* synthetic time series comparison for (a) **AR 0.5** and (b) **AR -0.5** time series models.

The *curvature* measures the extent in which a time series exhibits changes in direction while progressing over time. Higher results suggest more pronounced and frequent changes, while lower values indicate a smoother and more linear progression. Models such as **ARIMA**, **ARFIMA**, **GARCH** and **HMM** present a higher curvature measurement when compared to original counterparts, being more pronounced in **ARFIMA** and **ARIMA** (see line three of Figure 4.3). For these cases, the mapping method amplified the curvature behaviour in models where curvature was naturally present. In the other hand, remainder time series models measured lower in curvature. For these, a smoother and less complex synthetic series versions where generated. Overall, the method slightly amplifies the curvature for models where curvature was originally high, but maintains or slightly smooths the curvature level for models where the curvature was originally lower. Figure 4.6 shows an example of original versus synthetic time series data for a single repetition of time series models **HMM** and **GARCH**.



(a) **HMM** (b) **GARCH**

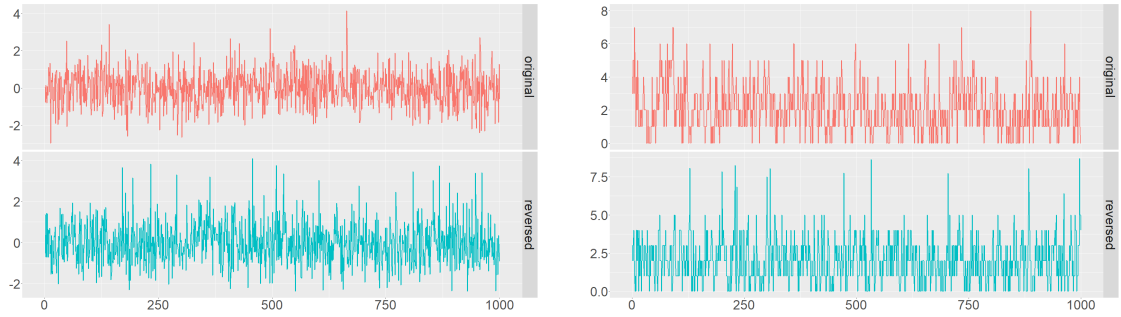
Figure 4.6: Original time series *vs* synthetic time series comparison for (a) **HMM** and (b) **GARCH** time series models.

Next we move to the analysis of *entropy*, *e_acf1*, *e_acf10*, *x_acf1* and *x_acf10* measurements, which are presented in Table 4.3.

Table 4.3: Statistical features (*entropy*, *e_acf1*, *e_acf10*, *x_acf1* and *x_acf10*) for original time series data and and respective synthetic counterparts. The values correspond to the mean (and standard deviation) of the 100 instances of each time series models for each feature.

Model	Series	entropy	e_acf1	e_acf10	x_acf1	x_acf10
WN	Original	0.99997 (0.00008)	0.00015 (0.00970)	0.00095 (0.00043)	0.00096 (0.00978)	0.00094 (0.00043)
	Synthetic	0.99996 (0.00008)	0.00069 (0.01410)	0.00112 (0.00052)	0.00150 (0.01408)	0.00111 (0.00052)
AR 0.5	Original	0.96869 (0.00116)	0.49827 (0.00792)	0.32903 (0.01614)	0.50068 (0.00788)	0.33517 (0.01644)
	Synthetic	0.96943 (0.00174)	0.49262 (0.01221)	0.32212 (0.02244)	0.49501 (0.01221)	0.32815 (0.02310)
AR -0.5	Original	0.96869 (0.00124)	−0.50096 (0.00837)	0.33540 (0.01660)	−0.50077 (0.00840)	0.33524 (0.01662)
	Synthetic	0.96992 (0.00181)	−0.49097 (0.01271)	0.32368 (0.02362)	−0.49076 (0.01270)	0.32350 (0.02360)
AR 0.9	Original	0.82055 (0.00405)	0.89522 (0.00412)	3.54816 (0.13601)	0.89947 (0.00386)	3.72246 (0.13876)
	Synthetic	0.82525 (0.00673)	0.88719 (0.00737)	3.44923 (0.20468)	0.89146 (0.00711)	3.61253 (0.20650)
ARIMA	Original	0.22995 (0.08406)	0.99802 (0.00028)	9.31591 (0.07944)	0.99970 (0.00016)	9.94006 (0.02629)
	Synthetic	0.28068 (0.10142)	0.96241 (0.06413)	8.32959 (0.98317)	0.99580 (0.01125)	9.80759 (0.28586)
ARFIMA	Original	0.64066 (0.02217)	0.98666 (0.00115)	7.42969 (0.18908)	0.99165 (0.00140)	8.32613 (0.26001)
	Synthetic	0.56911 (0.02308)	0.97944 (0.00398)	8.28260 (0.22745)	0.98799 (0.00257)	8.96321 (0.18152)
GARCH	Original	0.99996 (0.00007)	0.00073 (0.01156)	0.00117 (0.00045)	0.00153 (0.01160)	0.00116 (0.00045)
	Synthetic	0.99998 (0.00004)	0.00156 (0.01407)	0.00103 (0.00046)	0.00228 (0.01402)	0.00104 (0.00046)
INAR	Original	0.96886 (0.00143)	0.49879 (0.00962)	0.33247 (0.01975)	0.49929 (0.00961)	0.33383 (0.01984)
	Synthetic	0.97000 (0.00210)	0.48987 (0.01445)	0.32046 (0.02676)	0.49043 (0.01447)	0.32193 (0.02697)
SETAR	Original	0.99593 (0.00029)	−0.06764 (0.00483)	0.03562 (0.00263)	−0.06738 (0.00485)	0.03547 (0.00264)
	Synthetic	0.99431 (0.00041)	−0.09722 (0.00806)	0.04742 (0.00327)	−0.09704 (0.00809)	0.04730 (0.00328)
HMM	Original	0.98504 (0.00103)	0.25920 (0.00953)	0.17820 (0.01409)	0.26490 (0.00962)	0.19299 (0.01597)
	Synthetic	0.99260 (0.00071)	0.25384 (0.01174)	0.07077 (0.00703)	0.25578 (0.01164)	0.07211 (0.00706)

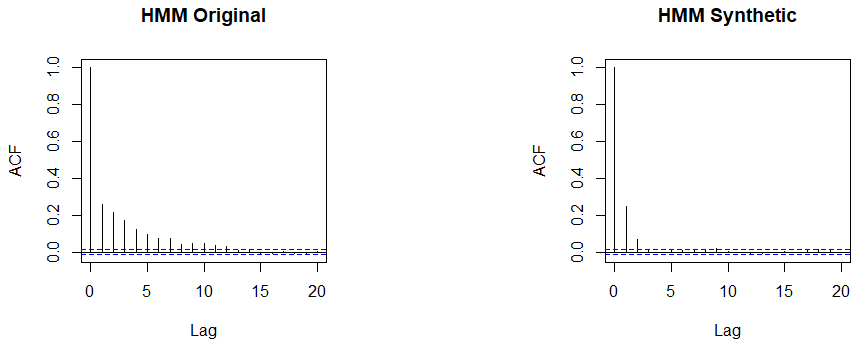
The *entropy* measures randomness or unpredictability in a time series. Higher values indicating more randomness and lower values more predictability and structure. Overall, synthetic versions present similar results accordingly to respective original counterparts. Differences in the values (as shown in first line of the Figure 4.4) suggest that the quantile-based method successfully preserves the original entropy of the original time series with minor deviations. Figure 4.7 shows an example of original versus synthetic time series data for a single repetition of time series models **WN** and **INAR**.



(a) **WN** (b) **INAR**

Figure 4.7: Original time series *vs* synthetic time series comparison for (a) **WN** and (b) **INAR** time series models.

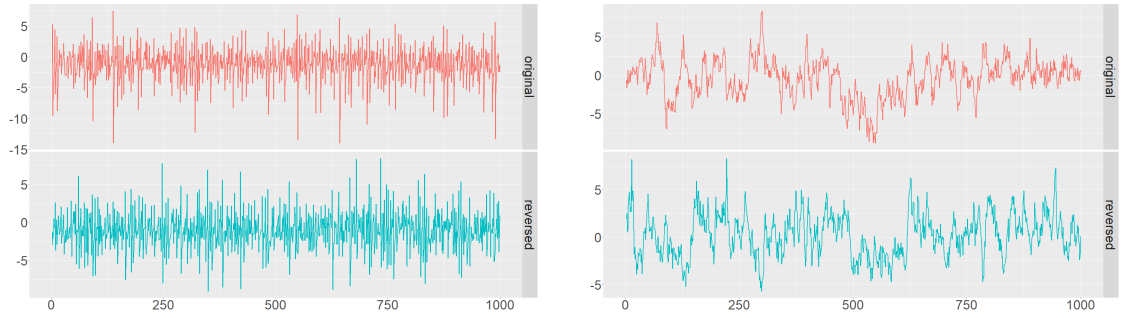
The *e_acf1* and *e_acf10* measures represent the first and tenth lags of the Autocorrelation Function (ACF) of the differenced series. These metrics analyse the correlation between a time series and a lagged version of itself by using either lag 1 (*e_acf1*) or lag 10 (*e_acf10*). The results assist in understanding temporal dynamics and dependencies present within the time series itself. Overall, the obtained measurements indicate that most models were able kept a similar value to original counterparts. Overall, the method successfully maintains, with a very slight reduction, the temporal dependencies in synthetic data. The worst case is **HMM**, where a higher magnitude loss occurred from original to synthetic versions, as seen in the ACF plot in Figure 4.8.



(a) Original **HMM** ACF (b) Synthetic **HMM** ACF

Figure 4.8: Autocorrelation function (ACF) plot of an original **HMM** time series instance and corresponding synthetic time series instance.

Finally, *x_acf1* and *x_acf10*, represent the first and tenth lags of the autocorrelation function of the original (non-differenced) series. The obtained results for these metrics are similar to *e_acf1* and *e_acf10* explained above and allow to reach the same conclusions. The method successfully captures, with a slight loss, temporal correlations from original data and correctly traverses it into synthetic data. In Figure 4.9 we present the time series models **SETAR** and **AR 0.9** to visualize original versus synthetic data.



(a) **SETAR** (b) **AR 0.9**

Figure 4.9: Original time series *vs* synthetic time series comparison for (a) **SETAR** and **AR 0.9** time series models.

In Figure 4.10 we present for each time series model a PCA plot containing resulting statistical features (*tsfeatures*) calculated from time series repetitions within each time series model. Red dots represents a complete set of *tsfeatures* for a single time series repetition calculated from original data, whereas blue dots represent synthetic counterparts.

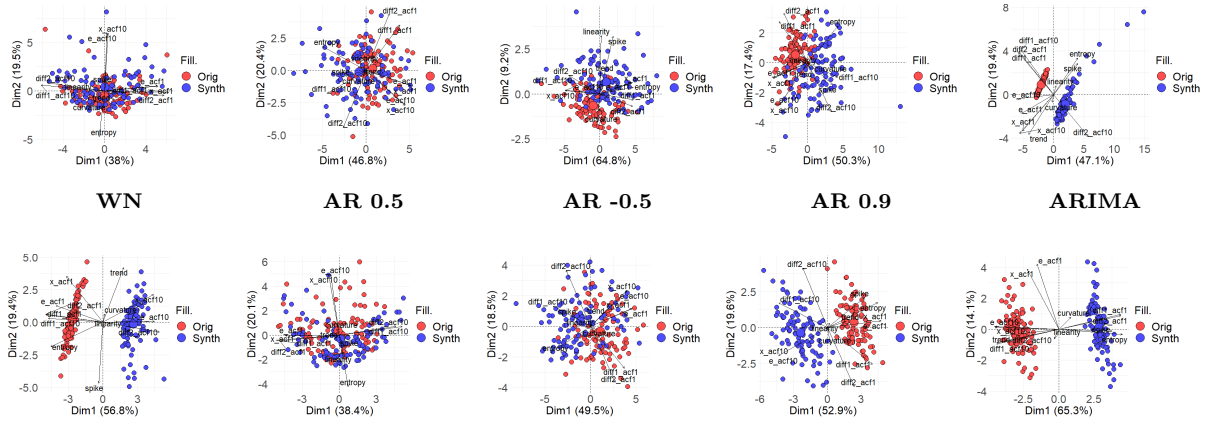


Figure 4.10: PCA for statistical metrics (*tsfeatures*) results of all time series repetitions (original and synthetic data) by model.

Observing Figure 4.10, it is possible to notice that synthetic time series exhibit patterns that are somewhat similar to the original data. However, there is also a noticeable tendency for original and synthetic data to cluster differently and apart from each other. This suggests that while original data and their synthetic counterparts successfully share certain common characteristics and dynamics, the distinct clustering behavior also indicate the present of some differences as well.

The most distinguishable clusters that are visibly separated include **ARIMA**, **HMM**, and **ARFIMA**. A detailed analysis of the key features set of *tsfeatures* responsible for this gap reveals that it is primarily due to the set of differenced correlation features. Other features show varying degrees of similarity.

The reason behind this phenomenon could be attributed to the lag-one probabilistic analysis inherent to the QG mapping method. This results in varying temporal correlations within synthetic time series, causing a divergence from the original data. As a consequence, synthetic data may exhibit deviations in temporal progression while preserving other key characteristics.

Additionally, it is also possible to observe a slight tendency for blue dots to spread wider in the plane when compared to red dots. This indicates the presence of a higher variability in synthetic data when compared to original data. The probable culprit of this behaviour is the random nature of the synthesizing process, which due to being probabilistic, introduces some degree if entropy in synthetic data.

Next, in Figure 4.11, we plot the same statistical features set information explored so far, but condensed into the same PCA plot.

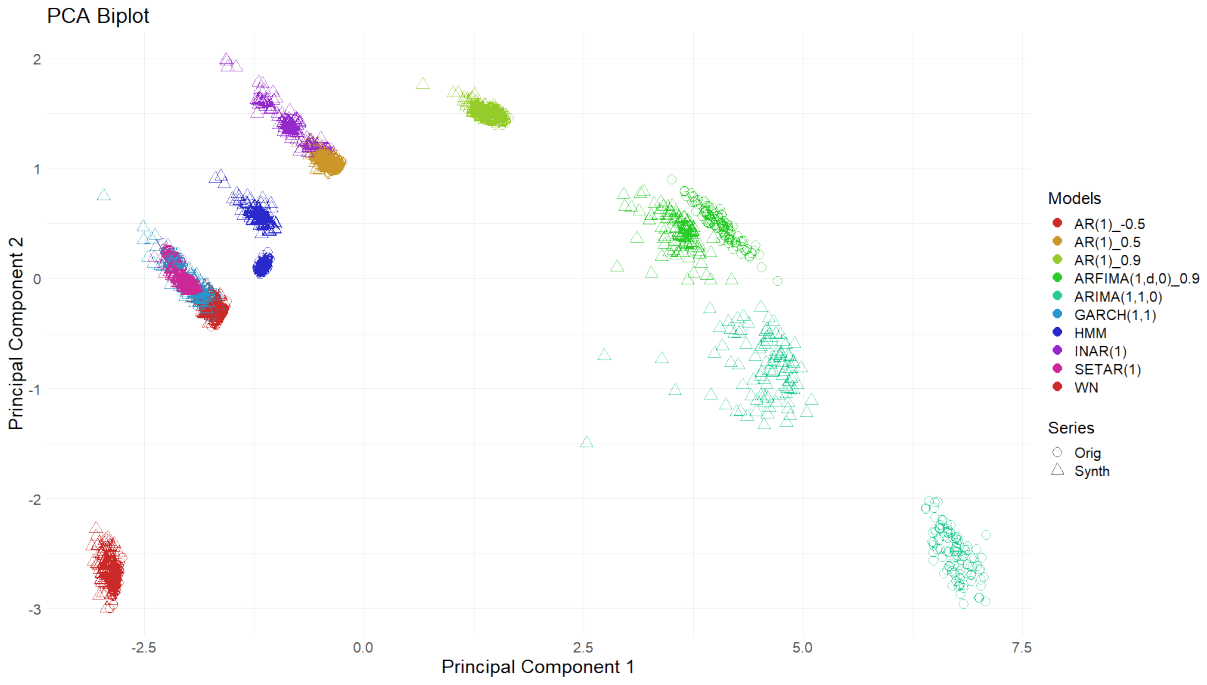


Figure 4.11: PCA of statistical features (*tsfeatures*) for all time series models.

Figure 4.11 indicates that each time series model tends to form its own distinct cluster, with varying degrees of variability observed between corresponding original and synthetic data. Models characterized by higher variability in temporal correlation, such as **ARIMA** and **ARFIMA**, exhibit greater dispersion between their original and synthetic counterparts. In contrast, other models show tighter clustering.

This distinctive clustering behavior of time series models suggests that they share fundamental similarities. Therefore, the QG method effectively captures essential characteristics including low-level temporal correlations. However, it may struggle to capture higher-order temporal dependencies, resulting in synthetic data that tends to be more linear. Moreover, the method introduces additional variability or randomness during the synthesis process.

4.3 *NetF*: Network Features

In this chapter, we evaluate the degree of similarity between complex networks that have been mapped using synthetic time series and networks that have been mapped using corresponding original data using.

In Figure 4.12, we present a PCA plot for each time series model, showing the *NetF* results. Red dots represent the complete set of *NetF* metrics derived from three complex networks (*NVG*, *HVG* and *QG*) mapped from original data, while blue dots represent their synthetic counterparts. A summary of the base metrics resulting from *NetF* can be found in Tables A.1 and A.2.

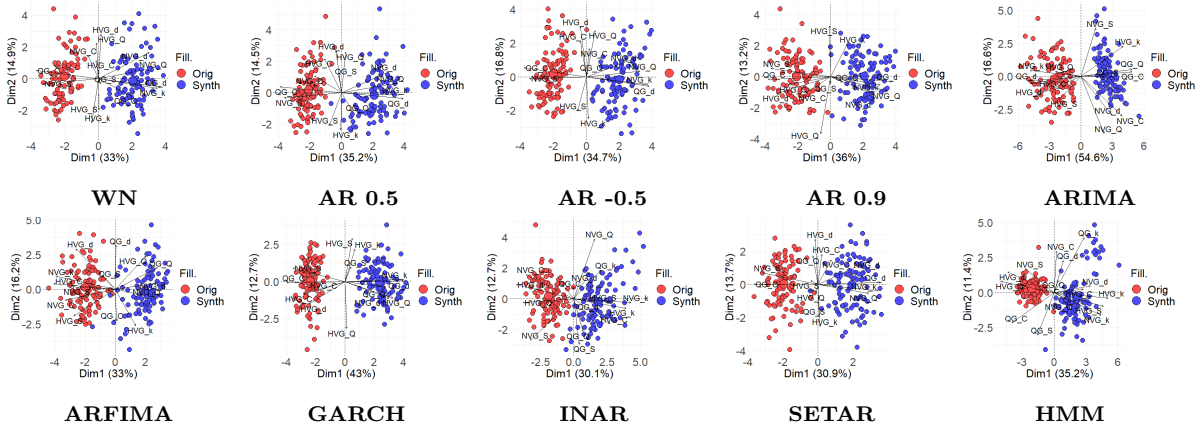


Figure 4.12: PCA for *NetF* results of all complex networks repetitions by model.

After a close analysis of all plots in Figure 4.12, we conclude that results are highly dependent, and vary accordingly, to the time series model and the mapping method used (*NVG*, *HVG* or *QG*). A close analysis of modularity (Q) and average weighted degree ($\bar{k}$) suggest that synthetic networks successfully captured these topological characteristics from original networks. However, it is also visibly clear a separation between original and synthetic networks in all time series models. This indicates that some topological characteristics where not fully captured, such as

clustering coefficient (C) and average path length ($\bar{d}$).

A divergence in the clustering coefficient (C) could suggest that synthetic data had an variability increase in time series data values, or different temporal correlations in resulting synthetic time series data. This is particularly relevant for visibility graphs, where observations with higher values form larger clusters. This cause them to be more sensitive to magnitude or amplitude changes. Data with more variability, or altered temporal correlations will create different visibility lines and ultimately alter the clustering behaviour in resulting synthetic networks.

QG is also prone to high sensitivity to this added variability in data values. Transitions between quantiles for two consecutive data points will occur differently with this type amplitude or magnitude changes. This unavoidably will result in different quantile transitions and therefore potentially creating different nodes and directed weighted links. Leading to a all new different network structure.

Consequently, changes in clustering behaviour will also naturally affect the number of communities (S) and average path length ($\bar{d}$) between nodes.

Next, in Figure 4.13, we plot the same *NetF* measures explored above, but condensed into the same PCA plot. For visualization purposes we excluded **ARIMA**. The full PCA plot including **ARIMA** can be found in Appendix A.3.

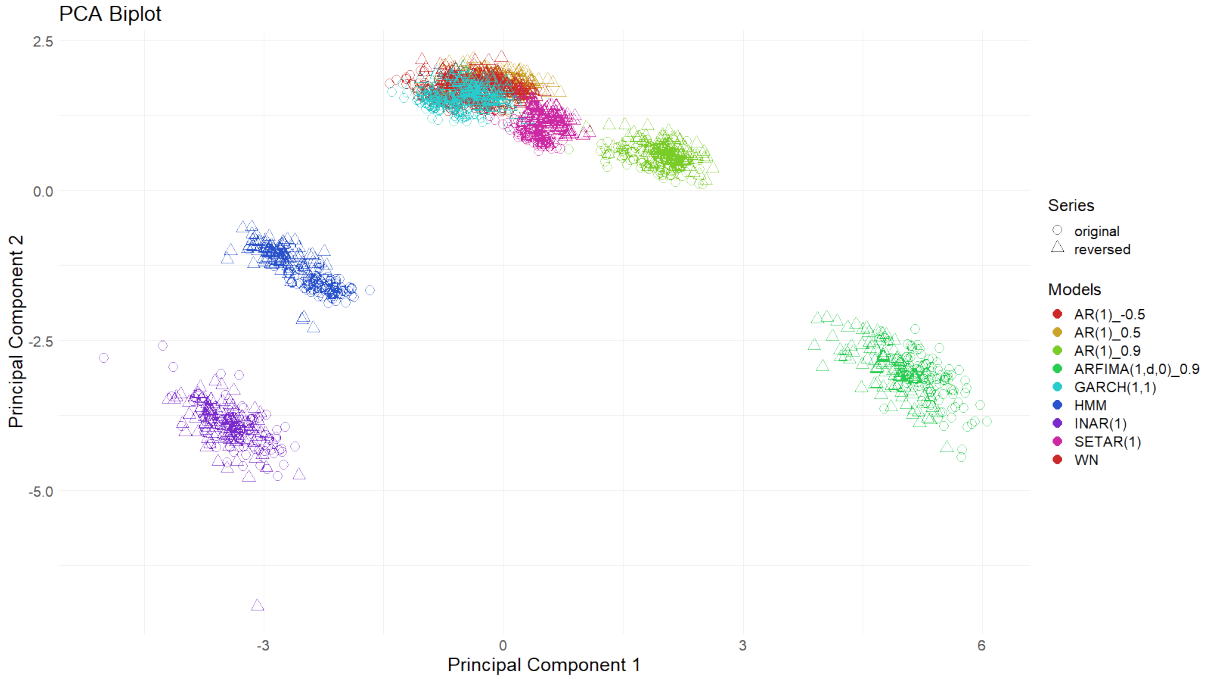


Figure 4.13: PCA of network features (*NetF*) for all time series models. **ARIMA** model was removed to better visualization.

Figure 4.13 permits to conclude that, similarly to the previous *tsfeatures* results, synthetic networks partially capture topological characteristics from original networks. However, due to the sensitivity nature of networks mapping methods, the clusters are not so well defined and apart when compared to plotting time series data. Nonetheless, we can still visibly see distinct clusters for each time series models networks. This suggests that synthetic networks also have utility and could potentially be used instead of original data networks.

4.4 TimeGAN *vs* Quantile-based Mappings

In this section we perform a comparison between the QG method versus TimeGAN. To perform this comparison, for each time series model, we select an original time series repetition and respective synthetic version generated using the QG method. We then generate a second synthetic time series using the TimeGAN method. Afterwards, using a method provided by the TimeGAN authors in [19], we plot t-SNE plots for each time series model and by and synthesizing method. This permits to visually compare the performance of these two synthesizing methods.

We present the results in Figures 4.14 and 4.15. The first row of plots corresponds to QG mapping method, whereas the second row is data generated by TimeGAN. Red dots represent a set of observations from original time series data and blue respective synthetic counterparts. By observing the Figures 4.14 and 4.15, we can conclude that in general QG method slightly outperformed TimeGAN method, specially in models **WN**, **AR -0.5**, **INAR** and **SETAR**.

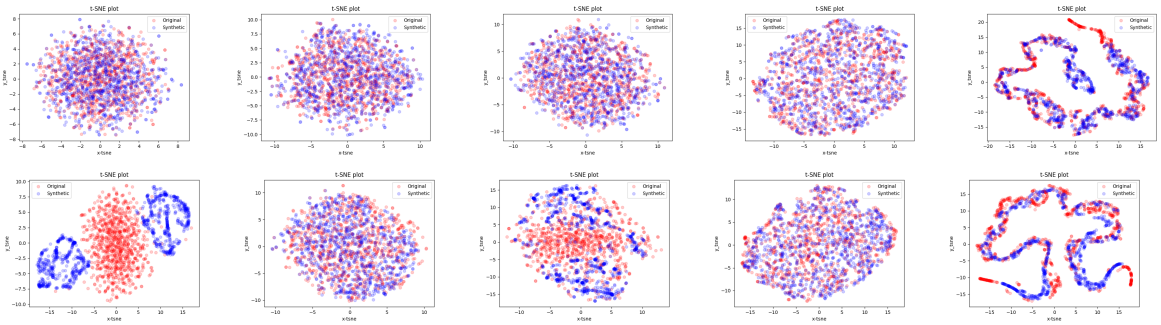
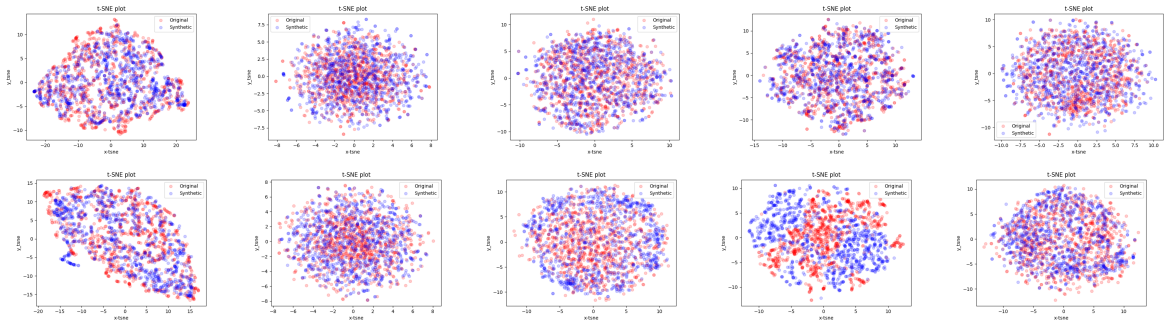


Figure 4.14: t-SNE comparison between quantile-based mapping (first row) and TimeGAN (second row) methods for **WN**, **AR 0.5**, **AR -0.5**, **AR 0.9** and **ARIMA** models (see Appendix A.4 for PCA).



ARFIMA **GARCH** **INAR** **SETAR** **HMM**

Figure 4.15: t-SNE comparison between quantile-based mapping (first row) and TimeGAN (second row) methods for **ARFIMA**, **GARCH**, **INAR**, **SETAR** and **HMM** models (see Appendix A.5 for PCA).

Chapter 5

Privacy Metrics and Application

In this chapter, we evaluate the degree of privacy of our InvQG method using two statistical privacy metrics that jointly analyse the original and synthetic counterpart. These metrics determine the level of common information shared between original and synthetic data, and whether the synthetic data still presents privacy concerns.

5.1 Mutual Information & Conditional Entropy

Mutual Information (MI) is widely used in privacy research [55–59] and evaluates the privacy level by quantifying the common information between clear and noisy data. The smaller the result, the lower the mutual dependence between clear and noisy data points is, and therefore the higher is the privacy protection. Higher results indicate that substantial information from the original data is maintained in the transformed or reconstructed data, potentially compromising privacy. Mutual information $I(Y;X)$ is defined in the following equation:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x) \cdot p(y)} \right) \tag{5.1}$$

Conditional Entropy (CE) quantifies the uncertainty or unpredictability of one random variable when knowledge of another is given. A higher CE level suggests that transformed data provides little to no insight into the original data. Or, in other words, high CE results means that observing the transformed data will yield little to no information about original data, and therefore effectively preserving data privacy. In the other hand, a low CE result suggests that transformed data does reveals significant information about the original series, highlighting a potential privacy concern. Conditional entropy $H(Y|X)$ is defined in the following equation:

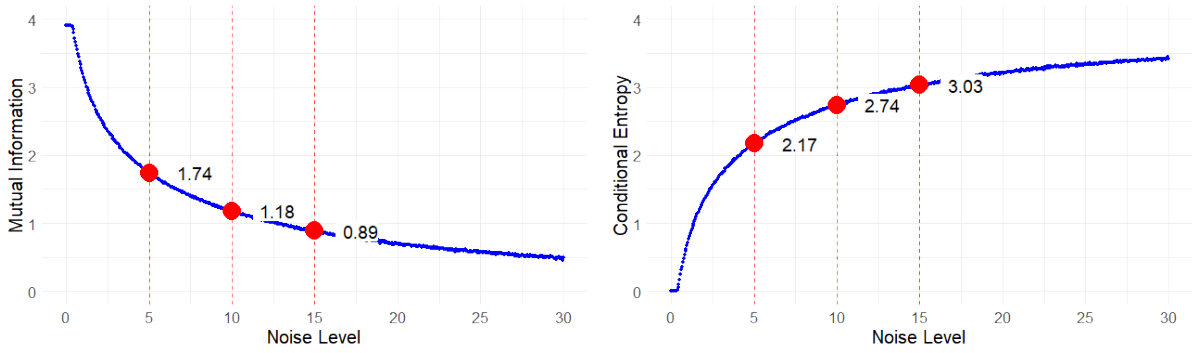
$$H(Y|X) = -\sum_{x \in X} \sum_{y \in Y} p(x,y) \log p(y|x) \tag{5.2}$$

To illustrate the behaviour of these metrics when measuring the degree of privacy between a time series and a synthesized, or perturbed, version of itself, we consider a time series $X(t)$ generated using the following model defined in [23]:

$$X(t) = \mod (X(t-1) + 0.02, 1) \tag{5.3}$$

with $T = 10000$ observations and $X(1)$ generated from a uniform distribution in $[0, 1]$. Then, we add to $X(t)$ noise generated from a normal distribution with mean 0 and standard deviation, sd, depending on the range of the data and a parameter denominated noise level. The values for the noise level vary in the range $n = 0(0.05)30$ in a total of 1001 values. The perturbed time series for noise level n is denoted by $X_n = (X_n(1), \ldots, X_n(T))$, note that $n = 0$ corresponds a time series without added noise.

The MI and CE metrics, calculated for all pairs of X_0 and X_n and discretizing the time series data values into $\sqrt{T}$ bins, are represented in Figure 5.1.



(a) Mutual Information

(b) Conditional Entropy

Figure 5.1: Mutual Information and Conditional Entropy when the data is perturbed with varying noise levels.

In Figure 5.1a we observe a progressive loss of mutual information shared between the time series and the perturbed version of itself, whereas in Figure 5.1b we observe an increase in the uncertainty of both series being progressively related. The three vertical dashed red lines indicate the values of the MI and CE for three specific noise levels $n = 5, 10$, and 15 with the corresponding time series represented in Figure 5.2.

The values of MI and CE *per se* are not easily interpretable since the range of values depend on the data. In the case of MI the minima is 0 and the maximum is attained when comparing X_0 with itself, corresponding to the maximum amount of common information shared with the original time series.

CE also attains the minima at 0, when X_0 is compared with itself. To approximate the maximum we proceed as follows:

1. compute $H(X_0|X_n)$ for varying levels of noise;
2. determine k such that $H(X_0|X_k) \approx H(X_0|X_n), n \geq k$ (elbow type test)
3. consider as maximum $H(X_k)$

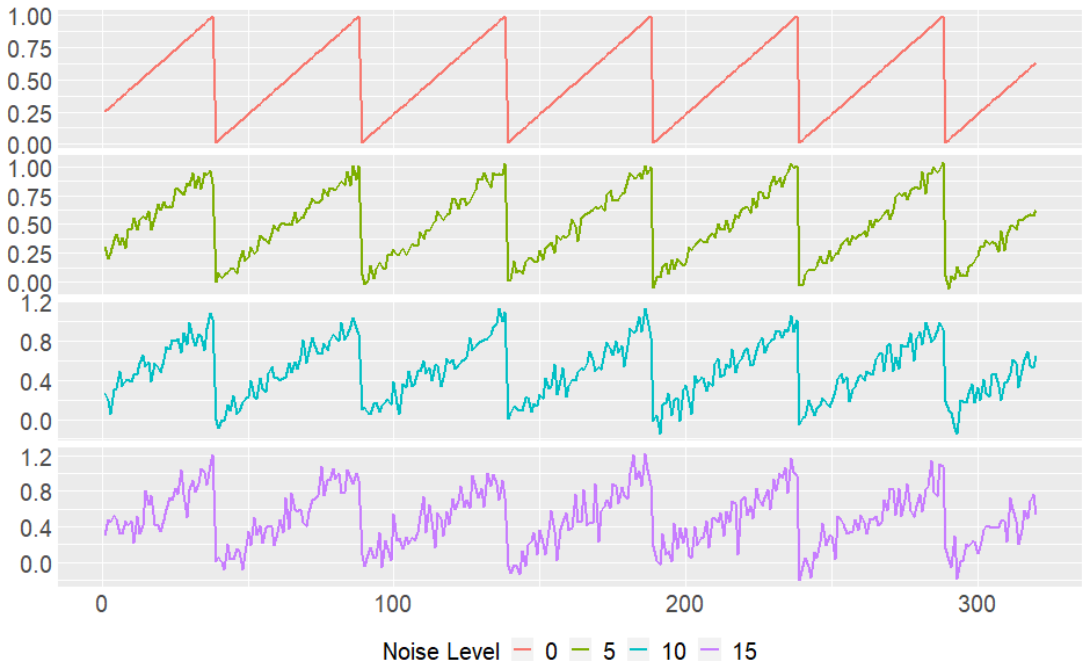


Figure 5.2: Illustrative example of perturbing a time series (X_0 in top panel) by adding noise. The three bottom panels represent X_5, X_{10} and X_{15}.

5.2 Application

In this section we proceed to evaluate the degree of privacy provided by data synthesized using the QG method. To achieve this we resort to the set of time series considered in Chapter 4.1: 10 time series models, 100 realizations of each and synthetic counterparts generated by QG. For each pair original, synthetic we calculated MI and CE. The means (standard deviations) are computed for each time series model and represented in red in figures 5.3 and 5.4.

To compare with perturbed data we use the approach delineated in the previous section for each time series model and plotted in blue in Figures 5.3 and 5.4.

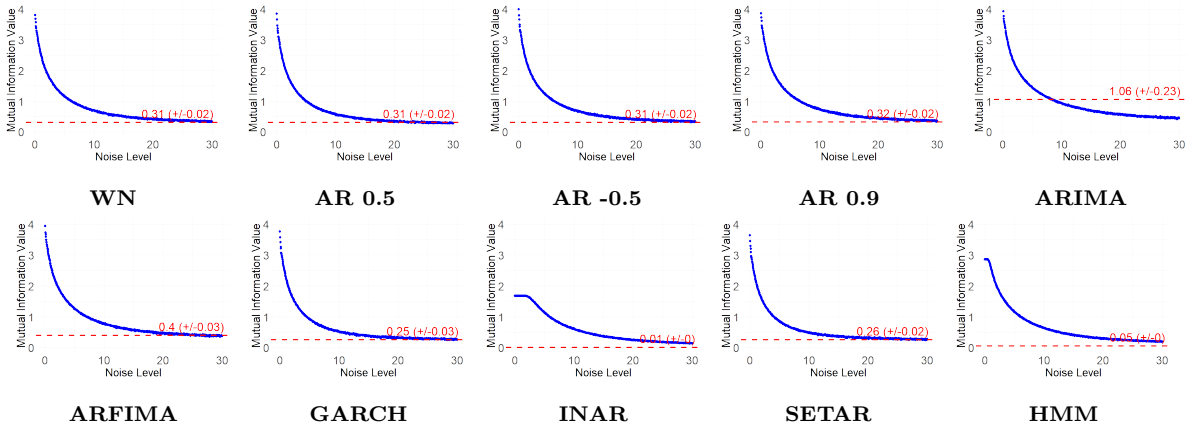


Figure 5.3: Results of the Mutual Information metric for the generated synthetic time series models. Metric plots are presented for each statistical model.

By analysing Figure 5.3, we observe that common information is nearly non-existing between original and synthetic data. The exception is **ARIMA** model, which scored slightly higher. These results indicate that the QG method succeeded in hiding key identifiers or patterns regarding individual behaviours in synthetic data. Therefore, synthetic data is voided of common information from original data, indicating a high privacy degree.

Next, in Figure 5.4, we present conditional entropy results.

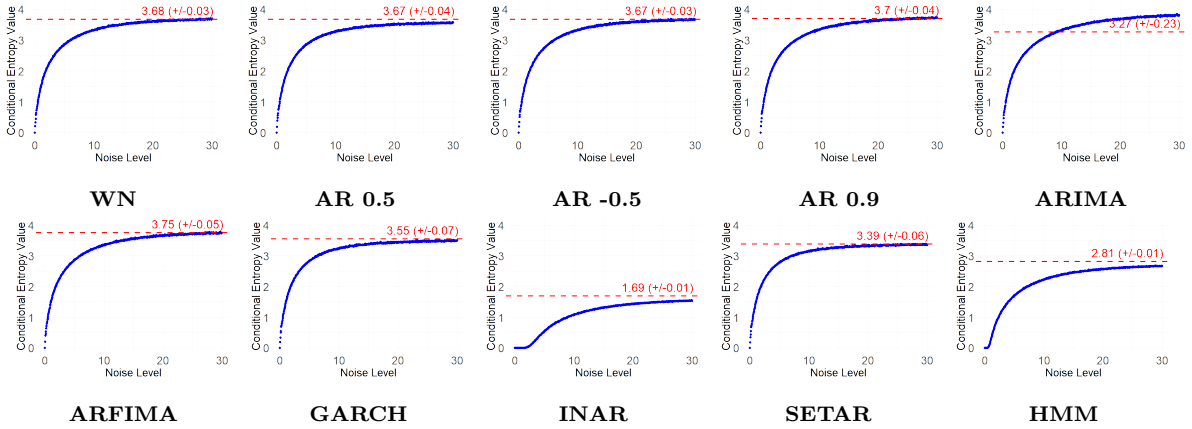


Figure 5.4: Results of the Conditional Entropy metric for the generated synthetic time series models. Metric plots are presented for each statistical model.

Figure 5.4 shows that conditional entropy behaved similarly to mutual information. Synthetic data is found to be measured at a near maximum amount of uncertainty. This indicates that analysing synthetic data will not provide any insight into original key identifiers or patterns regarding individual behaviours. It also confirms a high privacy degree.

We further detail the above analysis for 2 time series models: **ARIMA** and **INAR** models. Figure 5.5 represents an **ARIMA** time series (top panel), its synthetic counterpart (2nd panel from the top) and perturbed time series (3 bottom panels). Table 5.1 presents the corresponding MI and CE results.

Figure 5.5: Comparison between Original vs Synthetic vs Noisy for **ARIMA** time series data.

Series	**Mutual Information**	**Conditional Entropy**
Synthetic	1.0805859	3.188445
Noise Level 5	1.4664478	2.802583
Noise Level 10	0.9339420	3.335089
Noise Level 15	0.7032133	3.565817

Table 5.1: Mutual Information and Conditional Entropy results for **ARIMA** time series data illustrated (in part) in Figure 5.5.

The results in Table 5.1 indicate that the higher the level of the added noise the better the privacy measures, with $n = 10$ and $n = 15$ surpassing MI and CE values for the synthetic data. However, we can question the effect of such high levels of added noise in terms of data utility. A clue to the answer is provided in Figure 5.5 which indicates that high levels of noise may lead to diminished data utility.

The results for an **INAR** time series are presented in Figure 5.6 and Table 5.2. We note that the privacy metrics present better results in this case. This may be explained by the fact that an **ARIMA** is not stationary while an **INAR** is stationary.

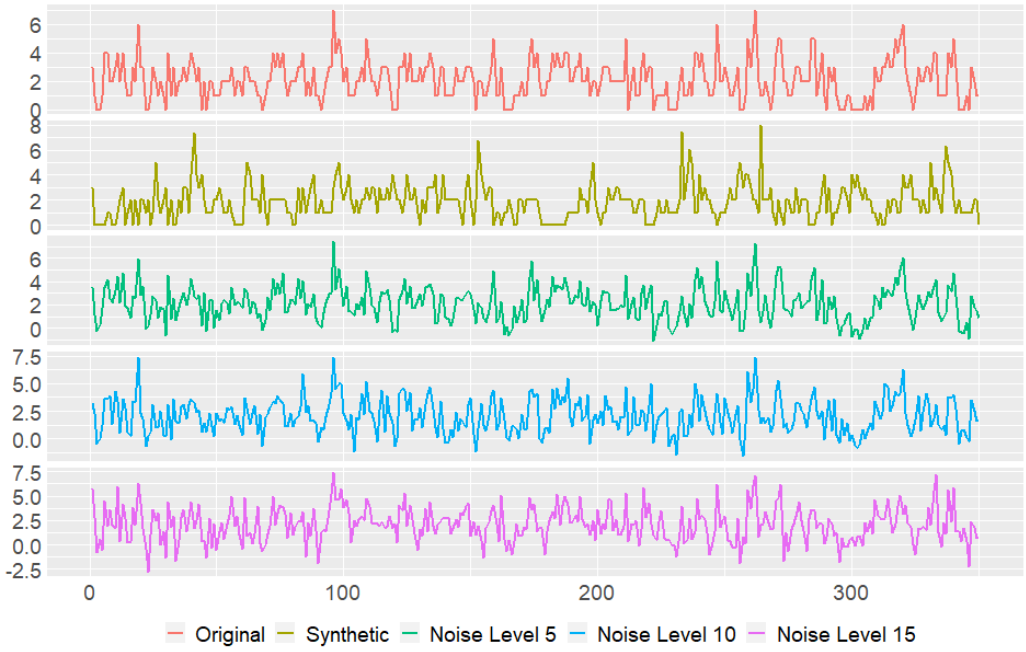


Figure 5.6: Comparison between Original vs Synthetic vs Noisy for **INAR** time series data.

Series	Mutual Information	Conditional Entropy
Synthetic	0.009005099	1.6878058
Noise Level 5	1.267379264	0.4294316
Noise Level 10	0.701395458	0.9954154
Noise Level 15	0.433923387	1.2628875

Table 5.2: Mutual Information and Conditional Entropy results for **INAR** time series data illustrated (in part) in Figure 5.6.

In summary, increasing the amount of perturbation in data could potentially lead to increased privacy measures. However, as seen in Figure 2.1, if applied above a certain threshold, perturbing data is a strategy that could render completely anonymous data. Such data would lose utility, since data characteristics have been diffused or removed. On the other hand, the present study indicates that the QG method and the synthesising process (InvQG) are capable of generating highly utility data while maintaining data privacy.

5.3 Other Privacy Metrics

In this section, for reference, we present extensions to MI and CE metrics proposed in the literature. These extensions aim to measure the same information detailed before in Section 5.1, but address specific challenges such as normalizing results between 0 and 1 and optimizing the metrics for use with time series data.

5.3.1 Normalized Mutual Information

Normalized Mutual Information (NMI) is defined as in Equation 5.4.

$$\mathrm{NMI}(X,Y) = \frac{I(X;Y)}{H(X)} = \frac{\sum_{x\in X}\sum_{y\in Y} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right)}{\sum_{x\in X} p(x)\log(p(x))} \tag{5.4}$$

The amount of mutual information $\mathrm{I}(X;Y)$ can vary significantly depending on the entropy $\mathrm{H}(X)$ present in the data. By normalizing to respect of $\mathrm{H}(X)$, NMI generates a measure between $[0,1]$ that is the *fraction* of mutual dependency, relative to the maximum possible mutual dependency, given the entropy's present in the data [59].

5.3.2 Mutual Information Under the Markov Assumption

Mutual Information Under the Markov Assumption [58] is an extension of MI that permits to capture temporal correlations within the data (e.g., X_t may depend on X_{t-1}). This extension is based on modeling the original and transformed time series into stationary first-order Markov chains ($f(t)$), and from these, calculate both the joint and marginal probabilities that are then applied in the Equation 5.5.

$$I(X||Y) = \sum_{i=1}^{T-1} I(X_{(i,i+1)}||Y_{(i,i+1)}) - \sum_{i=2}^{T-1} I(X_{(i)}||Y_{(i)}) \tag{5.5}$$

where $I(X_{(i,i+1)}||Y_{(i,i+1)}))$ is defined as:

$$\sum_{f_{(i)}}\sum_{f_{(i+1)}} p(f_{(i-1)})p(f_{(i)}|f_{(i-1)}))\log\left(\frac{p(f_{(i-1)})p(f_{(i)}|f_{(i-1)})}{p(X_{(i-1,i)})p(Y_{(i-1,i)})}\right) \tag{5.6}$$

and $I(X_{(i)}||Y_{(i)})$ is given in Equation 5.1.

5.3.3 Conditional Entropy Adapted for Sequences

In [59] the authors adapted the regular CE notion to the case of time series privacy protection by quantifying the uncertainty of a clear data point given a k length sequence of preceding noisy data points. It is defined as follows:

$$\begin{aligned} H(X_{k+1}|Y_k;Y_{k-1};\ldots;Y_1) &= \sum_{y\in Y_k} p(y)H(X_{k+1}|Y_k=y_k;Y_{k-1}=y_{k-1};\ldots;Y_1=y_1) \\ &= -\sum_{y\in Y_k} p(y)\sum_{x\in X_k} p(X_{k+1}=x|y)\log p(X_{k+1}=x|y) \end{aligned} \tag{5.7}$$

Similar to CE, the higher result, the higher the privacy protection.

5.3.4 Offline Conditional Entropy

Also proposed in [59], Offline Conditional Entropy (OCE) is a refined adaptation to the CE method. It is designed to assess the level of privacy protection in noisy data by analyzing both past and future data points in regard to a specific clear data point. For each clear data point, OCE uses a total of k neighboring noisy data points in the calculation. OCE is particularly useful in attack vectors where the adversary explores the full noisy data points as guiding information. OCE is defined as follows:

$$\begin{aligned} H(X_{k+1}|Y_1;Y_2;\ldots;Y_{2k+1}) &= \sum_{y\in Y_{2k+1}} p(y)H(X_{k+1}|Y_1=y_1;Y_2=y_2;\ldots;Y_{2k+1}=y_{2k+1}) \\ &= -\sum_{y\in Y_{2k+1}} p(y)\sum_{x\in X_1} p(X_{k+1}=x|y)\log p(X_{k+1}=x|y) \end{aligned} \tag{5.8}$$

Similar to CE, the higher result, the higher the privacy protection.

Due to time constraints, these metrics could not be applied in the context of this work.

Chapter 6

Conclusions and Future Work

A central piece in today's data ecosystem, time series analysis tools have become essential for the success of multidisciplinary applications such as climate and sustainability studies, economic analysis, monitoring of transportation and health. Time series may contain identifiable and sensitive information about individuals or organizations. Therefore, processing and extracting useful information from time series data while maintaining the privacy and utility of the data, is a current challenge.

Traditional privacy strategies such as data access control, cryptography and data perturbation are well-established for privacy-preserving data. However, these face challenges when applied to time series data due to naturally dynamic and complex environments, creating issues in scalability and maintaining data utility. In contrast, innovative synthetic data generation strategies are highly scalable and can be optimized to balance between privacy and utility. Though more complex to implement due to the need for specialized knowledge, as is the case for GANs, synthetic data generation methods offer robust privacy protection with higher data utility, making them a novel and preferable to preserving privacy.

In this work we have shown that network science could play a key role in analysing time series data whilst maintaining data privacy. we also have shown that by synthesizing new time series data from an existing complex network is a viable option for the time series privacy problem. Synthetic data has high utility and protects sensitive and private data, since resulting time series data is nearly voided of any mutual information from original data and extracting any sensitive insights of the original data from the synthetic data is an unfeasible task.

To achieve these conclusions we utilized an existing method in literature, Quantile Graph (QG), which maps an univariate time series into a complex network based on the concept of probability transition. First, we utilized a previously existing implementation for mapping a time series into a quantile-based graph. Afterward, we interpreted and implemented our own version for the reverse method, Inverse Quantile Graph (InvQG), which allows us to re-construct (synthesize) new time series data from a quantile-based graph. We then tested this method in various time series models and measured data utility using statistical measures. Afterward, we repeated the

process for complex networks and measured the utility of time series graphs using topological measures. Results from both empirical analyses have shown promising results by indicating that QG successfully captured unique behaviours and patterns of each time series model and the corresponding InvQG transposed them into synthetic time series data as well as on the corresponding synthetic graphs (generated using time series mappings on the synthetic data).

Additionally, we gathered and presented widely used statistical privacy metrics that aim to quantify common information between two time series (mutual information), and the uncertainty of a random variable when knowledge for another is given (conditional entropy). We then measured the degree of privacy provided by synthetic data by applying these privacy metrics. The results shown that a high degree of privacy is provided by the use of InvQG generated synthetic data. Retrieving any insights or identifying original observations would be an unfeasible task by simply analyzing the synthetic data.

Our findings suggest that although the QG and InvQG methods successfully capture key characteristics and dynamics from original data, the methods also have a slight tendency to reduce temporal correlations and introduce more variability in synthetic data. This is explained by the probabilistic nature of the analysis method when mapping time series into a quantile-based graph, and due to the naturally random process of synthesizing new data when reverting from graph to time series data. As future work, these issues could be addressed in order to improve not only the extra variability introduced in the synthesizing process, but also allow to capture higher order lags.

Another direction of future work is on multivariate time series data contexts, we intend to extend the InvQG method to multivariate time series data given recent advances in multivariate time series mapping methods. Additionally, an approach not pursued in this work is the use of complex networks representing a time series to extract insightful information. The use of complex networks for such goals could permit data privacy by masking private information by capturing essential data dynamics and masking original data observations.

We conclude that employing the quantile-based mapping method could be advantageous for applications where sharing data with investigators, third parties, or partners is necessary, but privacy regulations or concerns restrict or limit data sharing. By utilizing generated synthetic data, privacy can be ensured while maintaining the original data stored securely and privately.

Appendix A

Tables of results

A.1 *tsfeatures* Boxplots

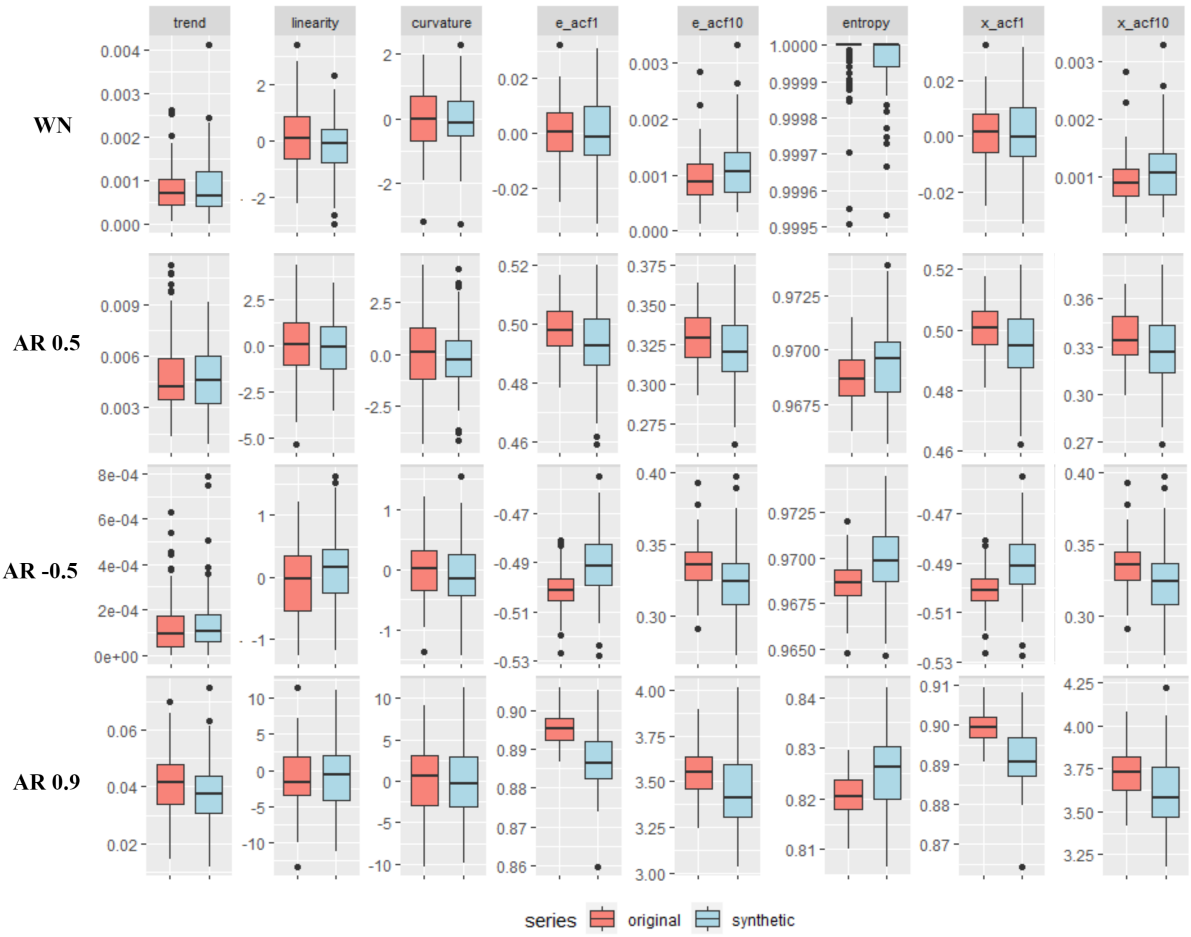


Figure A.1: Boxplot of *tsfeatures* metrics for **WN**, **AR 0.5**, **AR -0.5** and **AR 0.9** model.

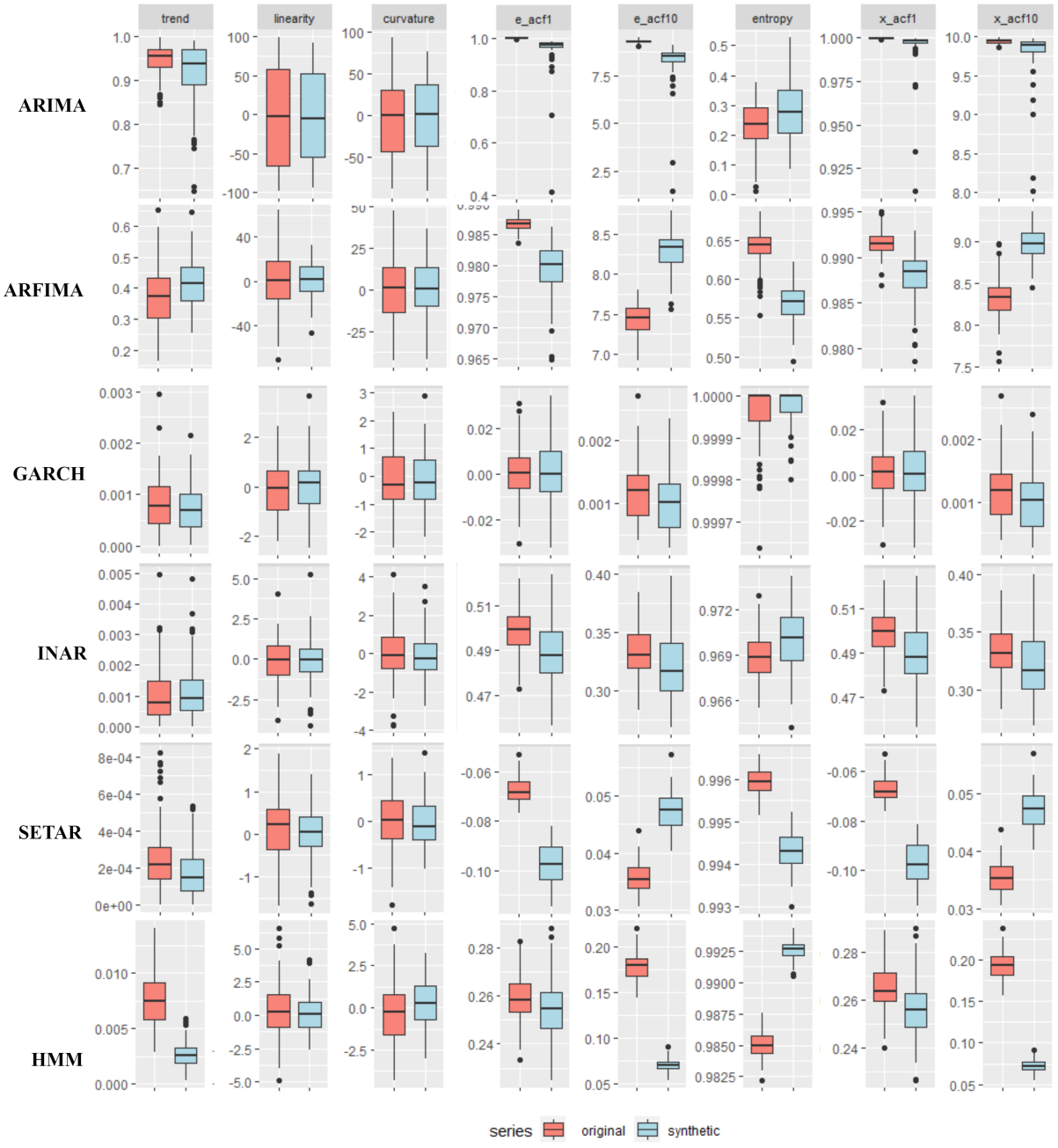


Figure A.2: Boxplot of *tsfeatures* metrics for **ARIMA**, **ARFIMA**, **GARCH**, **INAR**, **SETAR** and **HMM** models.

A.2 *NetF*: Network Features for Complex Networks

Table A.1: Network features, *NetF*, ($\bar{k}$, $\bar{d}$, S, C and Q) for complex networks mapped from original time series data and respective complex networks mapped from synthetic counterparts. The values correspond to the mean (and standard deviation) of the 100 instances of each time series models for each feature.

Series	Graph	Series	k	d	S	C	Q
WN	NVG	Original	5.9023 (0.0275)	6.7521 (0.2154)	166.3 (19.9208)	0.3711 (0.0027)	0.8587 (0.0093)
		Synthetic	6.0533 (0.048)	7.1889 (0.2245)	123.68 (25.7733)	0.37 (0.0047)	0.876 (0.013)
	HVG	Original	3.9961 (0.0008)	10.949 (0.4961)	287.63 (30.6221)	0.3346 (0.0011)	0.8688 (0.0075)
		Synthetic	3.9961 (0.0008)	10.9132 (0.4227)	288.23 (29.0492)	0.3345 (0.0011)	0.8683 (0.0071)
	QG	Original	2 (0)	0.0163 (0.0001)	3.04 (0.942)	0.8659 (0.0047)	0.0391 (0.0069)
		Synthetic	2 (0)	0.0171 (0.0001)	3.13 (0.9604)	0.8011 (0.0105)	0.047 (0.0088)
AR(1) 0.5	NVG	Original	6.7211 (0.038)	6.7005 (0.224)	173.43 (26.0641)	0.405 (0.0035)	0.8655 (0.0115)
		Synthetic	6.934 (0.0691)	7.143 (0.2537)	112.2 (30.0128)	0.4009 (0.006)	0.8924 (0.0169)
	HVG	Original	3.9957 (0.0008)	12.1519 (0.5226)	381.04 (42.559)	0.3579 (0.0012)	0.8665 (0.0081)
		Synthetic	3.9958 (0.0009)	11.9823 (0.4752)	374.32 (39.8421)	0.3572 (0.0013)	0.8662 (0.0082)
	QG	Original	2 (0)	0.0167 (0)	2.36 (0.5226)	0.8122 (0.0041)	0.1555 (0.0135)
		Synthetic	2 (0)	0.0174 (0.0001)	2.34 (0.4761)	0.7594 (0.0085)	0.16 (0.0143)
AR(1) -0.5	NVG	Original	5.6062 (0.0271)	6.7229 (0.1943)	155.74 (21.327)	0.3494 (0.0023)	0.8556 (0.0105)
		Synthetic	5.7473 (0.0445)	7.1891 (0.2473)	112.73 (24.068)	0.3492 (0.0045)	0.8764 (0.0157)
	HVG	Original	3.996 (0.0008)	10.8897 (0.4602)	209.32 (21.0511)	0.3227 (0.0009)	0.8808 (0.0073)
		Synthetic	3.9961 (0.0008)	10.761 (0.4256)	206.72 (22.6284)	0.3224 (0.0011)	0.8814 (0.0069)
	QG	Original	2 (0)	0.0167 (0)	1 (0)	0.8006 (0.0051)	0.0065 (0.0009)
		Synthetic	2 (0)	0.0174 (0.0001)	1 (0)	0.7398 (0.0114)	0.0065 (0.0011)
AR(1) 0.9	NVG	Original	9.1554 (0.1113)	6.4682 (0.21)	143.43 (26.7537)	0.4151 (0.0042)	0.8828 (0.0145)
		Synthetic	9.3806 (0.131)	6.8246 (0.2633)	91.93 (25.2063)	0.4106 (0.0075)	0.9125 (0.0171)
	HVG	Original	3.9942 (0.001)	18.858 (1.1034)	317.12 (66.61)	0.3811 (0.0012)	0.9062 (0.0098)
		Synthetic	3.9946 (0.0014)	16.8282 (0.7556)	317.84 (55.1065)	0.3795 (0.001)	0.9034 (0.0102)
	QG	Original	2 (0)	0.0203 (0.0002)	3.8 (0.8876)	0.7372 (0.0037)	0.3744 (0.0118)
		Synthetic	2 (0)	0.0213 (0.0003)	3.74 (0.7736)	0.7186 (0.0053)	0.3767 (0.0138)
ARIMA	NVG	Original	42.8148 (6.3144)	5.3717 (0.6518)	150.38 (62.2296)	0.1811 (0.0359)	0.7185 (0.0975)
		Synthetic	13.8526 (2.0888)	6.241 (0.8897)	226.41 (129.9969)	0.2424 (0.0377)	0.7623 (0.073)
	HVG	Original	3.9191 (0.034)	179.1994 (47.6796)	148.17 (21.7195)	0.4312 (0.001)	0.9678 (0.0023)
		Synthetic	3.9697 (0.0137)	72.5236 (22.3947)	119.16 (34.1102)	0.3703 (0.0074)	0.9598 (0.0137)
	QG	Original	2 (0)	2.3994 (0.7225)	7.37 (1.426)	0.4369 (0.1112)	0.3064 (0.0552)
		Synthetic	2 (0)	0.4494 (0.2934)	8.84 (1.7216)	0.6928 (0.046)	0.5542 (0.0566)

Table A.2: *(continued)* Network features, *NetF*, ($\bar{k}$, $\bar{d}$, S, C and Q) for complex networks mapped from original time series data and respective complex networks mapped from synthetic counterparts. The values correspond to the mean (and standard deviation) of the 100 instances of each time series models for each feature.

Series	Graph	Series	k	d	S	C	Q
ARFIMA	NVG	Original	16.2646 (0.4051)	5.9742 (0.2852)	123.22 (29.2587)	0.3626 (0.0106)	0.8719 (0.0224)
		Synthetic	13.2996 (0.4193)	6.3232 (0.3932)	96.71 (30.8258)	0.3265 (0.0124)	0.8794 (0.0256)
	HVG	Original	3.9861 (0.0034)	41.6997 (3.7249)	188.84 (38.2553)	0.4096 (0.0011)	0.9509 (0.0053)
		Synthetic	3.9884 (0.0039)	36.418 (3.3176)	146.18 (35.5461)	0.3847 (0.0013)	0.955 (0.0065)
	QG	Original	2 (0)	0.0576 (0.0074)	7.16 (1.2449)	0.7037 (0.0036)	0.5567 (0.0088)
		Synthetic	2 (0)	0.0571 (0.0071)	6.94 (1.1265)	0.7043 (0.0041)	0.5703 (0.0135)
GARCH	NVG	Original	6.007 (0.0314)	6.5631 (0.2027)	177.89 (27.07)	0.3642 (0.0037)	0.8451 (0.012)
		Synthetic	6.2157 (0.0743)	7.0561 (0.2662)	109.23 (28.893)	0.3594 (0.0081)	0.8773 (0.0196)
	HVG	Original	3.9958 (0.0008)	11.6686 (0.5006)	280.04 (34.9502)	0.3364 (0.001)	0.8709 (0.0096)
		Synthetic	3.9961 (0.0008)	11.0215 (0.4571)	286.08 (28.5366)	0.3344 (0.0011)	0.8717 (0.0067)
	QG	Original	2 (0)	0.0163 (0)	3.12 (1.225)	0.8637 (0.0043)	0.0376 (0.0062)
		Synthetic	2 (0)	0.0171 (0.0001)	3.44 (1.373)	0.8013 (0.0101)	0.0488 (0.0079)
INAR	NVG	Original	6.59 (0.0605)	6.6086 (0.5868)	164.33 (24.4483)	0.3574 (0.0042)	0.8722 (0.012)
		Synthetic	6.7662 (0.0949)	6.8495 (0.2244)	127.22 (35.2642)	0.3413 (0.0086)	0.876 (0.0202)
	HVG	Original	2.9211 (0.0088)	19.5755 (1.3157)	378.85 (60.7117)	0.1909 (0.0031)	0.9064 (0.0075)
		Synthetic	2.9445 (0.0126)	15.4378 (0.5833)	414.42 (42.176)	0.1944 (0.0034)	0.8957 (0.0059)
	QG	Original	0.16 (0)	0.0457 (0.0088)	93.7 (0.4606)	0.957 (0.0239)	0.1914 (0.0124)
		Synthetic	0.1614 (0.0051)	0.0518 (0.0161)	93.78 (0.4163)	0.9549 (0.0325)	0.1941 (0.0152)
SETAR	NVG	Original	5.7813 (0.0221)	6.6158 (0.1887)	136.71 (17.2042)	0.3561 (0.0028)	0.8684 (0.0085)
		Synthetic	5.9322 (0.0439)	7.0939 (0.2651)	107.63 (16.4744)	0.3568 (0.0046)	0.881 (0.0108)
	HVG	Original	3.9962 (0.0007)	10.354 (0.4161)	153.03 (23.3923)	0.3094 (0.0009)	0.9132 (0.0066)
		Synthetic	3.9963 (0.0008)	10.2739 (0.3724)	149.64 (21.4756)	0.3095 (0.001)	0.9127 (0.0072)
	QG	Original	2 (0)	0.0182 (0.0001)	2.31 (0.526)	0.7976 (0.0042)	0.0995 (0.0133)
		Synthetic	2 (0)	0.0191 (0.0002)	2.31 (0.5064)	0.7408 (0.0104)	0.0988 (0.0149)
HMM	NVG	Original	6.2162 (0.0388)	6.7011 (0.2855)	144.71 (23.0131)	0.374 (0.0035)	0.8753 (0.0118)
		Synthetic	6.3741 (0.0674)	7.0517 (0.2328)	124.75 (28.494)	0.3733 (0.0059)	0.8781 (0.0159)
	HVG	Original	3.6036 (0.0061)	13.2699 (0.7137)	293.85 (41.9587)	0.2992 (0.0015)	0.8996 (0.0078)
		Synthetic	3.637 (0.0109)	11.6603 (0.3992)	387.38 (37.3403)	0.3006 (0.0019)	0.8724 (0.0078)
	QG	Original	0.4202 (0.002)	0.029 (0.0016)	80.99 (0.1)	0.9825 (0.0065)	0.1243 (0.0115)
		Synthetic	0.4234 (0.0081)	0.0294 (0.0083)	80.91 (0.4516)	0.9698 (0.0111)	0.1259 (0.0127)

A.3 PCA of Network Features (*NetF*) for All Models

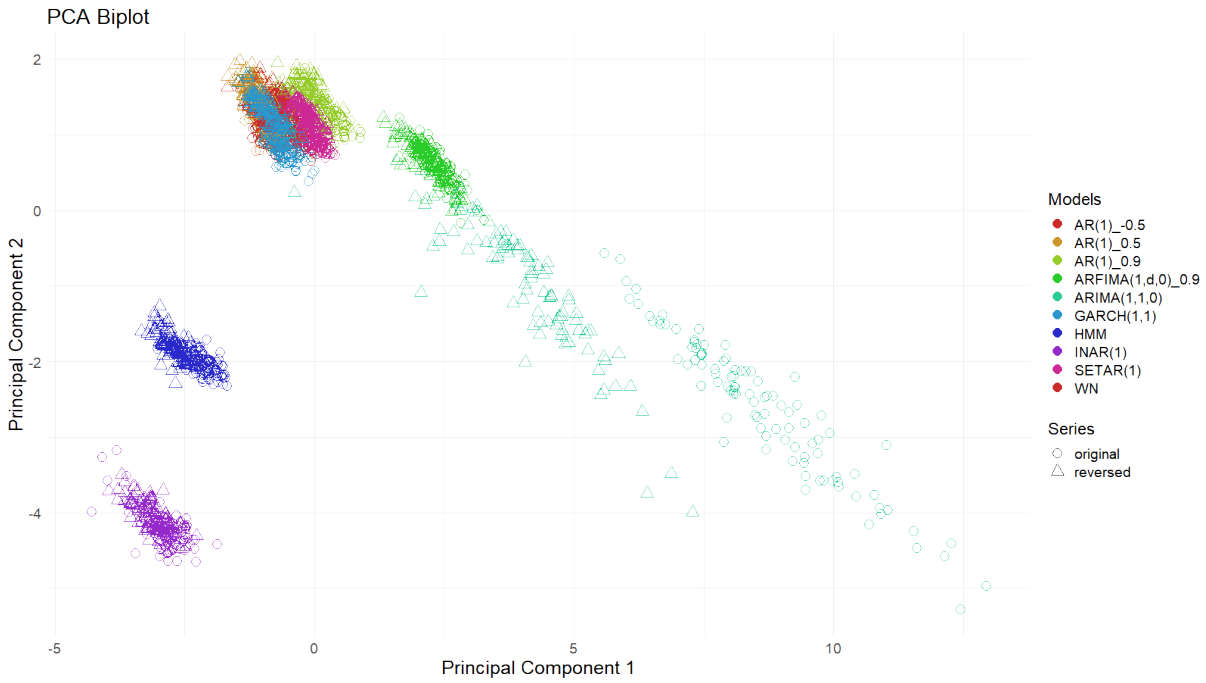
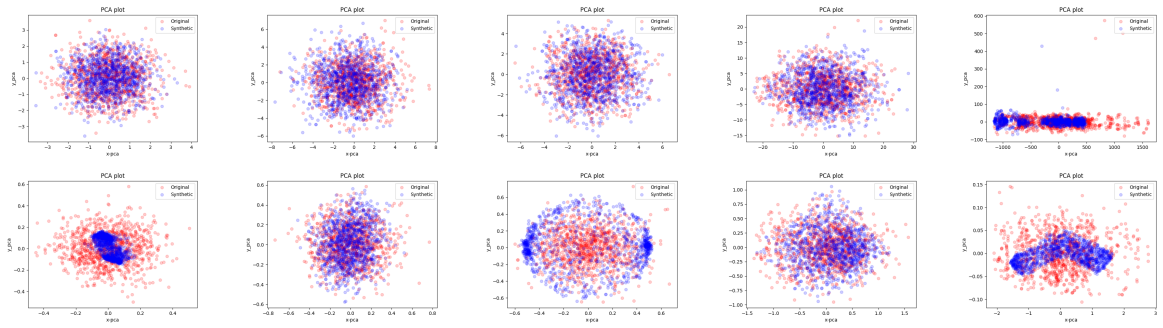


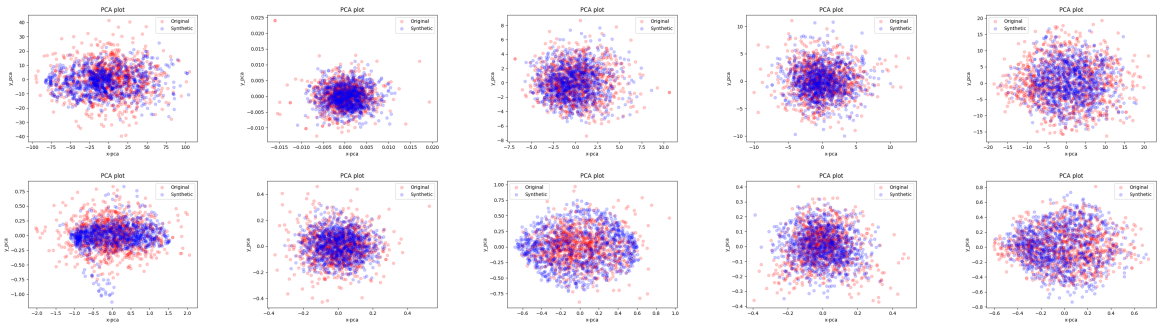
Figure A.3: PCA of network features (*NetF*) for all time series models.

A.4 TimeGAN vs Quantile-based Method



WN **AR 0.5** **AR -0.5** **AR 0.9** **ARIMA**

Figure A.4: PCA comparison between quantile-based mapping (first row) and TimeGAN (second row) methods for **WN** , **AR 0.5**, **AR -0.5**, **AR 0.9** and **ARIMA** models.



ARFIMA **GARCH** **INAR** **SETAR** **HMM**

Figure A.5: PCA comparison between quantile-based mapping (first row) and TimeGAN (second row) methods for **ARFIMA** , **GARCH**, **INAR**, **SETAR** and **HMM** models.

Bibliography

[1] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[2] Subir Halder and Thomas Newe. Enabling secure time-series data sharing via homomorphic encryption in cloud-assisted iiot. *Future Generation Computer Systems*, 133:351–363, 2022. ISSN: 0167-739X. doi:https://doi.org/10.1016/j.future.2022.03.032.

[3] Franklin Leukam Lako, Paul Lajoie-Mazenc, and Maryline Laurent. Privacy-preserving publication of time-series data in smart grid. *Security and Communication Networks*, 2021: 1–21, 2021.

[4] Mohammad Jabed Morshed Chowdhury, Alan Colman, Muhammad Ashad Kabir, Jun Han, and Paul Sarda. Continuous authorization in subject-driven data sharing using wearable devices. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 327–333. IEEE, 2019.

[5] Rohit Venugopal, Noman Shafqat, Ishwar Venugopal, Benjamin Mark John Tillbury, Harry Demetrios Stafford, and Aikaterini Bourazeri. Privacy preserving generative adversarial networks to model electronic health records. *Neural Networks*, 153:339–348, 2022.

[6] Adrian-Silviu Roman. Evaluating the privacy and utility of time-series data perturbation algorithms. *Mathematics*, 11(5):1260, 2023.

[7] Vicenç Torra and Guillermo Navarro-Arribas. Data privacy: A survey of results. *Advanced Research in Data Privacy*, pages 27–37, 2015.

[8] Muneeb Ul Hassan, Mubashir Husain Rehmani, and Jinjun Chen. Differential privacy techniques for cyber physical systems: a survey. *IEEE Communications Surveys & Tutorials*, 22(1):746–789, 2019.

[9] Muhammad Rizwan Asghar, György Dán, Daniele Miorandi, and Imrich Chlamtac. Smart meter data privacy: A survey. *IEEE Communications Surveys & Tutorials*, 19(4):2820–2835, 2017.

[10] Manos Katsomallos, Katerina Tzompanaki, and Dimitris Kotzinos. Landmark privacy: Configurable differential privacy protection for time series. In *Proceedings of the Twelfth ACM Conference on Data and Application Security and Privacy*, pages 179–190, 2022.

[11] Munshi Saifuzzaman, Tajkia Nuri Ananna, Mohammad Jabed Morshed Chowdhury, Md Sadek Ferdous, and Farida Chowdhury. A systematic literature review on wearable health data publishing under differential privacy. *International Journal of Information Security*, 21(4):847–872, 2022.

[12] Andrés Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, and David Irwin. Private memoirs of a smart meter. In *Proceedings of the 2nd ACM workshop on embedded sensing systems for energy-efficiency in building*, pages 61–66, 2010.

[13] Elaine Shi, HTH Chan, Eleanor Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *Annual Network & Distributed System Security Symposium (NDSS)*. Internet Society., 2011.

[14] Fabrice Benhamouda, Marc Joye, and Benoît Libert. A new framework for privacy-preserving aggregation of time-series data. *ACM Transactions on Information and System Security (TISSEC)*, 18(3):1–21, 2016.

[15] Erik Buchmann, Klemens Böhm, Thorben Burghardt, and Stephan Kessler. Re-identification of smart meter data. *Personal and ubiquitous computing*, 17:653–662, 2013.

[16] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using gans for sharing networked time series data: Challenges, initial promise, and open questions. In *Proceedings of the ACM Internet Measurement Conference*, pages 464–483, 2020.

[17] Sun-Kyong Hong, Kuldeep Gurjar, Hea-Suk Kim, and Yang-Sae Moon. A survey on privacy preserving time-series data mining. In *3rd International Conference on Intelligent Computational Systems (ICICS'2013)*, pages 44–48, 2013.

[18] Eoin Brophy, Zhengwei Wang, Qi She, and Tomás Ward. Generative adversarial networks in time series: A systematic literature review. *ACM Computing Surveys*, 55(10):1–31, 2023.

[19] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[20] Vanessa Freitas Silva, Maria Eduarda Silva, Pedro Ribeiro, and Fernando Silva. Time series analysis via network science: Concepts and algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(3):e1404, 2021.

[21] Lucas Lacasa, Bartolo Luque, Fernando Ballesteros, Jordi Luque, and Juan Carlos Nuno. From time series to complex networks: The visibility graph. *Proceedings of the National Academy of Sciences*, 105(13):4972–4975, 2008.

[22] Bartolo Luque, Lucas Lacasa, Fernando Ballesteros, and Jordi Luque. Horizontal visibility graphs: Exact results for random time series. *Physical Review E*, 80(4):046103, 2009.

[23] Andriana SLO Campanharo, M Irmak Sirer, R Dean Malmgren, Fernando M Ramos, and Luís A Nunes Amaral. Duality between time series and networks. *PloS one*, 6(8):e23378, 2011.

[24] UHWA Hewage, R Sinha, and M Asif Naeem. Privacy-preserving data (stream) mining techniques and their impact on data mining accuracy: a systematic literature review. *Artificial Intelligence Review*, 56(9):10427–10464, 2023.

[25] Yousra Abdul Alsahib S Aldeen, Mazleena Salleh, and Mohammad Abdur Razzaque. A comprehensive review on privacy preserving data mining. *SpringerPlus*, 4:1–36, 2015.

[26] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

[27] Fatoumata Dama and Christine Sinoquet. Time series analysis and modeling to forecast: A survey. *arXiv preprint arXiv:2104.00164*, 2021.

[28] Bryan Lim and Stefan Zohren. Time series forecasting with deep learning: A survey. *arXiv preprint arXiv:2004.13408*, 2020.

[29] Tânia Carvalho, Nuno Moniz, Pedro Faria, and Luís Antunes. Survey on privacy-preserving techniques for microdata publication. *ACM Computing Surveys*, 55(14s):1–42, 2023.

[30] Jing Qiu, Zhihong Tian, Chunlai Du, Qi Zuo, Shen Su, and Binxing Fang. A survey on access control in the age of internet of things. *IEEE Internet of Things Journal*, 7(6): 4682–4696, 2020.

[31] Simon Godik and Tim Moses. Oasis extensible access control markup language (xacml). *OASIS Committee Secification cs-xacml-specification-1.0*, 48, 2002.

[32] John Hughes and Eve Maler. Security assertion markup language (saml) v2. 0 technical overview. *OASIS SSTC Working Draft sstc-saml-tech-overview-2.0-draft-08*, 13:12, 2005.

[33] Alex Chiquito, Ulf Bodin, and Olov Schelén. Access control model for time series databases using ngac. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 1001–1004. IEEE, 2020.

[34] David Ferraiolo, Ramaswamy Chandramouli, Rick Kuhn, and Vincent Hu. Extensible access control markup language (xacml) and next generation access control (ngac). In *Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control*, pages 13–24, 2016.

[35] Michela Iezzi. Practical privacy-preserving data science with homomorphic encryption: an overview. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 3979–3988. IEEE, 2020.

[36] Omar G Abood and Shawkat K Guirguis. A survey on cryptography algorithms. *International Journal of Scientific and Research Publications*, 8(7):495–516, 2018.

[37] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4): 1–35, 2018.

[38] Haoxiang Wang and Chenye Wu. Privacy preservation for time series data in the electricity sector. *IEEE Transactions on Smart Grid*, 2022.

[39] Chiara Marcolla, Victor Sucasas, Marc Manzano, Riccardo Bassoli, Frank HP Fitzek, and Najwa Aaraj. Survey on fully homomorphic encryption, theory, and applications. *Proceedings of the IEEE*, 110(10):1572–1609, 2022.

[40] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[41] Jordi Nin and Vicenç Torra. Distance based re-identification for time series, analysis of distances. In *Privacy in Statistical Databases: CENEX-SDC Project International Conference, PSD 2006, Rome, Italy, December 13-15, 2006. Proceedings*, pages 205–216. Springer, 2006.

[42] Alphons Eggerth, Dieter Hayn, Karl Kreiner, Sai Veeranki, Heimo Traninger, Robert Modre-Osprian, and Günter Schreier. Patient record linkage for data quality assessment based on time series matching. In *dHealth*, pages 210–217, 2019.

[43] Nan Gao, Hao Xue, Wei Shao, Sichen Zhao, Kyle Kai Qin, Arian Prabowo, Mohammad Saiedur Rahaman, and Flora D Salim. Generative adversarial networks for spatio-temporal data: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(2):1–25, 2022.

[44] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Generating high-fidelity, synthetic time series datasets with doppelganger. *arXiv preprint arXiv:1909.13403*, 2019.

[45] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[46] Yusheng Huang, Xiaoyan Mao, and Yong Deng. Natural visibility encoding for time series and its application in stock trend prediction. *Knowledge-Based Systems*, 232:107478, 2021.

[47] Andriana SLO Campanharo, Erwin Doescher, and Fernando M Ramos. Application of quantile graphs to the automated analysis of eeg signals. *Neural Processing Letters*, 52(1): 5–20, 2020.

[48] Vanessa Freitas Silva, Maria Eduarda Silva, Pedro Ribeiro, and Fernando Silva. Multilayer quantile graph for multivariate time series analysis and dimensionality reduction. *International Journal of Data Science and Analytics*, pages 1–13, 2024.

[49] Zhiwei Hu, Tao Wu, Yunan Zhang, Jintao Li, and Longsheng Jiang. Time series anomaly detection based on graph convolutional networks. In *2020 2nd International Conference on Applied Machine Learning (ICAML)*, pages 138–145. IEEE, 2020.

[50] Vanessa Freitas Silva, Maria Eduarda Silva, Pedro Ribeiro, and Fernando Silva. Novel features for time series analysis: a complex networks approach. *Data Mining and Knowledge Discovery*, 36(3):1062–1101, 2022.

[51] Luciano da Fontoura Costa, Osvaldo N Oliveira Jr, Gonzalo Travieso, Francisco Aparecido Rodrigues, Paulino Ribeiro Villas Boas, Lucas Antiqueira, Matheus Palhares Viana, and Luis Enrique Correa Rocha. Analyzing and modeling real-world phenomena with complex networks: a survey of applications. *Advances in Physics*, 60(3):329–412, 2011.

[52] Albert-László Barabási. *Network Science*. Cambridge University Press, 2016.

[53] L da F Costa, Francisco A Rodrigues, Gonzalo Travieso, and Paulino Ribeiro Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in physics*, 56 (1):167–242, 2007.

[54] Rob Hyndman, Yanfei Kang, Pablo Montero-Manso, Mitchell O'Hara-Wild, Thiyanga Talagala, Earo Wang, Yangzhuoran Yang, Souhaib Ben Taieb, Cao Hanqing, D K Lake, Nikolay Laptev, J R Moorman, and Bohan Zhang. tsfeatures: Time series feature extraction, 2023. Accessed: January 15, 2024.

[55] S Raj Rajagopalan, Lalitha Sankar, Soheil Mohajer, and H Vincent Poor. Smart meter privacy: A utility-privacy framework. In *2011 IEEE international conference on smart grid communications (SmartGridComm)*, pages 190–195. IEEE, 2011.

[56] David Varodayan and Ashish Khisti. Smart meter privacy using a rechargeable battery: Minimizing the rate of information leakage. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1932–1935. IEEE, 2011.

[57] Taojun Wu, Yuan Xue, and Yi Chi. Preserving traffic privacy in wireless mesh networks. In *2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)*, pages 3–pp. IEEE, 2006.

[58] Weining Yang, Ninghui Li, Yuan Qi, Wahbeh Qardaji, Stephen McLaughlin, and Patrick McDaniel. Minimizing private data disclosures in the smart grid. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 415–427, 2012.

[59] Chris YT Ma and David KY Yau. On information-theoretic measures for quantifying privacy protection of time-series data. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 427–438, 2015.