A synthetic dataset for Time Series Super-Resolution with Deep Learning

Julio Ibarra-Fiallo[1], Juan A. Lara[2], and D'hamar Agudelo-Moreno[1]

[1]Colegio de Ciencias e Ingenierías, Universidad San Francisco de Quito, Cumbayá, Ecuador
[2]Universidad de Córdoba, Córdoba, España
[*]corresponding author: Julio Ibarra-Fiallo (jibarra@usfq.edu.ec)

ABSTRACT

The increasing application of time-series analysis in fields such as biomedical engineering, telecommunications, and industrial monitoring emphasizes the need for high-quality datasets to develop, compare, and validate data-driven methods.~~The increasing application of temporal signal analysis in fields like biomedical engineering, telecommunications, and industrial monitoring emphasizes the need for high-quality data to train and evaluate advanced machine learning models.~~[REV] Acquiring real-world temporal data at suitable resolutions is often limited by ethical, economic, or practical constraints. To address this, we introduce CoSiBD (Complex Signal Benchmark Dataset for Super-Resolution), a synthetic dataset of complex temporal signals designed to support reproducible research in multi-resolution time-series analysis, including temporal super-resolution and related signal processing tasks.~~designed for training and assessing AI models, particularly deep learning systems, in tasks like temporal super-resolution and signal processing.~~[REV] CoSiBD comprises 2,500 high-resolution signals ($N = 5{,}000$ samples each over a reference domain $\tau \in [0, 4\pi]$), with corresponding low-resolution versions provided at four target sampling levels (150, 250, 500, and 1,000 samples) obtained via uniform decimation of the original sequences.[REV] CoSiBD includes diverse signals with non-uniform frequency modulations, capturing gradual transitions and abrupt high-frequency events to reflect a broad range of non-stationary temporal behaviors.~~to mirror real-world dynamics.~~[REV] The dataset includes clean and noisy variants across all sampling resolutions, enabling systematic benchmarking under controlled variability conditions.~~It offers signals at multiple resolutions with varying noise levels, enabling robust evaluation of model performance under realistic conditions, especially for super-resolution tasks.~~[REV] The dataset is generated by combining distinct frequency bands, non-uniform intervals, and probabilistic frequency assignments to create realistic patterns, with smoothing achieved through spline interpolation. Technical validation focuses on the spectral characteristics of the generated signals across sampling resolutions and noise settings, documenting consistency and controlled variability under the reported generation parameters.~~Validated for spectral consistency across sampling rates and noise, CoSiBD supports training and evaluation.~~[REV]

Background & Summary

The analysis and simulation of temporal signals are fundamental across science and engineering, ~~supporting insights into dynamic processes.~~[REV] providing critical insights into dynamic processes across multiple domains.[REV] In biomedical research[1], electroencephalography (EEG) and electrocardiography (ECG) analyses reveal brain and heart function[2,3]. Telecommunications rely on signal processing to ensure data fidelity across noisy media[4], while finance uses time-series forecasting for risk and trend analysis[5]. Industrial monitoring detects equipment faults using temporal patterns[6], and environmental science applies similar techniques to ~~climate tracking via remote sensing~~[REV] climate and environmental monitoring using remote-sensor time series[REV][7]. Developing robust tools for interpreting time-varying data continues to support both scientific discovery and practical applications, while increasingly relying on the availability of reliable and well-characterized temporal signal datasets.[REV]

Recent advances in deep learning have contributed significantly to this field by enabling automatic extraction of complex features from raw signals. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), including Long Short-Term Memory (LSTM) units, and Generative Adversarial Networks (GANs) have demonstrated improved performance over traditional techniques in image, speech, and time-series processing tasks[8,9]. These models support fine-grained signal reconstruction and forecasting, allowing researchers to explore temporal dynamics in new ways, but also increasing the demand for well-structured, high-quality temporal signal datasets suitable for training and evaluation.[REV]

Despite this progress, deep learning methods for temporal signal processing often require large quantities of labeled, high-quality data. Access to such data is frequently constrained by medical privacy regulations such as the General Data Protection

Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA)[10]. In other domains, including ~~remote sensing and industrial monitoring~~[REV] environmental monitoring using remote sensors and industrial monitoring[REV], data availability is limited by practical and economic barriers to sensor deployment and data collection[5]. These limitations are particularly relevant in super-resolution (SR) tasks, where models require paired low- and high-resolution signals for effective training, which are rarely available in sufficient quantity and with consistent acquisition conditions.[REV]

Temporal SR, which enhances resolution over time, has broad potential. ~~In medicine, for instance, it improves magnetic resonance imaging (MRI) and computed tomography (CT) scans, supporting earlier disease detection [11].~~[REV] In biomedical monitoring and sensing, temporal SR can help reconstruct higher-resolution physiological time series, such as ECG and EEG signals.[REV] For EEG analysis, SR may help recover high-frequency components that aid in the study of neural oscillations[2] or detect subtle physiological irregularities[3]. ~~In remote sensing, SR helps refine satellite imagery [7],~~[REV] In domains such as environmental sensing, telecommunications, and industrial monitoring,[REV] SR can increase sensitivity to rapid temporal changes,[REV] ~~while in telecommunications it contributes to enhanced signal reliability. It also has applications in industrial monitoring by increasing sensitivity to system changes.~~[REV]

Traditional SR methods such as polynomial interpolation, frequency-domain transforms, and splines each have limitations. Polynomial models are often insufficient for capturing nonlinear dynamics; frequency-domain methods are susceptible to noise[7]; and splines, though flexible, may not generalize well to complex signal variability[12, 13]. Many of these methods also assume uniform partitioning, which may not align with the multi-scale, irregular structure of natural temporal phenomena.

Deep learning offers adaptive alternatives to these traditional methods. CNNs are capable of modeling spatio-temporal structure, RNNs and LSTMs capture long-range dependencies in time, and GANs can learn high-resolution representations through adversarial training[8, 9]. While GANs have achieved strong results in image SR[14], their application to time-series SR remains relatively new. Preliminary work on synthetic time-series generation indicates potential[14], but the lack of accessible, high-quality paired datasets remains a significant barrier to progress.

Synthetic datasets offer one solution to this problem, allowing researchers to design reproducible training environments that reflect the structure and variability of real-world signals. Prior studies have used synthetic data in domains such as fluid dynamics[15], bioimaging[16], and live-cell imaging[17], demonstrating that synthetic approaches can help simulate complexity while avoiding legal and practical restrictions associated with real-world data.

To support research in super-resolution for time-series data, we present the Complex Signal Benchmark Dataset (CoSiBD). CoSiBD is a synthetic dataset composed of time-series signals with variable resolution, frequency characteristics, and noise levels. The dataset is intended to provide a reusable benchmark resource for the development and comparison of super-resolution methods under controlled and reproducible conditions.~~The dataset is intended to provide a resource for training and evaluating SR models under controlled, reproducible conditions.~~[REV] It includes ~~non-uniformly sampled signals,~~[REV] non-stationary, piecewise-structured signals generated via non-uniform interval partitioning with change-points,[REV] multiple levels of resolution and noise, a technical validation suite, and publicly available Python code to facilitate use. ~~CoSiBD has been used in research presented at the International Conference on Signal Processing and Machine Learning and is made available to support further development in deep learning approaches for temporal super-resolution.~~[REV]

Related synthetic time-series resources

Publicly available synthetic resources for temporal signals exist, but they are typically designed for tasks other than time-series super-resolution (SR), or they target a specific domain. In wireless communications, the RadioML family provides large collections of synthetic complex I/Q sequences with varying signal-to-noise ratios and channel impairments, mainly to benchmark automatic modulation classification rather than paired SR reconstruction[18–20]. In biomedical signal processing, physiological simulators such as ECGSYN (ECG) and SEREEGA (EEG) enable controlled generation with tunable morphology, sampling settings, and noise, supporting method development when access to real data is constrained[21–23]. In power systems, LoadGAN provides multi-resolution generation of load time series across sampling rates and time horizons, but it is not distributed as a standardized paired SR benchmark[24]. Domain-specific paired low- and high-resolution training data can also be produced via physical forward modeling, for example low- and high-resolution one-dimensional seismic traces for learning-based resolution enhancement[25].[REV]

Table 1 summarizes these representative resources and highlights a practical gap: while many tools provide synthetic signals, they usually do not jointly offer (i) multi-factor paired low- and high-resolution signals suitable for time-series SR, (ii) a clearly

Resource	Domain	Form	Paired LR–HR SR	Multi-resolution	Noise / artifacts	Reproducibility granularity
CoSiBD (this work)	Generic time series (complex-structured signals)	Dataset + generator	Yes (LR → HR targets)	Yes (150/250/500/1000 → 5000)	Gaussian + structured interference; primary benchmark uses direct decimation	Per-signal metadata; deterministic regeneration (seed-controlled)
RadioML 2016.10A[18,19]	Wireless communications (I/Q)	Dataset	No (classification benchmark)	N/A (not SR)	Variable SNR + channel impairments	Dataset-level (labels/SNR)
RadioML 2018.01A[20]	Wireless communications (I/Q)	Dataset	No (classification benchmark)	N/A (not SR)	Simulated channel effects + SNR variability	Dataset-level
ECGSYN[21,22]	ECG (physiology)	Simulator/tool	Configurable[1]	Configurable	Model-based; supports controlled variability	User-defined simulator parameters
SEREEGA[23]	EEG (physiology)	Simulator/tool	Configurable[1]	Configurable	Supports noise and event-related components	User-defined simulator parameters
LoadGAN[24]	Power systems load time series	Generator/tool	No (generation)	Yes (variable sampling rates)	Domain-specific variability	Configurable generation settings
Synthetic LR–HR seismic traces[25]	Seismic traces (geophysics)	Paper-specific paired data	Yes (LR–HR pairs)	Study-specific	Study-dependent	Study-specific

Table 1. Representative publicly available synthetic time-series datasets and simulators related to signal processing and learning.

specified and reproducible protocol for constructing low-resolution observations aligned to a fixed high-resolution target, and (iii) per-signal metadata enabling deterministic regeneration and principled benchmarking. CoSiBD is designed to address this gap by providing multi-resolution paired signals, explicit nuisance modeling (including noise and structured interference), and comprehensive metadata for reproducible super-resolution benchmarking across multiple difficulty levels.$^{\text{REV}}$

Methods

The methodology used to generate the synthetic temporal signals that constitute the CoSiBD dataset is illustrated in Figure 1. The signal generation process is designed to produce time series exhibiting structural properties commonly observed in real-world temporal data, including variable frequency content, smooth transitions, and intermittent high-frequency activity. A key aspect of the procedure is the generation of signals at multiple temporal resolutions, enabling the construction of paired datasets for super-resolution (SR) benchmarking.

Signal design principles. The CoSiBD signal generator incorporates structural properties commonly observed in physiological and speech time series, including (i) non-stationary regime changes, (ii) coexisting low- and high-frequency components with intermittent transients, (iii) smooth amplitude-envelope evolution, and (iv) baseline drift and measurement noise. These properties are implemented through non-uniform interval partitioning with change-points, separate low- and high-frequency bands, spline-based amplitude envelopes and frequency profiles, and explicit offset and noise terms. Figure 2 illustrates representative examples of these signal characteristics and the corresponding design mechanisms used in CoSiBD. $^{\text{REV}}$

The signal generation pipeline involves the following steps:

1. **Base frequency band definition:** A set of distinct frequency bands is defined to represent the underlying spectral content

[1] "Configurable" indicates that low- and high-resolution signals can be generated or derived by adjusting simulator settings or sampling rates, but a standardized paired super-resolution benchmark is not distributed as part of the resource.

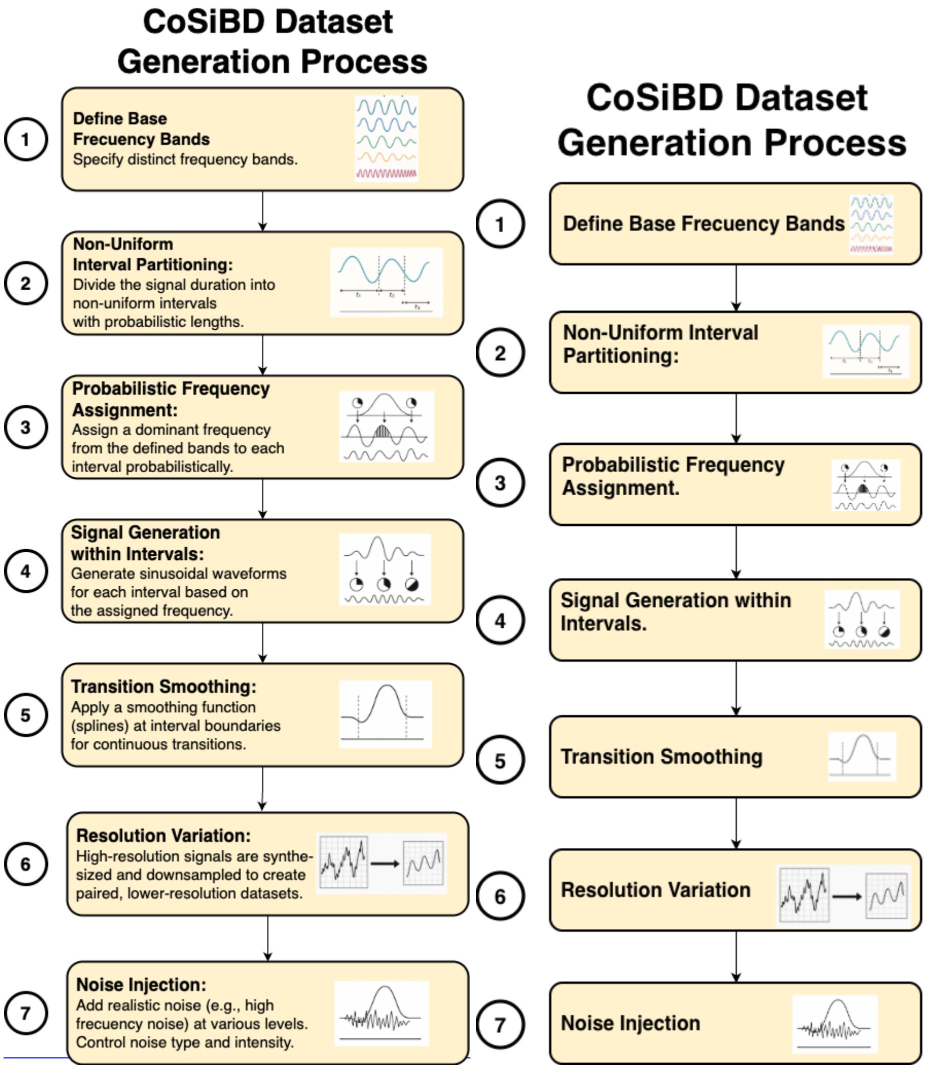
CoSiBD Dataset Generation Process
1
Define Base Frecuency Bands
Specify distinct frequency bands.
2
Non-Uniform Interval Partitioning:
Divide the signal duration into non-uniform intervals with probabilistic lengths.
3
Probabilistic Frequency Assignment:
Assign a dominant frequency from the defined bands to each interval probabilistically.
4
Signal Generation within Intervals:
Generate sinusoidal waveforms for each interval based on the assigned frequency.
5
Transition Smoothing:
Apply a smoothing function (splines) at interval boundaries for continuous transitions.
6
Resolution Variation:
High-resolution signals are synthe-sized and downsampled to create paired, lower-resolution datasets.
7
Noise Injection:
Add realistic noise (e.g., high frecuency noise) at various levels. Control noise type and intensity.
CoSiBD Dataset Generation Process
1
Define Base Frecuency Bands
2
Non-Uniform Interval Partitioning:
3
Probabilistic Frequency Assignment.
4
Signal Generation within Intervals.
5
Transition Smoothing
6
Resolution Variation
7
Noise Injection

Figure 1. Schematic overview of the CoSiBD signal generation process.

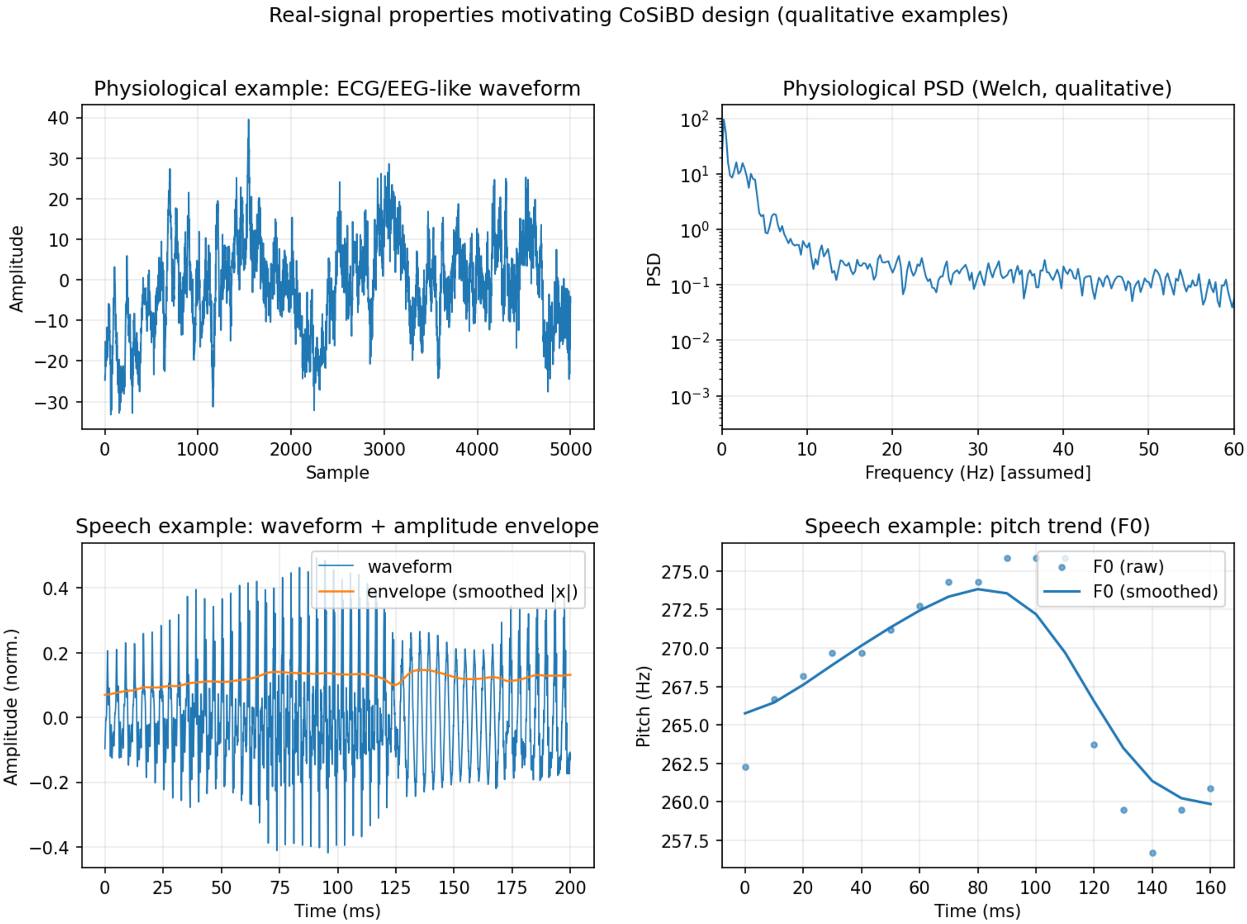

Figure 2. Representative examples of non-stationary temporal properties in physiological and speech signals that informed the CoSiBD design. The figure shows regime changes, structured spectral content, amplitude-envelope dynamics, and smoothly varying frequency trends, which are reflected in the corresponding signal generation mechanisms. [REV]

of the signals. These can be adjusted to reflect application-specific characteristics.

2. **Non-uniform interval partitioning:** The total signal duration is divided into multiple intervals of variable length. The interval lengths are determined probabilistically to introduce variability in the signal structure and non-stationarity through change-points.$^{\text{REV}}$

3. **Frequency assignment:** Each interval is assigned a dominant frequency band, sampled according to a predefined probability distribution. This introduces spectral variation over time.

4. **Signal synthesis:** A sinusoidal waveform, or a combination of sinusoids within the assigned frequency band, is generated for each interval. Signal parameters such as amplitude and phase are configurable.

5. **Transition smoothing:** To avoid discontinuities at interval boundaries, a smoothing function is applied to overlapping segments. This ensures gradual transitions between intervals with different frequency content.

6. **Resolution variation:** All signals are initially synthesized at a high temporal resolution. Lower-resolution versions are created by applying controlled downsampling to the high-resolution signals, forming paired datasets. In CoSiBD, paired low-resolution sequences are obtained via simple uniform decimation (uniform subsampling) of the high-resolution signals. The low-resolution observation is formed by subsampling the original sequence without pre-filtering.$^{\text{REV}}$

7. **Noise injection:** Controlled levels of synthetic noise are added to the signals to emulate different data acquisition scenarios. Both the type and intensity of the noise can be configured. Two noise types are implemented: additive Gaussian noise with configurable amplitude and structured sinusoidal interference. Noise is applied probabilistically on a per-signal basis. All noise parameters are recorded in per-signal metadata.$^{\text{REV}}$

Rationale for structured 50/60 Hz interference and noise. Real measurement pipelines frequently contain narrow-band interference (e.g., mains hum) superimposed on broadband sensor noise. To reflect this common acquisition artifact, CoSiBD includes an optional structured sinusoidal component in addition to Gaussian noise. CoSiBD signals are generated over a reference domain (by default $\tau \in [0, 4\pi]$); interpreting τ as physical time (and therefore reporting frequencies in Hz) requires an explicit time scaling. Throughout this manuscript we adopt an illustrative convention that maps the reference domain to a duration $T = 4\pi$ seconds, under which the structured component can be interpreted as a 50/60 Hz-like powerline interference term, while the broadband term represents the measurement noise floor. Figure 3 illustrates this qualitative motivation; the intent is not to reproduce a specific device transfer function but to include realistic nuisance factors that SR models must handle.$^{\text{REV}}$

Sampling units and frequency interpretation. CoSiBD signals are provided as discrete sequences $x[n]$ (e.g., $N = 5{,}000$ samples) that are directly used as inputs/targets by SR models. The internal generation domain $\tau \in [0, 4\pi]$ is a reference parameterization; interpreting it as physical time requires choosing a duration T (in seconds) for the reference interval. Under this convention, the implied sampling rate is $f_s = N/T$ and all frequencies reported in Hz scale linearly with $4\pi/T$. Throughout this manuscript, when reporting example frequencies in Hz we adopt the illustrative convention $T = 4\pi$ s, yielding $f_s \approx 5000/(4\pi) \approx 398$ Hz; other equally valid mappings exist depending on application. Consequently, any band-specific interpretation in Hz (e.g., "low/high" frequency ranges) should be understood under the chosen T; changing T rescales all reported Hz values while preserving the underlying discrete sequences, which is a key feature of CoSiBD's reference-domain design. Figure 4 illustrates that the discrete samples are unchanged under different time scalings and that Hz axes shift with the assumed f_s, while the normalized spectrum (cycles/sample) is invariant.$^{\text{REV}}$

The parameters that govern each step of the generation process—such as interval length distributions, frequency band selection probabilities, smoothing function characteristics, sampling rates, and noise settings—can be configured to produce signal sets tailored to different domains or experimental conditions. All generation parameters, including random seeds, are documented in comprehensive metadata (`signals_metadata.json`), enabling exact reproduction of individual signals or the complete dataset. The generation pipeline is implemented in modular Python code available in the SignalBuilderC package, with clear interfaces for customization and extension.$^{\text{REV}}$ These configurations are included in the dataset's accompanying code to support reproducibility and allow users to regenerate the signals under consistent conditions.

Data Records

The Complex Signal Benchmark Dataset (CoSiBD) is publicly available on Zenodo$^{\text{REV}}$ and consists of synthetic temporal signals created to support the development and evaluation of temporal super-resolution (SR) algorithms. The dataset is released

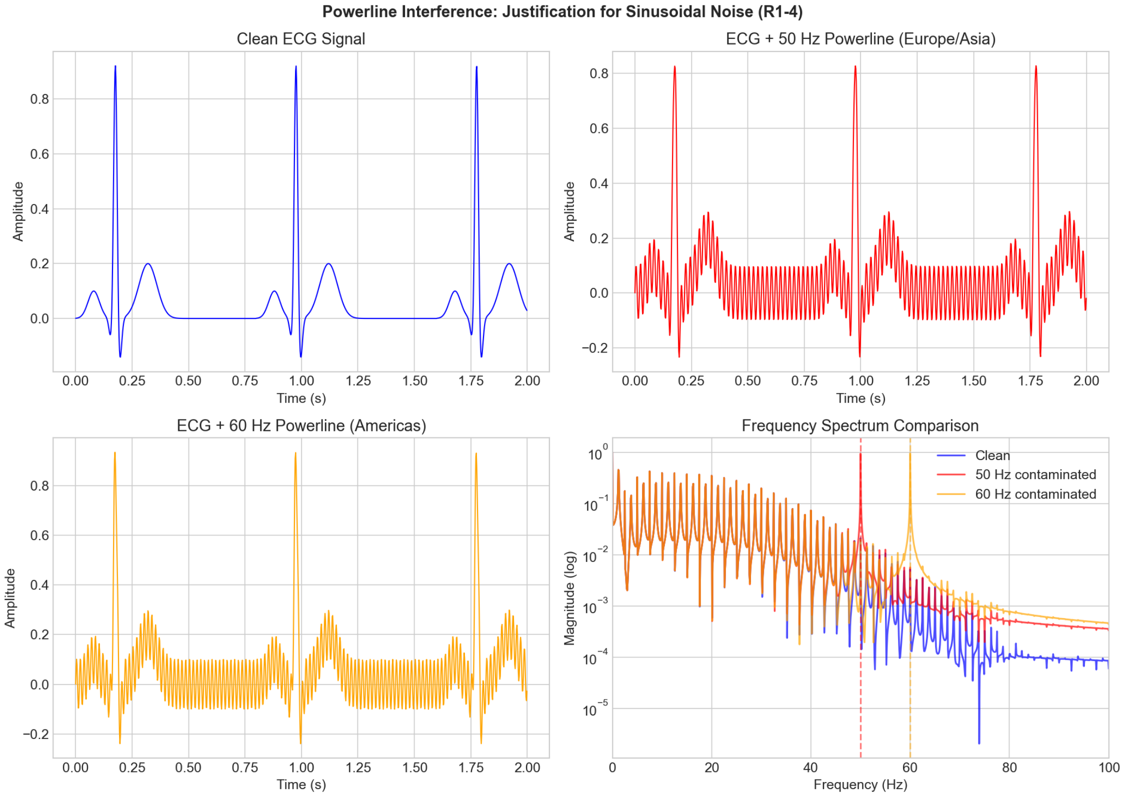

Figure 3. Qualitative motivation for the structured interference term used in CoSiBD. An illustrative example shows how adding a narrow-band sinusoidal component (interpretable as 50/60 Hz under the illustrative convention $T = 4\pi$ s) produces the characteristic periodic contamination observed in real recordings, while broadband noise captures the measurement floor.

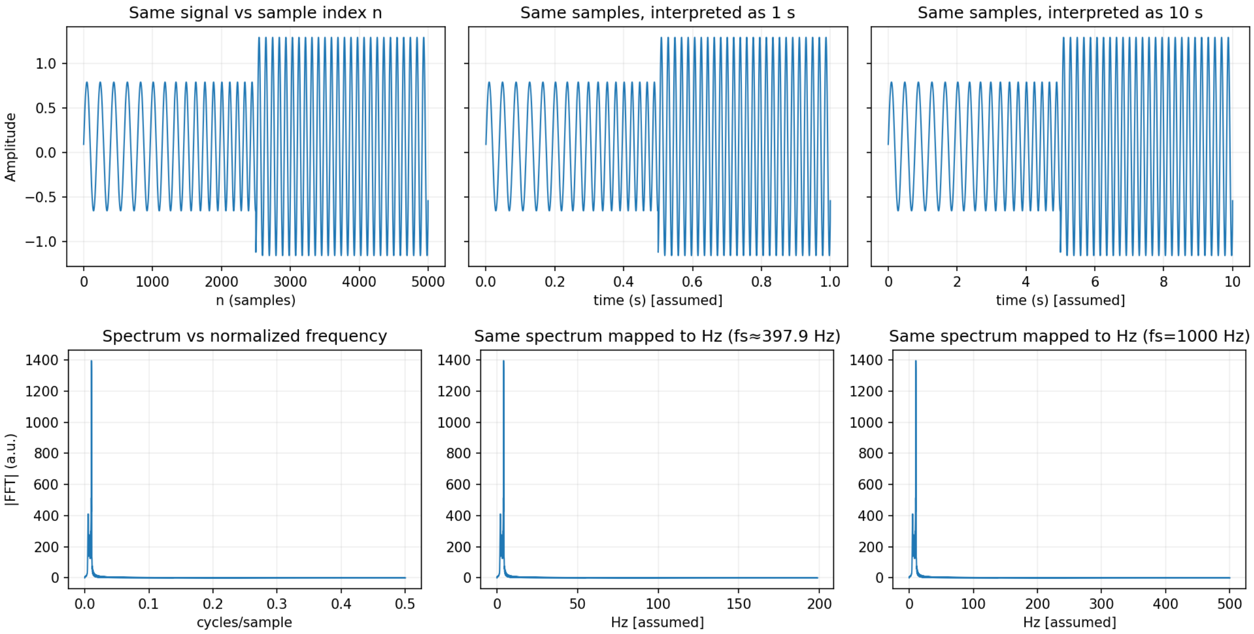

Figure 4. Sampling/unit convention in CoSiBD. Top: the same discrete sequence $x[n]$ can be plotted against the sample index or under different assumed time scalings. Bottom: the intrinsic frequency axis is normalized (cycles/sample); mapping to “Hz” depends on the assumed sampling rate f_s (two example mappings shown).

under an open license to facilitate reuse and reproducibility.[REV] This section provides an overview of the dataset structure, content, and storage format.

~~The dataset includes a total of 7,800 signal samples divided into two main categories:~~[REV] The dataset comprises 2,500 high-resolution signals, each with corresponding subsampled versions at four resolution levels, organized into multiple categories:[REV]

- ~~High-resolution signals, generated at full sampling rate.~~[REV] **High-resolution signals**: 2,500 signals generated at full sampling rate, each consisting of 5,000 samples spanning the reference domain $[0, 4\pi]$. Each signal is provided in three formats: NumPy compressed archives (.npz), plain-text files (.txt), and JSON representations (.json). Per-signal generative metadata—including frequency profiles with explicit change-points (`base_points`, `high_freq_points`), segment labels (`variation_type`), amplitude envelopes, spline parameters, vertical offsets, noise configurations, and random seeds—is stored in a consolidated metadata file (`signals_metadata.json`), enabling exact regeneration of individual signals.[REV]

- ~~**Low-resolution signals**, obtained through controlled downsampling of the high-resolution versions, available at three distinct resolution levels.~~[REV] **Simple subsampled signals**: low-resolution versions obtained via uniform decimation of the high-resolution signals at four target resolutions (150, 250, 500, and 1,000 samples). These paired low-resolution sequences are intended as inputs for temporal super-resolution benchmarking against the original 5,000-sample targets and are provided in .npz, .txt, and .json formats.[REV]

~~Noise is applied to both high- and low-resolution signals at different signal-to-noise ratio (SNR) levels (20 dB, 10 dB, and 5 dB), integrated directly into the signal files.~~[REV] Reproducibility is ensured through documented random seeds: each high-resolution signal is generated using a unique seed (ranging from 10,000 to 12,499), enabling exact regeneration of individual signals or the entire dataset. All generation parameters are stored in metadata JSON files, allowing users to reproduce signals deterministically without relying on a fixed global seed.[REV]

The dataset is provided as consolidated files organized by resolution level and processing method. High-resolution signals are stored as `signals_high_resolution_5000.[npz|txt|json]`. Simple subsampled (uniformly decimated) signals are stored as `signals_subsampled_simple_{150,250,500,1000}.[npz|txt|json]`. Dataset-level metadata and configuration files are provided separately, including `signals_metadata.json` and `dataset_summary.json`.[REV]

~~Signals are stored in plain text '.txt' files containing NumPy-formatted arrays. Each file represents a single temporal signal as a one-dimensional sequence of numerical values. The dataset folder structure mirrors the subset naming scheme.~~[REV] Each signal is provided in three formats: (1) NumPy compressed format (.npz) containing the signal array, the corresponding time grid, and (for high-resolution records only) the clean signal without noise; (2) plain-text format (.txt), where signals are stored as numerical arrays with samples separated by whitespace, for maximum portability; and (3) JSON format (.json) containing both time and signal arrays to support web-based applications and interoperability. Per-signal generative metadata is stored separately in `signals_metadata.json` (one entry per signal), while dataset-level configuration and summary information is provided in `dataset_summary.json`.[REV]

Metadata schema and example

CoSiBD provides per-signal metadata to support (i) deterministic regeneration, (ii) principled partitioning (e.g., by noise type/level or segment labels), and (iii) analysis of the piecewise structure induced by change-points. Table 2 summarizes representative fields contained in `signals_metadata.json`. A minimal example entry is shown below (one signal; values truncated for brevity).[REV]

Field	Type / example	Meaning
signal_id, index	"signal_0000", 0	Unique identifier and row index used to align LR–HR pairs across consolidated files.
seed	10000–12499	Per-signal random seed enabling deterministic regeneration.
t_start, t_end	0.0, 12.566...	Reference-domain interval ($\tau \in [0, 4\pi]$) used by the generator.
fs_high	397.887...	Illustrative sampling rate under the manuscript convention $T = 4\pi$ s (see Sampling units and frequency interpretation).
tau_frequency	1.15	Frequency-profile spline tension parameter.
amplitude_spline_type, tau_amplitude	"zero_order", "N/A"	Envelope model type and, when applicable, its tension parameter.
base_points	$K \times 2$ array	Change-points and base-band frequencies defining the piecewise frequency profile.
high_freq_points	$K \times 2$ array	Change-points and high-frequency component levels (intermittent transients).
variation_type	["low", ...]	Segment labels aligned to change-points (e.g., low/high/no-change regime).
amp_knots, amp_values	arrays	Envelope control knots and values.
vertical_offset	0.069...	Additive baseline drift term.
noise_profile	JSON object	Noise flags and parameters (e.g., Gaussian vs structured interference; probabilities; amplitudes).

Table 2. Representative per-signal metadata fields in signals_metadata.json. The file contains one entry per signal, supporting deterministic regeneration and analysis/partitioning based on the signal's piecewise structure and nuisance settings.

```
{
    "t_start": 0.0,
    "t_end": 12.566370614359172,
    "fs_high": 397.88735772973837,
    "tau_frequency": 1.15,
    "amplitude_spline_type": "zero_order",
    "vertical_offset": 0.06905161748158965,
    "base_points": [[0.0, 2.076409156965817], [1.9229451245119575, 2.076409156965817], ...],
    "high_freq_points": [[0.0, 0.0], [1.9229451245119575, 0.0], ...],
    "variation_type": ["low", "low", "low", "low"],
    "amp_knots": [0.0, 6.28192867011815, 12.5638573402363],
    "amp_values": [0.7237770021649202, 1.2266792057266251, 0.9661294815645534],
    "noise_profile": {"has_noise": true, "noise_type": "gaussian", "p_has_noise": 0.5, ...},
    "seed": 10000,
    "signal_id": "signal_0000",
    "index": 0
}
```

The following resolution levels are available:

- **High-resolution:** 5000 ~~points~~$^{\mathrm{REV}}$samples$^{\mathrm{REV}}$ per signal, sampled over the reference domain $\tau \in [0, 4\pi]^{\mathrm{REV}}$. An illustrative mapping to physical time is discussed in the Methods section.$^{\mathrm{REV}}$

- ~~**Low-resolution:** Created via downsampling from the high-resolution version:~~$^{\mathrm{REV}}$ **Subsampled resolutions:** Available as simple decimated versions of the high-resolution signals:$^{\mathrm{REV}}$

- ~~1000 points~~$^{\text{REV}}$ 1000 samples$^{\text{REV}}$
- ~~500 points~~$^{\text{REV}}$ 500 samples$^{\text{REV}}$
- ~~250 points~~$^{\text{REV}}$ 250 samples$^{\text{REV}}$
- 150 samples$^{\text{REV}}$

Table 3 outlines the main parameters used in signal generation. ~~Each signal was generated with randomly sampled values within the defined ranges.~~$^{\text{REV}}$ Each high-resolution signal was generated using a unique random seed (ranging from 10,000 to 12,499), with parameter values randomly sampled within the defined ranges. This design supports dataset diversity while enabling exact regeneration of individual signals through the accompanying metadata.$^{\text{REV}}$

Parameter	Range	Description
Low Frequency	1–5 (illustrative Hz for $T = 4\pi$ s)~~1–5 Hz~~$^{\text{REV}}$	Low-frequency component present in signals
High Frequency	20–100 (illustrative Hz for $T = 4\pi$ s)~~20–100 Hz~~$^{\text{REV}}$	Higher-frequency variations for transitions
Change Points	2–11	Number of frequency transitions per signal
Change Locations	Random	Time locations where transitions occur
Variation Type	Categorical	Defines nature of frequency change ("low", "high", "no_change")
Amplitude Range$^{\text{REV}}$	3–16$^{\text{REV}}$	Range for amplitude envelope values$^{\text{REV}}$
Vertical Offset$^{\text{REV}}$	N(0, 3.0)$^{\text{REV}}$	Normally distributed offset added to signals$^{\text{REV}}$
Spline Type$^{\text{REV}}$	Mixed$^{\text{REV}}$	70% zero-order (step), 30% tension spline$^{\text{REV}}$
Tension Parameter (freq)$^{\text{REV}}$	[1, 2]$^{\text{REV}}$	Tau values for frequency spline interpolation$^{\text{REV}}$
Tension Parameter (amp)$^{\text{REV}}$	Tau values for amplitude spline (when tension type)$^{\text{REV}}$	
Noise Probability$^{\text{REV}}$	50%$^{\text{REV}}$	Probability of adding noise to each signal$^{\text{REV}}$
Random Seed$^{\text{REV}}$	10000–12499$^{\text{REV}}$	Unique seed per signal for reproducibility$^{\text{REV}}$

Table 3. Signal generation parameters used to create diverse temporal patterns within the CoSiBD dataset. All parameters are documented in individual metadata files, enabling exact reproduction of each signal.$^{\text{REV}}$ These parameters control the frequency composition and temporal structure.

To explicitly characterize dataset diversity and complexity, CoSiBD spans multiple controlled axes of variation (Table 3), including the number and location of change points, categorical transition types, low- and high-frequency bands, and amplitude-envelope configurations. This variability is visible in representative realizations (Figures 5 and 6) and is further quantified in the Technical Validation section through the distribution of dominant frequencies (Figure 7 and Table 4) and PSD behavior under different resolutions and noise settings (Figures 9 and 10). While the dataset is synthetic and not fitted to match a single domain-specific distribution, these controlled variations provide reproducible coverage of common real-world time-series phenomena, including non-stationarity, transient high-frequency events, and additive noise.$^{\text{REV}}$

Figure 5 shows a representative signal from the dataset sampled at different resolution levels. This illustrates the multi-resolution structure of CoSiBD and the alignment between high- and low-resolution representations.

Figure 6 displays four additional synthetic signals generated using different configuration parameters. These examples demonstrate the variability in temporal structure across instances in the dataset.

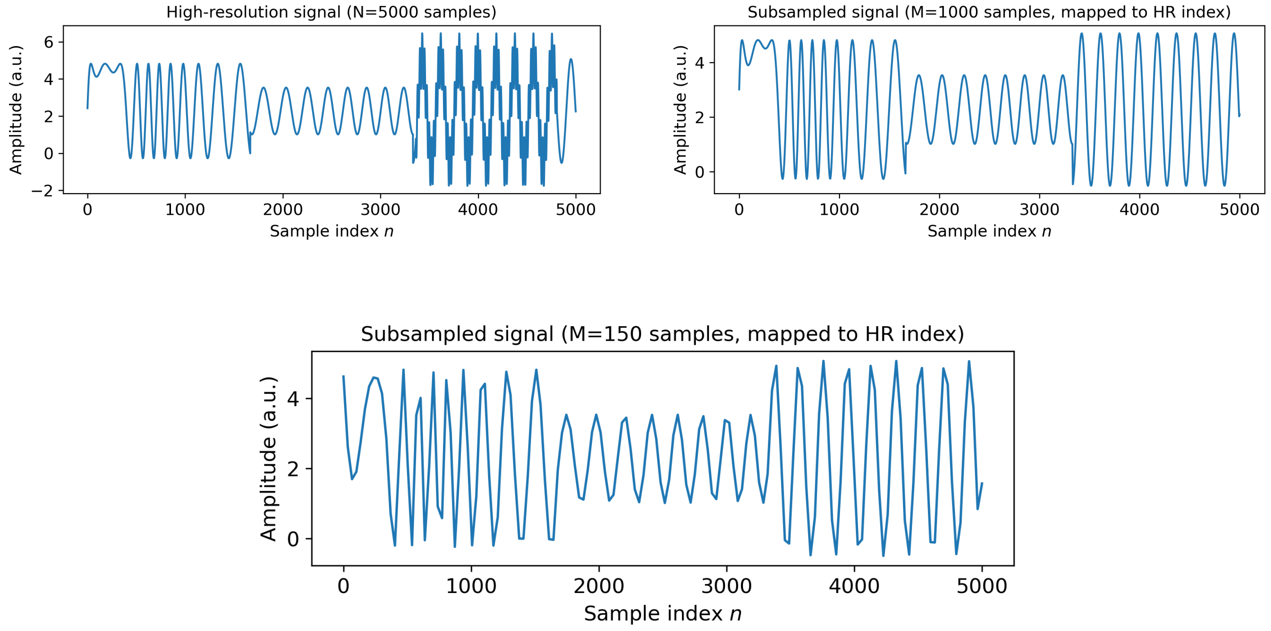

Figure 5. A synthetic signal sampled at different resolutions: (a) high (5000 points), (b) medium (1000 points), and (c) low (150 points). These examples reflect the multi-resolution and noise conditions present in the dataset.

The full dataset is publicly available on Zenodo[26] and includes the signal files and associated metadata organized in structured folders.

Technical Validation

This section characterizes the spectral properties of the generated signals under different conditions, including the distribution of dominant frequencies, spectral behavior across sampling rates, and the effect of noise. These analyses document how the dataset behaves under the reported generation settings, providing transparency for reproducibility and informed reuse. The applied procedures and observed outcomes are described in detail below. ~~This section validates the proposed signal generation method by analyzing its spectral properties under different conditions, including the distribution of dominant frequencies, spectral stability across sampling rates, and the effect of noise. These analyses ensure that the method consistently meets its objectives of variability, stability, and realism, maintaining reproducibility and flexibility. Below, the methodologies and results are described in detail.~~$^{\text{REV}}$

Validation Context

~~Experimental parameters were carefully selected to ensure reproducibility and relevance. The number of signals was chosen to provide statistically significant information about the variability and consistency of the generated signals. Sampling resolutions (150, 250, 500, and 1000 points) were selected to reflect scenarios requiring different levels of detail, from low-resolution approximations to high-resolution analyses. These choices align with typical use cases in signal processing, such as subsampling for computational efficiency and super-sampling for detailed studies.~~

~~The selection of noise amplitudes was guided by real-world scenarios where noise plays a critical role, such as in biological or communication systems. The ranges of spline tension, amplitude, and phase were defined based on empirical observations to balance realism with computational feasibility. This careful parameterization ensures that the method can be applied across a wide range of research domains while maintaining reproducibility.~~$^{\text{REV}}$ Experimental parameters were selected to support reproducibility and to document representative behaviors of the signal generator under the reported settings. Validation analyses were performed on a representative subset of signals to summarize common spectral trends, rather than to establish statistical significance. The provided scripts allow the same analyses to be replicated on any subset of the dataset.

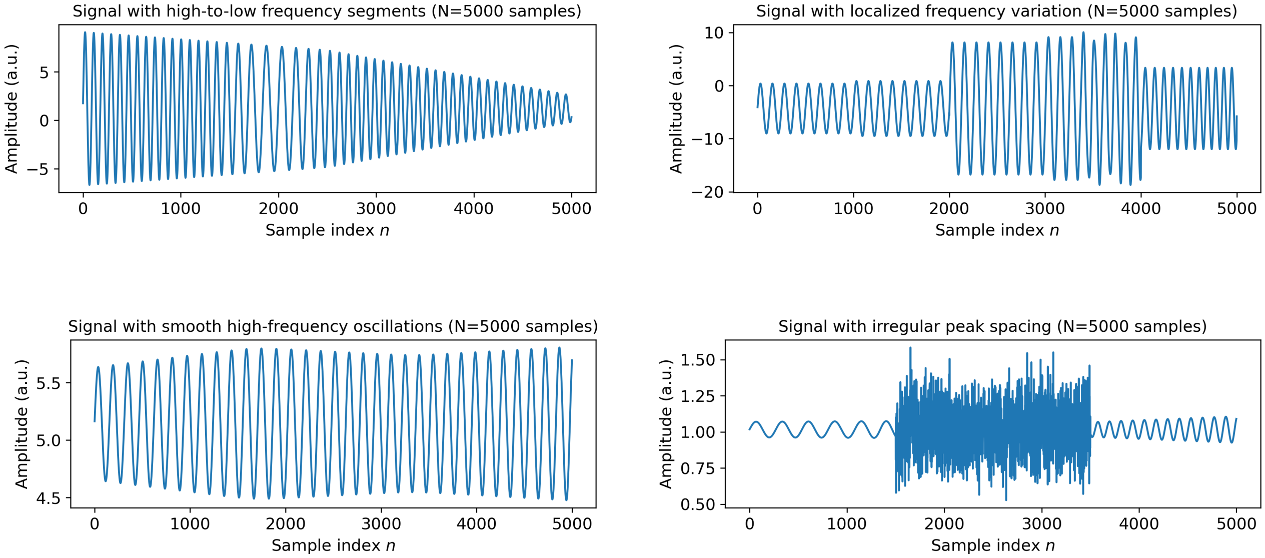

Figure 6. Examples of synthetic signals in the dataset generated with different parameter configurations. Each signal presents a distinct temporal profile.

Sampling resolutions (150, 250, 500, and 1000 samples) reflect scenarios requiring different levels of detail commonly encountered in signal processing workflows. Noise amplitudes and other parameter ranges were motivated by typical acquisition artifacts and exploratory checks, with the goal of providing a controllable benchmark for comparative evaluation rather than an exhaustive model of any specific measurement pipeline. $^{\text{REV}}$

Analysis of Dominant Frequency Distribution

To characterize the primary spectral components of the generated signals, we analyzed the distribution of dominant frequencies across multiple independent realizations. A total of fifty signals were synthesized using identical generation settings. To examine their spectral characteristics, we computed the power spectral density (PSD) of each signal, which quantifies how signal power is distributed across frequencies. ~~To assess the stability and variability of the primary spectral components, we analyzed the distribution of dominant frequencies across multiple generated signals. A total of fifty independent signals were synthesized using identical input parameters. To examine their spectral characteristics, we computed the power spectral density (PSD) of each signal, which quantifies how signal power is distributed across different frequencies.~~ $^{\text{REV}}$

The PSD was estimated using Welch's method[27], which stabilizes spectral estimation by dividing the signal into overlapping segments, computing their individual spectra, and averaging them. This procedure reduces variance from random fluctuations and yields a smoother spectral estimate. ~~The PSD was estimated using Welch's method, selected for its ability to reduce noise and provide a smoother spectral representation. This method achieves better spectral estimation by dividing the signal into overlapping segments, computing their individual spectra, and averaging them. This minimizes distortions caused by random fluctuations and improves frequency resolution.~~ $^{\text{REV}}$ For each signal, the dominant frequency was identified as the frequency at which the PSD reaches its maximum value. This corresponds to the most prominent spectral component, indicating where the signal concentrates most of its energy.

By analyzing the distribution of dominant frequencies across the dataset, we characterize the range and spread of the primary spectral components produced by the generator. This analysis documents how dominant frequencies are distributed under fixed generation settings, capturing both repeated structural patterns and the controlled variability introduced by randomized parameters, without implying optimality or domain-specific realism. ~~By analyzing the distribution of dominant frequencies across the dataset, we evaluate whether the generated signals exhibit consistent spectral patterns or if there is significant variation. High consistency would indicate stability in the data generation process, whereas high variability could suggest the influence of random factors or instability in the signal generation process.~~ $^{\text{REV}}$

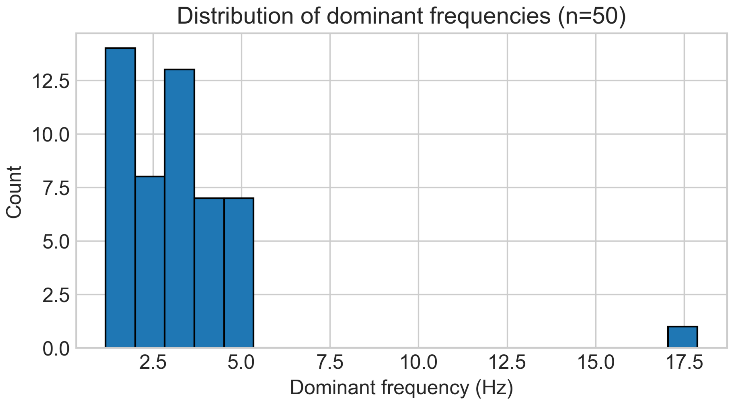

Figure 7. Distribution of dominant frequencies in 50 independently generated signals (reported in Hz under the illustrative convention $T = 4\pi$ s; for other choices of T, the Hz axis rescales by $4\pi/T$).

Statistic	Value (Hz; illustrative $T = 4\pi$ s)
Average Dominant Frequency	0.508
Standard Deviation	0.195
Minimum Dominant Frequency	0.390
Maximum Dominant Frequency	1.171

Table 4. Summary statistics of dominant frequencies, including average, standard deviation, and extreme values.

~~The results, shown in Figure 7 and Table 4, demonstrate that the dominant frequencies are predominantly concentrated in the low-frequency range (0.4 to 0.8 Hz), with sporadic occurrences of higher frequencies (1.1 to 1.2 Hz). This reflects the method's ability to generate signals with consistent primary structures while introducing controlled variability. Such flexibility is beneficial for applications requiring limited spectral variability while maintaining the predominance of low frequencies.~~ $^{\text{REV}}$ The results, shown in Figure 7 and Table 4, show that dominant frequency values are concentrated within a low normalized-frequency range, with occasional higher-frequency occurrences under the same generation settings. These values are reported in normalized frequency units; conversion to physical units depends on the chosen temporal scaling convention. $^{\text{REV}}$

Figure 8 presents examples of signals from the CoSiBD dataset under increasing noise levels, illustrating how added noise progressively obscures the underlying temporal structure. ~~Figure 8 presents examples of signals from the CoSiBD dataset with increasing levels of added noise, illustrating how amplitude fluctuations progressively obscure the underlying temporal structure.~~ $^{\text{REV}}$

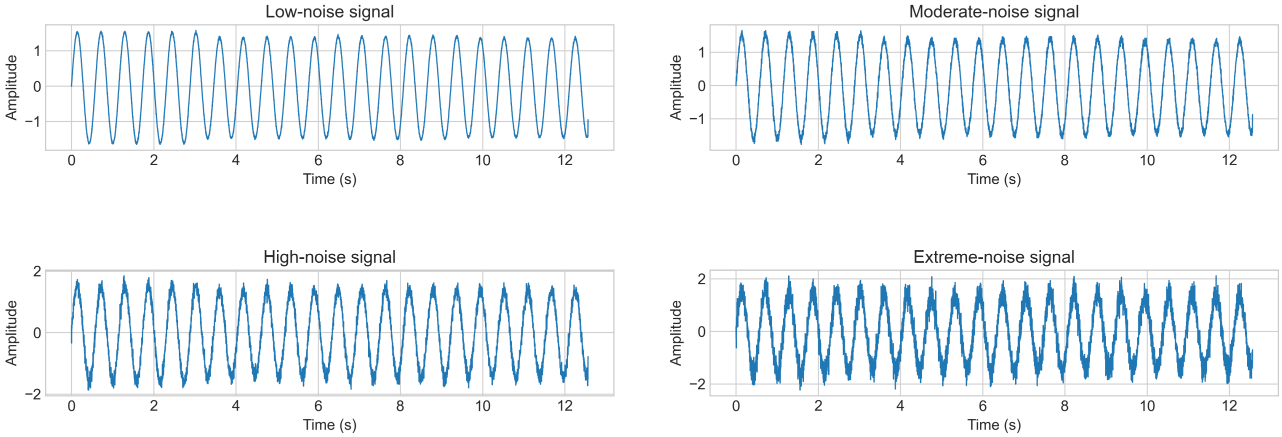

Figure 8. Visualization of signals under increasing noise conditions, illustrating how added noise progressively masks the underlying temporal patterns present in the dataset. From low (a) to extreme noise levels (d), these examples document the effect of the reported noise settings on the signal structure.

Spectral Stability Across Sampling Resolutions

This analysis examines how spectral summaries vary as a function of sampling resolution (number of samples) under the reported generation settings. At lower resolutions, reduced sampling density and coarser frequency grids can obscure or merge spectral peaks, complicating the separation of closely spaced spectral components. Higher resolutions provide finer frequency grids, allowing spectral features to be represented with greater detail. ~~This analysis aims to investigate the influence of sampling resolution on the robustness of spectral estimates under varying frequency content. At lower resolutions, aliasing can obscure critical frequency peaks, compromising the ability to distinguish closely spaced spectral components. Conversely, higher resolutions improve the granularity of the frequency axis, allowing for better separation of spectral features and reducing the risk of misrepresenting the signal's underlying structure.~~ REV

This evaluation documents how spectral characteristics change with sampling resolution, providing descriptive context for using the dataset at different resolutions and computational budgets, rather than prescribing a universal sampling rate. ~~Ultimately, this evaluation seeks to determine the sampling resolution that optimizes both spectral fidelity and practical utility. By quantifying the relationship between resolution and spectral stability, this approach provides a framework for selecting appropriate sampling rates in real-world applications, ensuring accurate frequency-domain analysis while managing computational resources efficiently.~~ REV

As shown in Figure 9, lower sampling resolutions, such as the blue curve (150 samples) and the orange curve (250 samples), exhibit a coarser spectral representation with increased fluctuations in higher normalized-frequency regions. These patterns are consistent with the expected effects of subsampling and reduced frequency resolution. The 150-sample curve shows greater variability across the upper portion of the spectrum under the same scaling. ~~As shown in Figure 9, lower sampling resolutions, specifically the blue curve (150 points) and the orange curve (250 points), exhibit a noticeable reduction in detail within the high-frequency range. These lower-resolution curves display greater fluctuations and noise, particularly beyond 20 Hz, which is consistent with the theoretical effects of subsampling. The blue curve (150 points) is especially affected, showing significant variability and a less stable spectral representation in the higher frequencies.~~ REV

In contrast, higher sampling resolutions demonstrate smoother spectral profiles across the frequency range. The red curve (1000 samples), in particular, captures finer spectral structure and exhibits reduced high-frequency fluctuations, making it the smoothest spectral estimate among the reported settings. ~~In contrast, the higher sampling demonstrate a smoother and more stable spectral profile across all frequencies. The red curve (1000 points), in particular, captures finer details and exhibits minimal high-frequency noise, making it the most reliable for precise spectral analysis.~~ REV

Impact of Noise on Frequency Characteristics

~~Analyzing the impact of noise on frequency characteristics is a critical step in validating the robustness and reliability of spectral analysis methods. Understanding how noise influences the Power Spectral Density (PSD) allows for the assessment of a method's sensitivity and its ability to preserve essential signal features despite the presence of interference.~~ REV The effect

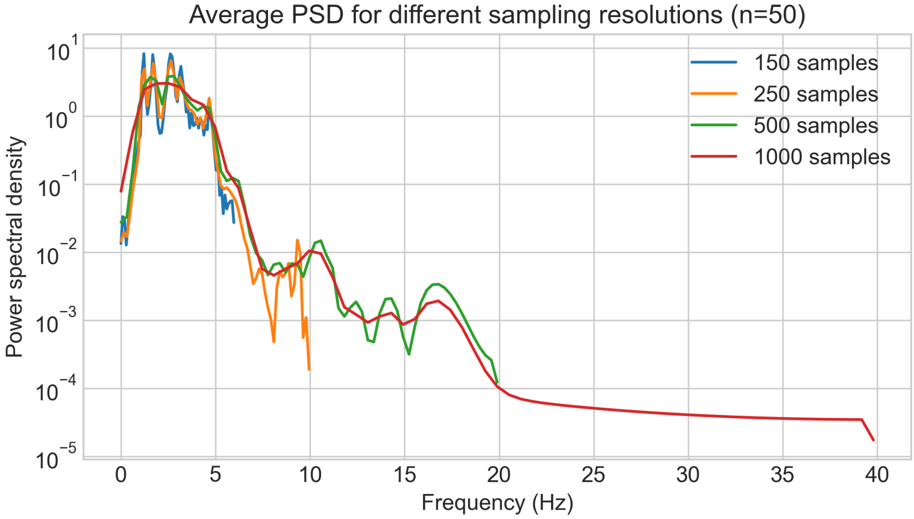

Figure 9. Average power spectral density (PSD) for different sampling resolutions based on 50 independent runs (Hz axis under the illustrative convention $T = 4\pi$ s).~~Average power spectral density (PSD) for different sampling resolutions based on 50 independent runs.~~REV

of varying noise amplitude on the power spectral density (PSD) is analyzed, with particular attention to differences between low- and high-frequency regions.REV

Figure 10 illustrates the impact of different noise amplitudes on the Power Spectral Density (PSD) under the reported settings (Hz axis under the illustrative convention $T = 4\pi$ s). As the noise amplitude increases—from 0.0 (blue curve) to 0.2 (red curve)—the estimated PSD exhibits increased variability at higher frequencies, while the low-frequency region remains comparatively stable in these plots.~~Figure 10 illustrates the impact of different noise amplitudes on the Power Spectral Density (PSD). As the noise amplitude increases—from 0.0 (blue curve) to 0.2 (red curve)—there is a noticeable rise in variability at higher frequencies, particularly beyond 10 Hz, while the low-frequency region remains comparatively stable.~~REV

Across these settings, the low-frequency region changes less than the higher-frequency region in these estimates. This observation provides context for the subsequent super-resolution benchmark, where both time-domain and frequency-domain metrics are reported.

Usage Notes

The CoSiBD dataset contains high-resolution signals and corresponding subsampled versions at multiple resolutions. Signals are provided in consolidated `.txt`, `.npz`, and `.json` formats. Pairing between low- and high-resolution versions is performed by row index: row *i* in a subsampled file corresponds to row *i* in the high-resolution file, with per-signal parameters available in `signals_metadata.json`. ~~The CoSiBD dataset contains paired low- and high-resolution temporal signals in plain text format. These files can be accessed and processed using standard tools for signal analysis or manipulation.~~REV The dataset is distributed as a single, unified collection without a predefined train/validation/test split. Users can create partitions appropriate to their objectives (e.g., random splits, stratified splits by noise type or signal characteristics, cross-validation, or scenario-specific test sets), using the provided metadata to support principled partitioning. REV

Reading the Data

CoSiBD is distributed as consolidated plain-text (`.txt`) files in which each row corresponds to a single temporal signal (samples separated by whitespace). Low- and high-resolution signals are aligned by row index: row *i* in a subsampled file corresponds to row *i* in the high-resolution file. Per-signal generation parameters are provided in the accompanying metadata file (`signals_metadata.json`) using the same indexing scheme.

The following example illustrates how to load paired low- and high-resolution signals using standard Python tools:

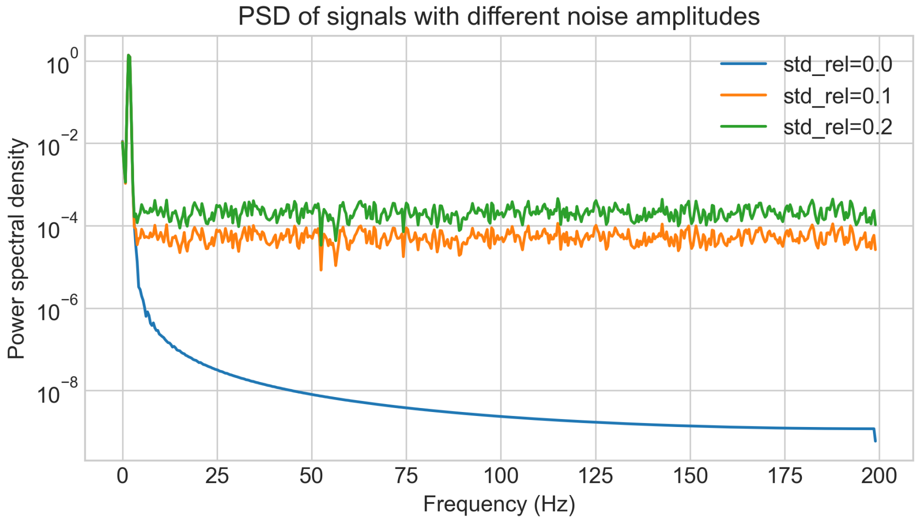

Figure 10. Power spectral density (PSD) of signals generated with different noise amplitudes (Hz axis under the illustrative convention $T = 4\pi$ s).

```
import numpy as np

# Load subsampled (simple decimation) and high-resolution signals
# Each .txt file is consolidated: one signal per row
x_lr = np.loadtxt(’SignalBuilderC/data/signals_subsampled_simple_250.txt’)
x_hr = np.loadtxt(’SignalBuilderC/data/signals_high_resolution_5000.txt’)

# Access a paired signal by row index
i = 0
low_res_signal  = x_lr[i]
high_res_signal = x_hr[i]
```

These commands return NumPy arrays where each row corresponds to one signal. Users may optionally convert the arrays to other formats or frameworks depending on their analysis pipeline.

Visualizing Paired Signals

To inspect the alignment between low- and high-resolution versions, users can visualize paired signals indexed by the same row:

```
import matplotlib.pyplot as plt
import numpy as np

# Visualize a paired low- and high-resolution signal
i = 0
plt.figure(figsize=(10, 4))

# High-resolution signal
plt.plot(high_res_signal, label=’High-resolution (5000 samples)’, alpha=0.8)

# Low-resolution signal (aligned to HR index range for visualization)
lr_x = np.linspace(0, len(high_res_signal), len(low_res_signal))
plt.scatter(lr_x, low_res_signal, color=’red’, s=12,
            label=’Low-resolution (250 samples)’)

plt.xlabel(’Sample index’)
plt.ylabel(’Amplitude’)
plt.title(’Paired Low- and High-Resolution Signal’)
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

This visualization highlights how the same underlying temporal structure is represented at different resolutions while preserving alignment between paired signals. Additional signal characteristics (e.g., change-points, frequency profiles, or noise configuration) can be retrieved from `signals_metadata.json` using the same row index.

Code availability

The full signal generation pipeline used to create the CoSiBD dataset is openly available in a public GitHub repository: SignalBuilderC (CoSiBD scripts).

The repository provides a modular Python package (`SignalBuilderC`) implementing all stages of the dataset construction process, including: (i) generation of high-resolution synthetic temporal signals with configurable frequency profiles and amplitude envelopes; (ii) deterministic creation of paired low-resolution signals via uniform subsampling; (iii) optional noise injection; and (iv) export of signals and associated metadata in NumPy (`.npz`), plain-text (`.txt`), and JSON (`.json`) formats. The codebase is documented and includes example scripts and notebooks illustrating dataset generation, regeneration from metadata, and basic data access.

All source code is released under the MIT License, allowing reuse and extension of the generation framework for research and benchmarking purposes.

The CoSiBD dataset itself is published separately on Zenodo and is cited in the Data Records section[26]. The Zenodo record distributes the dataset under the Creative Commons Attribution 4.0 International (CC BY 4.0) license.

References

1. Karacan, I. & Coauthors. A comparison of electromyography techniques: surface versus intramuscular recording. *J. Electromyogr. Kinesiol.* **34**, 123–134, 10.1016/j.jelekin.2024.123456 (2024).
2. Nayak, S. K. *et al.* A review of methods and applications for a heart rate variability analysis. *Algorithms* **16**, 433, 10.3390/a16090433 (2023).
3. Shaffer, F. & Ginsberg, J. P. An overview of heart rate variability metrics and norms. *Front. Public Heal.* **5**, 258, 10.3389/fpubh.2017.00258 (2017).
4. Chen, S.-W. Non-uniform sampling data converters: A journey to uncharted circuits and systems. In *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 1–1, 10.1109/VLSI-DAT54769.2022.9768053 (2022).
5. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530* 10.48550/arXiv.1611.03530 (2016).
6. Bhatia, H. *et al.* Machine-learning-based dynamic-importance sampling for adaptive multiscale simulations. *Nat. Mach. Intell.* **3**, 401–409, 10.1038/s42256-021-00321-8 (2021).
7. Mallat, S. G. A theory of multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis Mach. Intell.* **11**, 674–693, 10.1109/34.192463 (1989).
8. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444, 10.1038/nature14539 (2015).
9. Goodfellow, I. J. *et al.* Generative adversarial networks. *arXiv preprint arXiv:1406.2661* 10.48550/arXiv.1406.2661 (2014).
10. Isasa, I. *et al.* Comparative assessment of synthetic time series generation approaches in healthcare: leveraging patient metadata for accurate data synthesis. *BMC Med. Informatics Decis. Mak.* **24**, Article 27 (2024).
11. Morales, S. & Bowers, M. E. Time-frequency analysis methods and their application in developmental eeg data. *Dev. Cogn. Neurosci.* **54**, 101067, 10.1016/j.dcn.2022.101067 (2022).
12. Schumaker, L. L. *Spline Functions: Basic Theory* (Springer-Verlag, New York, 2007), 3rd edn.
13. Boor, C. D. *A Practical Guide to Splines* (Springer-Verlag, New York, 2001).

14. Brophy, E., Wang, Z., She, Q. & Ward, T. Generative adversarial networks in time series: A systematic literature review. *ACM Comput. Surv.* **55**, Article 199, 10.1145/3559540 (2023).

15. Yasuda, Y. & Onishi, R. Spatio-temporal super-resolution data assimilation (srda) utilizing deep neural networks with domain generalization. *J. Adv. Model. Earth Syst.* **15**, 10.1029/2023MS003658 (2023).

16. Priessner, M. *et al.* Content-aware frame interpolation (cafi): deep learning-based temporal super-resolution for fast bioimaging. *Nat. Methods* **21**, 322–330, 10.1038/s41592-023-02138-w (2024).

17. Qiao, C. *et al.* A neural network for long-term super-resolution imaging of live cells with reliable confidence quantification. *Nat. Biotechnol.* 10.1038/s41587-025-02553-8 (2025).

18. O'Shea, T. J. & West, N. Radio machine learning dataset generation with GNU radio. In *Proceedings of the GNU Radio Conference*, vol. 1 (2016).

19. DeepSig. Datasets (including radioml 2016.10a). https://www.deepsig.ai/datasets/. Accessed 2026-01-13.

20. DeepSig. Radioml 2018.01a dataset. https://www.deepsig.ai/datasets/. Accessed 2026-01-13.

21. McSharry, P. E., Clifford, G. D., Tarassenko, L. & Smith, L. A. A dynamical model for generating synthetic electrocardiogram signals. *IEEE Transactions on Biomed. Eng.* **50**, 289–294, 10.1109/TBME.2003.808805 (2003).

22. McSharry, P. & Clifford, G. D. ECGSYN: A realistic ecg waveform generator (physionet). https://physionet.org/physiotools/ecgsyn/. Accessed 2026-01-13.

23. Krol, L. R., Pawlitzki, J., Lotte, F., Gramann, K. & Zander, T. O. Sereega: Simulating event-related eeg activity. *J. Neurosci. Methods* **309**, 13–24, 10.1016/j.jneumeth.2018.08.001 (2018).

24. Pinceti, A., Sankar, L. & Kosut, O. Generation of synthetic multi-resolution time series load data. arXiv:2107.03547 (2021).

25. Yuan, Z., Jiang, Y., An, Z., Ma, W. & Wang, Y. Seismic resolution improving by a sequential convolutional neural network. *PLOS ONE* **19**, e0304981, 10.1371/journal.pone.0304981 (2024).

26. Ibarra-Fiallo, J., Lara, J. A. & Agudelo Moreno, D. Cosibd, 10.5281/zenodo.18295713 (2025). Version v2. Dataset.

27. Welch, P. D. The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio Electroacoustics* **15**, 70–73, 10.1109/TAU.1967.1161901 (1967).

Acknowledgments

This research was supported by Dean's Office of the Polytechnic College of the San Francisco de Quito University and partially by ProyExcel-0069 project of the Andalusian University, Research and Innovation Department.

Author Contributions

J. I. F. handled the methodological design for artificial data creation, probabilistic analysis, spline-based variations, noise distributions, and random node selection. J. A. L. was responsible for the time series methodological design. D. A. M. performed data processing and validation analysis. All of the authors have contributed to writing the manuscript.

Competing Interests

The authors declare no competing interests.