Time Series Diffusion in the Frequency Domain

Jonathan Crabbé [*1] Nicolas Huynh [*1] Jan Stanczuk [1] Mihaela van der Schaar [1]

Abstract

Fourier analysis has been an instrumental tool in the development of signal processing. This leads us to wonder whether this framework could similarly benefit generative modelling. In this paper, we explore this question through the scope of time series diffusion models. More specifically, we analyze whether representing time series in the frequency domain is a useful inductive bias for score-based diffusion models. By starting from the canonical SDE formulation of diffusion in the time domain, we show that a dual diffusion process occurs in the frequency domain with an important nuance: Brownian motions are replaced by what we call mirrored Brownian motions, characterized by mirror symmetries among their components. Building on this insight, we show how to adapt the denoising score matching approach to implement diffusion models in the frequency domain. This results in frequency diffusion models, which we compare to canonical time diffusion models. Our empirical evaluation on real-world datasets, covering various domains like healthcare and finance, shows that frequency diffusion models better capture the training distribution than time diffusion models. We explain this observation by showing that time series from these datasets tend to be more localized in the frequency domain than in the time domain, which makes them easier to model in the former case. All our observations point towards impactful synergies between Fourier analysis and diffusion models.

1. Introduction

Deep generative modelling leverages the inductive bias of neural networks to learn complex, high-dimensional probability distributions from real-world datasets. Among other applications, generative models allow for generation of new synthetic samples consistent with the distribution of the training data, yet distinct from the actual data encountered during training. Recently this field has seen tremendous progress in various modalities including image (Karras et al., 2020; Dhariwal & Nichol, 2021), audio (Kong et al., 2021; Donahue et al., 2018), video (Rombach et al., 2022) and text (Dieleman et al., 2022) generation, as well as addressing inverse problems such as in-painting (Lugmayr et al., 2022) or super-resolution (Saharia et al., 2022). Moreover deep generative models have started showing significant potential in contributing to natural sciences, though protein design (Watson et al., 2023), drug development (Xu et al., 2022) and material synthesis (Zeni et al., 2023). However, the application of these models to time series data has not seen the same level of advancement (Gatta et al., 2022). Some notable examples of time series generative models include TimeGAN (Yoon et al., 2019), FourierFlow (Alaa et al., 2021), and RCGAN (Esteban et al., 2017), yet this area remains less explored compared to other applications. While research on generative modeling for time series has not progressed as quicky as in the static setting, it is an equally important problem. For example, generative modelling for time series is a promising avenue to reconciliate *privacy* with the development of machine learning models, notably in high-stakes domains such as healthcare, where access to time series data is subject to strong regulations by medical institutions (Miller & Tucker, 2009). Another example is generating time series for *data augmentation*, in order to increase the dataset size for some downstream tasks or to address imbalance problems (Nikolaidis et al., 2019).

Diffusion Models. In recent years, diffusion models (Hyvärinen & Dayan, 2005; Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) have emerged as one of the most promising research avenues in deep generative modelling, achieving state-of-the art results across many generative modelling tasks (Dhariwal & Nichol, 2021; Saharia et al., 2022). Diffusion models have been applied to time series modelling, achieving promising results (Lin et al., 2023). However, there is substantial room for development and refinement in these early-stage applications.

Fourier Analysis. Fourier analysis is a remarkably powerful tool in signal processing, compression and machine learning (Körner, 2022). It has been shown to significantly

[*]Equal contribution [1]DAMTP, University of Cambridge. Correspondence to: Jonathan Crabbé <jc2133@cam.ac.uk>, Nicolas Huynh <nvth2@cam.ac.uk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

improve state-of-the-art the performance of many deep learning based time series analysis techniques (Yi et al., 2023), with some recent applications in dataset distillation (Shin et al., 2023). In the context of deep generative models, this is exemplified in (Alaa et al., 2021), where the application of normalizing flows to Fourier representations yielded promising results. More recently, some work by (Phillips et al., 2022) has been done on diffusion on functional spaces, which include Fourier representations of signals although the paper does not specifically focus on the Fourier basis.

Motivation. Despite Fourier analysis' widespread success, its application to diffusion models for time series remains largely unexplored. This paper seeks to fill this research gap, by examining whether spectral representations can improve diffusion models for time series modelling. Our focus is not on achieving state-of-the-art results, but rather investigating whether representing time series in the frequency domain is a useful inductive bias for diffusion models.

Our contributions. (1) Formalizing frequency diffusion. In Section 3, we show theoretically how to translate SDE-based diffusion of time series to the frequency domain. We demonstrate that the denoising score matching recipe can be adapted by replacing standard Brownian motions by what we call mirrored Brownian motions, characterized by mirror symmetries in their components. **(2) Comparing time and frequency diffusion.** In Section 4.1, we compare the ability of the time and frequency score models to generate samples that are faithful to the training sets by leveraging *sliced Wasserstein distances*. Through an extensive analysis on 6 real-world datasets illustrating fields like healthcare, finance, engineering and climate modelling, we demonstrate that frequency score models consistently outperform the time score models. **(3) Understanding why and when frequency diffusion is preferable.** In Section 4.2, we demonstrate that the signals in all 6 datasets concentrate most of their power spectrum on the low frequencies. We hypothesize that this localization in the frequency domain explains the superior performances of frequency diffusion models. In Section 4.3, we confirm this hypothesis by artificially delocalizing the spectral representation of real signals and showing that the gap between time and frequency diffusion closes.

2. Background

Notations. We consider multivariate time series of fixed size[1] $\mathbf{x} \in \mathbb{R}^{N \times M}$, where $N \in \mathbb{N}$ is the number of time steps and $M \in \mathbb{N}$ is the number of features tracked over time. Often, we will denote by $d_X = N \cdot M$ the total dimension of the time series $\mathbf{x}$. We shall use *Greek letters* for components of the time series. In this way, $\mathbf{x}_\tau \in \mathbb{R}^M$ denotes the feature vector at time $\tau \in [N]$ and $x_{\tau,\nu}$ denotes the value of feature $\nu \in [M]$ at time τ. We denote by $[K] := \{0, 1, \dots, K-1\}$ the integers between 0 (included) and $K \in \mathbb{N}$ (excluded). To avoid any confusion between time series steps and diffusion steps, we shall use *Latin letters* for the diffusion process. In this way, the diffusion process is described by a family of time series $\{\mathbf{x}(t) \in \mathbb{R}^{d_X}\}_{t=0}^T$ indexed by a continuous diffusion variable $t \in [0, T]$. Thanks to these notations, we unambiguously interpret $\mathbf{x}_\tau(t) \in \mathbb{R}^M$ as the feature vector at time step $\tau \in [N]$ and at diffusion step $t \in [0, T]$. We shall detail below how this diffusion process is defined.

[1]Padding over time can be used in cases where the datasets contain time series of different lengths.

2.1. Score-based Generative Modeling with SDEs

In continuous-time diffusion modelling, one assumes access to samples drawn from an unknown density p_{data}. The objective of generative modelling is to obtain a tractable approximation of this distribution.

Forward Diffusion. Score-based generative modeling with *stochastic differential equation* (SDEs) (Song et al., 2020) typically operates by first constructing a forward diffusion process. In the case of time series, forward continuous diffusion is described by the following SDE, with $t \in [0, T]$:

$$\mathrm{d}\mathbf{x} = \boldsymbol{f}(\mathbf{x}, t)\mathrm{d}t + \boldsymbol{G}(t)\mathrm{d}\boldsymbol{w}, \tag{1}$$

where $\boldsymbol{f} : \mathbb{R}^{d_X} \times [0, T] \to \mathbb{R}^{d_X}$ is the drift, $\boldsymbol{w}$ is a standard Brownian motion in $\mathbb{R}^{d_X}$, and $\boldsymbol{G} : [0, T] \to \mathbb{R}^{N \times N}$ is the diffusion matrix. We denote p_t the probability density of the solution $\mathbf{x}(t)$ of Equation (1) at time $t \in [0, T]$. With the slight abuse of notation from Song et al. (2020), we shall abbreviate $p_t(\mathbf{x}(t))$ by $p_t(\mathbf{x})$. Together with the SDE, we impose the initial condition $p_0 = p_{\mathrm{data}}$, which corresponds to samples initially drawn from the data density p_{data}. In practice, we consider $\boldsymbol{f}$ and $\boldsymbol{G}$ such that p_{data} is transported to a final density p_T close to an isotropic Gaussian.

Reverse Diffusion. The reverse diffusion process performs the inverse transformation by transporting the isotropic Gaussian density p_T to the data density $p_0 = p_{\mathrm{data}}$. Hence, applying reverse diffusion to samples drawn from the isotropic Gaussian permits to sample from the unknown density p_{data}. It was shown by Anderson (1982) that this reverse diffusion satisfies the following SDE:

$$\mathrm{d}\mathbf{x} = \boldsymbol{b}(\mathbf{x}, t)\mathrm{d}t + \boldsymbol{G}(t)\mathrm{d}\hat{\boldsymbol{w}}, \tag{2}$$

where $\boldsymbol{b}(\mathbf{x}, t) = \boldsymbol{f}(\mathbf{x}, t) - \boldsymbol{G}(t)\boldsymbol{G}(t)^T \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, $\mathrm{d}t$ is a negative infinitesimal time step, and $\hat{\boldsymbol{w}}$ is a Brownian time increment with time going backwards from T to 0.

Denoising Score Matching. In order to run the reverse diffusion process, one needs access to the score $\mathbf{s}(\mathbf{x}, t) :=$

$\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$. In practice, the ground-truth density p_t is unknown. Denoising score matching circumvents this problem by estimating the ground-truth score with a function $\mathbf{s}_{\theta^*}$ whose parameters θ^* *minimize* the following score matching objective computed from the data samples (Hyvärinen & Dayan, 2005; Song & Ermon, 2019):

$$\theta^* = \arg\min_{\theta \in \Theta} \mathbb{E}_{t,\mathbf{x}(0),\mathbf{x}(t)} \left[\mathcal{L}_{\mathrm{SM}}(\mathbf{s}_\theta, \mathbf{s}_{t|0}, \mathbf{x}, t)\right] \tag{3}$$

$$\mathcal{L}_{\mathrm{SM}}(\mathbf{s}_\theta, \mathbf{s}_{t|0}, \mathbf{x}, t) := \|\mathbf{s}_\theta(\mathbf{x}, t) - \mathbf{s}_{t|0}(\mathbf{x}, t)\|^2 \tag{4}$$

where $\|\cdot\|$ denotes the Frobenius norm, $t \sim \mathcal{U}(0, T)$, $\mathbf{x}(0) \sim p_0(\mathbf{x})$, $\mathbf{x}(t) \sim p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0))$ with $p_{t|0}$ denoting the transition kernel from 0 to t, and $\mathbf{s}_{t|0}(\mathbf{x}, t) := \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0))$. With sufficient model capacity, the parameters θ^* provide an approximation $\mathbf{s}_{\theta^*}$ that is equal to the score $\mathbf{s}$ for almost all $\mathbf{x}$ and t in the large data limit (Vincent, 2011). Equipped with an approximation of the score $\mathbf{s}_{\theta^*} \approx \mathbf{s}$, one can generate data by sampling according to the solution defined by the reverse diffusion process from Equation (2).

2.2. Discrete Fourier Transform

DFT. By considering a time series $\mathbf{x} = (\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}) \in \mathbb{R}^{d_X}$, the *Discrete Fourier Transform* (DFT), denoted as $\tilde{\mathbf{x}} = \mathcal{F}[\mathbf{x}]$, is defined as

$$\tilde{\mathbf{x}}_\kappa := \frac{1}{\sqrt{N}} \sum_{\tau=0}^{N-1} \mathbf{x}_\tau \exp\left(-\frac{\kappa 2\pi i}{N}\tau\right) \tag{5}$$

for all $\kappa \in [N]$. In the signal processing literature, each κ corresponds to a harmonic of frequency $\omega_\kappa := \frac{\kappa 2\pi}{N}$. For this reason, the DFT $\tilde{\mathbf{x}}$ is said to represent the time series $\mathbf{x}$ in the *frequency domain*, as opposed to the *time domain*. We also note that the DFT is complex-valued ($\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$).

Matrix Representation. We note that the DFT operator $\mathcal{F}$ is *linear* with respect to the time components $(\mathbf{x}_0, \ldots, \mathbf{x}_{N-1})$. It can therefore be expressed through a left matrix multiplication $\tilde{\mathbf{x}} = \mathcal{F}[\mathbf{x}] = U\mathbf{x}$, where $U \in \mathbb{C}^{N \times N}$ is defined as $[U]_{\kappa\tau} := N^{-1/2} \exp(-i\omega_\kappa \tau)$. It can easily be checked (see Appendix A.1) that the matrix U is unitary: $U^*U = UU^* = I_N$, where U^* is the conjugate transpose of U and I_N is the $N \times N$ identity matrix. This implies that the DFT operator is invertible and that the original time series can be reconstructed from its representation in the frequency domain: $\mathbf{x} = \mathcal{F}^{-1}[\tilde{\mathbf{x}}] := U^*\tilde{\mathbf{x}}$.

DFT of a Real-Valued Sequence. While the DFT $\tilde{\mathbf{x}}$ is defined in $\mathbb{C}^{d_X}$, some of its components are made redundant by the fact that $\mathbf{x}$ is a real-valued time-series. One can easily check (see Appendix A.1) that this constraint imposes the following *mirror symmetry* on the DFT for all $\kappa \in [N]$:

$$\tilde{\mathbf{x}}_\kappa = \tilde{\mathbf{x}}^*_{N-\kappa}, \tag{6}$$

where z^* denotes the complex conjugate of $z \in \mathbb{C}$ and we define $\tilde{\mathbf{x}}_N := \tilde{\mathbf{x}}_0$ for consistency. Through this symmetry, we observe that the components $\tilde{\mathbf{x}}_\kappa$ with $\kappa \leq \lfloor N/2 \rfloor$ uniquely define the DFT of a real-valued time series. For this reason, the frequencies beyond the Nyquist frequency $\omega_{\mathrm{Nyq}} := \omega_{\lfloor N/2 \rfloor}$ are redundant with respect to the lower frequencies. In the frequency domain, one then needs only to diffuse N real numbers extracted from the DFT and the rest of $\tilde{\mathbf{x}}$ can be deduced from Equation (6).

Signal Energy. An important quantity related to a time series $\mathbf{x}$ is its *total energy*, which simply corresponds to the squared Frobenius norm $\|\mathbf{x}\|^2 := \sum_{\tau=0}^{N-1} \sum_{\nu=1}^{M} |x_{\tau,\nu}|^2$, where $|\cdot|$ denotes the modulus of a complex number. Through *Parseval's theorem*, this energy can be evaluated by computing the same norm for the DFT $\tilde{\mathbf{x}}$ of $\mathbf{x}$: $\|\mathbf{x}\|^2 = \|\tilde{\mathbf{x}}\|^2$. We note that the total energy is obtained by summing over all time steps or frequencies. To characterize how the energy is distributed over the time steps $\tau \in [N]$, we use the *energy density* defined as the squared Euclidean norm $\|\mathbf{x}_\tau\|_2^2 := \sum_{\nu=1}^{M} |x_{\tau,\nu}|^2$. Similarly, the *spectral energy density* defined as $\|\tilde{\mathbf{x}}_\kappa\|_2^2$ describes how the signal energy is distributed across the frequencies $\kappa \in [N]$.

Probability Density in the Complex Space. Adapting the diffusion formalism to the frequency domain requires to define a probability density for the complex-valued random variable $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$. By following (Schreier & Scharf, 2010), this density is written in terms of the real and imaginary parts of the signal $\tilde{p}(\tilde{\mathbf{x}}) := \tilde{p}(\Re[\tilde{\mathbf{x}}], \Im[\tilde{\mathbf{x}}])$. Similarly, the score function follows a similar decomposition in terms of the signal real and imaginary parts $\tilde{\mathbf{s}}(\tilde{\mathbf{x}}) := \nabla_{\Re[\tilde{\mathbf{x}}]} \log \tilde{p}(\tilde{\mathbf{x}}) + i \cdot \nabla_{\Im[\tilde{\mathbf{x}}]} \log \tilde{p}(\tilde{\mathbf{x}})$. We note that the gradient involved in the definition of the scores is non-trivial when the constraint in Equation (6) is enforced. In Appendix A.2, we establish a formal definition in this setting by interpreting the complex signals fulfilling this constraint as a submanifold in $\mathbb{C}^{d_X}$. This constraint implies that the score components follow an analogous mirror symmetry: $\tilde{\mathbf{s}}_\kappa = \tilde{\mathbf{s}}^*_{N-\kappa}$ for all $\kappa \in [N]$. In the following, we shall implicitly rely on this definition.

3. Diffusing in the Frequency Domain

In the previous section, we have described how the typical diffusion formalism applies to time-series. We have also described how the DFT $\tilde{\mathbf{x}} = \mathcal{F}[\mathbf{x}]$ offers a full description of the time series $\mathbf{x}$ in the frequency domain. The first step is to define how time-based diffusion translates in the frequency domain. Note that this is non-trivial as the DFT are complex-valued $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$ signals. To solve this, we shall assume that the stochastic process in the time domain $\{\mathbf{x}(t)\}_{t=0}^{T}$, written compactly as $\mathbf{x}$, follows the diffusion process described in Equation (1). By leveraging the matrix formulation of the DFT $\tilde{\mathbf{x}} = U\mathbf{x}$, we will now derive diffusion SDEs in the frequency domain.

3.1. Diffusion SDEs

In order to derive the diffusion SDEs in the frequency domain, we shall simply apply the DFT operator to the forward diffusion SDE in the time domain from Equation (1). We note that this equation contains a standard Brownian motion $\boldsymbol{w}$. In the below lemma, we describe the DFT of $\boldsymbol{w}$ and show that it contains two copies of a non-standard Brownian motion related by the constraint from Equation (6). We refer to this as a *mirrored Brownian motion*.

Lemma 3.1. *(DFT of standard Brownian motion). Let $\boldsymbol{w}$ be a standard Brownian motion on $\mathbb{R}^{d_X}$ with $d_X = N \cdot M$, where $N \in \mathbb{N}^+$ is the number of time series steps and $M \in \mathbb{N}^+$ is the number of features tracked over time. Then $\boldsymbol{v} = U\boldsymbol{w}$ is a continuous stochastic process endowed with:*
(1) Mirror Symmetry. *For all $\kappa \in [N]$, $\boldsymbol{v}_\kappa = \boldsymbol{v}^*_{N-\kappa}$.*
(2) Real Brownian Motion. *$\boldsymbol{v}_0$ is a (real) standard Brownian motion on $\mathbb{R}^M$.*
(3) Complex Brownian Motions. *For all κ with $1 \leq \kappa \leq \lfloor N/2 \rfloor$, we can write $\boldsymbol{v}_\kappa = (\tilde{\boldsymbol{w}}^1_\kappa + i\tilde{\boldsymbol{w}}^2_\kappa)/\sqrt{2}$ where $\tilde{\boldsymbol{w}}^1_\kappa$ and $\tilde{\boldsymbol{w}}^2_\kappa$ are independent standard Brownian motions on $\mathbb{R}^M$, except when N is even and $\kappa = N/2$, where $\boldsymbol{v}_{N/2}$ is a real standard Brownian motion on $\mathbb{R}^M$.*
(4) Independence. *The stochastic processes $\{\boldsymbol{v}_\kappa\}_{\kappa=0}^{\lfloor N/2 \rfloor}$ are mutually independent.*
We call any stochastic process satisfying the above constraints a mirrored Brownian motion *on $\mathbb{C}^{d_X}$.*

Proof. The proof is given in Appendix A.3. □

Remark 3.2. Note that $\boldsymbol{v}$ is not strictly speaking a Brownian motion, since it contains duplicate components due to the mirror symmetry. However, our theoretical analysis in Appendix A demonstrates that we can treat it as such by restricting to a subset of non-redundant components.

We now leverage Lemma 3.1 to show that $\tilde{\mathbf{x}}$ can be described by diffusion SDEs in the frequency domain which involve mirrored Brownian motions.

Proposition 3.3. *(Diffusion process in frequency domain). Let us assume that $\boldsymbol{x}$ is a diffusion process that is a solution of Equation (1), with $\boldsymbol{G}(t) = g(t)\, I_N$. Then $\tilde{\boldsymbol{x}} = \mathcal{F}[\boldsymbol{x}]$ is a solution to the forward diffusion process defined by:*

$$\mathrm{d}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t)\mathrm{d}t + g(t)\mathrm{d}\tilde{\boldsymbol{v}}, \tag{7}$$

where $\tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t) = U\boldsymbol{f}(U^*\tilde{\boldsymbol{x}}, t)$ and $\tilde{\boldsymbol{v}}$ is a mirrored Brownian motion on $\mathbb{C}^{d_X}$. The associated reverse diffusion process is defined by:

$$\mathrm{d}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}}, t)\mathrm{d}t + g(t)\mathrm{d}\breve{\boldsymbol{v}} \tag{8}$$

where $\tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}}, t) = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t) - g^2(t)\Lambda^2\tilde{\mathbf{s}}(\tilde{\boldsymbol{x}}, t)$, $\Lambda \in \mathbb{R}^{N\times N}$ is a diagonal matrix such that

$$[\Lambda]_{\kappa,\kappa} = \begin{cases} 1 & \text{if } \kappa = 0, \text{ or } N \text{ is even and } \kappa = N/2 \\ \frac{1}{\sqrt{2}} & \text{otherwise} \end{cases},$$

$\mathrm{d}t$ is a negative infinitesimal time step, and $\breve{\boldsymbol{v}}$ is a mirrored Brownian motion on $\mathbb{C}^{d_X}$ with time going from T to 0.

Proof. The proof is given in Appendix A.3. □

Proposition 3.3 gives us a recipe to implement diffusion in the frequency domain. It guarantees that the formalism introduced by Song et al. (2020) extends to this setting with one important difference: the Brownian motion must be replaced by a mirrored Brownian motion. Ignoring this prescription by taking $\tilde{\boldsymbol{v}}$ to be a Brownian motion on $\mathbb{C}^{d_X}$ could lead to unintended consequences, such as generating complex-valued time series $\mathbf{x} \in \mathbb{C}^{d_X} \setminus \mathbb{R}^{d_X}$.

3.2. Denoising Score Matching

The reverse diffusion process given in Equation (8) provides an explicit way to samples time-series in the frequency domain provided we can compute $\tilde{\boldsymbol{b}}(\tilde{\mathbf{x}}, t)$, which involves the unknown score $\tilde{\mathbf{s}}$. Like in the time domain, and motivated by Equation (8), we build an approximation of the score with a function $\tilde{\mathbf{s}}_{\tilde{\theta}^*}$, whose parameters $\tilde{\theta}^*$ minimize the score matching objective:

$$\tilde{\theta}^* = \arg\min_{\tilde{\theta}\in\Theta} \mathbb{E}_{t,\tilde{\mathbf{x}}(0),\tilde{\mathbf{x}}(t)}\left[\mathcal{L}_{\mathrm{SM}}\left(\tilde{\mathbf{s}}_{\tilde{\theta}}, \Lambda^2\tilde{\mathbf{s}}_{t|0}, \tilde{\mathbf{x}}, t\right)\right] \tag{9}$$

with $t \sim \mathcal{U}(0, T)$, $\tilde{\mathbf{x}}(0) \sim \tilde{p}_0(\tilde{\mathbf{x}})$, $\tilde{\mathbf{x}}(t) \sim \tilde{p}_{t|0}(\tilde{\mathbf{x}}(t)|\tilde{\mathbf{x}}(0))$ and Λ is the diagonal matrix defined in Proposition 3.3. In practice, this objective is evaluated by first obtaining frequency representations of time-series, and then sampling from $\tilde{p}_{t|0}$ using Equation (7). Having trained $\tilde{\mathbf{s}}_{\tilde{\theta}^*}$, the backward process and $\tilde{\mathbf{s}}_{\tilde{\theta}^*}$ permit to draw samples from $\tilde{p}_0$. It then suffices to apply the inverse DFT $\mathcal{F}^{-1}$ to map the resulting complex-valued signals back into the time domain.

One important question remains at this stage. How does training a score in the frequency domain allow to generate DFT of time series sampled from p_{data}? In other words, how does minimizing the score matching in Equation (9) imply that $\tilde{p}_0 \approx \tilde{p}_{\mathrm{data}}$? To answer this question, a key observation is that we can associate an auxiliary score $\mathbf{s}'_{\tilde{\theta}}$ in the time domain to the score $\tilde{\mathbf{s}}_{\tilde{\theta}}$ by applying an inverse DFT $\mathcal{F}^{-1}$. Below, we show that minimizing the score matching loss from Equation (9) for the score $\tilde{\mathbf{s}}_{\tilde{\theta}}$ is equivalent to minimizing the score matching loss from Equation (3) for the auxiliary score $\mathbf{s}'_{\tilde{\theta}}$. This important observation connects the reverse diffusion process in the frequency domain described by Equation (8) with a reverse diffusion process in the time domain following Equation (2).

Proposition 3.4. *(Score matching equivalence). Consider a score $\tilde{\mathbf{s}}_{\tilde{\theta}} : \mathbb{C}^{d_X} \times [0, T] \to \mathbb{C}^{d_X}$ defined in the frequency* *domain and satisfying the mirror symmetry* $[\tilde{\mathbf{s}}_{\tilde{\theta}}]_\kappa = [\tilde{\mathbf{s}}^*_{\tilde{\theta}}]_{N-\kappa}$ *for all* $\kappa \in [N]$. *Let us define an auxiliary score* $\mathbf{s}'_{\tilde{\theta}} : \mathbb{R}^{dx} \times [0, T] \rightarrow \mathbb{R}^{dx}$ *as* $(\boldsymbol{x}, t) \mapsto \mathbf{s}'_{\tilde{\theta}}(\boldsymbol{x}, t) = U^* \tilde{\mathbf{s}}_{\tilde{\theta}}(U\boldsymbol{x}, t)$ *in the time domain. The score matching loss in the frequency domain is equivalent to the score matching loss for the auxiliary score in the time domain:*

$$\mathcal{L}_{\text{SM}}\left(\tilde{\mathbf{s}}_{\tilde{\theta}}, \Lambda^2 \tilde{\mathbf{s}}_{t|0}, \tilde{\boldsymbol{x}}, t\right) = \mathcal{L}_{\text{SM}}\left(\mathbf{s}'_{\tilde{\theta}}, \mathbf{s}_{t|0}, \boldsymbol{x}, t\right) \tag{10}$$

where $\tilde{\mathbf{s}}_{t|0}(\tilde{\boldsymbol{x}}, t) = \nabla_{\tilde{\boldsymbol{x}}(t)} \log \tilde{p}_{t|0}(\tilde{\boldsymbol{x}}(t) | \tilde{\boldsymbol{x}}(0))$, $\mathbf{s}_{t|0}(\boldsymbol{x}, t) = \nabla_{\boldsymbol{x}(t)} \log p_{t|0}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$, *and* Λ *is the diagonal matrix in Proposition 3.3.*

Proof. The proof is given in Appendix A.4. □

Propositions 3.3 and 3.4 provide an explicit way to translate diffusion in the time domain to diffusion in the frequency domain. We note that attempting to solve Equations (3) and (9) in the finite-sample regime yields a local minimum solution in practice. Hence, there is no guarantee that training a score model in the frequency domain will converge to an auxiliary score $\mathbf{s}'_{\tilde{\theta}^*} = \mathbf{s}_{\theta^*}$ identical to the one obtained by training the score model in the time domain. In particular, having a score function $\tilde{\mathbf{s}}_\theta$ defined in the frequency domain is an important inductive bias, which is likely to alter the training dynamic. Through our experiments in the next section, we study the effect of this inductive bias on the resulting diffusion processes.

Take-away 1. Diffusion in the frequency domain can be implemented by replacing the standard Brownian motions with mirrored Brownian motions in the diffusion SDEs. The associated score can be optimized by minimizing a denoising score matching loss.

4. Comparing Time and Frequency Diffusion

In this section, we empirically analyze the effect of performing time series diffusion in the frequency domain [2]. In Section 4.1, we show that frequency diffusion models better capture their training distribution than time models. In Section 4.2, we argue that these differences of performance can be attributed the localization of the time series in the frequency domain. Finally, in Section 4.3, we artificially create settings where time models outperform the frequency models in order to confirm this hypothesis. The code necessary to reproduce the results, along with detailed instructions, is provided as a supplementary material.

Data. To illustrate the breadth of time series applications, we work with 6 different datasets described in Table 1. We observe that these datasets cover many use-cases (healthcare, finance, engineering and climate modelling), sample sizes, sequence lengths N and number of features tracked over time M. All the datasets are standardized before being fed to models. We also split the datasets into a training set $\mathcal{D}_{\text{train}}$ and a validation set $\mathcal{D}_{\text{val}}$. We provide more details on the datasets in Appendix B.1.

Models. For each dataset, we parametrize the time score model $\mathbf{s}_\theta$ and the frequency score model $\tilde{\mathbf{s}}_{\tilde{\theta}}$ as transformer encoders with 10 attention and MLP layers, each with 12 heads and dimension $d_{\text{model}} = 72$. Both models have learnable positional encoding as well as diffusion time t encoding through random Fourier features composed with a learnable dense layer. This results in models with 3.2M parameters. We use a VP-SDE with linear noise scheduling and $\beta_{\text{min}} = 0.1$ and $\beta_{\text{max}} = 20$, as in (Song et al., 2020). The score models are trained with the denoising score-matching loss, as defined in Section 3. All the models are trained for 200 epochs with batch size 64, AdamW optimizer and cosine learning rate scheduling (20 warmup epochs, $\text{lr}_{\text{max}} = 10^{-3}$). The selected model is the one achieving the lowest validation loss.

Time and Frequency. Crucially, the only difference between the time and the fequency diffusion models is the domain in which their input time series are represented. Since all datasets are expressed in the time domain, they can directly be fed to the time diffusion model $\mathbf{s}_\theta$. When it comes to the frequency diffusion model $\tilde{\mathbf{s}}_{\tilde{\theta}}$, the data is first mapped to the frequency domain by applying a DFT $\mathcal{F}$ on each time series. In the time domain, the forward and reverse diffusion obey the SDEs in Equations (1) and (2). In the frequency domain, the forward and reverse diffusion obey the modified SDEs in Equations (7) and (8). The denoised samples $\tilde{\mathbf{x}}(0)$ obtained in the frequency domain can be pulled back to the time domain by applying an inverse DFT $\tilde{\mathbf{x}}(0) \mapsto \mathcal{F}^{-1}[\tilde{\mathbf{x}}(0)]$. In the following, we shall denote by $\mathcal{S}_{\text{time}} \subset \mathbb{R}^{dx}$ and $\mathcal{S}_{\text{freq}} \subset \mathbb{R}^{dx}$ the time representation of the samples generated by the time and frequency models. Similarly, we shall denote by $\tilde{\mathcal{S}}_{\text{time}} := \mathcal{F}[\mathcal{S}_{\text{time}}]$ and $\tilde{\mathcal{S}}_{\text{freq}} := \mathcal{F}[\mathcal{S}_{\text{freq}}]$ the frequency representations of these time series. We sample $|\mathcal{S}_{\text{time}}| = |\mathcal{S}_{\text{freq}}| = 10,000$ samples for each model by applying $T = 1,000$ diffusion time steps.

4.1. Which Samples Better Capture the Distribution?

Methodology. We are interested in the faithfulness of the samples generated by the time and frequency diffusion models. Ideally, this faithfulness should be evaluated by computing the Wasserstein distance between the true distribution and the distribution spanned by our diffusion models. However, this is impossible since the exact computation of the Wasserstein distance in intractable in input spaces of large

[2]The code is publicly available at the following links: https://github.com/JonathanCrabbe/FourierDiffusion https://github.com/vanderschaarlab/FourierDiffusion

Table 1. Various datasets used in our experiments and some of their properties.

Dataset	*Reference*	*Field*	*# Samples*	*# Steps N*	*# Features M*
ECG	(Kachuee et al., 2018)	Healthcare	87,553	187	1
MIMIC-III	(Johnson et al., 2016)		19,155	24	40
NASDAQ-2019	(Onyshchak, 2020)	Finance	4,827	252	5
NASA-Charge	(Saha & Goebel, 2007)	Engineering	2,396	251	4
NASA-Discharge			1,755	134	5
US-Droughts	(Minixhofer, 2021)	Climate	2,797	365	13

Table 2. Sliced Wasserstein distances (↓) evaluated in the time domain ($SW(\mathcal{D}_{\mathrm{train}}, \mathcal{S}_{\mathrm{time}})$, $SW(\mathcal{D}_{\mathrm{train}}, \mathcal{S}_{\mathrm{freq}})$) and in the frequency domain ($SW(\tilde{\mathcal{D}}_{\mathrm{train}}, \tilde{\mathcal{S}}_{\mathrm{time}})$, $SW(\tilde{\mathcal{D}}_{\mathrm{train}}, \tilde{\mathcal{S}}_{\mathrm{freq}})$) on the various datasets. For each distance, we report its mean ± 2 standard errors.

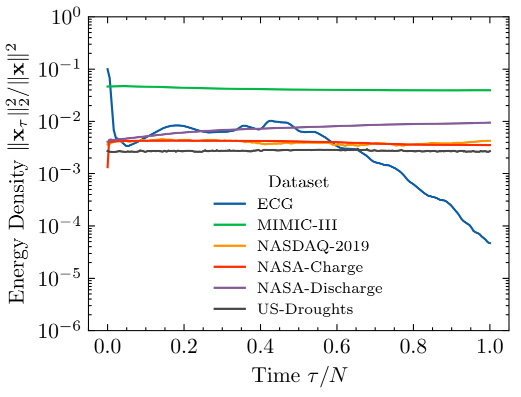
Dataset	*Metric Domain*	*Diffusion Domain*	
		Frequency	Time
ECG	Frequency	$\mathbf{0.012 \pm 0.000}$	0.020 ± 0.000
	Time	$\mathbf{0.015 \pm 0.000}$	0.021 ± 0.000
MIMIC-III	Frequency	$\mathbf{0.144 \pm 0.004}$	0.206 ± 0.006
	Time	$\mathbf{0.152 \pm 0.004}$	0.211 ± 0.006
NASDAQ-2019	Frequency	$\mathbf{45.812 \pm 2.096}$	64.056 ± 3.040
	Time	$\mathbf{43.602 \pm 2.044}$	60.512 ± 2.960
NASA-Charge	Frequency	$\mathbf{0.211 \pm 0.008}$	0.27 ± 0.006
	Time	$\mathbf{0.229 \pm 0.008}$	0.316 ± 0.008
NASA-Discharge	Frequency	$\mathbf{1.999 \pm 0.084}$	2.974 ± 0.134
	Time	$\mathbf{2.028 \pm 0.082}$	2.942 ± 0.134
US-Droughts	Frequency	$\mathbf{0.633 \pm 0.018}$	2.849 ± 0.090
	Time	$\mathbf{0.738 \pm 0.020}$	2.913 ± 0.092

dimension $d_X \gg 1$. In the case of images, (Heusel et al., 2017) mitigates these problems by mapping all the images in a lower dimensional representation space (the activation of the penultimate layer of an Inception-V3 model). This crucially relies on the fact that the Inception-V3 provides high-quality representations of images. Unfortunately, such a general representation of time series does not exist in practice. Hence, our evaluation needs to be performed in the input space $\mathbb{R}^{d_X}$ directly. For this reason, we shall rely on the sliced Wasserstein distance introduced by (Bonneel et al., 2015), which has similar properties to the Wasserstein distance and can be efficiently estimated in high dimension spaces. With a slight abuse of notation, we shall denote by $SW(\mathcal{S}_1, \mathcal{S}_2)$ the sliced Wasserstein distances between the empirical distributions corresponding to the samples $\mathcal{S}_1$ and $\mathcal{S}_2$. Its detailed definition is provided in Appendix B.2.

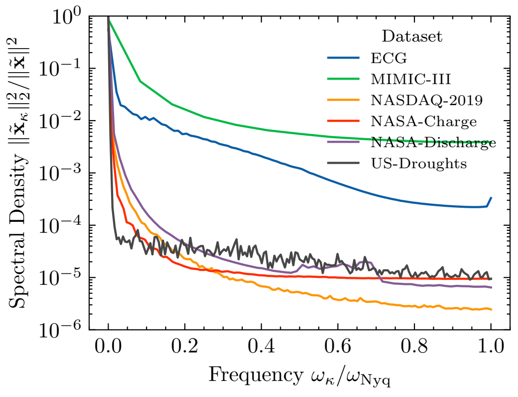
Analysis. The sliced Wasserstein distances are reported in Table 2. Interestingly, we observe that the frequency diffusion models consistently outperform the time diffusion models for all datasets both in the time and the frequency domain. In Appendix B.4, we show that using marginal Wasserstein distances instead of sliced Wasserstein distances essentially leads to the same conclusion. In order to verify that our observations are not specific to transformer backbones, we reproduce the same experiment with LSTM backbones in Appendix D and obtain similar results showing the superior performance of frequency diffusion models. While this observation already suggests the benefits of diffusing in the frequency domain rather than in the time domain, it is important to understand how these performance gains emerge. For this, we need to gain a better understanding of the training distributions, which is the object of next section.

4.2. How to Explain the Differences?

Signal energy analysis. Before formulating an hypothesis as to why the frequency models are better, it is helpful to gain a better understanding about the training distribution $\mathcal{D}_{\mathrm{train}}$. To that end, we leverage the energy and spectral densities related to the time series, described in Section 2. These densities are represented in Figure 1, where we have averaged the densities over all time series in $\mathcal{D}_{\mathrm{train}}$. By analyzing Figure 1b, we make a key observation: for all datasets, most of the time series energy in the frequency domain is localized on the frequency $\omega_0 = 0$ also known as the *fundamental frequency*. Furthermore, we observe that the energy quickly decays as the frequency increases. This observation suggests that the low frequencies capture most of the time series information. This is in stark contrast with the energy distribution over time in Figure 1a, which is more uniform over time for all the datasets. This asym-



(a) Time Domain.



(b) Frequency Domain.

Figure 1. Localization of time series in the time and frequency domains. Time series are more localized in the frequency domain.

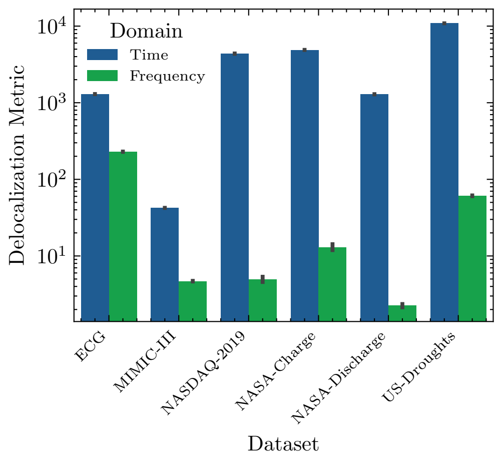


Figure 2. By evaluating our delocalization metrics in the time domain (Δ_{time}) and the frequency domain (Δ_{freq}), we quantitatively confirm that all the datasets are significantly more localized in the frequency domain. All the metrics are averaged over $\mathcal{D}_{\text{train}}$, their mean is reported with a 95% confidence interval.

metry between the time series spectral localization and their temporal delocalization is a promising candidate to explain the superior performances of frequency diffusion. We now make this observation more quantitative.

Quantitative Signal Localization. In order to measure how delocalized a time series $\mathbf{x} \in \mathbb{R}^{d_X}$ is in the time domain, we shall use the *delocalization metrics* introduced by (Nam, 2013):

$$\Delta_{\text{time}}(\mathbf{x}) := \min_{\tau \in [N]} \frac{1}{\|\mathbf{x}\|^2} \sum_{\tau' \in [N]} d_{\text{cyc}}(\tau, \tau') \|\mathbf{x}_{\tau'}\|_2^2, \quad (11)$$

where $d_{\text{cyc}} : [N]^2 \to [N]$ is the cyclic distance defined as $d_{\text{cyc}}(\tau, \tau') = \min(|\tau - \tau'|, N - |\tau - \tau'|)$, $\|\cdot\|$ denotes the Frobenius norm and $\|\cdot\|_2$ the Euclidean norm. Similarly, we can compute the delocalization in the frequency domain Δ_{freq} by replacing $\mathbf{x} \mapsto \tilde{\mathbf{x}}$ and $\tau, \tau' \mapsto \kappa, \kappa'$ in Equation (11). We report these delocalization metrics for each dataset in Figure 2. This quantitative analysis confirms our previous observations: the time series in all the datasets appear significantly more localized in the frequency domain. Interestingly, we never observe a time series that is localized in both the frequency and time domain simultaneously. This is in agreement with the *uncertainty principle* from (Nam, 2013), which echoes the foundational work of (Heisenberg, 1927). This verification is made in Appendix B.4.

A Localization Explanation. Based on the previous observation, we postulate that higher localization of the time series in the frequency domain contributes to the superior performance of frequency diffusion models. While we will test this hypothesis in the next section, it is useful to discuss the intuition behind it first. Due to the frequency localization, the frequency score model is presented with a representation of the time series where most of the relevant information is aligned with few components of the model's input (i.e. the lower frequencies, especially the fundamental). This is in contrast with the time model, which is presented with an input where all the components matter equally. It follows that the frequency model does not need to learn a good distribution over all frequencies in order to generate samples of high quality, provided the lower frequency distributions are properly learned. The time model, on the other hand, needs to model all the time steps accurately in order to generate high-quality samples. If this intuition is correct, it would imply that delocalizing the signal in the frequency domain should reduce the gap of performance between time and frequency models. This is analyzed in the next section.

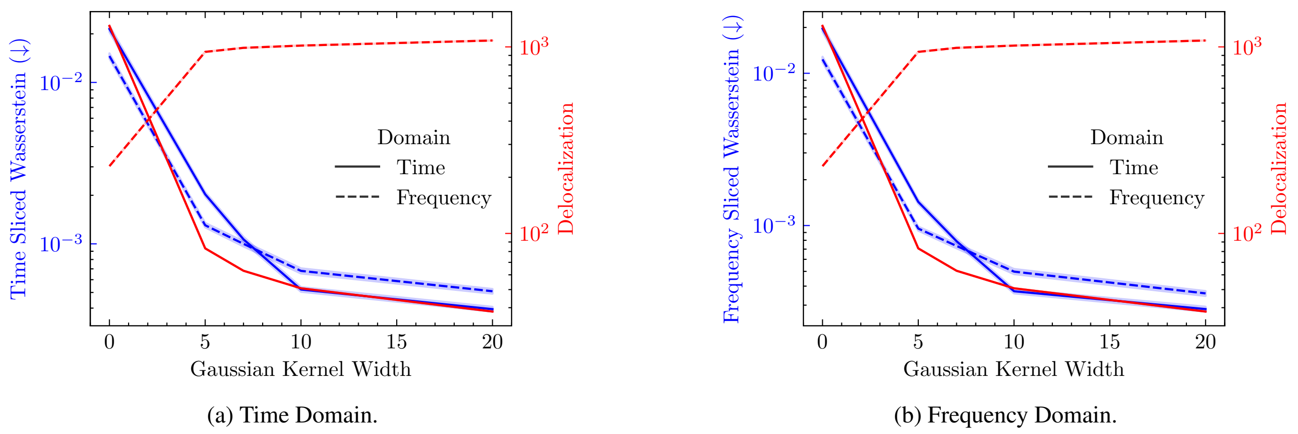


Figure 3. Sliced Wasserstein distances of time and frequency models (blue) and localization metrics in time and frequency domains (red) when smoothing the spectral representations of the time series with Gaussian kernels of variable width. Increasing the kernel width removes the localization in the frequency domain and increases the localization in the time domain. Coincidentally, the time diffusion model becomes better than the frequency diffusion model.

4.3. Should we Always Diffuse in the Frequency Domain?

Removing Spectral Localization. We would like to assess whether the localization of these signals in the frequency domain contribute to explain the superior performances of frequency diffusion models over time diffusion models. To test this hypothesis, we intervene on a given dataset with the objective of varying the localization of frequency and time representations. With this in mind, it is useful to start from a dataset where the imbalance between time and frequency localization is not severe. By looking at Figure 2, it is clear that the ECG dataset is the best candidate. In order to gradually remove the spectral localization from the ECG dataset $\mathcal{D}_{\mathrm{ECG}}$, we convolve the time series $\mathbf{x}$ in the frequency domain with Gaussians of increasing kernel width $\sigma \in \mathbb{R}^+$ and define $\mathbf{x}^\sigma := \mathcal{F}^{-1}\left[\mathcal{F}[\mathbf{x}] \star \mathbf{g}^\sigma\right]$, where $\star$ denotes the convolution between two signals and $g^\sigma_\kappa := Z^{-1}\exp[-\kappa^2/(2\sigma^2)]$ for all $\kappa \in [N]$ is a Gaussian kernel with normalization $Z = \sum_{\kappa=1}^N \exp[-\kappa^2/(2\sigma^2)]$. This results in a family of corrupted datasets $\mathcal{D}^\sigma_{\mathrm{ECG}}$, where the localization in the frequency domain decreases as σ increases. This is indeed what we observe in Figure 3 with the red curves. Coincidentally, the delocalization decreases in the time domain, in agreement with the uncertainty principle. The two curves cross at $\sigma \approx 2$, beyond which the time series are more localized in the time domain. Let us now analyze how the model performances evolve with different values of σ.

Analysis. We train time and frequency diffusion models on the datasets $\mathcal{D}^\sigma_{\mathrm{ECG}}$ for $\sigma \in \{0, 5, 7, 10, 20\}$, where $\sigma = 0$ corresponds to the original ECG dataset: $\mathcal{D}^{\sigma=0}_{\mathrm{ECG}} = \mathcal{D}_{\mathrm{ECG}}$. As in Section 4.1, we measure the quality of $10,000$ samples $\mathcal{S}$ produced by these models with the Wasserstein distances $SW(\mathcal{D}^\sigma_{\mathrm{ECG}}, \mathcal{S})$ in Figure 3a and $SW(\tilde{\mathcal{D}}^\sigma_{\mathrm{ECG}}, \tilde{\mathcal{S}})$ in Figure 3b. By inspecting the blue curves from these figures, we notice that decreasing the frequency closes the gap between the time and the frequency models. Moreover, the two curves cross around at $\sigma \approx 7$, beyond which the time model outperforms the frequency model. This confirms our hypothesis that the localization (at least) partially explains the better performance of the frequency diffusion model. Similar results can be obtained with other datasets, as discussed in Appendix B.4.

A Cautionary Remark. Before concluding, we would like to incorporate a bit of nuance in our analysis. While the above results suggest that localization is an important factor to explain the superior performance of frequency diffusion models, we *do not* claim that this is the only explanation. There are essentially two things that suggest that this only partially explains the observations from Section 4.1. (1) In Figure 3, the blue curves and the red curves don't cross for the same value of σ (i.e. the red curves cross at $\sigma \approx 2$ and the blue curves at $\sigma \approx 7$). Hence, the time model requires time series that are substantially more localized in the time domain in order to outperform the frequency model. (2) In the limit $\sigma \to +\infty$, the time series become constant in the frequency domain. While this corresponds to a minimal localization in the frequency model, this should also be very easy for a frequency diffusion model to learn. Hence, decreasing the localization of the time series in a given domain does not necessarily imply that the resulting time series are more difficult to model in that domain.

Take-away 2. For all datasets in this paper, diffusing in the frequency domain yields better performances than in the time domain. A promising explanation for this is that time series from these datasets are substantially more localized in their spectral representation than in their temporal representation.

5. Discussion

In this work, we have improved the understanding of how diffusion models should be used with time series. We constructed a theoretical framework that extends the score-based SDE formulation of diffusion to complex-valued times series representations in the frequency domain. We have then demonstrated empirically that implementing time series diffusion in the frequency domain consistently outperforms the canonical diffusion in the time domain. Finally, we showed that the spectral localization of the time series plays a significant role to explain this phenomenon. There is a number of interesting ways to extend our work.

Time Localized Datasets. While all 6 datasets we have studied appear substantially more localized in their spectral representation, we do not claim that is is a universal property of real-world time series. In particular, it would be of interest to survey a large amount of time series datasets to determine the extent to which this phenomenon occurs.

Multiresolution Analysis. It may be fruitful to represent time series at different resolutions to facilitate generative modeling. A multiresolution representation of time series can be obtained for example by computing a wavelet transform (Mallat, 1999) of these time series. Adapting the theory from Section 3 to wavelets in order to represent time series at multiple resolutions could represent an interesting avenue for future work.

Latent Diffusion. Latent diffusion has emerged as a fruitful direction of research in the diffusion literature (Rombach et al., 2022). A promising direction would be to study how spectral representations of time series can be incorporated to their latent representations and whether this benefits the quality of the generated samples. We leave these insightful research directions for future work.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

The authors are grateful to the 4 ICML reviewers for their useful comments on an earlier version of the manuscript. Jonathan Crabbé and Jan Stanczuk are funded by Aviva, Nicolas Huynh by Illumina and Mihaela van der Schaar by the Office of Naval Research (ONR), NSF 172251.

References

Alaa, A. M., Chan, A. J., and van der Schaar, M. Generative time-series modeling with fourier flows. In *International Conference on Learning Representations*, 2021.

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. ISSN 0304-4149. doi: https://doi.org/10.1016/0304-4149(82)90051-5.

Bonneel, N., Rabin, J., Peyré, G., and Pfister, H. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Dieleman, S., Sartran, L., Roshannai, A., Savinov, N., Ganin, Y., Richemond, P. H., Doucet, A., Strudel, R., Dyer, C., Durkan, C., et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022.

Donahue, C., McAuley, J., and Puckette, M. Adversarial audio synthesis. In *International Conference on Learning Representations*, 2018.

Esteban, C., Hyland, S. L., and Rätsch, G. Real-valued (medical) time series generation with recurrent conditional gans, 2017.

Gatta, F., Giampaolo, F., Prezioso, E., Mei, G., Cuomo, S., and Piccialli, F. Neural networks generative models for time series. *J. King Saud Univ. Comput. Inf. Sci.*, 34: 7920–7939, 2022.

Heisenberg, W. Über den anschaulichen inhalt der quantentheoretischen kinematik und mechanik. *Zeitschrift für Physik*, 43(3):172–198, 1927.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Hyvärinen, A. and Dayan, P. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Johnson, A., Pollard, T., and Mark, R. Mimic-iii clinical database (version 1.4). *PhysioNet*, 10(C2XW26):2, 2016.

Kachuee, M., Fazeli, S., and Sarrafzadeh, M. Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE international conference on healthcare informatics (ICHI)*, pp. 443–444. IEEE, 2018.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020.

Kloeden, P. E., Platen, E., Kloeden, P. E., and Platen, E. *Stochastic differential equations*. Springer, 1992.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

Körner, T. W. *Fourier analysis*. Cambridge university press, 2022.

Lin, L., Li, Z., Li, R., Li, X., and Gao, J. Diffusion models for time-series applications: a survey. *Frontiers of Information Technology & Electronic Engineering*, pp. 1–23, 2023.

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Van Gool, L. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11461–11471, 2022.

Mallat, S. *A wavelet tour of signal processing*. Elsevier, 1999.

Miller, A. R. and Tucker, C. Privacy protection and technology diffusion: The case of electronic medical records. *Management science*, 55(7):1077–1093, 2009.

Minixhofer, C. Predict droughts using weather and soil data, 2021.

Müller, M. Dynamic time warping. *Information retrieval for music and motion*, pp. 69–84, 2007.

Nam, S. An Uncertainty Principle for Discrete Signals. In *SampTA*, Bremen, Germany, 2013.

Nikolaidis, K., Kristiansen, S., Goebel, V., Plagemann, T., Liestøl, K., and Kankanhalli, M. Augmenting physiological time series data: A case study for sleep apnea detection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 376–399. Springer, 2019.

Onyshchak, O. Stock market dataset, 2020.

Phillips, A., Seror, T., Hutchinson, M., De Bortoli, V., Doucet, A., and Mathieu, E. Spectral diffusion processes. *arXiv preprint arXiv:2209.14125*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Saha, B. and Goebel, T. Battery data set, nasa ames prognostics data repository, 2007.

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4713–4726, 2022.

Schreier, P. J. and Scharf, L. L. *Statistical Signal Processing of Complex-Valued Data: The Theory of Improper and Noncircular Signals*. Cambridge University Press, 2010. doi: 10.1017/CBO9780511815911.

Shin, D., Shin, S., and Moon, I.-C. Frequency domain-based dataset distillation. *arXiv preprint arXiv:2311.08819*, 2023.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Wang, S., McDermott, M. B., Chauhan, G., Ghassemi, M., Hughes, M. C., and Naumann, T. Mimic-extract: A data extraction, preprocessing, and representation pipeline for mimic-iii. In *Proceedings of the ACM conference on health, inference, and learning*, pp. 222–235, 2020.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976): 1089–1100, 2023.

Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. Geodiff: a geometric diffusion model for molecular conformation generation. *ArXiv*, abs/2203.02923, 2022.

Yi, K., Zhang, Q., Cao, L., Wang, S., Long, G., Hu, L., He, H., Niu, Z., Fan, W., and Xiong, H. A survey on deep learning based time series analysis with frequency transformation. *CoRR, abs/2302.02173*, 2023.

Yoon, J., Jarrett, D., and Van der Schaar, M. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

Zeni, C., Pinsler, R., Zügner, D., Fowler, A., Horton, M., Fu, X., Shysheya, S., Crabbé, J., Sun, L., Smith, J., et al. Mattergen: a generative model for inorganic materials design. *arXiv preprint arXiv:2312.03687*, 2023.

A. Mathematical details

A.1. DFT properties

Proposition A.1 (Unitarity of the DFT operator)**.** *The DFT matrix* $U \in \mathbb{C}^{N \times N}$ *with elements* $[U]_{\kappa\tau} := N^{-1/2}\exp(-i\omega_\kappa \tau)$ *with* $\omega_\kappa := \frac{\kappa 2\pi}{N}$ *is unitary.*

Proof. Let U^* denote the conjugate transpose of U. For any κ and τ in $[N]$, we have:

$$\begin{aligned}
[UU^*]_{\kappa\tau} &= \sum_{\beta=0}^{N-1} [U]_{\kappa\beta}[U^*]_{\beta\tau} \\
&= \frac{1}{N}\sum_{\beta=0}^{N-1} \exp(-i\omega_\kappa \beta)\exp(i\omega_\beta \tau) \\
&= \frac{1}{N}\sum_{\beta=0}^{N-1} \exp(-i\omega_{\kappa-\tau}\beta)
\end{aligned}$$

Hence, if $\kappa = \tau$, we have $[UU^*]_{\kappa\tau} = 1$, otherwise $[UU^*]_{\kappa\tau} = 0$, since $\exp(-i\omega_{\kappa-\tau}N) = 1$. This is equivalent to $UU^* = I_N$, i.e. U is unitary. □

Proposition A.2. *The DFT* $\tilde{\boldsymbol{x}} = \mathcal{F}[\boldsymbol{x}] = U\boldsymbol{x}$ *of a real-valued time series* $\boldsymbol{x} \in \mathbb{R}^{d_X}$ *verifies the following* mirror symmetry *for all* $\kappa \in [N]$*:*

$$\tilde{\boldsymbol{x}}_\kappa = \tilde{\boldsymbol{x}}^*_{N-\kappa}.$$

Proof. Let κ and τ be in $[N]$. We first note that $\exp(i\omega_{N-\kappa}\tau) = \exp(i(\omega_N - \omega_\kappa)\tau) = \exp(-i\omega_\kappa\tau)$. Hence,

$$\begin{aligned}
\tilde{\mathbf{x}}^*_{N-\kappa} &= \sum_{\tau=0}^{N-1} [U]^*_{N-\kappa,\tau}\mathbf{x}_\tau && (\mathbf{x} \text{ is real valued}) \\
&= N^{-1/2}\sum_{\tau=0}^{N-1} \exp(i\omega_{N-\kappa}\tau)\mathbf{x}_\tau \\
&= N^{-1/2}\sum_{\tau=0}^{N-1} \exp(-i\omega_{\kappa}\tau)\mathbf{x}_\tau \\
&= \sum_{\tau=0}^{N-1} [U]_{\kappa,\tau}\mathbf{x}_\tau \\
&= \tilde{\mathbf{x}}_\kappa
\end{aligned}$$

□

A.2. Densities and scores for constrained signals

As we have mentioned in Section 2, the redundancy between certain components of the DFT $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$ of $\mathbf{x}$ expressed by Equation (6) needs to be taken into account if we wish to define a density $\tilde{p}$ for the time series distribution in the frequency domain. In particular, this redundancy implies that the density is really defined on a submanifold $\mathbb{C}^{d_X}_{\text{constr}} := \{\tilde{\mathbf{x}} = (\tilde{\mathbf{x}}_0, \dots, \tilde{\mathbf{x}}_{N-1}) \in \mathbb{C}^{d_X} \mid \tilde{\mathbf{x}}_\kappa = \tilde{\mathbf{x}}^*_{N-\kappa} \forall \kappa \in [N]\}$ of complex signals fulfilling the constraint. We can define a coordinate chart $\varphi : \mathbb{C}^{d_X}_{\text{constr}} \to \mathbb{R}^{d_X}$ on this submanifold by extracting the unconstrained part of a DFT $\tilde{\mathbf{x}}$. This operator simply concatenates the relevant real and imaginary parts of the DFT and is defined as follows for all $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}_{\text{constr}}$:

$$\varphi[\tilde{\mathbf{x}}] = \begin{cases} (\Re[\tilde{\mathbf{x}}_\kappa])_{\kappa=0}^{N/2} \oplus (\Im[\tilde{\mathbf{x}}_\kappa])_{\kappa=1}^{N/2-1} & \text{if } N \in 2\mathbb{N} \\ (\Re[\tilde{\mathbf{x}}_\kappa])_{\kappa=0}^{\lfloor N/2 \rfloor} \oplus (\Im[\tilde{\mathbf{x}}_\kappa])_{\kappa=1}^{\lfloor N/2 \rfloor} & \text{else,} \end{cases} \tag{12}$$

where $\boldsymbol{v}_1 \oplus \boldsymbol{v}_2$ denotes the concatenation of two vectors $\boldsymbol{v}_1 \in \mathbb{R}^{d_1}$ and $\boldsymbol{v}_2 \in \mathbb{R}^{d_2}$, with $d_1, d_2 \in \mathbb{N}$. Due to Equation (6), one can unambiguously reconstruct $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}_{\text{constr}}$ from $\varphi[\tilde{\mathbf{x}}]$. Hence, the coordinate chart admits an inverse $\varphi^{-1} : \mathbb{R}^{d_X} \to \mathbb{C}^{d_X}_{\text{constr}}$ defined as follows for all $\mathbf{z} = (\mathbf{z}_0, \dots, \mathbf{z}_{N-1}) \in \mathbb{R}^{d_X}$:

$$\varphi^{-1}[\mathbf{z}] = \begin{cases} (\mathbf{z}_0) \oplus (\mathbf{z}_\kappa + i \cdot \mathbf{z}_{N/2+\kappa})_{\kappa=1}^{N/2-1} \oplus (\mathbf{z}_{N/2}) \oplus (\mathbf{z}_{N/2-\kappa} - i \cdot \mathbf{z}_{N-\kappa})_{\kappa=1}^{N/2-1} & \text{if } N \in 2\mathbb{N} \\ (\mathbf{z}_0) \oplus (\mathbf{z}_\kappa + i \cdot \mathbf{z}_{\lfloor N/2 \rfloor+\kappa})_{\kappa=1}^{\lfloor N/2 \rfloor} \oplus (\mathbf{z}_{\lceil N/2 \rceil-\kappa} - i \cdot \mathbf{z}_{N-\kappa})_{\kappa=1}^{\lfloor N/2 \rfloor} & \text{else.} \end{cases} \tag{13}$$

With this coordinate chart, it becomes possible to rigorously define the density $\tilde{p} : \mathbb{C}^{d_X}_{\text{constr}} \to \mathbb{R}^+$. One simply defines a probability density $\tilde{p}_\varphi : \mathbb{R}^{d_X} \to \mathbb{R}^+$ over the real vector space $\mathbb{R}^{d_X}$ on which the coordinate chart is defined and pull it back to the manifold of constrained signals $\tilde{p} := \tilde{p}_\varphi \circ \varphi$. This indeed defines a density that depends on the real and imaginary parts of the frequency representations $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}_{\text{constr}}$ of time series, as announced in Section 2.

Finally, it remains to rigorously define the score $\tilde{\mathbf{s}} : \mathbb{C}^{d_X}_{\text{constr}} \times [0, T] \to \mathbb{C}^{d_X}$ in the frequency domain. This can be done by starting from the real score $\tilde{\mathbf{s}}_\varphi : \mathbb{R}^{d_X} \times [0, T] \to \mathbb{R}^{d_X}$ that is well-defined for all $\mathbf{z} \in \mathbb{R}^{d_X}$ and $t \in [0, T]$ as $\tilde{\mathbf{s}}_\varphi(\mathbf{z}, t) = \nabla_{\mathbf{z}} \log \tilde{p}_{\varphi,t}(\mathbf{z})$. Again, one can expand this vector field to the constrained manifold by defining for all $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}_{\text{constr}}$ and all $t \in [0, T]$:

$$\tilde{\mathbf{s}}(\tilde{\mathbf{x}}, t) := \varphi^{-1}[\tilde{\mathbf{s}}_\varphi(\varphi(\tilde{\mathbf{x}}), t)]. \tag{14}$$

This indeed defines a vector field involving partial derivatives of the log density with respect to the real and imaginary parts of the frequency representations $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}_{\text{constr}}$ of time series and respects the mirror symmetry by virtue of Equation (13). Everything is then consistent with the discussion from Section 2.

A.3. Diffusion SDEs in the frequency domain

Lemma 3.1. *(DFT of standard Brownian motion). Let $\boldsymbol{w}$ be a standard Brownian motion on $\mathbb{R}^{d_X}$ with $d_X = N \cdot M$, where $N \in \mathbb{N}^+$ is the number of time series steps and $M \in \mathbb{N}^+$ is the number of features tracked over time. Then $\boldsymbol{v} = U\boldsymbol{w}$ is a continuous stochastic process endowed with:*
(1) Mirror Symmetry. *For all $\kappa \in [N]$, $\boldsymbol{v}_\kappa = \boldsymbol{v}^*_{N-\kappa}$.*
(2) Real Brownian Motion. *$\boldsymbol{v}_0$ is a (real) standard Brownian motion on $\mathbb{R}^M$.*
(3) Complex Brownian Motions. *For all κ with $1 \leq \kappa \leq \lfloor N/2 \rfloor$, we can write $\boldsymbol{v}_\kappa = (\tilde{\boldsymbol{w}}^1_\kappa + i\tilde{\boldsymbol{w}}^2_\kappa)/\sqrt{2}$ where $\tilde{\boldsymbol{w}}^1_\kappa$ and $\tilde{\boldsymbol{w}}^2_\kappa$ are independent standard Brownian motions on $\mathbb{R}^M$, except when N is even and $\kappa = N/2$, where $\boldsymbol{v}_{N/2}$ is a real standard Brownian motion on $\mathbb{R}^M$.*
(4) Independence. *The stochastic processes $\{\boldsymbol{v}_\kappa\}_{\kappa=0}^{\lfloor N/2 \rfloor}$ are mutually independent.*
We call any stochastic process satisfying the above constraints a mirrored Brownian motion *on $\mathbb{C}^{d_X}$.*

Proof. **(1) Mirror Symmetry.** This point directly follows from the symmetry of the DFT, proved in Proposition A.2.

In what follows, we consider without loss of generality the case $M = 1$, since the cases $M > 1$ can be handled similarly by flattening matrices into vectors and using the same arguments as below. Let us first decompose U into its real and imaginary parts, i.e. $U = U_{re} + iU_{im}$, where U_{re} and U_{im} are in $\mathbb{R}^{N \times N}$. Note that these two matrices are both symmetric. Computing the distribution of the components of $\boldsymbol{v}$ will require the knowledge of covariance matrices, which will depend on U^2_{re}, U^2_{im} and $U_{re}U_{im}$. For any κ and τ in $[N]$,

$$\begin{aligned}[U^2_{re}]_{\kappa,\tau} &= \frac{1}{N} \sum_{\gamma=0}^{N-1} \cos(\frac{2\pi}{N}\kappa\gamma) \cos(\frac{2\pi}{N}\tau\gamma) \\ &= \frac{1}{2N} \sum_{\gamma=0}^{N-1} \big(\cos(\frac{2\pi}{N}(\kappa+\tau)\gamma) + \cos(\frac{2\pi}{N}(\kappa-\tau)\gamma)\big)\end{aligned}$$

Similarly,

$$\begin{aligned}
[U_{im}^2]_{\kappa,\tau} &= \frac{1}{N}\sum_{\gamma=0}^{N-1} \sin(\frac{2\pi}{N}\kappa\gamma)\sin(\frac{2\pi}{N}\tau\gamma) \\
&= \frac{1}{2N}\sum_{\gamma=0}^{N-1}\big(\cos(\frac{2\pi}{N}(\kappa-\tau)\gamma) - \cos(\frac{2\pi}{N}(\kappa+\tau)\gamma)\big)
\end{aligned}$$

To compute these sums, we consider the situations where:

- $N|\kappa-\tau$: this is equivalent to $\underline{\kappa=\tau}$, since $-(N-1)\leq \kappa-\tau \leq N-1$
- $N|\kappa+\tau$: this is equivalent to $\underline{\kappa=\tau=0}$ or $\underline{\kappa+\tau=N}$, since $0\leq\kappa+\tau\leq 2(N-1)$

Hence, if N is even:

$$[U_{re}^2]_{\kappa,\tau} = \begin{cases} 1 & \text{if } \kappa=\tau=0 \text{ or } \kappa=\tau=\frac{N}{2} \\ \frac{1}{2} & \text{if } \kappa\notin\{0,N/2\} \text{ and } \kappa=\tau \text{ or } \kappa=N-\tau \\ 0 & \text{otherwise}\end{cases} \text{ and } [U_{im}^2]_{\kappa,\tau} = \begin{cases} 0 & \text{if } \kappa=\tau=0 \text{ or } \kappa=\tau=\frac{N}{2} \\ \frac{1}{2} & \text{if } \kappa\notin\{0,N/2\} \text{ and } \kappa=\tau \\ -\frac{1}{2} & \text{if } \kappa\neq\ N/2 \text{ and } \kappa=N-\tau \\ 0 & \text{otherwise}\end{cases} \tag{15}$$

If N is odd:

$$[U_{re}^2]_{\kappa,\tau} = \begin{cases} 1 & \text{if } \kappa=\tau=0 \\ \frac{1}{2} & \text{if } \kappa\neq 0 \text{ and } \kappa=\tau, \text{ or } \kappa=N-\tau \\ 0 & \text{otherwise}\end{cases} \text{ and } [U_{im}^2]_{\kappa,\tau} = \begin{cases} 0 & \text{if } \kappa=\tau=0 \\ \frac{1}{2} & \text{if } \kappa\neq 0 \text{ and } \kappa=\tau \\ -\frac{1}{2} & \text{if } \kappa=N-\tau \\ 0 & \text{otherwise}\end{cases} \tag{16}$$

Finally, we compute $U_{re}U_{im}$:

$$\begin{aligned}
[U_{re}U_{im}]_{\kappa,\tau} &= \frac{1}{N}\sum_{\gamma=0}^{N-1}\cos(\frac{2\pi}{N}\kappa\gamma)\sin(\frac{2\pi}{N}\tau\gamma) \\
&= \frac{1}{2N}\sum_{\gamma=0}^{N-1}\big(\sin(\frac{2\pi}{N}(\tau-\kappa)\gamma) + \sin(\frac{2\pi}{N}(\kappa+\tau)\gamma)\big) \\
&= 0
\end{aligned}$$

Hence, $U_{re}U_{im} = 0_N$ and similarly $U_{im}U_{re} = 0_N$ by taking the transpose and using the symmetry of U_{re} and U_{im}. We can now characterize the distribution followed by the stochastic process $\boldsymbol{v}$. We first write $U_{col} = \begin{pmatrix} U_{re} \\ U_{im}\end{pmatrix}$, and then notice that $\boldsymbol{v}$ can be investigated through the lens of its flattened version $\boldsymbol{v}_{flat} = U_{col}\boldsymbol{w}$, which is a stochastic process in $\mathbb{R}^{2N}$.

(2) Real Brownian Motion. First, $\boldsymbol{v}_0$ is real-valued, by using **(1) Mirror Symmetry**. We then have the following:

- $\boldsymbol{v}_0(0)=0$ almost surely: this stems from $\boldsymbol{w}(0)=0$ almost surely, since $\boldsymbol{w}$ is a multivariate standard Brownian motion and $\boldsymbol{v}=U\boldsymbol{w}$.
- Continuity of $t\to\boldsymbol{v}_0(t)$ almost surely: $\boldsymbol{w}$ satisfies the continuity property and the DFT operator (seen as a complex operator) is linear, hence $\boldsymbol{v}_0$ is also continuous with respect to t almost surely.
- Stationary and independent increments: this follows from the linearity of the DFT operator and $\boldsymbol{w}$ being a Brownian motion.

- For any $t, s \geq 0$, $\boldsymbol{v}_0(t+s) - \boldsymbol{v}_0(s) \sim \mathcal{N}(0, t)$: to see this, we notice that $\boldsymbol{v}_0(t+s) - \boldsymbol{v}_0(s)$ is Gaussian[3] since it is a linear transform of $\boldsymbol{w}(t+s) - \boldsymbol{w}(s)$. Moreover, its mean and its variance are given by:

$$\begin{aligned}\mathbb{E}\big[\boldsymbol{v}_0(t+s) - \boldsymbol{v}_0(s)\big] &= \big[U_{col}\mathbb{E}[\boldsymbol{w}(t+s) - \boldsymbol{w}(s)]\big]_0 \\ &= 0\end{aligned}$$

$$\begin{aligned}\text{Var}(\boldsymbol{v}_0(t+s) - \boldsymbol{v}_0(s)) &= \big[U_{col}\text{Cov}\big(\boldsymbol{w}(t+s) - \boldsymbol{w}(s), \boldsymbol{w}(t+s) - \boldsymbol{w}(s)\big)U_{col}^T)\big]_{0,0} \\ &= t[U_{col}U_{col}^T]_{0,0} \\ &= t[U_{re}^2]_{0,0} \\ &= t\end{aligned}$$

Hence, we have shown that $\boldsymbol{v}_0$ is a real Brownian motion.

(3) Complex Brownian Motions. Let $1 \leq \kappa < \lfloor N/2 \rfloor$. Then, $\Re(\boldsymbol{v}_\kappa)$ and $\Im(\boldsymbol{v}_\kappa)$ follow the first three properties of a Brownian motion, using the same arguments as above. For the last point, we first characterize the distribution of $\Re(\boldsymbol{v}_\kappa)$ and $\Im(\boldsymbol{v}_\kappa)$, and then show that they are independent.

Distribution of $\Re(\boldsymbol{v}_\kappa)$. $\sqrt{2}\Re(\boldsymbol{v}_\kappa)$ is a standard Brownian motion in $\mathbb{R}$: For any $t, s \geq 0$, $\Re(\boldsymbol{v}_\kappa)(t+s) - \Re(\boldsymbol{v}_\kappa)(s)$ is Gaussian since it is a linear transform of $\boldsymbol{w}(t+s) - \boldsymbol{w}(s)$ which is a Gaussian vector. We can compute its mean and its variance:

$$\begin{aligned}\mathbb{E}\big[\Re(\boldsymbol{v}_\kappa)(t+s) - \Re(\boldsymbol{v}_\kappa)(s)\big] &= \big[U_{col}\mathbb{E}[\boldsymbol{w}(t+s) - \boldsymbol{w}(s)]\big]_\kappa \\ &= 0\end{aligned}$$

$$\begin{aligned}\text{Var}\big(\Re(\boldsymbol{v}_\kappa)(t+s) - \Re(\boldsymbol{v}_\kappa)(s)\big) &= \big[U_{col}\text{Cov}\big(\boldsymbol{w}(t+s) - \boldsymbol{w}(s), \boldsymbol{w}(t+s) - \boldsymbol{w}(s)\big)U_{col}^T)\big]_{\kappa,\kappa} \\ &= t[U_{col}U_{col}^T]_{\kappa,\kappa} \\ &= t[U_{re}^2]_{\kappa,\kappa} \\ &= \frac{1}{2}t\end{aligned}$$

Distribution of $\Im(\boldsymbol{v}_\kappa)$. Similarly, we can prove with the same arguments that $\sqrt{2}\Im(\boldsymbol{v}_\kappa)$ is a standard Brownian motion in $\mathbb{R}$.

Independence of $\Re(\boldsymbol{v}_\kappa)$ *and* $\Im(\boldsymbol{v}_\kappa)$. Let k and m be two strictly positive integers, and let $(t_1, ..., t_k) \in (\mathbb{R}_+^*)^k$ and $(t'_1, ..., t'_m) \in (\mathbb{R}_+^*)^m$. We need to show that the vectors $\boldsymbol{v}_\kappa^{re} = \big(\Re(\boldsymbol{v}_\kappa)(t_1), ..., \Re(\boldsymbol{v}_\kappa)(t_k)\big)$ and $\boldsymbol{v}_\kappa^{im} = \big(\Im(\boldsymbol{v}_\kappa)(t'_1), ..., \Im(\boldsymbol{v}_\kappa)(t'_m)\big)$ are independent. First, the concatenation of $\boldsymbol{v}_\kappa^{re}$ and $\boldsymbol{v}_\kappa^{im}$ can be expressed as a linear transform of $(\boldsymbol{w}(t_1), ..., \boldsymbol{w}(t_k), \boldsymbol{w}(t'_1), ..., \boldsymbol{w}(t'_m))$, which is a Gaussian vector since $\boldsymbol{w}$ is a Brownian motion. Consequently, $(\boldsymbol{v}_\kappa^{re}, \boldsymbol{v}_\kappa^{im})$ is also a Gaussian vector. Now, let $l \in \{1, ..., k\}$ and $n \in \{1, ..., m\}$. Then,

$$\begin{aligned}\text{Cov}\big(\Re(\boldsymbol{v}_\kappa)(t_l), \Im(\boldsymbol{v}_\kappa)(t'_n)\big) &= \big[U_{re}\text{Cov}\big(\boldsymbol{w}(t_l), \boldsymbol{w}(t'_n)\big)U_{im}\big]_{\kappa,\kappa} \\ &= \min(t_l, t'_n)[U_{re}U_{im}]_{\kappa,\kappa} \\ &= 0\end{aligned}$$

Given this covariance structure and the fact that $(\boldsymbol{v}_\kappa^{re}, \boldsymbol{v}_\kappa^{im})$ is a Gaussian vector, we have $\boldsymbol{v}_\kappa^{re} \perp\!\!\!\perp \boldsymbol{v}_\kappa^{im}$. Since this holds true for any choice of $(t_1, ..., t_k) \in {\mathbb{R}_+^*}^k$ and $(t'_1, ..., t'_m) \in {\mathbb{R}_+^*}^m$, we conclude that $\Re(\boldsymbol{v}_\kappa) \perp\!\!\!\perp \Im(\boldsymbol{v}_\kappa)$. The case $k = \lfloor \frac{N}{2} \rfloor$ can be handled using the same arguments, by distinguishing the cases N odd and N even.

(4) Independence. The mutual independence of the stochastic processes $\{\tilde{\boldsymbol{v}}_\kappa\}_{\kappa=0}^{\lfloor N/2 \rfloor}$ follows from the structure of U_{re}^2 and U_{im}^2. Indeed, for any m and n such that $m \neq n$, $0 \leq m \leq \lfloor \frac{N}{2} \rfloor$, and $0 \leq n \leq \lfloor \frac{N}{2} \rfloor$, we have $[U_{re}^2]_{m,n} = [U_{im}^2]_{m,n} = [U_{re}U_{im}]_{m,n} = 0$. We can then apply the same argument as in 3) of **(3) Complex Brownian Motions** to obtain the mutual independence of the stochastic processes $\{\tilde{\boldsymbol{v}}_\kappa\}_{\kappa=0}^{\lfloor N/2 \rfloor}$. $\square$

[3] Note that a random variable almost surely equal to 0 can be seen as a degenerate Gaussian with mean 0 and variance 0.

Proposition 3.3. *(Diffusion process in frequency domain). Let us assume that $\boldsymbol{x}$ is a diffusion process that is a solution of Equation (1), with $\boldsymbol{G}(t) = g(t)\, I_N$. Then $\tilde{\mathbf{x}} = \mathcal{F}[\mathbf{x}]$ is a solution to the forward diffusion process defined by:*

$$\mathrm{d}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t)\mathrm{d}t + g(t)\mathrm{d}\tilde{\boldsymbol{v}}, \tag{7}$$

where $\tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t) = U\boldsymbol{f}(U^*\tilde{\boldsymbol{x}}, t)$ and $\tilde{\boldsymbol{v}}$ is a mirrored Brownian motion on $\mathbb{C}^{dx}$. The associated reverse diffusion process is defined by:

$$\mathrm{d}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}}, t)\mathrm{d}t + g(t)\mathrm{d}\breve{\boldsymbol{v}} \tag{8}$$

where $\tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}}, t) = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}, t) - g^2(t)\Lambda^2\tilde{\mathbf{s}}(\tilde{\boldsymbol{x}}, t)$, $\Lambda \in \mathbb{R}^{N\times N}$ is a diagonal matrix such that $[\Lambda]_{\kappa,\kappa} = \begin{cases} 1 & \text{if } \kappa = 0, \text{ or } N \text{ is even and } \kappa = N/2 \\ \frac{1}{\sqrt{2}} & \text{otherwise} \end{cases}$, $\mathrm{d}t$ is a negative infinitesimal time step, and $\breve{\boldsymbol{v}}$ is a mirrored Brownian motion on $\mathbb{C}^{dx}$ with time going from T to 0.

Proof. Forward SDE. Since $\tilde{\mathbf{x}} = U\mathbf{x}$, we can apply the multivariate Itô's lemma (Eq. 8.3, (Kloeden et al., 1992)), and obtain a forward SDE for $\tilde{\mathbf{x}}$:

$$\mathrm{d}\tilde{\mathbf{x}} = U\boldsymbol{f}(\mathbf{x}, t)\mathrm{d}t + g(t)U\mathrm{d}\boldsymbol{w}' \tag{17}$$

where we have implicitly used the fact that $\boldsymbol{G}(t) = g(t)I_N$ and U commute. By Lemma 3.1, $\tilde{\boldsymbol{v}} = U\boldsymbol{w}'$ is a mirrored Brownian motion on $\mathbb{C}^{dx}$, which gives the result.

Reverse SDE. In order to derive the reverse SDE for $\tilde{\mathbf{x}}$, we follow three steps: (1) we write a forward SDE for which the truncation $\varphi[\tilde{\mathbf{x}}]$ is a solution ; (2) we write its associated reverse-time SDE (Anderson, 1982) and (3) we derive the full reverse SDE for the stochastic process $\tilde{\mathbf{x}}$.

Step 1: From Equation (17) and using Lemma 3.1, we can extract the following forward SDE for $\varphi[\tilde{\mathbf{x}}]$, which is defined in Equation (12) :

$$\mathrm{d}\varphi[\tilde{\mathbf{x}}] = \varphi\big[U\boldsymbol{f}(U^*\varphi^{-1}(\varphi[\tilde{\mathbf{x}}]), t)\big]\mathrm{d}t + g(t)\Lambda\mathrm{d}\breve{\boldsymbol{w}} \tag{18}$$

where:

- $\varphi^{-1} : \mathbb{R}^{dx} \to \mathbb{C}^{dx}$ satisfies $\varphi^{-1}(\varphi[U\mathbf{y}]) = U\mathbf{y}$ for all $\mathbf{y} \in \mathbb{R}^{dx}$ as defined in Equation (13).
- $\Lambda \in \mathbb{R}^{N\times N}$ is a diagonal matrix such that $[\Lambda]_{\kappa,\kappa} = \begin{cases} 1 & \text{if } \kappa = 0, \text{ or } N \text{ is even and } \kappa = N/2 \\ \frac{1}{\sqrt{2}} & \text{otherwise} \end{cases}$,
- $\breve{\boldsymbol{w}}$ is a stochastic process in $\mathbb{R}^{dx}$ which satisfies $\Lambda\breve{\boldsymbol{w}} = \varphi[U\boldsymbol{w}']$. The above point, together with Lemma 3.1), then implies that $\breve{\boldsymbol{w}}$ is in fact a standard multivariate Brownian motion.

Step 2: The associated reverse-time SDE (Anderson, 1982) is given by:

$$\mathrm{d}\varphi[\tilde{\mathbf{x}}] = \big\{\varphi\big[U\boldsymbol{f}(U^*\varphi^{-1}(\varphi[\tilde{\mathbf{x}}]), t)\big] - g(t)^2\Lambda\Lambda^T\nabla_{\varphi[\tilde{\mathbf{x}}]}\log\tilde{p}_t(\varphi[\tilde{\mathbf{x}}])\big\}\mathrm{d}t + g(t)\Lambda\mathrm{d}\widehat{\boldsymbol{w}} \tag{19}$$

where $\widehat{\boldsymbol{w}}$ is a standard Brownian motion on $\mathbb{R}^{dx}$.

Step 3: Since $\varphi^{-1}(\varphi[\tilde{\mathbf{x}}]) = \tilde{\mathbf{x}}$, we can recover the reverse-time SDE followed by $\tilde{\mathbf{x}}$ by applying the operator φ^{-1} to $\varphi[\tilde{\mathbf{x}}]$, and using the Itô's lemma:

$$\mathrm{d}\tilde{\mathbf{x}} = \big\{\varphi^{-1}(\varphi\big[U\boldsymbol{f}(U^*\varphi^{-1}(\varphi[\tilde{\mathbf{x}}]), t)\big]) - g(t)^2\varphi^{-1}\big(\Lambda^2\nabla_{\varphi[\tilde{\mathbf{x}}]}\log\tilde{p}_t(\varphi[\tilde{\mathbf{x}}])\big)\big\}\mathrm{d}t + g(t)\varphi^{-1}(\Lambda)\mathrm{d}\widehat{\boldsymbol{w}} \tag{20}$$

where we have exploited the fact that φ^{-1} is linear, and with the slight abuse of notation $\varphi^{-1}(\Lambda) \in \mathbb{C}^{N\times N}$, which is the matrix obtained by applying φ^{-1} to the columns of Λ. Using the definition of φ^{-1}, Equation (20) can further be simplified into:

$$\mathrm{d}\tilde{\mathbf{x}} = \big\{U\boldsymbol{f}(U^*\tilde{\mathbf{x}}, t) - g(t)^2\varphi^{-1}\big(\Lambda^2\nabla_{\varphi[\tilde{\mathbf{x}}]}\log\tilde{p}_t(\varphi[\tilde{\mathbf{x}}])\big)\big\}\mathrm{d}t + g(t)\mathrm{d}\breve{\boldsymbol{v}} \tag{21}$$

where $\breve{\boldsymbol{v}} = \varphi^{-1}(\Lambda)\widehat{\boldsymbol{w}} = \varphi^{-1}(\Lambda\widehat{\boldsymbol{w}})$ is a mirrored Brownian motion. Finally, notice that for any $\mathbf{y} \in \mathbb{R}^{dx}$, we have $\varphi^{-1}(\Lambda^2\mathbf{y}) = \Lambda^2\varphi^{-1}(\mathbf{y})$, which follows from the definition of φ^{-1}. Hence, Equation (21) is equivalent to:

$$\mathrm{d}\tilde{\mathbf{x}} = \big\{U\boldsymbol{f}(U^*\tilde{\mathbf{x}}, t) - g(t)^2\Lambda^2\tilde{\mathbf{s}}(\tilde{\mathbf{x}}, t)\big\}\mathrm{d}t + g(t)\mathrm{d}\breve{\boldsymbol{v}} \tag{22}$$

where $\tilde{\mathbf{s}}(\tilde{\mathbf{x}}, t)$ has been defined in Equation (14).

$\square$

A.4. Denoising score matching in the frequency domain

Proposition 3.4. *(Score matching equivalence). Consider a score $\tilde{\mathbf{s}}_{\tilde{\theta}} : \mathbb{C}^{d_X} \times [0, T] \to \mathbb{C}^{d_X}$ defined in the frequency domain and satisfying the mirror symmetry $[\tilde{\mathbf{s}}_{\tilde{\theta}}]_\kappa = [\tilde{\mathbf{s}}^*_{\tilde{\theta}}]_{N-\kappa}$ for all $\kappa \in [N]$. Let us define an auxiliary score $\mathbf{s}'_{\tilde{\theta}} : \mathbb{R}^{d_X} \times [0, T] \to \mathbb{R}^{d_X}$ as $(\boldsymbol{x}, t) \mapsto \mathbf{s}'_{\tilde{\theta}}(\boldsymbol{x}, t) = U^* \tilde{\mathbf{s}}_{\tilde{\theta}}(U\boldsymbol{x}, t)$ in the time domain. The score matching loss in the frequency domain is equivalent to the score matching loss for the auxiliary score in the time domain:*

$$\mathcal{L}_{\mathrm{SM}}\left(\tilde{\mathbf{s}}_{\tilde{\theta}}, \Lambda^2 \tilde{\mathbf{s}}_{t|0}, \tilde{\boldsymbol{x}}, t\right) = \mathcal{L}_{\mathrm{SM}}\left(\mathbf{s}'_{\tilde{\theta}}, \mathbf{s}_{t|0}, \boldsymbol{x}, t\right) \tag{10}$$

where $\tilde{\mathbf{s}}_{t|0}(\tilde{\boldsymbol{x}}, t) = \nabla_{\tilde{\boldsymbol{x}}(t)} \log \tilde{p}_{t|0}(\tilde{\boldsymbol{x}}(t)|\tilde{\boldsymbol{x}}(0))$, $\mathbf{s}_{t|0}(\boldsymbol{x}, t) = \nabla_{\boldsymbol{x}(t)} \log p_{t|0}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$, and Λ is the diagonal matrix in Proposition 3.3.

Proof. Let $\Lambda \in \mathbb{R}^{N\times N}$ be the diagonal matrix such that $[\Lambda]_{\kappa,\kappa} = \begin{cases} 1 & \text{if } \kappa = 0, \text{ or } N \text{ is even and } \kappa = N/2 \\ \frac{1}{\sqrt{2}} & \text{otherwise} \end{cases}$

Step 1: We first express the score of $\mathbf{x}$ with respect to the score of the truncation $\varphi[\tilde{\mathbf{x}}]$.

By definition of φ^{-1} in Equation (13), we have $\mathbf{x} = U^*\varphi^{-1}(\varphi[\tilde{\mathbf{x}}])$. Hence, we can write, using the change of variable formula:

$$p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0)) = C \cdot \tilde{p}_{t|0}\big(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)]\big) \tag{23}$$

where C is a constant which does not depend on $\mathbf{x}$, since $\mathbf{x} \mapsto \varphi[U\mathbf{x}]$ is linear. Moreover, let us write $\mathbf{x} \mapsto \varphi[U\mathbf{x}]$ in matrix form, i.e. $\forall \mathbf{x} \in \mathbb{R}^{d_X}$, $\varphi[U\mathbf{x}] = VU_{col}\mathbf{x} = Q\mathbf{x}$, where $V \in \mathbb{R}^{N\times 2N}$, $U_{col} = \begin{pmatrix} U_{re} \\ U_{im} \end{pmatrix}$ and Q is an invertible matrix in $\mathbb{R}^{N\times N}$. For the rest of the proof, we shall build on the below results:

Result 1. $QQ^T = \Lambda^2$. To see this, write $QQ^T = VU_{col}U_{col}^TV^T$. The matrix $U_{col}U_{col}^T$ is equal to $\begin{pmatrix} U_{re}^2 & 0_N \\ 0_N & U_{im}^2 \end{pmatrix}$ (cf. the proof to Lemma 3.1), while the multiplication by V, on the left and on the right of $U_{col}U_{col}^T$, extracts the submatrix corresponding to the indices represented by the truncature φ. Hence, $VU_{col}U_{col}^TV^T = \Lambda^2$.

Result 2. For any $\mathbf{x} \in \mathbb{R}^{d_X}$, we have $Q^T\mathbf{x} = U^*\varphi^{-1}[\Lambda^2\mathbf{x}]$. To see this, notice that Result 1 implies that $Q^T\mathbf{x} = Q^{-1}\Lambda^2\mathbf{x} = U^*\varphi^{-1}[\Lambda^2\mathbf{x}]$ for all $\mathbf{x} \in \mathbb{R}^{d_X}$.

Equipped with these results, we can now complete the rest of the proof. First, we have:

$$\begin{aligned} \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0)) &= \nabla_{\mathbf{x}(t)} \log \tilde{p}_{t|0}\big(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)]\big) && (24) \\ &= Q^T \nabla_{\varphi[\tilde{\mathbf{x}}(t)]} \log \tilde{p}_{t|0}(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)]) \qquad \text{(Chain rule)} && (25) \end{aligned}$$

Step 2: We then obtain:

$$\begin{aligned} \mathcal{L}_{\mathrm{SM}}\left(\mathbf{s}'_{\tilde{\theta}}, \mathbf{s}_{t|0}, \mathbf{x}, t\right) &:= \|\mathbf{s}'_{\tilde{\theta}}(\mathbf{x}, t) - \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0))\|^2 \\ &= \|U\mathbf{s}'_{\tilde{\theta}}(\mathbf{x}, t) - U\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}(0))\|^2 && \text{(Parseval identity)} \\ &= \|\tilde{\mathbf{s}}_{\tilde{\theta}}(\tilde{\mathbf{x}}, t) - UQ^T\nabla_{\varphi[\tilde{\mathbf{x}}(t)]} \log \tilde{p}_{t|0}(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)])\|^2 && \text{(Equation (25))} \\ &= \|\tilde{\mathbf{s}}_{\tilde{\theta}}(\tilde{\mathbf{x}}, t) - UU^*\varphi^{-1}[\Lambda^2\nabla_{\varphi[\tilde{\mathbf{x}}(t)]} \log \tilde{p}_{t|0}(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)]]\|^2 && \text{(Result 2)} \\ &= \|\tilde{\mathbf{s}}_{\tilde{\theta}}(\tilde{\mathbf{x}}, t) - \Lambda^2\varphi^{-1}[\nabla_{\varphi[\tilde{\mathbf{x}}(t)]} \log \tilde{p}_{t|0}(\varphi[\tilde{\mathbf{x}}(t)]|\varphi[\tilde{\mathbf{x}}(0)]]\|^2 && \text{(Proposition A.1 \& Definition of } \varphi^{-1}) \\ &= \|\tilde{\mathbf{s}}_{\tilde{\theta}}(\tilde{\mathbf{x}}, t) - \Lambda^2\tilde{\mathbf{s}}_{t|0}(\tilde{\mathbf{x}}, t)\|^2 && \text{(Equation (14))} \\ &= \mathcal{L}_{\mathrm{SM}}\left(\tilde{\mathbf{s}}_{\tilde{\theta}}, \Lambda^2\tilde{\mathbf{s}}_{t|0}, \tilde{\mathbf{x}}, t\right). \end{aligned}$$

$\square$

B. Empirical details

Compute resources. All the models were trained and used for sampling on a single machine equipped with a 18-Core Intel Core i9-10980XE CPU, a NVIDIA RTX A4000 GPU and a NVIDIA GeForce RTX 3080.

B.1. Details on datasets

In this subsection, we give detailed information about the 6 datasets used throughout our experiments and the preprocessing steps for each of them.

ECG. We use two collections of heartbeat signals, from the MIT-BIH Arrhythmia Dataset and the PTB Diagnostic ECG Database (Kachuee et al., 2018). No preprocessing was performed on this dataset.

MIMIC-III. MIMIC-III (Johnson et al., 2016) is a database consisting of deidentified records for patients who were in critical unit care units. *Preprocessing.* We use the "vitals labs" table of the database, which corresponds to time-varying vitals and labs. We extract the rows of the dataset which correspond to the first 24 hours of stay by using *MIMIC-Extract* (Wang et al., 2020). The features are then standardized across all times and patients. We also perform imputation to handle missing values in the dataset. To do so, we consider the mean features (average measurement over 1 hour). For each patient, and missing value, we propagate the last observation forward if this is possible. If not, we fill the missing value with the mean value for the patient (which is computed over the whole stay). If no mean value is available, we fill the entry with 0.

NASDAQ-2019. This dataset (Onyshchak, 2020) contains daily prices for tickers trading on NASDAQ, and contains prices for up to 1st of April 2020. *Preprocessing.* We considered one year of daily prices from 1st of January 2019 to 1st of January 2020. Each sample corresponds to one stock, and we remove the stocks which are not active in this whole time interval, or contain missing values.

NASA battery. The NASA battery dataset (Saha & Goebel, 2007) consists of profiles for Li-on batteries, under charge and discharge. *Preprocessing.* For both the charge and discharge datasets, we bin the time values (bins of size 10 for Charge, 15 for Discharge) and compute the mean of each feature inside each bin.

US-Droughts. This dataset (Minixhofer, 2021) consists of drought levels in different US counties, from 2000 to 2020. *Preprocessing.* We consider one year of history, from 1st of January 2011 to 1st of January 2012, and drop the columns with missing values.

B.2. Details on evaluation

Sliced Wasserstein Distances. The sliced Wasserstein distance (Bonneel et al., 2015) is a metric which can handle high-dimensional distributions. It is motivated by the fact that the Wasserstein distance is easy to compute when comparing two one-dimensional distributions. The idea of the sliced Wasserstein distance is to map the high-dimensional distributions of interest to one-dimensional distributions, by considering random projections on vectors of the unit sphere. For two distributions μ_1 and μ_2, it can be written as:

$$SW_p(\mu_1, \mu_2) \coloneqq \int_{\mathbb{S}^{d-1}} W_p(P_u\#\mu_1, P_u\#\mu_2)du \tag{26}$$

where $\mathbb{S}^{d-1}$ is the unit sphere in dimension d, $P_u(x) = u \cdot x$ denotes the projection of x on u, $P_u\#\mu$ is the push-forward of μ by P_u, and W_p is the Wasserstein distance of order p. To estimate this quantity in practice, we sample $n = 10,000$ random vectors $\{u_i | i \in [n]\}$ which follow a uniform distribution in $\mathbb{S}^{d-1}$ and consider $p = 2$. Hence, we can approximate SW_p by the Monte-carlo estimator:

$$S\hat{W}_p(\mu_1, \mu_2) = \frac{1}{n}\sum_{i=1}^{n} W_p(P_{u_i}\#\mu_1, P_{u_i}\#\mu_2) \tag{27}$$

Marginal Wasserstein Distances. In addition to the sliced Wasserstein distance, we also consider the marginal Wasserstein distance. For any $j \in \{1, ..., d\}$, the j-th marginal Wasserstein distance is defined as:

$$MW_p{}^{(j)}(\mu_1, \mu_2) = W_p(P_{e_j}\#\mu_1, P_{e_j}\#\mu_2) \tag{28}$$

where e_j is the j-th vector of the standard basis of $\mathbb{R}^d$. Throughout our experiments in Section 4, we compute the Wasserstein distances with respect to $\mathcal{D}_{\mathrm{train}}$ and $\tilde{\mathcal{D}}_{\mathrm{train}}$.

B.3. Computational cost

Since the Discrete Fourier Transform (DFT) is a bijection, and by virtue of the mirror symmetry from Equation (6), the frequency representation of a time series has the same number of real independent components as its time representation. As a consequence, the diffusion in the frequency domain can be performed without any additional cost by restricting to these independent components, while keeping in mind that other components can be deduced from the mirror symmetry.

We now make this more precise. Due to the mirror symmetry in the frequency representation $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$ of a time series $\mathbf{x} \in \mathbb{R}^d$, one can define a coordinate chart φ that extracts a subset of non-redundant real components of $\tilde{\mathbf{x}}$. This map is defined as (see also Equation (12) for the same definition used in a different context):

$$\varphi[\tilde{\mathbf{x}}] = \begin{cases} (\Re[\tilde{\mathbf{x}}_\kappa])_{\kappa=0}^{N/2} \oplus (\Im[\tilde{\mathbf{x}}_\kappa])_{\kappa=1}^{N/2-1} & \text{if } N \text{ is even} \\ (\Re[\tilde{\mathbf{x}}_\kappa])_{\kappa=0}^{\lfloor N/2 \rfloor} \oplus (\Im[\tilde{\mathbf{x}}_\kappa])_{\kappa=1}^{\lfloor N/2 \rfloor} & \text{else,} \end{cases}$$

where $\mathbf{v}_1 \oplus \mathbf{v}_2$ denotes the concatenation of two vectors $\mathbf{v}_1 \in \mathbb{R}^{d_1}$ and $\mathbf{v}_2 \in \mathbb{R}^{d_2}$, with $d_1, d_2 \in \mathbb{N}$. As we can see, this map truncates the full frequency representation by extracting half of the real components in $\tilde{\mathbf{x}} \in \mathbb{C}^{d_X}$. We deduce that this maps to $\mathbb{R}^{d_X}$ and, therefore, $\varphi[\tilde{\mathbf{x}}]$ has the same number of real components as $\mathbf{x}$. By virtue of the mirror symmetry, $\tilde{\mathbf{x}}$ can be reconstructed from $\varphi[\tilde{\mathbf{x}}]$ by applying the inverse map

$$\varphi^{-1}[\mathbf{z}] = \begin{cases} (\mathbf{z}_0) \oplus (\mathbf{z}_\kappa + i \cdot \mathbf{z}_{N/2+\kappa})_{\kappa=1}^{N/2-1} \oplus (\mathbf{z}_{N/2}) \oplus (\mathbf{z}_{N/2-\kappa} - i \cdot \mathbf{z}_{N-\kappa})_{\kappa=1}^{N/2-1} & \text{if } N \text{ is even} \\ (\mathbf{z}_0) \oplus (\mathbf{z}_\kappa + i \cdot \mathbf{z}_{\lfloor N/2 \rfloor+\kappa})_{\kappa=1}^{\lfloor N/2 \rfloor} \oplus (\mathbf{z}_{\lceil N/2 \rceil-\kappa} - i \cdot \mathbf{z}_{N-\kappa})_{\kappa=1}^{\lfloor N/2 \rfloor} & \text{else.} \end{cases}$$

One can check that the mirror symmetry permits to reconstruct the initial frequency representation via $\tilde{\mathbf{x}} = \varphi^{-1} \circ \varphi[\tilde{\mathbf{x}}]$. By exploiting this truncation with the coordinate chart φ, the implementation of frequency diffusion can be described as follows:

1. Bring the training time series $\mathcal{D}_{\mathrm{train}} \subset \mathbb{R}^{d_X}$ to the frequency domain by computing the DFT of each time series $\tilde{\mathcal{D}}_{\mathrm{train}} \leftarrow \mathcal{F}[\mathcal{D}_{\mathrm{train}}] \subset \mathbb{C}^{d_X}$.
2. Perform a truncation of the frequency representation to extract non-redundant components $\tilde{\mathcal{D}}^{\varphi}_{\mathrm{train}} \leftarrow \varphi(\tilde{\mathcal{D}}_{\mathrm{train}}) \subset \mathbb{R}^{d_X}$.
3. Train a frequency diffusion model by using denoising score matching on $\tilde{\mathcal{D}}^{\varphi}_{\mathrm{train}}$, using the diffusion process defined in Equation (7) and Equation (8).
4. Generate truncated samples $\tilde{\mathcal{S}}^{\varphi}_{\mathrm{freq}} \subset \mathbb{R}^{d_X}$ with the learned score model.
5. Compute the full frequency representation of these samples by using the inverse coordinate chart: $\tilde{\mathcal{S}}_{\mathrm{freq}} \leftarrow \varphi^{-1}(\tilde{\mathcal{S}}^{\varphi}_{\mathrm{freq}}) \subset \mathbb{C}^{d_X}$.
6. *Optional*: Bring these time series back to the time domain by computing their inverse DFT $\mathcal{S}_{\mathrm{freq}} \leftarrow \mathcal{F}^{-1}[\tilde{\mathcal{S}}_{\mathrm{freq}}] \subset \mathbb{R}^{d_X}$.

In Appendix A, we provide a detailed description on how this practical implementation relates to the theoretical discussion in Section 3. Since this implementation performs the frequency diffusion in $\mathbb{R}^{d_X}$, we deduce that it does not lead to additional computational costs compared to diffusing in the time domain (computing the DFT and its inverse using the Fast Fourier transform takes a negligeable amount of time compared to training / sampling from the diffusion model).

For transparency, we have also timed the training time of a few diffusion models trained in the time and frequency domain. These results are reported in Table 3. As we can observe, the computational times of the time and frequency models are similar.

Table 3. Training times for time domain and frequency domain models

Dataset	Training time	
	Time domain model	Frequency domain model
MIMIC-III	30m 2s	30m 13s
Nasa-Charge	44m 51s	43m 45s
US-Droughts	2h 1m 30s	2h 1m 33s

B.4. Additional plots

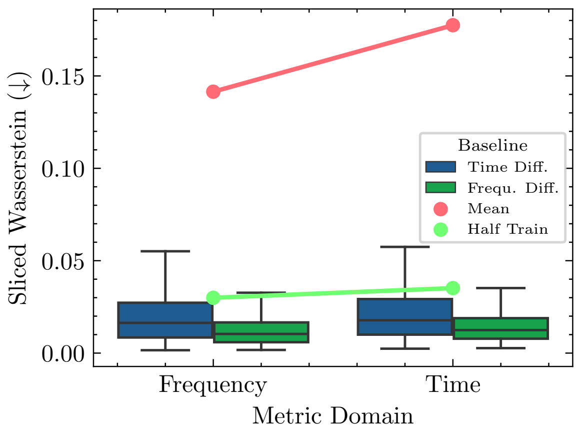
Sliced Wasserstein Distances. In Figure 4, we show the distribution of the sliced Wasserstein distances over all slices. In addition, we have included the average sliced Wasserstein distances obtained with 2 baselines. The first baseline is simply the Wasserstein distance between the training set and a set of sample only containing identical copies the average sample $SW(\mathcal{D}_{\mathrm{train}}, \mathcal{S}_{\mathrm{mean}})$, where $\mathcal{S}_{\mathrm{mean}} = \{\mathbb{E}_{X\sim U(\mathcal{D}_{\mathrm{train}})}[X]\}$. It represents the performance of a dummy generator that only generates the average time series and is denoted by *mean* in Figure 4. The second baseline is the Wasserstein distance between two random splits of the training set $SW(\mathcal{D}^{1/2}_{\mathrm{train}}, \mathcal{D}^{2/2}_{\mathrm{train}})$, where $\mathcal{D}_{\mathrm{train}} = \mathcal{D}^{1/2}_{\mathrm{train}} \bigsqcup \mathcal{D}^{2/2}_{\mathrm{train}}$ is a decomposition of the training set into two disjoint random splits of equal size $|\mathcal{D}^{1/2}_{\mathrm{train}}| = |\mathcal{D}^{2/2}_{\mathrm{train}}|$. It represents the distance between two samples from the ground-truth distribution and is denoted by *self* in Figure 4. As we can observe, both the time and frequency diffusion models substantially outperform the *mean* baseline (as expected) and perform on par with the *self* baseline. This indicates that the models learned a good approximation of the real distribution. Furthermore, we notice that the frequency diffusion models tend to have smaller quantiles than the time diffusion models. This confirms that frequency diffusion models outperform the time diffusion models, as discussed in Section 4.

Marginal Wasserstein Distances. In Figure 5, we show the distribution of the marginal Wasserstein distances over all slices. In addition, we have included the average marginal Wasserstein distances obtained with the 2 baselines defined in the previous paragraph. Again, both the time and frequency diffusion models tend to outperform the *mean* baseline (as expected) and perform on par with the *self* baseline. Furthermore, we notice that the frequency diffusion models tend to have smaller quantiles than the time diffusion models. This is consistent with the observations made in the above paragraph.

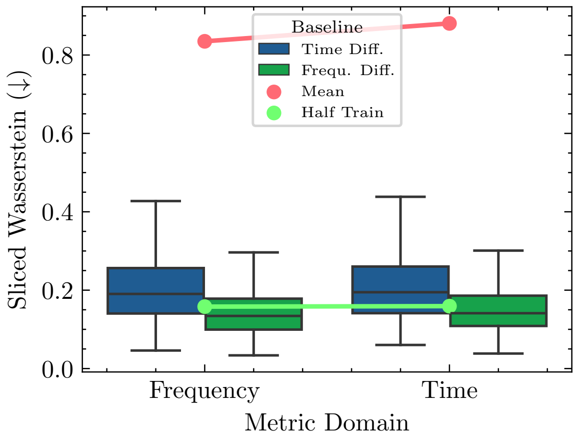
Per-Sample Localization. In Figure 6, we observe the distribution of our localization metrics Δ_{time} and Δ_{sigma} for each sample and each dataset from Section 4. We notice that most samples are located below the $y = x$ axis, which confirms the fact that most samples are more localized in the frequency domain. Interestingly, we also observe that none of the samples is located close to the origin. This confirms the uncertainty theorem from (Nam, 2013).

Empirical Evidence for Localization Claim. We repeat the same experiment as in Section 4.3 with the datasets which have the least extreme imbalance in time and frequency localization, namely *MIMIC-III*, *Nasa-Charge* and *US-Droughts*. In this experiment, we increase the delocalization of the frequency representations of the original samples by convolving them with several Gaussian kernels with various widths.

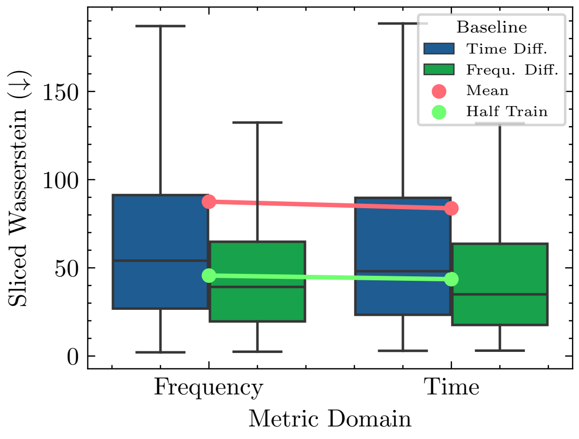
We report these results in Figure 7. These results show that increasing the delocalization of the frequency representations decreases the delocalization in the time domain (in agreement with the uncertainty principle from (Nam, 2013)). For MIMIC-III and US-Droughts, this leads to better time diffusion models, compared to their frequency counterparts. This observation directly mirrors the conclusion given in the case of the ECG example in Section 4.3. It is worth noting that in the case of the Nasa-Charge dataset, the frequency diffusion model is better than the time diffusion model for the different levels of smoothing considered. This corroborates the cautionary remark included in Section 4.3: while localization is a key element which can explain the superior performance of frequency diffusion models, it may not be the only explanation for this phenomenon.



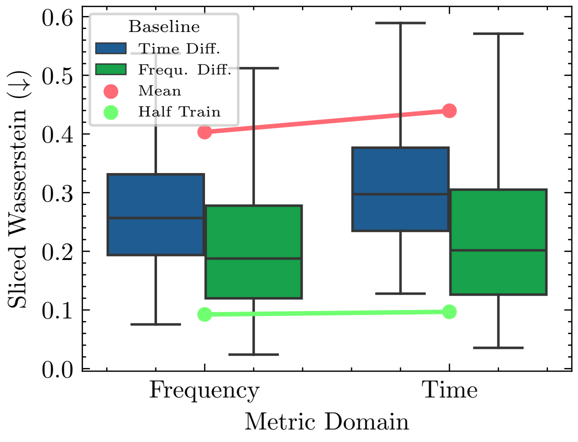
(a) ECG.



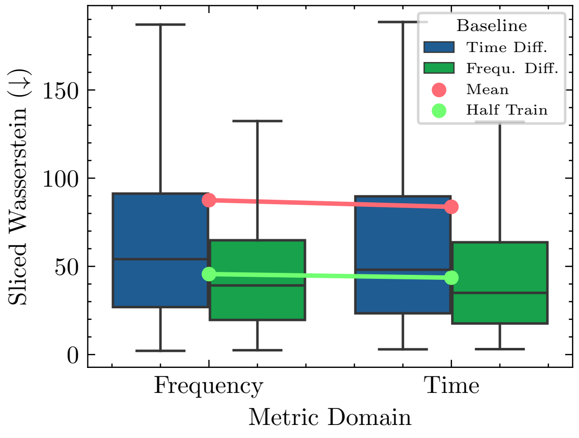
(b) MIMIC-III.



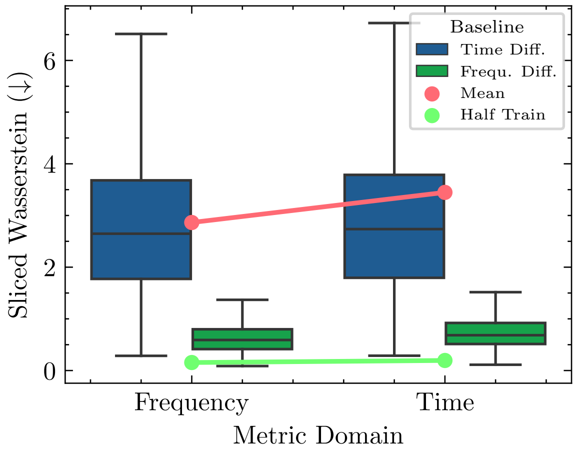
(c) NASDAQ-2019.



(d) NASA-Charge.

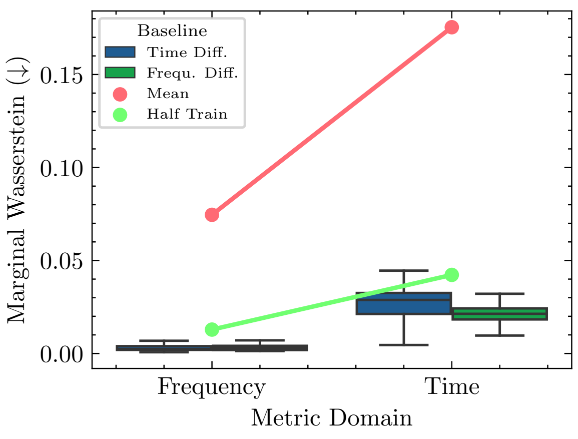


(e) NASA-Discharge.

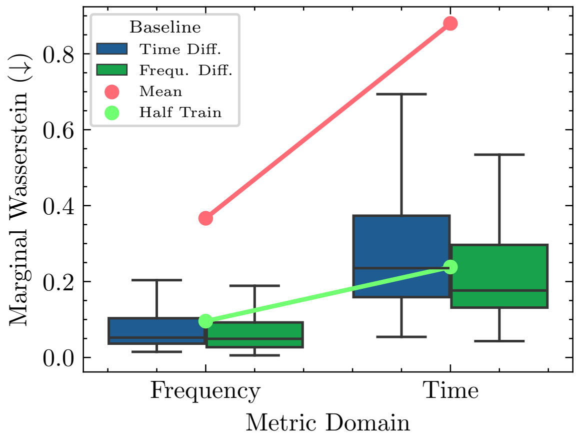


(f) US-Droughts.

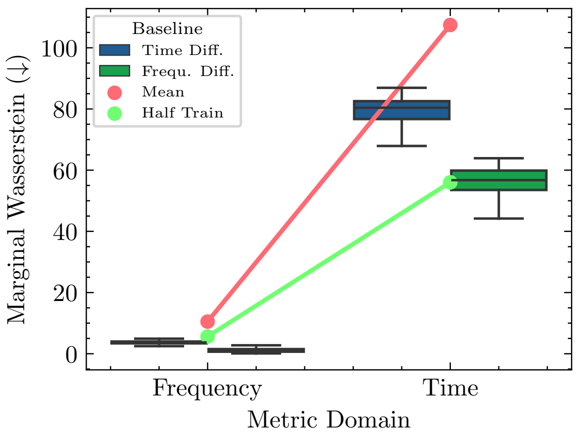
Figure 4. Sliced Wasserstein distances of time and frequency diffusion models.



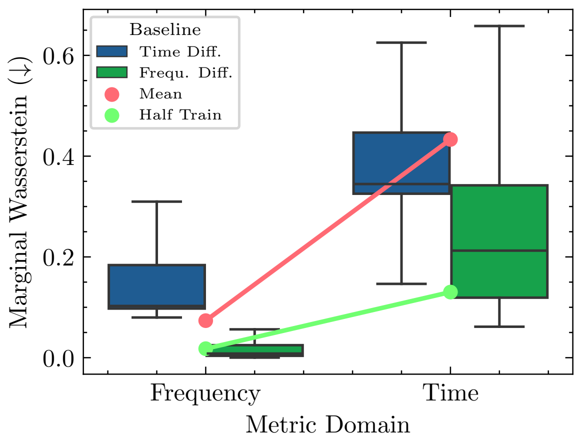
(a) ECG.



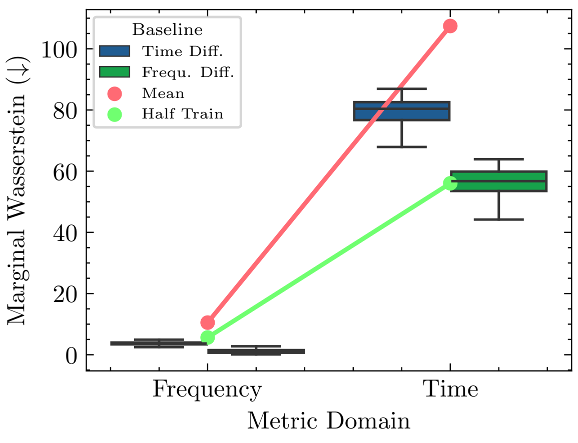
(b) MIMIC-III.



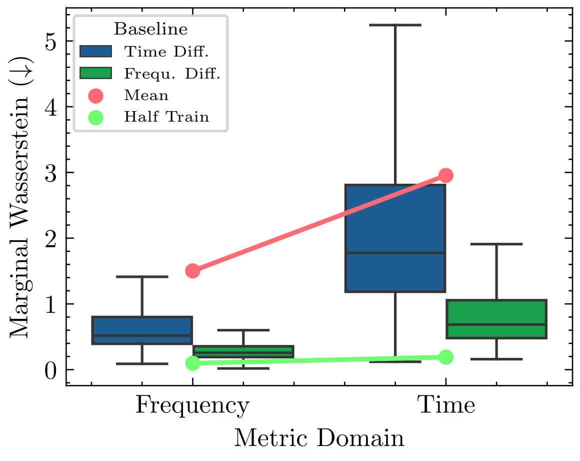
(c) NASDAQ-2019.



(d) NASA-Charge.



(e) NASA-Discharge.



(f) US-Droughts.

Figure 5. Marginal Wasserstein distances of time and frequency diffusion models.

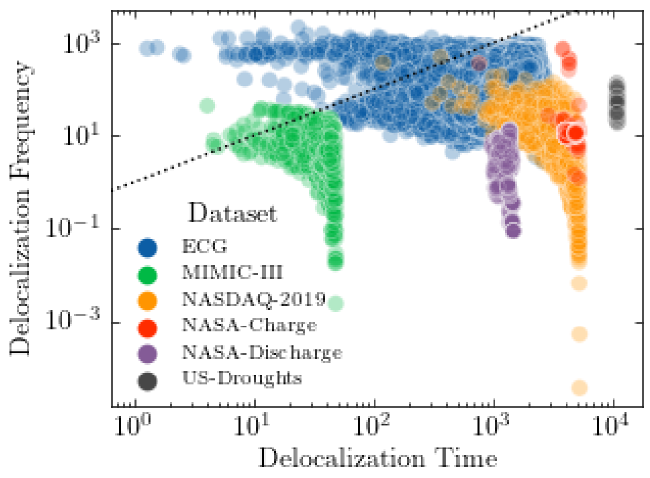


Figure 6. Localization metrics Δ_{time} and Δ_{freq} for all the samples of all datasets. We observe that no sample has a high localization (i.e. low Δ) in the time and frequency domain simultaneously.

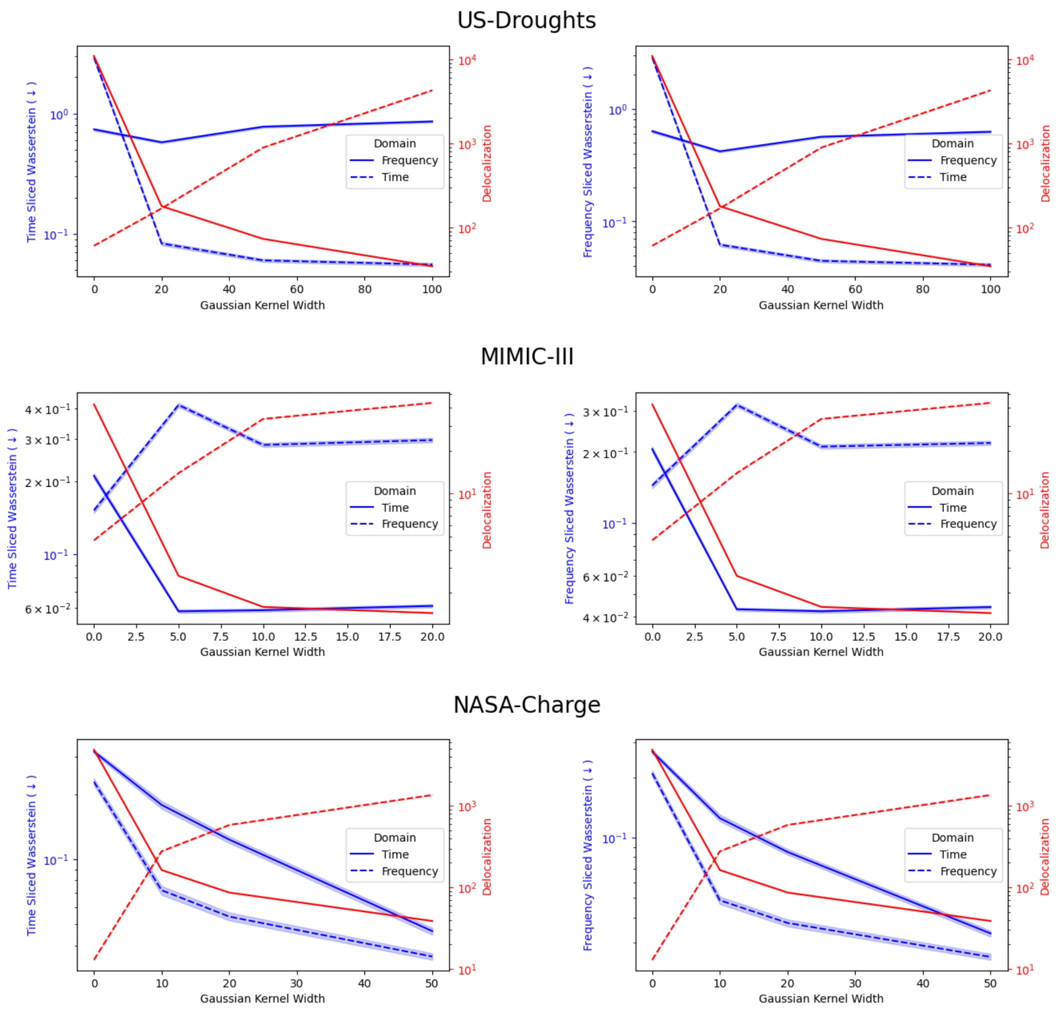


Figure 7. Additional results on the links between performance and localization.

C. Other Distance Metric

While the experimental results in Section 4 were reported with Sliced Wasserstein distances (see Appendix B.2 for details), which permits to quantify the distance between two probability distributions, more specialized metrics exist for time series. Indeed, the *dynamic time warping* (DTW) distance (Müller, 2007) permits to compare the dissimilarity $\mathrm{DTW}(\mathbf{x}, \mathbf{x}')$ between two time series $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_X}$. While this metric is not designed to compare two distributions of time series, it can be aggregated over all the pairs of time series we wish to compare. In our case, we want to compare the training set $\mathcal{D}_{\mathrm{train}} \subset \mathbb{R}^{d_X}$ with the samples $\mathcal{S} \subset \mathbb{R}^{d_X}$ generated by a diffusion model:

$$\overline{\mathrm{DTW}} := \frac{1}{|\mathcal{D}_{\mathrm{train}}|} \sum_{\mathbf{x} \in \mathcal{D}_{\mathrm{train}}} \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x}' \in \mathcal{S}} \mathrm{DTW}(\mathbf{x}, \mathbf{x}')$$

A clear limitation of this metric is that it needs to be evaluated on all possible pairs, which typically represents $\mathcal{O}(10^8)$ pairs when we generate 10000 samples for a dataset of comparable size. This is prohibitively expensive for some of our datasets where time series have many time steps (for instance, this metric would take ~ 7.9 days to evaluate on the NASA dataset). To make the computation time tractable, we resort to a Monte Carlo evaluation by sampling uniformly each pair of samples:

$$\widehat{\mathrm{DTW}} := \mathbb{E}_{\mathbf{x} \sim U(\mathcal{D}_{\mathrm{train}}), \mathbf{x}' \sim U(\mathcal{S})} \left[\mathrm{DTW}(\mathbf{x}, \mathbf{x}') \right].$$

For each diffusion model and each dataset, we compute this mean on 1000 such random pairs of samples. We report the results in Table 4 (computed mean $\pm$ 2 standard errors).

Table 4. Comparison of Frequency and Time Diffusion $\widehat{\mathrm{DTW}}$ across the different datasets

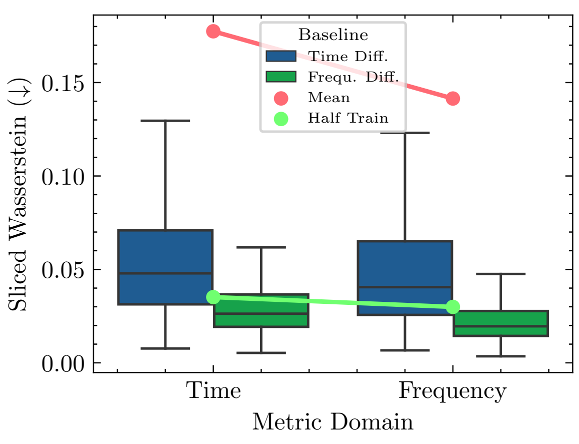
Dataset	**Frequency Diffusion** $\widehat{\mathrm{DTW}}$ ($\downarrow$)	**Time Diffusion** $\widehat{\mathrm{DTW}}$ ($\downarrow$)
ECG	$\mathbf{1.166 \pm 0.034}$	1.179 ± 0.036
MIMIC-III	$\mathbf{35.828 \pm 0.578}$	36.136 ± 0.76
NASA-Charge	$\mathbf{8.203 \pm 1.04}$	11.556 ± 0.954
NASA-Discharge	$\mathbf{138.993 \pm 6.71}$	175.405 ± 8.656
NASDAQ-2019	$\mathbf{1857.703 \pm 254.372}$	2236.094 ± 262.882
US-Droughts	$\mathbf{274.9 \pm 5.206}$	399.342 ± 12.794

These results lead to the same conclusions as the one stated in Section 4: frequency diffusion models outperform time diffusion models. We note that these metrics are considerably more computationally expensive than the Wasserstein metrics. Indeed, the computation of these metrics typically takes several minutes, while the sliced Wasserstein distances can be computed in a few seconds.

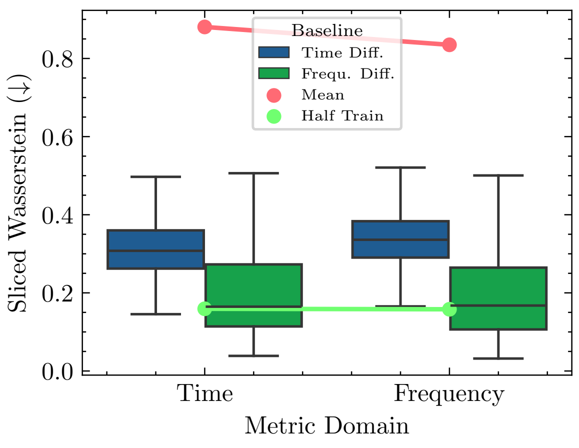
D. Alternative Backbone

LSTM Models. For each dataset, we try an alternative parametrization of the time score model $\mathbf{s}_\theta$ and the frequency score model $\tilde{\mathbf{s}}_{\tilde{\theta}}$ as LSTM encoders with 10 layers, each with dimension $d_{\mathrm{model}} = 72$. Both models have diffusion time t encoding through random Fourier features composed with a learnable dense layer. This results in models with 427k parameters. The data is noised by using a VP-SDE, as in (Song et al., 2020). The score models are trained with the denoising score-matching loss, as defined in Section 3. All the models are trained for 200 epochs with batch size 64, AdamW optimizer and cosine learning rate scheduling (20 warmup epochs, $\mathrm{lr}_{\max} = 10^{-3}$). The selected model achieves the lowest validation loss.

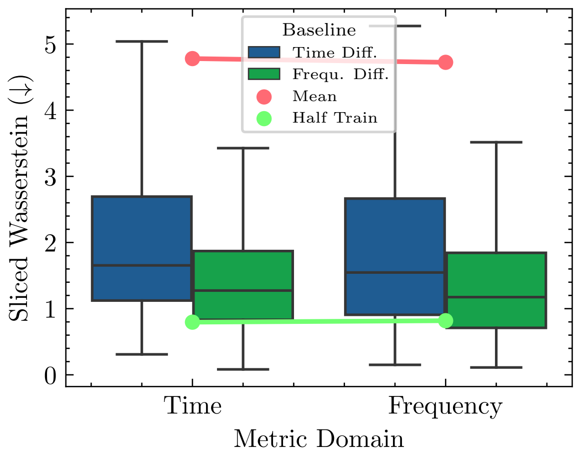
Sliced Wasserstein Distances. In Figure 8, we show the distribution of the sliced Wasserstein distances over all slices for the LSTM models. In addition, we have included the average sliced Wasserstein distances obtained with 2 baselines defined in Appendix B.4. As observed for the transformer models, both the time and frequency diffusion models substantially outperform the *mean* baseline (as expected) and perform on par with the *self* baseline. This indicates that the models learned a good approximation of the real distribution. Furthermore, we notice that the frequency diffusion models tend to have smaller quantiles than the time diffusion models. This confirms that frequency diffusion models outperform the time diffusion models, as observed for the transformer models in Section 4. We note that the Nasa-Charge and the NASDAQ-2019



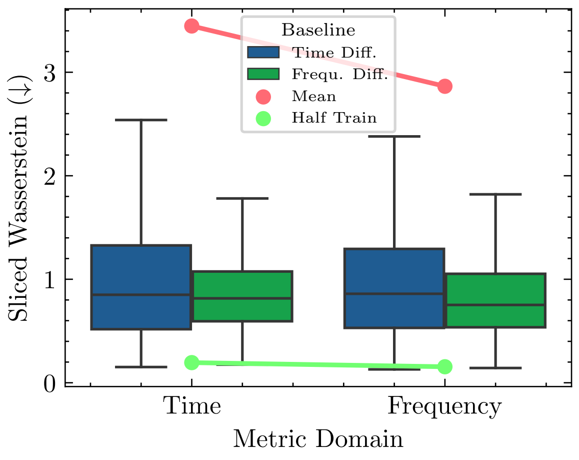
(a) ECG.



(b) MIMIC-III.



(c) NASA-Discharge.



(d) US-Droughts.

Figure 8. Sliced Wasserstein distances of time and frequency LSTM models.

are absent from Figure 8. This is because we did not manage to obtain diffusion models performing better than the *mean* baseline for these datasets, hence leading to non informative comparisons between models with poor performances.

Other Attempts. In order to minimize the inductive bias in our models, we also tried to train diffusion models with simple feed-forward neural networks. Unfortunately, this attempt was unsuccessful and resulted in models performing worse than the *mean* baseline in each case. This emphasizes the value of incorporating inductive biases in time series diffusion models.

E. Comparison with TimeGAN

We compare the performance of our diffusion models with TimeGAN models (Yoon et al., 2019), which are Generative Adversarial Networks specifically designed for time series data. Following Section 4.1 in our paper, we report the sliced Wasserstein distances of the samples generated by TimeGAN models (trained with the hyperparameters used by the authors) and compare them to our diffusion models. We report in Table 5 the distance as its mean $\pm$ 2 standard errors.

Table 5. Comparison with TimeGAN

Dataset	TimeGAN Wasserstein ($\downarrow$)	Frequency Diffusion Wasserstein ($\downarrow$)	Time Diffusion Wasserstein ($\downarrow$)
ECG	0.72 ± 0.031	$\mathbf{0.015 \pm 0.000}$	0.021 ± 0.000
MIMIC-III	0.88 ± 0.0071	$\mathbf{0.152 \pm 0.004}$	0.211 ± 0.006
NASDAQ-2019	89 ± 4.25	$\mathbf{43.602 \pm 2.044}$	60.512 ± 2.960
Nasa-Charge	1.9 ± 0.087	$\mathbf{0.229 \pm 0.008}$	0.316 ± 0.008
Nasa-Discharge	10.0 ± 0.47	$\mathbf{2.028 \pm 0.082}$	2.942 ± 0.134
US-Droughts	23.0 ± 1.0	$\mathbf{0.738 \pm 0.020}$	2.913 ± 0.092

As we can observe, the diffusion models significantly outperform GANs, even after performing a grid search for TimeGAN for a few datasets. Hence, the differences can be attributed to the type of generative model used (i.e., diffusion vs GAN) rather than its architecture. This further reinforces our motivation to focus on diffusion models, which seem to offer better performances.

F. Forecasting Use Case

The diffusion models we study in this work approximate a *joint* distribution $P(\mathbf{x}_0, ..., \mathbf{x}_{N-1})$. We note that modelling the joint distribution permits to derive other quantities, such as conditionals. For example, given a partition $I \sqcup J = [N]$, we can infer $P(\mathbf{x}_I | \mathbf{x}_J)$ from the joint distribution, where $\mathbf{x}_I$ denotes the restriction of $\mathbf{x} \in \mathbb{R}^{d_X}$ to the indices in the set I. We can then use the infered conditionals for practical tasks, such as forecasting.

Motivated by this insight, we conducted an additional experiment to illustrate the use of our frequency diffusion models for *forecasting*, in a data augmentation setting. For each of the datasets investigated in our manuscript, we use the synthetic time series $\mathcal{S}$ generated by the diffusion model trained in the frequency domain and augment the training set D_{train} with $\mathcal{S}$ to obtain the augmented dataset $D_{\text{aug}} = D_{\text{train}} \cup \mathcal{S}$.

The forecasting task requires the definition of a forecast window, which is the number of time steps for which we would like to predict the values of the time series. Let N be the sequence length of the samples in D_{train}, M be the number of features, and W be the size of the forecast window. For the ECG dataset, we let $W = \frac{N}{2}$ (because the electrocardiograms are zero-padded in the last time steps, which makes the forecasting task too easy if we use a small W). For the other datasets, we let $W = \frac{N}{4}$. We split every time series $\mathbf{x}$ of sequence length N present in the training set into an observable part $\mathbf{x}_{[N-W]}$ (which will be used as input to our forecasting model) and a target $\mathbf{x}_{[N] \setminus [N-W]}$ (to be predicted by the forecasting model).

We train a forecasting model by using an LSTM backbone, with 2 layers and a hidden dimension of 256. The last hidden state of the LSTM is used as input to a linear layer of output size $W \times M$, whose output is reshaped to obtain time series of dimension (W, M). The forecasting models are trained for 50 epochs, and we use early stopping based on a validation set, with a train/validation set ratio of 0.8. We then evaluate the forecasting performance by computing the test Mean Absolute Error (MAE) (the test set being held-out time series from the original datasets that are *not* observed by the diffusion model). We report the results in Table 6 (mean $\pm$ 2 $\times$ standard errors for 5 seeds).

The results show the potential of diffusion-based generative modelling in a data augmentation scenario, as we improve forecasting performance for half the datasets by augmenting D_{train} with time series generated by the diffusion model.

Table 6. Forecasting performance for different datasets when training on D_{train} and D_{aug}.

Dataset	**Train on D_{train}Test MAE (↓)**	**Train on D_{aug}Test MAE (↓)**
ECG	0.046 ± 0.0039	$\mathbf{0.035 \pm 0.0009}$
MIMIC-III	0.186 ± 0.001	$\mathbf{0.179 \pm 0.001}$
NASDAQ-2019	3.058 ± 0.65	$\mathbf{2.614 \pm 0.20}$
Nasa-Charge	$\mathbf{0.019 \pm 0.001}$	0.033 ± 0.002
Nasa-Discharge	$\mathbf{0.21 \pm 0.02}$	0.28 ± 0.02
US-Droughts	$\mathbf{0.60 \pm 0.02}$	0.76 ± 0.01

G. Sample Visualization

In Figures 9 to 13, we visualize a few examples generated by each diffusion model, along with ground-truth training examples. We do not include samples from the MIMIC-III dataset in accordance with the dataset licence. We observe that the frequency diffusion models generate samples that are substantially less noisy than than the ones generated by time diffusion models. All the generated samples resemble training samples, with the only exception of the NASDAQ-2019 dataset. The models appear to be struggling with the high correlation between the different features.

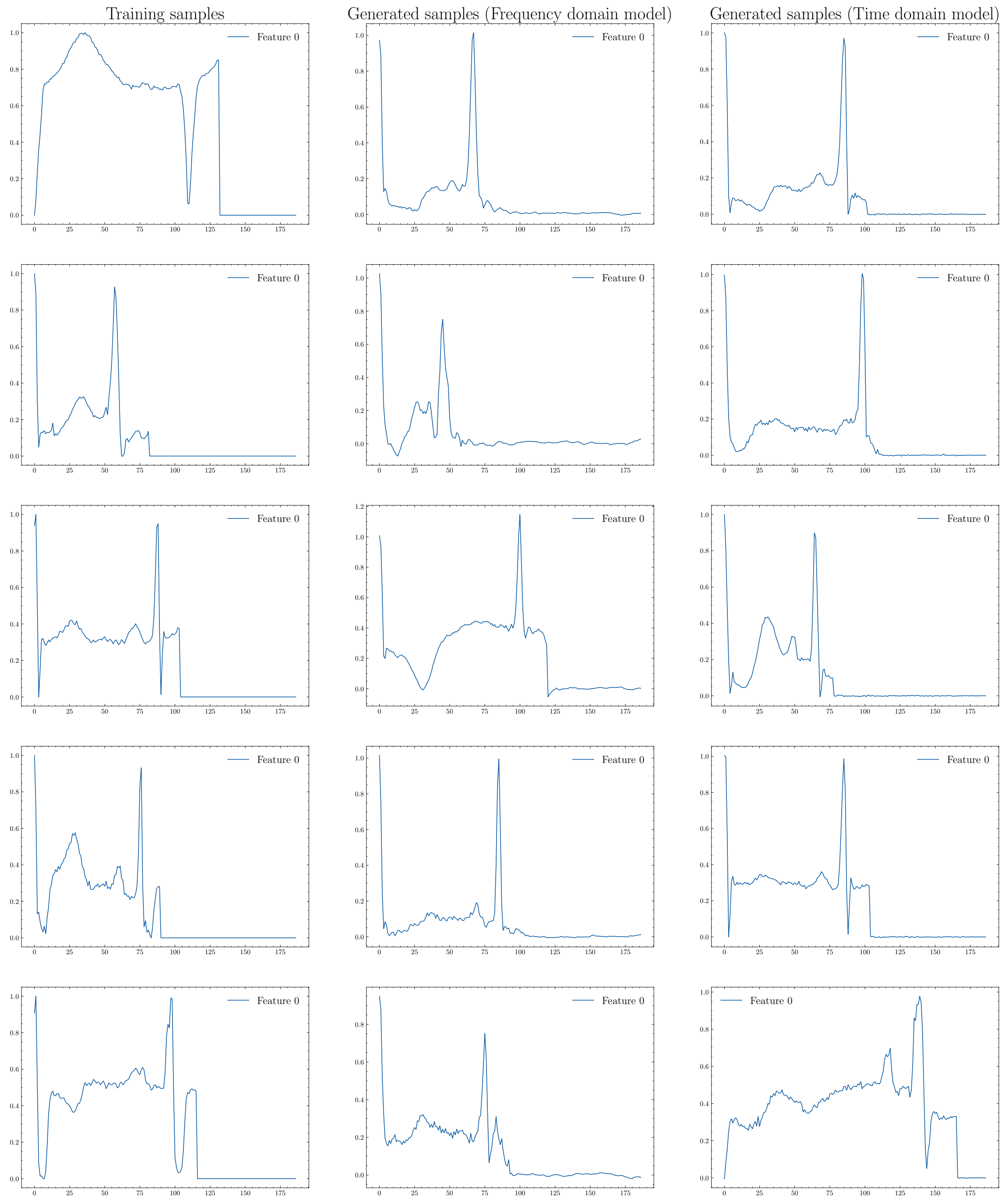


Figure 9. Samples for the ECG dataset. The y axis corresponds to the different features, while the x axis corresponds to the different time steps.

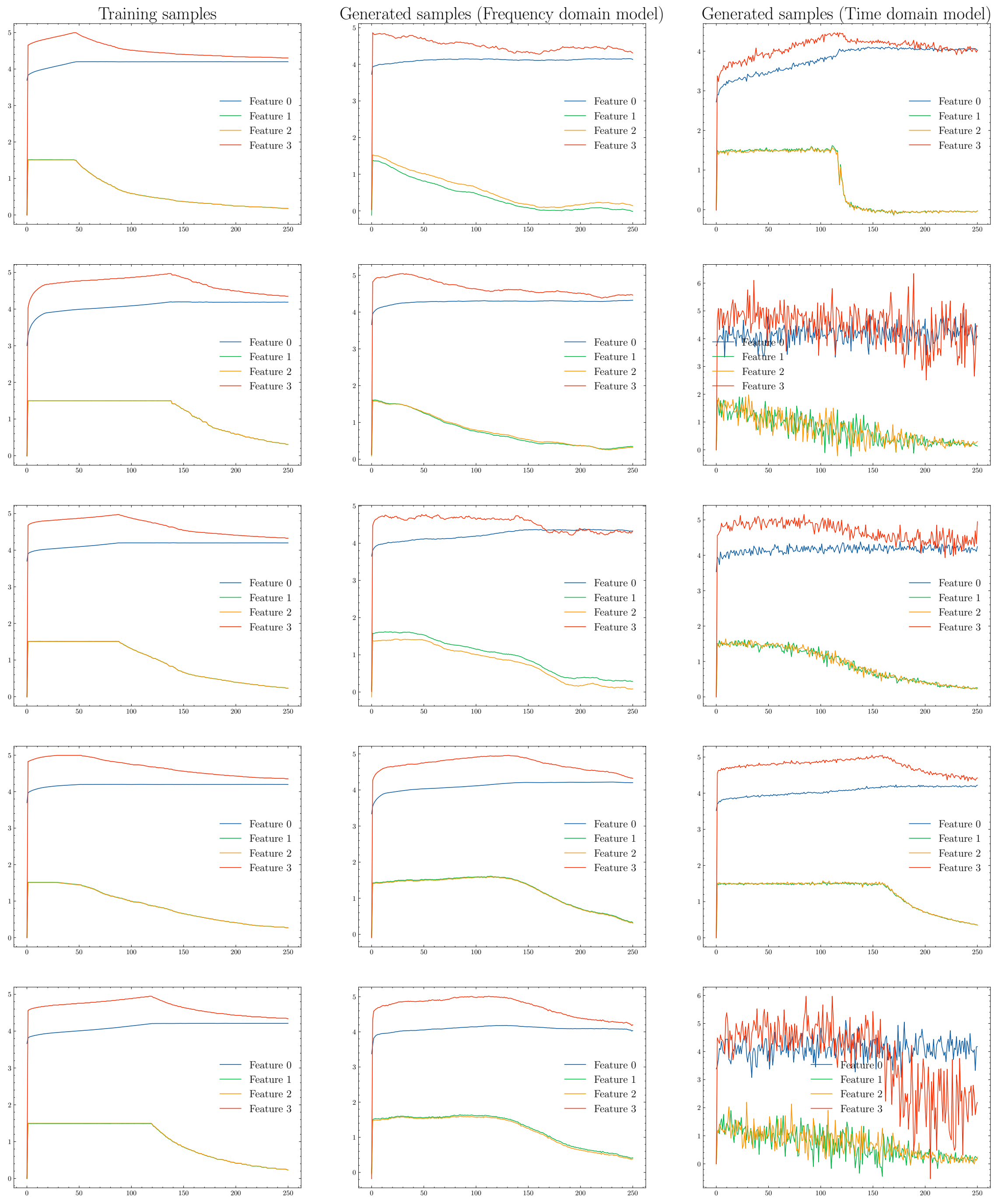


Figure 10. Samples for the NASA-Charge dataset. The y axis corresponds to the different features, while the x axis corresponds to the different time steps.

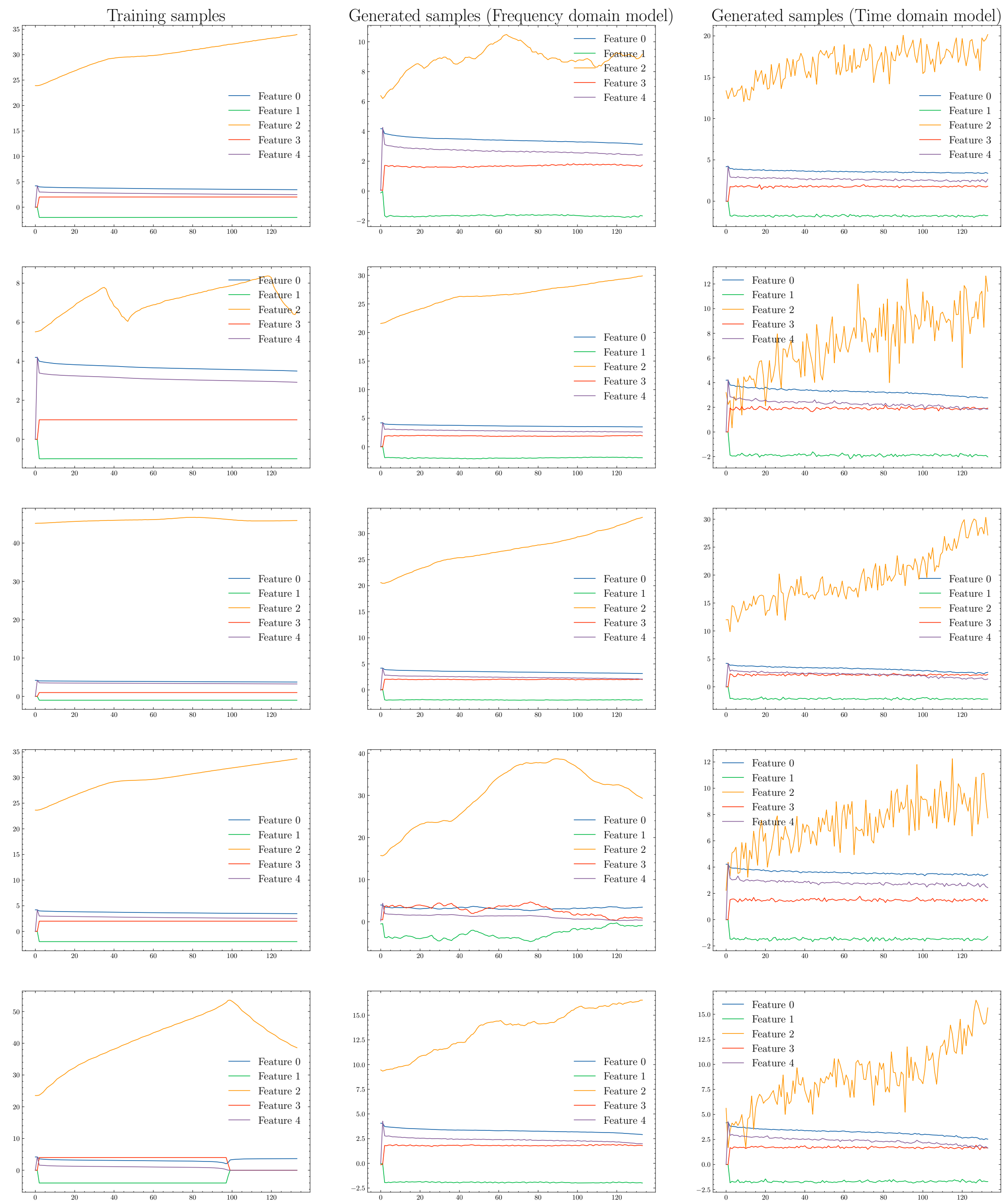


Figure 11. Samples for the NASA-Discharge dataset. The y axis corresponds to the different features, while the x axis corresponds to the different time steps.

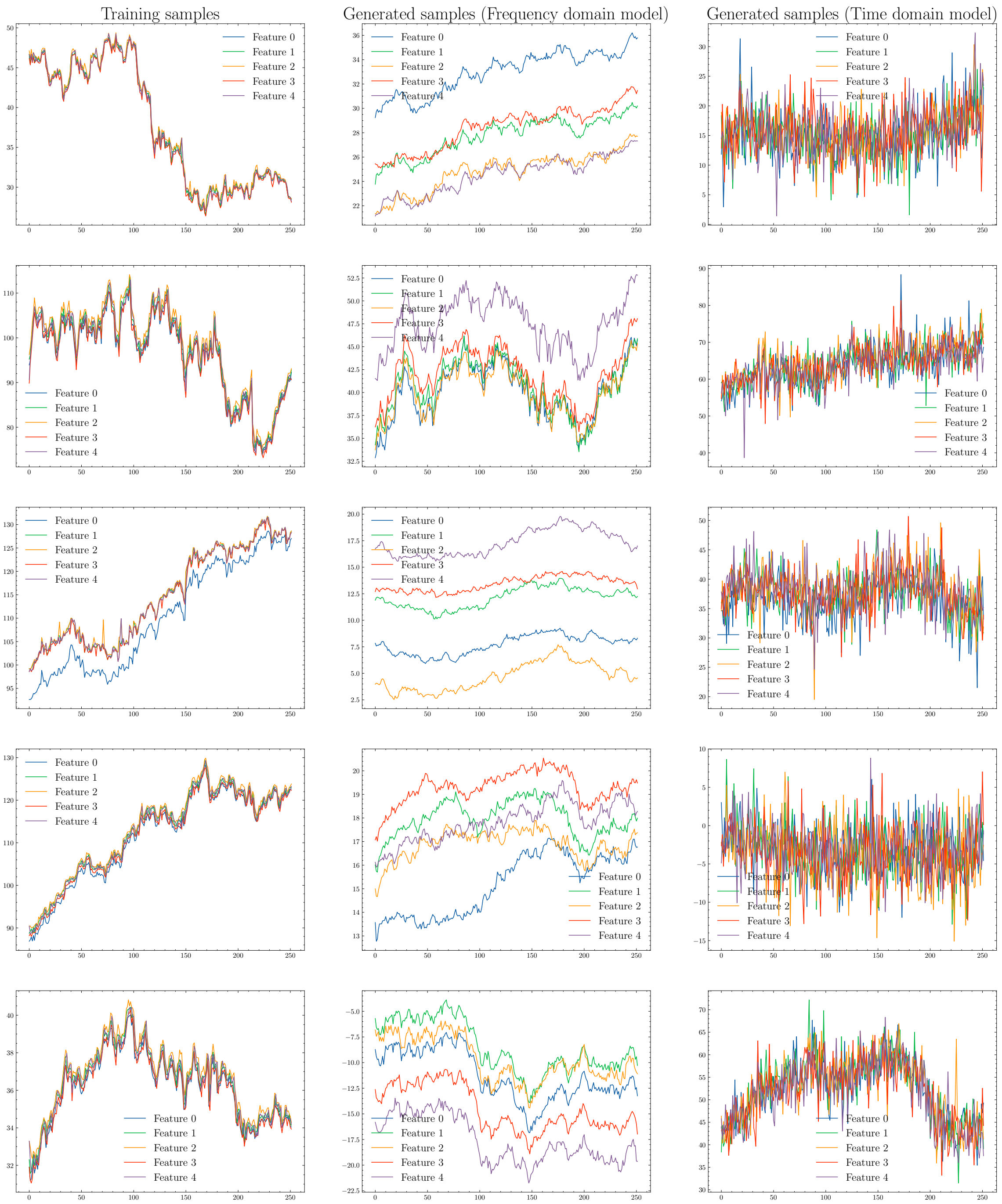


Figure 12. Samples for the NASDAQ-2019 dataset. The y axis corresponds to the different features, while the x axis corresponds to the different time steps.

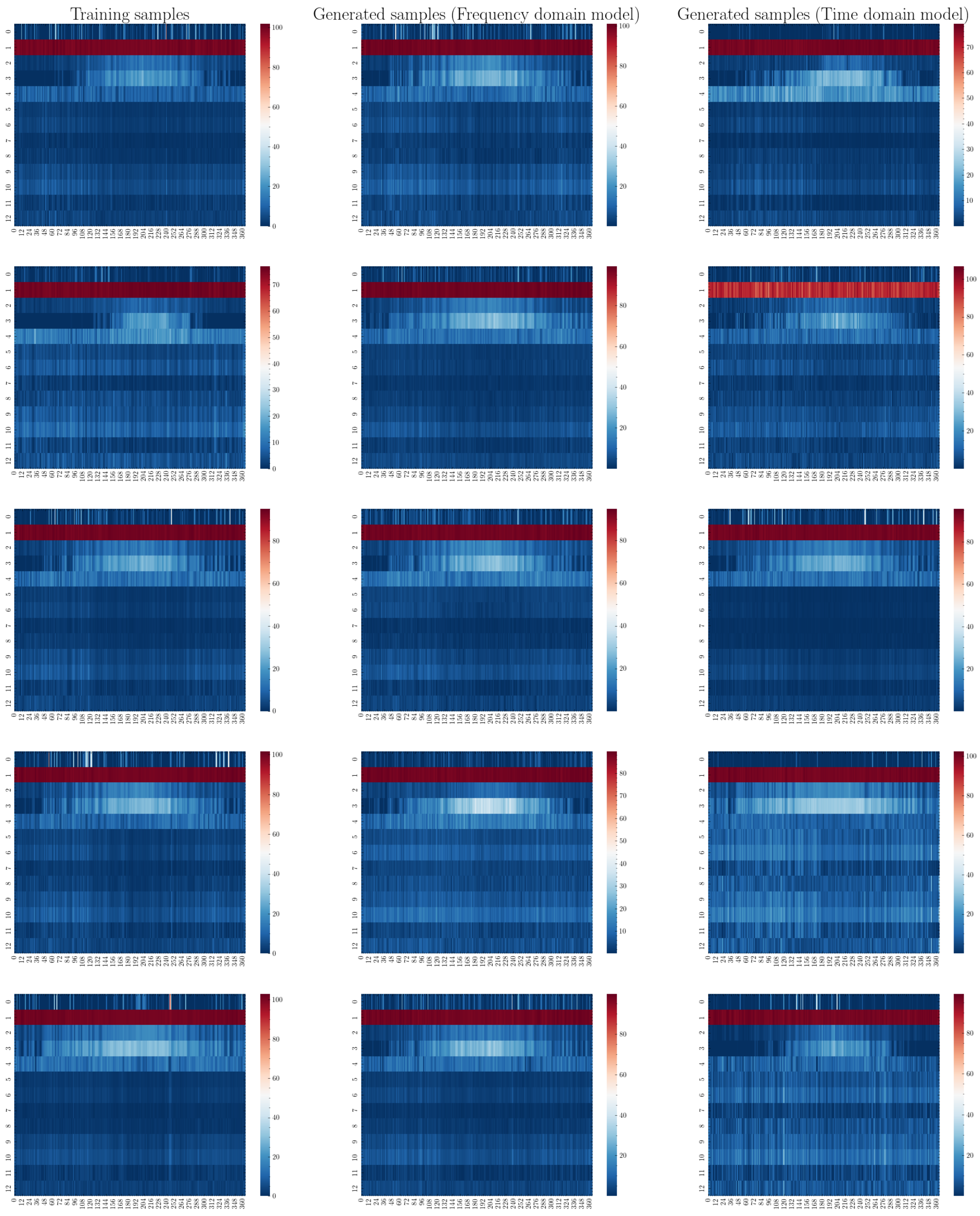


Figure 13. Samples for the US-Droughts dataset, represented as heatmaps. The y axis corresponds to the different features, while the x axis corresponds to the different time steps.