Anonymously Publishing Univariate Time-Series: With focus on *(k,P)*-Anonymity

ERIK WIK

MASTER´S THESIS

DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY

FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Anonymously Publishing Univariate Time-Series: With focus on (k,P)-Anonymity

Erik Wik

Supervisors: Peter Blomqvist, peter.blomqvist@sony.com
Christian Gehrmann, christian.gehrmann@eit.lth.se

Master's thesis carried out at Sony Mobile Communications AB

This dissertation is submitted for the degree of
Master of Science

Electronic and Information Tech. December 2019

Abstract

Anonymizing time-series data is a demanding task since there might be a lot of private information which can be inferred which might not initially be obvious. Therefore, it can be desired to anonymize with some established privacy guarantee. However, there is always a trade-off to be made between privacy and minimization of information loss. In this thesis, an investigation is made into how to protect univariate time-series. The main focus is on publishing anonymized time-series from individual users, but methods for anonymizing aggregate time-series and the removal of sensitive data is also investigated. This is done in order to find a wider understanding of how a blood glucose related database can be anonymized. The main achievement of this thesis is the implementation of PC-KAPRA, a novel extension to the KAPRA algorithm, which publishes data under (k, P)-anonymity. The results show that PC-KAPRA offers a large improvement in retaining pattern information compared to KAPRA, and publishes data which could be considered qualitative useful information.

Table of contents

Chapter 1

Introduction

This thesis will focus on the anonymization of univariate time-series data. Time-series is a type of data which is collected over time, and is highly useful for various types of analytics. In particular it can be used for prediction. From an anonymization point of view, a problem with time-series data is that non-obvious private and identifying data could be inferred from it. An example of this is shown in a competition hosted by Kaggle, where people could be identified by using accelerometer data from smart-phones [1]. In order to deal with this, it is desired to use established privacy measures which can guarantee that no such inferences can be made. Therefore, a large section of this thesis is a theoretical background on *k*-anonymity and differential privacy, which are two widely used anonymity measures.

The thesis was carried out at *Sony Mobile Communications AB*. The original purpose of the thesis was to find a way to anonymize time-series data in order to deal with problems which arose from the General Data Protection Regulation (GDPR), with special focus on datasets containing blood glucose values collected in combination with a glucose monitoring smart-watch. However, ironically, no blood glucose dataset could be used for testing the results because of GDPR privacy regulations. Therefore, albeit confusing, the main algorithm PC-KAPRA, will be show-cased on other kinds of univariate time-series.
Since this is not a thesis in law, there will not be any specific analysis into how the anonymization fulfills the requirements from GDPR. Rather, the focus will be on privatization and specifically anonymization of univariate time-series data from a more algorithmic perspective.

[1]https://www.kaggle.com/c/accelerometer-biometric-competition

1.1 Background

For a better understanding of the field of privacy it is important to first distinguish the different terminology. A popular definition of data privacy is made by A. Westin in "Privacy and Freedom" (1967) [1], "Privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others." Data privacy covers the scope of collection of personal data to usage and publishing. In this thesis the focus lies on the privacy regarding data publication. Data privacy sometimes gets mixed up with information security. Information security refers to protection of data by encryption or other methods, preventing unauthorized access to a database [2]. This differs from data privacy which focuses on how the data is collected, used and published, with regards to the prevention of leaking sensitive information. In any data processing of personal information it is essential that both information security and data privacy methods are used.

Data anonymization, or data de-identification, exists as a subfield within the field of privacy, dealing specifically at protecting an individual or entity by removing or transforming identifying information in a database, while retaining useful information [3]. Published anonymized data should not be allowed to be linked to any one individual or entity, regardless of auxiliary information. A common method for anonymization are methods aimed at establishing k-anonymity. This anonymity measure guarantees that each data provider is indistinguishable from at least $k-1$ others after data publishing [2]. A more detailed description of this measure will be given in section (2.1) along with (k,P)-anonymity, which is a similar measure but more aimed towards time-series.

A lighter anonymization requirement is *pseudo-anonymization*, where no such guarantee regarding auxiliary information needs to be made [3]. On December 17, 2009, Netflix released potentially sensitive and personal data from 480.000 users in connection with a public competition aimed at finding an improved film-recommendation algorithm [4] [5]. This data was published after first replacing the participant's identification with an anonymous unique numerical identifier. However, shortly after the competition, Arvind Narayanan and Vitaly Shmatikov were able to reveal the identity of target participants, by comparing the published data with auxiliary data from IMDB. This privacy breach eventually led to a lawsuit directed

[2]https://www.techopedia.com/definition/26464/data security

[3]https://www.investopedia.com/terms/d/data anonymization.asp

[4]https://money.cnn.com/galleries/2010/technology/1012/gallery.5$_{d}ata_{b}reaches/index.html$

[5]https://privacylaw.proskauer.com/2009/12/articles/invasion-of-privacy/netflix-sued-for-largest-voluntary-privacy-breach-to-date/

towards Netflix.

Due to this re-identifying risk, the focus of this thesis will lie on anonymization rather than pseudo-anonymization, i.e, an individual's data should be protected against an adversary, regardless of auxiliary information.

The main problem of this thesis is strongly inspired and motivated from an issue faced by Sony, and a multitude of other companies, in the time after the GDPR. Before going into the exact problem at hand, a short introduction of GDPR will be offered together with a privacy scenario inspired from a meeting at Sony. This will help to illuminate the problem of information sharing between entities after the introduction of GDPR.

1.1.1 GDPR

GDPR, or General Data Protection Regulations was introduced in 2016, offering protection to natural persons with regards to the processing of personal data[3]. The "Right of Erasure"; a right which legally enforces an entity to erase all personal data from an individual upon request, was introduced. This right severely complicates the process of sharing personal data from a natural person between entities.

Figure (1.1) illustrates how *personal data* might propagate between entities. There might be a *product* initiated by a *company*, which then receives various *contributions* from *external entities*. A natural prerequisite for these *contributions* could be access to the *personal data* gathered through the *product*. Parallel to the propagation of *personal data* there is a propagation of *permissions* from the *users* for the processing of this data.

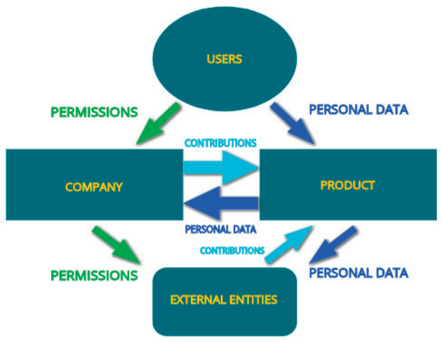


Fig. 1.1: Propagation of permissions and personal data

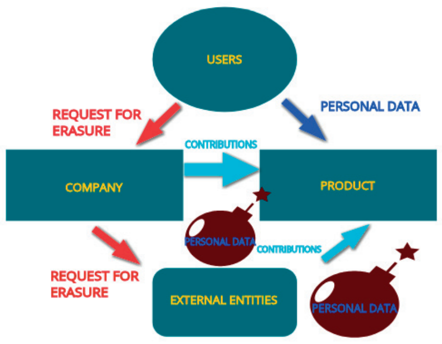


Fig. 1.2: Propagation of request for erasure.

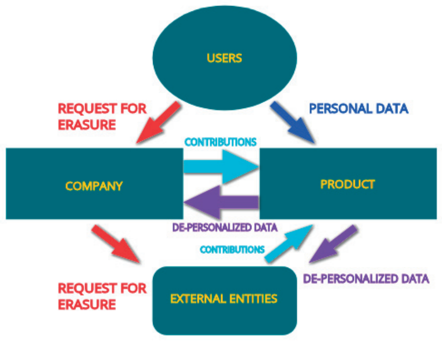


Fig. 1.3: Propagation of request for erasure and de-personalized data.

When a *request for erasure* is invoked, the following scenario schematized in figure (1.2) occurs. Where both the *company* and all the *external entities* have to remove all the gathered *personal data*. This could become problematic for the *company* for two main reasons. Firstly, the personal data often contain valuable information which can be used in various forms of analytics. Secondly, there could be legal issues concerning any of the *external entities* failing to comply to the *request for erasure*.

To avoid these problems, we can avoid the propagation of *personal data* all together and instead anonymize the data first, rendering it *de-personalized*, yielding the schematized scenario in figure (1.3). Both the *company* and the *external entities* are now allowed to keep the *de-personalized data*. If the personal data can be scrupulously anonymized while still retaining valuable information, then this could be an essential part for all data processing.

In addition to the rights given to individuals over their personal data, there are certain prohibitions on the processing of special categories of personal data [3]. These categories include data concerning health and biometric data for uniquely identifying a natural person.

These kinds of sensitive inferences could be found in many datasets. For instance, one can detect smoking by analyzing acceleration data from a wrist band [4]. Having that sort of information might not be of interest for an entity, and having those sensitive parts removed would offer both more freedom for data processing and a better privacy for the individual.

When dealing with problems in this category the goal is not to de-personalize the data, rather to clean the data for any sensitive information, possibly predefined by the data provider. It will therefore not be retained through a "request for erasure", but would nevertheless both offer a better alignment with GDPR and strengthen the trust between data provider and data receiver.

A brief overview of GDPR and how it relates to the incentive to anonymize data and removal of sensitive information has now been given. However, for the rest of the thesis, the focus is on the publishing of *de-personalized* data.

1.1.2 Problem description

Anonymizing personal data can be done in various ways, depending on the type of data being processed. The focus in this thesis is centered around blood glucose data, which can be classified as a univariate time-series or data stream. In this scenario a data surveyor, *Sony Mobile Communications AB* in this case, has access to a database with data collected from various users in the format *(user ID, blood glucose, time)*, see table (1.1), and wants to find a way to publish the data anonymously. The idea is that this type of data will be collected through a smart-watch device, which potentially can collect other type of data as well, such as GPS and activity level.

Due to a lack of, and legal complications accessing real blood glucose data from multiple subjects, the implemented algorithms in this thesis will be showcased on other types of univariate time-series data.

	Blood glucose (mmol/l)				
User ID	13:00	13:15	13:30	13:45	$\cdots$
1	7.5	7.4	7.3	7.6	$\cdots$
2	7.9	7.7	7.6	7.8	$\cdots$
3	5.4	5.2	5.3	5.5	$\cdots$
4	9.3	9.1	9.4	9.6	$\cdots$
5	6.4	6.7	6.6	6.6	$\cdots$
6	8.3	8.0	8.5	8.4	$\cdots$

Table 1.1: Table of blood glucose progression over time in a small population.

In figure (1.4) the data from an individual with diabetes is illustrated, where the blood glucose concentration is measured every 5 minutes over the span of roughly 2 days.

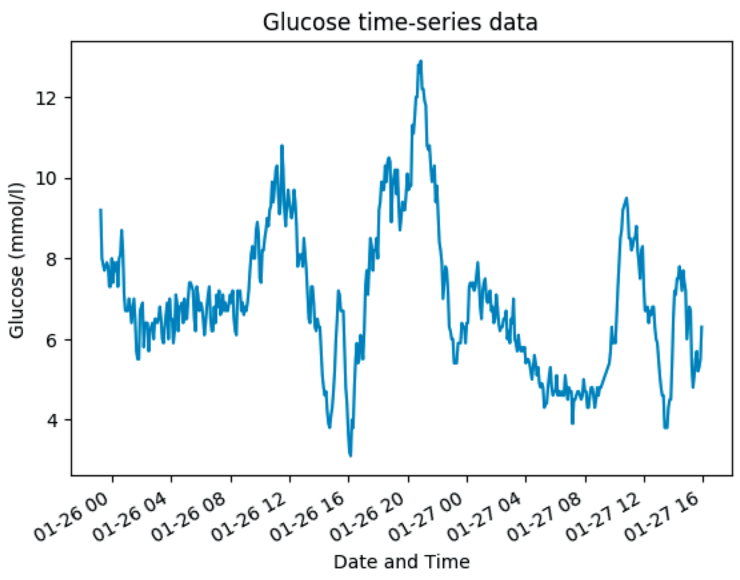


Fig. 1.4: Glucose data collected every 5 minutes over a time-span of roughly two days. Blood glucose measured in $mmol/L$.

The problem lies in anonymizing the time-series while still retaining as much information as possible, i.e anonymously publishing the data as closely to the original data as possible. A naïve approach could be to simply remove unique identifiers and replace them with a unique ID. This could however have severe privacy consequences in analogue with the aforementioned Netflix film recommendation system issue. A simple motivation why this might be the case is given in the following example.

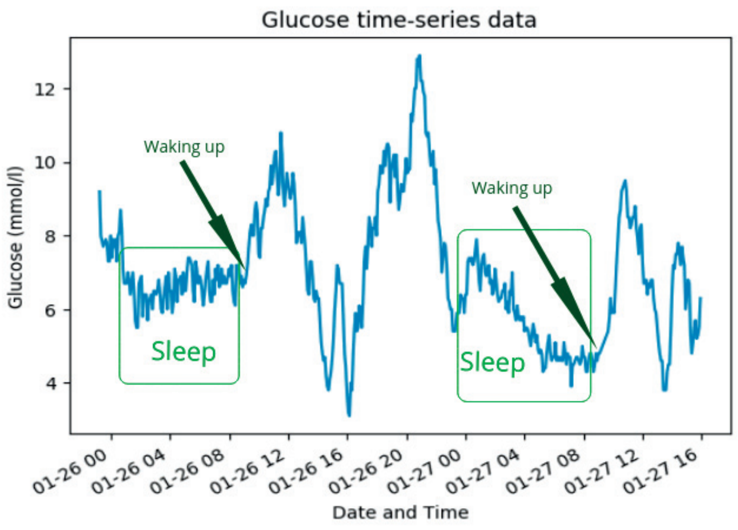


Fig. 1.5: Inferences

In figure (1.5) a reasonable guess regarding the sleeping patterns based on the blood glucose values of the data provider has been made. One could, from these guesses, draw the conclusion that a relatively stable blood glucose value is correlated to sleep, followed by a sudden dramatic increase which is correlated to waking up. Now imagine if the data was collected over multiple days and some adversary had access to auxiliary information regarding a populations sleeping patterns. In that case, there would be a risk that the adversary could combine both sources of information in order to deduce which individual the blood glucose time-series belongs to.

These inferences were made by a quick glance at the data. One could imagine that there might be more hidden inferences, allowing more sources of auxiliary data to be used for uniquely identifying the data provider. Therefore, the goal of this thesis is to find methods which can guarantee that the published data will remain anonymous, by some guarantee, regardless of auxiliary information.

1.2 Scope

In order to deal with the problem posed in the previous section, the focus will be widened from strictly looking at blood glucose data to time-series data of similar character, i.e., univariate time-series from individual data providers. This is mainly due to a lack of data samples, but also driven by a wish for a more generic solution; a method which is not restricted to only function on the unique characteristic of blood glucose data. However, the applicability of various algorithms will evaluated with regards to its applicability on blood glucose data. The goals of this thesis are the following.

- Anonymization of individual time-series. This is the main goal of the thesis. Here, methods for publishing time-series based on the *k*-anonymity measure [5], will be investigated. This is because *k*-anonymity offers a clear-cut privacy guarantee and simple implementation, that allows for the possibility of retaining the entire time-series provided by an individual.

- Anonymization of aggregate time-series. Here an investigation into methods aimed at publishing anonymized aggregate time-series, using differential privacy [6], has been made. Differential privacy is offers a rigorous mathematical privacy guarantee.

- Removal or transformation of sensitive information. This is important to give a small investigation, as it offers an alternative when anonymization is not possible.

Notice that algorithmic implementations are only done for the main goal which aims at anonymizing individual univariate time-series.

1.3 Methodology

In this thesis, the problem has been addressed through a combination of theoretical studies of privacy preserving technologies, new algorithm development and experimental evaluations.

Initially, a literature study was conducted in order to get as better understanding of the field of privacy. Special focus was aimed towards differential privacy, *k*-anonymity and general approaches regarding the problem of anonymizing time-series.

When a basic understanding of the field was established, a search for state-of-the-art algorithms aimed at anonymizing univariate time-series was conducted. Here, the investigation eventually got narrowed down to k-anonymity based methods. This was done in parallel with finding methods for anonymizing aggregate time-series using differential privacy and general methods aimed at the removal or transformation of sensitive data.

A promising algorithm using a measure called (k,P)-anonymity was found and further investigated. Inspired by this algorithm, a new algorithm was designed in this thesis using the same (k,P)-anonymity measure.

Finally, both algorithms were implemented and tested on databases containing univariate time-series and evaluated.

1.4 Outline

The outline of this thesis is given as follows:

Chapter 2 provides definitions and descriptions of privacy measures based on k-anonymity and selected state-of-the-art anonymization algorithms applicable on the main problem described in section (1.1.2). Furthermore, there will be an theoretical background explaining differential privacy, and a presentation of privacy methods used on aggregate time-series data and removal of sensitive data.

Chapter 3 contains a detailed description of the novel PC-KAPRA algorithm which is aimed towards establishing (k,P)-anonymity.

Chapter 4 provides the results from the PC-KAPRA and the benchmark KAPRA algorithm on different time-series datasets, by measuring information loss and pattern loss.

Chapter 5 contains a discussion based on the results from the previous chapter, along with possible improvement and ways to utilize algorithms based on (k,P)-anonymity. In addition, suggestions for future directions of anonymity and privacy are given.

Chapter6 conclude the thesis by stating the most important findings with respect to the original purpose.

Chapter 2

Privacy measures

This chapter aims at giving the reader an understanding of k-anonymity and other closely related measures, such as l-diversity and (k,P)-anonymity [7][2], as well as differential privacy. Following the theoretical background, a set of selected state-of-the-art methods aimed at the privatization of univariate time-series will be presented, and analyzed with regards to the problem stated in section (1.1.2).

In order to have a discussion regarding privacy it is essential to establish a privacy measure. The ones most typically used are k-anonymity and differential privacy, but there are more approaches depending highly on what kind of methods are applied. For instance, the goal could be to protect data against a certain type of attack. In that case the attained defence against that attack could be used as a measure.

2.1 k-anonymity

The k-anonymity measure was first introduced by Sweeney and Samaranti in 1998 [5] as an effort to introduce a scientific guarantee for an individual's privacy while trying to maintain practically useful data. An individual in a dataset is said to be k-anonymous if it is indistinguishable from at least $k-1$ other individuals in the same dataset.

In the database we are considering there are three types of information: Identification(ID), Quasi-identifiers (QIs) and sensitive information (A_s).

ID	Name	Nationality	Height	disease
1	Jane	Australia	170	cancer
2	John	Sweden	185	diabetes
3	Johanna	Sweden	165	diabetes
4	Jack	USA	176	chicken pox
5	Jenny	Germany	165	heart related
6	Jenny	Sweden	167	no illness

Table 2.1: An imaginary database, with ID, QIs(Name, Nationality, Height) and sensitive information(disease).

In the table (2.1) data is portrayed showing the identification, nationality, height and disease of a small population. The identification is anonymous in its nature, while the name, nationality and height is considered QIs and the disease in considered sensitive information. In the current scenario every individual can be uniquely identified. This can be changed by removing information, which can be seen in table (2.2) where 2-anonymity is achieved.

ID	Name	Nationality	Height	disease
1	Ja*	*	170-176	cancer
2	Jo*	Sweden	165-185	diabetes
3	Jo*	Sweden	165-185	diabetes
4	Ja*	*	170-176	chicken pox
5	Jenny	*	165-167	heart related
6	Jenny	*	165-167	no illness

Table 2.2: Imaginary database, protected under 2-anonymity/

However, the sensitive information of having diabetes is inferable, regardless of John and Johanna's 2-anonymity guarantee. This is a fundamental flaw of *k*-anonymity, where there is no protection against the possibility protecting individuals from releasing sensitive information. One way of dealing with this issue is to introduce *l*-diversity.

2.1.1 l-diversity

The privacy definition *l*-diversity was introduced as a way of addressing the privacy problems *k*-anonymity has with regards to leaking sensitive information [7]. The definition of *l*-diversity is the following:

A group is l-diverse if it contains at least l “well-represented” values for the sensitive attribute. A table is l-diverse if every group is l-diverse.

From the example on the previous table (2.2), one can notice that the group with Swedish nationality is 1-diverse, since there is only one "well-represented" sensitive attribute, namely "diabetes". In order to achieve 2-diversity one could introduce another member to the table: Jonathan from Sweden with height 175 and no illness.

ID	Name	Nationality	Height	disease
1	Ja*	*	170-176	cancer
2	Jo*	Sweden	165-185	diabetes
3	Jo*	Sweden	165-185	diabetes
4	Ja*	*	170-176	chicken pox
5	Jenny	*	165-167	heart related
6	Jenny	*	165-167	no illness
7	Jo*	Sweden	165-185	no illness

Table 2.3: 2-diversity

If now a random Swede in table (2.3) would be analyzed, one would not with certainty know if they have diabetes or no illness due to the newly inserted sensitive attribute.

The anonymization procedure discussed thus far is regarding non-temporal data. The problem a bit more complicated once time is also considered.

2.2 *k*-anonymity for time-series

In this thesis the problem lies in using k-anonymity on univariate time-series, which differs from static data portrayed in table (2.1) in that, typically, many more dimensions are introduced. Take for instance the pseudo-anonymized table (1.1) where blood glucose is measured every 5 minutes, in which the number of dimensions could be considered equal to the number of time-stamps:

In order to establish k-anonymity, the data must be transformed so that the QIs of each row is identical to at least $k-1$ other rows. This can be done by introducing an Anonymization Envelope AE in which the QI values A_i for each time-series are contained within a value range $R_i = [r_i^-, r_i^+]$, belonging to at least $k-1$ other time-series. An example of this is shown in table (2.4) where 2-anonymity is established by simply combining every second row.

		Blood glucose(mmol/l)				
ID	Group ID	13:00	13:05	13:10	13:15	···
1	1	7.3-8.0	7.2-7.9	7.0-7.8	7.3-8.1	···
2	1	7.3-8.0	7.2-7.9	7.0-7.8	7.3-8.1	···
3	2	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	···
4	2	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	···
5	3	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	···
6	3	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	···

Table 2.4: 2-anonymized table of blood glucose progression over time in a small population, by creating anonymization envelopes.

Notice that the values have been slightly perturbed, this is essential and the reason why can easiest be understood by looking at Group 2. Only two people belong to this group and the difference in blood glucose values are so substantial that one can easily deduce that the lower bound belongs to one individual and the higher to the other, thereby revealing each individual's entire time-series, if not perturbed.

Creating AEs is synonymous to creating clusters and then representing each time-series in the cluster identically. Therefore, clustering can be seen as a fundamental part of establishing k-anonymity.

2.2.1 Clustering

By the definition of the k-anonymity it is easily observed that clustering is central to any algorithm establishing the anonymity measure. Clearly there are "good" and "bad" ways of establishing k-anonymity. In table (2.4) one can see that the naive k-anonymity clustering could be improved with regards to minimizing information loss, by combining ID 3,5 and 4,6 rather than 3,4 and 5,6. This would lead to smaller envelope.

In time-series clustering, the most important elements are representation, distance measure, prototype extraction function, clustering algorithms, and cluster evaluation [8].

Time-series representation refers to how data should be represented before proceeding with further clustering. In table (2.4) the data is represented as raw-data, but it could be transformed to any other type more beneficial to clustering. In this step it is common to do some kind of dimensionality reduction, in order to deal with the high dimensionality of time-series [9][2].

The distance measure is used as a metric to compute the distance between time-series. One of the simplest and common ways of computing distances for a univariate time-series is using the Euclidean distance [10].

The prototype extraction function decides how the minimization processes of clusters should look like. In some sense, choosing the prototype is choosing the centre of the cluster. This is commonly chosen as the average sequence or medoid sequence of the set, where medoid is the sequence closest to the average of the set [10].

The specific clustering algorithm can then be chosen with regards to the representation, distance measure and prototype extraction, in addition to, a desired balance between accuracy and complexity. One of the most prominent clustering algorithm is the partitioning based *k*-means clustering [10], which makes *k* groups from *n* unlabeled objects. This is also closely related to the problem faced in *k*-anonymity, since we would like to partition the data into different groups.

In a paper published by Johnsana et al., they used performed a dimensionality reduction by transforming the data by random projection [9]. They then moved on to using *k*-means clustering, using the Euclidean distance function. Where *k*-means clustering is a method where the centre for each cluster is the average of the respective cluster. They called this methodology *CAT* (Clustered k-anonymization of Time Series Data), which is based on first creating clusters and then for each cluster establishing *k*-anonymity using the *ARX* anonymization tool[1]. Although this method is very applicable on the problem considered in this thesis, it can be improved by also taking patterns into consideration. This is done by Shou et al, and the (k,P)-anonymity which they introduced [2].

2.2.2 (k,P)-anonymity

The formation of envelopes or clusters could be based on one metric such as the euclidean distance or pattern similarity. However, as shown by Shou et al. [2], focusing on either of these metrics could give a large information loss regarding the other metric. The solution they

[1]https://arx.deidentifier.org/

present for this problem is the (k,P)-anonymity measure, which establishes P-anonymity by requiring P identical patterns within each k-anonymous envelope. The pattern and data then gets published in two different formats. A pattern can be defined as a vector $\mathbf{p}$ of features f.

$$\mathbf{p}(s) =< f(s)_1, \ldots, f(s)_n >$$

where f is a correlation function transforming time-series data (s) from value space to some other space.

$$f : (A_1, \ldots, A_n) \rightarrow Y$$

This way of forming patterns is quite generic, and with $f(s)_i = A_i$ the original time-series is returned.

The (k,P)-measure is by Shou et al. [2] defined as:

Definition: (k,P)-anonymity: Let T be a database of time-series, and $A_1, \cdots, A_n$ being the QI attributes. A published database T^* is said to satisfy (k,P)-anonymity, if T^* meet the following two requirements:

- k-requirement: Each Anonymization Envelope $AE = (R_1, \ldots, R_n)$, where R_i defines the value range at each time-step, appears at least k times in the entire database;
- P-requirement: Consider any k-group G of time-series having the identical anonymization envelope, if any time-series $s \in G$, there are at least $P-1$ other time-series in G having the same QI pattern representation as s.

These requirements gives a protection against linkage attacks based on auxiliary knowledge on QI attribute values, and the whole QI pattern representation or parts of it. Where a linkage attack is an attack aimed at finding a link between auxiliary information and information contained in the anonymized dataset. The $(k,P$-anonymity gives the privacy breach probability $P_{breach}[s] = 1/P$ for any target time-series s.

An example of (k,P)-anonymized data is given in table (2.5), with $k = 4$ and $P = 2$. In the table, one can see four time-series with identical value range, this satisfies $k = 4$. Then within each group, there is for each pattern two identical representations which satisfies $P = 2$. For instance, in Group ID 1, two time-series have the patterns *aabbcc* and *bbccaa*. The patterns are represented as strings, which will be explained in more detail in the next chapter.

		Blood glucose(mmol/l)					
ID	Group ID	13:00	13:15	13:30	13:45	$\cdots$	$\mathbf{p}(\mathbf{s_{ID}})$
1	1	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	$\cdots$	*aabbcc*
2	1	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	$\cdots$	*aabbcc*
3	1	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	$\cdots$	*bbccaa*
4	1	5.2-9.6	5.0-9.4	4.9-9.7	5.2-10.0	$\cdots$	*bbccaa*
5	2	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	$\cdots$	*bbaaca*
6	2	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	$\cdots$	*bbaaca*
7	2	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	$\cdots$	*bbccaa*
8	2	6.1-8.5	6.4-8.3	6.3-8.8	6.2-8.8	$\cdots$	*bbccaa*

Table 2.5: Table of blood glucose progression published under (4,2)-anonymization.

In the paper by Shou et al.[2], they suggested the algorithm KAPRA (*k and P- reinforced anonymization*) to establish (k,P)-anonymity. This algorithm has an outline which is reminiscent of the clustering outline given in the previous section (2.2.1). It starts by a preprocessing step of transforming the data to a pattern and attribute value representation, along with introducing two distance metrics for the two representations which are mainly used to evaluate the information loss. Then follows an algorithm which join together time-series with identical pattern representation into groups of size larger than P. These groups are then clustered together into groups of size k. In such a way, they manage to establish (k,P)-anonymity.

The KAPRA algorithm is quite applicable on the problem considered in this thesis. It offers a good way to maintain information on both patterns and attribute values, however the requirement of identical pattern representation which is inserted during the formation of P-groups may lead to considerable information loss. In particular if one wants to publish data with fine granularity or if the data has a large variance. Therefore, in the next chapter we will introduce PC-KAPRA (*pattern clustering - k and P-reinforced anonymity*), which is similar to the KAPRA algorithm, but applies a version of a k-means clustering method in forming the pattern groups. In that chapter also the steps used in the KAPRA algorithm will be clarified as both algorithm are identical except for the pattern clustering part.

Before looking more specifically into the method which aims at establishing (k,P)-anonymity, a look into differential privacy will be made. The differential privacy measure offers a methodology more aimed towards aggregate data. This could be a useful in combination with k-anonymity based methods in order to create a wider anonymity cover on a database generated by, for instance, a glucose measuring smart-watch.

2.3 Differential privacy

Differential privacy is the golden standard for database privacy and can be seen as a promise from data holder to data subject that they will not be affected by allowing their data to be studied, regardless of adversaries and auxiliary information; your data cannot be tracked back to you [6]. Establishing differential privacy allows for analysis of the entire population, but keeps individuals private. Therefore, the area of application will differ from the main problem as described in section (1.1.2), and instead be focused on publishing related data as univariate aggregate time-series. This could be data such as: number of people with diabetes in the database, average activity level and population trajectories based on inferences (see figure (1.5 and (2.2). The model we are considering is depicted in figure (2.1), where a trusted server receives data from individual users and then publishes a univariate aggregate time-series. Theoretically, for some algorithms, this could also be done without a trusted server [11].

In the initial definition, introduced by Dwork et al. in [12], they consider a trusted server that holds a database of sensitive information. This server allows for a query function f to be used to extract information, perturbed in order to establishing differential privacy. The definition of differential privacy given below is a reformulation of the original definition given by Dwork et al.

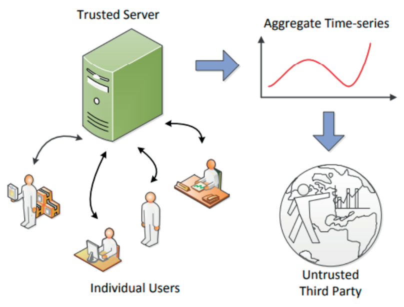


Fig. 2.1: Publishing aggregated time-series.

(Differential privacy). Given two datasets $D_1, D_2 \in \mathbf{D}$ differing in at most one record and a query function $f : \mathbf{D} \to \mathbb{R}$. Algorithm f satisfies ε-differential privacy if for any $y \in \mathbb{R}$ it holds that:

$$P[f(D_1) = y] \leq e^{\varepsilon} P[f(D_2) = y]$$

The implication is that if adversaries wants to find if the data from a data subject is contained in a dataset or not, they can at best only reach a guarantee described by e^{ε}. This definition will allow for data subjects to share their data for analytics, without being afraid of revealing any personal information. This ε is usually defined as the *privacy budget*, used as a measure of the upper limit for allowed queries posed on the database. Queries are questions posed to and answered by the database.

In this thesis the interest lies in publishing univariate time-series data. What this boils down to, with regards to differential privacy, is to pose the same query at multiple times to a database and then publish the results. Fulfilling the goal of publishing each individual time-series as accurately as possible, as described in section (1.1.2) is not possible, because differential privacy deals with aggregate data from a group rather than individuals. However, it can be used together with k-anonymity for publishing different types of anonymous data from a database. Where differential privacy could be used on anonymizing the aggregate data and k-anonymity used in order to keep utility from individual time-series. For that reason follows a common methodology used to reach differential privacy.

2.3.1 Reaching differential privacy

One of the most common methods to reach differential privacy is through the Laplace mechanism [12], which adds noise in the form of a zero mean Laplace distribution to the answer from a real or vector valued query. The magnitude of the Laplace noise added depends on ε and the sensitivity S of a query function f. A query function is a wide range of functions. One commonly used query is the counting query:

$$f(E) = |E|, E \in D \tag{2.1}$$

Where E is an event, and $|E|$ describes the number of occurrences of this event in the database D. The sensitivity can then be defined as:

$$S(f) = \max_{D_1 D_2} |f(D_1) - f(D_2)| \tag{2.2}$$

Where D_1 and D_2 are two adjacent databases. The sensitivity would vary depending on what kind of query is posed. For a counting query it would be 1, since that is the maximum counting difference between two adjacent databases, while the sensitivity would differ for other statistical queries such as mean, variance or median.

The Laplace mechanism outputs F; the summation of the answer to the original query and the added Laplace noise in the form

$$F(D) = f(D) + L(\lambda)$$

where $L(\lambda)$ is a random variable drawn from the Laplace probability distribution

$$Lap(x) = \frac{1}{2\lambda} exp(-\frac{|x|}{\lambda})$$

It is shown by Dwork et al. [6] that adding Laplace noise with magnitude $S(f)/\varepsilon$ to the answer of a query guarantees ε-differential privacy. Note that a smaller ε, i.e better privacy, leads to a larger Laplace noise.

Repeated queries on a database will lead to an accumulated privacy loss, according to the composition theorem [6]. If each query offers ε_i-differential privacy we get the total ε-differential privacy loss as:

$$\varepsilon = \sum_{i=1}^{n} \varepsilon_i \tag{2.3}$$

where n is the number of queries. This is one of the fundamental problems when it comes to offering a differential privacy guarantee to aggregated time-series. A typical characteristic of time-series is that it is correlated between time-steps; knowing that 400 people in the database had diabetes yesterday gives information about how many people have diabetes today. Therefore, using the average from repeated queries yields more details about the true value, and soon the privacy budget could be satiated.

2.3.2 Event- and user-level privacy

When it comes to privacy regarding time-series, there are two fundamental notions: event-level and user-level privacy [13]. Event-level privacy aims at hiding the presence or absence of one event. In regards to collecting data regarding blood glucose, this could for instance be an individual reporting whether or not they have diabetes, a disclosure typically only needed to be given once.

In [13], Dwork et al. introduced the *web counter problem* which described the problem of publishing the number of events that has happened so far, while obscuring at each time-step whether or not an event occurred. This is a useful problem statement regarding the problem of tracking the number of users with diabetes, while concealing the response of a single user. Dwork et al. continues to present an algorithm which yields a solution to this problem which enjoys ε-differential privacy [13]. This algorithm could be useful to implement in order to, for instance, to track the number of people with diabetes that are using the smart-watch mentioned in section (1.1.2). By repeated queries, the exact number of people with diabetes could be found, but the issue lies, from a privacy point of view, in knowing when someone submitted their answer.

User-level privacy aims at hiding the presence or absence of a user in a database. This is a generalization of event-level privacy, expanded to all the events provided by a user throughout the time-series. In this scenario a user is expected to continuously share information with the database. If the time-series has time-lag dependent correlations it will also be affected by the composition theorem (2.3).

An algorithm that adaptively chooses the frequency of queries posed to a dataset in real-time, is proposed by Fan and Xiong [11]. This algorithm aims at maximizing the privacy budget by performing less queries when the time-series is static. They add noise by using the Laplace Mechanism, and then use a Kalman filter and a PID controller to dynamically alter the query frequency. In this way they can fill the privacy budget ε in a way that yields higher utility. In this model they require prior knowledge of the length of the time-series. This methodology could be used well to track the activity level among the smart-watch users. Less activity during the night would yield a more static data, resulting in a lower sampling rate.

Cao and Yoshikawa introduced a measure called l-trajectory privacy, in order to deal with the problem of privatizing infinite trajectory streams [14]. A 3-trajectory could for instance be $Home \rightarrow gym \rightarrow work$. By ensuring l-trajectory by differential privacy, an adversary with prior knowledge of an individual's location cannot infer any of the other locations they have been situated. Cao and Yoshikawa aims in their model to continuously publish the number of people in each location at every time-stamp.

The data in figure (1.5) could be transformed to instead be represented by trajectories. Where every users state moves from either being *asleep* or *awake* along with possibly more states if more inferences are made possible, see figure (2.2).

By using the methodology by Cao and Yoshikawa, it would then be possible to publish the count of users being in different states. This could be interesting in order to see the general activity patters of the users in real-time.

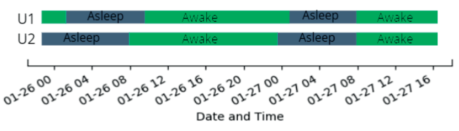


Fig. 2.2: Blood glucose transformed to represent inferences.

When dealing with historical data, there is more room to work with the data holistically. One approach proposed by Rastogi and Nath [15] is to transform a time-series of length n by Fourier transform and saving only the k most prominent Fourier coefficients which can approximately reconstruct answers to n queries posed on the time-series. These k-coefficients are then perturbed in order to establish differential privacy, which allows for a smaller amount of noise than if all n data points were perturbed.

Fan and Xiong; and Cao and Yoshikawa considered algorithms collecting data in real-time which can be interesting in many scenarios[14][11]. Consider, for instance, real-time collection of location data from vehicles, in order to monitor the traffic situation or the collection of health data for early prevention of an epidemic. In the case of the glucose data considered in this thesis, the applicability of real-time collection is less prominent. Therefore, the approach by Rastogi and Nath on historical data is more applicable, in which they have the possibility of lower the noise by using Fourier transforms.

An introduction to the anonymization measures k-anonymity and differential privacy has now been given. However, there are scenarios where anonymization can be very difficult, and not even necessary. In the next section a short introduction to the protection of sensitive information which offers a good privacy alternative to anonymization methods.

2.3.3 Protection of sensitive information

In this section the focus is shifted from anonymity to removal of sensitive information. Here the goal is no longer to keep an individual anonymous, but rather to guarantee that nothing harmful can be deducted from their data. Often times, this could be sufficient privacy guarantee for a data provider.

Sensitive information comes in any shape and form, depending on what kind of data we are dealing with. There might be a scenario where the speed of drivers are tracked.

These drivers might not want to reveal situations where they are over-speeding, but have no problems revealing general driving patterns. In that case one could, by adding noise, create a reasonable deniability on whether over-speeding occurred or not. In a study by Papademitriou et al. [16] they added noise to the entire time-series through wavelet transforms, which was shown to be strong against attacks from adversaries with auxiliary information about the true value at certain time-points or the underlying structure of the time-series.

This approach has is simple and can easily be implemented on univariate blood glucose time-series considered in this thesis. It could for instance be used for hiding the exact moment waking up, insulin injection, time of eating, etc. However, it loses a lot of information in the process of adding noise to the entire dataset.

Given a good understanding of the data it is possible to transform certain parts of the data that contains sensitive information. In two similar studies by Malekzadeh et al. [17] and Saleheen et al. [18] they considered multidimensional data where sensitive information such as smoking could be deduced. Where other approaches often use randomization, filtering or mapping [17], they implement a replacement method which transforms time sequences of sensitive data to non-sensitive data. This approach is superior at retaining desired data and protecting against inferences of sensitive data.

The replacement approach is however quite difficult to implement, due to the non-trivial problem of finding sensitive inferences in blood glucose data. Neither of the approaches give a guarantee that they have actually been able to remove all the sensitive data contained in the dataset. However, if a better understanding of blood glucose data is found in the future, then this method could become very valuable.

Chapter 3

Implementing (k,P)-anonymity for time-series

This chapter will describe the steps necessary for the implementation of the (k,P)- anonymity based algorithm PC-KAPRA (*Pattern clustering - k and P-reinforcing anonymization*). PC-KAPRA is identical to KAPRA in all aspects except the pattern formation, in which it offers a novel approach in applying a k-means inspired clustering method in order to satisfy the P-requirement introduced in section (2.2.2).

The motivation behind developing the alternative method of PC-KAPRA came from noticing the large pattern loss occurring in the KAPRA algorithm due to its requirement of identical pattern representation which will be explained closer in section (3.2). The first step in loosening up this requirement, was to move from identical patterns to similar patterns. Therefore, the approach in PC-KAPRA is to cluster similar patterns together and then fulfilling the P-requirement by changing their pattern representation to that of the average of the cluster.

3.1 Clustering

In this section a presentation of the three major steps of the clustering process will be given. These include, representation, distance metric and algorithm. The aim of the clustering is to create groups with similar patterns, in order to satisfy the P-requirement.

3.1.1 Data representation

Data representation is the first step of a clustering algorithm. Depending on the raw data and the desired output, there is a variety of possibilities of how the data can be represented [8]. In both KAPRA and PC-KAPRA the value attributes is represented as raw data, and the patterns using SAX(*Symbolic aggregate approximation*). The rest of this section will describe how to transform the data to a SAX-representation, in which the first step is transforming the raw data into a PAA (*Piecewise aggregate approximation*) representation.

The basic idea of PAA is to reduce the dimensionality of the time-series by splitting them into equal sized segments and replacing the values in each segment with their combined average value [19]. Assuming we have the time-series $Y = Y_1, Y_2, \cdots, Y_n$ and we want to reduce it to the *PAA*-representation $X = X_1, X_2, \cdots, X_m$ where $m < n$, the algorithm can be summarized by the following formula:

$$X_i = \frac{m}{n} \cdot \sum_{j=(n/m)(i-1)+1}^{(n/m)i} Y_j$$

Where m is the number of segments and will from here on be referred to as the PAA-size.

The *PAA*-representation X is then z-normalized before using SAX to transform the data into a symbolic sequence. The transformation is done by splitting the value axis in segments, each with an asserted symbol assigned to the X_i values within this segment. Each symbol should be equiprobable, which is easy to construct since z-normalized time-series are Gaussian distributed [20]. The quantiles β_i, which are the points which divides the Gaussian distribution into equiprobable parts, can easily be found once you have decided on the *SAX*-level, i.e. the number of symbols used for the SAX-representation. This is done by using the Probit function, which is the inverse of the cumulative distribution function for a normal distribution $\mathcal{N}(0,1)$, see figure (3.1)

The probit function does not have a closed form, but a numeric approximation can be found using *ppf*(Percent point function, which is another name for the probit function) from the *scipy.stats.norm* package in python. The quantiles are then found in the following way:

$$\beta_i = ppf(\frac{i}{SAX\text{-level}}) \tag{3.1}$$

where $i \in [1, \cdots, (SAX - \text{level}) - 1]$. A *SAX*-alphabet of size 10 is reached by dividing the x-axis in 10 equal sized intervals, i.e computing quantiles $\beta_1, \cdots, \beta_9$. The SAX-representation

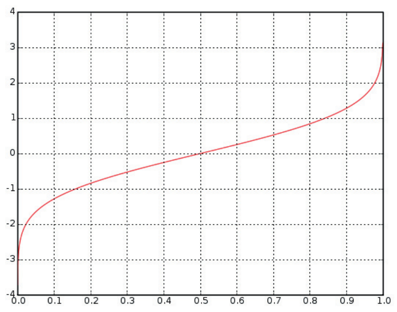


Fig. 3.1: Probit function [1]

used in this thesis will use the Latin alphabet, starting from the letter **a** and continuing until the predetermined *SAX*-level.

The SAX-representation creates patterns which are easily tunable. For instance, if the data is collected daily over the time span of a year, we could tune PAA to represent the daily, weekly or monthly patterns. Similarly, we can tune the SAX granularity depending on the variability of the data.

When the data representation has been chosen, the next step is to define a distance metric in the space created by these representations. In this thesis, each data representation has its own distance metric.

3.1.2 Distance metric

To do any clustering, one first needs to define a meaningful distance metric. The distance metric used for patterns is MINDIST [21]. It is a metric used specifically for SAX represented data, and is defined as follows.

$$MINDIST(SAX(s^{(i)}), SAX(s^{(j)})) = \sqrt{\frac{n}{PAAsize}} \sqrt{\sum_{k=1}^{PAAsize} dist(SAX(s_k^{(i)}), SAX(s_k^{(j)}))} \quad (3.2)$$

where n is the original length of the time-series $s^{(i)}$ and $s^{(j)}$, and the *dist* function implemented using a look-up table where each cell is defined as

$$cell_{(r,c)} = \begin{cases} 0 & \text{if } |r-c| \leq 1 \\ \beta_{max(r,c)-1} - \beta_{min(r,c)} & \text{otherwise} \end{cases} \tag{3.3}$$

where β are the quantiles, r are rows, c are columns, and the size of the table depends on the alphabet size. A table of alphabet cardinality 4 would have the following look.

	a	**b**	**c**	**d**
a	0	0	0.67	1.34
b	0	0	0	0.67
c	0.67	0	0	0
d	1.34	0.67	0	0

Fig. 3.2: Look-up table used in the MINDIST function for a *SAX*-level 4 representation. Here the alphabet cardinality is 4, and the distances are computed using equation (3.3).

Other distance measures could be used, yielding similar results. The importance lies in having a distance measure that makes sense when it comes to similar patterns yielding a small functional value and dissimilar yielding a large value. The MINDIST is however relatively quick, because using the look-up table has only the time complexity of O(1), yielding the final complexity of O(*PAA*-size).

In order to form k-groups the Instant Value Loss (IVL) proposed by Shou et al [2] will be used, which is defined as:

$$IVL(E) = \sqrt{\sum_{i=1}^{n} (r_i^+ - r_i^-)^2/n} \tag{3.4}$$

Where E is an envelope, n is the length of the envelope, and r_i^+ and r_i^- is the largest and smallest values respectively at each time step i within the envelope. This distance measure makes sense, considering the ultimate objective of creating small envelopes that still fulfill the (k,P)-requirement.

Both the MINDIST and IVL distance measures are later used for the evaluation of the information loss regarding the patterns and attribute values. Now that distance measures are defined, it is possible to look at at a clustering method which can achieve this.

3.1.3 Clustering algorithm

In creating a clustering algorithm fitting for the task of partitioning the SAX represented pattern data into clusters of size larger or equal to P, it was suitable to use a version of the k-means clustering algorithm [22]. Observe that k here is referring to the number of clusters, and not the requirement of indistinguishable time-series, as in k-anonymity. The clustering algorithm is ran in a *PAA*-size-dimensional space, where each *SAX*-representation of the time-series, can be seen as a data point in this space. The distance metric used is the MINDIST function.

The clustering k-means clustering algorithm is a partitioning based algorithm, and consist of three phases: *Initialization*, *Assignment* and *Update*. In the initialization phase, k cluster centers, i.e. centroids, are placed at random. Every *SAX*-representation of time-series, i.e data point is then assigned to the closest centroid in the *Assignment* phase. When all data points have been assigned a centroid, the centroid position gets updated in the *Update* phase to the average of all data points within its cluster. The location change of the centroids could lead to some data points now being closer to another centroids. These points would like to change clusters. Therefore, the two last steps, *Assignment* and *Update*, is repeated until convergence. A two-dimensional version of this can be seen in figure (3.3). In this case, only two assignment steps are needed for convergence, since no elements changed clustering in the second assignment step.

A problem with the k-means clustering method is knowing how many initial centroids to create. In the variation of k-means implemented in the *PC-KAPRA* algorithm, the number of clusters are allowed to change dynamically by removing all empty clusters, like the removal of the green centroid in figure (3.3). This allows for more flexibility when choosing the initial number of centroids, since a high number of centroids would also cause more of them to be removed. This cluster removal step is, for the best of my knowledge, novel to this thesis.

The average value, i.e. *SAX*-representation, within each group is computed by doing a 1 to 1 mapping from the symbolic representation to a set of consecutive integers of the same length as the *SAX*-level, computing the average, rounding to the closest integer and then inverse transforming it back to the *SAX*-representation. For instance, the *SAX*-representations *ACAD* and *DADA* would be transformed to $1-3-1-4$ and $4-1-4-1$ respectively. The

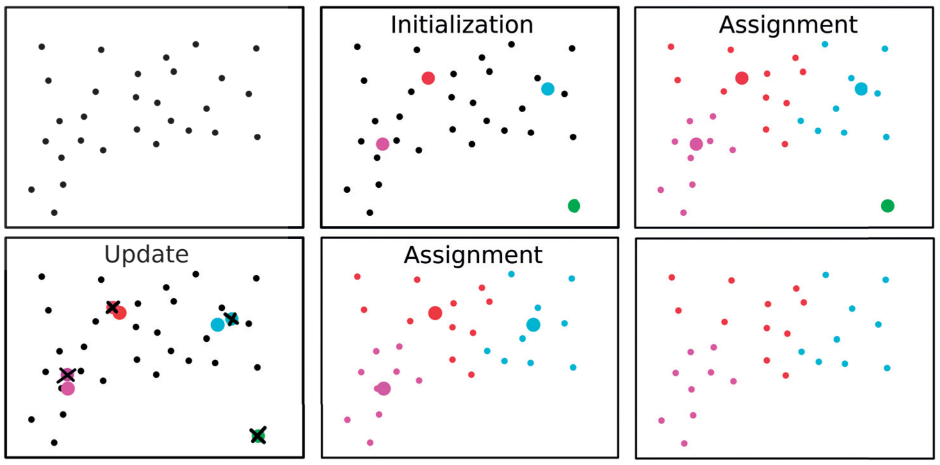


Fig. 3.3: Example of the kmeans clustering algorithm in two dimensions, step-wise moving from top-left corner, and ending with convergence in the bottom-right.

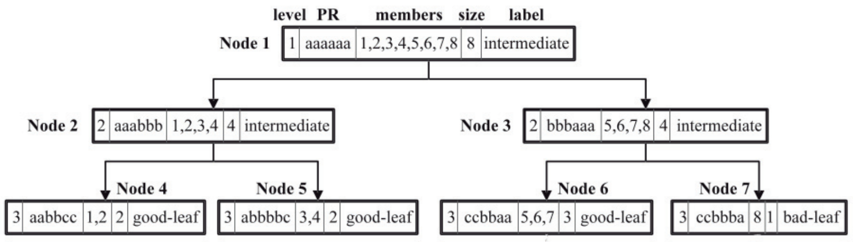
average between them would be computed to be $3-2-3-3$, which would make *CBCC* the "average" *SAX*-representation.

This pattern clustering step is the main difference between the method *PC-KAPRA* and *KAPRA*, where it was a requirement of equality between *SAX*-representation. Exactly how KAPRA requires equality, is explained in the next section.

3.2 KAPRA: P-groups formation

The KAPRA algorithm is in most part identical to the PC-KAPRA algorithm, but differs in how the pattern groups, i.e. *P-groups*, are formed. The schematics of the algorithm is shown in figure (3.4). The algorithm starts by creating a SAX-representation with *SAX*-level $= 1$, and puts all data within the same group. The *SAX*-level is then increased, and new groups are formed based on having identical pattern representation and *SAX*-level. This is done repeatedly for each group until the group size is smaller than $2 * P$ or until some predetermined max *SAX*-level has been reached. That level will be referred to as *max-level*.

Once the splitting is finished, the result is a number of *good-leaves* and *bad-leaves*, representing groups succeeding and failing the *P*-requirement. In order to avoid suppression of the *bad-leaves*, they are put through a recycling process. This process goes through the list of *bad-nodes* and tries to merge at least *P* of them together in order to fulfill the *P*-requirement. For two *bad-nodes* to be merged they must fulfill the requirement of identical *SAX*-representation and *SAX*-level. The process start from the predetermined *max-level* and moves down to *SAX*-level $= 1$, where all time-series have identical *SAX*-representation. This ensures that a maximum $P - 1$ time-series will be suppressed.



level PR members size label
Node 1 1 aaaaaa 1,2,3,4,5,6,7,8 8 intermediate
Node 2 2 aaabbb 1,2,3,4 4 intermediate
Node 3 2 bbbaaa 5,6,7,8 4 intermediate
Node 4 3 aabbcc 1,2 2 good-leaf
Node 5 3 abbbbc 3,4 2 good-leaf
Node 6 3 ccbbaa 5,6,7 3 good-leaf
Node 7 3 ccbbba 8 1 bad-leaf

Fig. 3.4: P-groups formation

The problem that occurs in the *P-groups* formation in the KAPRA algorithm, is that a relatively long *SAX*-sequence will lead to the creation of many *bad-nodes*. Furthermore, many of the *bad-nodes* will be given a *SAX*-level $= 1$, due to large number of possible *SAX*-representation, and the difficulty in finding equality between them. In the next chapter, both the KAPRA and PC-KAPRA algorithm will be used to anonymize on various datasets. It will in the next chapter be shown that much more information can be retained by using the PC-KAPRA method for *P-groups* formation.

3.3 Algorithmic implementation

In this section, each step of the *PC-KAPRA* algorithm will be explained. Keep in mind that most of these steps are identical in the KAPRA algorithm, except for the pattern clustering phase. As illustrated in figure (3.5), the algorithm is divided into three phases. First, the data is divided into groups with similar patterns in the *Pattern clustering phase*. These patterns are then broken into smaller *P-groups* of size between $[P, 2P]$ in the *Deconstruction phase*, and then finally, in the *Group formation phase*, pieced together into *k-groups* of size larger or equal to k, which can contain various pattern groups, or just one, depending on the size of k and P.

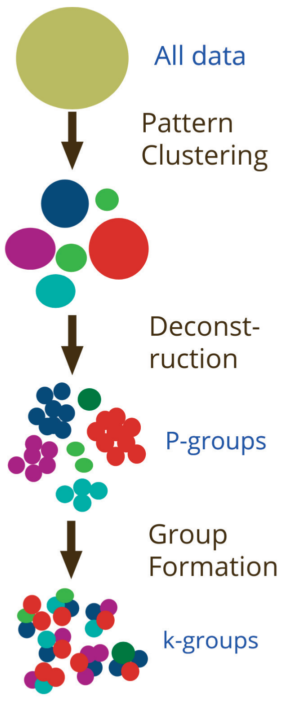


Fig. 3.5: Schematic of the PC-KAPRA algorithms and the three phases: *Pattern clustering*, *Deconstruction* and *Group formation*. The colors represent unique patterns.

The *Pattern clustering phase* starts with some preprocessing, in which the attribute values will be represented as raw data, and patterns as *SAX*-representations with a predetermined *SAX*-level, as explained in the previous section. Clusters are then formed using the k-means clustering method. Some of these clusters might have less than P time-series inside them. The time-series within those clusters will then be moved to the closest cluster which fulfills the P-requirement. Once every time-series belongs to a cluster which fulfills the P-requirement, their *SAX*-representation will be changed to that of the cluster center. The complexity of the k-means algorithm is $O(k*n*t)$, where k is the number of clusters, n is the number of data points and t is the number of iterations. The number of iterations until convergence is quite low.

After the *Pattern clustering phase*, there could be very large groups with identical SAX-representation. The goal of the *Deconstruction phase* is to partition these groups into smaller

P-groups with size $[P, 2P]$. The partitioning method used is called *top down greedy search* [23]. The *top down greedy search* creates a binary partitioning aimed at minimizing IVL. The heuristics of the method is to create a *seed* in each subgroup by taking the two time-series that differs the most with regards to the IVL. The rest of of the time-series are than paired with the *seed* which minimizes IVL. This partitioning is done recursively until the size of the groups are smaller than $2P$, resulting in the formation of what is referred to as *P-groups*. The complexity of this algorithms is $O(n^2)$, with n referring to number of data points.

The algorithm then forms the k-groups in the *Group formation phase*. This is done by merging the *P-groups* into *k-groups* by a greedy method aimed at minimizing the distance, defined by IVL. The method used in this thesis is called "Algorithm 3: Group formation" as described by Shou et al. [2]. The heuristic of this algorithm is to take the *P-group* with the smallest IVL and then merge it with another *P-group* such that their distance is minimized. This merging process continues until the size of the new group is larger or equal to k, in which the new group is moved from *P-groups* to *k-groups*. This merging process is then repeated until the size of *P-groups* is smaller than k. The groups left over in the end is added to *k-group* such that the IVL is minimized. This phase has the complexity $O(n^2)$.

3.4 Evaluation

In the evaluation of PC-KAPRA and KAPRA the *IVL* is used to compute the information loss of the attribute values. This is done by computing *IVL* for each *k-group* G_i, and then normalizing by the number of groups m.

$$TIVL = \frac{1}{m}\sum_{i=1}^{m} IVL(G_i) \tag{3.5}$$

The total pattern loss is computed by comparing the original *SAX*-representation with the one obtained after the PC-KAPRA or KAPRA algorithms. This is done by transforming the symbolic sequence back to the real values, using the following transformation

$$T(S) = ppf\left(\frac{(2f(S)-1)}{2(\text{SAX-level})}\right) \tag{3.6}$$

Where S is a symbol from a *SAX*-sequence, and $f(S) : A \to [1, \cdots, SAX - level]$ with $\mathbf{a} \to 1, \mathbf{b} \to 2$, etc. This transformation returns the point in the middle of each interval used in the original *SAX*-transformation described in section 3.1.1.

In order to compute the total pattern loss the following equation is used

$$TPL = \frac{n}{\text{PAA-size}} \sum_{i=1}^{n} \sum_{j=1}^{\text{PAA-size}} (T(SAX_{new}^{(i)}(j)) - T(SAX_{org}^{(i)}(j))^2 \tag{3.7}$$

Where $SAX_{org}^{(i)}(j)$ and $SAX_{new}^{(i)}(j)$ is jth symbol of the ith original and transformed *SAX*-sequence, n is the number of time-series in the database.

This distance measure is similar to that of *MINDIST*, but allows the symbolic sequences SAX_{new} and SAX_{org} to have a different *SAX*-level. This is necessary when computing the distance loss in the *KAPRA* algorithm, where SAX_{org} and SAX_{new} most likely will differ. In KAPRA, SAX_{org} is the *SAX*-representation using the predetermined *max-level*.

Chapter 4

Results

The results from using KAPRA and PC-KAPRA on two different time-series will be presented and discussed in this chapter. The first dataset will be used in order to evaluate the performance of PC-KAPRA, with KAPRA as a benchmark. Then, in order to get a more qualitative insight into how the published (k,P)-anonymized data may look like, an example implemented using PC-KAPRA will be presented on the first and second dataset. Both PC-KAPRA and KAPRA were implemented in Python.

The datasets used in this thesis is taken from the *UCR Time-series classification archive*[1]. The methods KAPRA and PC-KAPRA were used on many datasets from this archive, giving similar results with regards to how TIVL and TPL with regards to changing k, P and *PAA*-size. Therefore, only the results attained from the *ECGFiveDays* test dataset will be presented. This dataset contains the ECG data from a 67 old male, measured 5 days apart. It contains 861 time-series, each representing one heartbeat. The length of the time-series are 136 and the value range is between -6 and 7.2. ECG data is quite interesting since it shows medical data, and could for that reason be deemed similar to the glucose data considered in the problem description provided in section (1.1.2).

In the problem description the hypothetical dataset considered is containing univariate time-series from various unique individual users. In this results section, this is not the case because of lack of access to such data. That is however not necessarily an issue with regards to showcasing the result from the anonymization methods. Both KAPRA and PC-KAPRA will work similarly on the ECG dataset presented here as data consisting of time-series from multiple users.

[1]https://www.cs.ucr.edu/ eamonn/time$_{s}eries_{d}ata_{2}$018

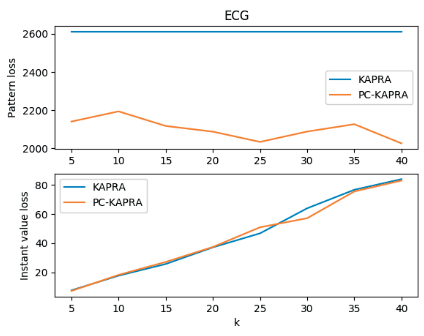
The limitations of the specific dataset, *ECGFiveDays*, is the fact that most of the heartbeats look very similar throughout the database, the characteristic of which is shown in figure (4.2). This exact characteristic is not expected to be found in a time-series database consisting of glucose data, where there probably is a larger variability between the time-series, i.e. the difference between the time-series data in the set. PC-KAPRA and KAPRA were however also used on other datasets from the *UCR Time-series classification archive* containing univariate time-series. These datasets were used to test the algorithms on data with different lengths, variability between the time-series and value ranges. In the end, the results were for all datasets practically identical with regards to dependency on k, P and PAA-size and difference between PC-KAPRA and KAPRA, as to the ones presented in (4.1)

In figure (4.1), a comparison is made between the KAPRA and PC-KAPRA algorithm with regards to TPL and TIVL, for different values of k, P and PAA-size, which otherwise would be kept at $k = 10$, $P = 5$ and PAA-size $= 10$. The pattern loss is done by computing the MINDIST between SAX_{org} and SAX_{new} which is here chosen to be a SAX-representation with SAX-level $= 10$ and the SAX-representation after anonymization, respectively.

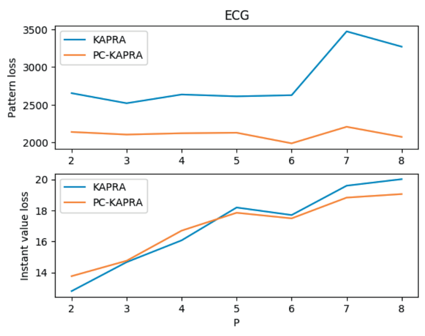
The parameters k,P and PAA-size are typically chosen after some predetermined privacy requirement and PAA-size would depend on the required granularity in the data. In this thesis, the specific values were chosen mainly similar to the ones in the experimental setup used in the study by Shou et al [2]. Other value combinations were also tried, and they gave a similar result regarding the dependency of TIVL and TPL with regards to k, P and PAA-size.

The worst case breach probability is $P_{breach} = 1/P$, see section (2.2.2). A larger k would not help towards the diminishing this worst case scenario with regards to finding a link between between auxiliary data and which k-group envelope with would belong to. However, it would make each time-series indistinguishable from $k-1$ other time-series, with regards to attribute information.

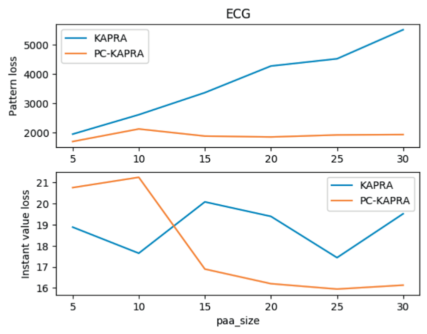
Since the breach probability P_{breach} is only decided from the formation of (k,P)-groups, there should not be a difference in this respect between PC-KARPA and KAPRA. However, the PC-KAPRA algorithms creates uncertainty with regards to the true value of the pattern representation, while KAPRA publishes the exact representation together with the corresponding SAX-level. Because of this, it is possible to find a 1-to-1 correspondence between published time-series and auxiliary data. In PC-KAPRA the pattern representation is perturbed and represented by centroid of the cluster, which makes it very unlikely to find a



(a) Varying k.



(b) varying P.



(c) varying *PAA*-size.

Fig. 4.1: Instant value loss and pattern loss after using KAPRA and PC-KAPRA on ECG data. The variables are kept as $k = 10$, $P = 5$ and *PAA*-size $= 10$ when not varied.

1-to-1 correspondence.

When k is allowed to vary, as in figure (4.1a), it can be seen that the pattern loss is constant in the KAPRA algorithm, which is expected since the size of k does not affect the P-group formation step. In PC-KAPRA there are small variations, which due to the stochasticity of the k-means clustering algorithm. The TIVL has a similar linear increase for both algorithms, which can similarly be seen when P is allowed to vary. This indicates that P and k should be kept as small as possible to optimize for information loss, while still be large enough for the desired privacy requirements.

When the *PAA*-size is small, the pattern loss for both methods are quite similar. It is expected that KAPRA will perform alright in this case, because there are less unique *SAX*-

representations in the database, allowing for more identical patterns to be grouped together.

All together, there is clearly a diminished total pattern loss in PC-KAPRA, while the TIVL is roughly identical in both of the methods. This allows for more freedom when using PC-KAPRA when it comes to choosing the size of both *PAA*-size and *SAX*-level, while still publishing a more accurate pattern representation.

In figure (4.2) a qualitative look at the published data from the PC-KAPRA algorithm is provided. To get a broad view of the results, the published data from four different *k-groups* corresponding to the largest and smallest pattern and attribute value loss are shown. These *k-groups* are found by computing the pattern loss and IVL for each group, after running the algorithm on the *ECGFiveDays* and the *ArticularyWorkRecognitionDimension1* test dataset [2]. The *ArticularyWorkRecognitionDimension1* is also from the *UCR Time-series classification archive* and measures the movement of the lips when different words are articulated. In this case, it was chosen to illustrate the results of the PC-KAPRA algorithm used on a database with a higher variance between the time-series than *ECGFiveDays*.

Due to the stochastic behaviour of the k-means algorithm, the k-groups will look different for every run. In this run the following values are used: $k = 10$, $P = 3$, *PAA*-size $= 5$, *SAX*-level $= 26$. These values were chosen because it gave a good illustration of the results. The interval between the two line segments is the envelope, in which all time-series are contained and they are represented by the *SAX*-representations given in the top-right corner. In this figure, the exact time-series are plotted out to give an indication of how the time-series might be grouped together, this will normally not be the case when publishing the data. Furthermore, the envelope corresponds exactly with the largest and smallest values respectively in the k-groups. This will of course carry a certain privacy risk. If for instance the entire top-side of the envelope is represented by a single time-series, then that time-series would be fully disclosed. This scenario can be avoided by adding noise to the envelope or by choosing the envelope in some other fashion. How to do this in a good way for retaining attribute information has not been investigated in this thesis.

It his hard to judge qualitatively how useful the published data is in figures (4.2) and (4.3). Even though the envelope for the published ECG data is small, the variance between the time-series was initially quite small as well. The usefulness of the data is hard to discuss, because it is too dependent on the case. However, in all the k-groups illustrated in the figures,

[2]http://www.timeseriesclassification.com/description.php?Dataset=ArticularyWordRecognition

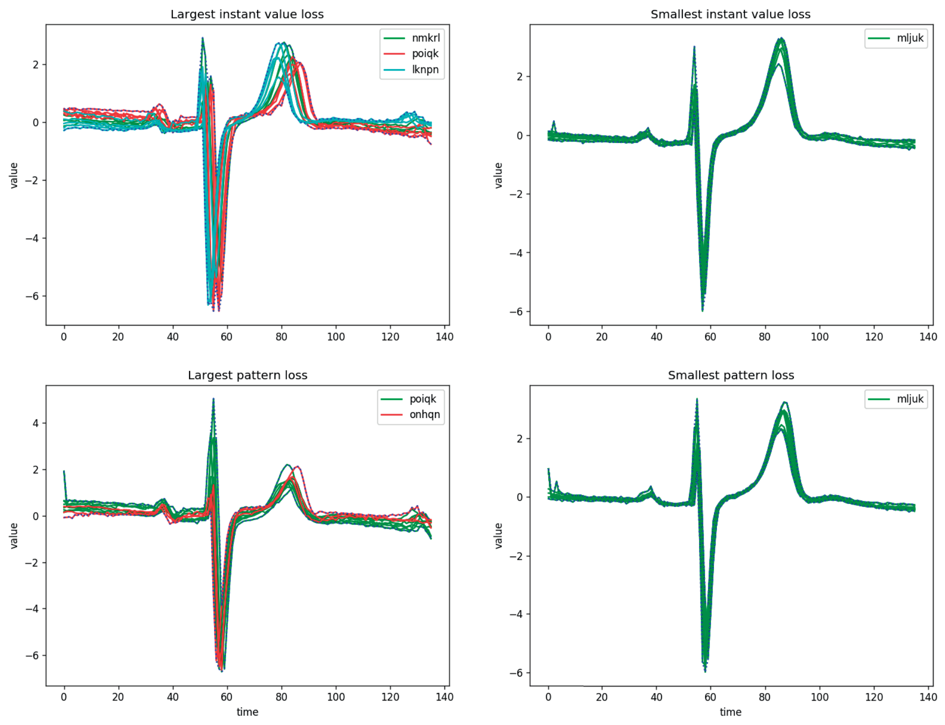


Fig. 4.2: Sample of published data from PC-KAPRA on the *ECGFiveDays* dataset, with the smallest and largest information loss with regards to pattern and value loss. Here we have use $k = 10$, $P = 3$ and PAA-size $= 5$.

there is a clear grouping of similar values, which goes well with the goal of publishing single univariate time-series as close to their original value as possible.

In figure (4.2), there seems to be a small time shift between the P-groups in the plot corresponding to largest attribute value loss. A temporal shift might in many cases not really be seen as a difference between time-series, but they will nevertheless contribute to a large instant value loss. This is a limitation of the PC-KAPRA algorithm as implemented in the thesis. In order to currently avoid this, a preprocessing step could be used to align the time-series in time.

One limitation of the PC-KAPRA algorithm, is with the regards to what types of time-series which might be deemed similar to each other. The similarity between time-series can be based on different requirements. For instance, considering blood glucose data, a similarity

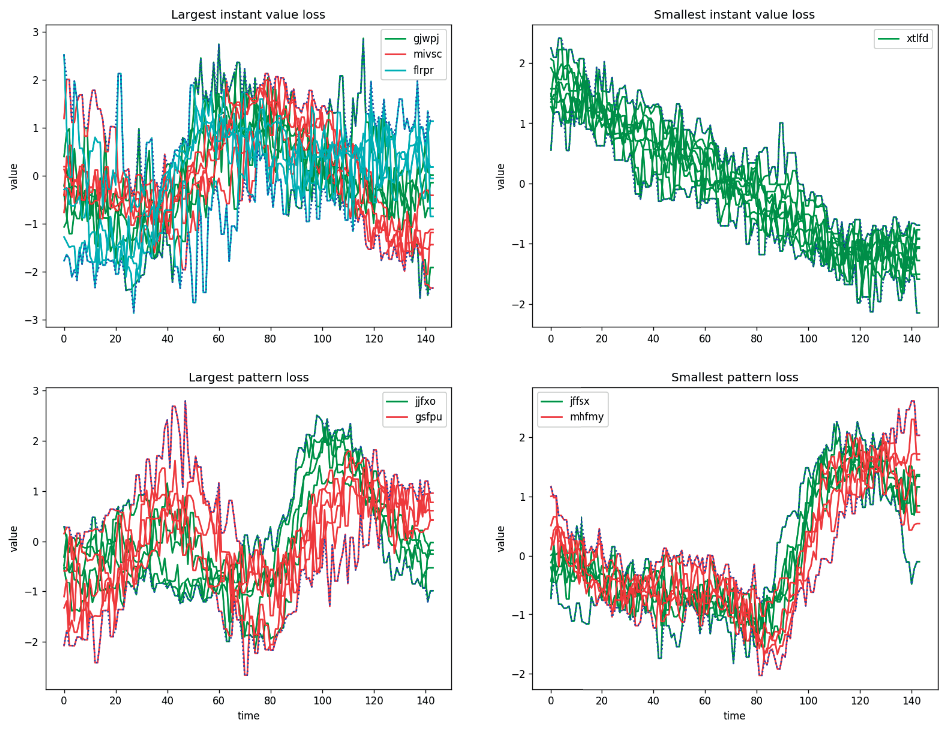


Fig. 4.3: Sample of published data from PC-KAPRA on the *ArticularyWorkRecognitionDimension1* dataset, with the smallest and largest information loss with regards to pattern and value loss. Here we have use $k = 10$, $P = 3$ and *PAA*-size $= 5$.

between two time-series could be defined by having similar glucose fluctuations after waking up, in this case some other distance metric and possibly data representation has be used.

Another limitation comes with regards to complexity. The bottleneck in PC-KAPRA is in the top-down partitioning algorithm and k-groups formation phases with complexity $O(n^2)$. If the dataset becomes too large, this could become an issue. A future direction for improvement could be to finding a quicker method for partitioning and forming the k-groups.

Beyond the methods implemented in this thesis, i.e KAPRA and PC-KAPRA, a look was put into using the algorithm *CAT*, mentioned in section (2.2.1). However, PC-KAPRA can be seen as a mixture between KAPRA and CAT, since it implements both a clustering methodology and the (k,P)-anonymity measure. Since there was a desire to establish (k,P)-anonymity, *CAT* was never implemented in this thesis. It could however be interesting to see how it compares with to PC-KAPRA with regards to TIVL. It also has its merits when it comes to anonymizing time-series when patterns are not of interest.

Chapter 5

Discussion and Conclusion

In this chapter, a holistic discussion regarding the thesis will be given. The applicability of the methods introduced in this thesis on the problem description given in section (1.1.2) will be discussed, where the scenario of a database generated by a glucose monitoring smart-watch was considered along with the main problem of anonymizing univariate time-series of glucose data. Future research directions will also be discussed, with focus on the implementation of the (k, P)-anonymity measure.

5.1 Differential privacy and Sensitive data

It is important to investigate a wide scope of state-of-the-art research to get a good overview of the possible directions that can be taken towards reaching privacy. The importance of this is even more enforced by the fact that univariate time-series can look quite different from each other. In this thesis the main focus has been on time-series from individual users, where (k, P)-anonymity was considered the strongest approach with regards to publishing data with minimum information loss. However, there was also an investigation into aggregate time-series data, where the strong guarantee of differential privacy could be achieved. Furthermore, methodologies of how sensitive information could be removed, was investigated.

Taking the scenario with a glucose monitoring smart-watch, it seems reasonable to continuously track the aggregate data considering users with diabetes. In this case a user is expected to only be queried once, and can therefore be protected under *event-level* differential privacy which is aimed at hiding the occurrence of a single event. If aggregate trajectory data is collected, then *user-level* differential is necessary, because a user is expected to answer multiple queries regarding their location.

Differential privacy can seems like quite a bit too drastic of a step for implementing privacy, considering the large information loss. However, in some scenarios, for instance with tracking the number of users with diabetes, the exact number is not that important. In those cases, it is good to establish a strong security for the privacy of the data provider and convenience of the data surveyor. Other types of relevant aggregate time-series data that could be gather include: The number of users, number of sold devices, user information such as age and gender, etc.

When it comes to hiding sensitive information, there are two ideas considered in this thesis. First, a simpler idea which focuses on adding noise to the entire time-series to generate a certain degree of uncertainty, or reasonable deniability. The problem with this approach is that a lot of information is lost when applying noise on all data points. The second approach replaces sections of the time-series which contains sensitive information. This approach has a lot of potential with regards to retaining information, but also requires a deep understanding of all the inferences that can be found in the dataset.

5.2 (k,P)-anonymity

Publishing data under (k,P)-anonymity while still retaining a good amount of information was possible using the PC-KAPRA method. The method is based on the idea of clustering similar time-series together, and then transforming the data so that they are all presented identically. In this regard, a version similar to the raw data is anonymously published. This method could be used not only to retain aggregate information, but also more qualitative parts of the data.

The measure of (k,P)-anonymity could be used as a basis to create different methods towards the anonymization of time-series data. In this thesis, the k-means clustering method was used to enhance the KAPRA algorithm, with regards to diminishing pattern loss. This clustering method is quite generic and easy to understand, and has a simple implementation which proved sufficient to give a substantial improvement with regards to preservation of pattern information. If more was knows about the database, then a more intelligent way of distributing initial clusters could be implemented, which could substantially help with reducing pattern loss.

For future work, it is completely essential that PC-KAPRA gets tested on actual glucose data, and gets evaluated in its usefulness. The pattern loss and instant value loss measures used in this thesis is mainly used to compare PC-KAPRA with the benchmark KAPRA, and does not offer any real information regarding how useful the published data is with regards to analysis. When this is done, potentially more suiting ways of representing the patterns and attribute values can be investigated.

References

[1] Katharine Toll. Privacy and freedom. by alan f. westin. new york: Atheneum press, 1967. *Social Work*, 13(4):114–115, 10 1968.

[2] L. Shou, X. Shang, K. Chen, G. Chen, and C. Zhang. Supporting pattern-preserving anonymization for time-series data. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):877–892, April 2013.

[3] THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION. Regulation (eu) 2016/679, 2016.
https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[4] Chiu M. C. Chadowitz C. Ganesan D. Kalogerakis E. Parate, A. Risq: Recognizing smoking gestures with inertial sensors on a wristband. In *MobiSys ... : the ... International Conference on Mobile Systems, Applications and Services*, International Conference on Mobile Systems, Applications, and Services, 2014, pages 149–161, 2014.

[5] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, 1998.

[6] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3&#8211;4):211–407, August 2014.

[7] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. L-diversity: privacy beyond k-anonymity. In *22nd International Conference on Data Engineering (ICDE'06)*, pages 24–24, April 2006.

[8] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering – a decade review. *Information Systems*, 53:16 – 38, 2015.

[9] J S. Adeline Johnsana, Rajesh Appusamy, and Kishore Verma. Cats-clustered k-anonymization of time series data with minimal information loss and optimal re-identification risk. *Indian Journal of Science and Technology*, 9, 12 2016.

[10] Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, Jun 2015.

[11] Liyue Fan and Li Xiong. Adaptively sharing time-series with differential privacy. *CoRR*, abs/1202.3461, 2012.

[12] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy Rothblum. Differential privacy under continual observation. In *STOC '10: Proceedings of the 42nd ACM symposium on Theory of computing*. ACM, June 2010.

[14] Y. Cao and M. Yoshikawa. Differentially private real-time data release over infinite trajectory streams. In *2015 16th IEEE International Conference on Mobile Data Management*, volume 2, pages 68–73, June 2015.

[15] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 735–746, New York, NY, USA, 2010. ACM.

[16] Spiros Papadimitriou, Feifei Li, George Kollios, and Philip S. Yu. Time series compressibility and privacy. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 459–470. VLDB Endowment, 2007.

[17] Mohammad Malekzadeh, Richard G. Clegg, and Hamed Haddadi. Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis. *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 165–176, 2018.

[18] Nazir Saleheen, Supriyo Chakraborty, Nasir Ali, Md Mahbubur Rahman, Syed Monowar Hossain, Rummana Bari, Eugene Buder, Mani Srivastava, and Santosh Kumar. msieve: Differential behavioral privacy in time series of mobile sensor data. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 706–717, New York, NY, USA, 2016. ACM.

[19] http://vigne.sh/posts/piecewise-aggregate approx/.

[20] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144, Oct 2007.

[21] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144, Oct 2007.

[22] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[23] Jian Xu, Wei Wang, Jian Pei, Xiaoyuan Wang, Baile Shi, and Ada Wai-Chee Fu. Utility-based anonymization for privacy preservation with less information loss. *SIGKDD Explor. Newsl.*, 8(2):21–30, December 2006.

Printed by Tryckeriet i E-huset, Lund 2020

LUND UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2020-740
http://www.eit.lth.se