Verifiable Anomaly and Similarity Detection Using Matrix Profile in Private Time-series

Xavier Bultel[0000−0002−8309−8984], Charlène Jojon[0009−0009−9597−541X], Benjamin Nguyen[0000−0003−0719−9684], and Haoying Zhang[0009−0000−3277−8827]

INSA Centre-Val de Loire, Université d'Orléans, Inria, France
firstname.lastname@insa-cvl.fr

Abstract. Analyzing time-series databases in a privacy-preserving manner has gained significant attention, especially when the data contains sensitive personal information such as medical records or spatio-temporal data such as trajectories. Motivated by scenarios where a user must show whether an anomaly (or similarity) is detected in a time series containing sensitive data, we propose a toolkit for proving these properties on (committed) private time series. We leverage Matrix Profile (MP), a state-of-the-art data-mining structure, to detect *subsequence* anomalies and similarities in time series, in contrast to many works that only detect anomalies and similarities on *complete time series*. As recent findings have shown, the aggregated data used by MP (such as subsequence distances or MP values) leak critical information about the time series. It is therefore crucial to consider a strong adversary model where all information other than the presence or absence of anomalies/similarities remains protected. To guarantee this, we propose a combination of commitment and zero-knowledge proof systems that ensure both the validity of the proven result and the (unconditional) protection of the time series. The proposed schemes maintain reasonable execution times, even for large real-time time series.

1 Introduction

Time-series applications today increasingly rely on cloud infrastructures for data storage and computation. The rapid growth of sensor-rich environments, such as wearable devices, medical monitors, and mobile endpoints, has led to massive volumes of personal and behavioral data being continuously collected. Outsourcing the processing of such data to untrusted cloud services raises serious privacy concerns. This makes secure computation techniques essential to ensure that sensitive information remains hidden from the service provider while computations are performed on the data.

A general task in this setting is performing similarity or anomaly queries over private time series. For example, a medical professional may need to check whether there are any anomalies in a patient's electrocardiogram (ECG) signal. To do so, a widely used approach in data analytics is to use a technique

called Matrix Profile (MP) [24] (explained in detail in Sec. 2.2). MP is a transformation of a time series which produces a structure that captures all pairwise similarity relationships between subsequences and enhances the detection of anomalies or absence of similarities, in an unsupervised way, i.e. without needing any annotated training data or reference data to detect the anomalies. Despite being unsupervised, MP performs well, for instance Paparrizos *et al.* show in their benchmark that MP consistently achieves state-of-the-art performance in anomaly detection [16]. To motivate the uses of MP, we illustrate (see Figure 1) a classical application: detecting ECG anomalies using MP.

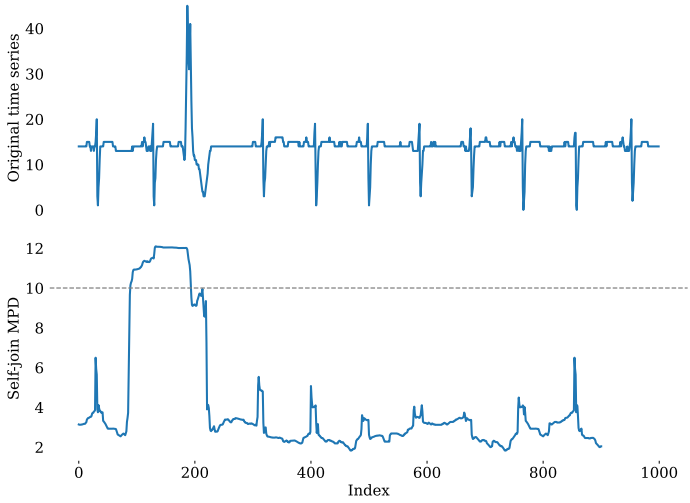

Fig. 1. Detecting a self-anomaly using the Matrix Profile. Top: ECG with an anomaly in the third heartbeat. Bottom: self-join MPD (m = 100), where the anomaly appears as distances exceeding the threshold. The position of the anomaly is left-shifted by the length of the subsequence window used in the distance computation.

It is important to stress that MP is used in an unsupervised manner, *i.e.,* by comparing a time series with itself in order to find anomalies. No other data is needed, contrary to an approach where *e.g.,* the time series would be compared to many "template" time series, which makes the approach both simpler to use, and with no need for any additional (and potentially private) training data. For instance, the Statewide California Earthquake Center [19] explains that they use the self-join MP to identify previously uncatalogued seismic events by matching all sub-windows (of a set length) in a continuous stream against the remainder of the stream, which cannot be accomplished using template-based matching methods.

Indeed, as an ECG is sensitive data, in certain scenarios, the person checking for anomalies should not learn this data. For example, marathon runners must undergo an ECG to detect any anomalies that would make running a marathon

contraindicated. The marathon organizer may want to verify that registered athletes do not present any risks, but cannot require athletes to provide their ECG data in plain text. On the other hand, the presence of anomalies may give access to certain medical certificates or reimbursement for expensive treatments and exams. The insurance company of a patient claiming these rights may want proof of the patient's legitimacy in their care pathway, but again, providing the ECG would give too much sensitive health information to the insurance company.

A naive approach would be to provide only the MP values, which contain only an aggregated version of the data, but not the time series itself, in the hope that it would be difficult to reverse MP. Unfortunately, recent work [25] exploring privacy-preserving data sharing with MP shows that this is not a viable approach. Indeed, [25] demonstrate that adversaries can exploit subsequence distance patterns to reconstruct substantial portions of the original time series or infer sensitive attributes. This reveals a fundamental gap that while MP is a highly expressive and widely adopted tool for unsupervised time series mining, just like the time series itself, it cannot be directly shared or computed in the clear without privacy protection.

Objectives We aim to bridge this gap by proposing a way for a user called the *prover*, who has a time series committed and authenticated by the *data provider* that generated it (i.e., for the ECG example, the physician or medical device that collected the data), to prove to a *verifier* the presence or absence of anomalies without leaking anything other than the public MP parameters (i.e., the size of the time series/sub-sequences, the distance used, and the anomaly threshold). This setting is illustrated in Figure 2. Note that the proving and verification phases can be performed offline, thus real-time efficiency is not expected, as long as the approach scales to realistic time series sizes (*i.e.,* 1,000 points). Conversely, calculations performed by the data provider may be carried out using embedded medical equipment and must be as standard and efficient as possible. From an application perspective, we consider the following examples of privacy-preserving scenarios for each of the cases discussed in the article:

- **No anomaly**: our protocol allows an athlete to prove that their private ECG is compliant with competition requirements;
- **Anomaly**: our protocol allows patients to prove an anomaly on their private ECG in order to obtain a medical certificate, treatment, or examination;
- **No similarity**: our protocol allows a user to prove that there is no similarity between their private trajectory and that of an infected patient (contact tracing);
- **Similarity**: our protocol allows a patient to prove that their private biomedical signal shares a clinically relevant pattern with a reference biomedical signal in order to confirm a medical diagnosis.

Contributions We instantiate the commitment using the classic Pedersen commitment. The signature can be instantiated by any unforgeable signature. Our main contribution is the design of a non-interactive zero-knowledge proof for each

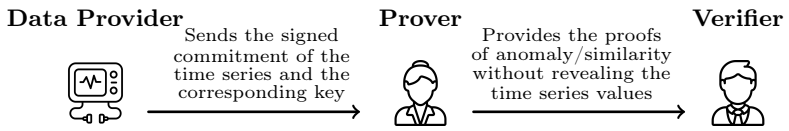

Fig. 2. Scenario description

of the cases studied (anomaly or not, and similarity or not). To our knowledge, this is the first work to apply zero-knowledge proofs with subsequence distance metric in time series. To do this, we opt for a tailor-made proof design using Schnorr sigma protocols and the homomorphism properties of Pedersen's commitment to securely compute commitments including the aggregated statistical data used by Matrix Profile. Notably, our protocol verifies strong properties: the proof is extractable (i.e., to successfully build the proof, the prover must know the committed time series, and this series must verify the desired property), and the time series is perfectly protected by the hiding commitment and the zero-knowledge proof (even against an unbounded computational adversary). Furthermore, our proof uses very standard tools and relies on very common and weak assumptions (only the discrete logarithm assumption, and the random oracle model in the non-interactive setting). Finally, we present a full Rust implementation of our protocols and evaluate the scalability on realistic time-series workloads.

Paper organization. The paper is organized as follows: Section 2 introduces the cryptographic and MP background of our work. Section 3 presents our construction, and Section 4 its instantiation. Section 5 presents the performance results of our Rust implementation, Section 6 discusses related works, and Section 7 concludes.

2 Preliminaries and Notations

Notation We denote by $[\![n]\!]$ the set $\{1, \cdots, n\}$ and by $[\![i; j]\!]$ the set $\{i, \cdots, j\}$. We write $x \xleftarrow{\$} S$ to mean that x is picked uniformly at random from a set S. For simplicity, we write $x, y \in S$ (resp. $x, y \leftarrow S$) to say that $x \in S$ and $y \in S$ (resp. $x \leftarrow S$ and $y \leftarrow S$). We use the acronym s.t. for *such that*, PT for *Polynomial Time*, and PPT for *Probabilistic Polynomial Time*.

2.1 Background on Cryptography

We first recall the Discrete Logarithm (DL) assumption.

Definition 1 (Discrete Logarithm (DL) assumption). *Let λ be a security parameter and $\mathbb{G} = \langle g \rangle$ be a group of prime order p, the Discrete Logarithm (DL) assumption states that there exists a negligible function ϵ s.t. for any PPT adversary $\mathcal{A}$, it holds that* $\Pr[h = g^x : h \xleftarrow{\$} \mathbb{G}; x \leftarrow \mathcal{A}(h)] \leq \epsilon(\lambda)$.

A Non-Interactive Proof (NIP) allows a prover to prove the knowledge of a witness w s.t. $(s, w) \in \mathcal{R}$ where s is a statement and $\mathcal{R}$ a relation without revealing information on w.

Definition 2 (Non-Interactive Proofs (NIP) [2]). *Let $\mathcal{R}$ be a binary relation and let $\mathcal{L}$ be a language s.t. $s \in \mathcal{L} \Leftrightarrow (\exists w, (s, w) \in \mathcal{R})$. A* Non-Interactive Proof *NIP for $\mathcal{L}$ is a couple of PPT algorithms* $(\mathsf{NIP}, \mathsf{Verify})$ s.t.*:*

$\mathsf{NIP}\{w : (s, w) \in \mathcal{R}\}$**:** *on input a statement/witness (s, w), outputs a proof π.*
$\mathsf{Verify}(s, \pi)$**:** *on input a statement s and a proof π, outputs a bit b.*

We define the zero-knowledge and extractability properties of a NIP *as follows:*

(Perfect) Zero-knowledge. *A proof π* leaks no information*, i.e., there exists a PT algorithm* Simulator *(called the* simulator*) s.t.* $\mathsf{NIP}\{w : (s, w) \in \mathcal{R}\}$ *and* $\mathsf{Simulator}(s)$ *follow (exactly) the same probability distribution.*

Extractability. *There exists a PPT knowledge extractor* Ext *and a negligible function ϵ_{ext} s.t. for any algorithm $\mathcal{A}^{\mathsf{Simulator}(\cdot)}(\lambda)$ having access to a simulator that forges proofs for chosen statement and that outputs a fresh pair (s, π) with $\mathsf{Verify}(s, \pi) = 1$, the extractor $\mathsf{Ext}^{\mathcal{A}}(\lambda)$ outputs w s.t. $(s, w) \in \mathcal{R}$ having access to $\mathcal{A}(\lambda)$ with probability at least $1 - \epsilon(\lambda)$.*

We now give the definition of a commitment and Pedersen's commitment

Definition 3 (Commitment Scheme). *A commitment scheme is a pair of two PPT algorithms* $(\mathsf{Com}, \mathsf{Ope})$*, s.t. the algorithm* Com *takes a message and a secret key as input and returns a commitment, and the algorithm* Ope *takes a commitment, a secret key and a message as input and returns an acceptance bit. We define the hiding and binding properties of a commitment scheme as follows:*

(Perfect) Hiding. *Given two messages of its choice, no PPT algorithm $\mathcal{A}$ is able to distinguish whether the commitment was generated from the first or the second message, except with a probability* $1/2$*.*

Binding. *No PPT algorithm $\mathcal{A}$ is able to generate a commitment and two different messages s.t. the commitment can be opened to both messages with a non-negligible probability.*

Definition 4 (Pedersen Commitment Scheme [17]). *Let λ be a security parameter, $\mathbb{G}$ be group of prime order p s.t. $p > 2^{\lambda}$, and g and h be two random generators of $\mathbb{G}$. The Pedersen Commitment Scheme consists of the algorithms* $(\mathsf{Com}, \mathsf{Ope})$ *s.t. the algorithm* Com *takes as input a message m and a secret key k and returns a commitment $c = g^m h^k$, and the algorithm* Ope *takes as input a commitment c, a secret key k and a message m and returns 1 iff $c = g^m h^k$.*

Pedersen Commitment Scheme is known to support additive homomorphic operations while preserving perfect hiding and computational binding under the DL assumption.

2.2 Background on Matrix Profile

A *Time Series* can be represented by a vector $x = (x_i)_{i\in[\![n]\!]}$ together with a timestamp vector $\mathsf{tmp} = (\mathsf{tmp}_i)_{i\in[\![n]\!]}$. In what follows, we assume that temporal dependency is implicitly given by the ordering of x, and we omit explicit use of tmp for clarity.

Window notation. Consider a time series x of length n. We denote a window $s^x_{i,m}$ as a continuous subvector of values of x of length m, starting at index i. We denote the set of all the windows of size m of a time series x of length n, as $\mathsf{Sub}^x_m = \{s^x_{1,m}, \ldots, s^x_{n-m+1,m}\}$, and $\mathcal{I}$ as the set of indices $[\![n-m+1]\!]$.

Definition 5 (Self-join Matrix Profile [24]). *Let x be a time series of length n, let d be a distance function, let m be a window size, and let $\mathcal{J}_i$ be the set $\{j \in \mathcal{I} : |j-i| > m/2\}$. Self-join Matrix Profile (MP) is a data mining structure $\mathsf{MP} : \mathbb{R}^n \to \mathbb{R}^{n-m+1} \times \mathbb{N}^{n-m+1}$ composed of two vectors: Matrix Profile Distance (MPD) and Matrix Profile Index (MPI) s.t. :*

$$\mathsf{MPD}(x) = (\mathsf{MPD}_i(x))_{i\in\mathcal{I}} = \left(\min_{j\in\mathcal{J}_i}\left(d(s^x_{i,m}, s^x_{j,m})\right)\right)_{i\in\mathcal{I}};$$

and $\mathsf{MPI}(x) = (\mathsf{MPI}_i(x))_{i\in\mathcal{I}}$ where $\mathsf{MPI}_i(x)$ stores the (minimum) index j verifying $\mathsf{MPD}_i(x) = d(s^x_{i,m}, s^x_{j,m})$.

Remark 1. The definition 5 restricts the windows $(s^x_{j,m})_{j\in\mathcal{I}}$ s.t. $|j-i| > m/2$ which avoids trivial self-matches with the window $s^x_{i,m}$.

Although MPI is useful in certain applications, it is not relevant for the scenarios considered in our work and we do not use it in this article. Therefore, for simplicity, we use the term (and notation) MP instead of MPD.

In what follows, we define two functions allowing to detect anomaly or similarity using the structure of Matrix Profile.

Definition 6 (Anomaly and Similarity detection function using MP). *Let x be a time series. The anomaly detection function $\mathsf{Ano} : \mathbb{R}^n \to \mathbb{B}$ (resp. the similarity detection function $\mathsf{Sim} : \mathbb{R}^n \to \mathbb{B}$) on x returns 1 if there exists a value in MP bigger than (resp. smaller than) a predefined threshold ϵ, (resp. δ), and 0 otherwise. That is,*

$$\mathsf{Ano}(x) = \begin{cases} 1 & \text{if } \exists v \in \mathsf{MP}(x),\ v > \epsilon, \\ 0 & \text{if } \forall v \in \mathsf{MP}(x),\ v \le \epsilon. \end{cases} \quad \mathsf{Sim}(x) = \begin{cases} 1 & \text{if } \exists v \in \mathsf{MP}(x),\ v < \delta, \\ 0 & \text{if } \forall v \in \mathsf{MP}(x),\ v \ge \delta. \end{cases}$$

Here, $\epsilon, \delta \in \mathbb{R}$ are predefined threshold values. Their selection is carried out independently of the private time series and depends on its nature, which is not considered as private information. These values are typically determined on established expert knowledge or prior empirical analysis [6,8].

In the following of the paper, d will always denote the classic (unnormalized) Euclidean distance: $d((a_i)_{i\in[\![n]\!]}, (b_i)_{i\in[\![n]\!]}) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$. The Euclidean

distance is common in Matrix Profile applications, especially in health care. Although most MP algorithms were developed for the z-normalized Euclidean distance, normalization can be inappropriate in many settings, such as ECG or heart-rate analysis, where the raw scale carries clinical meaning [22,1,27].

For the cases $\mathsf{Ano}(x) = 1$ or $\mathsf{Sim}(x) = 1$, the detection should work in an unsupervised way (no training data or reference data is available). Furthermore, the specific locations of anomalies or similarities are not known in advance and must not be revealed to the verifier, as disclosing them could leak timing information about sensitive events.

3 Our construction

In this section, we design cryptographic schemes that address our problem. Our goal is to enable a user to prove to another whether or not a (private) time series contains similarities or anomalies using its matrix profile.

In our scenario, the time series comes from a medical device (trusted and not colluding with the prover) that commits the time series to make it private and signs it to guarantee its provenance. The device communicates the time series, the commitment, its opening key and the signature to the user who owns the data. This user must then be able to prove the existence or absence of anomalies and similarities based on the signed commitment, without revealing anything else to the verifier. To this end, we propose four zero-knowledge proofs corresponding to each of the four cases (proof of similarity, proof of anomaly, proof of no similarity and proof of no anomaly). These proofs are all proofs of knowledge of the committed time series and the corresponding opening key.

Our solution applies to a time series $x = (x_i)_{i \in [\![n]\!]}$ with integer values (cases with truncated floating point values can be achieved by simply discretizing the values by multiplying them by a power of 10). In what follows, we consider the *self-join* matrix profile where a (unique) time series is compared to itself. Readers can refer to Table 1 to recall the meaning of the notations used in this section.

3.1 Setup and commitment

Each of our algorithms implicitly has access to a public setup $(\lambda, \mathbb{G}, p, g, h, \lambda, n, m, \ell)$ where λ is a security parameter, n is a sequence length, m is a window size, ℓ is an integer encoding bit size, $\mathbb{G} = \langle g \rangle$ is a group of prime order p s.t. $p > 2^\lambda$ (where the DL assumption is assumed to hold), and h is a random generator of $\mathbb{G}$.

In order to keep the tools as standard as possible on the device side, we use Pedersen's commitment to commit each of the values x_i in the time series. In what follows, $(\mathsf{Com}, \mathsf{Ope})$ denotes Pedersen's commitment using the bases (g, h). Note that when we use Pedersen's commitment on other bases (g', h'), they will be made explicit by the notation $(\mathsf{Com}_{g',h'}, \mathsf{Ope}_{g',h'})$. Our time series commitment $(\mathsf{MP.Com}, \mathsf{MP.Ope})$ is defined as follows.

Table 1. List of some notations and their descriptions

Notation	Description
x	a private time series
n	length of the private time series x
m	window size
$s^x_{i,m}$	subsequence window of size m
ϵ / δ	value of the anomaly/similarity threshold
$\mathcal{I}$	set $[\![n-m+1]\!]$
$\mathcal{J}_i$	set $\{j \in \mathcal{I} : \lvert j-i\rvert > m/2\}$
ℓ	encoding bit length
g and h	generators of the group $\mathbb{G}$
x_i	i-th value of the time series
$x_{i,j}$	value $x_i - x_j$ for $j < i$
c_i	Pedersen commitment of x_i using g and h as generators
k_i	opening key of the commitment c_i
$c_{i,j}$	Pedersen commitment of $x_{i,j}$ using g and h as generators
$k_{i,j}$	value $k_i - k_j$, opening key of the commitment $c_{i,j}$ for $j < i$
$\widetilde{c}_{i,j}$	Pedersen commitment of $x_{i,j}$ using $c_{i,j}$ and h as generators (or Pedersen commitment of $x^2_{i,j}$)
$\widetilde{k}_{i,j}$	opening key of the commitment $c_{i,j}$
$\mathbf{d}_{i,j}$	squared Euclidean distance between $s^x_{i,m}$ and $s^x_{j,m}$
$D_{i,j}$	Pedersen commitment of $\mathbf{d}_{i,j}$ using g and h as generators
$w_{i,j}$	opening key of the commitment $D_{i,j}$
$\mathbf{d}_{i,j,u}$	binary value of the u^{th} bit of $\mathbf{d}_{i,j}$
$D_{i,j,u}$	Pedersen commitment of $\mathbf{d}_{i,j,u}$ using g and h as generators
$w_{i,j,u}$	opening key of the commitment $D_{i,j,u}$

$\underline{\mathsf{MP.Com}(x)}$**:** parses x as $(x_i)_{i\in[\![n]\!]}$. For each $i \in [\![n]\!]$, picks $k_i \xleftarrow{\$} \mathbb{Z}^*_p$, and computes $c_i = \mathsf{Com}(x_i, k_i)$. Sets $C = (c_i)_{i\in[\![n]\!]}$ and $K = (k_i)_{i\in[\![n]\!]}$, and returns (K, C).
$\underline{\mathsf{MP.Ope}(K, C, x)}$**:** parses x as $(x_i)_{i\in[\![n]\!]}$, C as $(c_i)_{i\in[\![n]\!]}$ and K as $(k_i)_{i\in[\![n]\!]}$, and returns $\bigwedge_{i\in[\![n]\!]} \mathsf{Ope}(c_i, k_i, x_i)$.

We omit the signature that can be instantiated by any signature scheme (we also omit the use of a secure channel between the device and the owner of the time series used to transmit K and x).

3.2 Overview of our Proof of Anomaly/Similarity Detection

We will now present our methodology for proving anomaly/similarity and absence of anomaly/similarity. As a preliminary step, we first rewrite the relations in Definition 6 in an equivalent manner that is more convenient for the design of our proof. Indeed, each case can be stated differently by considering the involved function min in MP. Specifically,

$$\mathsf{Ano}(x) = 1 \Leftrightarrow \exists v \in \mathsf{MP}(x) \text{ s.t. } v > \epsilon \Leftrightarrow \exists i \in \mathcal{I} \text{ s.t. } \min_{j\in\mathcal{J}_i}(d(s^x_{i,m}, s^x_{j,m})) > \epsilon$$
$$\Leftrightarrow \exists i \in \mathcal{I}, \forall j \in \mathcal{J}_i, d(s^x_{i,m}, s^x_{j,m}) > \epsilon.$$

As the condition $\mathsf{Ano}(x) = 0$ is the negation of $\mathsf{Ano}(x) = 1$, we have:

$$\mathsf{Ano}(x) = 0 \Leftrightarrow \forall i \in \mathcal{I}, \exists j \in \mathcal{J}_i, d(s^x_{i,m}, s^x_{j,m}) \leq \epsilon < \epsilon + 1.$$

Similarly:

$$\mathsf{Sim}(x) = 1 \Leftrightarrow \exists i \in \mathcal{I}, \exists j \in \mathcal{J}_i, d(s^x_{i,m}, s^x_{j,m}) < \delta;$$
$$\mathsf{Sim}(x) = 0 \Leftrightarrow \forall i \in \mathcal{I}, \forall j \in \mathcal{J}_i, d(s^x_{i,m}, s^x_{j,m}) > \delta - 1.$$

Our proof therefore shows one of these relations on a committed time series. To do this, we use Pedersen's commitment partial homomorphism property for addition, and Schnorr proofs to show relations between committed values. In particular, when comparing committed values with thresholds, we use the following method proposed in [3]: commit each bit of the committed value in binary, use homomorphism properties and Schnorr proofs to show that the committed bits correspond to the originally committed value, and prove the comparison using the committed binary-encoded values with Schnorr proofs. This method provides efficient comparison proofs whose complexity is logarithmic in the maximum size of the committed values. In this way, we obtain a tailor-made proof using very standard and efficient tools that implement the actual computations performed to evaluate the matrix profile of the committed time series.

However, one limitation of this methodology is that it requires working with integer data. This presents a problem when computing the Euclidean distance, since we need to compute a square root. To bypass this problem, we use the square of the Euclidean distance (denoted d^2) instead: $d^2((a_i)_{i\in[\![n]\!]}, (b_i)_{i\in[\![n]\!]}) = \sum_{i=1}^{n}(a_i - b_i)^2$. Note that to still obtain a consistent result for the computation of anomaly and similarity, it suffices to consider the squares of the thresholds ϵ^2 and δ^2 instead of ϵ and δ.

We now provide an overview of the steps involved in the proof of anomaly detection denoted $\mathsf{MP.NIP}\{(K, x) : \mathsf{MP.Ope}(K, C, x) \wedge \mathsf{Ano}(x)\}$, then explain how to adapt this proof to other cases.

1. **Computation of the commitment of the** $(x_i - x_j)$**:** this is the first step in computing the square of the Euclidean distance between two windows. In order to achieve this, for all $i, j \in [\![n]\!]$ s.t. $j < i$, anybody can publicly evaluate the commitment $c_{i,j}$ of $x_i - x_j$ by computing c_i/c_j. Note that the prover can also evaluate the corresponding opening key $k_i - k_j$. Moreover, note that the value $x_i - x_j$ will be squared in the next step of the proof, and therefore it is not necessary to compute the values $c_{j,i}$, which can instead be set to $c_{j,i} = c_{i,j}$.
2. **Commitment of the** $(x_i - x_j)^2$**:** the prover picks a secret key $\widetilde{k}_{i,j} \in \mathbb{Z}^*_p$ and generates $\widetilde{c}_{i,j} = c_{i,j}^{(x_i - x_j)} h^{\widetilde{k}_{i,j}}$. This value is the Pedersen commitment of $(x_i - x_j)$ using the bases $(c_{i,j}, h)$, but it can also be seen as the commitment of $(x_i - x_j)^2$:

$$\begin{aligned}\widetilde{c}_{i,j} &= c_{i,j}^{(x_i - x_j)} h^{\widetilde{k}_{i,j}} = \left(g^{(x_i - x_j)} h^{(k_i - k_j)}\right)^{(x_i - x_j)} h^{\widetilde{k}_{i,j}} \\ &= g^{(x_i - x_j)^2} h^{((k_i - k_j)(x_i - x_j) + \widetilde{k}_{i,j})}\end{aligned}$$

In order to prove that this commitment $\widetilde{c}_{i,j}$ is well-formed, the prover uses a Schnorr proof to show that $\widetilde{c}_{i,j}$ commits the same message as $c_{i,j}$ (i.e. $(x_i - x_j)$) in the respective bases $(c_{i,j}, h)$ and (g, h).

3. **Computation of the commitment of the distance $d^2(s^x_{i,m}, s^x_{j,m})$:** remember that $d^2(s^x_{i,m}, s^x_{j,m}) = \sum_{r=0}^{m-1}(x_{i+r} - x_{j+r})^2$. We can therefore publicly evaluate the commitment of this value by simply computing $D_{i,j} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$. Note that the corresponding opening key, denoted $w_{i,j}$ in the following, can be reconstructed by the prover who knows the values x_{i+r}, k_{i+r}, and $\widetilde{k}_{i+r,j+r}$ for all $0 \leq r < m$.
4. **Bit commitments of binary-encoded distance:** let ℓ (defined in the setup set) denote the number of bits used to encode integers in binary, and let $d^2_u(s^x_{i,m}, s^x_{j,m})$ denote the u^{th} bit of binary-encoded $d^2(s^x_{i,m}, s^x_{j,m})$. The prover generate a commitment $D_{i,j,u}$ with an opening key $w_{i,j,u}$ of each $d^2_u(s^x_{i,m}, s^x_{j,m})$, and proves that each $D_{i,j,u}$ commits a bit (i.e. a message $\mathbf{d}_{i,j,u} = 0$ or $\mathbf{d}_{i,j,u} = 1$). All but one keys $w_{i,j,u}$ (for $u < \ell$) are chosen at random, and the last one $w_{i,j,\ell}$ is chosen in such a way that:

$$\prod_{u=1}^{\ell} D_{i,j,u}^{2^{u-1}} = D_{i,j}.$$

Since it only uses commitments, this relation can be verified by the verifier, and since each $w_{i,j,u}$ commits a bit $\mathbf{d}_{i,j,u}$ and $w_{i,j}$ commits the distance $d^2(s^x_{i,m}, s^x_{j,m})$, this ensures that, according to the homomorphism properties of the Pedersen commitment :

$$\sum_{u=1}^{\ell} \mathbf{d}_{i,j,u} \cdot 2^{u-1} = d^2(s^x_{i,m}, s^x_{j,m}).$$

In this way, the verifier verifies that the committed bits $\mathbf{d}_{i,j,u}$ are indeed the bits of the binary decomposition of the distance $d^2_u(s^x_{i,m}, s^x_{j,m})$.

5. **Comparisons of the distances $d^2(s^x_{i,m}, s^x_{j,m})$ with the squared threshold ϵ^2:** based on the commitments $D_{i,j,u}$ containing the binary encodings of the secret distances $d^2(s^x_{i,m}, s^x_{j,m})$, the prover proves the following relation:

$$\exists i \in \mathcal{I}, \forall j \in \mathcal{J}_i, d^2(s^x_{i,m}, s^x_{j,m}) > \epsilon^2.$$

To this end, the prover uses proofs of comparison of committed binary-encoded integers, following a method similar to that proposed in [3].

Note that in this overview, we omit the proof of knowledge of the message/key pair (x_1, k_1) that opens the commitment c_1, which is present in our construction. This proof ensures the extractability of our proof, since knowledge of x_1 (resp. k_1) coupled with knowledge of $(x_i - x_j)$ (resp. $(k_i - k_j)$) proven in Step 2 allows the extractor to recover all the x_i (resp. k_j).

For other cases (similarity detection, and absence of anomalies or similarities), only the relation proven in the last step changes, but these other relations

can also be proved in a similar way, by compiling proofs of comparison on committed data.

We emphasize that the choice of parameter ℓ depends on the threshold ϵ and the domain size of the time series values and must be chosen carefully. Let $\mathbf{x} > \max_{i,j\in[\![n]\!]}(|x_i - x_j|)$ be an upper bound depending on the values of the time series. We deduce that for all i and j, the squared Euclidian distance verifies $d^2(s^x_{i,m}, s^x_{j,m}) = \sum_{r=0}^{m-1}(x_{i+r} - x_{j+r})^2 < m \cdot \mathbf{x}^2$. Since ℓ must allow to encode both $d^2(s^x_{i,m}, s^x_{j,m})$ and ϵ^2 for comparison, we must therefore set $\ell > \log_2(\max(m \cdot \mathbf{x}^2, \epsilon^2))$.

3.3 Proof of Anomaly/Similarity Detection

We now give the complete proof/verification algorithms. The instantiation of non-interactive proofs will be specified later in Section 4 of the paper.

$\underline{\mathsf{MP.NIP}\{(K,x) : \mathsf{MP.Ope}(K,C,x) \wedge \mathsf{Ano}(x)\}}$**:** this algorithms parses K (resp. C and x) as $(k_i)_{i\in[\![n]\!]}$ (resp. $(c_i)_{i\in[\![n]\!]}$ and $(x_i)_{i\in[\![n]\!]}$), and sets $\mathcal{I} = [\![n-m+1]\!]$ and $\mathcal{J}_i = \{j \in \mathcal{I} : |j-i| > m/2\}$ for all $i \in \mathcal{I}$.

1. For each $i, j \in [\![n]\!]$ s.t. $j < i$, this algorithm sets $c_{i,j} = c_i/c_j$; $c_{j,i} = c_{i,j}$; $x_{i,j} = x_i - x_j$; $x_{j,i} = x_{i,j}$; $k_{i,j} = k_i - k_j$ and $k_{j,i} = k_{i,j}$.
2. For each $i, j \in [\![n]\!]$ s.t. $j < i$, this algorithm picks $\widetilde{k}_{i,j} \xleftarrow{\$} \mathbb{Z}^*_p$, computes $\widetilde{c}_{i,j} = \mathsf{Com}_{c_{i,j},h}(x_{i,j}, \widetilde{k}_{i,j})$ and sets $\widetilde{c}_{j,i} = \widetilde{c}_{i,j}$. It generates the following proof:
$$\pi^{\mathsf{sq}}_{i,j} \leftarrow \mathsf{NIP}\left\{(k_{i,j}, \widetilde{k}_{i,j}, x_{i,j}) : \begin{array}{c}\mathsf{Ope}(k_{i,j}, c_{i,j}, x_{i,j}) \\ \wedge\, \mathsf{Ope}_{c_{i,j},h}(\widetilde{k}_{i,j}, \widetilde{c}_{i,j}, x_{i,j})\end{array}\right\}.$$
3. For each $i \in \mathcal{I}$ and $j \in \mathcal{J}_i$ s.t. $j < i$, this algorithm computes the squared distance $\mathbf{d}_{i,j} = d(s^x_{i,m}, s^x_{j,m})^2$ and the key:
$$w_{i,j} = \sum_{r=0}^{m-1} x_{i+r,j+r} \cdot k_{i+r,j+r} + \widetilde{k}_{i+r,j+r}.$$
It sets $\mathbf{d}_{j,i} = \mathbf{d}_{j,i}$; $w_{j,i} = w_{i,j}$ and sets $D_{i,j} = \mathsf{Com}(\mathbf{d}_{i,j}, w_{i,j})$ and $D_{j,i} = D_{i,j}$. Note that this commitment verifies $D_{i,j} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$.
4. For each $i \in \mathcal{I}$, $j \in \mathcal{J}_i$ s.t. $j < i$ and each $u \in [\![\ell - 1]\!]$, this algorithm picks $w_{i,j,u} \xleftarrow{\$} \mathbb{Z}^*_p$, sets $w_{j,i,u} = w_{i,j,u}$ and sets:
$$w_{i,j,\ell} = \left(w_{i,j} - \sum_{u=1}^{\ell-1} w_{i,j,u} \cdot 2^{u-1}\right) 2^{-(\ell-1)}.$$
For each $i \in \mathcal{I}$, $j \in \mathcal{J}_i$ s.t. $j < i$, and $u \in [\![\ell]\!]$, it sets the commitments $D_{i,j,u} = \mathsf{Com}(\mathbf{d}_{i,j,u}, w_{i,j,u})$ and $D_{j,i,u} = D_{i,j,u}$ where $\mathbf{d}_{i,j,u}$ denotes the u^{th} bit of $\mathbf{d}_{i,j}$, and generates the following proof:
$$\pi^{\mathsf{bin}}_{i,j,u} \leftarrow \mathsf{NIP}\left\{w_{i,j,u} : \mathsf{Ope}(w_{i,j,u}, D_{i,j,u}, 0) \vee \mathsf{Ope}(w_{i,j,u}, D_{i,j,u}, 1)\right\}.$$
Note that these commitments verify $\prod_{u=1}^{\ell} D_{i,j,u}^{2^{u-1}} = D_{i,j}$.

5. This algorithm sets the witness $\omega = (w_{i,j,u}, \mathbf{d}_{i,j,u})_{i\in\mathcal{I}, j\in\mathcal{J}_i, u\in[\![\ell]\!]}$ and generates the following proof:

$$\pi^{\mathsf{cmp}} \leftarrow \mathsf{NIP}\left\{\omega : \bigvee_{i\in\mathcal{I}}\left(\bigwedge_{j\in\mathcal{J}_i} \mathbf{d}_{i,j} > \epsilon^2\right) \wedge \bigwedge_{\substack{i\in\mathcal{I},\\ j\in\mathcal{J}_i,\\ u\in[\![\ell]\!]}} \mathsf{Ope}(w_{i,j,u}, D_{i,j,u}, \mathbf{d}_{i,j,u})\right\}.$$

Finally, this algorithm generates $\pi_1^{\mathsf{com}} \leftarrow \mathsf{NIP}\{(x_1, k_1) : \mathsf{Ope}(k_1, c_1, x_1)\}$ and returns:

$$\pi = ((\widetilde{c}_{i,j}, \pi_{i,j}^{\mathsf{sq}})_{i,j\in[\![n]\!];j<i}, (D_{i,j,u}, \pi_{i,j,u}^{\mathsf{bin}})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}, \pi^{\mathsf{cmp}}, \pi_1^{\mathsf{com}}).$$

$\underline{\mathsf{MP.Verify}(C,\pi)}$**:** this algorithm parses the commitment C as $(c_i)_{i\in[\![n]\!]}$ and the proof π as $((\widetilde{c}_{i,j}, \pi_{i,j}^{\mathsf{sq}})_{i,j\in[\![n]\!];j<i}, (D_{i,j,u}, \pi_{i,j,u}^{\mathsf{bin}})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}, \pi^{\mathsf{cmp}}, \pi_1^{\mathsf{com}})$. For each $i, j \in [\![n]\!]$ such that $j < i$, it sets $c_{i,j} = c_i/c_j$ and $c_{j,i} = c_{i,j}$, and for each $i \in \mathcal{I}$ and $j \in \mathcal{J}_i$ s.t. $j < i$, it computes $D_{i,j} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$ and sets $D_{j,i} = D_{i,j}$. If all the proofs $\pi_{i,j}^{\mathsf{sq}}$, $\pi_{i,j,u}^{\mathsf{bin}}$, π^{cmp} and π_1^{com} are valid, and if $D_{i,j} = \prod_{u=1}^{\ell}(D_{i,j,u})^{2^{u-1}}$ for all $i \in \mathcal{I}$ and all $j \in \mathcal{J}_i$, it accept the proof and returns 1. Otherwise, it rejects the proof and returns 0.

For the other cases $\neg\mathsf{Ano}(x)$, $\mathsf{Sim}(x)$ and $\neg\mathsf{Sim}(x)$, the first four prove steps are identical. In the last step, only the first part of the proven relation in π^{cmp} needs to be adapted, and will express respectively:

$$\bigwedge_{i\in\mathcal{I}}\left(\bigvee_{j\in\mathcal{J}_i} \mathbf{d}_{i,j} < \epsilon^2 + 1\right); \bigvee_{i\in\mathcal{I}}\left(\bigvee_{j\in\mathcal{J}_i} \mathbf{d}_{i,j} < \delta^2\right); \text{and } \bigwedge_{i\in\mathcal{I}}\left(\bigwedge_{j\in\mathcal{J}_i} \mathbf{d}_{i,j} > \delta^2 - 1\right).$$

Furthermore, we show in appendix A that our construction can be adapted to cases comparing two time series with minor modifications.

3.4 Security of Our Protocol

Our commitment and non-interactive proof verify the following properties:

Theorem 1. *The commitment scheme* $(\mathsf{MP.Com}, \mathsf{MP.Ope})$ *is binding under the discrete logarithm assumption and perfectly hiding. The non-interactive anomaly detection (resp. similarity detection, absence of anomaly, and absence of similarity) proof* $(\mathsf{MP.NIP}, \mathsf{MP.Verify})$ *is extractable and perfectly zero-knowledge when instantiated with the Pedersen commitment and extractable and perfect zero-knowledge non-interactive proofs.*

Proof (Sketch). The commitment scheme trivially inherits the properties of Pedersen's commitment (it is a vector of perfectly hidden commitments). The proof

can be simulated by choosing the commitments randomly (they are perfectly hidden) and simulating all the sub-proofs used (they are perfectly zero-knowledge). Finally, the extraction algorithm consists of extracting all the witnesses used for all the sub-proofs. Note that the proofs $\pi_{i,j}^{\mathsf{sq}}$ allow the extractor to extract the values $x_j - x_i$ and $k_j - k_i$, but not the exact values x_i and k_i for all i. This is why we also need the proof π_1^{com}, which allows the extractor to extract x_1 and k_1, and thus, by recursion, to find all the other values from the $x_j - x_i$ and $k_j - k_i$. Any extraction that is inconsistent with the other elements would allow certain commitments to be opened with alternative values and keys, which is computationally impossible since Pedersen's commitment is binding under the discrete logarithm assumption.

4 Instantiation

In this section, we describe how we instantiate the NIP we use. All of these are derived from Schnorr-based sigma protocol [18].

Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p. The Schnorr sigma protocol allows a prover to prove to a verifier that they know the discrete logarithm x in basis g of an element $y \in \mathbb{G}$. It proceeds in three phases: the prover sends to the verifier a commitment R, then the verifier sends a random challenge ch, finally the prover returns a response z to the verifier. This proof (as any sigma protocol) can be made non-interactive with the Fiat-Shamir heuristic [10], also known as the random oracle model, by choosing the challenge as the hash of the public parameters and the commitment. Let us now consider a Pedersen commitment $c = g^x h^k$. Proving that the knowledge of a key such that $\mathsf{Ope}(k, c, x) = 1$ for a known message x is therefore equivalent to proving that $(c/g^x) = h^k$, i.e., proving knowledge of the discrete logarithm k of (c/g^x) in base h. Thus, the proof $\mathsf{NIP}\{k : \mathsf{Ope}(k, c, x)\}$ can be directly implemented by the Schnorr protocol.

Sigma protocols can be combined generically to obtain a (sigma protocol) proof of knowledge of a secret witness verifying multiple relations [5] (which we will call AND-proof), and proof of knowledge of a secret verifying one relation among several [7] (which we will call OR-proof). These generic transformation are recalled in Appendix B.1.

Instantiation of the proofs $\boldsymbol{\pi_1^{\mathsf{com}}}$ and $\boldsymbol{\pi^{\mathsf{sq}}}$. The proof π_1^{com} can be expressed as $\mathsf{NIP}\{x, k : \mathsf{Ope}(k, c, x)\}$, and is therefore equivalent to proof $\mathsf{NIP}\{x, k : c = g^x h^k\}$. This proof can be achieved by slightly modifying the Schnorr protocol, as shown in [4]. Let us now consider two Pedersen commitment of the same value $c_1 = g_1^x h^{k_1}$ and $c_2 = g_2^x h^{k_2}$. Similarly, the proof π^{sq} can be expressed as $\mathsf{NIP}\{(k_1, k_2, x) : c_1 = g_1^x h^{k_1} \wedge c_2 = g_2^x h^{k_2}\}$, and can be achieved by running two iterations of the previous protocol for the relations $c_1 = g_1^x h^{k_1}$ and $c_2 = g_2^x h^{k_2}$ using the same challenge to prove knowledge of x, according to the method described in [5]. We build the corresponding protocol in Appendix B.1 and we provide the proof of zero-knowledge and extractability properties in Appendix C.1.

Instantiation of the proofs π^{bin}. We have already seen that proving the opening of a commitment on a value is a Schnorr proof. Each π^{bin} is therefore an OR-proof on the original Schnorr protocol.

Instantiation of the proofs π^{cmp}. The final proof π^{cmp} in the cases $\mathsf{Ano}(x)$ or $\neg\mathsf{Sim}(x)$ (resp. $\neg\mathsf{Ano}(x)$ or $\mathsf{Sim}(x)$) consists of combining *comparison* proofs that values d_u committed in Pedersen commitments correspond to the binary encoding of an integer $\mathbf{d}$ larger (resp. smaller) than a public integer α using AND/OR-proofs (where $\alpha = \epsilon^2$ or $\alpha = \delta^2 - 1$, resp. $\alpha = \epsilon^2 + 1$ or $\alpha = \delta^2$). These comparison proofs can be obtained naively by showing that there exists an index u_* such that $\mathsf{d}_{u_*} = 1$, $\alpha_{u_*} = 0$, and $\mathsf{d}_u = \alpha_u$ for all $u > u_*$ (resp. such that $\mathsf{d}_{u_*} = 0$, $\alpha_{u_*} = 1$, and $\mathsf{d}_u = \alpha_u$ for all $u > u_*$ for the case $\mathbf{d} < \alpha$). This method is quadratic in ℓ.

For the case $\mathbf{d} < \alpha$, a more efficient method (i.e. linear in ℓ) for a similar concern is given in [3]. It consists of scanning the bits $\mathbf{d}_u$ (only once) from the most significant to the least significant and proving that:

- when $\alpha_u = 0$, then $\mathbf{d}_u = 0$ as well;
- when $\alpha_u = 1$, either $\mathbf{d}_u = 0$, which proves that $d < \alpha$, or (provided that it is possible) the scan continues.

In our case, since each bit $\mathbf{d}_u$ is committed in a commitment D_u with a key w_u, proving that $\mathbf{d}_u = 0$ consists of proving $\mathsf{Ope}(w_u, D_u, 0)$, which corresponds to the Schnorr protocol, as we already explained earlier.

The other case $\mathbf{d} > \alpha$ can be handled in a similar way. The proof then consists of scanning the bits $\mathbf{d}_u$ (only once) from the most significant to the least significant and proving that when $\alpha_u = 1$, then $\mathbf{d}_u = 1$ as well, and when $\alpha_u = 0$, either $\mathbf{d}_u = 1$, which proves that $d > \alpha$, or (provided that it is possible) the scan continues. In this case, showing $\mathbf{d}_u = 1$ means proving $\mathsf{Ope}(w_u, D_u, 1)$.

Finally, in the four cases $\mathsf{Ano}(x)$, $\mathsf{Sim}(x)$, $\neg\mathsf{Ano}(x)$ and $\neg\mathsf{Sim}(x)$, the proof π^{cmp} is instantiated by a combination of comparison proofs using AND-proof and OR-proof transforms. In all cases, the computational complexity of the proof/verification is $O(n^2\ell)$. Further details on these NIP are given in Appendix B.1.

Interactive vs. non-interactive proofs instantiation. Finally, note that our proof can be instantiated with the interactive version of the sub-proofs presented in this section, in order to make our scheme secure in the standard model. In this case, in order to optimize the number of interactions by limiting it to 3, the prover begins by computing the elements $\widetilde{c}_{i,j}$ and $D_{i,j,u}$, then performs all interactive proofs in parallel (all these proofs are sigma protocols and therefore require 3 interactions). The elements $\widetilde{c}_{i,j}$ and $D_{i,j,u}$ must be sent during the first flow (at the same time as the commitments). However, note that the result is only honest-verifier zero-knowledge. The proof can be made perfectly zero-knowledge even against dishonest verifiers in the standard model at the cost

of a factor λ in complexity, by choosing the challenges in $\{0, 1\}$ and running λ iterations of each proof in parallel.

5 Experiments

In this section, we describe how we instantiate the public parameters used in the proofs, and the experiment settings to evaluate the performances.

5.1 Parameters instantiation

We set n, the length of the private time series, to 10, 100, 200 and 1000, to observe how the runtime scales. $n = 1000$ is a realistic time series length using in ECG analysis [24,27]. We initiate randomly time series values x_i with positive integers in $[\![45]\!]$, a sufficient precision for raw ECG data (that we used to illustrate in Figure 1). As for the choice of window size m, it should be defined as the lower bound of any potential similarity or anomaly length. We use $m = 3$ for $n = 10$, $m = 10$ for $n = 100$ and $n = 200$, and $m = 100$ for $n = 1000$. These choices are motivated by physiological considerations: at the minimum clinically acceptable ECG sampling frequency (*i.e.,* 100 Hz [14]), the most visually prominent component of a heartbeat (the QRS complex) spans approximately 10 points, while a complete cardiac cycle covers roughly 100 points. For the bit length used to encode $\mathbf{d}_{i,j,u}$ values, we use $\ell = 13$, $\ell = 15$ and $\ell = 20$ for $m = 3$, $m = 10$ and $m = 100$, calculated with the formula described in Section 3. For the values of thresholds ϵ and δ, it often depends on the nature of the input time series values and the task. We dynamically set the thresholds to evaluate worst-case performance in each scenario while ensuring that the proofs remain valid under random inputs x. For the $\mathsf{Ano}(x)$ and $\neg\mathsf{Sim}(x)$ scenarios, the proof length depends on the least significant zero bit in the binary decomposition of ϵ^2 and $\delta^2 - 1$. To control this behavior, we choose even values of ϵ and odd values of δ, ensuring that the least significant zero bit in ϵ^2 and $\delta^2 - 1$ are located at index 1. In addition, to guarantee proof validity with arbitrary random time-series values, we set: (i) a sufficiently small threshold for $\mathsf{Ano}(x)$, and (ii) a value larger than the maximum possible distance $\mathbf{d}_{i,j}$ for $\neg\mathsf{Sim}(x)$, while remaining within the representable range defined by the bit-length ℓ. Conversely, when evaluating $\mathsf{Sim}(x)$ and $\neg\mathsf{Ano}(x)$, we select the smallest and largest odd values of $\epsilon^2 + 1$ and even values of δ^2 respectively. This ensures that the least significant non-zero bit is located at index 1, and that the claim remains valid after verification for any instantiation of the input x. The specific parameter values used in the experiments are reported in Table 7 (Appendix D).

5.2 Results

We implemented all proofs introduced in Section 3 in Rust[1]. We use the elliptic curve-based group Ristretto (of prime order $2^{255} - 19$) from the Dalek library.

[1] https://github.com/HaoyingZhang/ZKMP/tree/main

In addition, we use the SHA-2 hash function. Experiments were run on a laptop equipped with a 12-core 13th Gen Intel Core i7-1365U processor and 32 GB of RAM.

We first evaluate the device's signature phase, which consists of the MP.Com, Sign, and Verify algorithms. The runtime measurements for these steps are reported in Table 2. Each step is efficient, with execution times bounded by 52.24 ms, 2.9 ms, and 3.0 ms, respectively, under real-world workloads. We then evaluate proof π_1^{com}, which incurs 74.3 μs for MP.NIP and 101.7 μs for MP.Verify. Note that both operations are independent of the time-series length n, as a proof of only one element of x is required. Table 3 reports the runtime of each individual proof described in Section 4. The corresponding total runtimes and communication costs for the four scenarios are then summarized in Tables 4 and 8 (Appendix D), respectively. For π^{sq} and π^{bin}, the prover must generate new commitments that are required by the verifier during the verification phase. The commitment time is shown separately in the parentheses; the values outside the parentheses then correspond to the sum of the commit time and the proof-generation time. As for the comparison proofs, some optimizations are used in the implementations. For the scenarios $\mathsf{Sim}(x)$ and $\neg\mathsf{Sim}(x)$, we invoke the unit proof π^{cmp} only once for each pair (i, j) with $i < j$, as the underlying distance function is symmetric. For the scenario $\neg\mathsf{Ano}(x)$, the set of proofs is generated independently for each value of i.

As described in the introduction, the scenarios we consider do not require strict time constraints, as both the proving and verification procedures can be executed in an offline setting. Among all the unit proofs, the main performance bottlenecks arise in π^{bin} and π^{cmp} proofs (in line with their theoretical complexity). The best performance comes from $\mathsf{Sim}(x)$ and $\neg\mathsf{Sim}(x)$ scenario, with a total proving time of around 18 seconds (respectively 1.5 minutes, 42 minutes) and verification time of around 11 seconds (respectively 1 minute, 25 minutes) for a time series of 100 points (respectively of 200 points, and 1,000 points). The worst case comes for $\mathsf{Ano}(x)$, since no optimization can be applied to it. Nevertheless, even for $n = 1,000$, the overall runtime remains practical, totaling approximately 50 minutes for MP.NIP and 33 minutes for MP.Verify. Finally, for the $\neg\mathsf{Ano}(x)$ scenario, we report approximately 46 minutes in total proving time and 29 minutes in total verification time, with time series of 1,000 points. These results are consistent with the expected asymptotic complexity, and a complete analysis of the asymptotic complexity of our algorithms is provided in Appendix B.2.

Table 2. Runtime of commit, sign and verify for C (in milliseconds)

n	MP.Com	Sign	Ver
10	0.50	0.06	0.09
100	4.94	0.31	0.33
200	9.89	0.59	0.60
1000	52.24	2.90	3.00

Table 3. Detailed runtime for all proofs (in seconds)

	MP.NIP (**Where Commit**)						MP.Verify					
n	π^{sq}	π^{bin}	$\pi^{\mathsf{cmp}}_{\mathsf{Ano}}$	$\pi^{\mathsf{cmp}}_{\neg\mathsf{Ano}}$	$\pi^{\mathsf{cmp}}_{\mathsf{Sim}}$	$\pi^{\mathsf{cmp}}_{\neg\mathsf{Sim}}$	π^{sq}	π^{bin}	$\pi^{\mathsf{cmp}}_{\mathsf{Ano}}$	$\pi^{\mathsf{cmp}}_{\neg\mathsf{Ano}}$	$\pi^{\mathsf{cmp}}_{\mathsf{Sim}}$	$\pi^{\mathsf{cmp}}_{\neg\mathsf{Sim}}$
10	8e-3 (3e-3)	0.07 (0.04)	0.02	0.01	0.02	0.02	6e-3	0.03	0.02	0.01	0.02	0.01
100	0.8 (0.3)	13.5 (6.6)	6.8	5.9	3.2	3.5	0.8	6.9	6.7	5.7	3.2	3.0
200	3.7 (1.1)	64.9 (29.9)	33.5	27.5	15.9	16.6	3.4	36.1	31.4	27.2	15.4	14.3
1000	89.3 (26.2)	1.9e3 (981.3)	991.3	687.8	448.8	532.8	89.4	953.8	935.3	700.2	433.8	460.7

Table 4. Total runtime for the four scenarios (in seconds)

	MP.NIP				MP.Verify			
n	$\mathsf{Ano}(x)$	$\neg\mathsf{Ano}(x)$	$\mathsf{Sim}(x)$	$\neg\mathsf{Sim}(x)$	$\mathsf{Ano}(x)$	$\neg\mathsf{Ano}(x)$	$\mathsf{Sim}(x)$	$\neg\mathsf{Sim}(x)$
10	0.10	0.97	0.10	0.10	0.05	0.05	0.05	0.05
100	21.1	20.2	17.5	17.8	14.4	13.4	10.9	10.7
200	102.1	96.1	84.5	85.2	70.9	66.7	54.9	53.8
1000	3014.6	2717.1	2478.1	2562.1	1978.5	1743.4	1477.0	1503.9

6 Related Works

A line of work considers privacy-preserving similarity on time series. Zhu *et al.* [29] proposed a two-party protocol that privately evaluates Dynamic Time Warping (DTW) and discrete Fréchet distance between two time series using partially homomorphic encryption (HE). Liu and Yi [15] proposed a collaborative medical analytics system based on DTW that combines multiparty computation (MPC) to enable privacy-preserving DTW queries over distributed electrocardiogram (ECG) and photoplethysmogram (PPG) data. Zheng *et al.* [28] introduced a solution supporting similarity queries over time series of different lengths by using time-warp edit distance (TWED) as the similarity metric and symmetric HE.

However, none of these methods employ the MP, which supports subsequence-level similarity and anomaly detection, with or without a reference time series. Second, some of these applications [29,15] consider a different setting in which several independent data providers want to determine a consensus without revealing their raw time series. Other works around similarity query on private data [28] are mostly based on MPC which does not allow for a non-interactive protocol, or the prover only sends an element that can be verified later. Finally, generic cryptographic tools could be used to construct a protocol similar to ours, such as using generic Zero-Knowledge (ZK) proofs or MPC. However, these techniques typically require expressing the computation as Boolean or arithmetic circuits, whose construction is non-trivial and often incurs significant

overhead, tending to be less efficient than protocols tailored to a specific structure. Some generic ZK proofs (*e.g.,* ZK-snarks [13,11]) are non-interactive and allow the generation of proofs of constant size, but their proving and verification times scale with the size of the proven circuit. In principle, such proofs could serve as an alternative to our work, allowing less data to be exchanged. However, these proofs require less standard tools (*e.g.,* pairings), trusted setups (e.g. strong common reference string model (CRS)), and rely on strong assumptions (*e.g.,* q-SDH) and strong heuristic security models (*e.g.,* algebraic/generic group model (AGM/GGM) and random oracle model (ROM)). In comparison, as our non-interactive proof is a combination of a Pedersen commitment and Schnorr proofs, it does not require any trusted setup, relies only on the discrete logarithm assumption, and, in its non-interactive version, on the ROM (which is well-understood and more established heuristic model compared to AGM/GGM).

Another line of work examines the privacy issues related to the Matrix Profile (MP). Introduced broadly to the community in 2016 by Keogh and colleagues [24], the MP has become a standard tool for time-series mining, particularly for detecting pairwise anomalies and similarities. Recently, some studies have claimed that the Matrix Profile provides certain privacy-preserving properties, such as robustness against sensitive-attribute inference [9] or reconstruction attacks [26], often arguing that the non-linear *min* operation involved in its computation is not reversible. However, Zhang *et al.* [25] recently demonstrated that the MP is actually vulnerable to singling-out, linkability, attribute inference, and reconstruction attacks. By solving the MP constraints using a powerful optimizer, they show that an attacker can reconstruct the original time series with up to 0.99 Pearson correlation, revealing that MP does not inherently guarantee privacy. Given that MP is frequently applied to sensitive domains such as healthcare data [21,12] and geo-location time series [23], we therefore treat the values of the MP as sensitive and protected outputs in our threat model.

7 Conclusion and future works

This paper presents a secure protocol for proving claims about anomalies and similarities detected using Matrix Profile in private time series, a challenging and often overlooked scenario in the literature. The protocol relies on Pedersen commitments and Schnorr-based non-interactive zero-knowledge proofs. We consider a strong adversarial model in which the private time series itself, the pairwise subsequence distances, the corresponding Matrix Profile values, as well as the indices of potential anomalies and similarities are all treated as sensitive information and must not be disclosed. Under these stringent constraints, we design non-interactive zero-knowledge proofs and evaluate their performance on real-world time series with lengths of up to 1,000 data points, demonstrating the practicality of our approach for real-world applications. Finally, we believe that our scheme can also be adapted to other related tasks, such as authentication, by reusing the similarity proof and verification components.

Acknowledgments. This work was supported by the ANR France 2030 project TracIA (22-PESN-0006), the ANR France 2030 project CyberINSA (23-CMAS-0019) and the ANR project PRIVA-SIQ (ANR-23-CE39-0008).

References

1. Bijlani, N., Maldonado, O.M., Kouchaki, S.: G-cmp: Graph-enhanced contextual matrix profile for unsupervised anomaly detection in sensor-based remote health monitoring. arXiv preprint arXiv:2211.16122 (2022)
2. Blum, M., Feldman, P., Micali, S.: Non-Interactive Zero-Knowledge and Its Applications. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing. p. 103–112. STOC '88, Association for Computing Machinery, New York, NY, USA (1988). https://doi.org/10.1145/62212.62222, `https://doi.org/10.1145/62212.62222`
3. Bultel, X., Olivier-Anclin, C.: Taming delegations in anonymous signatures: k-times anonymity for proxy and sanitizable signature. In: Kohlweiss, M., Di Pietro, R., Beresford, A. (eds.) Cryptology and Network Security. pp. 165–186. Springer Nature Singapore, Singapore (2025)
4. Camenisch, J., Stadler, M.: Proof systems for general statements about discrete logarithms (1997), `https://api.semanticscholar.org/CorpusID:15336229`
5. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) Advances in Cryptology — CRYPTO' 92. pp. 89–105. Springer Berlin Heidelberg, Berlin, Heidelberg (1993)
6. Christov, I.I.: Real time electrocardiogram qrs detection using combined adaptive threshold. Biomedical engineering online **3**(1), 28 (2004)
7. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) Advances in Cryptology — CRYPTO '94. pp. 174–187. Springer Berlin Heidelberg, Berlin, Heidelberg (1994)
8. Dau, H.A., Bagnall, A., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Keogh, E.: The ucr time series archive. IEEE/CAA Journal of Automatica Sinica **6**(6), 1293–1305 (2019)
9. Der, A., Yeh, C.C.M., Zheng, Y., Wang, J., Chen, H., Zhuang, Z., Wang, L., Zhang, W., Keogh, E.: Time series synthesis using the matrix profile for anonymization. In: 2023 IEEE International Conference on Big Data (BigData). pp. 1908–1911. IEEE (2023)
10. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) Advances in Cryptology — CRYPTO' 86. pp. 186–194. Springer Berlin Heidelberg, Berlin, Heidelberg (1987)
11. Gabizon, A., Williamson, Z.J., Ciobotaru, O.M.: Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. IACR Cryptol. ePrint Arch. **2019**, 953 (2019), `https://api.semanticscholar.org/CorpusID:201685538`
12. Goldberger, A.L., Amaral, L.A., Glass, L., Hausdorff, J.M., Ivanov, P.C., Mark, R.G., Mietus, J.E., Moody, G.B., Peng, C.K., Stanley, H.E.: Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. circulation **101**(23), e215–e220 (2000)
13. Groth, J.: On the size of pairing-based non-interactive arguments. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 305–326. Springer (2016)

14. Kwon, O., Jeong, J., Kim, H.B., Kwon, I.H., Park, S.Y., Kim, J.E., Choi, Y.: Electrocardiogram sampling frequency range acceptable for heart rate variability analysis. Healthcare informatics research **24**(3), 198–206 (2018)
15. Liu, X., Yi, X.: Privacy-preserving collaborative medical time series analysis based on dynamic time warping. In: European Symposium on Research in Computer Security. pp. 439–460. Springer (2019)
16. Paparrizos, J., Kang, Y., Boniol, P., Tsay, R.S., Palpanas, T., Franklin, M.J.: Tsb-uad: an end-to-end benchmark suite for univariate time-series anomaly detection. Proceedings of the VLDB Endowment **15**(8), 1697–1711 (2022)
17. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Advances in Cryptology—CRYPTO'91: Proceedings. pp. 129–140. Springer (2001)
18. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) Advances in Cryptology — CRYPTO' 89 Proceedings. pp. 239–252. Springer New York, New York, NY (1990)
19. Shakibay Senobari, N., Zimmerman, Z., Funning, G., Shearer, P.M., Brisk, P., & Keogh, E.: Using the matrix profile to detect seismic events – from the lab experiment scale to local and global scales. In: SCEC Annual Meeting (2019)
20. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs. IACR Cryptology ePrint Archive **2004**, 332 (01 2004)
21. Wankhedkar, R., Jain, S.K.: Motif discovery and anomaly detection in an ecg using matrix profile. In: Progress in Advanced Computing and Intelligent Engineering: Proceedings of ICACIE 2019, Volume 1. pp. 88–95. Springer (2021)
22. Yang, J.J., Buu, A.: Efficient matrix profile computation with euclidean distance using eigen transformation: Performance evaluation based on beat-to-beat interval (bbi) data. Statistics in medicine **43**(16), 3051–3061 (2024)
23. Yeh, C.C.M., Der, A., Saini, U.S., Lai, V., Zheng, Y., Wang, J., Dai, X., Zhuang, Z., Fan, Y., Chen, H., Aboagye, P.O., Wang, L., Zhang, W., Keogh, E.: Matrix profile for anomaly detection on multidimensional time series. In: 2024 IEEE International Conference on Data Mining (ICDM). pp. 911–916 (2024). https://doi.org/10.1109/ICDM59182.2024.00114
24. Yeh, C.C.M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H.A., Silva, D.F., Mueen, A., Keogh, E.: Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In: 2016 IEEE 16th international conference on data mining (ICDM). pp. 1317–1322. Ieee (2016)
25. Zhang, H., Anciaux, N., Nguyen, B., Girard, F., de Fuentes, J.M., Boiret, A.: Privacy attacks on matrix profiles via reconstruction techniques. In: Privacy Enhancing Technologies Symposium. vol. 2026, p. 65–86 (2026)
26. Zhang, L., Ding, J., Gao, Y., Lin, J.: Pmp: Privacy-aware matrix profile against sensitive pattern inference for time series. In: Proceedings of the 2023 SIAM International Conference on Data Mining (SDM). pp. 891–899. SIAM (2023)
27. Zhao, P., Yang, J.J., Buu, A.: Applied statistical methods for identifying features of heart rate that are associated with nicotine vaping. The American journal of drug and alcohol abuse **51**(2), 165–172 (2025)
28. Zheng, Y., Lu, R., Guan, Y., Shao, J., Zhu, H.: Efficient and privacy-preserving similarity range query over encrypted time series data. IEEE Transactions on Dependable and Secure Computing **19**(4), 2501–2516 (2021)
29. Zhu, H., Meng, X., Kollios, G.: Privacy preserving similarity evaluation of time series data. In: EDBT. vol. 2014, pp. 499–510 (2014)

A Alternative of MP.NIP for Comparing two Time-series

The following Appendix propose a variation of our construction that considers two time series: x, a private time series, and $\hat{x}$, a public time series. Our aim is to enable the prover to prove presence or absence of anomaly/similarity in its time series relatives to a reference time series.

We begin by recalling the definition of Matrix Profile and Anomaly and Similarity detection function for two time series.

Definition 7 (Matrix Profile [24]). *Let x be an input time series and $\hat{x}$ a reference time series of lengths n and $\hat{n}$, respectively, let d be a distance function, let m be a window size and let $\mathcal{J}$ be the set $[\![\hat{n}-m+1]\!]$. Matrix Profile (MP) is a data mining structure composed of two vectors: Matrix Profile Distance (MPD) and Matrix Profile Index (MPI) s.t. :*

$$\mathsf{MPD}(x,\hat{x}) = (\mathsf{MPD}_i(x,\hat{x}))_{i\in\mathcal{I}} = \left(\min_{j\in\mathcal{J}}\big(d(s^x_{i,m}, s^{\hat{x}}_{j,m})\big)\right)_{i\in\mathcal{I}}$$

and $\mathsf{MPI}(x,\hat{x}) = (\mathsf{MPI}_i(x,\hat{x}))_{i\in\mathcal{I}}$ where $\mathsf{MPI}_i(x,\hat{x})$ stores the (minimum) index $j \in \mathcal{J}$ verifying $\mathsf{MPD}_i(x,\hat{x}) = d(s^x_{i,m}, s^{\hat{x}}_{j,m})$.

Definition 8 (Anomaly and Similarity detection function). *Let x and $\hat{x}$ be two times series. The anomaly detection function $\mathsf{Ano} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{B}$ (resp., the similarity detection function $\mathsf{Sim} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{B}$) on input x and $\hat{x}$, returns 1 if there exists a value in MP bigger than (resp. smaller than) a predefined threshold ϵ, (resp. δ), and 0 otherwise. That is,*

$$\mathsf{Ano}(x,\hat{x}) = \begin{cases} 1 & \text{if } \exists v \in \mathsf{MP}(x,\hat{x}), v > \epsilon, \\ 0 & \text{if } \forall v \in \mathsf{MP}(x,\hat{x}), v \leq \epsilon. \end{cases} \tag{1}$$

$$\mathsf{Sim}(x,\hat{x}) = \begin{cases} 1 & \text{if } \exists v \in \mathsf{MP}(x,\hat{x}) \text{ s.t. } v < \delta, \\ 0 & \text{if } \forall v \in \mathsf{MP}(x,\hat{x}),\ v \geq \delta. \end{cases} \tag{2}$$

We describe the modifications for the case $\mathsf{Ano}(x,\hat{x})$, and for simplicity we omit the other cases which are analogous. Although the scheme remains unchanged for the algorithms Setup and MP.Com, however, the algorithm MP.NIP differs in the first three steps. In the following, we present the modified steps, omitting the remainder of the construction which follows the step 4 and 5 of the Section 3.

1. **Computation of the commitment of the $(x_i - \hat{x}_j)^2$:** the prover picks a secret key $\widetilde{k}_i \in \mathbb{Z}_p^*$ and generates $\widetilde{c}_i = c_i^{x_i} h^{\widetilde{k}_i}$, a Pedersen commitment of x_i^2 using the basis (c_i, h). As $\hat{x}$ is public, one can easily deduce a commitment of $(x_i - \hat{x}_j)^2$ by considering the identity $x_i^2 - 2x_i\hat{x}_j + \hat{x}_j^2$, it follows that $\widetilde{c}_{i,j} = \widetilde{c}_i\,(c_i)^{-2\hat{x}_j}\, g^{\hat{x}_j^2}$. Additionally, the prover proves that $\widetilde{c}_i$ commits the same message as c_i on their respective bases.

2. **Computation of the commitment of the distance $d^2(s^x_{i,m}, s^{\hat{x}}_{j,m})$:** The commitments of the distances can be publicly derived via the following computation: $D_{i,j} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$, and associated opening key $w_{i,j}$ can be determined by the prover given the values x_{i+r}, $\hat{x}_{j+r}$, k_{i+r} and $\widetilde{k}_{i+r}$ for $0 \leq r < m$.

$\underline{\mathsf{MP.NIP}\{(K,x) : \mathsf{MP.Ope}(K,C,x) \wedge \mathsf{Ano}(x,\hat{x})\}}$**:** this algorithm parses K (resp. C and x) as $(k_i)_{i\in[\![n]\!]}$ (resp. $(c_i)_{i\in[\![n]\!]}$ and $(x_i)_{i\in[\![n]\!]}$), and sets $\mathcal{I} = [\![n-m+1]\!]$ and $\mathcal{J} = [\![\hat{n}-m+1]\!]$.

1. For each $i \in [\![n]\!]$, this algorithm picks $\widetilde{k}_i \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $\widetilde{c}_i = \mathsf{Com}_{c_i,h}(x_i, \widetilde{k}_i)$, and generates the following proof:

$$\pi_i^{\mathsf{sq}} \leftarrow \mathsf{NIP}\left\{(k_i, \widetilde{k}_i, x_i) : \mathsf{Ope}(k_i, c_i, x_i) \wedge \mathsf{Ope}_{c_i,h}(\widetilde{k}_i, \widetilde{c}_i, x_i)\right\}.$$

   For each $j \in [\![n]\!]$ s.t. $j < i$, this algorithm computes $\widetilde{c}_{i,j} = \widetilde{c}_i \, (c_i)^{-2\hat{x}_j} \, g^{\hat{x}_j^2}$ and sets $\widetilde{c}_{j,i} = \widetilde{c}_{i,j}$.
2. For each $i \in \mathcal{I}$ and $j \in \mathcal{J}$ s.t. $j < i$, this algorithm computes the squared distance $\mathbf{d}_{i,j} = d(s^x_{i,m}, s^{\hat{x}}_{j,m})^2$ and the key:

$$w_{i,j} = \sum_{r=0}^{m-1} (x_{i+r} - 2\hat{x}_{j+r}) k_{i+r} + \widetilde{k}_{i+r}.$$

   It computes $D_{i,j} = \mathsf{Com}(\mathbf{d}_{i,j}, w_{i,j})$. Note that this commitment verifies $D_{i,j} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$. It sets $\mathbf{d}_{j,i} = \mathbf{d}_{i,j}$ and $D_{j,i} = D_{i,j}$.

B Asymptotic Complexity

B.1 Complexity of the NIP building blocks

In this paper, we use several NIP as building blocks to construct the NIP produces by MP.NIP. Figure 3 describes the sigma protocol (which we denoted Open-proof) that allows a prover to prove the opening of a Pedersen commitment, Figure 4 shows the sigma protocol (which we denoted EQ-Open-proof) allowing to prove the knowledge of α the hidden value, and the opening keys β_1 and β_2 of two Pedersen commitments y_1 and y_2 s.t. the hidden value is the same for the two commitments. Furthermore, Figure 5 describes the sigma protocol for proving the knowledge of a set of discrete logarithms (in basis g_i) of elements y_i in the group $\mathbb{G}$ (AND-proof), and Figure 6 describes the sigma protocol which allows one to prove the knowledge of the discrete logarithm (in basis g_i) of one among several elements y_i in $\mathbb{G}$ (OR-proof). We notice that in OR-proof, for the relations for which the prover does not know the witness, the prover uses a simulator denoted Simulator that on input a statement (y, g), generates a response z and a challenge ch, and outputs $(g^u \cdot y^{-\mathsf{ch}}, \mathsf{ch}, z)$. Table 5 summarizes the asymptotic complexity of all NIP building blocks.

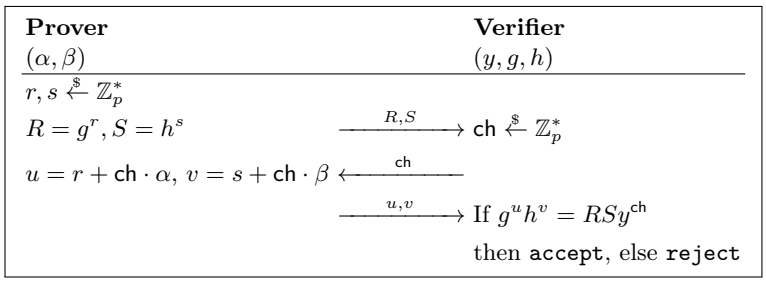

Fig. 3. Open-proof for the relation $y = g^\alpha h^\beta$.

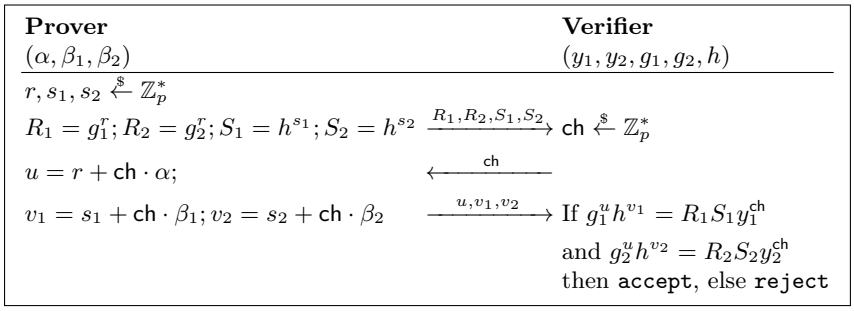

Fig. 4. EQ – Open-proof for the relation $y_1 = g_1^\alpha h^{\beta_1} \wedge y_2 = g_2^\alpha h^{\beta_2}$.

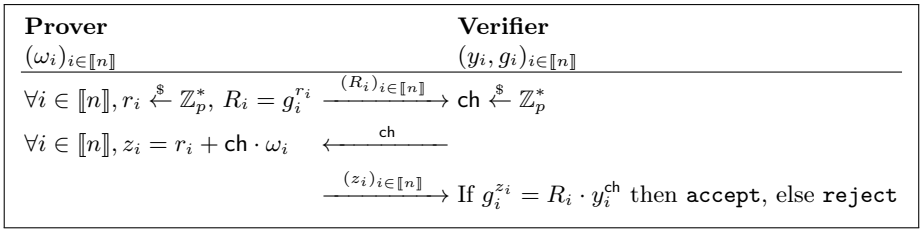

Fig. 5. AND-proof for the relation $\bigwedge_{i=1}^{n} y_i = g_i^{\omega_i}$.

Table 5. Asymptotic complexity of the NIP building blocks

NIP	Prover time	Verifier time
Fig. 3 (Open-proof)	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Fig.4(EQ-Open-proof)	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Fig.5 (AND-proof)	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Fig.6 (OR-proof)	$\mathcal{O}(n)$	$\mathcal{O}(n)$

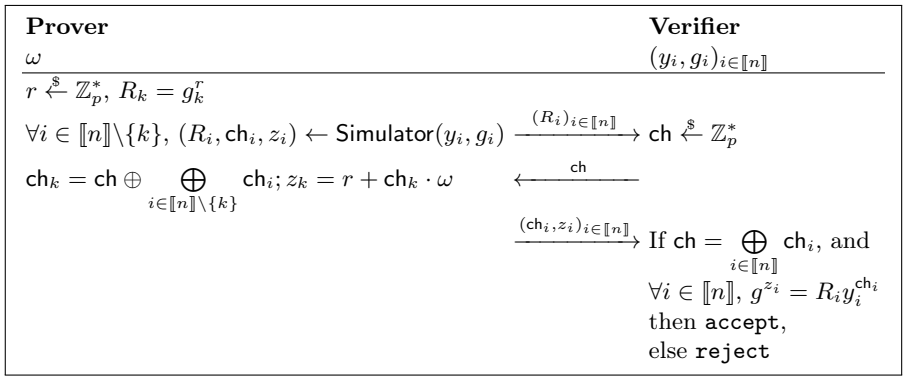

Fig. 6. OR-proof for the relation $\bigvee_{i=1}^{n} y_i = g_i^{\omega}$ (s.t. $\exists k \in [\![n]\!]$; $y_k = g_k^{\omega}$).

B.2 Asymptotic complexity of our construction

In Table 6, we compare the asymptotic complexity of the generation of the proof π and its verification. The complexity is evaluated in terms of the number of exponentiations in the group $\mathbb{G}$. We notice that the complexity of the proof π^{cmp} depends on the position of the least significant one bit (for the no anomaly and similarity cases) or least significant zero bit (for the anomaly and no similarity cases) in the binary decomposition of α, therefore, for this proof, we express the complexity in the worst case (when this position is equal to one). Furthermore, the running time of the proof generation/verification is not only determined by n, the number of values in the time series, but also by m, the window size, and ℓ, the number of bits in the binary decompositions. Some proofs are executed for $i \in \mathcal{I}$ and $j \in \mathcal{J}_i$ where the number of elements in each $\mathcal{J}_i$ depends on i. This results in a complexity that depends on the sum over all $i \in \mathcal{I}$ of the cardinality of $\mathcal{J}_i$ where each set $\mathcal{J}_i$ contains less than $n - m + 1$ elements. This gives us a quadratic asymptotic complexity in terms of the number n.

Table 6. Asymptotic complexity of the generation/verification of the proof π.

Element	Prover time	Verifier time
$(\widetilde{c}_{i,j})_{i,j\in[\![n]\!];j<i}$	$\mathcal{O}(n^2)$	–
$(\pi^{\mathsf{sq}})_{i,j\in[\![n]\!];j<i}$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
$(D_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!]}$	$\mathcal{O}\left(n^2\ell\right)$	–
$(\pi^{\mathsf{bin}}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!]}$	$\mathcal{O}\left(n^2\ell\right)$	$\mathcal{O}\left(n^2\ell\right)$
π^{cmp}	$\mathcal{O}\left(n^2\ell\right)$	$\mathcal{O}\left(n^2\ell\right)$
π	$\mathcal{O}\left(n^2\ell\right)$	$\mathcal{O}\left(n^2\ell\right)$

C Security Proofs

C.1 Proof that $\mathsf{EQ-Open}$-proof (Figure 4 in Appendix B.1) is perfectly zero-knowledge and extractable

In this section, we denote by $\mathcal{R}_{\mathrm{EQ-Open}}$ the relation $y_1 = g_1^{\alpha} h^{\beta_1} \wedge y_2 = g_2^{\alpha} h^{\beta_2}$ and we denote by (R, ch, z) the transcript for this relation.

Perfect zero-knowledge. Let $\mathsf{Simulator}_{\mathrm{EQ-Open}}$ be a simulator for the relation $\mathcal{R}$ defined as follows:
$\underline{\mathsf{Simulator}_{\mathrm{EQ-Open}}(y_1, y_2, g_1, g_2, h)}$: picks $\mathsf{ch} \xleftarrow{\$} \mathbb{Z}_p^*$, $u \xleftarrow{\$} \mathbb{Z}_p^*$, $v_1 \xleftarrow{\$} \mathbb{Z}_p^*$, $v_2 \xleftarrow{\$} \mathbb{Z}_p^*$, $S_1 \xleftarrow{\$} \mathbb{G}$ and $S_2 \xleftarrow{\$} \mathbb{G}$. Computes $R_1 = \frac{g_1^u h^{v_1}}{S_1 y_1^{\mathsf{ch}}}$ and $R_2 = \frac{g_2^u h^{v_2}}{S_2 y_2^{\mathsf{ch}}}$, and returns $((R_1, R_2, S_1, S_2), \mathsf{ch}, (u, v_1, v_2))$.

$\mathsf{NIP}\{(\alpha, \beta_1, \beta_2) : y_1 = g_1^{\alpha} h^{\beta_1} \wedge y_2 = g_2^{\alpha} h^{\beta_2}\}$ and $\mathsf{Simulator}_{\mathrm{EQ-Open}}(y_1, y_2, g_1, g_2, h)$ are identically distributed.

Extractability. Let (R, ch, z) and (R, ch', z') be two valid transcripts for the statement (y_1, y_2, g_1, g_2, h) s.t. $\mathsf{ch} \neq \mathsf{ch}'$. Parses R as (R_1, R_2, S_1, S_2), z as (u, v_1, v_2) and z' as (u', v_1', v_2'). Since $\mathsf{ch} \neq \mathsf{ch}'$, we can extract the witnesses as follows:
$y_1^{\mathsf{ch}-\mathsf{ch}'} = g_1^{u-u'} h^{v_1 - v_1'}$ implies that $y_1 = g_1^{\frac{u-u'}{\mathsf{ch}-\mathsf{ch}'}} h^{\frac{v_1 - v_1'}{\mathsf{ch}-\mathsf{ch}'}}$, and
$y_2^{\mathsf{ch}-\mathsf{ch}'} = g_2^{u-u'} h^{v_2 - v_2'}$ implies that $y_2 = g_2^{\frac{u-u'}{\mathsf{ch}-\mathsf{ch}'}} h^{\frac{v_2 - v_2'}{\mathsf{ch}-\mathsf{ch}'}}$.
Finally, $\alpha = \frac{u-u'}{\mathsf{ch}-\mathsf{ch}'}$, $\beta_1 = \frac{v_1 - v_1'}{\mathsf{ch}-\mathsf{ch}'}$ and $\beta_2 = \frac{v_2 - v_2'}{\mathsf{ch}-\mathsf{ch}'}$.

C.2 Proof of theorem 1

Proof that $(\mathsf{MP.Com}, \mathsf{MP.Ope})$ is perfectly hiding and computationally binding. The perfect hiding property (resp. the computational binding property) of $(\mathsf{MP.Com}, \mathsf{MP.Ope})$ results directly from the perfect hiding property (resp. the computational binding property) of the Pedersen commitment scheme.

Proof that $(\mathsf{MP.NIP}, \mathsf{MP.Verify})$ is perfectly zero-knowledge. Since π, the proof returned by the algorithm $\mathsf{MP.NIP}$, is assumed to be zero-knowledge, there exists a simulator that simulates it.

We begin with the observation that in our construction, there exists a publicly verifiable relation between the commitments $D_{i,j,u}$ and the $\widetilde{c}_{i+r,j+r}$:

$$\prod_{u=1}^{\ell} (D_{i,j,u})^{2^{u-1}} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}. \tag{3}$$

In order to achieve indistinguishability between a proof simulated by the simulator, that we denote by $\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}$ (for the anomaly case), and a proof generated by $\mathsf{MP.NIP}$, the above relation must be preserved in the simulator.

We denote by $\mathsf{Simulator}_{\mathsf{cmp}}$, the simulator for the relation:

$$\bigvee_{i\in\mathcal{I}}\left(\bigwedge_{j\in\mathcal{J}_i}\mathbf{d}_{i,j}>\epsilon^2\right)\wedge\bigwedge_{\substack{i\in\mathcal{I},\\ j\in\mathcal{J}_i,\\ u\in[\![\ell]\!]}}\mathsf{Ope}(w_{i,j,u},D_{i,j,u},\mathbf{d}_{i,j,u}),$$

by $\mathsf{Simulator}_{\mathsf{OR}}$, the simulator for the relation $\bigvee_{i=1}^{n} y_i = g_i^{\omega}$ and by $\mathsf{Simulator}_{\mathsf{Open}}$, the simulator for the relation $y = g^{\alpha}h^{\beta}$. We now define the simulator that we denote $\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{NIP.MP}}$ and which allows us to simulate the proof π outputted by the algorithm NIP.MP in the anomaly case.

$\underline{\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{NIP.MP}}(C)}$**:** this algorithms parses C as $(c_i)_{i\in[\![n]\!]}$, and sets $\mathcal{I}=[\![n-m+1]\!]$ and $\mathcal{J}_i=\{j\in\mathcal{I}:|j-i|>m/2\}$ for all $i\in\mathcal{I}$.

1. For each $i,j\in[\![n]\!]$ s.t. $j<i$, this algorithm sets $c_{i,j}=\frac{c_i}{c_j}$ and $c_{j,i}=c_{i,j}$.
2. For each $i,j\in[\![n]\!]$ s.t. $j<i$, this algorithm picks $\widetilde{c}_{i,j}\xleftarrow{\$}\mathbb{G}$, simulates the proof $\pi_{i,j}^{\mathsf{sq}}\leftarrow\mathsf{Simulator}_{\mathsf{EQ-Open}}(c_{i,j},\widetilde{c}_{i,j},g,c_{i,j},h)$, and sets $\widetilde{c}_{j,i}=\widetilde{c}_{i,j}$.
3. For each $i\in\mathcal{I}$, $j\in\mathcal{J}_i$ s.t. $j<i$, this algorithm computes $D_{i,j}=\prod_{r=0}^{m-1}\widetilde{c}_{i+r,j+r}$, and sets $D_{j,i}=D_{i,j}$.
4. For each $i\in\mathcal{I}$, $j\in\mathcal{J}_i$, $u\in[\![\ell-1]\!]$ s.t. $j<i$, this algorithm picks $D_{i,j,u}\xleftarrow{\$}\mathbb{Z}_p^*$, it sets $D_{i,j,\ell}=\left(\frac{D_{i,j}}{\prod_{u=1}^{\ell-1}D_{i,j,u}^{2^{u-1}}}\right)^{\frac{1}{2^{\ell-1}}}$, and simulates the proof $\pi_{i,j,u}^{\mathsf{bin}}$ using $\mathsf{Simulator}_{\mathsf{OR}}$. For each $i\in\mathcal{I}$, $j\in\mathcal{J}_i$, $u\in[\![\ell]\!]$ s.t. $j<i$, it sets $D_{j,i,u}=D_{i,j,u}$.
5. This algorithm simulates the proof π^{cmp} using $\mathsf{Simulator}_{\mathsf{cmp}}$.

Finally, this algorithm simulates π_1^{com} using $\mathsf{Simulator}_{\mathsf{Open}}$ and returns:

$$\pi=((\widetilde{c}_{i,j},\pi_{i,j}^{\mathsf{sq}})_{i,j\in[\![n]\!];j<i},(D_{i,j,u},\pi_{i,j,u}^{\mathsf{bin}})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i},\pi^{\mathsf{cmp}},\pi_1^{\mathsf{com}}).$$

The simulator replaces each commitment $\widetilde{c}_{i,j}$ and $D_{i,j,u}$ with a random element in $\mathbb{G}$ while ensuring the relation 3. Furthermore, it replaces the proofs $\pi_{i,j}^{\mathsf{sq}}$, $\pi_{i,j,u}^{\mathsf{bin}}$, π^{cmp} and π_1^{com} with simulated proofs. Since the commitment scheme is perfectly hiding, the commitments generated by MP.NIP are indistinguishable from those returned by $\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}$, and since each proof is zero-knowledge, the proofs generated by MP.NIP are indistinguishable from those simulated in $\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}$. We denote by $\mathcal{A}$ the adversary, we have:

$$\begin{aligned}&\Pr\left[b\leftarrow\mathcal{A}(C,\pi):\begin{array}{l}C\leftarrow\mathsf{MP.Com}(x);\\ \pi\leftarrow\mathsf{MP.NIP}\{(K,x):\mathsf{MP.Ope}(K,C,x)\wedge\mathsf{Ano}(x)\}\end{array}\right]\\ &=\Pr\left[b\leftarrow\mathcal{A}(C,\pi):\begin{array}{l}C\leftarrow\mathsf{MP.Com}(x);\\ \pi\leftarrow\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}(C)\end{array}\right].\end{aligned}$$

The proofs for the cases of no anomaly, similarity and no similarity are similar, except that we consider $\mathsf{Simulator}_{\mathsf{cmp}}$ as the simulator for the relations:

- $\bigwedge_{i\in\mathcal{I}}\left(\bigvee_{j\in\mathcal{J}_i}\mathbf{d}_{i,j}<\epsilon^2+1\right)\wedge\bigwedge_{\substack{i\in\mathcal{I},\\ j\in\mathcal{J}_i,\\ u\in[\![\ell]\!]}}\mathsf{Ope}(w_{i,j,u},D_{i,j,u},\mathbf{d}_{i,j,u})$ for the no anomaly case,
- $\bigvee_{i\in\mathcal{I}}\left(\bigvee_{j\in\mathcal{J}_i}\mathbf{d}_{i,j}<\delta^2\right)\wedge\bigwedge_{\substack{i\in\mathcal{I},\\ j\in\mathcal{J}_i,\\ u\in[\![\ell]\!]}}\mathsf{Ope}(w_{i,j,u},D_{i,j,u},\mathbf{d}_{i,j,u})$ for the similarity case,
- and $\bigwedge_{i\in\mathcal{I}}\left(\bigwedge_{j\in\mathcal{J}_i}\mathbf{d}_{i,j}>\delta^2-1\right)\wedge\bigwedge_{\substack{i\in\mathcal{I},\\ j\in\mathcal{J}_i,\\ u\in[\![\ell]\!]}}\mathsf{Ope}(w_{i,j,u},D_{i,j,u},\mathbf{d}_{i,j,u})$ for the no similarity case.

Proof that (MP.Com, MP.Ope) is extractable. This proof show that the probability that the adversary, denoted $\mathcal{A}$, wins the extractability experiment is negligible. In the following, we consider a sequence of games [20] where the first game corresponds to the extractability experiment and the last game is the game where the adversary has no possibility of producing fresh and valid proof s.t. the extractor (fails or) extracts the witness (K,x) s.t. $\mathsf{MP.Ope}(K,C,x)\neq 1\vee\mathsf{Ano}(x)\neq 1$.

Game G_0: This game is the same as the extractability experiment, we have:

$$\begin{aligned}&\Pr\left[\mathcal{A}\text{ wins }G_0\right]\\&=\Pr\left[\begin{array}{c}\neg\left(\mathsf{MP.Ope}(K,C,x)\wedge\mathsf{Ano}(x)\right)\\ \wedge\mathsf{MP.Verify}(C,\pi)\end{array}:\begin{array}{l}(C,\pi)\leftarrow\mathcal{A}^{\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}(\cdot)}(\lambda);\\ (K,x)\leftarrow\mathsf{Ext}^{\mathcal{A}}(\lambda)\end{array}\right]\end{aligned}$$

Game G_1: This game is the same as G_0 except that G_1 uses the extractors on the proofs $(\pi^{\mathsf{sq}}_{i,j})_{i,j\in[\![n]\!];j<i}$, $(\pi^{\mathsf{bin}}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$, π^{cmp} and π^{com}_1, and aborts and returns 0 on the event $F_1=$ "An extractor fails on at least one proof".
Let $\epsilon_{\mathsf{ext}}(\lambda)$ be the maximum probability of failure of the extractors. Since all NIP are extractable, we have:

$$\left|\Pr[\mathcal{A}\text{ wins }G_1]-\Pr[\mathcal{A}\text{ wins }G_0]\right|\leq\left(n^2\ell+\frac{n(n-1)}{2}+2\right)\epsilon_{\mathsf{ext}}(\lambda).$$

We can note that if $\mathcal{A}$ wins G_1, then the extractors did not fail, and the witnesses are correctly extracted. G_1 extracts the witnesses:

- $(x_{i,j},k_{i,j},\widetilde{k}_{i,j})_{i,j\in[\![n]\!];j<i}$ from $(\pi^{\mathsf{sq}}_{i,j})_{i,j\in[\![n]\!];j<i}$,
- $(w_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$ from $(\pi^{\mathsf{bin}}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$,
- $(w'_{i,j,u},\mathbf{d}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!]}$ from π^{cmp} (where u_1 represents the index of the least significant bit equal to zero in ϵ^2+1),
- (x_1,k_1) from π^{com}_1,
- $(x_i)_{i\in[\![2;n]\!]}$ (resp. $(k_i)_{i\in[\![2;n]\!]}$) obtained by computing for each $i\in[\![2;n]\!]$, $x_{i,1}+x_1$ (resp. $k_{i,1}+k_1$).

– and $(\mathbf{d}_{i,j,u})_{u\in[\![1;u_1-1]\!]}$ by computing $\mathbf{d}_{i,j} = \sum_{r=0}^{m-1} x_{i+r,j+r}^2$, and finds the binary decomposition of $\mathbf{d}_{i,j}$.

The witnesses $(w'_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!];j<i}$ and $(w_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!];j<i}$ are the same because the proofs $(\pi^{\mathsf{bin}}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$ and π^{cmp} implicitly use the same $(D_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!];j<i}$.

Game G_2: This game is the same as G_1 except that G_2 aborts and returns 0 on the event F_2 ="The extractors extract $(x_i)_{i\in[\![n]\!]}$, $(k_i)_{i\in[\![n]\!]}$ and $(x_{i,j})_{i,j\in[\![n]\!];j<i}$ s.t. there exists $i,j\in[\![n]\!]$ with $j<i$ s.t. $x_i - x_j \neq x_{i,j}$ and $k_i - k_j \neq k_{i,j}$". We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_2] - \Pr[\mathcal{A} \text{ wins } G_1]| \leq \Pr[F_2].$$

We prove this claim by reduction, we suppose that the event F_2 occurs with a non negligible probability, we build a PPT algorithm $\mathcal{B}$ whose purpose is to break the binding property of Pedersen commitment.

$\mathcal{B}(\mathsf{set}_{\mathcal{B}})$: this algorithm parses $\mathsf{set}_{\mathcal{B}}$ as $(\widetilde{g},\widetilde{h})$, and simulates the game G_1 to $\mathcal{A}$ except that:

– this algorithm runs $\mathsf{set} \leftarrow \mathsf{MP.Setup}(\lambda,n,m,\ell)$, parses set as $(\mathbb{G},p,g,h,\lambda,n,m,\ell)$ and replaces g with $\widetilde{g}$ and h with $\widetilde{h}$.
– this algorithm runs $(C,\pi) \leftarrow \mathcal{A}^{\mathsf{Simulator}^{\mathsf{MP.NIP}}_{\mathsf{Ano}}(.)}(\lambda,\mathsf{set})$, and if $\mathsf{MP.Verify}(C,\pi)\neq 1$, aborts and returns $\perp$.
– this algorithm extracts the witnesses as in G_1, and we denote by :
  • $(x_{i,j},k_{i,j},\widetilde{k}_{i,j})_{i,j\in[\![n]\!];j<i}$, the witnesses extracted from $(\pi^{\mathsf{sq}}_{i,j})_{i,j\in[\![n]\!];j<i}$,
  • $(w_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$, the witnesses extracted from the proofs $(\pi^{\mathsf{bin}}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$,
  • $(w'_{i,j,u},\mathbf{d}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!]}$, the witnesses extracted from π^{cmp},
  • (x_1,k_1) the witnesses extracted from π_1^{com},
  • $(x_i)_{i\in[\![2;n]\!]}$ (resp. $(k_i)_{i\in[\![2;n]\!]}$) the witnesses obtained by computing for each $i\in[\![2;n]\!]$, $x_{i,1}+x_1$ (resp. $k_{i,1}+k_1$),
  • and $(\mathbf{d}_{i,j,u})_{u\in[\![1;u_1-1]\!]}$ by computing $\mathbf{d}_{i,j} = \sum_{r=0}^{m-1} x_{i+r,j+r}^2$, and finds the binary decomposition of $\mathbf{d}_{i,j}$.
– If the event F_2 occurs, $\mathcal{B}$ finds $i,j\in[\![n]\!]$ with $j<i$ s.t. $x_i-x_j\neq x_{i,j}$ and $k_i-k_j\neq k_{i,j}$, this algorithm sets $m_0=x_i-x_j$, $m_1=x_{i,j}$, $k'_0=k_i-k_j$ and $k'_1=k_{i,j}$, and returns (m_0,m_1,k'_0,k'_1,C), otherwise, that is, if F_2 does not occur, aborts and returns $\perp$.

If $\mathcal{A}$ wins the game G_1 and the event F_2 holds, therefore $\mathcal{B}$ returns two different ways to open the Pedersen commitment $c_{i,j}$, since the Pedersen commitment scheme is computationally binding, this occurs with a negligible probability denoted $\epsilon_{\mathsf{binding}}(\lambda)$. Hence, we have:

$$|\Pr[\mathcal{A} \text{ wins } G_2] - \Pr[\mathcal{A} \text{ wins } G_1]| \leq \Pr[F_2] \leq \epsilon_{\mathsf{binding}}(\lambda).$$

Game G_3: This game is the same as G_2 except that G_3 aborts and returns 0 on the event F_3 ="The extractors extract $(x_{i,j}, k_{i,j}, \widetilde{k}_{i,j})_{i,j\in[\![n]\!];j<i}$, $(w_{i,j,u}, w'_{i,j,u}, \mathbf{d}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$ s.t. there exists $i \in \mathcal{I}$, $j \in \mathcal{J}_i$ with $j < i$ s.t. $\sum_{u=1}^{\ell} \mathbf{d}_{i,j,u}\cdot 2^{u-1} \neq \sum_{r=0}^{m-1} x_{i+r,j+r}^2$ and $\sum_{u=1}^{\ell} w_{i,j,u}\cdot 2^{u-1} \neq \sum_{r=0}^{m-1} x_{i+r,j+r}\cdot k_{i+r,j+r} + \widetilde{k}_{i+r,j+r}$, and $\prod_{u=1}^{\ell} (D_{i,j,u})^{2^{u-1}} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$".
We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_3] - \Pr[\mathcal{A} \text{ wins } G_2]| \leq \Pr[F_3].$$

We prove this claim by reduction, we suppose that the event F_3 occurs with a non negligible probability, we build a PPT algorithm $\mathcal{B}$ whose purpose is to break the binding property of Pedersen commitment.

$\mathcal{B}(\mathsf{set}_{\mathcal{B}})$: this algorithm parses $\mathsf{set}_{\mathcal{B}}$ as $(\widetilde{g}, \widetilde{h})$, and simulates the game G_2 to $\mathcal{A}$ except that:

- this algorithm runs $\mathsf{set} \leftarrow \mathsf{MP.Setup}(\lambda, n, m, \ell)$, parses set as $(\mathbb{G}, p, g, h, \lambda, n, m, \ell)$ and replaces g with $\widetilde{g}$ and h with $\widetilde{h}$.
- this algorithm runs $(C, \pi) \leftarrow \mathcal{A}^{\mathsf{Simulator}_{\mathsf{Ano}}^{\mathsf{MP.NIP}}(.)}(\lambda, \mathsf{set})$, and if $\mathsf{MP.Verify}(C, \pi) \neq 1$, aborts and returns $\perp$.
- this algorithm extracts the witnesses as in G_1, and we denote by :
  - $(x_{i,j}, k_{i,j}, \widetilde{k}_{i,j})_{i,j\in[\![n]\!];j<i}$, the witnesses extracted from $(\pi_{i,j}^{\mathsf{sq}})_{i,j\in[\![n]\!];j<i}$,
  - $(w_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$, the witnesses extracted from the proofs $(\pi_{i,j,u}^{\mathsf{bin}})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![\ell]\!];j<i}$,
  - $(w'_{i,j,u}, \mathbf{d}_{i,j,u})_{i\in\mathcal{I},j\in\mathcal{J}_i,u\in[\![u_1;\ell]\!]}$, the witnesses extracted from π^{cmp},
  - (x_1, k_1) the witnesses extracted from π_1^{com},
  - $(x_i)_{i\in[\![2;n]\!]}$ (resp. $(k_i)_{i\in[\![2;n]\!]}$) the witnesses obtained by computing for each $i \in [\![2; n]\!]$, $x_{i,1} + x_1$ (resp. $k_{i,1} + k_1$),
  - and $(\mathbf{d}_{i,j,u})_{u\in[\![1;u_1-1]\!]}$ by computing $\mathbf{d}_{i,j} = \sum_{r=0}^{m-1} x_{i+r,j+r}^2$, and finds the binary decomposition of $\mathbf{d}_{i,j}$.
- If the event F_3 occurs, $\mathcal{B}$ finds $i \in \mathcal{I}$, $j \in \mathcal{J}_i$ with $j < i$ s.t. $\sum_{u=1}^{\ell} \mathbf{d}_{i,j,u} \cdot 2^{u-1} \neq \sum_{r=0}^{m-1} x_{i+r,j+r}^2$ and $\sum_{u=1}^{\ell} w_{i,j,u} \cdot 2^{u-1} \neq \sum_{r=0}^{m-1} x_{i+r,j+r} \cdot k_{i+r,j+r} + \widetilde{k}_{i+r,j+r}$, and $\prod_{u=1}^{\ell} (D_{i,j,u})^{2^{u-1}} = \prod_{r=0}^{m-1} \widetilde{c}_{i+r,j+r}$, this algorithm sets $m_0 = \sum_{u=1}^{\ell} \mathbf{d}_{i,j,u} \cdot 2^{u-1}$, $m_1 = \sum_{r=0}^{m-1} x_{i+r,j+r}^2$, $k_0 = \sum_{u=1}^{\ell} w_{i,j,u} \cdot 2^{u-1}$ and $k_1 = \sum_{r=0}^{m-1} x_{i+r,j+r} \cdot k_{i+r,j+r} + \widetilde{k}_{i+r,j+r}$, and returns (m_0, m_1, k_0, k_1, C), otherwise, that is if F_3 does not occur, aborts and returns $\perp$.

We can remark that if $\mathcal{A}$ wins the game G_2 and the event F_3 occurs, therefore $\mathcal{B}$ returns two different ways to open the Pedersen commitment $D_{i,j}$, since the Pedersen commitment scheme is computationally binding this happens with a negligible probability $\epsilon_{\mathsf{binding}}(\lambda)$, we have:

$$|\Pr[\mathcal{A} \text{ wins } G_3] - \Pr[\mathcal{A} \text{ wins } G_2]| \leq \Pr[F_3] \leq \epsilon_{\mathsf{binding}}(\lambda).$$

Finally, at this step, the adversary has no other possibility to win because if the events F_1, F_2 and F_3 do not occur, then $\mathsf{MP.Ope}(K, C, x) = 1$ and $\mathsf{Ano}(x) = 1$. Hence, we have:

$$\begin{aligned}
&\Pr\left[\begin{array}{c}\neg\left(\mathsf{MP.Ope}(K,C,x) \wedge \mathsf{Ano}(x)\right) \\ \wedge \mathsf{MP.Verify}(C,\pi)\end{array} : \begin{array}{l}(C,\pi) \leftarrow \mathcal{A}^{\mathsf{Simulator}^{\mathsf{MP.NIP}}_{\mathsf{Ano}}(.)}(\lambda); \\ (K,x) \leftarrow \mathsf{Ext}^{\mathcal{A}}(\lambda)\end{array}\right] \\
&= \left|\Pr\left[\mathcal{A} \text{ wins } G_3\right] - \Pr\left[\mathcal{A} \text{ wins } G_0\right]\right| \\
&\leq \left(\frac{n(n-1)}{2} + n^2\ell + 2\right)\epsilon_{\mathsf{ext}}(\lambda) + 2\epsilon_{\mathsf{binding}}(\lambda).
\end{aligned}$$

The proof for the no anomaly (resp. similarity and no similarity) case is similar, except that in the extractability experiment, we consider $\neg\mathsf{Ano}(x)$ (resp. $\mathsf{Sim}(x)$ and $\neg\mathsf{Sim}(x)$) instead of $\mathsf{Ano}(x)$.

D Experiment Details

We report below the threshold values used in the evaluation. In each case, the threshold corresponds to the right-hand side of the comparison operator. Precisely, it is set to ϵ^2, $\epsilon^2 + 1$, δ^2, and $\delta^2 - 1$ for $\mathsf{Ano}(x)$, $\neg\mathsf{Ano}(x)$, $\mathsf{Sim}(x)$, and $\neg\mathsf{Sim}(x)$, respectively. When a significance threshold must be defined (*i.e.,* for $\neg\mathsf{Ano}(x)$ and $\neg\mathsf{Sim}(x)$), it should be chosen strictly larger than any attainable value. This upper bound depends on the window length m. At the same time, the threshold must remain smaller than the maximum representable value determined by the bit-length ℓ.

Table 7. Threshold value used in evaluation

m	$\mathsf{Ano}(x)$	$\neg\mathsf{Ano}(x)$	$\mathsf{Sim}(x)$	$\neg\mathsf{Sim}(x)$
3	0	6077	1	6076
10	0	20251	1	20252
100	0	202501	1	202502

Below we report the communication cost for each scenario in Table 8.

Table 8. Total communication cost for the four scenarios

n	$\mathsf{Ano}(x)$	$\neg\mathsf{Ano}(x)$	$\mathsf{Sim}(x)$	$\neg\mathsf{Sim}(x)$
10	110.3 KB	94.5 KB	82.69 KB	84.66 KB
100	21.5 MB	18.8 MB	15.0 MB	16.4 MB
200	100.9 MB	88.0 MB	70.3 MB	76.7 MB
1000	2.8 GB	2.5 GB	2.0 GB	2.1 GB