

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

Julio Ibarra-Fiallo<sup>1</sup>, Juan A. Lara<sup>2</sup>, and D'hamar Agudelo-Moreno<sup>1</sup>

<sup>1</sup>Colegio de Ciencias e Ingenierías, Universidad San Francisco de Quito, Cumbayá, Ecuador

<sup>2</sup>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. 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. 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. 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. The dataset includes clean and noisy variants across all sampling resolutions, enabling systematic benchmarking under controlled variability conditions. 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.

## Background & Summary

The analysis and simulation of temporal signals are fundamental across science and engineering, providing critical insights into dynamic processes across multiple domains. In biomedical research<sup>1</sup>, electroencephalography (EEG) and electrocardiography (ECG) analyses reveal brain and heart function<sup>2,3</sup>. Telecommunications rely on signal processing to ensure data fidelity across noisy media<sup>4</sup>, while finance uses time-series forecasting for risk and trend analysis<sup>5</sup>. Industrial monitoring detects equipment faults using temporal patterns<sup>6</sup>, and environmental science applies similar techniques to climate and environmental monitoring using remote-sensor time series<sup>7</sup>. 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.

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<sup>8,9</sup>. 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.

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)<sup>10</sup>. In other domains, including environmental monitoring using remote sensors and industrial monitoring, data availability is limited by practical and economic barriers to sensor deployment and data collection<sup>5</sup>. 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.

Temporal SR, which enhances resolution over time, has broad potential. In biomedical monitoring and sensing, temporal SR can help reconstruct higher-resolution physiological time series, such as ECG and EEG signals. For EEG analysis, SR may help recover high-frequency components that aid in the study of neural oscillations<sup>2</sup> or detect subtle physiological irregularities<sup>3</sup>. In domains such as environmental sensing, telecommunications, and industrial monitoring, SR can increase sensitivity to rapid temporal changes.

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<sup>7</sup>; and splines, though flexible, may not generalize well to complex signal variability<sup>11,12</sup>. 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<sup>8,9</sup>. While GANs have achieved strong results in image SR<sup>13</sup>, their application to time-series SR remains relatively new. Preliminary work on synthetic time-series generation indicates potential<sup>13</sup>, 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<sup>14</sup>, bioimaging<sup>15</sup>, and live-cell imaging<sup>16</sup>, 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. It includes non-stationary, piecewise-structured signals generated via non-uniform interval partitioning with change-points, multiple levels of resolution and noise, a technical validation suite, and publicly available Python code to facilitate use.

## 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<sup>17–19</sup>. 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<sup>20–22</sup>. 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<sup>23</sup>. 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<sup>24</sup>.

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 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.

## 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

<sup>1</sup>“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.

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 meta-data; deterministic regeneration (seed-controlled)
RadioML 2016.10A <sup>17,18</sup>	Wireless communications (I/Q)	Dataset	No (classification benchmark)	N/A (not SR)	Variable SNR + channel impairments	Dataset-level (labels/SNR)
RadioML 2018.01A <sup>19</sup>	Wireless communications (I/Q)	Dataset	No (classification benchmark)	N/A (not SR)	Simulated channel effects + SNR variability	Dataset-level
ECGSYN <sup>20,21</sup>	ECG (physiology)	Simulator/tool	Configurable <sup>1</sup>	Configurable	Model-based; supports controlled variability	User-defined simulator parameters
SEREEGA <sup>22</sup>	EEG (physiology)	Simulator/tool	Configurable <sup>1</sup>	Configurable	Supports noise and event-related components	User-defined simulator parameters
LoadGAN <sup>23</sup>	Power systems load time series	Generator/tool	No (generation)	Yes (variable sampling rates)	Domain-specific variability	Configurable generation settings
Synthetic LR-HR seismic traces <sup>24</sup>	Seismic traces (geophysics)	Paper-specific paired data	Yes (LR-HR pairs)	Study-specific	Study-dependent	Study-specific

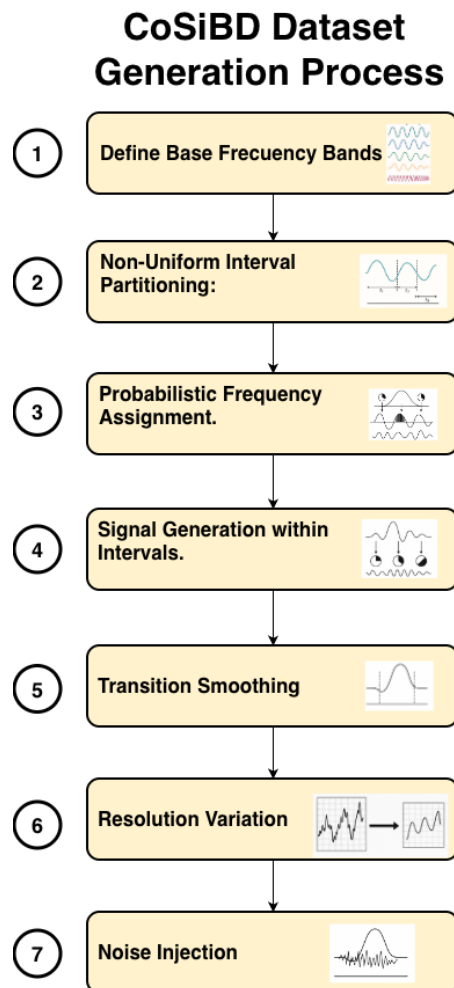
**Table 1.** Representative publicly available synthetic time-series datasets and simulators related to signal processing and learning.

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.

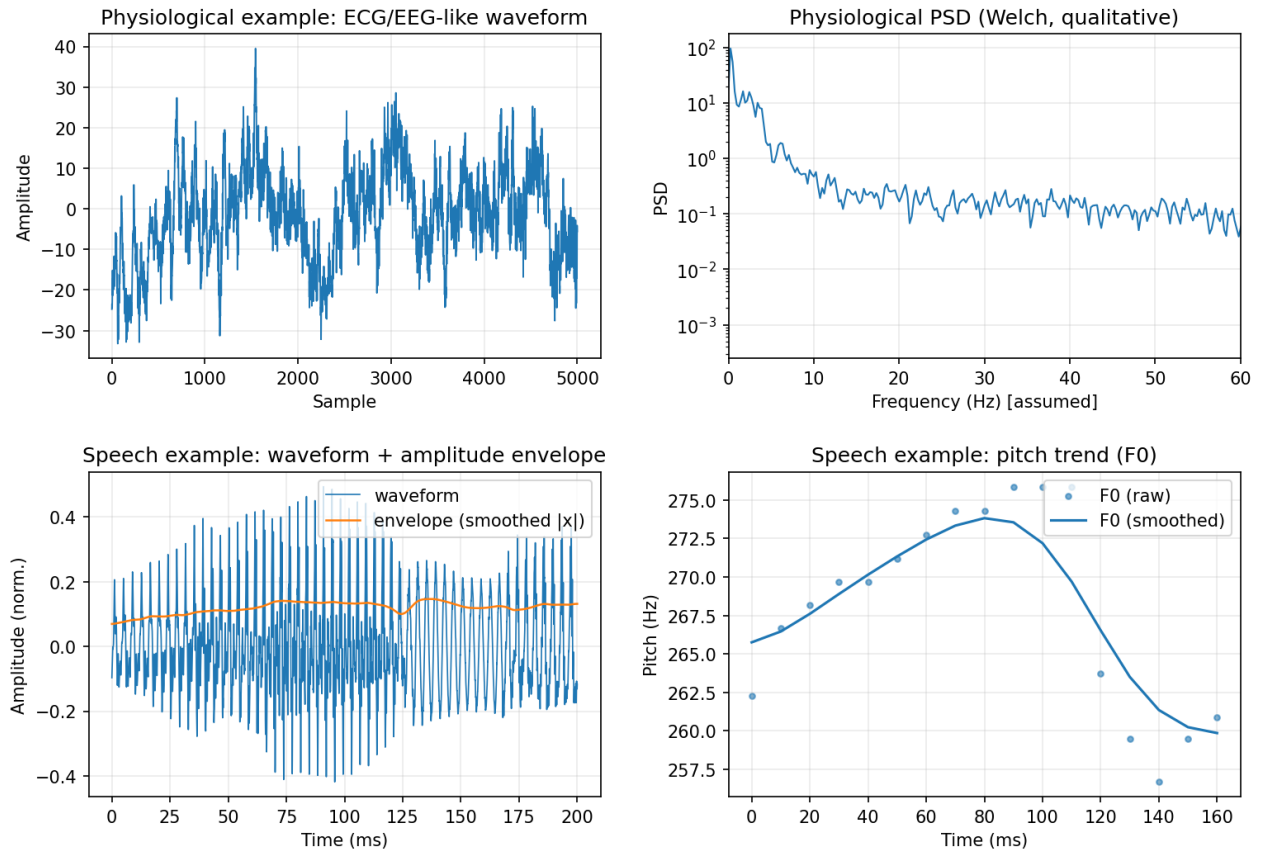
The signal generation pipeline involves the following steps:

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

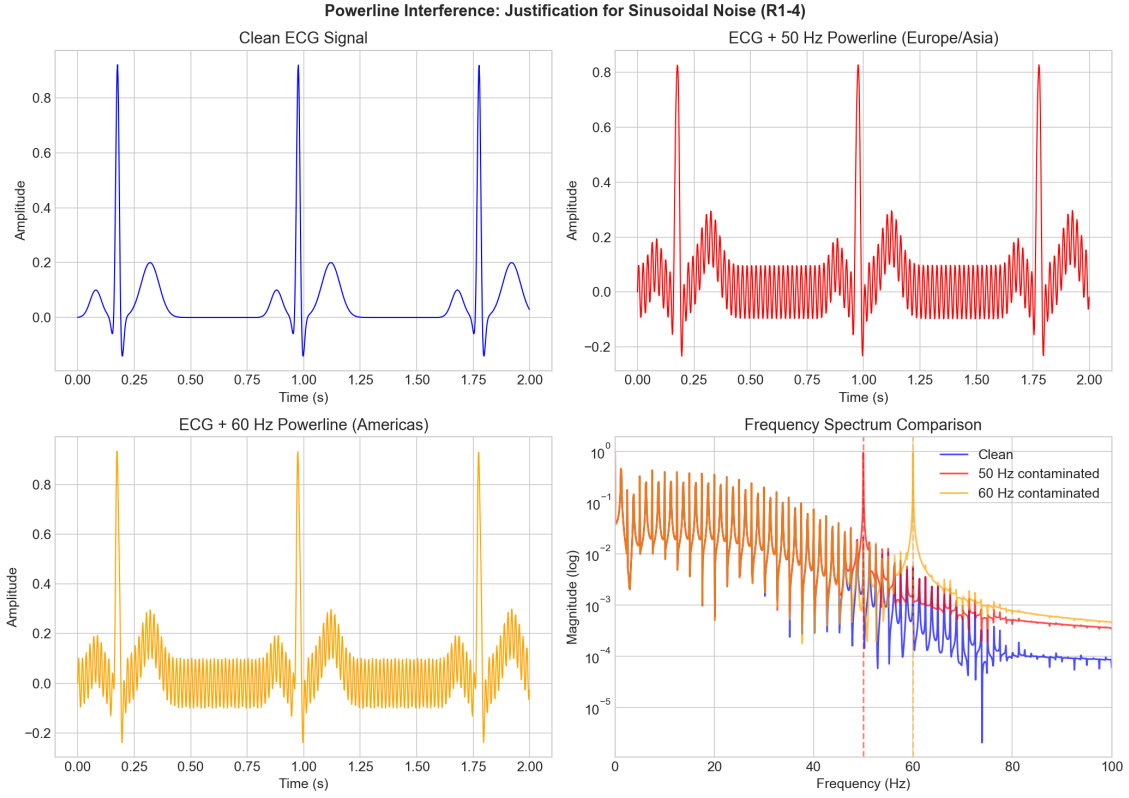


**Figure 1.** Schematic overview of the CoSiBD signal generation process.

Real-signal properties motivating CoSiBD design (qualitative examples)



**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.

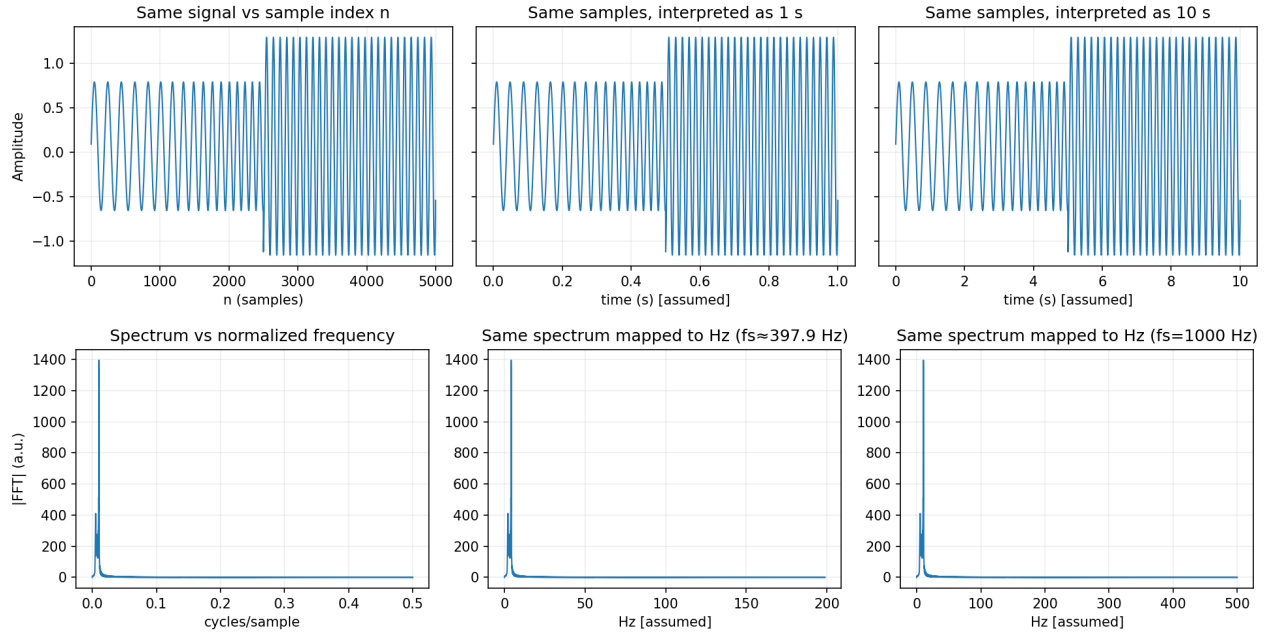


**Figure 3.** Qualitative motivation for the structured interference term used in CoSiBD. An illustrative example shows how adding a narrow-band sinusoidal component (interpretable as 50/60 Hz under the illustrative convention  $T = 4\pi$  s) produces the characteristic periodic contamination observed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

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.
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.

**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  $\tau$  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.

**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 or 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



**Figure 4.** Sampling and unit convention in CoSiBD. Top: the same discrete sequence  $x[n]$  plotted against the sample index or under different assumed time scalings. Bottom: the intrinsic frequency axis is normalized (cycles per sample); mapping to physical frequency units depends on the assumed sampling rate  $f_s$ .

interpretation in Hz 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 per sample) is invariant.

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. 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 and consists of synthetic temporal signals created to support the development and evaluation of temporal super-resolution (SR) algorithms. The dataset is released under an open license to facilitate reuse and reproducibility. This section provides an overview of the dataset structure, content, and storage format.

The dataset comprises 2,500 high-resolution signals, each with corresponding subsampled versions at four resolution levels, organized into multiple categories:

- **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.

- **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.

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.

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`.

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`.

### Metadata schema and example

CoSiBD provides per-signal metadata to support (i) deterministic regeneration, (ii) principled partitioning (e.g., by noise type 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).

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.
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.
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).

**Table 2.** Representative per-signal metadata fields in `signals_metadata.json`.

```

185     "t_start": 0.0,
186     "t_end": 12.566370614359172,
187     "fs_high": 397.88735772973837,
188     "tau_frequency": 1.15,
189     "amplitude_spline_type": "zero_order",
190     "vertical_offset": 0.06905161748158965,
191     "base_points": [[0.0, 2.0764], [1.9229, 2.0764], ...],
192     "high_freq_points": [[0.0, 0.0], [1.9229, 0.0], ...],
193     "variation_type": ["low", "low", "low", "low"],
194     "amp_knots": [0.0, 6.2819, 12.5639],
195     "amp_values": [0.7238, 1.2267, 0.9661],
196     "noise_profile": {"has_noise": true, "noise_type": "gaussian", ...},
197     "seed": 10000,
198     "signal_id": "signal_0000",
199     "index": 0
200 }

```

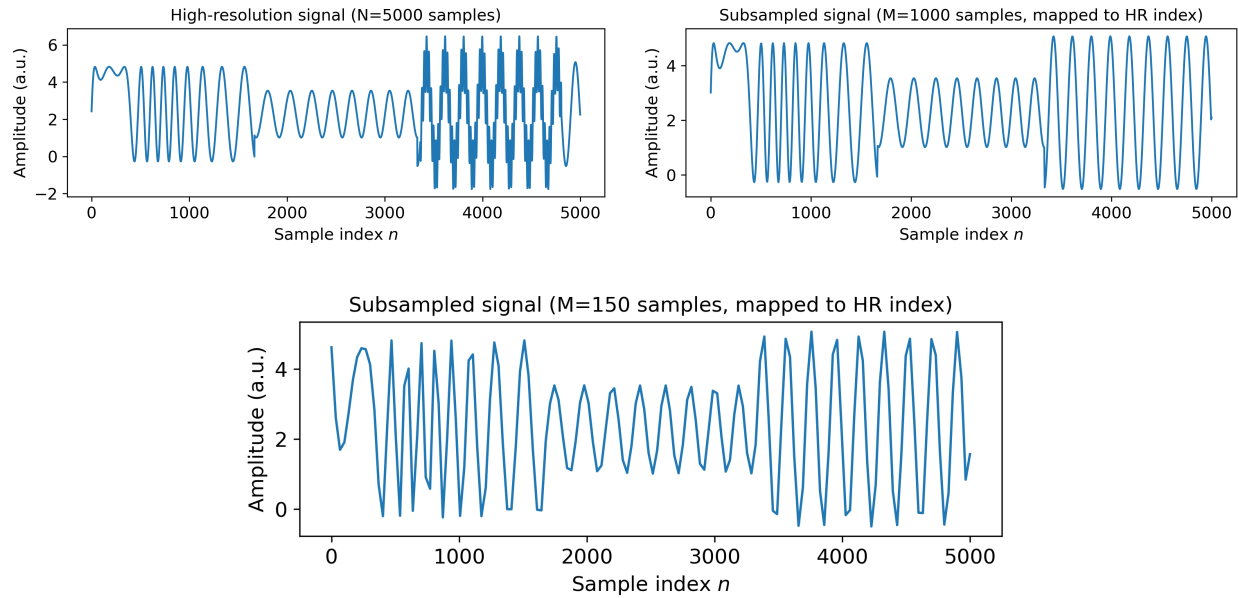
201 The following resolution levels are available:

- 202 • **High-resolution:** 5,000 samples per signal, sampled over the reference domain  $\tau \in [0, 4\pi]$ .
- 203 • **Subsampled resolutions:** simple decimated versions of the high-resolution signals:
  - 204 – 1,000 samples
  - 205 – 500 samples
  - 206 – 250 samples
  - 207 – 150 samples

208 Table 3 outlines the main parameters used in signal generation. Each high-resolution signal was generated using a unique  
 209 random seed (ranging from 10,000 to 12,499), with parameter values randomly sampled within the defined ranges.

Parameter	Range	Description
Low Frequency	1–5 (illustrative Hz for $T = 4\pi$ s)	Low-frequency component present in signals
High Frequency	20–100 (illustrative Hz for $T = 4\pi$ s)	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
Amplitude Range	3–16	Range for amplitude envelope values
Vertical Offset	$\mathcal{N}(0, 3.0)$	Normally distributed offset added to signals
Spline Type	Mixed	70% zero-order (step), 30% tension spline
Tension Parameter (freq)	[1,2]	Tau values for frequency spline interpolation
Tension Parameter (amp)	Configurable	Tau values for amplitude spline when applicable
Noise Probability	50%	Probability of adding noise to each signal
Random Seed	10000–12499	Unique seed per signal for reproducibility

**Table 3.** Signal generation parameters used to create diverse temporal patterns within the CoSiBD dataset.



**Figure 5.** A synthetic signal sampled at different resolutions: (a) high (5,000 samples), (b) medium (1,000 samples), and (c) low (150 samples).

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

Figure 6 displays additional synthetic signals generated using different configuration parameters, demonstrating variability in temporal structure across instances in the dataset.

The full dataset is publicly available on Zenodo<sup>25</sup> 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.

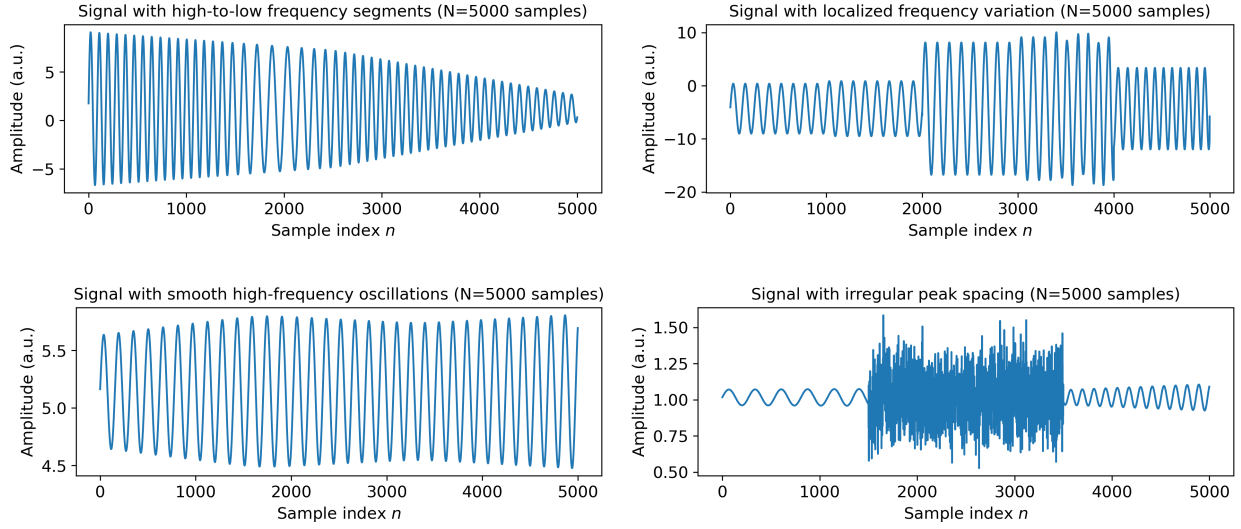
### Validation Context

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.

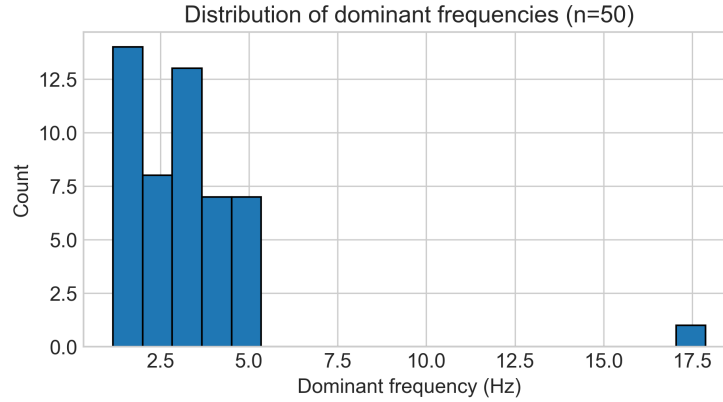
Sampling resolutions (150, 250, 500, and 1000 samples) reflect scenarios requiring different levels of detail commonly encountered in signal processing workflows. Noise amplitudes and other parameter ranges were motivated by typical acquisition artifacts and exploratory checks, with the goal of providing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

### 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.



**Figure 6.** Examples of synthetic signals in the dataset generated with different parameter configurations.



**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  $4\pi/T$ ).

The PSD was estimated using Welch’s method<sup>26</sup>, 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. 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.

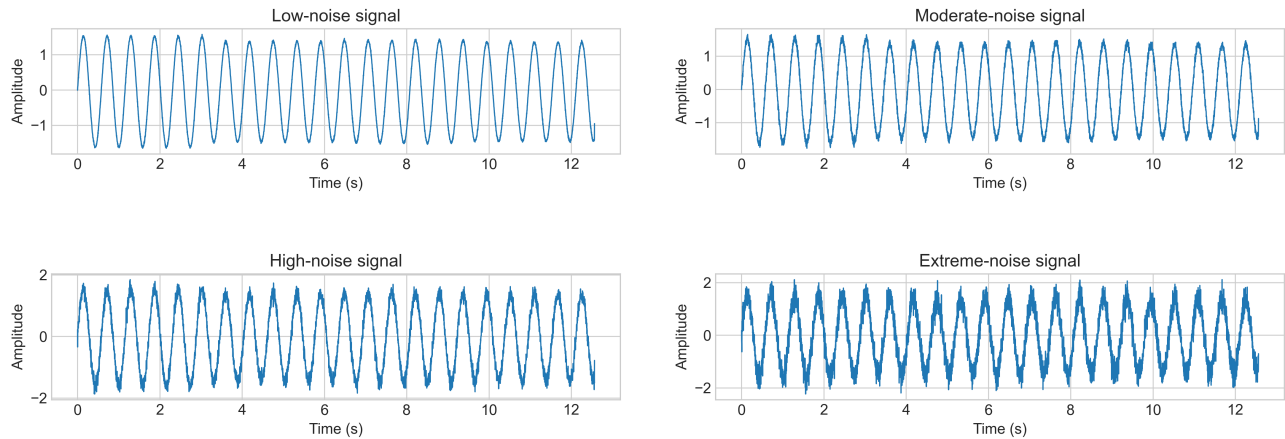
By analyzing the distribution of dominant frequencies across the dataset, we characterize the range and spread of the primary spectral components produced by the generator. This analysis documents how dominant frequencies are distributed under fixed generation settings, capturing both repeated structural patterns and the controlled variability introduced by randomized parameters, without implying optimality or domain-specific realism.

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.

Figure 8 presents examples of signals from the CoSiBD dataset under increasing noise levels, illustrating how added noise progressively obscures the underlying temporal structure.

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.



**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 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.

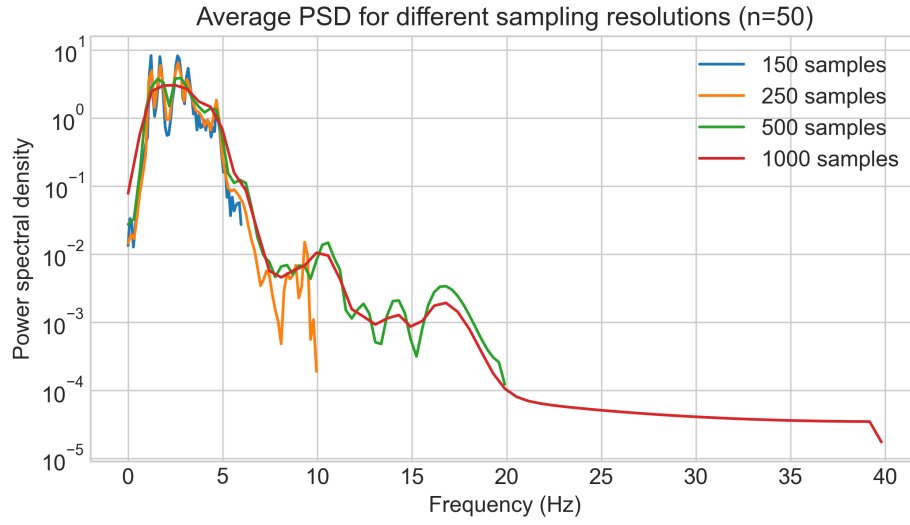
As shown in Figure 9, lower sampling resolutions, such as the blue curve (150 samples) and the orange curve (250 samples), exhibit a coarser spectral representation with increased fluctuations in higher normalized-frequency regions. These patterns are consistent with the expected effects of subsampling and reduced frequency resolution. The 150-sample curve shows greater variability across the upper portion of the spectrum under the same scaling.

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.

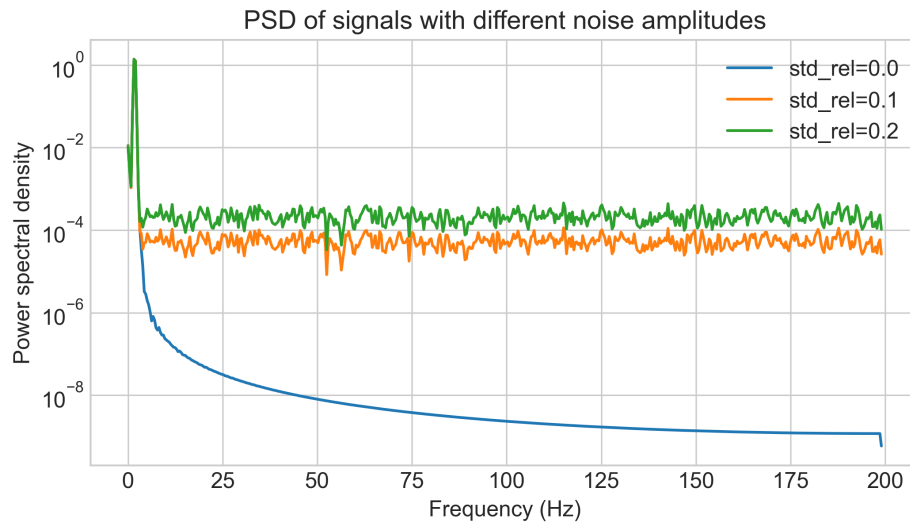
## Impact of Noise on Frequency Characteristics

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

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



**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).



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

280 comparatively stable in these plots.

281  
282 Across these settings, the low-frequency region changes less than the higher-frequency region in these estimates. This  
283 observation provides context for the subsequent super-resolution benchmark, where both time-domain and frequency-domain  
284 metrics are reported.

## 285 Usage Notes

286 The CoSiBD dataset contains high-resolution signals and corresponding subsampled versions at multiple resolutions. Signals  
287 are provided in consolidated `.txt`, `.npz`, and `.json` formats. Pairing between low- and high-resolution versions is per-  
288 formed by row index: row  $i$  in a subsampled file corresponds to row  $i$  in the high-resolution file, with per-signal parameters  
289 available in `signals_metadata.json`. The dataset is distributed as a single, unified collection without a predefined  
290 train/validation/test split. Users can create partitions appropriate to their objectives (e.g., random splits, stratified splits by noise  
291 type or signal characteristics, cross-validation, or scenario-specific test sets), using the provided metadata to support principled  
292 partitioning.

## 295 Reading the Data

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

300  
301 The following example illustrates how to load paired low- and high-resolution signals using standard Python tools:

```
302 import numpy as np
303
304 # Load subsampled (simple decimation) and high-resolution signals
305 # Each .txt file is consolidated: one signal per row
306 x_lr = np.loadtxt('SignalBuilderC/data/signals_subsampled_simple_250.txt')
307 x_hr = np.loadtxt('SignalBuilderC/data/signals_high_resolution_5000.txt')
308
309 # Access a paired signal by row index
310 i = 0
311 low_res_signal = x_lr[i]
312 high_res_signal = x_hr[i]
```

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

## 315 Visualizing Paired Signals

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

```
317 import matplotlib.pyplot as plt
318 import numpy as np
319
320 # Visualize a paired low- and high-resolution signal
321 i = 0
322 plt.figure(figsize=(10, 4))
323
324 # High-resolution signal
325 plt.plot(high_res_signal, label='High-resolution (5000 samples)', alpha=0.8)
326
327 # Low-resolution signal (aligned to HR index range for visualization)
328 lr_x = np.linspace(0, len(high_res_signal), len(low_res_signal))
329 plt.scatter(lr_x, low_res_signal, color='red', s=12,
```

```

330         label='Low-resolution (250 samples)')
331
332 plt.xlabel('Sample index')
333 plt.ylabel('Amplitude')
334 plt.title('Paired Low- and High-Resolution Signal')
335 plt.legend()
336 plt.grid(True)
337 plt.tight_layout()
338 plt.show()

```

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

## 342 Code availability

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

345  
346 The repository provides a modular Python package (`SignalBuilderC`) implementing all stages of the dataset construction  
347 process, including: (i) generation of high-resolution synthetic temporal signals with configurable frequency profiles and  
348 amplitude envelopes; (ii) deterministic creation of paired low-resolution signals via uniform subsampling; (iii) optional noise  
349 injection; and (iv) export of signals and associated metadata in NumPy (`.npz`), plain-text (`.txt`), and JSON (`.json`) formats.  
350 The codebase is documented and includes example scripts and notebooks illustrating dataset generation, regeneration from  
351 metadata, and basic data access.

352  
353 All source code is released under the MIT License, allowing reuse and extension of the generation framework for research and  
354 benchmarking purposes.

355 The CoSiBD dataset itself is published separately on Zenodo and is cited in the Data Records section<sup>25</sup>. The Zenodo record  
356 distributes the dataset under the Creative Commons Attribution 4.0 International (CC BY 4.0) license.

## 357 References

- 358 1. Karacan, I. & Coauthors. A comparison of electromyography techniques: surface versus intramuscular recording. *J.*  
359 *Electromyogr. Kinesiol.* **34**, 123–134, [10.1016/j.jelekin.2024.123456](#) (2024).
- 360 2. Nayak, S. K. *et al.* A review of methods and applications for a heart rate variability analysis. *Algorithms* **16**, 433,  
361 [10.3390/a16090433](#) (2023).
- 362 3. Shaffer, F. & Ginsberg, J. P. An overview of heart rate variability metrics and norms. *Front. Public Heal.* **5**, 258,  
363 [10.3389/fpubh.2017.00258](#) (2017).
- 364 4. Chen, S.-W. Non-uniform sampling data converters: A journey to uncharted circuits and systems. In *2022 International*  
365 *Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 1–1, [10.1109/VLSI-DAT54769.2022.9768053](#) (2022).
- 366 5. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding deep learning requires rethinking generalization.  
367 *arXiv preprint arXiv:1611.03530* [10.48550/arXiv.1611.03530](#) (2016).
- 368 6. Bhatia, H. *et al.* Machine-learning-based dynamic-importance sampling for adaptive multiscale simulations. *Nat. Mach.*  
369 *Intell.* **3**, 401–409, [10.1038/s42256-021-00321-8](#) (2021).
- 370 7. Mallat, S. G. A theory of multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern*  
371 *Analysis Mach. Intell.* **11**, 674–693, [10.1109/34.192463](#) (1989).
- 372 8. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444, [10.1038/nature14539](#) (2015).
- 373 9. Goodfellow, I. J. *et al.* Generative adversarial networks. *arXiv preprint arXiv:1406.2661* [10.48550/arXiv.1406.2661](#)  
374 (2014).
- 375 10. Isasa, I. *et al.* Comparative assessment of synthetic time series generation approaches in healthcare: leveraging patient  
376 metadata for accurate data synthesis. *BMC Med. Informatics Decis. Mak.* **24**, Article 27 (2024).

- 377 11. Schumaker, L. L. *Spline Functions: Basic Theory* (Springer-Verlag, New York, 2007), 3rd edn.
- 378 12. Boor, C. D. *A Practical Guide to Splines* (Springer-Verlag, New York, 2001).
- 379 13. 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](https://doi.org/10.1145/3559540) (2023).
- 380
- 381 14. 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](https://doi.org/10.1029/2023MS003658) (2023).
- 382
- 383 15. 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](https://doi.org/10.1038/s41592-023-02138-w) (2024).
- 384
- 385 16. 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](https://doi.org/10.1038/s41587-025-02553-8) (2025).
- 386
- 387 17. O’Shea, T. J. & West, N. Radio machine learning dataset generation with GNU radio. In *Proceedings of the GNU Radio Conference*, vol. 1 (2016).
- 388
- 389 18. DeepSig. Datasets (including radioml 2016.10a). <https://www.deepsig.ai/datasets/>. Accessed 2026-01-13.
- 390 19. DeepSig. Radioml 2018.01a dataset. <https://www.deepsig.ai/datasets/>. Accessed 2026-01-13.
- 391 20. 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](https://doi.org/10.1109/TBME.2003.808805) (2003).
- 392
- 393 21. McSharry, P. & Clifford, G. D. ECGSYN: A realistic ecg waveform generator (physionet). <https://physionet.org/physiobank/physiotools/ecgsyn/>. Accessed 2026-01-13.
- 394
- 395 22. 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](https://doi.org/10.1016/j.jneumeth.2018.08.001) (2018).
- 396
- 397 23. Pinceti, A., Sankar, L. & Kosut, O. Generation of synthetic multi-resolution time series load data. arXiv:2107.03547 (2021).
- 398
- 399 24. 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](https://doi.org/10.1371/journal.pone.0304981) (2024).
- 400
- 401 25. Ibarra-Fiallo, J., Lara, J. A. & Agudelo Moreno, D. Cosibd, [10.5281/zenodo.18295713](https://doi.org/10.5281/zenodo.18295713) (2025). Version v2. Dataset.
- 402 26. 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](https://doi.org/10.1109/TAU.1967.1161901) (1967).
- 404

## 405 Acknowledgments

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

## 408 Author Contributions

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

## 412 Competing Interests

413 The authors declare no competing interests.