TimeWak: Temporal Chained-Hashing Watermark for Time Series Data

Zhi Wen Soi[1,*] **Chaoyi Zhu**[2,*] **Fouad Abiad**[2] **Aditya Shankar**[2]
Jeroen M. Galjaard[2] **Huijuan Wang**[2] **Lydia Y. Chen**[1,2,†]

[1]University of Neuchâtel [2]Delft University of Technology

{zhi.soi, yiyu.chen}@unine.ch
{c.zhu-2, f.abiad, a.shankar, j.m.galjaard, h.wang}@tudelft.nl

Abstract

Synthetic time series generated by diffusion models enable sharing privacy-sensitive datasets, such as patients' functional MRI records. Key criteria for synthetic data include high data utility and traceability to verify the data source. Recent watermarking methods embed in homogeneous latent spaces, but state-of-the-art time series generators operate in data space, making latent-based watermarking incompatible. This creates the challenge of watermarking directly in data space while handling feature heterogeneity and temporal dependencies. We propose TimeWak, the first watermarking algorithm for multivariate time series diffusion models. To handle temporal dependence and spatial heterogeneity, TimeWak embeds a temporal chained-hashing watermark directly within the temporal-feature data space. The other unique feature is the ϵ-exact inversion, which addresses the non-uniform reconstruction error distribution across features from inverting the diffusion process to detect watermarks. We derive the error bound of inverting multivariate time series while preserving robust watermark detectability. We extensively evaluate TimeWak on its impact on synthetic data quality, watermark detectability, and robustness under various post-editing attacks, against five datasets and baselines of different temporal lengths. Our results show that TimeWak achieves improvements of 61.96% in context-FID score, and 8.44% in correlational scores against the strongest state-of-the-art baseline, while remaining consistently detectable. Our code is available at https://github.com/soizhiwen/TimeWak.

1 Introduction

Multivariate time series data drive key applications in healthcare [19], finance [13], and science [25]. However, access to real-world datasets is often restricted by privacy regulations, limited availability, and high acquisition costs. To address these issues, synthetic time series generated by models are increasingly adopted as practical alternatives [9, 25]. Among generative techniques, *diffusion* models [10] have gained prominence for producing high-quality samples, often outperforming the mainstream Generative Adversarial Networks and Variational Autoencoders [6, 12].

Beyond generation quality, *traceability* is equally critical, as it ensures verifiability and safeguards against misuse [18, 41]. In this context, *watermarking* has become the de-facto approach for tracking and auditing synthetic data [32, 18]. The challenge lies in striking a delicate balance: embedding imperceptible signals that preserve the quality of generated content while remaining detectable, even

[*]Equal contribution.
[†]Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

under post-processing [40]. Recent works embed watermarks *during* generation by adding them into the *latent* space, offering advantages such as generality and lightweight computation [32, 41]. However, latent-space watermarks are not always viable, especially since many state-of-the-art (SOTA) time series generators operate within the data space [26, 1, 34]. Additionally, latent generators introduce a trade-off in detectability, as diffusion inversion and the encode-decode cycle are inherently lossy and can degrade the watermarks [21].

While generation-time watermarks have proven effective for images [32] and tables [41], their applicability to time series data remains unexplored. Multivariate time series data possess temporal dependencies and heterogeneous features, like gender versus income. The ensuing challenges are twofold: (i) embedding watermarks *directly* in the data space while preserving inter- *and* intra-variate temporal dependencies of the generated time series, and (ii) ensuring accurate watermark detection despite the lossy nature of diffusion inversion, whilst handling mixed feature types.

We propose `TimeWak`, the first *generation-time* watermark for multivariate time series diffusion models, featuring **temporal chained-hashing** with ϵ**-exact inversion**. `TimeWak` first embeds cyclic watermark patterns, i.e., the positional seeds of Gaussian noises, along the temporal direction. First, we *chained-hash* the seeds along the temporal axis, then shuffle the seeds across features to maintain temporal correlations while preserving the unique characteristics of each feature. To ensure reliable watermark detection, we introduce an ϵ-exact inversion strategy that makes a practical concession in the otherwise exactly invertible diffusion process: the *Bi-Directional Integration Approximation* (BDIA) [36]. We further provide theoretical guarantees on the resulting inversion error in Appendix C.

We evaluate `TimeWak` against five SOTA watermarking methods on five datasets under varying temporal lengths. `TimeWak` achieves the best detectability under six post-editing attack configurations with the minimum data quality degradation. Additionally, results show that `TimeWak` preserves temporal and cross-variate dependencies with high quality with near-exact watermark bit reconstruction. To summarize, we list our contributions as follows:

- We propose `TimeWak`, the first generation-time watermarking scheme for multivariate time series diffusion models, preserving realistic spatio-temporal dependencies while remaining detectable.
- To preserve temporal characteristics and boost robustness against post-processing operations, we design a *temporal chained-hashing* scheme that embeds watermark seeds along the temporal direction, followed by a shuffle across features.
- For robust detectability, we propose ϵ*-exact* inversion by extending BDIA sampling into a data space diffusion generator, and provide a theoretical error bound analysis.
- Our extensive evaluation shows that `TimeWak` achieves up to **61.96%** better context-FID scores and **8.44%** better correlation scores compared to the strongest SOTA watermarking method.

2 Related work

We summarize related works on watermarking diffusion models according to their generating method (post-processing or generation-time generation), and data modality. To the best of our knowledge, our work is the first generation-time watermarking scheme for time series diffusion models.

Watermarking diffusion models. Watermarking has become a critical solution for tracing and authenticating machine-generated content. *Post-generation* techniques embed watermarks after synthesis, often degrading the generated data's quality due to direct modifications [5, 38]. Alternatively, recent advancements embed watermarks within the training process. Studies such as *Stable Signature* [7] and *FixedWM* [18] *fine-tune* diffusion models to embed and extract watermarks. However, these methods modify model parameters, risking overfitting which hurts generalizability.

Watermarking images. *Tree-Ring* (TR) [29] embeds the watermark during sampling by modifying the initial noise latent vector in the Fourier space. However, it disrupts the Gaussian noise distribution, reducing the diversity and quality of generated samples. *Gaussian Shading* (GS) [32] improves robustness by embedding watermarks directly in the latent space using invertible transformations. However, GS is tailored towards images and requires reversing synthetic samples into the latent space for detection, which is a noisy and error-prone process that limits the detection accuracy.

Watermarking tables. *TabWak* [41] watermarks tabular data in latent space using seeded *self-cloning*, *shuffling* with a secret key, and a *valid-bit* mechanism. However, it relies on latent models and does

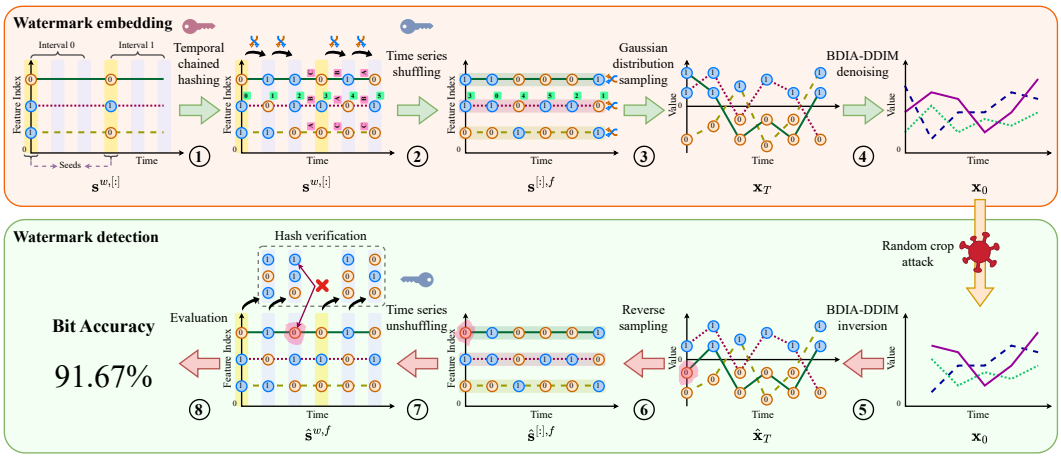


Figure 1: Overview of TimeWak. First, we assign random seeds at the beginning of each interval. ① Temporally chained-hashing. A, B, and C (pink) show seeds being copied from the previous step and the feature order shuffled. ② Shuffling the seeds for each series. Positional indices are highlighted in green. ③ Constructing an initial Gaussian noise. ④ Generating multivariate time series. ⑤ Reversing the diffusion process. ⑥ Recovering the watermark seed. ⑦ Unshuffling the seeds in the opposite way they were shuffled. ⑧ Bit accuracy between the hash and recovered seed.

not account for temporal dependencies in a time series. Furthermore, its detectability is limited by the invertibility of the diffusion process and the lossy conversion to and from latent representations.

3 TimeWak

We first highlight the unique challenges of watermarking multivariate time series, motivating the design of TimeWak, shown in Figure 1. Then we introduce TimeWak's key novelties: (i) temporal chained-hashing the watermark seeds cyclically along the temporal axis; (ii) shuffling the seeds across features, accounting for feature heterogeneity in the time series; and (iii) an adapted BDIA-DDIM sampling method with a theoretically bounded ϵ-exact inversion, enhancing robust detectability. Key notations are summarized in Appendix A.

3.1 Time series diffusion and observations

Time series diffusion. We define a time series sample of F features (variates) and W timesteps as $\mathbf{x}_0 \in \mathbb{R}^{W \times F}$, with $\mathbf{x}_0^{w,f}$ denoting the value of feature f at timestep w. Unlike image and tabular diffusion, SOTA time series diffusion models operate on the data space, which have heterogeneous features, e.g., income vs. gender, and temporal dependence [26, 1, 34]. These time series generators use *Denoising Diffusion Implicit Models* (DDIM) [23] to synthesize time series starting from Gaussian noise, $\mathbf{x}_T$, by iteratively denoising over T steps, i.e., $\mathbf{x}_T, \mathbf{x}_{T-1}, \ldots, \mathbf{x}_0$. Specifically, DDIM sets the state $\mathbf{x}_{t-1}$ at diffusion step $t-1$ as follows:

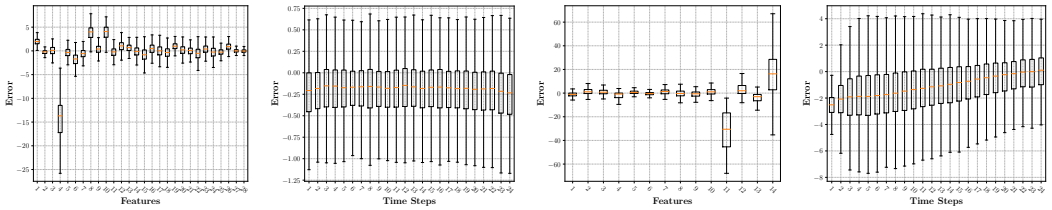
$$\mathbf{x}_{t-1} = \alpha_{t-1}\left(\frac{\mathbf{x}_t - \sigma_t \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\alpha_t}\right) + \sigma_{t-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \tag{1}$$

where α_t and σ_t are time-dependent diffusion coefficients, and $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ represents the model's noise estimate. DDIM approximates $\mathbf{x}_t$ as follows:

$$\mathbf{x}_t = \alpha_t\left(\frac{\mathbf{x}_{t-1} - \sigma_{t-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\alpha_{t-1}}\right) + \sigma_t\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx \alpha_t\left(\frac{\mathbf{x}_{t-1} - \sigma_{t-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}, t)}{\alpha_{t-1}}\right) + \sigma_t\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}, t). \tag{2}$$

However, this approximation introduces errors, producing inconsistencies between the forward and backward processes. It also introduces the following time series-specific watermarking challenges:

Spatial heterogeneity. Watermarks must be embedded directly within the temporal and feature spaces of the data. Features can be very diverse, e.g., gender vs. income distribution, which increases



(a) Feature (Energy) (b) Temporal (Energy) (c) Feature (MuJoCo) (d) Temporal (MuJoCo)

Figure 2: Average reconstruction error distribution across feature indices and timesteps on Diffusion-TS with DDIM and DDIM inversion. Reconstruction error is the signed absolute difference between reconstructed and original values.

the difficulty of detecting watermarks. Specifically, the key detection step inverts the time series back to Gaussian noise. Unfortunately, this inversion process is inexact, yielding reconstruction errors during noise-estimation. Figure 2 shows the impact of heterogeneity on the reconstruction errors for the Energy and MuJoCo datasets. Due to spatial heterogeneity, the reconstruction errors across the features vary significantly more than they do along the temporal axis. Existing tabular watermarks [41] implicitly assume a uniform distribution across features and compare watermark seeds across features, which prevents reliable watermark verification in multivariate time series.

Temporal dependence. Time series consist of values that are inherently correlated across timesteps. It is critical to preserve such temporal consistencies when generating time series. Consequently, reconstruction errors are not fully independent across timesteps within each sample, as errors at neighbouring timesteps often exhibit stronger correlations than more distant ones. To ensure robustness, solutions must embed the watermark in a way that respects these temporal dependencies while remaining detectable. This requires designing watermarking strategies that align with the sequential nature of time series diffusion models, invalidating the applicability of existing watermark approaches that neglect the temporal dependence of time series data.

3.2 `TimeWak` algorithm

To address the challenges of spatial heterogeneity and temporal dependence, we propose `TimeWak`, a method that enables per-sample watermark detection while mitigating non-uniform reconstruction errors across features and preserving temporal structure. Through a structured propagation mechanism, `TimeWak` enhances the watermark's robustness, even in the presence of inversion errors.

Overview. We begin with a high-level overview of `TimeWak`'s watermarking pipeline, which consists of four main stages: watermark embedding, time series generation, inversion, and detection. Each stage involves multiple steps that we describe briefly here and explain in detail in the subsequent sections. Following Figure 1, the complete process works as follows:

1. **Embedding I:** generating watermark seeds ($\mathbf{s}$). We first split a multivariate time series into intervals along the time axis, then we randomly sample seeds $\mathbf{s}^{w,[:]}$ (with values in $\{0, 1\}$) at the start of each interval. ① We temporally chain-hash the seeds in a cyclic manner across timesteps until the end of the current interval, by applying a unique permutation key at each timestep. Then, ② we independently shuffle the seeds for each feature using distinct permutation keys.
2. **Embedding II:** generating time series from the watermarked seeds ($\mathbf{s} + \mathbf{x}_T \rightarrow \mathbf{x}_0$). ③ Sampling from a feature-wise pseudo-random Gaussian distribution based on all the aforementioned seeds, where the seeds determine the sign of the sampled values ($\mathbf{s}^{w,f} = 1$ becomes positive, $\mathbf{s}^{w,f} = 0$ becomes negative). These noise signals are used as input to a BDIA variant of a DDIM diffusion model, which is then ④ used to generate a multivariate time series. An attack, such as a random crop attack, may occur at this stage.
3. **Detection I**: inversion of the time series ($\mathbf{x}_0 \rightarrow \hat{\mathbf{x}}_T$). ⑤ The inverse BDIA-DDIM process is applied to the time series, inverting it to Gaussian noise $\hat{\mathbf{x}}_T$. Then ⑥ we reverse-sample each time series to get the seeds (positive values become 1, negative values become 0); we now have the shuffled seed features.
4. **Detection II:** watermark detection ($\hat{\mathbf{x}}_T \rightarrow \hat{\mathbf{s}}$). Given shuffled seeded features, we ⑦ unshuffle the seeds in the opposite way they were shuffled (using the inverse of the permutation keys of

step ②), to obtain the retrieved seed features $\hat{\mathbf{s}}$. We then ⑧ verify the hash of these retrieved seed features by comparing them with the original temporal chained-hash seeds $\mathbf{s}$, for which we compute the bit accuracy between the hashed and recovered versions of each seed.

3.2.1 Chained-hashing watermark seeds ($\mathbf{s} + \mathbf{x}_T \rightarrow \mathbf{x}_0$)

Existing watermarking methods assign a watermark seed to each feature dimension with L bits, forming a seed matrix of dimensions $W \times F$, denoted as $\mathbf{s} \in \mathbb{R}^{W \times F}$, where W and F are the respective total timesteps and total features of the time series [32, 41]. However, such approaches lack the ability to leverage temporal dependencies and may introduce inconsistencies across timesteps. To improve the watermark's temporal coherence, we partition the time series data into $n = \lfloor W/H \rfloor$ non-overlapping intervals, each of length H. At the start of each interval, the watermark seed across all features, $\mathbf{s}^{kH+1,[:]} \in \mathbb{R}^F$, is sampled from a discrete uniform distribution $\mathcal{U}\left(\{0, L-1\}\right)^F$, with $[:]$ denoting all indices along a dimension.

Within each interval, the watermark seed evolves over timesteps using a **temporal chained-hashing** mechanism. Specifically, for all features, the seed at timestep w is recursively derived from the seed at the previous timestep $w-1$, ensuring temporal consistency. Formally, for $k = 0, \ldots, n-1$, we initialize the watermark seed as:

$$\mathbf{s}^{w,[:]} = \begin{cases} \mathcal{U}\left(\{0, L-1\}\right)^F & \textbf{if } w = kH + 1, \\ \mathcal{H}\left(\kappa, w, \mathbf{s}^{w-1,[:]}\right) & \textbf{otherwise}, \end{cases} \tag{3}$$

where κ is a cryptographic key controlling the hashing process, $\mathcal{H}$ is a deterministic permutation hash function ensuring temporal consistency, and $\mathcal{U}\left(\{0, L-1\}\right)^F$ is a vector of F i.i.d discrete uniform samples over $\{0, 1, \ldots, L-1\}$. The parameter $n = \lfloor W/H \rfloor$ denotes the total integer count of intervals, while k indexes the intervals, ranging from 0 to $n-1$. While temporal chaining preserves coherence across timesteps by linking each seed to its past, it may lead to repetitive patterns across intervals. To increase diversity, we further permute the seeds along the temporal axis for each feature:

$$\mathbf{s}^{[:],f} \leftarrow \pi_\kappa(\mathbf{s}^{[:],f}), \tag{4}$$

where π_κ is a permutation function parameterized by the cryptographic key κ. This step preserves inter-feature seed correlations while adding generation diversity.

After obtaining the watermark seed, we construct an initial Gaussian noise sample as follows. First, we draw a variable from the continuous uniform distribution $u \sim \mathcal{U}(0, 1)$ and use it to generate the noise variable $\mathbf{x}_T^{w,f}$ at diffusion step T as:

$$\mathbf{x}_T^{w,f} = \Phi^{-1}\left(\frac{u + \mathbf{s}^{w,f}}{L}\right), \tag{5}$$

where $\Phi^{-1}(\cdot)$ is the percent point function (PPF) of the standard Gaussian distribution $\Phi(\cdot)$, and $\mathbf{s}^{w,f}$ is the watermark seed for feature f at timestep w. Finally, the final time series sample $\mathbf{x}_0$ is obtained by denoising the initial noise $\mathbf{x}_T$ with the learned diffusion model.

3.2.2 ϵ-Exact inversion ($\mathbf{x}_0 \rightarrow \hat{\mathbf{x}}_T$)

Here, we propose a near-lossless inversion procedure by adopting the Bi-directional Integration Approximation (BDIA) technique [36], a novel approach to address inconsistencies in DDIM inversion [24]. We introduce a practical approximation in BDIA by removing the assumption of known $\mathbf{x_1}$, and derive the bound of the inversion error. BDIA improves upon DDIM by jointly leveraging the forward and backward diffusion updates. Specifically, obtaining each $\mathbf{x}_{t-1}$ as a linear combination of $(\mathbf{x}_{t+1}, \mathbf{x}_t, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t))$, where $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ represents the noise estimator of the diffusion model:

$$\mathbf{x}_{t-1} = \gamma\left(\mathbf{x}_{t+1} - \mathbf{x}_t\right) - \gamma\left(\frac{\mathbf{x}_t}{a_{t+1}} - \frac{b_{t+1}}{a_{t+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{x}_t\right) + \left(a_t\mathbf{x}_t + b_t\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right), \tag{6}$$

where $\gamma \in [0, 1]$, a_t and b_t are differentiable functions of t with bounded derivatives. Consequently, the inversion process can be directly calculated without approximation as follows:

$$\mathbf{x}_{t+1} = \frac{\mathbf{x}_{t-1}}{\gamma} - \frac{1}{\gamma}\left(a_t\mathbf{x}_t + b_t\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right) + \left(\frac{\mathbf{x}_t}{a_{t+1}} - \frac{b_{t+1}}{a_{t+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right). \tag{7}$$

By design, the introduced symmetry ensures time-reversible updates, meaning that if $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ are known, $\mathbf{x}_{t+1}$ can be computed without error. However, obtaining an exact inversion requires knowing both $\mathbf{x}_1$ and $\mathbf{x}_0$, the latter of which is available only as the model's denoised sample in practice.

To address this limitation, we introduce an adaptation to BDIA: we directly approximate $\mathbf{x}_1$ by equating it to $\mathbf{x}_0$. This seemingly simple estimation effectively enables practical application of BDIA while maintaining reasonable accuracy. To quantify the estimation error, we establish Theorem 3.1, which demonstrates that the final error remains bounded in terms of the initial estimation $\epsilon = \mathbf{x}_1 - \mathbf{x}_2$. We refer to this property as 'ϵ-exact' inversion. We defer our proof of Theorem 3.1 to Appendix C.

Theorem 3.1. *Let $\{\mathbf{x}_t\}_{t=0}^T$ be the sequence of diffusion states governed by the BDIA-DDIM recurrence for a given dataset, following Equation* (7)*. Given the noise estimator $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ follows Assumption 1. Suppose that instead of the exact terminal state $\mathbf{x}_1$, an approximation of $\mathbf{x}_1$, termed as $\mathbf{x}_1^{approx}$, is used with a small perturbation ϵ, given by:*

$$\mathbf{x}_1^{approx} = \mathbf{x}_2 = \mathbf{x}_1^{orig} + \epsilon. \tag{8}$$

Let the propagated error at time t be defined as,

$$\delta_t = \|\mathbf{x}_t^{approx} - \mathbf{x}_t^{orig}\|. \tag{9}$$

Then, for $t \geq 1$, the error is bounded by,

$$\|\delta_T\| \leq |\epsilon| \prod_{t=1}^{T-1} \left(\left| \frac{1}{\gamma} - \frac{a_t}{\gamma} + \frac{1}{a_{t+1}} \right| + \frac{b_t}{\gamma}\Delta_t + \frac{b_{t+1}}{a_{t+1}}\Delta_t \right), \tag{10}$$

where Δ_t quantifies the sensitivity of the noise estimator at timestep t.

Assumption 1 (Lipschitz continuity of the noise estimator)**.** *There exists a time-dependent constant $\Delta_t > 0$ and $\delta_t > 0$ such that, for any diffusion state $\mathbf{x}_t^{orig}$ encountered during sampling from a given dataset and any $\mathbf{x}_t^{approx}$ satisfying $\|\mathbf{x}_t^{approx} - \mathbf{x}_t^{orig}\| \leq \delta_t$, the noise estimator $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ satisfies the Lipschitz condition:*

$$\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t^{approx}, t) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t^{orig}, t)\| \leq \Delta_t \left\| \mathbf{x}_t^{approx} - \mathbf{x}_t^{orig} \right\|. \tag{11}$$

3.2.3 Watermark detection ($\mathbf{x}_0 \to \hat{\mathbf{x}}_T \to \hat{\mathbf{s}}$)

To verify the presence of `TimeWak`'s watermark in a generated time series, we first recover the initial noise used in the generative process via **diffusion inversion**. Given a time series instance $\mathbf{x}_0$, we estimate the initial noise $\hat{\mathbf{x}}_T$ through inversion. The watermark seed at each timestep and feature is then recovered by inverting the Gaussian mapping as follows:

$$\hat{\mathbf{s}}^{w,f} = \lfloor L \cdot \Phi(\hat{\mathbf{x}}_T^{w,f}) \rfloor, \tag{12}$$

where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard Gaussian distribution. This operation reconstructs the discrete watermark values embedded during sample generation.

Since the watermarking mechanism applies a feature-wise permutation controlled by a cryptographic key, the extracted watermark values are initially shuffled across feature dimensions. We restore the original seed assignments along the timesteps using the inverse permutation:

$$\hat{\mathbf{s}}^{[:],f} \leftarrow \pi_\kappa^{-1}(\hat{\mathbf{s}}^{[:],f}). \tag{13}$$

Beyond feature-space consistency, the extracted watermark sequence should exhibit structured temporal dependencies. Specifically, the watermark seed at each step must follow a predefined hash function, given by:

$$\forall w \neq kH + 1, \quad \hat{\mathbf{s}}^{w,[:]} = \mathcal{H}(\kappa, w, \hat{\mathbf{s}}^{w-1,[:]}). \tag{14}$$

To quantify detection confidence, we compute the **bit accuracy** of the extracted watermark sequence, measuring the proportion of correctly recovered bits:

$$\text{Acc} = \frac{1}{|\mathcal{W}^*|F} \sum_{w \in \mathcal{W}^*} \sum_{f=1}^{F} \mathbb{I}\left[\hat{\mathbf{s}}^{w,f} = \mathbf{s}^{w,f}\right], \tag{15}$$

where $\mathbb{I}[\cdot]$ is an indicator function that evaluates to 1 if the extracted bit matches the ground-truth watermark, and $\mathcal{W}^* = \{w \mid w \neq kH + 1\}$ represents the valid timesteps for comparison. By combining diffusion inversion, feature unshuffling, and temporal consistency verification, this detection framework ensures robust identification of watermarked time series samples.

To assess the statistical significance, we compute the Z-score to measure how strongly the observed bit accuracy deviates from the expected accuracy under a null hypothesis, detailed in Appendix D.2.

4 Evaluation

4.1 Experiments setup

Datasets. We use five time series datasets to evaluate `TimeWak`'s impact on generation quality, watermark detection accuracy, and robustness towards post-editing operations. These are: *Stocks* [33], *ETTh* [39], *MuJoCo* [27], *Energy* [3], and *fMRI* [22]. Additional dataset details are in Appendix D.1.

Metrics. *Synthetic data quality:* Context-FID score [11] measures the closeness between the real and synthetic time series distributions using the Fréchet distance [8]. Correlational score [15] measures the cross-correlation error between the real and synthetic multivariates. Discriminative score [33] trains a classifier to distinguish between synthetic and real data, with low scores implying they are indistinguishable. Predictive score [33] measures the downstream task performance by training a sequence model on the synthetic data and evaluating on real data. *Watermark detectability:* Z-score quantifies the difference in mean values between synthetic data with and without the watermark, with larger positive values indicating better detectability. TPR@X%FPR measures the True Positive Rate (TPR) at a fixed False Positive Rate (FPR) of X% in detecting watermarked time series. We provide additional details on these metrics in Appendix D.2.

Baselines. We compare against three sampling-based diffusion watermarks: TR [29], GS [32], and TabWak [41]. We also compare with TabWak$^\top$, an adaptation of TabWak that transposes the representation and watermarks along the temporal axis instead of feature-wise. We also compare against a post-generation watermarking method for time series, *Heads Tails Watermark* (HTW), which embeds a watermark by slightly adjusting the time series values by assigning a 'heads' or 'tails' based on a predefined ratio on the proportion of 'heads' values [28]. Detailed implementations of these methods are provided in Appendix D.3. Hardware specifications are detailed in Appendix D.4.

4.2 Synthetic data quality and watermark detectability

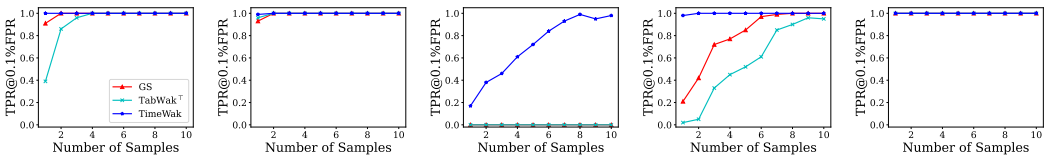
When evaluating synthetic time series quality, Table 1 shows that `TimeWak` consistently delivers top-tier performance across all metrics, outperforming or comparable to other baselines like HTW and TabWak$^\top$. While HTW sometimes surpasses un-watermarked data, possibly due to subtle perturbations introduced during watermarking that unintentionally bring synthetic samples closer to the ground truth, it fails to offer strong detectability, as reflected in its low Z-scores. In contrast, `TimeWak` and TabWak$^\top$ offer a far more favorable trade-off between quality and detectability. When benchmarked against TabWak$^\top$, `TimeWak` shows substantial gains, achieving up to 61.96% better Context-FID score on MuJoCo and 8.44% better correlation score on fMRI. Moreover, low discriminative and predictive scores further emphasize that `TimeWak`'s watermarking remains imperceptible and does not degrade downstream utility. This is made possible by its temporal chained-hashing mechanism, which precisely embeds the watermark while preserving both temporal structure and inter-variate relationships. Meanwhile, traditional image-based watermarking methods such as TR and GS perform poorly across all quality metrics. These methods struggle with time series data because they are optimized for spatial domains. Time series, however, are governed by temporal continuity and feature heterogeneity, thus requiring fundamentally different treatment.

`TimeWak` achieves significant improvements in detection performance. Using ϵ-exact inversion via BDIA-DDIM, it reconstructs high-fidelity noise estimates $\hat{\mathbf{x}}_T$ that closely resemble the ground truth $\mathbf{x}_T$. This results in consistently higher Z-scores across all datasets, outperforming all baselines, except on the fMRI dataset, where TabWak$^\top$ slightly edges out. Unlike GS, which maintains moderate detection at the cost of quality, or TR, which fails on both fronts, `TimeWak` delivers strong detectability without compromising fidelity.

Table 1: Results of synthetic time series quality and watermark detectability. No watermarking ('W/O') is included. Quality metrics are for 24-length sequences. Best results are in **bold**, and second-best are underlined.

Dataset	Method	Quality Metric ↓				Z-score ↑		
		Context-FID	Correlational	Discriminative	Predictive	24-length	64-length	128-length
Stocks	W/O	0.258±0.047	0.027±0.015	0.120±0.049	0.038±0.000	-	-	-
	TR	1.069±0.231	0.091±0.007	0.209±0.056	0.039±0.000	0.43±0.04	0.40±0.08	0.08±0.10
	GS	8.802±2.415	0.052±0.026	0.403±0.031	0.041±0.003	86.07±0.74	148.92±1.08	172.23±1.08
	HTW	0.279±0.052	0.017±0.003	0.122±0.034	**0.037±0.000**	4.45±0.62	7.34±0.94	10.39±1.33
	TabWak	0.292±0.064	0.017±0.012	0.124±0.024	0.038±0.000	-67.22±1.17	16.14±0.89	-10.49±1.28
	TabWak$^\top$	0.314±0.071	**0.016±0.017**	0.132±0.022	**0.037±0.000**	55.39±0.86	88.81±0.82	129.39±0.90
	`TimeWak`	**0.277±0.019**	0.020±0.018	**0.120±0.039**	0.038±0.000	**182.10±0.73**	**395.34±1.24**	**550.05±1.18**
ETTh	W/O	0.232±0.018	0.086±0.023	0.093±0.016	0.120±0.009	-	-	-
	TR	1.570±0.102	0.187±0.017	0.283±0.020	0.134±0.004	7.84±0.12	7.73±0.13	6.18±0.16
	GS	4.530±0.393	0.433±0.017	0.390±0.015	0.169±0.007	101.07±1.17	197.41±2.10	327.47±4.44
	HTW	0.243±0.024	**0.077±0.025**	0.103±0.003	0.123±0.002	3.43±0.83	5.08±1.48	6.84±2.22
	TabWak	0.251±0.027	0.335±0.029	**0.085±0.013**	0.125±0.002	-14.95±1.08	-6.16±1.18	-20.57±0.96
	TabWak$^\top$	0.450±0.057	0.116±0.020	0.096±0.014	**0.120±0.005**	109.35±0.91	162.44±1.11	235.03±1.48
	`TimeWak`	**0.237±0.017**	0.212±0.043	0.102±0.014	0.122±0.002	**134.83±0.95**	**236.08±1.63**	**340.36±2.06**
MuJoCo	W/O	0.065±0.011	0.419±0.084	0.032±0.026	0.008±0.001	-	-	-
	TR	1.512±0.179	1.153±0.065	0.261±0.069	0.015±0.003	1.38±0.03	1.49±0.04	1.31±0.04
	GS	6.548±1.267	1.327±0.061	0.474±0.006	0.014±0.002	21.13±0.77	10.13±0.62	39.63±0.71
	HTW	0.261±0.067	0.493±0.056	0.413±0.024	0.010±0.002	2.89±0.54	3.41±0.96	4.20±1.37
	TabWak	0.545±0.122	0.975±0.061	0.207±0.046	0.009±0.001	31.30±1.07	1.26±0.96	-3.00±1.04
	TabWak$^\top$	0.234±0.032	**0.463±0.059**	0.123±0.011	**0.007±0.001**	-4.85±0.87	-4.51±0.85	3.91±0.88
	`TimeWak`	**0.089±0.017**	0.532±0.137	**0.044±0.021**	0.008±0.001	**85.69±1.08**	**56.45±1.26**	**123.36±1.43**
Energy	W/O	0.118±0.021	1.245±0.236	0.137±0.014	0.253±0.000	-	-	-
	TR	0.649±0.128	3.870±0.537	0.455±0.017	0.337±0.007	9.51±0.09	17.29±0.11	22.58±0.19
	GS	1.480±0.273	3.831±0.272	0.494±0.004	0.330±0.004	51.22±0.88	68.67±1.20	45.42±1.05
	HTW	**0.099±0.009**	**1.312±0.280**	0.138±0.019	**0.253±0.000**	3.06±0.36	4.30±0.68	5.42±1.00
	TabWak	0.179±0.027	2.724±0.203	0.162±0.011	0.255±0.001	3.26±0.89	3.86±1.02	0.57±0.87
	TabWak$^\top$	0.213±0.024	1.740±0.290	**0.129±0.013**	0.265±0.004	40.82±0.81	46.68±0.86	26.00±1.12
	`TimeWak`	0.121±0.016	1.977±0.750	0.142±0.008	0.254±0.000	**231.28±1.45**	**267.53±2.60**	**245.37±2.88**
fMRI	W/O	0.190±0.006	1.952±0.087	0.132±0.027	0.100±0.000	-	-	-
	TR	2.474±0.341	13.312±0.254	0.496±0.003	0.146±0.004	6.49±0.05	8.25±0.05	9.94±0.04
	GS	0.714±0.051	14.628±0.052	0.499±0.001	0.108±0.001	420.02±1.44	321.52±0.59	701.90±0.72
	HTW	**0.180±0.011**	**1.900±0.047**	0.140±0.019	**0.100±0.000**	4.32±0.22	6.80±0.41	9.43±0.61
	TabWak	0.326±0.042	6.825±0.395	0.452±0.092	0.112±0.000	84.02±1.04	204.16±0.82	47.29±0.83
	TabWak$^\top$	0.350±0.014	2.191±0.095	0.208±0.049	0.101±0.000	**464.67±0.50**	**743.33±0.55**	**1031.96±0.79**
	`TimeWak`	0.199±0.010	2.006±0.053	**0.122±0.033**	**0.100±0.000**	379.51±0.82	595.68±1.03	526.81±13.12

This strength is further shown in Figure 3, which plots TPR@0.1%FPR on 64-length sequences as a function of the number of samples. Across all settings, `TimeWak` consistently outperforms GS and TabWak$^\top$, achieving significantly higher TPR values. Notably, `TimeWak` reaches a perfect TPR of 1.0 in all cases, requiring only one sample in three settings and two in the remaining one. This high sensitivity makes `TimeWak` well-suited for real-world use, where only limited samples may be available. Additional results are provided in Appendix E.8.



(a) Stocks (b) ETTh (c) MuJoCo (d) Energy (e) fMRI

Figure 3: TPR@0.1%FPR against number of samples across five datasets under 64-length sequences.

4.3 Robustness against post-editing attacks

To evaluate the robustness, we first design a set of post-editing attacks. *Offsetting* perturbs the time series by adding a constant offset to each feature based on 5% or 30% of its magnitude, and applied uniformly across all timesteps. *Random cropping* masks out a subregion of the time series by a fixed proportion (5% or 30%), along the rows and columns, similar to the image domain [32]. The *min-max*

Table 2: Results of robustness against post-editing attacks. Average Z-score on 64-length sequences, including un-attacked scores from Table 1. Best results are in **bold**, and second-best are underlined.

Dataset	Method	Without Attack	Offset ↑		Random Crop ↑		Min-Max Insertion ↑	
			5%	30%	5%	30%	5%	30%
Stocks	TR	0.40±0.08	0.35±0.07	0.17±0.09	57.09±1.09	**76.87±1.29**	0.99±0.10	5.43±0.18
	GS	148.92±1.08	152.81±1.33	164.23±1.54	42.53±1.11	32.80±0.78	136.73±0.90	77.20±0.86
	HTW	7.34±0.94	7.34±0.94	7.34±0.94	-0.81±0.42	-0.92±0.32	6.41±1.16	2.72±1.15
	TabWak	16.14±0.89	25.74±1.04	45.02±1.40	40.59±1.16	19.71±1.06	19.11±0.75	30.87±0.45
	TabWak$^\top$	88.81±0.82	88.84±0.85	85.87±0.86	30.55±0.92	1.71±0.87	65.25±0.79	15.10±0.56
	TimeWak	**395.34±1.24**	**375.96±0.96**	**371.04±1.02**	**78.20±2.23**	10.40±1.05	**296.60±1.62**	**83.74±1.55**
ETTh	TR	7.73±0.13	7.96±0.11	8.83±0.12	27.22±0.39	38.53±0.32	21.86±0.26	49.85±0.64
	GS	197.41±2.10	186.62±2.21	159.04±2.20	**105.11±2.09**	-39.87±1.24	**182.03±2.09**	**105.23±1.21**
	HTW	5.08±1.48	5.08±1.48	5.08±1.48	0.54±1.33	-1.99±0.80	4.47±1.45	2.03±1.08
	TabWak	-6.16±1.18	-2.75±1.29	4.92±1.52	84.37±2.67	**88.07±2.59**	4.29±0.94	36.83±0.71
	TabWak$^\top$	162.44±1.11	157.75±1.09	135.66±1.23	70.38±1.16	10.79±1.14	99.23±0.96	9.01±0.80
	TimeWak	**236.08±1.63**	**243.86±1.61**	**207.90±1.70**	75.35±1.18	2.47±1.08	171.51±1.46	27.78±1.22
MuJoCo	TR	1.49±0.04	1.46±0.03	1.54±0.03	7.14±0.08	7.17±0.07	6.09±0.11	**14.83±0.22**
	GS	10.13±0.62	10.35±0.69	8.76±0.75	28.99±0.80	**10.09±0.91**	9.35±0.76	10.40±1.07
	HTW	3.41±0.96	3.41±0.96	3.41±0.96	1.13±0.78	-1.32±0.67	3.09±0.91	1.67±0.65
	TabWak	1.26±0.96	0.48±0.89	5.48±1.22	-1.75±1.21	-6.65±0.71	-0.57±0.91	-0.31±0.59
	TabWak$^\top$	-4.51±0.85	-3.46±0.99	-0.24±0.94	-32.20±1.04	-42.74±1.25	-34.16±0.90	-80.63±1.05
	TimeWak	**56.45±1.26**	**58.90±1.36**	**51.53±1.31**	**49.30±1.20**	1.85±1.12	**36.59±1.19**	10.35±1.18
Energy	TR	17.29±0.11	16.74±0.11	15.09±0.09	**128.26±0.39**	**148.31±0.29**	37.72±0.20	**75.09±0.29**
	GS	68.67±1.20	63.93±1.09	54.84±1.09	66.30±0.99	96.29±0.80	66.40±1.27	53.06±1.65
	HTW	4.30±0.68	4.30±0.68	4.30±0.68	0.40±0.48	-0.60±0.34	3.87±0.66	2.03±0.48
	TabWak	3.86±1.02	-5.32±1.01	-8.99±1.03	-8.31±0.65	9.94±0.60	4.21±0.78	2.67±0.36
	TabWak$^\top$	46.68±0.86	43.10±0.84	42.38±0.89	12.26±1.01	-49.34±1.74	11.87±0.85	-43.15±0.71
	TimeWak	**267.53±2.60**	**296.74±2.49**	**191.63±2.13**	4.91±0.96	15.73±0.96	**195.37±1.97**	34.26±0.99
fMRI	TR	8.25±0.05	8.22±0.05	8.10±0.05	8.24±0.05	7.84±0.05	11.45±0.06	23.36±0.10
	GS	321.52±0.59	319.93±0.60	312.57±0.62	286.05±1.13	116.24±0.87	320.52±1.66	275.34±2.20
	HTW	6.80±0.41	6.80±0.41	6.80±0.41	4.82±0.46	-0.70±0.46	5.95±0.45	2.56±0.41
	TabWak	204.16±0.82	205.01±0.91	215.64±1.07	154.27±2.15	**452.61±3.14**	248.19±3.95	**297.10±8.67**
	TabWak$^\top$	**743.33±0.55**	**743.16±0.59**	**742.43±0.53**	**636.28±0.67**	317.24±1.18	**614.25±0.77**	224.27±0.85
	TimeWak	595.68±1.03	601.68±0.81	601.53±0.96	459.66±1.00	112.68±0.85	498.39±0.84	189.43±1.00

insertion attack perturbs the series by randomly replacing a proportion of points (5% or 30%) in each feature with random values drawn uniformly between the feature's minimum and maximum values.

Table 2 presents the Z-scores of 64-length watermarked synthetic time series data under these attacks, and averaged over 100 trials. Random cropping at 30% proves especially challenging, with several methods showing negative Z-scores. Nevertheless, TimeWak demonstrates the best overall robustness, consistently outperforming all baselines across most attack scenarios while maintaining high generation quality and accurate watermark detection. In contrast, although HTW has better quality, it does poorly under attacks, indicating a struggle in balancing trade-offs between quality and robustness.

Interestingly, TR's detection scores further improve under certain post-processing attacks. In particular, significant gains are observed under random cropping and min-max insertion, likely due to the inherent robustness of watermarking in the Fourier domain. However, its overall performance lags behind TimeWak. Both TabWak and TabWak$^\top$ show significant degradation under attacks, particularly on MuJoCo and Energy datasets, where detection frequently fails. GS overall demonstrates strong robustness, maintaining detectability under all attacks except for 30% cropping on the ETTh dataset. However, it produces low-quality synthetic samples, highlighting the need for a time series specific watermark that can navigate the trade-offs between generation quality and watermark robustness.

5 Conclusion

Motivated by the need to ensure the traceability of synthetic time series, we propose TimeWak, the first watermarking algorithm for multivariate time series diffusion models. TimeWak embeds seeds through a temporally chained-hash and feature-wise shuffling in data space, preserving the temporal and feature dependencies and enhancing the watermark detectability. To address non-uniform error distribution in the time series diffusion process, we optimize TimeWak for ϵ-exact inversion and provide the bounded error analysis. Compared to multiple SOTA watermarking algorithms, TimeWak

balances synthetic data quality, watermark detectability, and robustness against post-editing attacks. Extensive evaluations on five datasets show that `TimeWak` improves context-FID score by 61.96% and correlational scores by 8.44%, against the strongest SOTA baseline, while maintaining strong detectability.

Limitations. While `TimeWak` does not natively support streaming data, it can be applied to streaming scenarios by processing data in small, fixed-length windows and watermarking each window independently. However, finer-grained cases, such as watermarking at the per-timestep level, remain unexplored. This represents a key limitation of `TimeWak` and an important direction for future work.

Acknowledgments

This research is partly funded by Priv-GSyn project, 200021E_229204 of Swiss National Science Foundation, the DEPMAT project, P20-22 / N21022, of the research programme Perspectief of the Dutch Research Council, and ASM International NV.

References

[1] Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Trans. Mach. Learn. Res.*, 2023.

[2] Yihao Ang, Qiang Huang, Yifan Bao, Anthony K. H. Tung, and Zhiyong Huang. Tsgbench: Time series generation benchmark. *Proc. VLDB Endow.*, 17(3):305–318, 2023.

[3] Luis Candanedo. Appliances energy prediction. UCI Machine Learning Repository, 2017. `https://doi.org/10.24432/C5VC8G`.

[4] Ping Chang, Huayu Li, Stuart F. Quan, Shuyang Lu, Shu-Fen Wung, Janet Roveda, and Ao Li. A transformer-based diffusion probabilistic model for heart rate and blood pressure forecasting in intensive care unit. *Comput. Methods Programs Biomed.*, 246:108060, 2024.

[5] Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich, and Ton Kalker. *Digital Watermarking and Steganography*. Morgan kaufmann, 2007.

[6] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, pages 8780–8794, 2021.

[7] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *ICCV*, pages 22409–22420, 2023.

[8] Maurice Fréchet. Sur la distance de deux lois de probabilité. *Annales de l'ISUP*, VI(3):183–198, 1957.

[9] Xu Guo and Yiqiang Chen. Generative AI for synthetic data generation: Methods, challenges and the future. *CoRR*, abs/2403.04190, 2024.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[11] Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir-Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. PSA-GAN: progressive self attention gans for synthetic time series. In *ICLR*, 2022.

[12] Marcel Kollovieh, Abdul Fatir Ansari, Michael Bohlke-Schneider, Jasper Zschiegner, Hao Wang, and Yuyang Wang. Predict, refine, synthesize: Self-guiding diffusion models for probabilistic time series forecasting. In *NeurIPS*, 2023.

[13] Hui Li, Yunpeng Cui, Shuo Wang, Juan Liu, Jinyuan Qin, and Yilin Yang. Multivariate financial time-series prediction with certified robustness. *IEEE Access*, 8:109133–109143, 2020.

[14] Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. Generative time series forecasting with diffusion, denoise, and disentanglement. In *NeurIPS*, 2022.

[15] Shujian Liao, Hao Ni, Marc Sabate-Vidales, Lukasz Szpruch, Magnus Wiese, and Baoren Xiao. Conditional sig-wasserstein gans for time series generation. *CoRR*, abs/2006.05421, 2020.

[16] Haksoo Lim, Minjung Kim, Sewon Park, and Noseong Park. Regular time-series generation using SGM. *CoRR*, abs/2301.08518, 2023.

[17] Lequan Lin, Zhengkun Li, Ruikun Li, Xuliang Li, and Junbin Gao. Diffusion models for time-series applications: a survey. *Frontiers Inf. Technol. Electron. Eng.*, 25(1):19–41, 2024.

[18] Yugeng Liu, Zheng Li, Michael Backes, Yun Shen, and Yang Zhang. Watermarking diffusion model. *CoRR*, abs/2305.12502, 2023.

[19] Mohammad Amin Morid, Olivia R. Liu Sheng, Kensaku Kawamoto, and Samir E. Abdelrahman. Learning hidden patterns from patient multivariate time series data using convolutional neural networks: A case study of healthcare cost prediction. *J. Biomed. Informatics*, 111:103565, 2020.

[20] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *ICML*, volume 139, pages 8857–8868, 2021.

[21] Aditya Shankar, Hans Brouwer, Rihan Hai, and Lydia Chen. SiloFuse: Cross-silo synthetic data generation with latent tabular diffusion models. In *ICDE*, pages 110–123, 2024.

[22] Stephen M. Smith, Karla L. Miller, Gholamreza Salimi Khorshidi, Matthew A. Webster, Christian F. Beckmann, Thomas E. Nichols, Joseph D. Ramsey, and Mark William Woolrich. Network modelling methods for FMRI. *NeuroImage*, 54(2):875–891, 2011.

[23] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[24] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[25] Namjoon Suh, Yuning Yang, Din-Yin Hsieh, Qitong Luan, Shirong Xu, Shixiang Zhu, and Guang Cheng. Timeautodiff: Combining autoencoder and diffusion model for time series tabular data synthesizing. *CoRR*, abs/2406.16028, 2024.

[26] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: conditional score-based diffusion models for probabilistic time series imputation. In *NeurIPS*, pages 24804–24816, 2021.

[27] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy P. Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Softw. Impacts*, 6:100022, 2020.

[28] Nicolas van Schaik. Robust watermarking in large language models for time series generation. Master's thesis, Delft University of Technology, 2024. `https://repository.tudelft.nl/record/uuid:dbbf9d34-08ba-43d6-8ec4-0240ec00a2b7`.

[29] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In *NeurIPS*, 2023.

[30] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*, 2023.

[31] Tijin Yan, Hongwei Zhang, Tong Zhou, Yufeng Zhan, and Yuanqing Xia. Scoregrad: Multivariate probabilistic time series forecasting with continuous energy-based generative models. *CoRR*, abs/2106.10121, 2021.

[32] Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. In *CVPR*, pages 12162–12171, 2024.

[33] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *NeurIPS*, pages 5509–5519, 2019.

[34] Xinyu Yuan and Yan Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. In *ICLR*, 2024.

[35] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *AAAI*, pages 8980–8987, 2022.

[36] Guoqiang Zhang, John P. Lewis, and W. Bastiaan Kleijn. Exact diffusion inversion via bidirectional integration approximation. In *ECCV (57)*, volume 15115, pages 19–36, 2024.

[37] Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *ICLR*, 2024.

[38] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Robust invisible video watermarking with attention. *CoRR*, abs/1909.01285, 2019.

[39] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, pages 11106–11115, 2021.

[40] Chaoyi Zhu, Jeroen Galjaard, Pin-Yu Chen, and Lydia Y. Chen. Duwak: Dual watermarks in large language models. In *ACL (Findings)*, pages 11416–11436, 2024.

[41] Chaoyi Zhu, Jiayi Tang, Jeroen M. Galjaard, Pin-Yu Chen, Robert Birke, Cornelis Bos, and Lydia Y. Chen. Tabwak: A watermark for tabular diffusion models. In *ICLR*, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction mention `TimeWak`'s contributions as the first generation-time watermarking scheme for multivariate time series diffusion models; using a temporal chained-hashing scheme to embed watermark seeds, and ϵ-exact inversion to detect the watermarks.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of the work are included in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumptions are detailed in Assumption 1, the verification of assumptions and derivations of proof are provided in Appendix C.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We keep the code for the proposed `TimeWak` and baselines on the following open sourced repository anonymously: `https://anonymous.4open.science/r/TimeWak`. We also provide the experiment scripts and the datasets used in this study. The implementation details can be found in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We keep the code for the proposed `TimeWak` and baselines on the following open sourced repository anonymously: `https://anonymous.4open.science/r/TimeWak`. We also provide the experiment scripts and the datasets used in this study. The implementation details can be found in Appendix D.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The implementation details can be found in Appendix D, including details of 5 datasets, evaluation metrics and baselines we used. We keep the code for the proposed `TimeWak` and baselines on the following open sourced repository anonymously: `https://anonymous.4open.science/r/TimeWak`. We also provide the experiment scripts and the datasets used in this study.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We provide the mean and standard deviation for results, providing appropriate measures for variability (see Table 1, Table 2).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources are mentioned in Section D.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper conforms in every respect to sections 'Potential Harms Caused by the Research Process', 'Societal Impact and Potential Harmful Consequences', and 'Impact Mitigation Measures' in the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Multivariate time series are an important data modality for real-world applications, e.g., health care, science and finance. The traceability of time series generative models is crucial to achieve the trustworthy and responsible analysis. Our solution of embedding watermarks on time series yields insights into auditing the use of synthetic time series to help overcome data sharing barriers.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Code is verified by all authors.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Datasets details are in Section 4.1 and Appendix D , with appropriate citations for the source(s). The datasets are widely recognized as publicly available and their use complies with standard academic practice.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Assets introduced in the paper, in the form of code, are available at the anonymized open-source repository: `https://anonymous.4open.science/r/TimeWak`.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [No]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [No]

Justification: The paper uses no LLM for any important, original, or non-standard component of the core method.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Nomenclature

α_t	Noise schedule parameter
β_t	Variance schedule for noise addition
Δ_t	Lipschitz constant for noise estimator sensitivity at time step t
ϵ	Perturbation error in inversion
γ	Scaling factor in BDIA inversion
$\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$	Noise estimator function
κ	Cryptographic key for watermarking
$\mathbb{I}[\cdot]$	Indicator function
$\mathbf{s}^{[:],f}$	Watermark seed across all timesteps at feature f
$\mathbf{s}^{w,[:]}$	Watermark seed across all features at timestep w
$\mathbf{s}^{w,f}$	Watermark seed at timestep w and feature f
$\mathbf{x}_0$	Generated sequence of length W and F features
$\mathbf{x}_T$	Noised sequence at timestep T
$\mathbf{x}_t^{approx}$	Approximate diffusion state at step t
$\mathbf{x}_t$	Diffusion state at step t
$\mathcal{H}(\kappa, w, \mathbf{s}^{w-1,[:]})$	Temporal chained hashing function
$\mathcal{N}(0, I)$	Standard normal distribution
$\mathcal{U}(0, 1)$	Continuous uniform distribution between 0 and 1
$\mathcal{U}(\{0, L-1\})^F$	Vector in $\mathbb{R}^F$ with i.i.d. discrete uniform components over $\{0, \ldots, L-1\}$
$\mathcal{W}^*$	Set of valid timesteps for comparison
Φ	CDF of standard normal distribution
Φ^{-1}	Inverse CDF of standard normal distribution
π_κ	Feature permutation function
σ_t	Standard deviation at timestep t
a_t, b_t	BDIA parameters for inversion process
F	Number of features
H	Interval length for watermark partitioning
L	Bit length
n	Number of intervals in time window
T	Total diffusion steps
W	Time window length

B Diffusion and diffusion inversion

B.1 Time series diffusion model

This section covers the necessary background knowledge for time series diffusion models.

Denoising Diffusion Probabilistic Models (DDPMs) are a powerful generative models, especially for time series synthesis [10, 1, 12, 34]. As shown in Figure 4, they work by iteratively forward noising data and then learning to invert this process through during the backward step [10]. For a time series window, one step of forward noise is given by:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon_t. \tag{16}$$

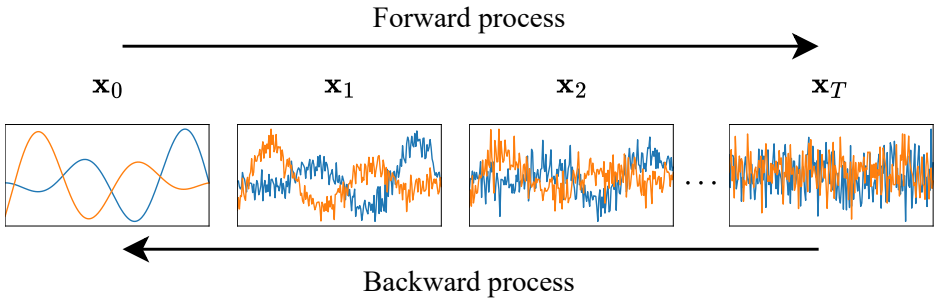


Figure 4: Forward and backward diffusion process. $\mathbf{x}_0$ denotes the initial signal window and $\mathbf{x}_T$ corresponds to the fully diffused version of the signal obtained after T forward diffusion steps.

In this formulation, $\mathbf{x}_t$ represents a multivariate time series signal after undergoing t steps of noise addition. The term β_t denotes the noise variance at step t. The noise component ϵ_t is sampled from a normal distribution, $\mathcal{N}(0, I)$. Using this notation, $\mathbf{x}_0$ and $\mathbf{x}_T$ define a noise-free sequence and a fully noised sequence, respectively, for an arbitrary window slice, where T is the total number of noise steps. Without iteratively applying Equation (16), an intermediate noising step can be efficiently computed using a *reparameterization trick* [10]:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \tag{17}$$

where $\bar{\alpha}_t$ is the cumulative product of noise reduction factors up to step t, expressed as $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s = \prod_{s=1}^{t}(1 - \beta_s)$, with β_s being the noise variance at timestep s and $\epsilon \sim \mathcal{N}(0, I)$.

The denoising process reverses the noising steps to reconstruct $\mathbf{x}_0$ from $\mathbf{x}_t$, ideally restoring the original signal. This is achieved by training a neural network, $\boldsymbol{\epsilon_\theta}$, to estimate the noise component at each step. A single reverse step is given by:

$$\tilde{\mathbf{x}}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\tilde{\mathbf{x}}_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}}\boldsymbol{\epsilon_\theta}(\tilde{\mathbf{x}}_t, t)\right) + \sigma_t z. \tag{18}$$

Here, $\tilde{\mathbf{x}}_t$ and $\tilde{\mathbf{x}}_{t-1}$ denote the signal estimates at time steps t and $t-1$, respectively, where $\tilde{\mathbf{x}}_j = \mathbf{x}_j$ at the final noising step. The term σ_t is typically a function of β to introduce stochasticity in the sampling process, and $z \sim \mathcal{N}(0, I)$ [10]. The objective is to iteratively refine the denoised sample so that $\tilde{x}_0 \approx x_0$.

B.2 Diffusion models for time series

Diffusion models show great promise in time series tasks, excelling in both forecasting [20, 1, 14] and generation [16, 37]. Among these, Denoising Diffusion Probabilistic Models (DDPMs) [10] are a leading framework, which progressively denoise samples to reconstruct data from noise [17] and rely on stochastic noise addition during sampling [10]. On the other hand, Denoising Diffusion Implicit Models (DDIMs) use a deterministic sampling process that removes noise, enabling faster sampling and fewer steps to generate high-quality samples with greater predictability [23]. However, this reduces the model's ability to explore a wide range of outputs, leading to lower diversity and reduced robustness (i.e., consistency in generating diverse and reliable samples) [10, 23].

Diffusion-TS, which serves as the backbone model of `TimeWak` presented in this paper, employs a DDPM combined with seasonal-trend decomposition to better capture underlying structures and dependencies of multivariate time series [34]. It also introduces a Fourier-based loss to optimize reconstruction, improving accuracy by better matching the frequency components [34]. Its innovation lies in the integration of seasonal-trend decomposition with DDPMs, and the use of the Fourier-based loss to enhance the model's ability to capture complex temporal patterns.

Some diffusion models that synthesize time series data include ScoreGrad [31], SSSD^{S4} [1], TSDiff [12], but some also incorporate transformer-based elements like TimeGrad [20], CSDI [26], and TDSTF [4]. Diffusion-TS [34] effectively addresses weaknesses in these models. Unlike ScoreGrad and TimeGrad which use autoregressive models [31, 20], it avoids error accumulation and slow inference over long horizons by using DDPM, a stable diffusion-based framework. It outperforms SSSD^{S4} and TSDiff [1, 12] by replacing resource-intensive S4 layers with an efficient latent layer

that simplifies handling multivariate data. Diffusion-TS handles incomplete datasets better than CSDI [26] by avoiding the need for explicit pairing of observed and missing data during training, while being more adaptable and efficient on diverse datasets compared to TDSTF [4], which struggles with real-time forecasting.

B.3 DDIM and DDIM inversion

The Denoising Diffusion Implicit Model (DDIM) [23] offers deterministic diffusion and sampling, extending the traditional Markovian diffusion process to a broader class of non-Markovian processes. Given a initial noise vector $\mathbf{x}_T$ and a neural network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ that predicts the noise $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ at each timestep t, the DDIM sampling step to generate sample $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$, is defined as:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{x}_t, t\right)}{\sqrt{\alpha_t}} \right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{x}_t, t\right) + \sigma_t \epsilon_t, \tag{19}$$

where $\alpha_1, \ldots, \alpha_T$ are computed from a predefined variance schedule, $\epsilon_t \sim \mathcal{N}(0, I)$ denotes standard Gaussian noise independent of $\mathbf{x}_t$, and the σ_t values can be varied to yield different generative processes. Setting σ_t to 0 for all t makes the sampling process deterministic:

$$\mathbf{x}_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\mathbf{x}_t + \left(\sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}}{\alpha_t} - \alpha_{t-1}} \right) \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{x}_t, t\right). \tag{20}$$

This sampling process ensures the same latent matrix $\mathbf{x}_0$ is consistently generated by a given noise matrix $\mathbf{x}_T$.

Having large T values (being limited with small steps) allows to cross the timesteps in the backward direction toward increasing noise levels, which gives out a deterministic diffusion process from $\mathbf{x}_0$ to $\mathbf{x}_T$; this is also known as DDIM inversion:

$$\mathbf{x}_{t+1} \approx \sqrt{\frac{\alpha_{t+1}}{\alpha_t}}\mathbf{x}_t + \left(\sqrt{1-\alpha_{t+1}} - \sqrt{\frac{\alpha_{t+1}}{\alpha_t} - \alpha_{t+1}} \right) \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{x}_t, t\right). \tag{21}$$

C Proof

Theorem 3.1. *Let $\{\mathbf{x}_t\}_{t=0}^T$ be the sequence of diffusion states governed by the BDIA-DDIM recurrence for a given dataset, following Equation (7). Given the noise estimator $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ follows Assumption 1. Suppose that instead of the exact terminal state $\mathbf{x}_1$, an approximation of $\mathbf{x}_1$, termed as $\mathbf{x}_1^{approx}$, is used with a small perturbation ϵ, given by:*

$$\mathbf{x}_1^{approx} = \mathbf{x}_2 = \mathbf{x}_1^{orig} + \epsilon. \tag{8}$$

Let the propagated error at time t be defined as,

$$\delta_t = \|\mathbf{x}_t^{approx} - \mathbf{x}_t^{orig}\|. \tag{9}$$

Then, for $t \geq 1$, the error is bounded by,

$$\|\delta_T\| \leq |\epsilon| \prod_{t=1}^{T-1} \left(\left| \frac{1}{\gamma} - \frac{a_t}{\gamma} + \frac{1}{a_{t+1}} \right| + \frac{b_t}{\gamma}\Delta_t + \frac{b_{t+1}}{a_{t+1}}\Delta_t \right), \tag{10}$$

where Δ_t quantifies the sensitivity of the noise estimator at timestep t.

Proof:

For $t = 1$, we have:

$$\boldsymbol{\delta}_1 = \mathbf{x}_1^{approx} - \mathbf{x}_1^{orig} = \boldsymbol{\epsilon}. \tag{22}$$

For $t = 2$, using the recurrence,

$$\mathbf{x}_2 = \frac{\mathbf{x}_0}{\gamma} - \frac{a_1\mathbf{x}_1 + b_1\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_1, 1)}{\gamma} + \left(\frac{\mathbf{x}_1}{a_2} - \frac{b_2}{a_2}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_1, 2) \right). \tag{23}$$

Subtracting the approximated and original cases and using Assumption 1, we obtain:

$$\boldsymbol{\delta}_2 = -\frac{a_1\boldsymbol{\delta}_1}{\gamma} + \frac{\boldsymbol{\delta}_1}{a_2} - \frac{b_1}{\gamma}\Delta_1\boldsymbol{\delta}_1 - \frac{b_2}{a_2}\Delta_1\boldsymbol{\delta}_1. \tag{24}$$

Thus, defining

$$C_2 = \left| \frac{1}{a_2} - \frac{a_1}{\gamma} \right| + \frac{b_1}{\gamma}\Delta_1 + \frac{b_2}{a_2}\Delta_1, \tag{25}$$

we bound

$$\|\boldsymbol{\delta}_2\| \leq C_2\|\boldsymbol{\delta}_1\| = C_2\|\boldsymbol{\epsilon}\|. \tag{26}$$

Suppose for some $t \geq 2$, there exists a constant C_t such that

$$\|\boldsymbol{\delta}_t\| \leq C_t\|\boldsymbol{\epsilon}\|. \tag{27}$$

For $t+1$, using the recurrence:

$$\mathbf{x}_{t+1} = \frac{\mathbf{x}_{t-1}}{\gamma} - \frac{a_t\mathbf{x}_t + b_t\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\gamma} + \left(\frac{\mathbf{x}_t}{a_{t+1}} - \frac{b_{t+1}}{a_{t+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t+1) \right). \tag{28}$$

Taking differences, we obtain

$$\boldsymbol{\delta}_{t+1} = \frac{\boldsymbol{\delta}_{t-1}}{\gamma} - \frac{a_t\boldsymbol{\delta}_t}{\gamma} - \frac{b_t}{\gamma}\Delta_t\boldsymbol{\delta}_t + \frac{\boldsymbol{\delta}_t}{a_{t+1}} - \frac{b_{t+1}}{a_{t+1}}\Delta_t\boldsymbol{\delta}_t. \tag{29}$$

Bounding the terms, we define:

$$C_{t+1} = C_t \left(\left| \frac{1}{\gamma} - \frac{a_t}{\gamma} + \frac{1}{a_{t+1}} \right| + \frac{b_t}{\gamma}\Delta_t + \frac{b_{t+1}}{a_{t+1}}\Delta_t \right). \tag{30}$$

Thus, we conclude:

$$\|\boldsymbol{\delta}_{t+1}\| \leq C_{t+1}\|\boldsymbol{\epsilon}\|. \tag{31}$$

By induction, we obtain:

$$C_T = \prod_{t=1}^{T-1} \left(\left| \frac{1}{\gamma} - \frac{a_i}{\gamma} + \frac{1}{a_{t+1}} \right| + \frac{b_t}{\gamma}\Delta_t + \frac{b_{t+1}}{a_{t+1}}\Delta_t \right). \tag{32}$$

Thus, the perturbation remains bounded for all T, i.e.

$$\|\delta_T\| \leq |\epsilon| \prod_{t=1}^{T-1} \left(\left| \frac{1}{\gamma} - \frac{a_t}{\gamma} + \frac{1}{a_{t+1}} \right| + \frac{b_t}{\gamma}\Delta_t + \frac{b_{t+1}}{a_{t+1}}\Delta_t \right), \tag{33}$$

completing the proof. □

To further validate Assumption 1, we empirically computed Δ_t across four different datasets, using 10,000 samples from each. Specifically, Δ_t is calculated as the maximum ratio for different t using the L_1 norm of the data samples, as illustrated in Figure 5.

We also generate 10,000 samples with 64-length sequences across five datasets, computing the final output $\mathbf{x}_0$ and the one step prior $\mathbf{x}_1$. Results in Table 3 show consistently small L_1 norm between $\mathbf{x}_1$ and $\mathbf{x}_0$, with an average of 5.1×10^{-3} to 7.0×10^{-3} and maximum of < 0.23, validating our ϵ-exact approximation. These empirically small errors confirm Theorem 3.1's theoretical bounds and demonstrate reliable watermark detection despite the approximation.

Table 3: L_1 norm between $\mathbf{x}_1$ and $\mathbf{x}_0$ for 64-length sequences over 10,000 samples.

Dataset	Avg. L_1 $(\times 10^{-3})$	Max. L_1
Stocks	7.031	0.082
ETTh	6.776	0.104
MuJoCo	5.146	0.081
Energy	5.687	0.230
fMRI	5.945	0.070

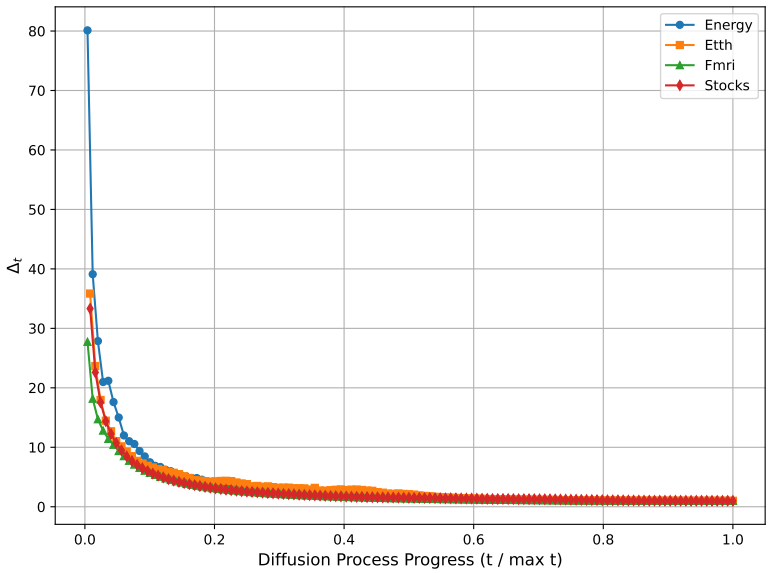


Figure 5: Δ_t for different datasets.

D Experiment details

D.1 Datasets

Table 4 shows the details of all datasets used in the experiments.

Table 4: Details of datasets used in experiments.

Dataset	Number of Rows	Number of Features	Source
Stocks	3,773	6	https://finance.yahoo.com/quote/GOOG
ETTh	17,420	7	https://github.com/zhouhaoyi/ETDataset
MuJoCo	10,000	14	https://github.com/deepmind/dm_control
Energy	19,711	28	https://archive.ics.uci.edu/ml/datasets
fMRI	10,000	50	https://www.fmrib.ox.ac.uk/datasets

D.2 Evaluation metrics

Context-FID Jeha et al. [11] introduced the Context-FID score, which is a refined adaptation of the Fréchet Inception Distance (FID) used for evaluating the similarity between real and synthetic time series distributions. Unlike traditional FID, which relies on the Inception model as a feature extractor for images, Context-FID uses TS2Vec [35], which is a specialized time series embedding model. Yue et al. [35] demonstrated that models with lower Context-FID scores tend to perform well in downstream tasks, revealing a strong correlation between Context-FID and the forecasting performance of generative models. Ultimately, a lower Context-FID score signifies a closer resemblance between real and synthetic distributions.

Correlational We calculates the covariance between the i^{th} and j^{th} features of a time series using the following equation [15]:

$$\text{Cov}_{i,j} = \frac{1}{W}\sum_{t=1}^{W} \mathrm{K}_i^t \mathrm{K}_i^t - \left(\frac{1}{W}\sum_{t=1}^{W} \mathrm{K}_i^t\right)\left(\frac{1}{W}\sum_{t=1}^{W} \mathrm{K}_j^t\right). \tag{34}$$

where W is the total number of time steps, K_i^t and K_j^t are the values of the i^{th} and j^{th} features at time step t, and the summations compute the average product of these values, subtracting the product

of their individual means. To assess the correlation between real and synthetic time series, we use the following metric [34]:

$$\frac{1}{10}\sum_{i,j}^{d}\left|\frac{\mathrm{Cov}_{i,j}^{R}}{\sqrt{\mathrm{Cov}_{i,i}^{R}\mathrm{Cov}_{j,j}^{R}}}-\frac{\mathrm{Cov}_{i,j}^{S}}{\sqrt{\mathrm{Cov}_{i,i}^{S}\mathrm{Cov}_{j,j}^{S}}}\right|, \tag{35}$$

where Cov is the covariance between its subscripts ($\langle i,i\rangle$, $\langle j,j\rangle$, $\langle i,j\rangle$), where i and j are the features in the real (denoted by R) and synthetic (denoted by S) time series data, and d is the total number of features, with the summation taken over all feature pairs.

Discriminative The discriminative score is computed as $|\text{accuracy} - 0.5|$, quantifying the model's ability to distinguish between real and synthetic time series. A lower score indicates better performance, as it indicates greater difficulty in differentiation, implying a higher degree of similarity between the two distributions. To ensure consistency, we follow the experimental setup of TimeGAN [33] by using a two-layer GRU-based neural network as the classifier.

Predictive The predictive score is evaluated using the mean absolute error (MAE) between the predicted and actual values on the test data. Again, we use the experimental setup of TimeGAN [33] by using a two-layer GRU-based neural network for sequence prediction.

Z-score The Z-score is a statistical measure used to assess watermark detectability by quantifying the deviation between watermarked and non-watermarked samples. It facilitates hypothesis testing, where the null hypothesis H_0 states that a given sample is not watermarked by the corresponding watermarking method. A sufficiently high positive Z-score provides evidence against H_0, suggesting the presence of a watermark.

For sample-wise bit accuracy in time series, the Z-score is computed as $Z = \frac{\mu_{\text{Acc, W}} - \mu_{\text{Acc, NW}}}{\sigma_{\text{Acc, NW}}/\sqrt{n}}$, where $\mu_{\text{Acc, W}}$ and $\mu_{\text{Acc, NW}}$ are the mean bit accuracy of watermarked and non-watermarked samples, respectively, $\sigma_{\text{Acc, NW}}$ is the standard deviation of bit accuracy in the non-watermarked samples, and n is the number of watermarked samples. Under H_0, the expected difference in means is negligible, resulting in a Z-score close to zero. A large positive Z-score provides statistical evidence for the presence of a watermark.

For the Tree-Ring watermarking method, the Z-score is computed in the Fourier domain instead of the sample-wise bit accuracy domain. It measures the deviation in the amplitude spectrum between watermarked and non-watermarked samples, given by $Z = \frac{\mu_{\mathcal{F}_{\text{NW}}} - \mu_{\mathcal{F}_{\text{W}}}}{\sigma_{\mathcal{F}_{\text{NW}}}}$ where $\mathcal{F}_{\text{W}}$ and $\mathcal{F}_{\text{NW}}$ denote the Fourier amplitude spectrum of watermarked and non-watermarked samples, respectively, with μ and σ representing their mean and standard deviation. Since this method does not rely on per-sample bit accuracy, the test statistic is independent of n, and the computation utilizes the opposite tail of the distribution.

TPR@X%FPR This metric measures the True Positive Rate (TPR) at a specified False Positive Rate (FPR) of X%, where X% denotes a fixed false positive threshold. It reflects the effectiveness of watermark detection by quantifying how reliably watermarked samples are identified under controlled false positive conditions. A higher TPR@X%FPR indicates stronger detection performance and greater robustness of the watermarking method.

D.3 Baselines

Tree-Ring We adapt the latent-representation-based Tree-Ring watermark for multivariate time series in the data space. Initially, the Tree-Ring watermark was proposed for images with square dimensions as it places the circular ring watermark pattern centrally. However, multivariate time series often have rectangular dimensions with varying numbers of features and timesteps. To accommodate this structure, we apply a flexible ring pattern with a predefined radius indicating the outermost circle of the watermark. Finally, we embed and detect the watermark in the Fourier domain.

Gaussian Shading Similar to Tree-Ring, we implement Gaussian Shading from the image domain to multivariate time series by embedding the watermark directly in the data space. To maintain

coherence and efficiency, we use a single control seed across all samples to avoid the need for additional indices for each sample.

Heads Tails Watermark The HTW is a post-watermarking technique designed for univariate time series. To extend its applicability, we adapt it for multivariate time series by iterating through each variate in the generated synthetic sample. During evaluation, the watermarked time series is reversed, and processing each series independently. In short, we treat each variate as a single series.

TabWak TabWak was originally designed for the tabular data domain, where it operates effectively in the latent space. We extend its application to multivariate time series in the data space. However, tabular data primarily captures feature dependencies, while time series data inherently relies on both feature and temporal dependencies. Therefore, we implement another version called TabWak$^\top$ by transposing the watermarking direction onto the temporal axis. Similarly, we evaluate the watermark detectability on a sample by sample basis.

D.4 Training and sampling

All code implementations are done in PyTorch (version 2.3.1) using a single NVIDIA GeForce RTX 2080 Graphics Card coupled with an Intel(R) Xeon(R) Platinum 8562Y+ CPU for all experiments. Dataset splits are 80% for training and 20% for testing. Table 5 shows training and sampling time for all datasets across window of sizes 24, 64 and 128. We train the time series diffusion model following the Diffusion-TS settings [34], and generate 10,000 watermarked synthetic samples using `TimeWak` for each sampling run.

Table 5: Details of training and sampling time.

Dataset	Window Size	Training Time ($\sim$ min.)	Sampling Time ($\sim$ min.)
Stocks	24	6.1	0.4
	64	6.2	2.1
	128	8.0	4.6
ETTh	24	11.6	0.4
	64	12.7	2.4
	128	13.7	5.2
MuJoCo	24	9.7	0.7
	64	9.9	4.9
	128	10.6	12.6
Energy	24	23.5	1.4
	64	24.2	10.5
	128	24.3	23.0
fMRI	24	16.8	1.8
	64	17.5	13.5
	128	18.1	28.5

E Additional experimental results

E.1 Quality performance

Table 6 shows the quality of synthetic time series generated in 64 and 128 window sizes. `TimeWak` remains stable across all datasets and even comparable to the quality of non-watermarked samples.

Table 6: Results of synthetic time series quality. No watermarking ('W/O') is included.

		64-length ↓				128-length ↓			
Dataset	Method	Context-FID	Correlational	Discriminative	Predictive	Context-FID	Correlational	Discriminative	Predictive
Stocks	W/O	0.444±0.114	0.030±0.023	0.104±0.014	0.037±0.000	0.536±0.135	0.020±0.015	0.148±0.019	0.037±0.000
	TR	1.812±0.210	0.102±0.012	0.206±0.038	0.038±0.001	3.555±0.868	0.130±0.025	0.276±0.030	0.041±0.004
	GS	1.643±0.126	0.011±0.005	0.252±0.072	0.042±0.001	2.647±0.262	0.025±0.015	0.186±0.071	0.040±0.000
	HTW	0.426±0.073	0.026±0.016	0.105±0.032	0.037±0.000	0.622±0.092	0.017±0.013	0.152±0.086	0.037±0.000
	TabWak	0.242±0.045	0.011±0.008	0.141±0.021	0.037±0.000	0.394±0.034	0.009±0.009	0.150±0.031	0.037±0.000
	TabWak$^\top$	0.284±0.091	0.005±0.005	0.099±0.025	0.037±0.000	0.367±0.035	0.010±0.007	0.129±0.054	0.037±0.000
	`TimeWak`	0.387±0.054	0.017±0.017	0.092±0.041	0.037±0.000	0.316±0.044	0.021±0.024	0.140±0.029	0.037±0.000
ETTh	W/O	0.384±0.034	0.070±0.011	0.106±0.009	0.115±0.006	1.086±0.070	0.098±0.021	0.166±0.013	0.111±0.008
	TR	2.224±0.209	0.217±0.011	0.284±0.032	0.131±0.004	2.450±0.128	0.270±0.017	0.284±0.063	0.138±0.005
	GS	3.398±0.384	0.248±0.025	0.356±0.029	0.154±0.001	4.998±0.603	0.233±0.027	0.381±0.042	0.162±0.010
	HTW	0.372±0.027	0.076±0.006	0.118±0.024	0.116±0.006	1.119±0.065	0.091±0.014	0.165±0.010	0.120±0.006
	TabWak	0.597±0.044	0.346±0.027	0.176±0.020	0.122±0.007	1.931±0.254	0.466±0.058	0.248±0.017	0.128±0.005
	TabWak$^\top$	0.503±0.045	0.095±0.043	0.119±0.012	0.120±0.010	1.477±0.075	0.129±0.037	0.143±0.014	0.112±0.011
	`TimeWak`	0.297±0.038	0.133±0.040	0.097±0.015	0.115±0.003	1.090±0.100	0.135±0.057	0.174±0.007	0.110±0.009
MuJoCo	W/O	0.103±0.016	0.341±0.025	0.023±0.015	0.007±0.000	0.179±0.011	0.290±0.025	0.055±0.027	0.005±0.001
	TR	2.627±0.230	0.948±0.064	0.270±0.128	0.016±0.005	2.348±0.236	0.865±0.025	0.358±0.008	0.008±0.001
	GS	7.162±1.301	1.034±0.087	0.447±0.011	0.011±0.002	4.797±1.290	1.335±0.041	0.454±0.023	0.007±0.001
	HTW	0.316±0.034	0.334±0.044	0.248±0.079	0.011±0.001	0.431±0.051	0.309±0.064	0.255±0.174	0.009±0.001
	TabWak	0.372±0.064	0.671±0.083	0.137±0.027	0.007±0.001	0.369±0.073	0.589±0.059	0.175±0.042	0.006±0.002
	TabWak$^\top$	0.238±0.019	0.339±0.059	0.087±0.025	0.006±0.001	0.275±0.033	0.333±0.066	0.084±0.028	0.006±0.001
	`TimeWak`	0.108±0.014	0.413±0.062	0.038±0.021	0.007±0.001	0.155±0.016	0.316±0.022	0.046±0.030	0.005±0.001
Energy	W/O	0.112±0.016	1.032±0.289	0.124±0.008	0.251±0.001	0.120±0.011	0.798±0.213	0.202±0.073	0.249±0.000
	TR	0.902±0.093	3.503±0.589	0.427±0.072	0.307±0.005	1.195±0.111	2.303±0.569	0.498±0.001	0.287±0.003
	GS	2.205±0.192	3.277±0.129	0.479±0.010	0.310±0.005	3.680±0.444	4.233±0.287	0.474±0.039	0.280±0.006
	HTW	0.133±0.015	1.045±0.357	0.135±0.013	0.251±0.001	0.133±0.017	0.822±0.310	0.103±0.043	0.249±0.001
	TabWak	0.168±0.021	1.811±0.530	0.136±0.013	0.251±0.000	0.201±0.020	2.001±0.440	0.156±0.077	0.250±0.001
	TabWak$^\top$	0.237±0.029	1.321±0.252	0.138±0.015	0.252±0.000	0.274±0.019	1.211±0.196	0.136±0.023	0.249±0.001
	`TimeWak`	0.143±0.019	1.662±0.298	0.145±0.019	0.251±0.000	0.148±0.027	1.687±0.328	0.140±0.057	0.249±0.000
fMRI	W/O	0.435±0.033	1.899±0.075	0.268±0.150	0.100±0.000	0.859±0.058	1.823±0.064	0.209±0.263	0.100±0.000
	TR	3.358±0.402	13.699±0.132	0.411±0.158	0.141±0.001	5.391±0.919	13.000±0.084	0.434±0.072	0.149±0.002
	GS	0.756±0.098	8.378±0.028	0.500±0.001	0.104±0.001	1.046±0.052	5.941±0.048	0.499±0.001	0.106±0.001
	HTW	0.413±0.017	1.822±0.111	0.338±0.032	0.100±0.000	0.811±0.051	1.757±0.053	0.063±0.064	0.100±0.000
	TabWak	0.655±0.137	6.532±0.158	0.242±0.302	0.108±0.001	1.114±0.135	5.967±0.033	0.153±0.218	0.111±0.001
	TabWak$^\top$	0.554±0.049	1.955±0.058	0.331±0.039	0.100±0.000	0.919±0.024	1.807±0.024	0.265±0.201	0.100±0.000
	`TimeWak`	0.441±0.035	1.786±0.043	0.314±0.041	0.100±0.000	0.855±0.072	1.704±0.060	0.298±0.227	0.100±0.000

E.2 Post-editing attacks

Table 7–8 show the detectability results of several post-editing attacks on 24 and 128 window sizes, respectively.

Table 7: Results of robustness against post-editing attacks. Average Z-score on 24-length sequences, including un-attacked scores from Table 1. Best results are in **bold**, and second-best are underlined.

Dataset	Method	Without Attack	Offset ↑		Random Crop ↑		Min-Max Insertion ↑	
			5%	30%	5%	30%	5%	30%
Stocks	TR	0.43±0.04	0.41±0.04	0.26±0.05	61.01±1.18	**76.69±1.17**	0.18±0.05	0.81±0.08
	GS	86.07±0.74	85.74±0.71	84.38±0.67	-34.10±0.69	3.08±0.58	82.31±0.75	**57.07±0.72**
	HTW	4.45±0.62	4.45±0.62	4.45±0.62	-0.30±0.39	-0.40±0.30	3.96±0.72	1.80±0.71
	TabWak	-67.22±1.17	-66.62±1.23	-75.89±1.10	-137.99±1.23	-72.71±1.76	-70.11±1.14	-89.22±0.64
	TabWak$^\top$	55.39±0.86	55.59±0.74	54.76±0.78	9.32±0.94	7.43±0.73	45.69±0.73	21.19±0.56
	TimeWak	**182.10±0.73**	**181.98±0.84**	**180.68±0.77**	**62.29±1.01**	9.33±1.04	**155.72±0.91**	56.86±1.06
ETTh	TR	7.84±0.12	7.94±0.12	8.19±0.13	26.95±0.32	**34.36±0.26**	14.11±0.28	28.72±0.50
	GS	101.07±1.17	99.45±1.09	105.70±1.08	**98.99±1.21**	2.10±1.37	98.78±1.13	**73.67±1.30**
	HTW	3.43±0.83	3.43±0.83	3.43±0.83	0.38±0.77	-1.08±0.47	3.09±0.82	1.54±0.66
	TabWak	-14.95±1.08	-7.92±0.97	36.49±1.08	-11.76±1.31	8.65±1.38	-7.95±0.93	21.70±0.82
	TabWak$^\top$	109.35±0.91	110.22±0.85	103.34±0.82	50.64±0.96	1.06±1.02	80.98±0.87	21.83±0.95
	TimeWak	**134.83±0.95**	**130.50±0.99**	**118.65±1.03**	35.30±1.14	5.99±1.04	**101.53±0.96**	20.77±0.95
MuJoCo	TR	1.38±0.03	1.34±0.03	1.22±0.03	5.31±0.05	5.65±0.06	2.42±0.05	5.53±0.09
	GS	21.13±0.77	23.95±0.71	27.15±0.76	23.06±0.86	-10.38±0.72	22.89±0.83	22.89±1.06
	HTW	2.89±0.54	2.89±0.54	2.89±0.54	0.85±0.48	-0.47±0.40	2.65±0.53	1.51±0.43
	TabWak	31.30±1.07	31.59±0.96	39.05±1.05	38.80±1.11	**25.04±1.13**	31.60±0.92	28.70±0.59
	TabWak$^\top$	-4.85±0.87	-6.27±0.90	0.74±0.78	-15.92±0.88	-56.15±1.50	-10.95±1.01	-43.08±1.17
	TimeWak	**85.69±1.08**	**77.78±1.26**	**49.48±1.12**	**48.03±1.17**	11.46±1.18	**74.94±1.10**	**38.87±1.14**
Energy	TR	9.51±0.09	9.30±0.10	8.77±0.08	**110.47±0.23**	**129.09±0.19**	17.65±0.15	35.69±0.18
	GS	51.22±0.88	50.27±0.96	47.31±0.97	11.65±0.72	38.70±1.34	48.80±0.97	39.80±0.94
	HTW	3.06±0.36	3.06±0.36	3.06±0.36	0.56±0.35	-0.13±0.23	2.81±0.36	1.61±0.30
	TabWak	3.26±0.89	0.52±1.03	-2.91±1.08	3.27±0.63	3.54±0.53	2.20±0.87	-2.08±0.49
	TabWak$^\top$	40.82±0.81	38.47±0.72	40.20±0.82	33.83±1.08	-7.47±1.50	31.89±0.89	5.37±0.93
	TimeWak	**231.28±1.45**	**228.22±1.71**	**185.48±1.52**	-9.20±0.96	-1.54±1.02	**189.15±1.44**	**56.39±1.16**
fMRI	TR	6.49±0.05	6.43±0.05	6.12±0.05	5.42±0.05	4.40±0.05	5.54±0.05	1.37±0.07
	GS	420.02±1.44	416.83±1.43	401.86±1.72	360.27±1.40	171.38±1.16	386.10±1.54	**245.44±1.28**
	HTW	4.32±0.22	4.32±0.22	4.32±0.22	2.92±0.27	-0.39±0.28	3.86±0.26	1.78±0.25
	TabWak	84.02±1.04	83.55±1.08	78.94±1.16	78.63±1.24	27.05±1.30	76.49±1.17	50.55±1.33
	TabWak$^\top$	**464.67±0.50**	**464.04±0.48**	**458.81±0.57**	**400.47±0.72**	**202.57±1.04**	**403.49±0.66**	168.64±0.75
	TimeWak	379.51±0.82	378.95±0.85	374.99±0.78	277.64±0.78	77.74±1.05	327.13±0.86	133.68±0.85

Table 8: Results of robustness against post-editing attacks. Average Z-score on 128-length sequences, including un-attacked scores from Table 1. Best results are in **bold**, and second-best are underlined.

Dataset	Method	Without Attack	Offset ↑		Random Crop ↑		Min-Max Insertion ↑	
			5%	30%	5%	30%	5%	30%
Stocks	TR	0.08±0.10	0.05±0.10	0.02±0.12	**93.62±0.99**	**119.45±1.16**	6.87±0.22	23.98±0.55
	GS	172.23±1.08	167.39±1.10	145.89±1.83	13.22±1.43	-2.23±1.03	144.05±0.98	64.79±1.01
	HTW	10.39±1.33	10.39±1.33	10.39±1.33	-1.30±0.44	-1.42±0.33	9.05±1.67	3.76±1.63
	TabWak	-10.49±1.28	49.98±3.31	36.78±3.15	69.44±3.32	70.32±2.93	9.26±1.41	33.81±0.81
	TabWak$^\top$	129.39±0.90	131.70±0.97	129.83±1.08	7.70±1.04	-4.86±1.20	80.70±1.03	15.70±0.57
	TimeWak	**550.05±1.18**	**535.16±0.96**	**525.09±1.20**	83.81±2.15	15.42±1.02	**385.68±2.48**	**85.38±1.37**
ETTh	TR	6.18±0.16	6.14±0.16	6.08±0.14	24.37±0.28	36.22±0.21	23.02±0.25	55.97±0.40
	GS	327.47±4.44	322.35±5.15	**296.89±5.10**	**189.84±2.98**	8.49±1.16	**296.61±4.10**	**172.81±2.05**
	HTW	6.84±2.22	6.84±2.22	6.84±2.22	0.79±1.90	-2.94±1.15	6.00±2.16	2.59±1.54
	TabWak	-20.57±0.96	-18.36±1.21	-3.56±1.91	37.32±2.48	**78.54±2.07**	-7.75±0.79	22.65±0.45
	TabWak$^\top$	235.03±1.48	227.57±1.33	194.84±1.65	101.64±1.73	23.82±1.45	155.78±1.24	23.70±0.91
	TimeWak	**340.36±2.06**	**333.66±1.93**	276.99±2.04	107.52±1.90	7.13±1.14	244.68±1.72	44.75±1.21
MuJoCo	TR	1.31±0.04	1.33±0.04	1.50±0.04	6.93±0.07	6.08±0.05	10.52±0.12	25.61±0.20
	GS	39.63±0.71	38.07±0.67	32.10±0.81	37.22±0.89	10.61±0.85	38.46±0.89	31.58±1.16
	HTW	4.20±1.37	4.20±1.37	4.20±1.37	1.73±1.18	-2.13±0.99	3.78±1.29	2.02±0.90
	TabWak	-3.00±1.04	-5.68±0.92	-3.91±1.16	-8.02±1.07	16.29±0.61	-0.31±0.87	2.90±0.55
	TabWak$^\top$	3.91±0.88	5.10±1.00	9.07±0.97	3.36±0.90	**24.65±1.15**	25.56±1.00	**41.50±1.00**
	TimeWak	**123.36±1.43**	**139.07±1.49**	**111.27±1.15**	**93.19±1.46**	12.44±1.01	**71.22±1.21**	7.26±0.95
Energy	TR	22.58±0.19	21.92±0.18	20.13±0.17	**167.48±0.64**	**207.75±0.39**	68.40±0.29	**148.79±0.50**
	GS	45.42±1.05	43.41±0.96	30.53±0.93	71.60±0.94	67.39±1.11	44.20±1.16	44.13±1.18
	HTW	5.42±1.00	5.42±1.00	5.42±1.00	0.24±0.62	-1.11±0.43	4.86±0.95	2.47±0.67
	TabWak	0.57±0.87	-21.06±0.96	-28.01±1.32	-25.69±0.82	-10.44±0.76	-0.07±0.63	-7.15±0.36
	TabWak$^\top$	26.00±1.12	33.08±0.98	19.25±0.95	9.30±1.14	10.84±1.92	24.21±0.95	33.38±0.69
	TimeWak	**245.37±2.88**	**307.31±1.98**	**183.59±1.74**	9.57±0.97	2.48±0.84	**171.81±1.77**	21.44±1.10
fMRI	TR	9.94±0.04	9.93±0.04	9.88±0.04	10.47±0.05	8.96±0.04	16.62±0.06	39.21±0.13
	GS	701.90±0.72	701.47±0.65	699.01±0.63	621.08±1.08	256.13±1.63	667.14±2.43	**528.66±2.82**
	HTW	9.43±0.61	9.43±0.61	9.43±0.61	6.75±0.67	-1.09±0.65	8.24±0.66	3.45±0.57
	TabWak	47.29±0.83	43.52±0.79	24.95±0.86	-57.42±2.04	71.03±3.25	78.43±4.67	0.17±8.00
	TabWak$^\top$	**1031.96±0.79**	**1031.53±0.78**	**1030.14±0.72**	**889.62±0.84**	**383.15±1.37**	**801.16±0.83**	229.37±1.15
	TimeWak	526.81±13.12	834.78±1.10	834.62±1.24	632.77±1.07	160.62±1.02	651.80±1.13	216.50±0.94

E.3 Reconstruction attack

We implement the reconstruction attack using the original diffusion model. Specifically, we first applied the q-sampling process up to half of the total diffusion steps (i.e., midpoint timestep), and then performed reverse sampling starting from this midpoint. The results obtained from this approach are presented in Table 9. We observe that although the Z-score decreases, our watermark remains detectable.

Table 9: Results of synthetic time series quality and watermark detectability. Comparing `TimeWak` and `TimeWak` $_{recon}$ under reconstruction attack. Quality metrics and Z-score are for 24-length sequences.

Dataset	Method	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	`TimeWak`	0.277±0.019	0.020±0.018	0.120±0.039	0.038±0.000	182.10±0.73
	`TimeWak` $_{recon}$	4.570±0.502	0.031±0.028	0.393±0.073	0.148±0.017	179.41±0.81
ETTh	`TimeWak`	0.237±0.017	0.212±0.043	0.102±0.014	0.122±0.002	134.83±0.95
	`TimeWak` $_{recon}$	1.743±0.225	0.138±0.012	0.290±0.009	0.158±0.002	82.11±2.58
MuJoCo	`TimeWak`	0.089±0.017	0.532±0.137	0.044±0.021	0.008±0.001	85.69±1.08
	`TimeWak` $_{recon}$	0.925±0.047	0.622±0.063	0.261±0.010	0.008±0.002	81.97±1.33
Energy	`TimeWak`	0.121±0.016	1.977±0.750	0.142±0.008	0.254±0.000	231.28±1.45
	`TimeWak` $_{recon}$	5.158±0.568	6.678±0.087	0.444±0.006	0.263±0.002	39.99±2.03
fMRI	`TimeWak`	0.199±0.010	2.006±0.053	0.122±0.033	0.100±0.000	379.51±0.82
	`TimeWak` $_{recon}$	0.595±0.036	2.374±0.105	0.431±0.017	0.102±0.000	457.90±0.81

E.4 BDIA-DDIM on other baselines

The baselines in our main experiments were not evaluated with BDIA-DDIM. This is because BDIA-DDIM tends to degrade data quality compared to standard DDIM, representing a trade-off for achieving lower inversion error. For many baselines, such as Tree-Ring and Gaussian Shading, the generated quality is already poor. Applying BDIA-DDIM in these cases might improve detectability but would further deteriorate quality, making the results less meaningful. And for TabWak, we include results with BDIA-DDIM applied to both TabWak and TabWak$^\top$ in Table 10. While BDIA-DDIM improves detectability for both variants, it comes at the cost of further quality degradation compared to the original results in Table 1. In contrast, `TimeWak` maintains stable performance across both quality metrics and Z-score, highlighting its robustness.

Table 10: Results of synthetic time series quality and watermark detectability. All results are applied with BDIA-DDIM. Quality metrics are for 24-length sequences.

		Quality Metric ↓				Z-score ↑		
Dataset	Method	Context-FID	Correlational	Discriminative	Predictive	24-length	64-length	128-length
Stocks	TabWak	0.267±0.042	0.017±0.014	0.122±0.029	0.039±0.000	43.10±0.75	92.56±0.37	89.18±0.39
	TabWak$^\top$	0.273±0.103	0.011±0.008	0.115±0.042	0.037±0.000	117.12±0.15	170.41±0.14	267.27±0.18
	`TimeWak`	0.277±0.019	0.020±0.018	0.120±0.039	0.038±0.000	182.10±0.73	395.34±1.24	550.05±1.18
ETTh	TabWak	0.431±0.033	0.436±0.020	0.133±0.029	0.134±0.002	1.73±0.76	16.26±0.98	31.59±0.87
	TabWak$^\top$	0.454±0.049	0.132±0.018	0.104±0.024	0.119±0.006	149.16±0.62	220.72±0.84	315.34±1.16
	`TimeWak`	0.237±0.017	0.212±0.043	0.102±0.014	0.122±0.002	134.83±0.95	236.08±1.63	340.36±2.06
MuJoCo	TabWak	0.489±0.036	0.958±0.067	0.204±0.056	0.010±0.004	13.97±1.07	20.52±0.85	4.23±0.93
	TabWak$^\top$	0.270±0.024	0.378±0.033	0.128±0.015	0.008±0.002	68.32±1.14	93.55±1.06	228.10±1.67
	`TimeWak`	0.089±0.017	0.532±0.137	0.044±0.021	0.008±0.001	85.69±1.08	56.45±1.26	123.36±1.43
Energy	TabWak	0.189±0.022	2.915±0.410	0.166±0.012	0.255±0.000	3.10±0.63	-9.67±0.72	-18.64±0.71
	TabWak$^\top$	0.199±0.007	1.648±0.188	0.137±0.019	0.265±0.004	246.86±1.30	258.45±1.48	302.78±1.98
	`TimeWak`	0.121±0.016	1.977±0.750	0.142±0.008	0.254±0.000	231.28±1.45	267.53±2.60	245.37±2.88
fMRI	TabWak	0.319±0.019	6.772±0.129	0.484±0.007	0.110±0.001	57.99±1.06	268.67±0.88	-94.33±0.99
	TabWak$^\top$	0.317±0.028	2.185±0.148	0.218±0.031	0.100±0.000	471.86±0.45	739.39±0.60	1014.93±0.74
	`TimeWak`	0.199±0.010	2.006±0.053	0.122±0.033	0.100±0.000	379.51±0.82	595.68±1.03	526.81±13.12

E.5 Watermarked dataset on downstream tasks

We implement time series forecasting and imputation as the downstream tasks, i.e, taking either the real, synthetic, or watermarked synthetic data to build a diffusion model that can predict the future or missing values of time series. We compared the mean squared error (MSE) between the predicted and actual values and summarized the results in Table 11 and Table 12. The results indicate that training on watermarked synthetic data has a minimal impact on forecasting and imputation performance compared to training on non-watermarked synthetic data.

Table 11: Results of 64-length time series forecasting that trains on real and synthetic data ($\text{Synth}_{\text{W/O}}$ and $\text{Synth}_{\texttt{TimeWak}}$) and tests on real data with a 24 timesteps forecast horizon. MSE values $\times 10^{-3}$.

Dataset	Training Data	MSE ↓
Stocks	Real	2.119
	$\text{Synth}_{\text{W/O}}$	2.022
	$\text{Synth}_{\texttt{TimeWak}}$	2.014
ETTh	Real	6.678
	$\text{Synth}_{\text{W/O}}$	8.541
	$\text{Synth}_{\texttt{TimeWak}}$	8.655
MuJoCo	Real	1.312
	$\text{Synth}_{\text{W/O}}$	1.615
	$\text{Synth}_{\texttt{TimeWak}}$	1.741
Energy	Real	12.715
	$\text{Synth}_{\text{W/O}}$	13.480
	$\text{Synth}_{\texttt{TimeWak}}$	13.717
fMRI	Real	36.423
	$\text{Synth}_{\text{W/O}}$	67.796
	$\text{Synth}_{\texttt{TimeWak}}$	67.944

Table 12: Results of 64-length time series imputation that trains on real and synthetic data ($\text{Synth}_{\text{W/O}}$ and $\text{Synth}_{\texttt{TimeWak}}$) and tests on real data with 70% missing ratio. MSE values $\times 10^{-3}$.

Dataset	Training Data	MSE ↓
Stocks	Real	1.020
	$\text{Synth}_{\text{W/O}}$	0.855
	$\text{Synth}_{\texttt{TimeWak}}$	0.858
ETTh	Real	1.526
	$\text{Synth}_{\text{W/O}}$	1.842
	$\text{Synth}_{\texttt{TimeWak}}$	1.963
MuJoCo	Real	0.101
	$\text{Synth}_{\text{W/O}}$	0.364
	$\text{Synth}_{\texttt{TimeWak}}$	0.387
Energy	Real	7.926
	$\text{Synth}_{\text{W/O}}$	8.258
	$\text{Synth}_{\texttt{TimeWak}}$	8.279
fMRI	Real	27.439
	$\text{Synth}_{\text{W/O}}$	45.412
	$\text{Synth}_{\texttt{TimeWak}}$	47.349

E.6 Ablation study

To assess the effectiveness of the `TimeWak`, we compare its full version with three distinct variants outlined in Table 13. Table 14 presents the quality of the watermarked time series data for sequences of length 24, and the detectability of the watermarks across lengths of 24, 64 and 128. `TimeWak` demonstrates a comparable quality performance and high detectability.

Table 13: List of methods, including `TimeWak`, to be compared.

Method	Watermarking Direction	Sampling
SpatDDIM	Spatial	DDIM
SpatBDIA	Spatial	BDIA-DDIM
TempDDIM	Temporal	DDIM
`TimeWak`	Temporal	BDIA-DDIM

Table 14: Results of synthetic time series quality and watermark detectability. Quality metrics are for 24-length sequences. All methods are originally found in Table 13. Best results are in **bold**, and second-best are underlined.

		Quality Metric ↓				Z-score ↑		
Dataset	Method	Context-FID	Correlational	Discriminative	Predictive	24-length	64-length	128-length
Stocks	SpatDDIM	0.233±0.025	**0.012±0.005**	0.127±0.019	**0.037±0.000**	11.06±1.07	0.18±0.77	-1.03±0.96
	SpatBDIA	**0.199±0.024**	0.024±0.028	0.124±0.023	**0.037±0.000**	126.97±1.43	156.84±0.80	170.13±1.59
	TempDDIM	0.277±0.054	0.013±0.005	0.124±0.043	0.038±0.000	25.14±0.89	43.24±0.87	58.82±0.93
	`TimeWak`	0.277±0.019	0.020±0.018	**0.120±0.039**	0.038±0.000	**182.10±0.73**	**395.34±1.24**	**550.05±1.18**
ETTh	SpatDDIM	0.249±0.020	**0.145±0.026**	**0.094±0.011**	0.124±0.002	10.80±1.06	11.13±0.81	7.29±0.95
	SpatBDIA	0.246±0.015	0.150±0.034	0.098±0.011	0.123±0.007	28.11±1.10	40.67±1.21	47.73±1.04
	TempDDIM	0.249±0.013	0.150±0.020	0.097±0.010	**0.121±0.005**	63.78±0.97	102.06±1.16	152.63±1.29
	`TimeWak`	**0.237±0.017**	0.212±0.043	0.102±0.014	0.122±0.002	**134.83±0.95**	**236.08±1.63**	**340.36±2.06**
MuJoCo	SpatDDIM	0.091±0.008	0.476±0.049	0.062±0.027	0.008±0.002	-4.62±0.91	15.42±0.96	7.12±1.22
	SpatBDIA	0.090±0.016	0.491±0.078	0.051±0.023	0.008±0.002	11.73±0.94	18.02±0.95	25.42±0.90
	TempDDIM	0.098±0.010	**0.450±0.047**	0.059±0.027	**0.007±0.001**	21.85±0.84	-0.88±0.99	-2.58±1.03
	`TimeWak`	**0.089±0.017**	0.532±0.137	**0.044±0.021**	0.008±0.001	**85.69±1.08**	**56.45±1.26**	**123.36±1.43**
Energy	SpatDDIM	0.135±0.021	1.814±0.373	0.142±0.013	**0.253±0.000**	1.55±0.90	6.75±1.00	-0.70±0.98
	SpatBDIA	0.142±0.027	2.104±0.254	0.149±0.025	**0.253±0.000**	52.86±0.90	63.56±1.26	81.46±1.11
	TempDDIM	**0.110±0.019**	**1.724±0.270**	0.142±0.023	0.254±0.000	1.72±0.92	3.64±0.90	2.24±0.93
	`TimeWak`	0.121±0.016	1.977±0.750	**0.142±0.008**	0.254±0.000	**231.28±1.45**	**267.53±2.60**	**245.37±2.88**
fMRI	SpatDDIM	0.198±0.023	2.014±0.046	0.139±0.030	0.101±0.001	90.69±0.94	61.12±0.87	81.31±0.70
	SpatBDIA	**0.188±0.004**	**1.974±0.074**	0.124±0.035	0.101±0.000	76.48±0.89	93.74±0.82	75.93±0.66
	TempDDIM	0.193±0.018	2.097±0.086	0.143±0.020	0.101±0.000	**381.15±0.81**	**617.91±0.98**	**828.89±1.01**
	`TimeWak`	0.199±0.010	2.006±0.053	**0.122±0.033**	**0.100±0.000**	379.51±0.82	595.68±1.03	526.81±13.12

E.7 Challenging time series datasets

We evaluate `TimeWak` on 2 additional and more challenging real-world datasets, (i) the ILI dataset, which records influenza-like illness cases in the United States, and (ii) the Weather dataset, which is sparse and noisy. Both datasets are standard benchmarks and are used in TimesNet [30]. Results run on these datasets are shown in Table 15, where `TimeWak` achieves robust watermark detectability while preserving the quality of the synthetic data.

Table 15: Results of synthetic time series quality and watermark detectability for 64-length sequences. No watermarking ('W/O') is included. Best results are in **bold**, and second-best are underlined.

Dataset	Method	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Illness	W/O	0.411±0.040	0.073±0.061	0.147±0.102	0.028±0.002	-
	TR	1.530±0.177	0.149±0.056	0.286±0.096	0.035±0.003	5.09±0.06
	GS	0.734±0.030	0.159±0.035	0.397±0.054	0.030±0.001	78.18±0.84
	HTW	0.439±0.040	**0.069±0.041**	0.217±0.060	0.032±0.002	7.37±0.75
	TabWak	**0.239±0.030**	0.070±0.036	0.131±0.132	**0.027±0.001**	-2.06±0.88
	TabWak$^\top$	0.295±0.045	0.071±0.035	0.114±0.041	0.028±0.002	21.26±1.13
	`TimeWak`	0.240±0.009	0.076±0.050	**0.111±0.074**	0.028±0.001	**151.03±1.60**
Weather	W/O	0.647±0.079	1.429±0.089	0.175±0.011	0.002±0.000	-
	TR	3.381±0.577	2.518±0.039	0.388±0.020	**0.002±0.000**	0.40±0.02
	GS	4.495±0.804	2.194±0.140	0.446±0.008	**0.002±0.000**	15.36±0.92
	HTW	0.712±0.062	1.463±0.098	0.190±0.013	**0.002±0.000**	4.82±0.83
	TabWak	0.751±0.088	1.571±0.168	0.200±0.007	**0.002±0.000**	-1.23±1.00
	TabWak$^\top$	**0.588±0.081**	1.369±0.089	**0.178±0.012**	**0.002±0.000**	39.58±0.89
	`TimeWak`	0.717±0.071	**0.951±0.088**	0.184±0.007	**0.002±0.000**	**205.53±1.86**

E.8 TPR@0.1%FPR performance

In Figure 6, we present the TPR@0.1%FPR metric against the number of samples across five datasets under 24, 64 and 128 window sizes. In most cases, `TimeWak` consistently outperforms other baselines, such as Gaussian Shading and TabWak$^\top$, by achieving significantly higher TPR values. Notably, `TimeWak` reaches a perfect 1.0 TPR@0.1%FPR in the majority of scenarios, with 7 cases requiring only a single sample and 4 cases needing just 2 samples, demonstrating its strong detectability with minimal data requirements.

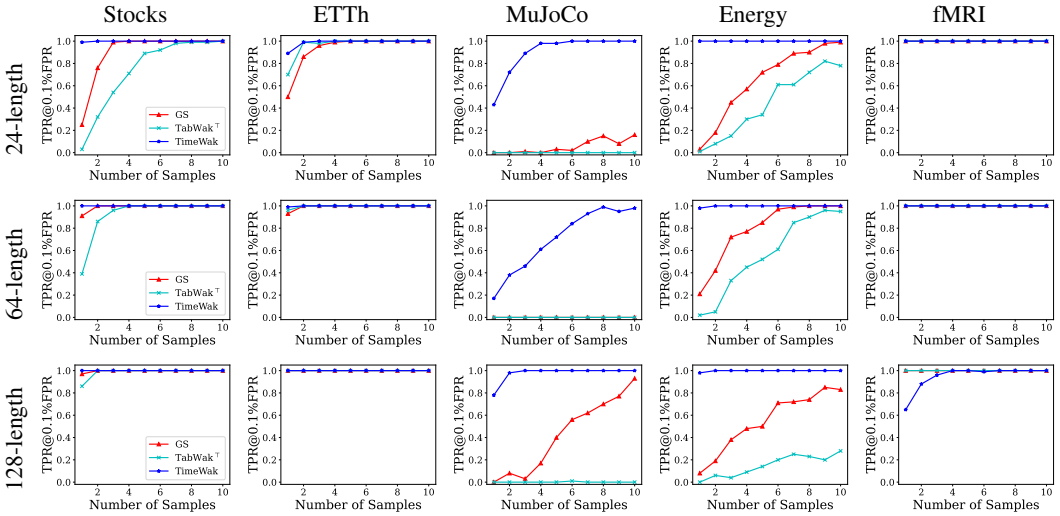


Figure 6: TPR@0.1%FPR against the number of samples across five datasets under different window sizes.

E.9 TPR@0.1%FPR on mixed dataset

We construct a mixed dataset containing equal proportions ($1/3$ each) of real data, synthetic data without watermarks, and synthetic data with watermarks, totaling 100 trials with a window length of 64. We evaluate TPR@0.1%FPR with 1, 10, and 20 samples per record, as shown in Table 16. For comparison, we select the methods with the best detectability: GS and TabWak$^\top$. The results demonstrate that `TimeWak` achieves 99 to 100% true positive rates in this mixed data setting when the number of samples is 20. But GS completely fails to detect watermarks in the MuJoCo dataset with 0% TPR across all sample sizes, while TabWak$^\top$ fails on both MuJoCo with 0% TPR and Energy datasets with 0 to 1% TPR.

Table 16: Results of TPR@0.1%FPR on mixed dataset when the number of samples is 1, 10, and 20.

Dataset	Method	1 ↑	10 ↑	20 ↑
Stocks	GS	**0.33**	0.87	**0.99**
	TabWak$^\top$	0.13	0.47	0.68
	`TimeWak`	0.22	**0.96**	**0.99**
ETTh	GS	0.43	**0.98**	**1.0**
	TabWak$^\top$	**0.49**	0.91	0.99
	`TimeWak`	0.38	0.92	**1.0**
MuJoCo	GS	0.0	0.0	0.0
	TabWak$^\top$	0.0	0.0	0.0
	`TimeWak`	**0.34**	**0.84**	**0.99**
Energy	GS	**0.42**	**1.0**	**1.0**
	TabWak$^\top$	0.01	0.0	0.0
	`TimeWak`	0.32	0.97	**1.0**
fMRI	GS	**0.4**	0.98	**1.0**
	TabWak$^\top$	0.31	**1.0**	**1.0**
	`TimeWak`	0.26	0.97	**1.0**

E.10 Preservation of key signal characteristics in watermarked time series data

To assess the preservation of key signal characteristics, we add 4 additional metrics from TSG-Bench [2]: Marginal Distribution Difference (MDD), AutoCorrelation Difference (ACD), Skewness Difference (SD), and Kurtosis Difference (KD). These measures are designed to capture inter-series correlations and temporal dependencies, thereby evaluating how well the generated time series preserves the original characteristics. For all these metrics, the lower the score, the better. As shown in Table 17, `TimeWak` consistently achieves scores very close to the non-watermarked (W/O) baseline across all datasets, indicating minimal distortion introduced by the watermark.

Table 17: Results of Marginal Distribution Difference (MDD), AutoCorrelation Difference (ACD), Skewness Difference (SD), and Kurtosis Difference (KD) for 64-length sequences.

Dataset	Method	MDD ↓	ACD ↓	SD ↓	KD ↓
Stocks	W/O	0.491	0.044	0.473	1.522
	TR	0.881	0.445	0.759	4.408
	GS	1.063	0.444	0.727	4.889
	HTW	0.491	**0.044**	0.473	1.522
	TabWak	0.470	0.078	0.180	0.544
	TabWak$^\top$	0.478	0.098	**0.060**	0.852
	`TimeWak`	**0.431**	0.068	0.103	**0.295**
ETTh	W/O	0.176	0.421	0.255	1.359
	TR	0.546	1.128	0.633	3.474
	GS	0.712	1.581	0.723	2.725
	HTW	0.384	**0.459**	0.276	1.468
	TabWak	0.211	0.532	0.245	1.422
	TabWak$^\top$	**0.183**	0.753	**0.141**	**0.707**
	`TimeWak`	0.184	0.511	0.197	1.109
MuJoCo	W/O	0.379	0.225	0.062	0.306
	TR	1.296	1.494	0.426	1.555
	GS	1.500	1.550	0.533	1.221
	HTW	0.941	0.434	0.076	**0.287**
	TabWak	0.420	**0.262**	0.087	0.311
	TabWak$^\top$	0.521	0.532	0.084	0.351
	`TimeWak`	**0.394**	0.300	**0.069**	0.340
Energy	W/O	0.188	0.157	0.112	0.703
	TR	0.558	0.606	0.355	1.681
	GS	0.660	0.958	0.373	1.015
	HTW	0.363	**0.148**	**0.108**	0.701
	TabWak	0.235	0.270	0.126	0.618
	TabWak$^\top$	0.263	0.336	0.112	0.682
	`TimeWak`	**0.215**	0.241	0.111	**0.601**
fMRI	W/O	0.099	0.153	0.041	0.128
	TR	0.447	1.356	0.066	0.428
	GS	0.818	1.435	0.096	0.174
	HTW	0.452	0.151	0.048	0.133
	TabWak	0.126	0.159	0.043	0.138
	TabWak$^\top$	0.127	0.261	**0.040**	**0.114**
	`TimeWak`	**0.120**	**0.148**	**0.040**	0.132

E.11 Hyperparameter evaluation

E.11.1 Intervals

Interval, also referred to as H, is one of the key hyperparameters in our approach. Based on our experiments in Table 18 (24-length), Table 19 (64-length), and Table 20 (128-length), we found that setting $H = 2$ yields the best results across most datasets. For instance, consider the Stocks dataset, which consists of 6 features and 24 time steps. When $H = 8$, the number of bit templates that can be generated is $2^{(3\times6)}$, whereas for $H = 2$, the number of bit templates increases significantly to $2^{(12\times6)}$. A lower H value allows for the generation of a greater number of bit combinations, leading to a more diverse seed distribution. However, as shown in Table 20, for datasets such as fMRI, tuning H can enhance detectability while preserving the quality of the synthetic data. This could be attributed to the inherently noisy nature of the fMRI dataset, where adjusting H helps balance detectability and data fidelity.

Table 18: Results of synthetic time series quality and watermark detectability with different intervals on `TimeWak`. Quality metrics and Z-score are for 24-length sequences.

Dataset	Interval	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.277±0.019	0.020±0.018	0.120±0.039	0.038±0.000	182.10±0.73
	4	0.419±0.098	0.006±0.002	0.162±0.022	0.039±0.000	202.45±1.11
	8	1.006±0.085	0.023±0.018	0.197±0.015	0.041±0.000	216.56±1.43
ETTh	2	0.237±0.017	0.212±0.043	0.102±0.014	0.122±0.002	134.83±0.95
	4	0.463±0.094	0.435±0.047	0.113±0.018	0.130±0.001	166.50±1.27
	8	0.925±0.140	0.576±0.082	0.150±0.014	0.135±0.005	167.54±1.36
MuJoCo	2	0.089±0.017	0.532±0.137	0.044±0.021	0.008±0.001	85.69±1.08
	4	0.148±0.031	0.739±0.086	0.087±0.020	0.007±0.000	71.95±1.36
	8	0.327±0.059	1.179±0.168	0.177±0.019	0.009±0.002	76.90±1.29
Energy	2	0.121±0.016	1.977±0.750	0.142±0.008	0.254±0.000	231.28±1.45
	4	0.186±0.017	3.315±0.395	0.159±0.011	0.254±0.000	266.77±1.99
	8	0.363±0.093	5.402±0.499	0.184±0.024	0.255±0.000	279.40±1.90
fMRI	2	0.199±0.010	2.006±0.053	0.122±0.033	0.100±0.000	379.51±0.82
	4	0.191±0.013	2.117±0.124	0.125±0.034	0.101±0.000	464.70±0.92
	8	0.204±0.021	2.354±0.110	0.171±0.043	0.103±0.000	506.95±1.10

Table 19: Results of synthetic time series quality and watermark detectability with different intervals on `TimeWak`. Quality metrics and Z-score are for 64-length sequences.

Dataset	Interval	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.387±0.054	0.017±0.017	0.092±0.041	0.037±0.000	395.34±1.24
	4	0.466±0.099	0.021±0.006	0.077±0.025	0.037±0.000	397.68±1.31
	8	1.053±0.099	0.017±0.013	0.155±0.037	0.037±0.000	406.56±1.77
ETTh	2	0.297±0.038	0.133±0.040	0.097±0.015	0.115±0.003	236.08±1.63
	4	0.514±0.027	0.335±0.040	0.098±0.031	0.119±0.008	272.70±1.79
	8	0.911±0.048	0.540±0.030	0.147±0.038	0.123±0.008	268.14±2.00
MuJoCo	2	0.108±0.014	0.413±0.062	0.038±0.021	0.007±0.001	56.45±1.26
	4	0.205±0.024	0.522±0.024	0.088±0.031	0.007±0.002	58.75±1.18
	8	0.338±0.068	0.768±0.077	0.159±0.020	0.007±0.001	68.29±1.19
Energy	2	0.143±0.019	1.662±0.298	0.145±0.019	0.251±0.000	267.53±2.60
	4	0.195±0.017	2.760±0.504	0.150±0.011	0.252±0.000	289.07±2.77
	8	0.407±0.087	4.285±0.229	0.170±0.025	0.252±0.000	345.97±3.48
fMRI	2	0.441±0.035	1.786±0.043	0.314±0.041	0.100±0.000	595.68±1.03
	4	0.425±0.027	1.834±0.122	0.273±0.076	0.100±0.000	749.10±1.08
	8	0.469±0.027	1.823±0.065	0.294±0.141	0.100±0.000	817.23±1.20

Table 20: Results of synthetic time series quality and watermark detectability with different intervals on `TimeWak`. Quality metrics and Z-score are for 128-length sequences.

Dataset	Interval	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.316±0.044	0.021±0.024	0.140±0.029	0.037±0.000	550.05±1.18
	4	0.410±0.104	0.019±0.017	0.140±0.067	0.037±0.000	571.54±1.22
	8	1.132±0.551	0.035±0.021	0.217±0.019	0.038±0.000	599.41±1.39
ETTh	2	1.090±0.100	0.135±0.057	0.174±0.007	0.110±0.009	340.36±2.06
	4	1.445±0.119	0.333±0.034	0.173±0.026	0.114±0.003	374.69±2.39
	8	1.838±0.099	0.497±0.054	0.173±0.021	0.116±0.003	392.83±2.39
MuJoCo	2	0.155±0.016	0.316±0.022	0.046±0.030	0.005±0.001	123.36±1.43
	4	0.172±0.038	0.410±0.057	0.083±0.011	0.006±0.000	178.16±1.60
	8	0.330±0.077	0.685±0.055	0.124±0.026	0.006±0.002	170.77±1.93
Energy	2	0.148±0.027	1.687±0.328	0.140±0.057	0.249±0.000	245.37±2.88
	4	0.261±0.025	3.246±0.345	0.114±0.030	0.250±0.001	354.38±2.32
	8	0.506±0.114	4.550±0.745	0.147±0.009	0.251±0.000	395.20±2.91
fMRI	2	0.855±0.072	1.704±0.060	0.298±0.227	0.100±0.000	526.81±13.12
	4	0.884±0.124	1.708±0.050	0.348±0.201	0.100±0.000	1022.29±1.75
	8	0.866±0.075	1.698±0.050	0.345±0.195	0.100±0.000	1097.78±1.57

E.11.2 Bits

We perform an empirical validation of the expected bit accuracy using simulations. Specifically, we set the synthetic time series data size as 24 time steps (window length) and 10 features, and evaluate the watermarking and detection pipeline using `TimeWak`. In this experiment, we intentionally omit both the forward diffusion and reverse (inversion) diffusion processes to focus solely on the effect of noise during reconstruction. Instead, we simulate the reconstruction noise directly by adding noise to the clean initial noise.

We generate initial samples using our watermarking method and simulate reconstruction error by adding noise with feature-specific means sampled from $\mathcal{N}(0, 5)$ and a shared variance σ. This results in a noise distribution of $\mathcal{N}(\mu_f, \sigma)$ per feature, where $\mu_f \sim \mathcal{N}(0, 5)$. We then apply watermark detection to the perturbed samples and compute the average bit accuracy.

We run the simulation using 100,000 samples, grouped into trials of 2,000 samples each. This process is repeated across 50 independent rounds to compute the average bit accuracy. Figure 7 shows the results we get. We observe that a larger L leads to higher bit accuracy, indicating better detectability. In addition, we evaluate a "transposed" version of `TimeWak`, denoted as $\texttt{TimeWak}^\top$, where the chained hash is applied along the feature dimension instead of the time axis. We find that the bit accuracy of this variant remains close to 0.5 and is significantly lower than that of the original `TimeWak`. This simulation further validates the importance of applying the watermark along the time axis, rather than across features, to ensure reliable detection.

Additionally, we present the values of bit-length L used across different experiments in Table 21–23. For bit-lengths greater than 2, we applied the valid bit mechanism from [41]. In general, a larger L tends to improve watermark detectability. This is because, during bit accuracy calculation, a larger L places more emphasis on the tail bits, which are less likely to be affected by reconstruction errors or noise. However, increasing L also leads to lower sample quality, as it involves modifying more of the initial noise, making it deviate further from a standard Gaussian distribution. Our results show this trade-off holds across most scenarios, with the exception of the Stocks dataset.

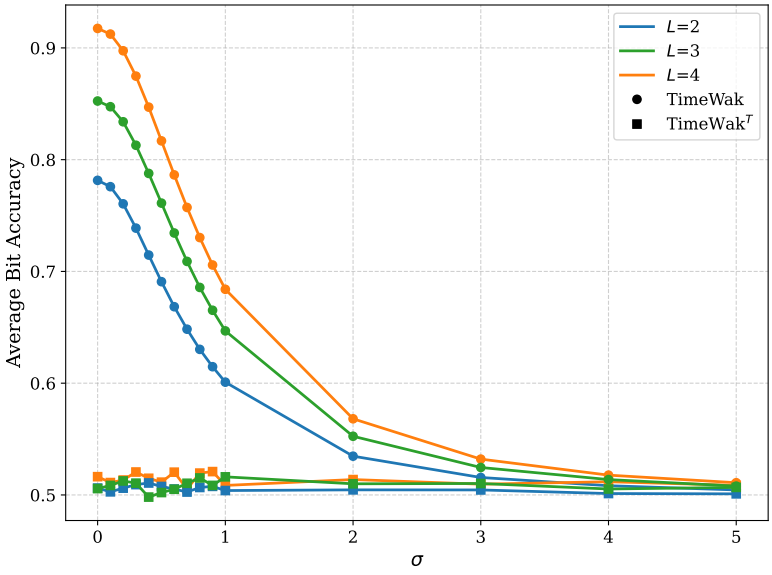


Figure 7: Average bit accuracy of different bit-length L.

Table 21: Results of synthetic time series quality and watermark detectability with different bits on `TimeWak`. Quality metrics and Z-score are for 24-length sequences.

Dataset	Bit	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.277±0.019	0.020±0.018	0.120±0.039	0.038±0.000	182.10±0.73
	3	0.214±0.039	0.024±0.019	0.130±0.033	0.038±0.000	194.87±0.56
	4	0.328±0.110	0.023±0.026	0.155±0.027	0.038±0.000	182.58±0.33
ETTh	2	0.237±0.017	0.212±0.043	0.102±0.014	0.122±0.002	134.83±0.95
	3	0.211±0.020	0.206±0.031	0.093±0.003	0.124±0.003	162.25±0.89
	4	0.238±0.025	0.225±0.043	0.095±0.016	0.124±0.001	149.74±0.66
MuJoCo	2	0.089±0.017	0.532±0.137	0.044±0.021	0.008±0.001	85.69±1.08
	3	0.092±0.022	0.520±0.105	0.054±0.014	0.008±0.001	73.09±1.29
	4	0.099±0.019	0.524±0.079	0.056±0.013	0.007±0.000	67.67±1.23
Energy	2	0.121±0.016	1.977±0.750	0.142±0.008	0.254±0.000	231.28±1.45
	3	0.121±0.014	1.799±0.395	0.156±0.023	0.254±0.001	268.24±1.69
	4	0.143±0.015	1.721±0.347	0.155±0.010	0.254±0.000	269.83±1.60
fMRI	2	0.199±0.010	2.006±0.053	0.122±0.033	0.100±0.000	379.51±0.82
	3	0.195±0.008	1.987±0.076	0.113±0.031	0.101±0.000	456.02±0.67
	4	0.183±0.012	2.032±0.030	0.111±0.026	0.101±0.000	440.88±0.55

Table 22: Results of synthetic time series quality and watermark detectability with different bits on `TimeWak`. Quality metrics and Z-score are for 64-length sequences.

Dataset	Bit	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.387±0.054	0.017±0.017	0.092±0.041	0.037±0.000	395.34±1.24
	3	0.312±0.046	0.014±0.006	0.121±0.010	0.037±0.000	334.15±0.49
	4	0.251±0.053	0.014±0.018	0.095±0.022	0.037±0.000	309.59±0.33
ETTh	2	0.297±0.038	0.133±0.040	0.097±0.015	0.115±0.003	236.08±1.63
	3	0.369±0.043	0.182±0.036	0.102±0.013	0.117±0.003	249.67±1.41
	4	0.365±0.031	0.185±0.030	0.102±0.010	0.113±0.007	261.13±1.20
MuJoCo	2	0.108±0.014	0.413±0.062	0.038±0.021	0.007±0.001	56.45±1.26
	3	0.136±0.012	0.423±0.051	0.073±0.018	0.007±0.002	84.07±1.48
	4	0.126±0.017	0.381±0.063	0.036±0.030	0.007±0.001	96.73±1.45
Energy	2	0.143±0.019	1.662±0.298	0.145±0.019	0.251±0.000	267.53±2.60
	3	0.182±0.047	1.284±0.400	0.165±0.019	0.251±0.000	323.04±2.37
	4	0.143±0.010	1.460±0.354	0.152±0.023	0.251±0.000	322.38±2.25
fMRI	2	0.441±0.035	1.786±0.043	0.314±0.041	0.100±0.000	595.68±1.03
	3	0.423±0.024	1.782±0.082	0.216±0.175	0.100±0.000	724.69±0.85
	4	0.440±0.027	1.783±0.033	0.256±0.106	0.100±0.000	712.67±0.59

Table 23: Results of synthetic time series quality and watermark detectability with different bits on `TimeWak`. Quality metrics and Z-score are for 128-length sequences.

Dataset	Bit	Context-FID ↓	Correlational ↓	Discriminative ↓	Predictive ↓	Z-score ↑
Stocks	2	0.316±0.044	0.021±0.024	0.140±0.029	0.037±0.000	550.05±1.18
	3	0.380±0.059	0.019±0.015	0.176±0.046	0.037±0.000	459.23±0.45
	4	0.391±0.087	0.017±0.020	0.134±0.060	0.037±0.000	427.53±0.23
ETTh	2	1.090±0.100	0.135±0.057	0.174±0.007	0.110±0.009	340.36±2.06
	3	1.111±0.137	0.151±0.040	0.153±0.010	0.118±0.005	352.26±1.63
	4	1.173±0.131	0.233±0.058	0.166±0.013	0.113±0.006	362.55±1.39
MuJoCo	2	0.155±0.016	0.316±0.022	0.046±0.030	0.005±0.001	123.36±1.43
	3	0.183±0.029	0.317±0.068	0.062±0.011	0.005±0.000	183.47±1.55
	4	0.150±0.013	0.349±0.028	0.051±0.031	0.006±0.001	174.45±1.55
Energy	2	0.148±0.027	1.687±0.328	0.140±0.057	0.249±0.000	245.37±2.88
	3	0.230±0.037	1.154±0.446	0.166±0.054	0.249±0.001	380.35±2.49
	4	0.167±0.014	1.700±0.524	0.168±0.069	0.249±0.001	389.26±2.01
fMRI	2	0.855±0.072	1.704±0.060	0.298±0.227	0.100±0.000	526.81±13.12
	3	0.819±0.010	1.688±0.049	0.374±0.193	0.100±0.000	986.17±1.08
	4	0.828±0.053	1.713±0.020	0.336±0.209	0.100±0.000	967.53±0.86

E.12 Watermark detection overhead

To evaluate the practical feasibility of real-time deployment, we measure the watermark detection overhead for `TimeWak` using a single NVIDIA L40S GPU and Intel(R) Xeon(R) Platinum 8562Y+ CPU. Table 24 presents the computational overhead for watermark detection across different datasets and configurations. The results demonstrate that detection overhead remains consistently low, ranging from approximately 1.5 to 7.5 seconds depending on the dataset complexity and batch size. We consider this overhead acceptable for streaming scenarios, particularly given the security benefits provided by the watermarking system. Adapting the hashing mechanism to handle variable-length sequences without padding represents a promising direction for future research that could further enhance the method's applicability to diverse real-world scenarios.

Table 24: Watermark detection overhead in seconds for `TimeWak` when the batch size is 1 and 100.

Dataset	Window Size	1	100
Stocks	24	1.58	2.09
	64	1.50	2.27
	128	1.55	2.59
ETTh	24	1.72	2.23
	64	1.64	2.41
	128	1.63	2.62
MuJoCo	24	2.96	3.84
	64	2.90	4.25
	128	2.90	4.49
Energy	24	3.89	5.22
	64	3.88	5.73
	128	3.89	6.58
fMRI	24	4.69	6.31
	64	4.75	7.00
	128	4.81	7.47

F Synthetic samples

Figures 8–12 show synthetic time series generated unconditionally by Diffusion-TS with/without watermark embedding. Each figure corresponds to one of the following datasets: Stocks, ETTh, MuJoCo, Energy, and fMRI. Within each figure, the columns represent the following algorithms: no watermark, `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarks. Up to 4 features are randomly selected from each dataset.

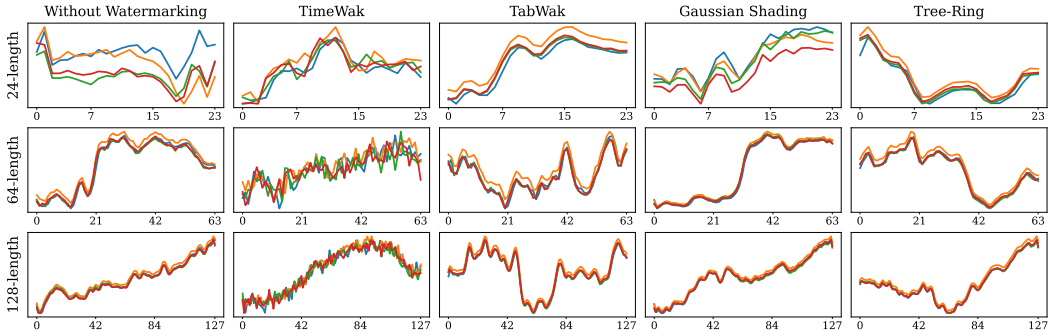


Figure 8: Non-watermarked (leftmost column) and watermarked (remaining columns) time series generated by `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarking for the Stocks dataset.

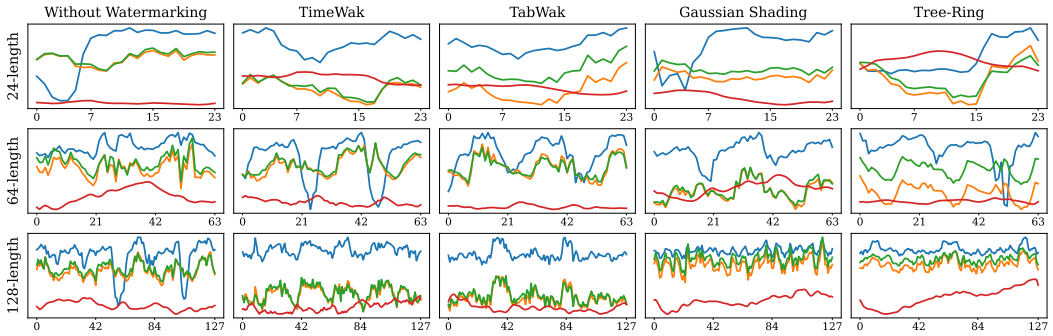


Figure 9: Non-watermarked (leftmost column) and watermarked (remaining columns) time series generated by `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarking for the ETTh dataset.

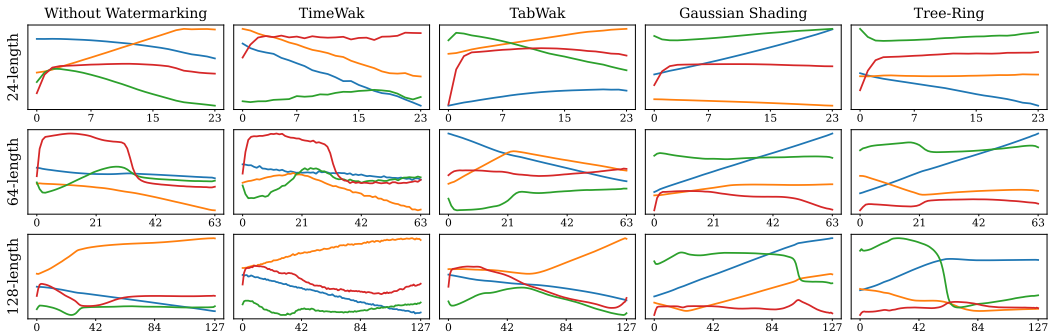


Figure 10: Non-watermarked (leftmost column) and watermarked (remaining columns) time series generated by `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarking for the MuJoCo dataset.

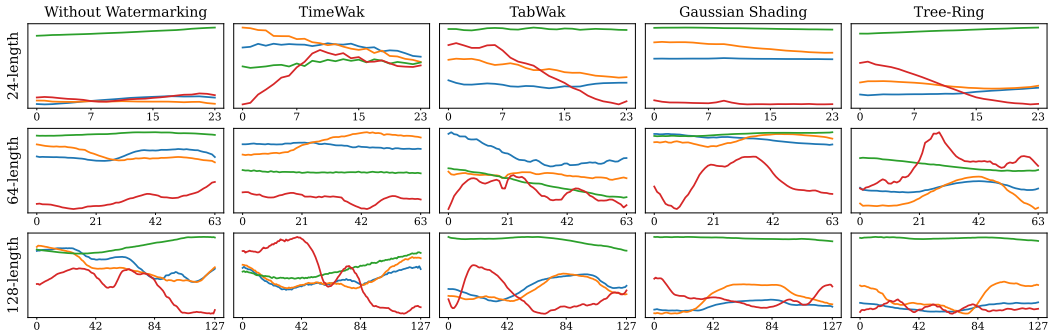


Figure 11: Non-watermarked (leftmost column) and watermarked (remaining columns) time series generated by `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarking for the Energy dataset.

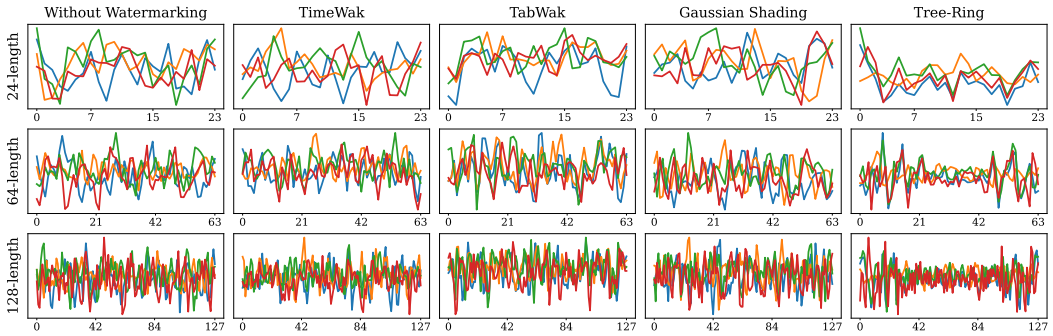


Figure 12: Non-watermarked (leftmost column) and watermarked (remaining columns) time series generated by `TimeWak`, TabWak, Gaussian Shading, and Tree-Ring watermarking for the fMRI dataset.