SURVEY

Open Access

Evaluation is key: a survey on evaluation measures for synthetic time series

Michael Stenger[1*], Robert Leppich[1], Ian Foster[2], Samuel Kounev[1] and André Bauer[1,2]

*Correspondence:
michael.stenger@uni-wuerzburg.de

[1] Faculty of Mathematics and Computer Science, University of Würzburg, Würzburg, Germany
[2] Department of Computer Science, University of Chicago, Chicago, USA

Abstract

Synthetic data generation describes the process of learning the underlying distribution of a given real dataset in a model, which is, in turn, sampled to produce new data objects still adhering to the original distribution. This approach often finds application where circumstances limit the availability or usability of real-world datasets, for instance, in health care due to privacy concerns. While image synthesis has received much attention in the past, time series are key for many practical (e.g., industrial) applications. To date, numerous different generative models and measures to evaluate time series syntheses have been proposed. However, regarding the defining features of high-quality synthetic time series and how to quantify quality, no consensus has yet been reached among researchers. Hence, we propose a comprehensive survey on evaluation measures for time series generation to assist users in evaluating synthetic time series. For one, we provide brief descriptions or - where applicable - precise definitions. Further, we order the measures in a taxonomy and examine applicability and usage. To assist in the selection of the most appropriate measures, we provide a concise guide for fast lookup. Notably, our findings reveal a lack of a universally accepted approach for an evaluation procedure, including the selection of appropriate measures. We believe this situation hinders progress and may even erode evaluation standards to a "do as you like"-approach to synthetic data evaluation. Therefore, this survey is a preliminary step to advance the field of synthetic data evaluation.

Keywords: Survey, Synthetic Data, Time Series, Measures, Evaluation

Introduction

Time series are ubiquitous. For instance, (IoT) sensors in factories, medical machinery in hospitals, personal smart devices, or financial transaction systems produce these real-valued, sequential data, leading to a seemingly unlimited pool of data to learn from. Indeed, various research problems surround time series, including forecasting [1], classification [2], and anomaly detection [3]. Over the last decades, another task has emerged involving the creation of even more data with a given set of desired properties: Time series synthesis [4]. The synthesis is vital as there are circumstances limiting the availability or usability of real data, such as time series collected in the healthcare sector [5]. While sharing datasets is common practice, for instance, to ensure reproducible results, this has to be done with extreme care for medical datasets, as simply publishing the data as-is can endanger patient privacy. One way to reliably prevent this is by training a model on the real data capturing all relevant properties, that is, the data distribution, and artificially creating new patient data samples. We call this process synthesis or generation, and in this work, we are interested in time series synthesis specifically. While some synthesis tasks are sufficiently addressed with straightforward methods like averaging two real time series [6] or applying time warping to an original sample [7], others with complex data distributions and patterns require deep neural networks to model and reproduce the data properties [8]. In any case, the evaluation of the generated data is always crucial.

The aim of our work is not the design and implementation of a new generation method but rather to provide means for the fundamental problem of evaluating the synthesis. In other words, we want to facilitate the following research questions.

1. How can the quality of a synthesized time series be determined?
2. What qualifies a (set of) synthesized time series as being "good"?
3. What are the most effective methods to evaluate quality of synthesized time series?

Currently, there is no consensus within the research community on how to answer these questions satisfactorily [4, 9, 10]. Instead, a plethora of evaluation approaches exists, with each using a unique set of measures. Hence, the evaluation of generative models in general and time series generators in particular is widely considered an active area of research. It presents formidable challenges owing to several inherent complexities. First, the absence of a definitive ground truth poses a significant hurdle, as there is no authoritative benchmark against which generated time series can be objectively evaluated. Second, assessing the synthesis quality becomes a multidimensional task, encompassing various aspects such as fidelity, diversity, generalizability, and privacy considerations. Therefore, a holistic measure has to consider many quality criteria, while more specific measures may only deliver a complete view in combination. Furthermore, designing universally applicable evaluation measures becomes intricate, as the criteria for success may vary across different applications, demanding a nuanced and adaptable approach to assessment in synthetic time series generation. Last, the challenges are exacerbated in evaluating synthetic time series due to the absence of an intuitive understanding of the data, as opposed to image, video, or even audio data. While synthetic images, for instance, can be plotted and quickly evaluated by human judges at least qualitatively, time series are often too noisy, long, or high-dimensional to be effectively analyzed visually. Consequently, this study aims to provide an overview and analysis for experienced researchers. In addition, this work can be an entry point to the field for novices to time series synthesis and generally a foundation for unifying the evaluation process in the future.

The contributions of this work can be summarized as follows.

- This is the first comprehensive review of evaluation measures for synthetic time series. In this work, we collect 83 measures from 56 works targeting time series data or general measures that apply to time series.

- To promote organization of knowledge, we propose a taxonomy of these measures under different aspects of quality in order to structure this long list.
- We offer a valuable analysis of the usage statistics of evaluation measures in the literature, examining their applicability with respect to conditional generators, embeddings, time series length, the number of channels, and dataset sizes.

All this introduces novices to the topics of time series generation and its evaluation while still providing value to experienced colleagues as a reference and summary. For those interested in a quick selection of measures, we refer to Table 1 as a guidepost.

The remainder of this survey article is structured as follows: In "Related work" section, we introduce previous reviews and explain what sets our work apart from them. In "Approach" section, we describe our approach regarding both the review and analysis part. In "Evaluation measures for synthetic time series" and "Analyzing evaluation measures" sections, we present the findings of the literature review and subsequent analysis, respectively. In Sect. "Conclusion", we conclude the survey and outline further research directions.

Related work

This chapter provides an overview of closely related or significant prior surveys and comparative studies on the evaluation of synthetic time series and related data types.

Evaluation in related fields of data synthesis

The evaluation of generative models for the synthesis of time series and other data types, such as images or text, is considered an open problem by many researchers [4, 11–13]. To gain a wider awareness of the current standing in related fields of data synthesis, we provide a selection of works on generative adversarial networks (GANs), sequential data in general, software libraries and frameworks, tabular data, audio, and text.

A few years ago, Xu et al. [14] already set out to address the problem of how to "evaluate the evaluation metrics" for GANs in particular, with a focus on images. Firstly, they introduced six generator-agnostic measures, namely inception score (IS),Fréchet inception distance (FID), Wasserstein-1 distance (WD), mode score, kernel 112 maximum mean discrepancy (MMD), and classifier two-sample test (C2ST). Furthermore, the authors briefly describe four conditions they consider necessary for measures to adhere to. These are distinctiveness, robustness to transformations, efficiency, and detecting overfitting. An experiment is conducted for each condition and presented measure to check if the latter suffices said conditions. In a concluding discussion, the strengths and weaknesses of the six measures are outlined. While insightful, the list of measures considered is minimal and outdated. Many novel measures have been presented in the meantime, especially data type-agnostic ones, which can be adapted to any type of data. Overall, our focus is pointed toward time series, not images.

In a recent article, Brophy et al. [4] reviewed state-of-the-art GAN models for time series generation, augmentation, and imputation. They provide background on the workings of this learning paradigm and a classification of methods. The work primarily covers the latest popular architectures but also features a short section on evaluation strategies for this type of model. The overview contains some elementary measures, mostly applicable only to individual time series but not datasets. As presented there, it is a listing of popular measures used, but neither a complete nor critical one. Furthermore, the categorization of knowledge is limited to "quantitative" and "qualitative".

Lately, Eigenschink et al. [15] proposed an evaluation framework for (mainly deep) generative models for synthesizing all kinds of sequential data: Text, audio, video, and time series. They aimed to overcome the isolated evaluation of generators in these areas and present a universally applicable set of criteria to check. Namely, they argue for five categories: Representativeness, novelty, realism, diversity, and coherence. The paper then goes into the current usage of measures in and the relevance of each category for the different data types. However, their table of measures for synthetic time series is very sparse, with six works reviewed and 13 measures found and limited to healthcare and mobility.

Recently, Borji [11] surveyed recent developments in GAN evaluation measures research, updating a previous paper on the topic [16]. Again, the work is not restricted to the time series domain but to one class of generation methods. Moreover, the focus is on image generation as it dominates research on synthetic data in general. It furnishes an extensive array of measures accompanied by concise explanations and numerous generated images. The measures are organized into sections based on their similarity rather than grouping them by applicable data types. No additional structure is provided. Thus, those relevant to time series need to be tediously searched for, while most measures we found were not covered there. Lastly, our analysis is tailored toward time series.

For the field of tabular data synthesis, Dankar et al. [17] introduced a scheme of four abstract criteria generated data may be tested on to demonstrate utility to end users: attribute fidelity, bivariate fidelity, population fidelity, and application fidelity. For each criterion, a representative measure was selected by the authors based on popularity and consistency after reviewing relevant literature. Still, only tabular data is considered here. Most of those measures cannot trivially be applied to time series.

Another related field of synthetic data is audio generation. Deep learning (DL) techniques have been used excessively to generate artificial music inspired by human-made samples. In a survey by Ji et al. [18], past developments and future directions of DL-based generators are presented, including typical evaluation measures of the domain. The authors found that there is no unified evaluation criterion, neither present nor to be expected, as music is a form of art and thus made to appeal to humans. Instead, there are various suggestions for how to approximate human evaluation.

The division of evaluation strategies into human assessment and machine-computed measures can equally be found in text generation. In a survey by Fatima et al. [19], however, the evaluation by human language and domain experts is attributed little importance. This is due to the intensive labor and cost involved as well as subjectivity and susceptibility to human error, while quantitative measures are correlated well enough with human perception. Another study conducted by Iqbal and Qureshi [12] also sees a variety of evaluation measures present. However, it concludes that evaluation is still an open research problem, as many methods poorly correlate with human assessments.

Assefa et al. [20] presented a review of the current standing and developments of synthetic data generation in finance, which addresses the evaluation of the similarity of real and generated datasets, among other things. They mention four works relevant to evaluating time series, while our survey not only considers these works but is domain-independent, includes far more measures, and analyses them.

Recently, Figueira and Vaz [13] published a survey on the generation and evaluation of tabular data with a focus on GANs. In a short section on evaluation measures, they list only seven that are also applicable to time series data, which our work contains as well. They do not provide sources for some of the measures and refrain from significant further analysis.

Summary of delimiters

Above, we identified several mismatches between this survey and prior works. Most address specific domains or data types other than time series [12, 14, 18, 19], while a transfer of findings is non-trivial. Another one addresses the evaluation of sequential data generation, but in too broad a scope to provide detailed insights for time series [15]. Others are tailored towards images or are data type-agnostic but limit their scope to GANs [4, 21]. Compared to the reviewed articles, our study goes into more detail in the time series domain, both in terms of works covered and subsequent organization of knowledge.

Approach

We first describe the terminology we use in our study in Subsection "Terminology". Afterwards, we outline our approach to literature search, selection, and the subsequent analysis part in Subsection "Acquisition and systematization of knowledge".

Terminology

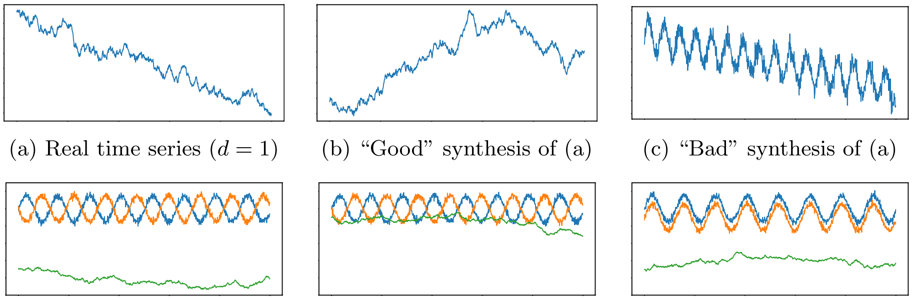
This section briefly defines the key terminology used in this work.

Definition 1 A "measure" is a qualitative or quantitative scoring function that assigns some sort of numerical or categorical value of quality to synthetic data or its generator.

In some works, the notation "metric" is used instead of measure. However, we opted for the latter since the former encompasses mathematical characteristics that might only hold for some of the proposed measures found in the literature.

Definition 2 Let $D_r = \{x_1, \ldots, x_n\} \subseteq \mathcal{X}$ be a dataset of individual data samples x_i from data space $\mathcal{X}$ collected by observing some real-world environment. D_r follows a distribution P_r, that is, $x_i \sim P_r$. A synthesis or generation of a new dataset $D_g = \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ or stream means to apply an (unconditional) generator function $g_\theta : Z \to \mathcal{X}$ m times. Z is some input that can trigger the generation of the desired samples and is typically just Gaussian noise. θ is a set of parameters that are learned using D_r. We call D_g synthetic or generated data.

For convenience, we reference additional literature on statistical and deep learning-based data synthesis by [22] and [23], respectively. Examples of classic time series models include SARIMA and Holt-Winter [24], while deep learning models often utilize recurrence [25], adversarial learning [26], or autoencoding [27].



(a) Real time series ($d = 1$)

(b) "Good" synthesis of (a)

(c) "Bad" synthesis of (a)

(d) Real multivariate time series ($d = 3$)

(e) Synthesis of (d), green channel is too high

(f) Synthesis of (d), orange and blue channels are positively correlated

Fig. 1 Illustration of six exemplary time series. The top row depicts univariate sequences, where **a** is the exemplary real time series and the remaining two its syntheses. **b** can be considered a successful generation, demonstrating the same chaotic structure. **c**, however, has a clear seasonal pattern and linearly falling trend. The bottom row contains three multivariate time series with three channels (blue, orange, and green). **d** was sampled from the real data distribution,**e**, **f** from different synthetic ones. In this case, both are partially realistic: e accurately depicts the negative correlation between the blue and orange channel but with the green channel intersecting them. In (**f**), the correlation is positive, in contrast.

Definition 3 We say a generator g_θ is conditional if it is of the form $g_\theta : Y \times Z \rightarrow \mathcal{X}$, that is, it accepts an additional input to create a data object.

For instance, the additional input can be a class label or, specifically for time series, the previous time step $\mathbf{x}_{t-1}$.

Definition 4 Let $\mathcal{Y}$ be a stochastic process $\mathcal{Y} = \{Y_t\}$ indexed over time t with random vectors $Y_t = (Y_t^1, \ldots, Y_t^d)^T$ of dimension $d \in \mathbb{N}^+$. Then, a time series X is a sequence of realizations $\mathbf{x}_t = (x_t^1, \ldots, x_t^d)^T$ of these random vectors for some t.

In practice, a time series is simply a finite set $X = \{\mathbf{x}_t \mid t \text{ is a point in time}\}$ of observations $\mathbf{x}_t \in \mathbb{R}^d, d \in \mathbb{N}^+$, typically some measurement or event. For our purposes, we implicitly assume an ordering on X given by time t. Often, the observations are also equidistant, but not necessarily. Still, some generators and measures assume this. Also see Fig. 1 for illustrative examples.

Definition 5 Let $X = \{\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, \ldots, \mathbf{x}_{t_l}\}$ be a time series with $\mathbf{x}_{t_i} \in \mathbb{R}^d$. Then, we define the length l of X as the number of observations, $l = |X|$, and the dimensionality as d, the number of dimensions of the associated Euclidean space $\mathbb{R}^d$. Furthermore, if $d = 1$, we say X is univariate, and otherwise, that is, $d > 1$, multivariate.

We typically denote a general data space by $\mathcal{X}$, an unspecified data object by $x \in \mathcal{X}$, a time series by $X \in \mathbb{R}^{l \times d}$, and one of its channels $X^c \in \mathbb{R}^{l \times 1}$. Furthermore, $D, D_r, D_g \subset \mathcal{X}$ are datasets and $f : \mathcal{X} \rightarrow \Omega$ an embedding function from the data space into some other feature space Ω, usually $\Omega = \mathbb{R}^d$. Lastly, many evaluation measures depend on distance functions $\delta : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_0^+$ between two time series. While the authors of the reviewed works exclusively used the Euclidean distance (ED) and dynamic time warping (DTW), respectively, it should be mentioned that some measures might improve in accuracy or execution time when combined with other time series distance functions or time series models. Examples include Bag-of-SFA-Symbols (BOSS) [28], complexity invariance distance (CID)time series shapelets [29], complexity invariance distance (CID) [30], and maximum shifting correlation distance (MSCD) [31].

Acquisition and systematization of knowledge

We conducted thorough searches for previous publications related to the evaluation of time series generation and synthesis. To this end, we applied the technique of Webster and Watson [32] to conduct a comprehensive literature review. The process consisted of four steps as described below:

Keywords and Data Sources The first step consists of a broad search for statement candidates based on the following keyword sets and their permutations:

- {autoencoder, generative adversarial network, boltzmann machine, ∅}
- {time series, data}
- {synthesis, generation, ∅}
- {evaluation, measure, metric, ∅},

leading to search queries such as “time series synthesis”, “data synthesis evaluation”, or “generative adversarial network time series generation”. We used Google Scholar,[1] the IEEE Xplore,[2] and ACM DL digital libraries.[3]

Conducting/Filtering In the second step, we went through the first 30 results of each search query in the order they were returned in, and filtered out papers whose titles and short summaries were not connected to time series generation or time series synthesis evaluation, that is, being false positives. Then, we read the abstract of the remaining papers and again selected all mentioning at least one of the keywords above. This resulted in 56 papers that were studied.

Extracting Data As a third step, we performed a detailed review of the selected papers, extracting name and definition of the measures contained as well as other information relevant to our analysis, such as year of publication, the evaluation goals, and code availability. For the measure names, we use abbreviations from the original work if possible, and otherwise our own where deemed appropriate.

Synthesizing Knowledge Certainly, there are many ways to approach the analysis of the collection of measures found. Hence, we list the nine dimensions considered in this study in Table 1 and briefly justify each choice. They are ordered by appearance. For someone interested in finding a suitable measure quickly, this may serve as a guidepost. We suggest starting with the dimension of highest personal priority, going to the respective location in this article, and making an initial selection of measures. Afterwards, the reader may return to this table, select the next-most relevant dimension, and refine the selection repeatedly until satisfaction.

[1] URL: https://scholar.google.de/.

[2] URL: ieeexplore.ieee.org/Xplore/home.jsp.

[3] URL: dl.acm.org.

Table 1 Guidepost for measure selection

Dimension	Location	Description
Sample-/distribution level	Sec. "Evaluation measures for synthetic time series"	While most measures operate on entire datasets, some can produce scores for individual samples as well. Hence, we partition the presentation of measures into distribution-level measures, comparing entire datasets, in Subsection "Distribution-level measures" and sample-level measures, evaluating individual synthetic time series, in Subsection "Sample-level measures".
Dependence on embeddings	Sec. "Evaluation measures for synthetic time series"	Many measures, especially in the specialized evaluation literature, operate in the real vector space rather than on time series. In such cases, training a time series embedding model on each dataset is necessary to prepare real and synthetic data to be received by a measure. However, this step certainly yields additional effort.
Code availability	Tables 2 & 5	Naturally, researchers and practitioners alike are interested in a fast and easy evaluation. Having a publicly available implementation of the measure helps to facilitate that.
Conditional Generation	Tables 2 & 5	Some generators can be conditioned on additional inputs, such as class labels, resulting in labeled datasets. Hence, it is essential to differentiate measures designed for such data from those that were not. Also see Definition 3.
Miscellaneous limitations	Tables 2 & 5	It is also important to know whether a measure imposes further restrictions regarding data type, generator, additional inputs, and the like. These can be found in Column "Applicability".
Categorization	Subsec. "Taxonomy of evaluation measures and criteria"	The collection of measures is quite extensive. Hence, we propose a structure on said collection based on what we believe to be the primary concern for selecting a measure: The quality aspect(s) assessed by a measure, which we denote by evaluation *criterion*, such as fidelity or diversity.
Community adaptation	Subsec. "Theory and practice of evaluation measures"	Popularity and publishing context are likely factors influencing the selection of measures. Hence, we analyze the impact measures have via the number of their reuses. To account for context, we propose differentiating the reviewed works into two groups.
Result classification	Figure 22	Moreover, we report the type of result as either quantitative or qualitative and integrate it into our categorization. In the latter case, we also include range and optimum, which can be found in the measure's description in Section "Evaluation measures for synthetic time series".
Quantitative constraints	Subsec. "Requirements on the input data format"	Measures often impose limitations on the data they can operate on, specifically regarding time series length, dimensionality, and the size of input datasets. For instance, they must have constant length, be multivariate, or have a minimum amount of samples in each dataset.

Overview of our approach to analyzing the measures found and guidepost for measure selection. Listed are the dimensions considered for analysis, their location in the article, and their description

Evaluation measures for synthetic time series

As the list of measures proposed in the past is quite extensive, we subdivide them into two groups based on whether they can be applied to individual synthetic samples (referred to as "sample-level") or only directly on a reasonably sized dataset (referred to as "distribution-level"). Hence, if a measure is calculated for a single real time series but requires an entire synthetic dataset, we deem it distribution-level. At the same time, this is a soft differentiation, meaning that many sample-level measures can be adapted towards the distribution-level and vice versa.

Distribution-level measures

Below, we provide an introduction to each distribution-level measure found. This list is sorted alphabetically to foster a faster look-up for non-sequential reading. A summary can be found in Table 2.

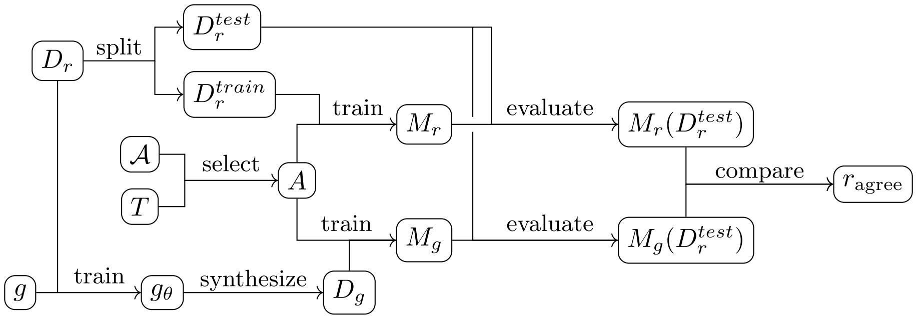


Fig. 2 Flow chart for measure agreement rate. The illustration depicts the application of agreement rate to real dataset D_r and generator function g. $\mathcal{A}$ denotes the set of ML algorithms, T the given task, and M_x a trained ML model for algorithm A. This figure serves as an orientation for other train on synthetic, test on real (TSTR)-based measures as well

Agreement rate. This measure tracks the usability of the synthetic data in a downstream machine learning (ML) task, namely an improvised classification [33]. Figure 2 depicts the computation of this measure with the help of a flow chart. Generally speaking, an ML algorithm A is trained on synthetic data D_g and evaluated on real D_r to judge D_g's practical value. More specifically, each data sample is annotated by setting one of its own features (an attribute of tabular data, a channel for time series, etc.) as a class label. If the model is conditional, this step is redundant. Then, the real dataset is split into train and test components, $D_r = (D_r^{train}, D_r^{test})$. Afterwards, a classifier is trained separately on D_r^{train} and the synthetic D_g, yielding two models, M_r and M_g, respectively. Finally, the agreement rate r_{agree} is defined as the fraction of D_r^{test} for which both models make the same class assignments. More formally, it is

$$r_{\text{agree}} := \frac{1}{|D_{test}|} \sum_{x \in D_{test}} \mathbf{1}\{M_r(x) = M_g(x)\}, \tag{1}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. The rate falls in [0, 1], with higher being better. An embedding is implicitly given by adjusting the classifier's input layers for differently shaped inputs.

Algorithm comparison. Given real and synthetic datasets D_r and D_g, respectively, Lin et al. [34] first create train-test-partitions $D_r = (D_r^{train}, D_r^{test}), D_g = (D_g^{train}, D_g^{test})$. For a downstream task T, of which one assumes it might be a good indicator for the utility of the synthetic data, they select a group $\mathcal{A}$ of algorithms and an appropriate performance score s for T. Now, algorithm comparison is a measure that evaluates a synthesis on whether it preserves the ranking within $\mathcal{A}$ with regard to s when applied to task T. To give an example, T might be the classification of time series, the group of algorithms be given by $\mathcal{A} = \{$ multi-layer perceptron, linear support vector machine, naive Bayes, decision tree, logistic regression$\}$, and s be the F1 score. Note that this setting is only applicable to labeled datasets, that is, conditional generators. A flow chart illustrating the procedure is given in Fig. 3. To apply this measure, they train each algorithm on D_r^{train} and D_g^{train}, which returns two sets of trained models $\mathcal{M}_r$ and $\mathcal{M}_g$. Second, Lin et al. [34] evaluate all models $M_r \in \mathcal{M}_r$ on D_r^{test} and $M_g \in \mathcal{M}_g$ on D_g^{test}. As a result, they obtain the same number of scores, grouped in two sets as well. To this pair of sets, one can apply

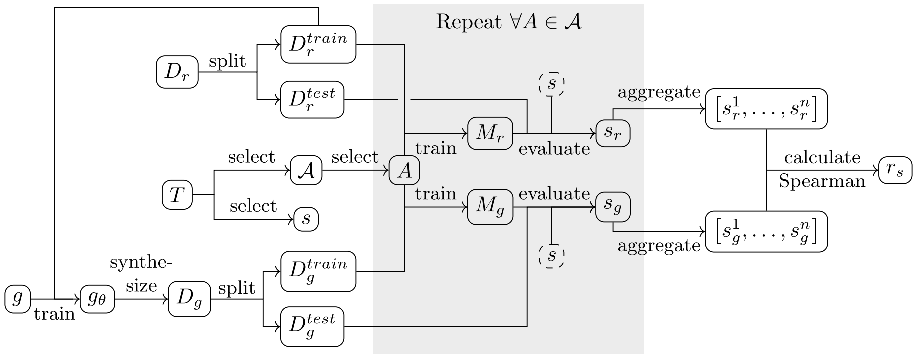


Fig. 3 Flow chart for measure algorithm comparison. Following Fig. 2, this flow chart visualizes the application of algorithm comparison to real dataset D_r and generator function g. T denotes the downstream task, $\mathcal{A}$ the chosen set of ML algorithms, s a performance measure for T, M_x a trained model, and finally s_x a score. $n := |\mathcal{A}|$

the Spearman rank correlation coefficient, which is the Pearson correlation coefficient (PCC) between the ranks of each model's score within each group. As a result, we get a scalar value $r_s \in [-1, 1]$, indicating to what degree each algorithm performs equally on the generated data relative to the other algorithms, compared to their performance on real data. The higher it is, the better.

Annealed importance sampling (AIS) is a quantitative measure for evaluating log-likelihoods for decoder-based generative models [35]. By that, we mean parameterized neural networks defining a generative distribution by transforming samples from some simple distribution to the data manifold. AIS is a Monte Carlo algorithm commonly used to estimate normalizing constants, and employed here to estimate the log-likelihood $\log p(\mathbf{x}_{test})$ the model assigns to a held-out test sample [36]. The measure requires an embedding $f : \mathcal{X} \to \mathbb{R}^k$ as well as a held-out test set (AIS $\in \mathbb{R}$, the higher the better).

Approximate entropy (ApEn). This is a channel-wise entropy measure specifically for time series used to determine their regularity and complexity. For more details, we refer to [37]. In [38], the measure is applied to all available real and synthetic data, while Leznik et al. [9] sample subsets $R \subseteq D_r$ with $|R| = 500$ and $G \subseteq D_g$ with $|G| = 10$. The latter approach is used to guide GAN training. ApEn yields a score for each channel of every sample in both the real and synthetic dataset. To retrieve a final measure, the individual scores are aggregated first via calculating the mean for each channel over all samples in each dataset separately, and then applying the squared difference between scores on real and synthetic time series over the channels. A smaller value indicates better performance.

Augmentation test. This measure is designed to probe the usefulness of the synthesized data D_g for augmenting the real data D_r [39]. To this end, the authors split the real data into training, validation and test sets $D_r = (D_r^{train}, D_r^{val}, D_r^{test})$. The generator only gets to learn from the training set to create D_g. Next, the augmented dataset D_{aug} is created by merging D_r^{train} with D_g, $D_{aug} := D_r^{train} \cup D_g$. Furthermore, an ML task is defined to measure the usefulness of D_{aug} and an appropriate algorithm A selected. Training is conducted on D_{aug} to learn a model M_1 and separately on D_r^{train} for a baseline model M_2, while validation is performed on D_r^{val} in both cases. Validation and testing is conducted with respect to some measure s appropriate for the selected task. In the context of their work, the authors chose a classification task as they had a labeled dataset. In principle, other tasks that do not require labels are also imaginable. For instance, Jeha et al. [40] employ prediction, but without reference to [39]. The performance of M_1 and M_2 is measured in terms of precision and area under ROC curve (AUROC) on the test set. For D_g to be useful for augmentation, we expect $s(M_1) > s(M_2)$, the greater the difference the better. The applicability to labeled datasets depends on the choice of the task.

Average cosine similarity (ACS). This measure compares pairs of real and synthetic time series with respect to their cosine similarity and computes a score for each class by averaging [41]. In this setting, class labels for both datasets are required. Let D_r and D_g be real and synthetic datasets. Instead of calculating the cosine similarity directly on the time series, the authors define an embedding $f : \mathbb{R}^{l \times d} \rightarrow \mathbb{R}^{7 \cdot d}$ for time series of length l and dimensionality d, which extracts a vector of seven features from each of the d channels and concatenates them. Namely, these are median, mean, standard deviation, variance, root mean square, maximum, and minimum. For each class c, ACS is defined by

$$\mathrm{ACS}_c := \frac{1}{\left|D_r^c\right| \cdot \left|D_g^c\right|} \sum_{X \in D_r^c} \sum_{\hat{X} \in D_g^c} \frac{f(X) \cdot f(\hat{X})}{\left|\left|f(X)\right|\right|_2 \cdot \left|\left|f(\hat{X})\right|\right|_2}, \tag{2}$$

where D_r^c, D_g^c denote the samples of class c in dataset D_r and D_g, respectively ($\mathrm{ACS}_c \in [-1, 1]$, the higher the better).

Average euclidean distance (AED) focuses on the distribution of amplitudes in the frequency domain within the synthetic dataset [42]. The goal is to ensure preservation of inter-channel correlation within the synthetic time series. The measure targets those with two channels exclusively. To do so, they first transform all synthetic samples into the frequency domain and extract the most likely amplitude in both channels of each sample. Then, they interpret the two amplitudes of each sample as a coordinate in the 2D plane. This allows them to compute the distance to the line through the origin with slope 1. However, the calculation may fail on time series without any seasonality.

Average Jensen-Shannon distance (JS distance) computes the distance between the distribution of each feature over the real dataset and the distribution of the corresponding feature over the synthetic dataset [41]. More precisely, the measure assumes a partition of the datasets into (the same) classes, with the score being computed for each class separately. To this end, they first define the feature vector $f(X)$ for each time series X via $f : \mathbb{R}^{l \times d} \rightarrow \mathbb{R}^{7 \cdot d}$, which is the same embedding used for ACS above. The seven features extracted are identical as well. Now, one can determine the distance between the distributions of each of the $7d$ features and take the average to arrive at a score AJSD_c for class c as

$$\mathrm{AJSD}_c := \sum_{i=1}^{7d} \mathrm{JSD}'(\{f(X)_i \mid X \in D_r^c\}, \{f(\hat{X})_i \mid \hat{X} \in D_g^c\}). \tag{3}$$

JSD' is an approximation of the conventional Jensen-Shannon divergence (JS divergence) on distributions, taking just two sets of samples. Furthermore, D_r^c, D_g^c denote the samples of class c in dataset D_r and D_g, respectively. It holds $\mathrm{AJSD}_c \in \mathbb{R}_{\geq 0}$, the lower the better.

Average Wasserstein distance (AWD) compares the distribution of amplitudes in the frequency domain within the real dataset to that of the synthetic dataset [42]. The goal is to ensure the diversity with respect to the syntheses' periodicity. To this end, fast Fourier transform (FFT) is used to determine the most likely period in each channel of each sample. Afterwards, the WD between the amplitudes extracted from real dataset on one side, and synthetic dataset on the other, is calculated for each channel, and finally averaged across channels. For the resulting real-valued score holds lower is better. However, the calculation may fail on time series without any seasonality and an order over the samples in both datasets is required, but not provided.

Classification accuracy score (CAS) is an evaluation method for conditional generative models proposed in [43]. Sometimes, they are also denoted by $g_\theta(x|y)$, where the typically random input z is left aside, and g_θ is viewed as producing samples x conditioned on an additional input y, here, a tag or class label. The idea is to train another model on the produced dataset for a downstream task, in this case classification using the class labels assigned during generation. The accuracy of this model on a held-out, real test set is the CAS for the generator. Compared to the score computed with a classifier trained on the real dataset, this measure tells if the utility of the synthesis is on par with the original. The entire procedure is depicted in Fig. 4. The closer the two are, the better the synthesis. Furthermore, it provides insight into problems with generating specific classes.

Classifier two-sample test (C2ST) is no measure specialized on the evaluation of synthetic data, but rather assesses whether two sets of data points are sampled from the same distribution [44]. As the name suggests, this is realized through a binary classifier $c : \mathcal{X} \to [0,1]$ combined with a hypothesis test. The basic procedure is also illustrated in Fig. 5. Specifically for the evaluation of generative models, these two sets are real dataset D_r sampled from P and synthetic set D_g sampled from Q. First, the authors split off a substantial part of D_r to train the generator on, denoted by D_r^{gen}. Afterwards, they split the remaining part of D_r and D_g into two parts each, one for training the classifier and the other for calculating the p-value of the hypothesis test based on the classification accuracy of c. This leaves them with $D_r = (D_r^{gen}, D_r^{train}, D_r^{test}), D_g = (D_g^{train}, D_g^{test})$ with $|D_r^{train}| = |D_g^{train}|, |D_r^{test}| = |D_g^{test}|$. Additionally, each data point in D_r^{train}, D_r^{test} is assigned label 0 and, similarly, D_g^{train}, D_g^{test} label 1. After training c, one applies the model to $D_{test} := D_r^{test} \cup D_g^{test}$ and obtain classification accuracy

$$A_{test} := \frac{1}{|D_{test}|} \sum_{(z,l)\in D_{test}} \mathbf{1}\left\{\mathbf{1}\left\{c(z) > \frac{1}{2}\right\} = l\right\}, \tag{4}$$

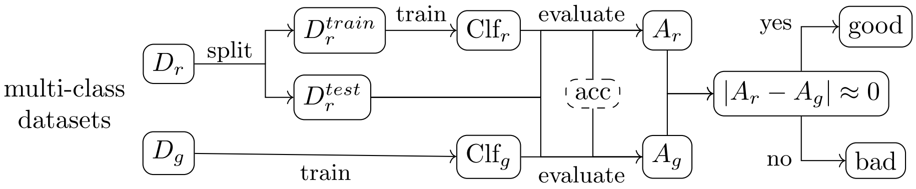


Fig. 4 Flow chart for measure CAS. The control flow goes from left to right. *D*s represent datasets, "Clf"s classifier models, "acc" the accuracy measure itself, and *A*s the actual accuracy values

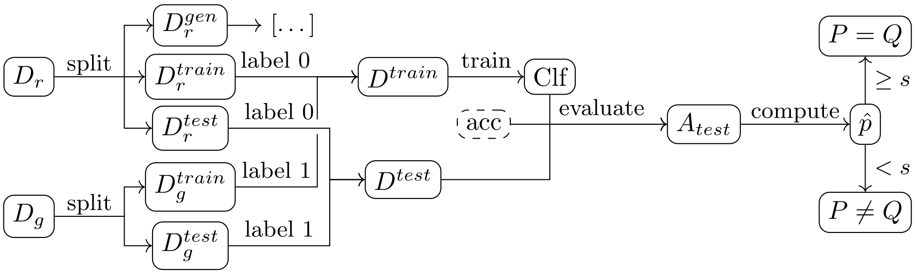


Fig. 5 Flow chart for measure C2ST. Depicted are only the major parts of C2ST. Starting on the left, both input datasets are split into multiple subsets, before being used for training the generator, training the classifier "Clf", or evaluating it, respectively. This is done with accuracy "acc". Afterwards, the hypothesis test is applied for significance level s to get p-value $\hat{p}$ on the right

which acts as the two-sample test statistic. z is a data point, l its label, and $\mathbf{1}\{\cdot\}$ the indicator function. If indeed $P = Q$, A_{test} should be around $\frac{1}{2}$, indicating a random assignment of samples to classes. For a predefined significance level s, one can now compute p-value $\hat{p} = \Pr(A \geq A_{test} \mid P = Q)$, that is, the probability that the test returns an accuracy A as high as it was, given that $P = Q$ holds. Finally, if $\hat{p} < s$, one rejects this hypothesis, otherwise we can confidently assume the equality. Besides this final result, C2ST has the additional advantage of interpretable intermediary results. On the sample-level, one can see which samples are identified with high confidence and thus cause the generator trouble, and which are already close to real data. On the distribution-level, one can also relate directly to A_{test} as a score to compare to other generators' performance. Besides, an embedding is implicitly given by the classifier.

Computational complexity. With this term, we refer to a group of measures tracking the efficiency of generators concerning resource requirements and costs incurred. Although applicable on a sample-level in many cases, the usual approach is to compute measures for many synthetic samples and break down the result by providing an expected time/resources/cost per sample. The scope can be limited to the generation process itself or may include training or preparation of the generator. Bindschaedler et al. [33] evaluate their model with respect to time taken for inference, producing over one million samples in total. They specify the experimental setup as well as the parameters tested. Kulkarni et al. [45] provided the CPU time of model training and sample inference each. Clearly, the goal is to provide generators with fast and reliable convergence during resource-aware training while allowing quick and easy generation of high-quality samples. Note that this is irrelevant to the end user of the synthetic data but can still be a limiting factor for many creators of synthetic data, which might indirectly affect quality.

Confusion matrix. This measure requires labeled real and synthetic datasets with a matching set of classes, that is, the same labels in both cases [46]. By aggregating the result of a downstream classification task in a confusion matrix, one can quickly determine classes of generated samples that cause the generator most trouble synthesizing. On the other hand, classes that are overwhelmingly correctly recalled by the classifier indicate that the generative model learned what sets such samples apart from other classes. Figure 6 contains two example matrices. First, a classifier is trained and validated on the real data. Afterwards, one samples a synthetic dataset and infers the class label

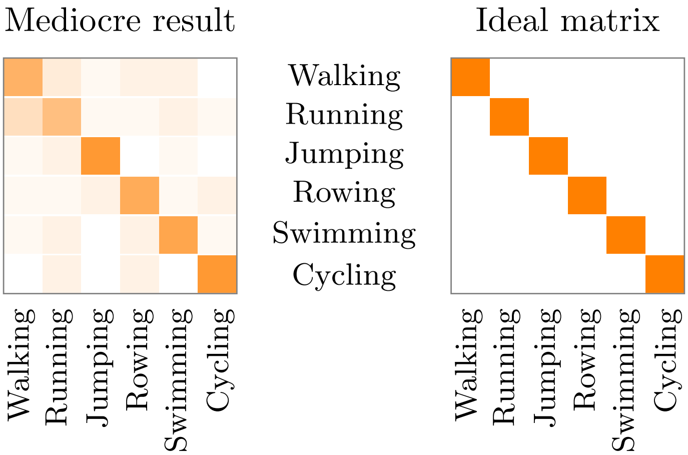


Fig. 6 Example for measure confusion matrix. Two confusion matrices are shown for the classification of labeled synthetic data objects into six classes of activities. White indicates a PCC of 0, saturated orange 1, e.g. the main diagonal in the ideal matrix

for each sample therein. Finally, we can compute and visualize the confusion matrix. Generally, a concentration of probabilities on the main diagonal is desirable. The classifier model is expected to compute its own data embedding.

Context-FID is an evaluation measure that aims to quantify the similarity of real and synthetic time series distributions following the example of the existing FID score for image synthesis. Specifically, this approach replaces the image feature extractor of regular FID, InceptionV3 [47], with the encoder component of an unsupervised time series embedding model [48]. After the embedding model is trained for each dataset separately, it infers latent space encodings for a batch of real and synthetic samples, respectively. This is followed by computing the Fréchet distance [49] between these two subsets. Figure 7 depicts the essential idea. Unfortunately, the authors do not provide details on the training procedure for the embedding model. In any case, the lower the score, the closer the two distributions presumably are and, therefore, the better the synthesis, with 0 being the best [40].

Correlation structure. This measure compares the correlations found in multivariate time series of the real dataset to those of the synthetic dataset. Initially, Remlinger et al. [50] described it as the "term-by-term mean squared error (MSE) between empirical correlation from reference samples on one side and from generated samples on the other side". However, this is very vague, as it remains unclear which correlation is used, how it is applied, and how the MSE are aggregated. While Boursin et al. [51] propose covariance matrix for correlation and average as aggregation, it still does not specify the arrangement of input vectors for the covariance from the three dimensions samples, time steps, and channels. Hence, as code is unavailable as well, a definition cannot be given here.

Coverage. Naeem et al. [52] proposed this measure to improve upon the recall measure by a change of perspective from regions around synthetic samples to such around real samples as follows. The measure is calculated via the indicator function $\mathbf{1}\{\cdot\}$ on sample level, which is then aggregated over all real samples $x \in D_r$. The function returns 1 if there exists a synthetic sample $\hat{x}$ within a k-dimensional ball B around

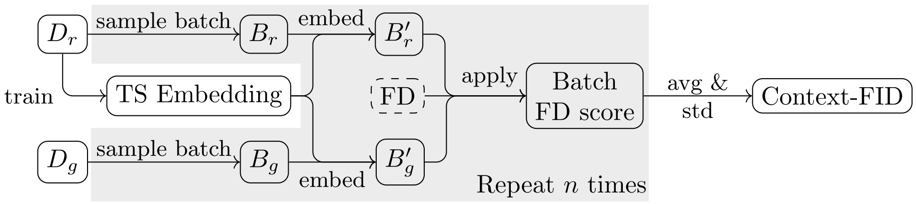


Fig. 7 Flow chart depicting the steps of Context-FID. The process starts on the left with the two input datasets and ends with the computed score on the right. The gray-shaded area represents the steps repeated for each sampled data batch. Each iteration results in the Fréchet distance for the respective batch. These intermediate values are aggregated using mean and standard deviation

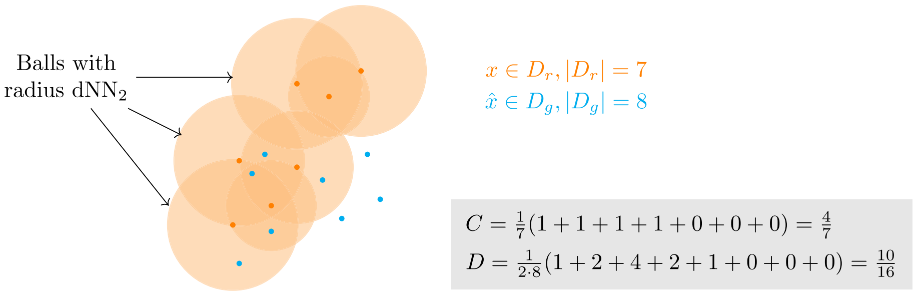

Balls with radius dNN_2
$x \in D_r, |D_r| = 7$
$\hat{x} \in D_g, |D_g| = 8$
$C = \frac{1}{7}(1+1+1+1+0+0+0) = \frac{4}{7}$
$D = \frac{1}{2\cdot 8}(1+2+4+2+1+0+0+0) = \frac{10}{16}$

Fig. 8 Visualization of the computation of coverage and density [52] using *k*-dimensional balls and the distance to the *K* th-nearest neighbor for $K = 2$

the real sample, else 0. Its radius is given by the distance to the K th-nearest neighbor (dNN_K) in D_r. An example is given in Fig. 8. Formally, coverage C is defined by

$$C := \frac{1}{|D_r|} \sum_{x \in D_r} \mathbf{1}\{\exists \hat{x} \in D_g : \hat{x} \in B(x, \mathrm{dNN}_K(x, D_r))\}. \tag{5}$$

Like recall, coverage falls into the range [0, 1] where 1 is best and requires an embedding $f : \mathcal{X} \to \mathbb{R}^k$ into Euclidean space. The measure is not to be confused with the category of the same name.

Data-copying test (C_T) checks a particular type of overfitting behavior of generator g_θ to the real data [53]. Data copying refers to the tendency to reproduce minimal variations of a subset of the data instead of covering the entire true data distribution. In this regard, it differs from over-representation of a certain data region. In preparation, they split the real dataset into a train set, which may be presented to the generator, and a held-out test set for evaluation only, $D_r = (D_r^{train}, D_r^{test})$. Additionally, they sample the generator to create a synthetic dataset D_g. Assuming that overfitting manifests itself in synthetic samples that are generally too close to training data, the measure employs a hypothesis test on the average distance between the train and test datasets, respectively, the train and synthetic datasets. H_0 suggests that these are approximately even. To improve regional awareness, the authors apply the test on each set of a data space partition. Without further details, this finally yields a measure $C_T(D_r^{train}, D_r^{test}, D_g)$. The only requirement is a distance function on the data points. For time series, this may be DTW, for instance. In principle, one can use embeddings for data spaces that do not support an adequate distance calculation. Furthermore, it holds $C_T \in \mathbb{R}$, where 0 is optimal, $C_t \ll 0$ signals data copying, and $C_t \gg 0$ implies model underfitting.

Density is intended to improve the precision measure by putting less weight on outliers in the real data. Naeem et al. [52] calculate the measure via indicator function $\mathbf{1}\{\cdot\}$ on sample level, which is then aggregated over all real samples x and finally synthetic samples $\hat{x}$. The function returns 1 if the synthetic sample is within a k-dimensional ball B around the real sample with dNN_K as radius, else 0. An example is given in Fig. 8. In summary, one obtains density D by

$$D := \frac{1}{K\left|D_g\right|} \sum_{\hat{x} \in D_g} \sum_{x \in D_r} \mathbf{1}\{\hat{x} \in B(x, \mathrm{dNN}_K(x, D_r))\}. \tag{6}$$

Unlike precision, density may take values beyond 1, $D \in \mathbb{R}_{\geq 0}$. However, it still holds that higher is better and an embedding $f : \mathcal{X} \to \mathbb{R}^k$ is required.

Dependence scores subsume two scores designed to compare the dependence properties of the real and synthetic time series [54, 55]. Note, however, that both of them are only applicable to an individual, univariate real time series $X \in \mathbb{R}^l$ and a set D_g of its syntheses. First, denote the autocorrelation $\mathrm{Corr} : \mathbb{R}^l \times \mathbb{R}^l \to [-1, 1]$ of the time lag $\tau \leq l$ between the current time step t and a previous step $t - \tau$ of time series X for all possible t. Formally, they define the autocorrelation function (ACF) as

$$\mathrm{ACF} := \begin{cases} \mathbb{R}^l \to [-1, 1]^\tau \\ X \mapsto (\mathrm{Corr}(X_{t-1}, X_t), \ldots, \mathrm{Corr}(X_{t-\tau}, X_t)). \end{cases} \tag{7}$$

Both scores build on these intra-time series correlations. The first variant, s_{ACF}, uses the ACF as defined above, the second one, s_{LE}, modifies the function slightly by computing the correlations between the squared lagged times series and the time series itself, that is, $\mathrm{Corr}(X_{t-1}^2, X_t), \ldots, \mathrm{Corr}(X_{t-\tau}^2, X_t)$. Hence, one obtains for the ACF variant the definition

$$s_{\mathrm{ACF}} := \left\| \mathrm{ACF}(X) - \frac{1}{|D_g|} \sum_{\hat{X} \in D_g} \mathrm{ACF}(\hat{X}) \right\|_2 . \tag{8}$$

The formulation of s_{LE} is analogous using the modified autocorrelation function ($s_{\mathrm{ACF}}, s_{\mathrm{LE}} \in \mathbb{R}_{\geq 0}$, where smaller is better).

Discriminative score is based on the performance of a binary classifier on a combined synthetic-real dataset [26]. First, each sample from real and generated data is labeled either "real" or "synthetic", depending on where it was taken from. A dataset with two classes is created from these labeled samples and split into train and test sets again. Then, the authors train a simple two-layer long short-term memory (LSTM) network in standard supervised fashion to classify the merged dataset. They report its accuracy on the held-out test set minus 0.5 as discriminative score. Hence, it ranges from an optimal 0.0 to a worst 0.5. We depict this procedure in Fig. 9. An embedding is implicitly given by adjusting the classifier's input layers for differently shaped input. An application to labeled data is impractical.

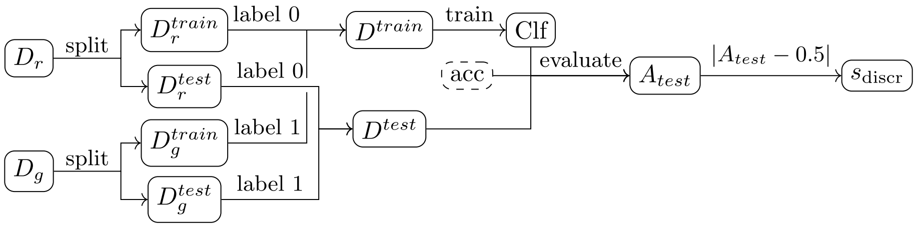


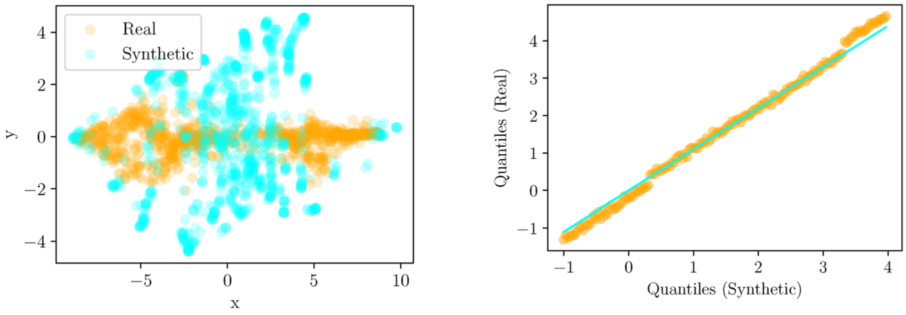
Fig. 9 Depiction of measure discriminative score. The control flow starts with the input datasets on the left, and ending with the final score on the right. The procedure is very similar to C2ST

Distribution of reconstruction errors (DRE). Based on the nearest neighbor approach, this measure is intended to detect a generator just memorizing and reproducing noisy versions of its training data. While others do this by calculating the dNN_K (for $K = 1$) for all synthetic samples, Esteban et al. [56] suggest a supposedly less computationally expensive approach. The idea is to put the actual D_g aside and explicitly generate the nearest synthetic neighbor for all $x \in D_r$, and then test if the neighbors of the train set samples of D_r are systematically closer than those of held-out test samples D_r^{test}. This is implemented by minimizing the reconstruction error $\mathcal{L}_{r(x)}$ between x and a generated neighbor, given by

$$\mathcal{L}_{r(x)}(z) := 1 - K(g_\theta(z), x), \tag{9}$$

where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the kernel function used in MMD. Moreover, g_θ is the (potentially) parameterized generator function and we have $z \in Z$ as its input. However, finding z such that $g_\theta(z)$ is the approximate nearest neighbor still requires performing this optimization to approximate convergence for each x. Only then, given these $\hat{x} = g_\theta(z)$, the authors can compare the results for D_r^{train}, which g_θ was trained on, with D_r^{test}, which serves as a baseline. Namely, they test if the distribution of the reconstruction errors over D_r^{train} is significantly different from that over D_r^{test}. If that is the case and the average reconstruction error is lower for the train than the test set, one can be confident that the generator memorized the data. In order to test the hypothesis of divergence, they employ the Kolmogorov-Smirnov two-sample test with a predefined significance level. A produced p-value below this level supports the hypothesis, while a good, generalizing g_θ should fail the test.

Distribution visualization. With this term, we refer to a group of seven evaluation measures that employ some data transformation, feature extraction, or representation learning technique to map the data into a low-dimensional space. The image of both datasets in this space can then be visualized and inspected by a human judge. Hence, this is not about the visual assessment of individual time series but the dataset collectively. Usually, the synthesis is considered successful if the arrangement of real sample images matches that of synthetic samples. If they diverge strongly in shape or sample density, the generator performed poorly. The most prominent examples of such mappings are principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) [26]. For convenience, we depict such a visualization on the right in Fig. 10. Without going into further detail, we just mention other measures here: QQ-plots and scatter plots are used in [57] and [45], respectively, to visually compare the similarity of



(a) t-SNE visualization of real samples in orange and synthetic ones in cyan. The two distributions do not match.

(b) QQ-plot of the distances between real and synthetic samples in orange. The ideal placement would be on the cyan line.

Fig. 10 Two examples of distribution visualization measures: **a** t-SNE and **b** QQ-plot

sets of geographic trajectories. Pastor-Serrano et al. [58] visualize the latent space distribution of their autoencoder model. Wang et al. [39] propose another approach called dimension-wise probability, which is similar to [59]. Both Pan et al. [60] and Xu et al. [61] evaluate their methods by computing and visualizing probability density functions (PDFs) of marginal distributions.

Distributional metric. The measure proposed in [55] compares real data distribution P and synthetic Q via their respective empirical probability density functions (PDFs) f_P and f_Q. These are determined using a binning approach. Let d be the dimensionality of the time series and $0 \leq c < d$ be a channel. Further, let $\mathcal{B}_c = \{B_c^1, \dots, B_c^n\}$ be a binning of the real time series dataset D_r for channel c, such that $\forall B \in \mathcal{B}_c, X \in D : \left|\{\mathbf{x}_t \in X \mid x_t^c \in B\}\right| \approx 20$ for suitable n. Naturally, this requires the time series to be sufficiently long to define a reasonably accurate density function over the binning. The construction of $\mathcal{B}$ is not further specified. Instead, they define the empirical PDF $f_c : \mathcal{B}_c \to R_{\geq 0}, B_c \mapsto |B_c|$ for each c separately. The authors follow this procedure for both real dataset D_r and its synthesis D_g, which gives them functions f_c^r for P and f_c^g for Q, respectively. However, they reuse the bins defined for D_r on D_g. This gives them a measure M_{epdf} for the absolute difference of all these pairwise empirical distributions in the form of

$$M_{\text{epdf}} := \frac{1}{d} \sum_{c} \sum_{B \in \mathcal{B}_c} \left| f_c^r(B) - f_c^g(B) \right|. \tag{10}$$

It holds that $M_{\text{epdf}} \in \mathbb{R}_{\geq 0}$, where lower is better.

Distributional scores. The term refers to two closely related measures to capture the propensity of the generator to synthesize extremal values [55]. However, it is defined in a way that allows direct application to real time series D_r with $|D_r| = 1$, that is, the comparison of (an arbitrary number of) synthetic time series to a single original, for instance when the latter is extremely long, while each synthesis is relatively short. Let $f \in \{\text{skew}, \text{kurtosis}\}$ denote the higher moment used in the measure. The remainder of their calculation is identical. For real time series $X \in D_r$ and synthetic dataset D_g, define the respective measure by

$$\mathrm{DS}_f := \frac{1}{d}\sum_{c=0}^{d-1}\sqrt{\sum_{\hat{X}\in D_g}\left(f(X_c) - f\left(\hat{X}_c\right)\right)^2}. \tag{11}$$

$X_c, \hat{X}_c$ denotes the cth channel of the respective time series. $\mathrm{DS}_f \in \mathbb{R}_{\geq 0}$, lower is better.

Duality gap. Based on concepts from game theory, duality gap is a distribution similarity measure for guiding and evaluating GAN models. Although data type-agnostic, it relies on the presence of a generator and discriminator positioned as opponents in a zero-sum game. In this context, duality gap measures the sub-optimality of both entities' performance compared to an equilibrium, a game state in which no entity can increase its reward unless the opponent behavior changes as well. The game objective, which measures their performance, was left generic by the authors. For instance, it might be the accuracy of a separate binary classifier trained on real and generated data. However, this also means that researchers need to agree on a universal classifier architecture to make results comparable between publications. Figure 11 contains the specifics. The measure is always a non-negative real number, 0 is optimal. Note that duality gap is tailored towards GANs and primarily indicates a model's convergence or divergence, not the quality of generated samples [62]. Recently, Sidheekh et al. [63, 64] proposed two variants called perturbed and proximal duality gap, which are more accurate than the plain version, especially in cases where the two-player game need not converge to a Nash equilibrium for the generator to model P.

Feature-based correlation analysis. In order to analyze the correlation structure between the channels of individual time series, Seyfi et al. [42] utilize a feature extraction tool called pycatch22 [65]. Specifically, it computes 22 numerical features for each channel of all multivariate time series in both D_r and D_g, which are expected to be of equal length and dimensionality $d > 1$. Afterwards, for each pair of channels (X^i, X^j) with $1 \leq i < j \leq d$, a 22 × 22-matrix of Pearson correlations between the extracted features is computed for real and synthetic dataset each. This is across the sample dimension. Now, for fixed (i, j), the matrices for real and synthetic data are compared with mean absolute error (MAE), MSE, Kendall's τ, Spearman's rank correlation, as well as Frobenius norm. The final score is the quintuple of these statistics averaged across all channel pairs (i, j), where lower is better.

FID was proposed in [66] as evaluation measure for synthetic images. It utilizes an inception model for extracting features from the samples. The model learns a Gaussian distribution with mean m and covariance matrix C for the real data (m_P, C_P) and synthetic (m_Q, C_Q). To these "inceptions", the Fréchet distance d_f is applied:

$$d_f^2((m_Q, C_Q), (m_P, C_P)) = \left|\left|m_Q - m_P\right|\right|_2^2 + \mathrm{Tr}(C_Q + C_P - 2\sqrt{C_Q C_P}). \tag{12}$$

The resulting score measures the similarity between the two Gaussians and therefore, by approximation, between the real and synthetic data distribution. Generally, it is FID $\in \mathbb{R}_0^+$, where lower is better [67].

Hedging effectiveness. In the financial world, hedging refers to the effort to reduce potential losses at the expense of gains achieved in transactions or speculation. In order to evaluate different time series generators with respect to their ability to produce useful synthetic option prices, Boursin et al. [51] propose hedging effectiveness.

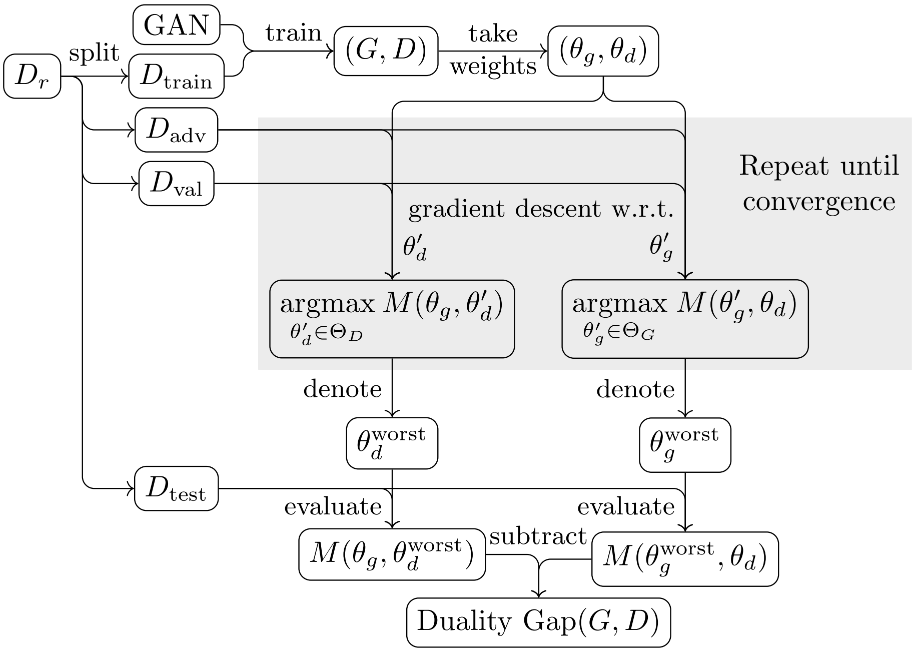

GAN
train
(G, D)
take
weights
(θ_g, θ_d)
split
D_r
D_{train}
D_{adv}
D_{val}
Repeat until
convergence
gradient descent w.r.t.
θ'_d
θ'_g
$\underset{\theta'_d \in \Theta_D}{\text{argmax}}\, M(\theta_g, \theta'_d)$
$\underset{\theta'_g \in \Theta_G}{\text{argmax}}\, M(\theta'_g, \theta_d)$
denote
denote
θ_d^{worst}
θ_g^{worst}
D_{test}
evaluate
evaluate
$M(\theta_g, \theta_d^{\text{worst}})$
subtract
$M(\theta_g^{\text{worst}}, \theta_d)$
Duality Gap(G, D)

Fig. 11 Flow chart for measure duality gap. The inputs are the real dataset D_r and a GAN to be optimized. To this end, we take the generator and discriminator parameters θ_g and θ_d from the trained model, respectively. Then, we seek the worst-case (i.e., best performing) D for fixed θ_g in the left branch and worst-case G for fixed θ_d in the right branch. This is done iteratively via gradient descent and the objective M. The latter is also applied to the resulting G_{worst} and D_{worst} to obtain the final score

The measure is based on the accuracy of a neural network that was taught hedging strategies using the synthesized option prices, compared to the potential losses without hedging and the help of synthetic data. The score falls in the range [0, 1], where greater is better.

Improved precision (p_i) builds on the notion of precision measures for two distributions introduced in [68]. This alternative definition, however, is more straightforward and based on the binary question: "Does a given synthetic sample lie in the support of P?". This idea is illustrated on the left in Fig. 12. In this regard, the support is estimated using the union of regions around all samples of the respective dataset. Practically, the authors utilize k-dimensional balls B around real samples. Furthermore, the question gives rise to the indicator function $\mathbf{1}(\cdot)$ seen in the definition of coverage and density before. The same is true for the distance to the Kth nearest neighbor, which serves as B's radius. Although Sajjadi et al. [68] have chosen another syntax, we stick to the ball notation for consistency. For real dataset D_r and synthetic equivalent D_g, we define (p_i) as

$$p_i(D_r, D_g) := \frac{1}{|D_g|} \sum_{\hat{X} \in D_g} \mathbf{1}\{\exists x \in D_r : \hat{x} \in B(x, \text{dNN}_K(x, D_r))\}. \tag{13}$$

The measure is calculated in the k-dimensional metric space, requiring a prior embedding $f : \mathcal{X} \to \mathbb{R}^k$. A neighborhood size of $K = 3$ and dataset sizes $|D_r|, |D_g| \geq 50\,000$ are recommended for reliable scores, as indicated by tests on image data performed in [69].

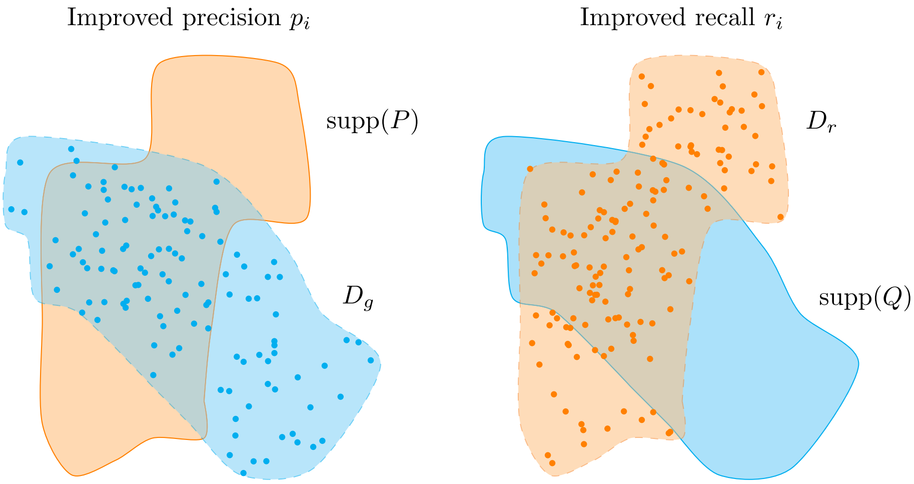


Fig. 12 Visualization of measures improved precision and recall. Visualization of the concepts of improved precision and recall as proposed by Kynkäänniemi et al. [69]. Improved precision measures the fraction of synthetic samples (D_g) located within the support of P. Vice versa, improved recall approximates the fraction of real samples (D_r) located within the support of Q

Improved recall (r_i). As the name suggests, this is the compliment measure to improved precision. In order to approximate the fraction of the real data distribution captured by the generator, Kynkäänniemi et al. [69] reduce the problem to an aggregate of binary decisions on the sample level, as seen before. Hence, the question proposed in this case is "Does a given real sample lie in the support of Q?". This idea is illustrated in Fig. 12 on the right. We get a definition of improved recall exactly mirroring Eq. 13:

$$r_i(D_r, D_g) := \frac{1}{|D_r|} \sum_{x \in D_r} \mathbf{1}\{\exists \hat{x} \in D_g : x \in B(\hat{x}, \mathrm{dNN}_K(\hat{x}, D_g))\} = p_i(D_g, D_r). \tag{14}$$

Analogously, the measure is calculated in the k-dimensional, real-valued feature space, requiring a prior embedding $f : \mathcal{X} \to \mathbb{R}^k$ [69]. The recommendations $K = 3$, $|D_r|, |D_g| \geq 50\,000$ hold.

Intra-class distance (ICD). Given an additional sample-level distance d, this measure represents the average similarity of the samples within a dataset, in this case applied to a subset G of the generated time series $D_g, |G| = 10$.

$$\mathrm{ICD}(D) := \frac{\sum_{x \in D} \sum_{x' \in D} d(x, x')}{|D|^2} \tag{15}$$

The higher the distance the better, that is, more diverse the synthesis. Leznik et al. [9] use the ED for d.

JS divergence on marginals. Generally speaking, JS divergence measures the dissimilarity between two (empirical) distributions. Naturally, one can employ it on data distribution P and synthetic distribution Q. However, for joint distributions P, Q over a high dimensional space, this is computationally infeasible. Therefore, most approaches use an embedding step beforehand to map the data points into a lower dimensional space. Another way is to work with the marginal distributions of P, Q instead, at least in cases where they can be properly identified. An example where this works well is given in a paper on synthesizing human mobility trajectories by Ouyang et al. [57]. Therein, marginals are inferred from a semantic point of view. They define four distributions less complex than P, among them the visiting probability over all locations in the geographic area of interest. Each of the four distributions can be estimated using D_r for the real data and D_g for the synthetic. Moreover, the authors split D_r into two and compute the JS divergence on a held-out test set, which the generator has not seen during training, and the synthetic dataset. As a result, we get a divergence value for each marginal defined and thereby compare different generators on a statistical, yet informative level.

Length histogram. With this term, we title a measure for time series datasets and generators respectively containing and producing sequences of variable length. The intention is that the distribution of the lengths in both real and synthetic datasets should match. As the underlying space of realizations is finite and rather dense in practice, the approach chosen here is simply to compute the histogram of time series lengths in both datasets. By superimposing them in one figure, one can visually compare the histograms and find differences [34].

Manifold topology divergence (MTop-Div). Similar to JS divergence or WD, this measure represents the discrepancy between two distributions, P and Q. Here, these are data and model distribution, respectively. MTop-Div is different in that it is topology-based and views the real data and synthesis as manifolds, on which their respective datasets are point clouds. These manifolds are estimated using a simplicial complex, a concept of topology. To estimate the similarity between the two manifolds, Barannikov et al. [70] propose a mathematical tool called Cross-Barcode, which quantifies the evolution of topological features over multiple scales. The details require quite extensive mathematical explanations. Hence, we omit them here and refer to [70] instead. For the same reason, we provide a high-level depiction of the measure in Fig. 13. We know that $MTop - Div \in \mathbb{R}_{\geq 0}$, where smaller is better. Since the measure operates on the k-dimensional real space, a data type-dependent embedding $f : \mathcal{X} \to \mathbb{R}^k$ is required.

Marginal metrics. This is a combination of three classical statistics used to roughly compare the marginal distribution of each time step in the real dataset to its counterpart in the synthetic dataset [50]. Namely, these are the average, 95th percentile, and 5th percentile, which we refer to as s_1, s_2, s_3, respectively, below. With that, define marginal metrics as the triple

$$\mathrm{MM}(D_r, D_g) := \left(\frac{1}{T} \sum_{t=0}^{T-1} (s_i(V_r) - s_i(V_g))^2 \right)_{i=1,2,3}, \tag{16}$$

where

$$V_\delta := \left\{ X_t^c \mid 0 \leq c < d \wedge X \in D_\delta \right\}. \tag{17}$$

Above, X_t^c denotes the value in channel c at step t from dataset D_δ.

Maximum mean discrepancy (MMD). For a class $\mathcal{F}$ of functions $f : \mathcal{X} \to \mathbb{R}$, MMD is defined as

$$\mathrm{MMD}(\mathcal{F}, P, Q) = \sup_{f \in \mathcal{F}} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{y \sim Q}[f(y)]). \tag{18}$$

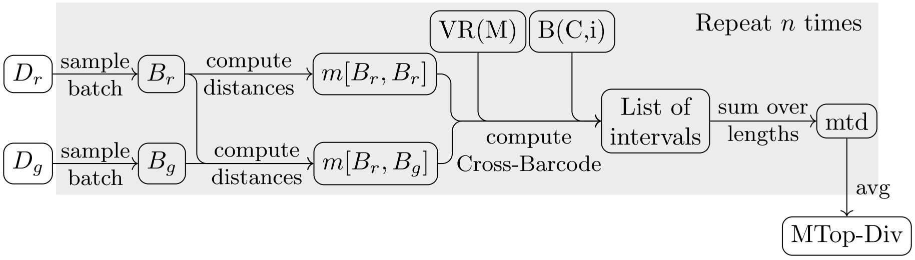


Fig. 13 Schematic depiction of measure MTop-Div. First, batches are sampled from each dataset and distance matrices *m* calculated. Afterward, one utilizes both matrices to calculate the Cross-Barcode between the real and synthetic batch using auxiliary functions computing the Vietoris-Rips complex (VR) of some matrix *M* and the persistence intervals of a complex *C* in dimension *i*. The result is a set of intervals marking the beginning and ending of topological features in the simplicial complex. The sum of all interval lengths is taken as an indicator of similarity. After repeating this process *n* times, the final score is given by the average over all "mtds"

This statistic computes the difference between the mean function values of the real distribution on the one hand, and the model distribution on the other and returns it as a measure for dissimilarity [71]. Later, Esteban et al. [56] proposed a new approximation for P and Q denoted $\text{MMD}_{\approx}$ using a kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ applied to D_r, D_g. This essentially boils down to learning an embedding and a dissimilarity function for time series in one. A lower score is better.

Maximum real to synthetic similarity (Max-RTS). This is meant to be an extreme-case similarity measure, determining the cosine similarity between the real and synthetic sample closest to each other [46]. A visual aid is provided in Fig. 14. The intuition is that individual synthetic samples must not get too close to real ones, in order to prevent data leakage into the synthesis and ensure generalization. Let D_r, D_g be the real and synthetic datasets, respectively. Then, Max-RTS is given by

$$\text{Max-RTS} := \max_{x \in D_r, \hat{x} \in D_g} \left\{ \frac{x \cdot \hat{x}}{||x||_2 \cdot ||\hat{x}||_2} \right\}. \tag{19}$$

Since the cosine similarity is defined on vectors, an embedding $f : \mathcal{X} \to \mathbb{R}^k$ is required. We have Max-RTS $\in [-1, 1]$ and theoretically, the lower the better.

Membership inference attack (MIA). In general, this term refers to the process of determining whether a specific sample from the data distribution is present in the training set used for learning a given ML black-box model [72]. One way to approach this is by training an adversarial model to differentiate the behavior of the target model on inputs from the train set versus other samples. In other words, this can be used to measure the information leakage on sample membership by looking at the target model's outputs. In this case, the goal is to infer (parts of) D_R given the synthesized data D_g, as depicted in Fig. 15. [73] uses this attack as a means to detect presence disclosure. The work mentions a range of real samples, a threshold for the mean ED between "all" samples, and the number of generated samples being utilized. However, the work fails to provide more details on how the attack is conducted. If we were to guess, the author proposes to calculate the mean ED on D_r and filter those real samples that are closer to any

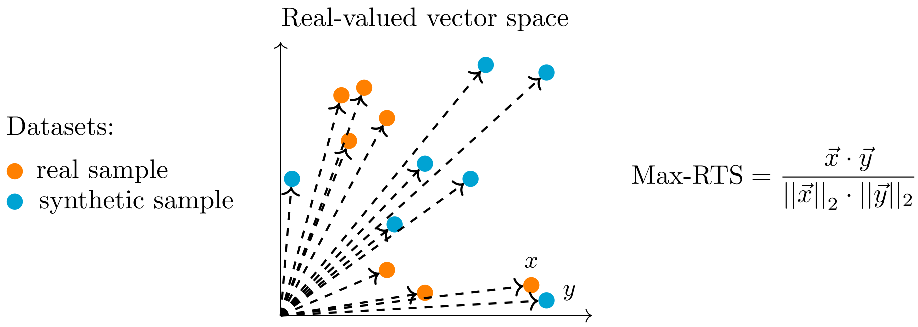

Real-valued vector space
Datasets:
● real sample
● synthetic sample
x
y
Max-RTS $= \frac{\vec{x} \cdot \vec{y}}{||\vec{x}||_2 \cdot ||\vec{y}||_2}$

Fig. 14 Conceptual visualization of measure Max-RTS. This figure contains a small example on how to compare real dataset D_r (orange) and synthetic dataset D_g (cyan) using Max-RTS. Points denote data objects located in $\mathbb{R}^d$, the dashed arrows their respective position vectors. Among all pairs of orange and cyan points, x and y are the closest in cosine similarity

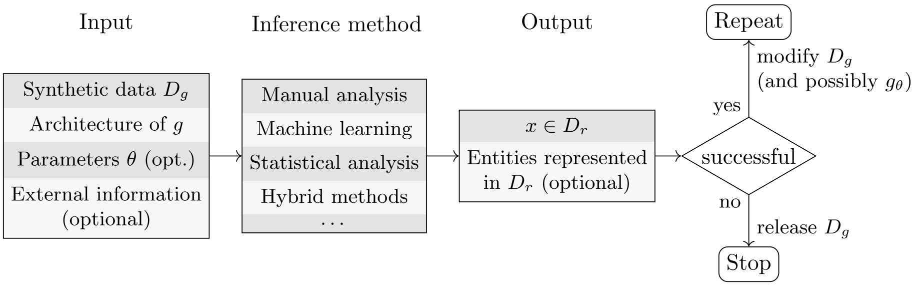


Fig. 15 Procedure and artifacts for a MIA. This diagram sketches the procedure and artifacts for a MIA. In three boxes, we list the input to the attack, the method used to conduct it, and its output, respectively. Depending on whether the attack was successful, one needs to adjust synthetic data and generator or can release them in good conscience

$\hat{x} \in D_g$ than the threshold allows. One may calculate a score in [0, 1] from this fraction of real samples. However, one needs access to D_r for this, which does not conform with the strict black-box assumption. Kulkarni et al. [45] are more specific and reference the location-privacy and mobility meter in [74]. This approach implements both a location-sequence attack and a membership inference attack. The former determines the accuracy with which an attacker can reconstruct trajectories, that is, time series capturing locations of an entity over time, in the real dataset. The latter tries to infer the identity of the entity contributing to a trajectory. However, these are domain-specific tests.

Memorization-informed Fréchet inception distance (MiFID). As the name suggests, this measure is an extension to the "classical" FID, addressing the issue of memorizing and regurgitating parts of the real dataset instead of producing novel samples. Hence, for real dataset D_r and synthesis D_g, MiFID is given by the following combination of a memorization penalty m_τ and the FID itself:

$$\mathrm{MiFID}(D_g, D_r) := m_\tau(D_g, D_r) \cdot \mathrm{FID}(D_g, D_r). \tag{20}$$

To construct m_τ, Bai et al. [75] proposed a memorization distance s between the real and synthetic dataset. First, they fall back to the similarity between individual samples as per the cosine similarity, followed by a minimum over the real data and averaging across all synthetic samples. Formally, this is described by

$$s(D_g, D_r) := \frac{1}{|D_g|} \sum_{\hat{x} \in D_g} \min_{x \in D_r} \left\{ 1 - \frac{\hat{x} \cdot x}{||\hat{x}||_2 \cdot ||x||_2} \right\}. \tag{21}$$

Using this function, define m_τ for an $\epsilon > 0$ as

$$m_\tau(D_g, D_r) := \begin{cases} \frac{1}{s(D_g, D_r) + \epsilon} & \text{if } s(D_g, D_r) < \tau, \\ 1 & \text{otherwise.} \end{cases} \tag{22}$$

τ serves as a threshold above which similarity is considered overfitting and thus penalized inversely with respect to the memorization distance. Below this threshold, the FID is not modified. Choosing suitable values for parameters τ and ϵ is up to the user. For MiFID, a lower value is better. Analogous to FID, this extension operates on the real-valued vector space and therefore needs an embedding $f : \mathcal{X} \rightarrow \mathbb{R}^k$ for all other data spaces.

Minimax loss is a derivative of the duality gap measure but with the motivation to assess synthesis quality and generalize to all kinds of generation methods. Based on the same idea of an equilibrium of two entities in a min-max game, the implementation rather comes across as merely evaluating the generated dataset through another, discriminative model very much like discriminative score does. More precisely, Grnarova et al. [62] suggest a split of real and synthetic data into three subsets each, one for training G, one for finding the "worst case discriminator" D_{worst}, and one for determining the utility M for G. In this regard, D_{worst} is the best classifier network obtainable with a predefined architecture, that is, an optimized model. The abstract function M is given by the classification loss of D_{worst} on the third pair of real and synthetic subsets. The higher the loss, or inversely, the lower the classification accuracy, the better G performs. We assume that the classification model implicitly computes an embedding. The application to labeled datasets is impractical. CAS is more suited to this task, according to [62].

Multi-sequence aggregate similarity (MSAS) is a similarity measure for normalized time series built around an additional statistic $f : \mathbb{R}^l \rightarrow \mathbb{R}$, which maps a column onto a scalar value [76]. Examples include length, mean, and standard deviation. Furthermore, it is restricted in its application by the assumption of a generator of the form $g_\theta : \mathbb{R}^{l \times d} \rightarrow \mathbb{R}^{m \times d}$, where the input is a seed sequence of length l and d channels and the output one with length m and the same dimensionality. Hence, the algorithm can iterate the pairs $(X, \hat{X})$ of real time series and one of its syntheses, each channel separately. In each iteration, one computes $f(X)$,$f(\hat{X})$. In order to compare feature distributions, one applies the 2-sample Kolmogorov-Smirnov test on the set of features computed on the real samples and the one computed on the synthetic samples. This statistical test returns the probability that the two sets are taken from the same distribution. The scores for the columns are averaged to finally arrive at MSAS ($MSAS \in [0, 1]$, where higher is better).

Neural network divergence (NND) is a measure originating from the evaluation of image synthesis, but is at its core data type-agnostic [77]. Like WD or FID, it tries to estimate the discrepancy $D(P, Q)$ of real and synthetic distributions using finite sets of samples D_r and D_g, respectively. The idea is to use the loss of a neural network trained on discriminating D_r and D_g as empirical proxies for P and Q, respectively. The model architecture must be standardized across the generators for evaluation. In order to apply this measure, both real and synthetic data need to be split into train and test sets, $D_r = (D_r^{train}, D_r^{test}), D_g = (D_g^{train}, D_g^{test})$. The generator may only use D_r^{train}. Now, the neural network is trained on a supervised classification task, differentiating the two train sets. The classification loss during inference on the test sets is then used as an empirical discrepancy measure estimating $D(P, Q)$. To foster generalization of the synthesis even for a rather small D_r^{test}, Gulrajani et al. [77] propose to use proportionally bigger generated datasets, $D_g^{train} \gg D_r^{train}$, $D_g^{test} \gg D_r^{test}$ in order to detect an overfitting generator. This generalization aspect sets NND apart from prior statistical discrepancy measures like FID ($NND \in \mathbb{R}_{\geq 0}$, and the smaller the better).

Number of statistically different bins (NDB) measures the degree to which a generator over-emphasizes particular modes of the data distribution within its synthesis, neglecting other, less prevalent regions [78]. This is commonly referred to as "mode collapse" and a major concern among GANs. The idea is to employ a two-sample hypothesis test assuming that in every region of the data space, real distributions P and Q should be equal with respect to a significance level of $s = 0.05$. In preparation for this, the k-means clustering algorithm is applied to the real D_r in order to create a partition Π, creating k subsets referred to as "bins". The synthetic D_g is split into k bins in a similar fashion using the cluster centers of Π. Alternatively, one can think of this binning as a discrete probability distribution. Given a sufficient number of samples falling into each bin, the distribution of real samples in bin π, P_π, can be compared to that of synthetic samples, Q_π, using said hypothesis test. We get a score Z_π and can compute the number of bins, for which the score indicates a significant deviation of Q from P as

$$\mathrm{NDB} = \sum_{\pi \in \Pi} \mathbf{1}\{Z_\pi > s\}, \tag{23}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. For variable k, it is $NDB/k \in [0, 1]$, where lower is better. Figure 16 depicts a high-level summary. Besides, one can also compare the relative sizes of corresponding bins P_π and Q_π. The binning requires a distance function which may directly be applied in the data space (e.g., DTW for time series), or the feature space after an embedding.

NDB over-representation and under-representation (NDB-over/under). This measure is an adaption of the NDB measure in [78], intended as a complement to data-copying test [53]. The latter is unable to measure over- and under-representation of individual regions in the data space caused by an ill-fit generator, which can be countered with the help of NDB. To this end, the data space – or feature space after an embedding – is partitioned into regions. This time, the null hypothesis is the assumption that data distribution P and model distribution Q are equal in every region π, $H_0 : P(\pi) - Q(\pi) = 0$. By applying a statistical test on H_0, a region-specific score Z_π is calculated. By comparing each Z_π with a significance level s, one can determine the number NDB-under (resp. NDB-over) of regions under-represented (resp. over-represented) by the generator as follows

$$\text{NDB-under} = \sum_{\pi} \mathbf{1}\{Z_\pi < -s\}, \quad \text{NDB-over} = \sum_{\pi} \mathbf{1}\{Z_\pi > s\}. \tag{24}$$

For short, we call this pair $\mathrm{NDB} - \mathrm{over/under} \in \mathbb{N}^2$ with (0, 0) being optimal.

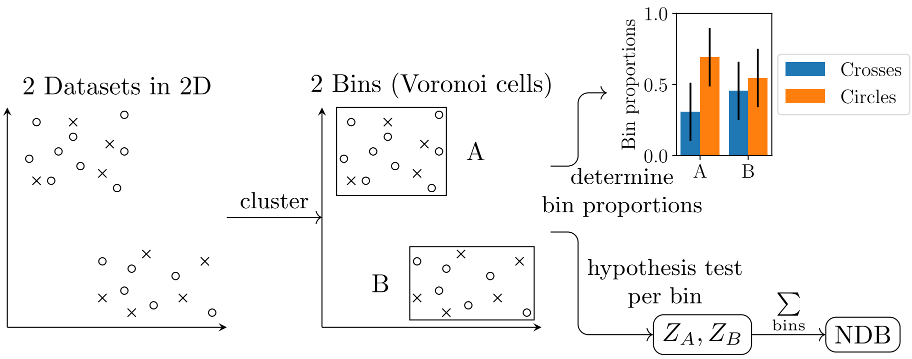


Fig. 16 Visualization of measure NDB. We use two small 2-dimensional datasets (crosses and circles) as an example. The first step is to cluster the data, in this case into two bins A and B. Afterwards, one can apply a hypothesis test on each bin to determine if the number of circles is statistically different from that of crosses and count such bins to arrive at the final score. Alternatively, we can compare the proportions of individual bins of both datasets, for instance, graphically as depicted here

Outgoing nearest neighbor distance (ONND). Let $G \subseteq D_g$ be a random subset of the synthetic dataset and $d \in \{\text{ED}, \text{DTW}\}$ be a distance measure. For each real time series X, the ONND is given by

$$\text{ONND(X)} = \min_{\hat{X} \in G} d(X, \hat{X}). \tag{25}$$

Both Arnout et al. [79] and Leznik et al. [9] use PCA as an embedding before employing the function, but this is optional. The smaller the distance, the better. To get a scalar score for D_r, one can take the mean over all X, whereas in reviewed works, it is used during training.

Pairwise Pearson correlation (PPC). This measure aims to facilitate the visual comparison of real and synthetic time series datasets via heat maps of their internal pairwise correlation [5, 39]. Let D be a time series dataset with time series $X \in \mathbb{R}^{l \times d}$ of length l and dimension d. Further, let $X' \in \mathbb{R}^{l \cdot d}$ denote the concatenation of all channels of X. Now, they construct $l \cdot d$ ordered sets of points across D, such that set S_i contains the value of X' at position i for the entire dataset, that is, $S_i := [X'_i \mid X \in D]$. At this point, one can compute the set C of pairwise correlations between these sets with

$$C := \{\text{PCC}(S_i, S_j) \mid 1 \leq i, j \leq l \cdot d\}. \tag{26}$$

In this case, these are not the correlations between time series, but across sets of points, one from each times series in the dataset. They do the above for both real data D_r and synthetic D_g and get C_r and C_g, respectively. Finally, they can arrange each one in an $ld \times ld$ grid, visualize both using heat maps and compare the two. An examples is provided in Fig. 17.

Precision and recall for distributions (PRD). This measure does not compute a single score but rather a set PRD(P, Q) of precision-recall pairs between data distribution P and model distribution Q [68]. PRD is interpreted as a precision-recall curve using the points of the set farthest from the origin. Some examples are provided in Fig. 18. Although its computation is different from the formal definition and only an approximation of this curve is needed in practice, we try to convey the intuition here and thus stick to the definition.

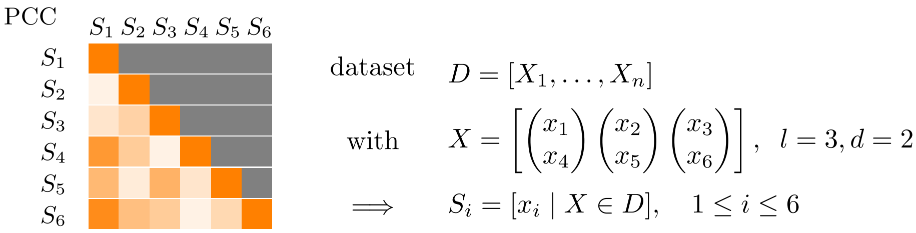

PCC
S_1 S_2 S_3 S_4 S_5 S_6
S_1
S_2
S_3
S_4
S_5
S_6
dataset $D = [X_1, \ldots, X_n]$
with $X = \left[\begin{pmatrix} x_1 \\ x_4 \end{pmatrix} \begin{pmatrix} x_2 \\ x_5 \end{pmatrix} \begin{pmatrix} x_3 \\ x_6 \end{pmatrix}\right], \quad l = 3, d = 2$
$\Longrightarrow \quad S_i = [x_i \mid X \in D], \quad 1 \leq i \leq 6$

Fig. 17 Heat map for measure PCC. Depicted is a heat map showing pairwise Pearson correlations between positions within time series for measure PCC. Each set *S* used in the computation consists of the values of one position. A saturated orange indicates 1, and a white cell 0. Grey cells are redundant. Here, the correlations are random

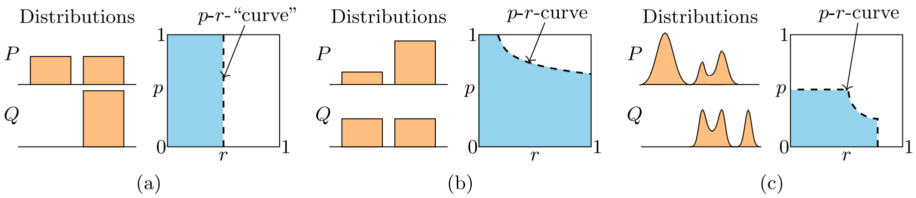


Fig. 18 Three qualitative examples for measure PRD. Each example shows a pair *P*, *Q* of distributions (both orange) with their PRD set (cyan) and curve. **a**, **b** depict simple cases of discrete distributions, where the *p*-*r*-curve in **a** reflects the high precision and low recall of Q and the curve in **b** shows deficits in precision. **c** depicts distributions in a continuous space with three modes each. Significant parts of their probability density do not match, hence the relatively bad *p*-*r*-curve

Intuitively, the precision component measures the probability that a random synthetic sample falls within the support of P and vice versa for recall. We say Q has precision p and recall $r \in (0, 1]$, with respect to P if there exist distributions μ, ν_P, ν_Q such that

$$P = r\mu + (1 - r)\nu_P \quad \text{and} \quad Q = p\mu + (1 - p)\nu_Q. \tag{27}$$

Therein, μ is the shared probability mass, ν_P denotes the remaining mass of P, and symmetrically, ν_Q is the remaining mass of Q, all scaled accordingly. Now, PRD(P, Q) consists exactly of the pairs (p, r) satisfying Eq. 27. Both measure components fall into (0, 1] where higher is better. However, the approximation algorithm proposed by Sajjadi et al. [68] only works on discrete, finite feature spaces Ω. Hence, they use a histogram-based embedding $f : \mathcal{X} \to \Omega$. Later, Simon et al. [80] simplified the definition and conceived a new algorithm exploiting type I and II error rates of a binary classifier trained on the merged real and synthetic datasets. As a result, the distributions can be compared in a real-valued, potentially infinite feature space. One can either use an embedding of the form $f : \mathcal{X} \to \mathbb{R}^k$ or adapt the classifier to accommodate the input space instead.

Predictive score. Specifically proposed for time series data, predictive score measures the usefulness of the synthetic dataset for training one-step predictions. This task was chosen to stress the requirement of generative models to capture temporal correlations. Yoon et al. [26] suggest a simple LSTM-based model to perform the predictions. Training is conducted independently from the generator on the synthetic dataset, while the evaluation takes place on the real dataset by simply calculating the average prediction error across all time series. This averaged error – they use MAE – serves as a relative score for comparing the usefulness of different models. However, it is no absolute measure of quality. The predictive score lies in $\mathbb{R}_{\geq 0}$, where lower is better.

Relative MMD test. Bounliphone et al. [81] proposed this general measure for data synthesis evaluation, meaning it is not specific to any data type. Relative MMD is a test of pairwise relative similarity between two synthetic datasets, D_g and D'_g, on one side and a real dataset, D_r, on the other. A hypothesis test based on MMD, which acts as the distance between embeddings of each distribution in a reproducing kernel Hilbert space, asserts whether D_g is significantly closer to D_r. Hence, the resulting score is binary, stating which of the synthetic datasets is closer to the real one but without any absolute or more fine-grained information. Furthermore, it requires some embedding $f : \mathcal{X} \rightarrow \mathbb{R}^k$, which may simply be the concatenation of channels in the case of time series. Lastly, it assumes some real data to be held back from generator training.

Spatial Correlation. Let $R \subseteq D_r, G \subseteq D_g$ for a real and generated dataset [9]. Using these subsets, spatial correlation estimates the correlation between the channels of multivariate time series in the generated data compared to its original. For all X in both sets, they calculate all pairwise Pearson correlations $\mathrm{PCC}(X^i, X^j)$ between channels i, j of X, $i \neq j$. This gives us $k = d(d-1)/2$ coefficients for dimensionality d, that is, the number of channels, which they average across each dataset. These k averages for real and synthetic datasets are now separately aggregated using the squared difference. Smaller is better. Xu et al. [61] also compare the correlation coefficient between channels. Specifically, they claim to take the "sum of the absolute difference of the correlation coefficients between channels averaged over time". It remains unclear, however, how averaging over time can be conducted when the time dimension is needed for the correlation between channels. Additionally, the absolute difference can be applied between a coefficient from a real sample versus one from a synthetic sample, two samples from within one dataset, or the $d \cdot d$ coefficients of one sample. In any case, their formulation is unclear as the explanation is limited to this one sentence. Besides, a smaller score is still better. Jarret et al. [10] propose this measure later as x-Corr score.

Synthetic to synthetic similarity (STS) measures the intra-class similarity between the samples synthesized by a conditional generator, that is, one that produces for each data point an accompanying label [46]. Let $D_g = \{(x_i, l_i)\}_i$ be a labeled synthetic dataset. The similarity score is independently calculated for each class and finally combined into a vector, one position for each class. Hence, for class c with size $1 < n_c < |D_g|$, they choose a complexity factor $\gamma \in (0, 1]$, and uniformly randomly sample $\gamma \cdot n_c$ data points from c. Denote this set by S_c. Now, they calculate the ACS between each $(x, c) \in S_c$ and five other random points $(x_1, c), \ldots, (x_5, c) \in D_g$ in that class as follows

$$\mathrm{STS}_c = \frac{1}{5|S_c|} \sum_{(x,c) \in S_c} \sum_{i=1}^{5} \frac{x \cdot x_i}{||x||_2 \cdot ||x_i||_2}. \tag{28}$$

Since one needs vectors for the cosine similarity, we can either apply the measure directly in case of univariate time series, for instance, or require an embedding $f : \mathcal{X} \rightarrow \mathbb{R}^k$. The choice of γ depends on the computational resources and time available, and the desired accuracy of the score. During training, a low value seems appropriate, whereas final evaluation should warrant a larger proportion of samples. Since the goal is to ensure intra-class diversity and prevent mode collapse, values close to 1 should be avoided. However, Norgaard et al. [46] do not comment on which values of the codomain [0, 1] are desirable or acceptable. This is problematic since datasets are often scaled to [0, 1], heavily limiting the angle between data vectors.

Temporal correlation. Let $R \subseteq D_r, G \subseteq D_g$ for a real and generated dataset, respectively [9]. Using these subsets, temporal correlation estimates the correlation between time steps in the generated data compared to the original. Per channel, sample, and dataset, the discrete Fourier transform (DFT) between the time steps is calculated and scaled. Then, the authors extract the k largest values (peaks), where k is tuned manually on a per-dataset basis, and determine their distance. The aggregation method is the same as for ApEn. Smaller is better.

topN locations. Intended for use on mobility trajectories, that is, time series describing the location of some entity over time, this measure can be easily generalized to other use cases [45, 57]. The only condition is that we can identify a property or set of properties we are particularly interested in, and on that, each time series can be practically tested. Staying in the context of trajectories, this can be the time spent at or the number of visits to each location from a discrete, finite set of possible locations. We can view this as an embedding $f : \mathbb{R}^{l \times d} \to \Omega$, where Ω is the set of possible values for the property. Now, one can compute $\omega_X = f(X), \omega_{\hat{X}} = f(\hat{X}) \; \forall X \in D_r, \hat{X} \in D_g$, where D_r and D_g are real and synthetic datasets, respectively. This opens up another distribution over the real and synthetic data, which the authors exploit by determining their discrepancy statistically or visually through plots. Assuming each ω stands for a vector holding the number of visits of the time series to location l at position l, they take the sum over all vectors and select the N counts for the most frequently visited locations via

$$\text{topN}_r := \text{topN} \sum_{X \in D_r} f(X) \quad \text{and} \quad \text{topN}_g := \text{topN} \sum_{\hat{X} \in D_g} f(\hat{X}). \tag{29}$$

Finally, one can apply a discrepancy measure or plot the N values for both and judge their similarity.

Train on real, test on synthetic (TRTS) was proposed a few years ago by Esteban et al. [56]. Given a supervised task T applicable to the real data D_r, the parameterized TRTS(T) determines the generator's ability to mimic essential features of D_r, such that a solution for the real data is also a solution for the synthetic one under T. Common choices for T include classification and prediction. In preparation for the measure, they split the real dataset into a train and test pair, $D_r = (D_r^{train}, D_r^{test})$. The generator is trained on D_r^{train} following standard practice, as is the model M used to solve the task. Then, M is evaluated on both D_g and D_r^{test}, which gives them two scores s_g, s_r for M, computed using an evaluation measure appropriate for T and M. Without loss of generality, we assume higher is better for the sake of simplicity. If $s_g \approx s_r$ or – even better – $s_g > s_r$, then one can safely assume that the samples in the generated test set encode features distinctive and similar enough to those in the real test set. If, however, $s_g \ll s_r$, the generated samples deviate too much from those M has seen before in D_r^{train}. A typical constellation is to use classification as T, a deep neural network as M, and a score like area under precision-recall curve (AUPRC) or AUROC. Strictly speaking, the need for an embedding depends on the choice of T, whereas typically, M provides its own integrated embedding. The applicability to classes in the datasets depends on T, for instance, classification requires labeled D_r and D_g.

Train on synthetic, test on real (TSTR). With the above TRTS in mind, we can now consider the reverse case [56]. Similarly, given a supervised task T, the parameterized measure TSTR(T) determines the usefulness of a synthetic dataset D_g for T. Common choices for T include classification and prediction. In preparation for the measure, they split the real dataset into a train and test pair, $D_r = (D_r^{train}, D_r^{test})$. The generator is trained on D_r^{train} following standard practice, whereas the ML algorithm M used to solve the task is independently trained on D_r^{train} and D_g, yielding models M_1 and M_2, respectively. Then, both are evaluated on D_r^{test}, which gives us two performance indicators s_1 for M_1 and s_2 for M_2 to compare. We assume higher is better for the underlying measure. If $s_1 \approx s_2$ or $s_1 < s_2$, then the synthesized data demonstrates high practical value with respect to T and can presumably replace D_r without a compromising model quality. Otherwise, a user knows that D_g lacks essential properties required for M to learn as well as possible with real data. To give one example tailored towards time series, T might be the prediction of the last steps in a time series, M an LSTM-based neural network, and s the average root mean squared error [38]. An embedding is usually not necessary, as most M provide their own integrated embedding. The applicability to classes depends on T, for instance, time series prediction requires no dedicated labels.

WD is defined as

$$W(P||Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(x,y)\sim\gamma}[\ ||x-y||\ ], \tag{30}$$

where $\Pi(P, Q)$ is the set of all joint distributions $\gamma(x, y)$ whose marginals are P and Q, respectively. P is the marginal on x, that is, $\int_{\mathcal{X}} \gamma(x, y)dy = P(x)$, and similarly Q the marginal for y. We implicitly assume a metric space, for instance, $\mathcal{X} = \mathbb{R}^n$. Figuratively speaking, γ is a function that says “how much” of one distribution needs to be moved from point x to point y in the metric space to transform it locally into the other. Then, the WD specifies the “cost” associated with the aggregated shortest distance to travel in order to transform p into q, point by point [82]. For more details and background, see the work of Villani [83]. In the context of time series synthesis, this measure was applied in [54, 84] to determine the discrepancy between real distribution p and synthetic q.

WD on cumulative distribution function (CDF). Lin et al. [34] applied this measure on time series data with values from a range of integers, while it can also be applied to those from the real-valued data domain that allow binning over the time steps. Let $X \in \mathbb{N}^{l\times d}$ and $\hat{X} \in \mathbb{N}^{l\times d}$ be real and synthetic uni- or multivariate time series from datasets D_r and D_g, respectively. For each channel independently, one computes the Wasserstein-1 distance between the overall value distributions in both datasets. More specifically, the authors compute the empirical CDF for function $f : \mathbb{N} \to \mathbb{N}$ counting the number of occurrences of some value $x \in \mathbb{N}$ in all the time series of a set D. Doing this for both D_r and D_g gives us two CDFs, for which one can now determine the WD by taking the integrated absolute error between them. Doing this for each channel, this would amount to a vector $\mathbf{v} \in \mathbb{R}^d_{\geq 0}$ of distances. Of course, a visual comparison of their graphs is also possible.

Wavelet coherence score (WCS). Wavelet coherence is one of many ways to measure the correlation between two univariate time series [85]. It is especially useful to analyze non-stationary time series, that is, such with major changes in statistical characteristics over time like a moving mean caused by a trend. Like many other measures, it requires the transformation into a different feature space. Similar to DFT, this is also a frequency domain, with the continuous wavelet transform as mapping. The definition of wavelet coherence is quite complex and its explanation lengthy. We refer to the work of Grinsted et al. [86] for more details. Here, it is applied to a pair of channels at a time, one from a real, one from a synthetic time series. The immediate output of the wavelet coherence computation is a matrix $wcoh$ with shape frequencies × time steps. Hence, an intermediate aggregation step is required to obtain a scalar value $wcoh_s \in \mathbb{R}$ for each pair of channels. To this end, one simply sums over the time steps and then takes the mean over the frequency axis. For entire datasets D_r (real) and D_g (synthetic), WCS is simply the average of all $wcoh_s$ computed between real-synthetic pairs.

$$\text{WCS} := \frac{1}{|D_r| \cdot |D_g| \cdot d} \sum_{X \in D_r} \sum_{\hat{X} \in D_g} \sum_{c=1}^{d} wcoh_s(X_c, \hat{X}_c), \tag{31}$$

with higher scores are better.

***β*-recall** is a parameterized measure denoted by R_β, where $\beta \in [0, 1]$ is the fraction of synthetic samples assumed to be “typical” for the generator [87]. It represents the fraction of real samples covered by those generator-typical data. More formally, R_β is the probability of a real sample falling within a region $\mathcal{S}$. Therein, $\mathcal{S}$ is the minimum volume subset of the support of the synthetic distribution that supports a probability mass of β. We illustrate this concept on the left of Fig. 19. Single real samples are assigned a value of 1 if they fall within $\mathcal{S}$, and otherwise 0. Implementation-wise, this is done by checking if there is at least one synthetic sample near the real sample. Although computed for individual real samples, R_β counts as distribution-level, as an entire synthetic dataset must be given to check this. Ordinary recall is a special case of this measure, which is instead computed for all β. The individual scores are arranged as a recall curve and can be interpreted directly or aggregated to a single score called integrated R_β via

$$IR_\beta = 1 - 2 \int_0^1 \left| R_\beta - \beta \right| d\beta. \tag{32}$$

Also, see Fig. 19 right for an example curve. It holds that $IR_\beta \in [0, 1]$, where higher is better. β-recall is data type-agnostic, operates on a fixed-size vector space, and therefore requires an embedding. By default, this is an LSTM-based autoencoder (AE) model.

Sample-level measures

Below, we provide an introduction to each sample-level measure found. This list is sorted in alphabetical order to foster faster look-up for non-sequential reading. A summary can be found in Table 5.

Authenticity measures the rate at which a generator produces samples that appear novel [87]. The underlying assumption is that the model creates samples randomly through one of two ways: With probability p_A, a new sample is innovated, and with $1 - p_A$, a memorized

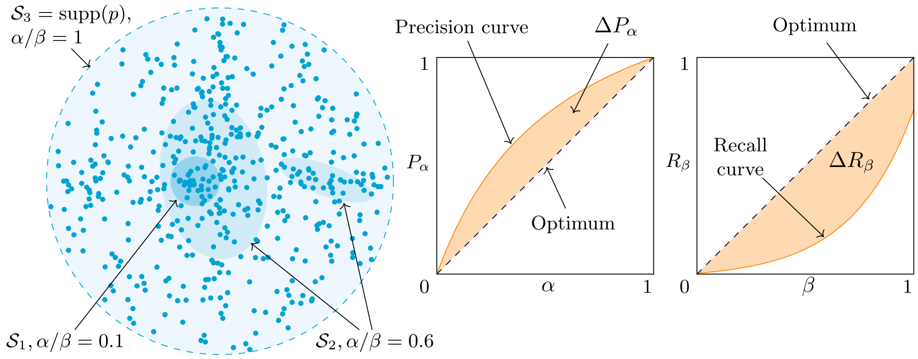


Fig. 19 Illustrations for measures α-precision and β-recall. On the left, the objects of some dataset D following an unknown distribution p are plotted. We depict the supposed support of p, including three potential minimum volume sets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$, formalizing "typical samples". In the middle, we provide the graph of an exemplary result of an α-precision calculation. Analogously, on the right for β-recall

real sample with some added noise is presented. For a good model, p_A is close to one. The measure tries to determine which decision was made through a hypothesis test on the distances between two real samples, and the synthetic and a real sample. We denote authenticity by A, with $A \in [0, 1]$ on a distribution-level and $A \in \{0, 1\}$ for samples, where $A(\hat{x}) = 1$ classifies a sample as authentic. The generalization to a generated dataset D_g is given by

$$A(D_g) = \frac{\sum_{\hat{x} \in D_g} A(\hat{x})}{|D_g|} \tag{33}$$

and thus, the higher the score for the dataset, the more authentic it is. The measure is data type-agnostic, operates on a fixed-size vector space and therefore requires an embedding.

Cointegration tests. Cointegration is a statistical property that might be observed in a multivariate time series or a collection of univariate ones. It is present when all channels involved correlate in the long term, most likely following the same overall trend. These tests are quite popular in the financial world, for instance, for verifying a lasting correlation between the market values of two securities. Popular examples include the Engle-Granger test, which applies only to pairs of channels, and the Johansen test, which can handle arbitrarily many relationships and, therefore, channels. In the context of synthetic time series, Leznik et al. [38] employed it to check if a real time series and its synthesis follow the same trend. This implies that each synthesis has a unique original in the real dataset, which limits the applicability of these tests to generators of the form $g_\theta : \mathcal{X} \rightarrow \mathcal{X}$. Although they do not mention their aggregation method, it can be assumed that a simple arithmetic mean is adequate.

Elementary statistics. We use the term to collectively refer to an open group of rather simple statistical properties that are defined on arbitrary real-valued data samples or time series in particular. Concretely, we identified the use of moments of different ordinal [59], covariance matrix [5, 61], and ACF [34, 38, 60, 93]. Comparison can be done via manual inspection of a plot or taking the difference between the score on the real and synthetic samples [59, 93]. In principle, these statistics can be easily adapted to the dataset level, that is, by averaging [34].

Table 2 Distribution-level evaluation measures found during literature study

Measure	Applicability	Embedding	L	C	Uses
Agreement rate	Unrestricted	Case-dependent	×	×	[33]
Algorithm comparison	Unrestricted	Case-dependent	◊	×	[34]
AIS	Decoder-based models	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[35]
ApEn	(TS)	–	×	✓	[9, 38]
Augmentation test	Known downstream task	Case-dependent	×	×	[39, 40]
ACS	TS	$f : \mathbb{R}^{l\times d} \to \mathbb{R}^{7\cdot d}$	✓	×	[41]
Average JS distance	TS	$f : \mathbb{R}^{l\times d} \to \mathbb{R}^{7\cdot d}$	✓	×	[41]
AED	TS ($d = 2$)	–	×	✓	[42]
AWD	seasonal TS	–	×	✓	[42]
CAS	Unrestricted	–	✓	✓	[43, 58]
C2ST	Unrestricted	–	×	✓	[44]
Computational complexity	Access to generator	Case-dependent	◊	×	[33, 45, 88]
Confusion matrix	Unrestricted	–	✓	×	[46, 85]
Context-FID	TS	$f : \mathcal{X} \to \mathbb{R}^k$	×	×	[40]
Correlation structure	Multivariate TS	–	×	×	[50, 51]
Coverage	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[52, 87]
Data-copying test	Distance required	Case-dependent	×	✓	[53]
Density	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[52, 87]
Dependence scores	Single real TS	–	×	×	[54, 55]
Discriminative score	Unrestricted	Case-dependent	×	✓	[26, 27, 58, 70, 88–90]
DRE	Unrestricted	–	×	✓	[56]
Distribution visualization	Case-dependent	–	◊	✓	[26, 27, 39, 41, 42, 45, 57–61, 70, 85, 89–92]
Distributional metric	TS	–	×	✓	[55, 93]
Distributional scores	Single real TS	–	×	✓	[55]
Duality gap	GANs	–	×	✓	[62, 63]
Feature-based correlation analysis	Multivariate TS	–	×	✓	[42]
FID	images	$f : \mathcal{X} \to \mathbb{R}^{k\times k}$	×	✓	[67]
Measure	Applicability	Embedding	L	C	Uses
Hedging effectiveness	Financial hedging	–	×	×	[51]
Improved precision	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[69, 87]
Improved recall	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[69, 87]
ICD	Unrestricted	–	×	✓	[9]
JS divergence on marginals	Suitable marginals	–	×	×	[57]
Length histogram	Variable-length TS	–	×	×	[34]
Marginal metrics	TS	–	×	×	[50, 51]
MTop-Div	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[70]
MMD	Unrestricted	$f : \mathcal{X} \to \mathbb{R}$	×	✓	[45, 56, 67, 73]
Max-RTS	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	×	[46]
MIA	Unrestricted	Case-dependent	×	✓	[34, 45, 73]
MiFID	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[75]
Minimax loss	Unrestricted	–	×	✓	[62]
MSAS	$g_\theta : \mathbb{R}^{l\times d} \to \mathbb{R}^{m\times d}$	any statistic $f : \mathbb{R}^l \to \mathbb{R}$	×	×	[76]
NND	Unrestricted	–	×	✓	[77]
NDB	Distance required	Case-dependent	×	✓	[78]
NDB-over/under	Distance required	Case-dependent	×	✓	[53]

Table 2 (continued)

Measure	Applicability	Embedding	L	C	Uses
ONND	Unrestricted	PCA (opt.)	×	✓	[9, 79]
PPC	Real-valued data	–	×	×	[5, 39, 90]
PRD	Unrestricted	$f:\mathcal{X}\to\Omega / f:\mathcal{X}\to\mathbb{R}^k$	×	✓	[68, 69, 80, 87, 91, 92]
Predictive score	TS	–	×	✓	[10, 26, 27, 88–92]
Relative MMD test	multiple D_gs	$f:\mathcal{X}\to\mathbb{R}^k$	×	✓	[81]
Spatial correlation	TS	–	×	✓	[9, 10, 38, 61]
STS	Unrestricted	$f:\mathcal{X}\to\mathbb{R}^k$	✓	×	[46]
Temporal correlation	TS	–	×	✓	[9]
topN locations	Discrete, decidable property	$f:\mathbb{R}^{l\times d}\to T$	×	×	[45, 57]
TRTS	Known downstream task	case-dependent	◊	✓	[42, 56]
TSTR	Known downstream task	case-dependent	◊	✓	[5, 10, 34, 38–40, 42, 56, 85, 90, 93, 94]
WD	Unrestricted	$f:\mathcal{X}\to\mathbb{R}^k$	×	×	[54, 84]
WD on CDF	TS	$f:\mathbb{R}^{l\times d}\to\mathbb{N}^{l\times d}$	×	×	[34]
WCS	TS	–	×	×	[85]
β-recall	Unrestricted	LSTM-AE	×	✓	[87]

Next to each measure, we provide potential restrictions in terms of applicability to a specific generation method or type of data, as well as the embedding needed and the works it has been used in so far. Column "L" indicates if the measure is designed for labeled data (✓), unlabeled data (×), or both (◊). "C" says if code is publicly available (✓) or not (×). "Uses" refers to the applications of the measure to general or time series synthesis evaluation found, where the first reference is also the first application (and authorship, if applicable)

Feature-based distance. This is an aggregate measure of seven sub-scores, which are all calculated following the same procedure but use a different feature $f:\mathbb{R}^l\to\mathbb{R}$ of the time series as embedding [95]. These are three deterministic components of time series (mean, trend, seasonality) and four stochastic features (standard deviation, skewness, kurtosis, lag-1 autocorrelation), all calculated on a per-sample basis. Furthermore, the measure assumes a generator of the form $g_\theta:\mathbb{R}^l\to\mathbb{R}^l$, where each synthetic sample is deduced from an old one. Based on a scaled version f^s of some feature and with a pair $(X,\hat{X}=g_\theta(X))$ of real and synthetic time series, the authors define its sub-score as

$$d_f(X,\hat{X}) = \left| f^s(X) - f^s(\hat{X}) \right|, \tag{34}$$

where the aggregate is the set of all seven d_f. The feature-based distance is targeted at and tested on univariate time series.

Fréchet distance was originally introduced in [49] to calculate the similarity of two curves from the ordering and location of their points in $\mathbb{R}^d$. Much later, Hazra and Byun [94] used it for the evaluation of synthetic time series by interpreting them as point sequences on such curves. This is done directly to a real time series on one, and its synthesis on the other side, implying that this version is applicable to generators of the form $g_\theta:\mathcal{X}\to\mathcal{X}$ only. The score is always in $\mathbb{R}^+$, where lower is better. In the following, we use this version when referring to Fréchet distance, although applications on the distribution level similar to FID and Context-FID are possible as well.

Incoming nearest neighbor distance (INND). Let $d \in \{\mathrm{ED}, \mathrm{DTW}\}$ be a distance measure, and $R \subseteq D_r$ be a random subset of the real dataset [79]. For each generated time series $\hat{X}$, the INND is given by

$$\mathrm{INND}(\hat{X}) := \min_{X \in R} d(\hat{X}, X). \tag{35}$$

The authors use PCA as an embedding before employing the function, but this is optional. The smaller the distance, the better. To get a scalar score for D_g, one can take the mean over all $\hat{X}$.

Multivariate, dependent DTW (DTW$_D$) is a rather simple modification to the algorithm computing the standard DTW in order to accommodate multivariate time series. Dynamic time warping requires to calculate the distance d between the values at two time steps $\mathbf{x} \in X, \mathbf{y} \in Y$, where X, Y are time series and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$. d is commonly the squared Euclidean distance. In the univariate case, this comes down to $d(\mathbf{x}, \mathbf{y}) = (x_1 - y_1)^2$. DTW_D allows to “warp” across the channels of X and Y by simply using the generalization to the k-dimensional space $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||_2^2 = \sum_i (x_i - y_i)^2$ [96]. Brophy [73] used the measure for evaluating a GAN producing multivariate time series. However, he failed to explain the extension of this sample-based measure to real and synthetic datasets D_r, D_g. In cases where each synthesis $\hat{X} \in D_g$ can be assigned an original $X \in D_r$ due to the (probabilistic) relationship $g_\theta(X, z) = \hat{X}$ for conditional input z, they find a straight-forward implementation as

$$\mathrm{DTW}_D(D_r, D_g) := \frac{1}{|D_g|} \sum_{\hat{X} \in D_g} \mathrm{DTW}_D(X, \hat{X}) \quad \text{with } \hat{X} = g_\theta(X), X \in D_r. \tag{36}$$

For general g_θ, however, taking the average DTW_D for many randomly selected pairs $X \in D_r, \hat{X} \in D_g$, if not all such pairs, is a reasonable and probable approach.

Nearest neighbor in training (NNT) is similar to INND in its formulation, but opposing in its purpose [34]. Where the INND tries to ensure that each generated sample stays true to the real data D_r via proximity to a real sample, this measure determines the degree to which the generator is memorizing the data $D_r^{train} \subseteq D_r$ it was trained on. For a random synthetic sample $\hat{x} \in D_g$, they select its K nearest neighbors in D_r^{train} with respect to the Euclidean distance via

$$\text{neighborhood} := \underset{x \in D_r^{train}}{\mathrm{topK}}\ \mathrm{ED}(\hat{x}, x). \tag{37}$$

Lin et al. [34] chose $K = 3$ and repeated the procedure multiple times, if not for all $\hat{x}$. One example is given in Fig. 20. A quantitative measure can be defined on the mean of the neighbor distances, while a qualitative one is given by visually comparing the plot of $\hat{x}$ with those of the neighborhood. The Euclidean distance may require an additional embedding $f : \mathcal{X} \to \mathbb{R}^k$ for the data. Finally, the higher the score and visually different the plots, the better.

Pairwise measures. With this term, we refer to a group of measures m characterized by their common form $m : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. They are only applicable to a pair of data objects, typically a real sample $x \in D_r$ and its synthesis $\hat{x} \in D_g$. In other words, we imply that each synthesis can be backtracked to a real sample from which it was deduced

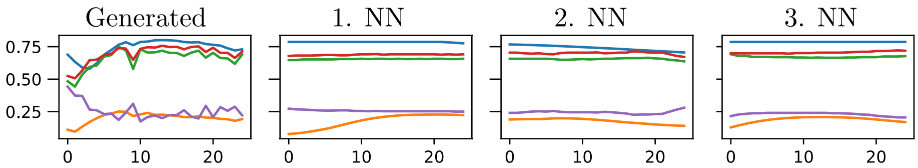


Fig. 20 Example for measure NNT. On the far left is the generated time series with a length of 25 and five channels, which is to be compared with its three nearest neighbors to its right. As we can see, the basic structures are similar, but the generated one contains much more noise and has a stronger drift at the beginning

somehow. Such measures typically only consider direct divergences between the samples. A possible application scenario includes the generation of very few or even a single particularly long time series, where it is clear which original the synthesis is expected to mimic. Specifically, we found the five measures: PCC, root mean square error (RMSE), MAE, percent root mean square error (PRE) [94], and cross-correlation score [55]. An extension to a measure for entire datasets may be given by averaging over each pair to obtain a score or using visualizations such as plots and histograms.

Plausible deniability. Targeting data privacy, this measure is an independent extension to probabilistic generative models of the form $g_\theta : \mathcal{X} \to \mathcal{X}$ that allow the efficient calculation or estimation of $\Pr\{\hat{x} = g_\theta(x)\}$ for real sample x and its synthesis $\hat{x}$ [33]. In Fig. 21, we provide a diagram depicting the steps of applying the measure. The idea is to take a high-utility generative model providing the synthetic samples and append a sample-level privacy test, which decides whether to release the sample or draw a new one from the generator. This way, high-quality data can be produced while making sure every synthesis adheres to a privacy policy. The test's underlying concept is to allow a synthetic sample $\hat{x}$ if and only if there exists an entire set of real data points D_x that could have caused $\hat{x}$ with similar probability. As a consequence, an adversary can no longer infer with confidence which of these $x \in D_x$ is actually part of the real dataset. Therefore, the creator of the synthetic set could always plausibly deny the membership of a particular data point.

Quadratic variation. This measure targets the time series generator's ability to reproduce the temporal structure of each channel [50]. Quadratic variation originally comes from the analysis of stochastic processes and can be simplified for time series to

$$\mathrm{QVar}(X) = \sum_{i=1}^{T} (X_i - X_{i-1})^2. \tag{38}$$

It is initially only applicable to individual univariate time series and needs further aggregation to be useful for the dataset-level. According to Boursin et al. [51], this is realized by computing the MSE between real samples on one side and synthetic samples on the other side. However, neither there nor in the original work is the order of the samples defined needed for the MSE or the aggregation method across channels.

Real to synthetic similarity (RTS). Proposed in [46], this measure compares the (approximate) similarity between real and synthetic samples to that among real samples. Similarity is determined by applying the cosine similarity to selected pairs of each dataset. To approximate the real-to-synthetic similarity for a synthetic sample $\hat{x}$,

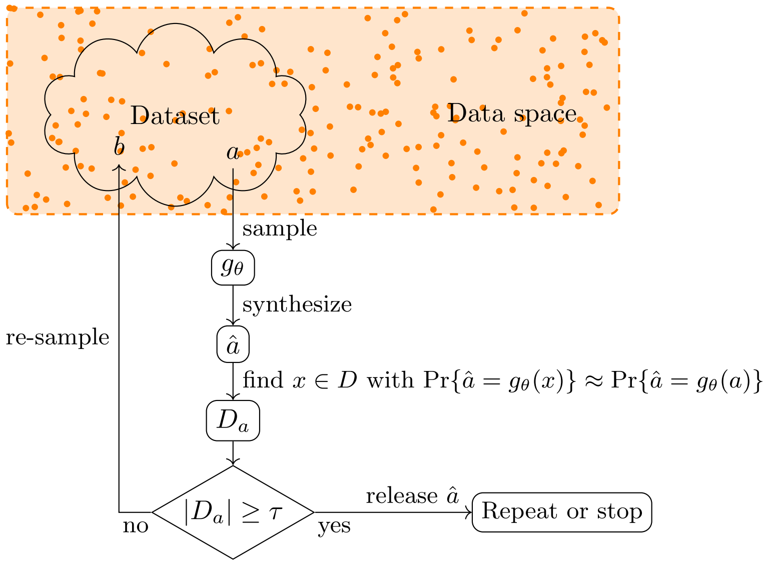


Fig. 21 Flow chart illustrating measure plausible deniability. In the upper part, real data space $\mathcal{X}$ and data set are indicated. Below, a real data object a is sampled and used as input to synthesize $\hat{a}$. If we can find at least $\tau \in \mathbb{N}$ other real objects x as likely to cause $\hat{a}$ as a itself, we are done. Otherwise, the procedure must be repeated until a suitable a is found

Table 3 Sample-level evaluation measures found during literature study

Measure	Applicability	Embedding	L	C	Uses
Authenticity	Unrestricted	LSTM-AE	×	✓	[87]
ACF	Time series	–	×	✓	[34, 38, 60, 93]
Cointegration tests	$g_\theta : \mathbb{R}^{l\times d} \to \mathbb{R}^{m\times d}$	–	×	×	[38]
Covariance matrix	unrestricted	–	×	×	[5, 61]
Feature-based distance	$g_\theta : \mathbb{R}^{l} \to \mathbb{R}^{l}$, univariate	$f : \mathbb{R}^k \to \mathbb{R}$	×	×	[95]
Fréchet distance	$g_\theta : \mathcal{X} \to \mathcal{X}$	Case-dependent	×	×	[50, 94]
INND	Unrestricted	PCA (opt.)	×	✓	[9, 79]
Moments	Unrestricted	–	×	×	[59]
DTW$_D$	Time series	–	×	✓	[73]
NNT	Low-dim time series	–	×	×	[34]
Pairwise measures	$g_\theta : \mathcal{X} \to \mathcal{X}$	case-dependent	×	×	[55, 94]
Plausible deniability	$g_\theta : \mathcal{X} \to \mathcal{X}$	$f : \mathcal{X} \to \mathbb{R}^k$	×	×	[33]
Quadratic variation	$g_\theta : \mathcal{X} \to \mathcal{X}$	–	×	×	[50, 51]
RTS	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[46]
Realism score	Unrestricted	$f : \mathcal{X} \to \mathbb{R}^k$	×	✓	[69]
Securities order marginals	$g_\theta : \mathcal{X} \to \mathcal{X}$	–	×	×	[97]
Visual assessment	Unrestricted	Case-dependent	×	✓	[5, 9, 38, 39, 41, 56, 60, 61, 67, 73, 79, 85, 88, 94]
α-precision	Unrestricted	LSTM-AE	×	✓	[87]

Next to each measure, we provide potential restrictions in terms of applicability to a specific generation method or type of data, as well as the embedding needed and works it has been used in so far. Column "L" indicates if the measure is designed for labeled data (✓), unlabeled data (×), or both (◊). "C" says if code is publicly available (✓) or not (×). "Uses" refers to the applications of the measure to general or time series synthesis evaluation found, where the first reference is also the first application (and authorship, if applicable) they calculate its average cosine similarity to 10 random real samples $x_1, \ldots, x_{10} \in D_r$, resulting in the score

$$\mathrm{RTS}_{\hat{x}} := \frac{1}{10} \sum_{i=1}^{10} \frac{\hat{x} \cdot x_i}{||\hat{x}||_2 \cdot ||x_i||_2}. \tag{39}$$

This can be used on individual samples to guide training or on a dataset-level for evaluation purposes at the end. As a baseline, they also calculate the average real to real similarity (RTR) for real dataset D_r using

$$\mathrm{RTR} := \binom{|D_r|}{2} \sum_{i \neq j} \frac{x_i \cdot x_j}{||x_i||_2 \cdot \left|\left|x_j\right|\right|_2}. \tag{40}$$

The RTR needs to be computed only once per real dataset. The goal are similar RTR and RTR scores, meaning $\mathrm{RTS} - \mathrm{RTR} \approx 0$, where $RTS, RTR \in [0, 1]$. In case the time series happen to be univariate, one can apply the measure directly. Otherwise, an embedding $f : \mathcal{X} \to \mathbb{R}^k$ is required.

Realism score is intended as an adaption of the improved precision measure to the sample-level, in order to rank individual samples by their quality [69]. In practice, it approximates the distance of a synthetic sample to the real data manifold using a continuous extension of the binary indicator function $\mathbf{1}\{\cdot\}$ used in improved precision and improved recall. For a synthetic $\hat{x} \in D_g$ and real dataset D_r, realism score (R) is given by

$$R(\hat{x}, D_r) := \max_{x \in D_r'} \left\{ \frac{\mathrm{dNN}_K(x, D_r)}{\left|\left|\hat{x} - x\right|\right|_2} \right\}, \tag{41}$$

where

$$D_r' := \left\{ x \in D_r \mid \mathrm{dNN}_K(x, D_r) < \underset{x \in D_r}{\mathrm{median}} \left\{ \mathrm{dNN}_K(x, D_r) \right\} \right\}. \tag{42}$$

D_r' is the half of D_r for which the distance to the Kth-nearest neighbor is smaller than the median dNN_k. This pruning of D_r strives to shrink the data manifold down to the densest region to increase robustness to outliers. Note that dNN_k also uses the Euclidean distance. The higher R the more realistic the sample, which effectively means that realism is determined by how close a synthetic is to a real sample after embedding $f : \mathcal{X} \to \mathbb{R}^k$.

Securities order marginals is a specialized measure tailored towards order streams observed in security transaction systems of stock exchanges [97]. It measures the quality of a generated sequence by computing five specialized statistics, targeting different channels like price and volume. Each statistic covers a marginal relevant to the financial domain, namely the distributions of security price, quantity bid/asked, inter-arrival time of orders, the evolution of the number of orders over time, and the ratio of bid-to-ask orders over time. For each statistic, real and synthetic samples are compared by calculating the Kolmogorov-Smirnov distance between the real marginal and synthetic marginal distribution. It is always in the interval [0, 1], where 0 is best. Consequently, the overall score for a pair $(X, \hat{X})$ is a quintuple of the similarities as expressed by the distances above.

Visual assessment. Using human judges as decision-makers on synthesis quality is a very simplistic and popular approach [4]. Indeed, entire works are built around the idea of making the assessment as clear and fast as possible using visualization tools. In the most basic form, evaluators may just look at a plot of each channel of the sequence. More complex approaches may account for their frequency domain using DFT, Wavelet transform [98], and z-transform [99] or specialized embeddings, extrapolating certain features of interest. Arnout et al. [79] propose a visualization framework that allows tracking of the appearance of individual samples during the training of the generation model and assessing the final result. In addition to ordinary plots, they also employ *TimeHistograms* [100] and *Colorfields* [101] to create more appealing embeddings. In two works, judgments are passed by domain experts and aggregated using mean opinion score [5, 39].

α-precision. Mirroring β-recall, this is a parameterized measure denoted by P_α, where $\alpha \in [0, 1]$ is the fraction of real samples assumed to be "typical", that is, no outliers [87]. It represents the fraction of synthetic samples resembling that fraction of real data. In other words, a high score ensures that the typical generator output is similar to typical real data, where typical can be interpreted very loosely on the one end or strictly on the other. Its formal definition and aggregation via mean absolute deviation is symmetrical to that of β-recall, that is, based on a minimum volume subset of the support of the synthetic distribution. Single real samples are assigned a value of 1 if they fall within the subset, and otherwise 0. Based on the mean absolute deviation between precision curve and fraction α, a summarizing score called integrated P_α is given by

$$IP_\alpha = 1 - 2\int_0^1 |P_\alpha - \alpha| d\alpha. \tag{43}$$

An example for the precision curve can be found in Fig. 19 center. $IP_\alpha \in [0, 1]$ where higher is better. Similar to β-recall, an embedding is required. By default, this is an LSTM-based autoencoder model.

Analyzing evaluation measures

In this section, we conduct a three-fold analysis of the measures above as per Subsection "Acquisition and systematization of knowledge". First, we introduce structure to the collection of measures via a taxonomy of criteria and evaluation measures. Afterwards, we investigate their impact in Subsection "Theory and practice of evaluation measures". Finally, in Subsection "Requirements on the input data format", we outline the relevance of time series length, dimensionality, and dataset sizes.

Taxonomy of evaluation measures and criteria

In the following, we suggest grouping the 83 reviewed measures by evaluation criteria. Simply put, each criterion refers to a specific data property that each measure is designed to test. For instance, spatial correlation and MIA check different data characteristics to evaluate quality. This is why, commonly, a set of measures is used for evaluation, covering a variety of properties and, in turn, aspects of quality of synthetic time series. Currently, all these measures coexist in literature without structure or order, which prohibits researchers and practitioners from quickly and clearly identifying measures suitable to the properties they wish to test. Notably, Borji [11] previously addressed the problem for the evaluation of synthetic images through the structure of their article itself. Instead, we propose a taxonomy of evaluation criteria and measures, which provides an explicit and search-friendly solution to this hindrance. The result is depicted in Fig. 22 as a horizontal tree structure with the root on the left and the measures as leaves on the right. From the root outward, we arranged different criteria in boxed nodes, which themselves fan out to measures and sometimes other, subordinate criteria. Note that measures can also be subordinated under other measures if they test a more specific property than the general one, for instance, predictive score under TSTR. Additionally, we follow previous works [14, 39, 85] and incorporate another differentiation between measures into the figure based on color coding: Qualitative measures (orange) are characterized by their need for human interpretation of an intelligible but ambiguous result such as t-SNE. Quantitative measures (cyan), on the other hand, produce a numerical or categorical value, allowing a clear interpretation. In gray, we mark notable secondary purposes (i.e., criteria) that a measure may fulfill.

We constructed this taxonomy in three steps: First, we clustered the measures based on the properties they tested. In case a measure has secondary purposes, it may belong to multiple clusters. Then, we assigned each cluster a suitable criterion, which was already mentioned in the literature in most cases. Finally, we arranged the clusters in a hierarchy such that more general clusters (e.g., *fidelity*) subsume more specific ones (e.g., *temporal dynamics*). In the following, we explain the chosen criteria and the assignment of measures in top-down order.

We start with *fidelity*, which refers to an individual synthesized time series to preserve characteristics, patterns, and noise levels present in the underlying real time series, often associated with notions of resemblance or typicality. While we found most measures in this cluster to be moved into four sub-criteria, we assign AIS, which produces likelihoods for each time series, and TRTS, which is based on the performance of a real-trained model on synthetic data, to that criterion directly. Both represent broad indicators able to catch many facets of *fidelity*. *Coherence* is one sub-criterion by which we denote the consistency of the time series' internal structure, especially along channel and time axes. Its only sub-criterion, *temporal dynamics*, contains AED, which is concerned with the preservation of periodicities within the synthetic time series. The second one is *perceptual quality*, meaning the aspects of quality perceptible by human judges during the inspection of synthetic data. We place visual assessment here.

With *instance similarity*, we opened a third cluster with eleven measures comparing a real time series to a synthetic one, determining the similarity on a sample-level. Directly assigned are ACS, correlation structure, feature-based distance, pairwise measures, RTS, and elementary statistics. ACS extracts and compares seven features from each time series individually. Similarly, feature-based distance extracts its own set of features, while correlation structure calculates the covariance matrix for each data instance, comparing real and synthetic time series pair-by-pair. Pairwise measures and RTS compare

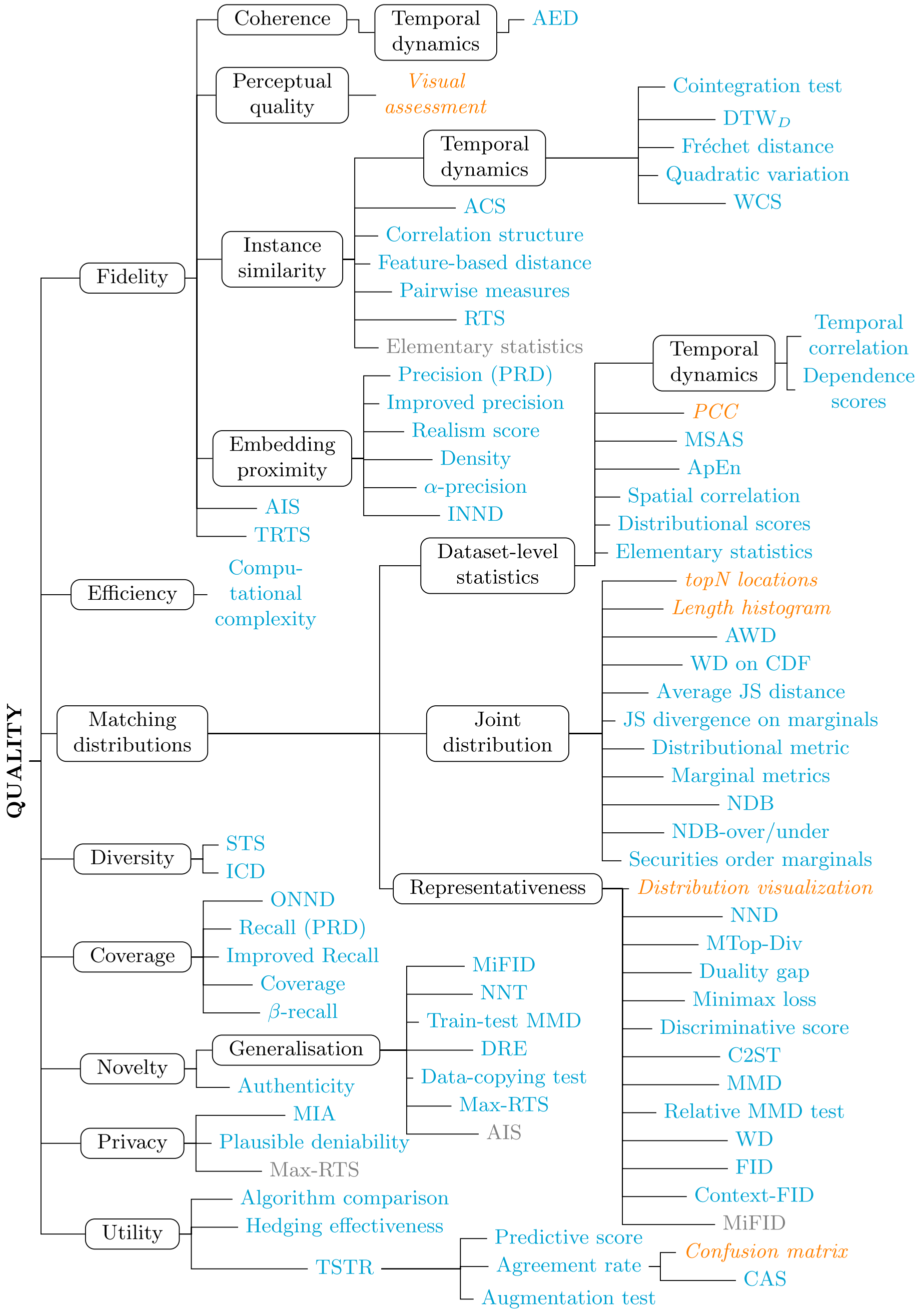


Fig. 22 Taxonomy of the evaluation of time series syntheses. This includes quality as the overarching goal, criteria representing aspects of quality, and finally, measures quantifying them. The measures are color-coded, where orange stands for qualitative, cyan for quantitative, and gray for secondary purposes

individual real and synthetic time series using different similarity functions. Elementary statistics typically compare the calculated statistical score on the dataset-level and sometimes between instances. As the remaining five measures all target temporal aspects of *instance similarity*, we introduced a separate sub-category called *temporal dynamics* once again. These measures are cointegration test, DTW_D, Fréchet distance, quadratic variation, and WCS, addressing trend comparison, time warping, curve alignment, stepwise changes, and wavelet coherence, respectively.

Lastly, we place measures that interpret the *fidelity* of a synthetic sample as the proximity to its nearest real neighbors in an embedding space under criterion *embedding proximity*. We identified six measures following this description. To handle the two sides of PRD, we divided the measure into its two components and deemed precision suitable for evaluating *fidelity*. Its conceptual successors, improved precision, realism score, density, and α-precision have the same purpose and thus belong here as well. INND is a special case in that it only takes the minimum distance to one real sample.

Efficiency highlights the ratio of synthetic data quality achieved to the effort required. The only such measure found is computational complexity, which tracks the computing time required to train a generator and create the synthetic data. Then, it puts it into relation to the other aspects of quality.

While *fidelity* focuses on sample quality, *matching distributions* requires real and synthetic data distributions P and Q to match in some aspects, if not overall. For instance, a generator may produce time series of exceptional quality but in the wrong proportions, such that P and Q still do not match. In this case, we differentiate three sub-criteria based on these aspects. First, there is a group *dataset-level statistics* of eight measures that compare P and Q with regard to some statistical characteristics, a weak demand given the complexity of typical time series datasets. For PCC, this characteristic is the Pearson correlation, provided qualitatively as a heat map for each dataset. MSAS does not extract one but multiple statistical characteristics like channel-wise mean or sequence length and subsequently compares real and synthetic datasets on these. As their name indicates, ApEn, temporal correlation, and spatial correlation all deliver a statistical value for the entire dataset. Along with dependence scores, we split off temporal correlation into a sub-criterion *temporal dynamics*. Dependence scores compare the ACF between real time series and their syntheses. Two measures remain, one of which being distributional scores, computing and comparing skewness and kurtosis for the single real and the set of synthetic time series. At last, we have the group of elementary statistics, such as statistical moments or ACF.

The next sub-criterion is called *joint distribution*, subsuming measures that view real P and synthetic Q as joint distributions of multiple marginals and compare the two with respect to one or more of these. In this cluster, we aggregated 11 measures. First, there is topN locations, which compares the distributions over bins of coordinates extracted from the real dataset on one, and the synthetic dataset on the other. The coordinates represent locations that can be visited in each step of the time series. The second one is length histogram, which is intended for the evaluation of synthetic time series of varying lengths. Here, the marginal distribution is length. AWD focuses on the various periodicities within the time series in each dataset, considers them as marginal distributions, and compares them using the WD. In the original work, they state to use it for *diversity*, but we find *joint distribution* to be the better-fitting cluster. WD on CDF first computes an empirical CDF over a discrete data space or binning of a real-valued data space. In the second step, the Wasserstein-1 distance is computed between the CDF of real and synthetic data, respectively. Since binning is typically required, the CDF only reflects a part of the actual distribution. Average JS distance extracts a set of features from each time series and interprets the values of each feature as a marginal distribution. After applying these steps to real and synthetic datasets, the JS divergences are determined and averaged. As one can imagine, JS divergences on marginals is a generalization of this concept, conceived by different authors and apparently without influence from one to the other. Distributional metric, similar to others before, is based on an empirical distribution over bins of data objects. As opposed to WD on CDF, for instance, it uses the PDF on the binnings for both real and synthetic data and computes the bin-wise absolute difference between the two. Nevertheless, it operates on marginal distributions and not the real ones. In the case of marginal metrics, the marginals of interest are the distributions of each time step. The last two, NDB and NDB-over/under, also use binnings to approximate the actual distributions, concentrate on the regional distributions and thereby define marginals. Securities order marginals compares the statistics of marginal distributions of financial time series such as price and volume.

By *representativeness*, we denote a criterion that requires the overall distributions to be similar, simply put, $P \approx Q$. We reviewed a variety of measures that pursue this fundamental aspect of quality, among them the established, more general MMD, WD, or the qualitative distribution visualization approaches like t-SNE. We also found derivations like the relative MMD test. We also have C2ST, which uses a binary classifier model and a hypothesis test on its predictions, whereas the former methods use kernel functions and distances. Furthermore, five measures (NND, MTop-Div, duality gap, minimax loss, and discriminative score) use the loss or accuracy of a binary classifier model trained to distinguish real from synthetic samples. Their results depend solely on the capabilities of the classifier, and only what it considers relevant goes into the differentiation task. Finally, FID and its two variants are in this cluster. All three use feature extraction layers from neural models to map the data into a feature space before applying the Fréchet distance to compare the distributions. In the case of MiFID, FID is one of its components, and testing the distributions for a match its secondary purpose.

Coverage refers to the idea that the synthetic data as a whole should "cover" the entire region assumed by the real data distribution. It is occupied by four measures developed in sequence with the intention of improving on their respective predecessor, and a fifth one developed independently. The first in line is recall, which is one part of the composite measure PRD. Then, there are its successors, improved recall, coverage, and β-recall proposed in this order. They are all similar in that they use the proximity of real to synthetic objects in a feature space as an indicator of the generator's ability to cover the entire support of the real data distribution. The fifth, ONND, is based on the distance of each real sample to its closest neighbors in the synthetic dataset. With our assessment of measures coverage and ONND, we disagree with their respective authors, who consider them to evaluate diversity primarily. Still, we acknowledge *diversity* as an aspect of quality but assign two what we believe to be more suitable measures: STS and ICD. STS measures and compares the average cosine similarity within real and synthetic datasets, respectively, and interprets the result as an indicator of diversity (or the lack thereof). ICD follows the same procedure but with a different similarity measure.

In the literature, the terms novelty and generalization are both used to describe a generator that produces samples beyond noisy versions of the training data. We differentiate the terms here and use *generalization* if reproducing some training samples is acceptable and *novelty* if each generated time series must be noticeably different from all training samples. We identified one measure suitable to test *novelty* directly, namely authenticity. It can be applied to each synthetic sample individually and even provide a binary decision on whether it is novel using a hypothesis test. The selection of measures for *generalization* is more extensive. MiFID employs a so-called memorization distance based on the cosine similarity between samples and penalizes the generator for producing samples too close to training samples. NNT computes the neighborhood of each synthetic sample to determine if its neighbors in the real dataset are too close. Intuitively, MMD just compares two distributions with respect to their overlap in density. However, Esteban et al. [56] consider an additional setting in which the discrepancy between the real train and a real hold-out set is compared to that between the synthetic set and the same hold-out set. We call this variant train-test MMD. What is more, they developed this idea further and conceived DRE for the same purpose. The data-copying test uses DTW to calculate the average distance between the train and synthetic set as well as between the train and the hold-out set. Afterwards, a hypothesis test determines if the average distance between the first pair of sets is significantly shorter than between the second. Max-RTS computes the most similar pair of real and synthetic samples and uses it as an indicator for *generalization*. Lastly, AIS tests *generalization* in a secondary role by comparing the average log-likelihood on training and test sets.

Privacy is an often-cited criterion for evaluation and a main reason for synthesizing data in the first place. Upholding *privacy* means reducing - if not eliminating - the risk of disclosure of sensitive information within the original dataset. MIA and plausible deniability set out to control and uncover the degree to which information on individual real data objects passes through into the generated set. As Max-RTS computes the closest pair of real and synthetic objects in terms of cosine similarity, it can be used as an indicator for the worst-case information leakage on a single data object.

Utility interprets the quality of synthetic data as its usefulness in a downstream (machine learning) task. The eight measures in its cluster are not directly assigned, but only TSTR, hedging effectiveness, and algorithm comparison. The remaining five are specializations of TSTR, a generic measure of *utility*. Downstream task T and the ML model for T are free of choice by default. By fixing T, we get a variant TSTR(T). In the case of time series, this is usually prediction and sometimes classification. Depending on T, another aspect of *utility* is evaluated each time. Except for minor implementation differences, predictive score is a variant of TSTR with prediction as T. Agreement rate, confusion matrix, and CAS use classification as task. Agreement rate includes a strategy for handling unlabeled time series data, which makes it slightly more general. The last variant is the augmentation test, where the task is augmentation of the real data using the synthesis in a predefined proportion. Algorithm comparison, however, not only utilizes one kind of ML model but several and bases its score on the similarity of the two performance rankings of these models, one on the real dataset and one on the synthetic set. Hedging effectiveness is different in that it does not train the same model on the real data as on the synthetic but uses a simple baseline for comparison.

Theory and practice of evaluation measures

Next, we analyze the usage statistics of these evaluation measures in the academic literature. We refrain from investigating the statistics of real-world adoption, as this would necessitate surveys conducted among practitioners. Instead, we discuss the impact a measure has on subsequent works depending on what kind of work it was proposed. To this end, we partition the collection of measures above into two sets based on whether they originate from work focused on the evaluation of synthetic data [69, 70] or on their generation [26, 92]. In the latter case, proposed measures are side contributions and directly employed in evaluating the presented generators. We call the first class of works *Theory* and the second *Practice*. Transitively, we apply the same distinction for the measures presented therein.

By reuse, we mean the number of works in *Practice* in which a measure was used minus the initial use. In other words, how often it was adopted. We find this statistic to be a more meaningful indicator for impact than citations, for instance, as these can instead reflect reuses of other measures in the work, references of the generator presented, or a citation for a survey article like ours. Following our approach, we list the number of reuses as well as the year of inception or first use, respectively, for each measure in Table 6. The latter allows us to take the age of the measure into account. The 28 measures assigned to *Theory* are listed first, and the 55 in *Practice* second and are additionally marked with *. Furthermore, note that we consider the measures of groups distribution visualization (QQ-plot, scatter plot, latent space visualization, PDF visualization, dimension-wise probability, PCA, t-SNE) and elementary statistics (ACF, covariance, moments) individually.

Based on these data, we immediately observe that most measures are never reused, some are used again once, and only a few have reached some level of popularity in the community. To support this further and differentiate *Theory* and *Practice*, we calculate four intuitive indicators of impact. To this end, let G be a set of measures, namely one of *Total*, *Practice*, and *Theory*. Further, we define $r(m)$ as the number of reuses of measure m.

As the first indicator, we take the maximum number of reuses within each group,

$$I_{max} := \max_{m \in G}(r(m)), \tag{44}$$

which serves as an indicator for the extremal case of impact. This value is only 2 for *Theory*, compared to 11 for *Practice*. The picture repeats for the next highest reuse values, showing that the most impactful measures belong to *Practice*.

Next, we compute the relative and normalized impacts I_r and I_n, which consider the aggregate number of reuses in *Theory* and *Practice*, expressed as a fraction of total reuses in the first case, and a normalization by the group size in the second case.

$$I_r(G) := \frac{\sum_{m \in G} r(m)}{\sum_{m \in P \cup T} r(m)} \quad \text{and} \quad I_n(G) := \frac{\sum_{m \in G} r(m)}{|G|} \tag{45}$$

As a result, I_r attributes only 6.3% of the aggregated impact to Theory, the remaining 93.7% to Practice. Similarly, the average impact of a measure in *Theory* is only $I_n(T) \approx 0.179$, but $I_n(P) \approx 1.345$ for *Practice*. While this is considerably better, it still means that combined, measures are not even once reused on average.

Table 4 Measures with reuse statistics and first appearance

Measure	Uses	Year	Measure	Uses	Year
Agreement rate	0	2017	FID	0*	2019
AIS	0	2017	Hedging effectiveness	0*	2022
Authenticity	0	2022	JS divergence on marginals	0*	2018
C2ST	0	2017	Latent space visualisation	0*	2021
Coverage	0	2020	Length histogram	0*	2019
Data-copying test	0	2020	Max-RTS	0*	2018
Density	0	2020	Moments	0*	2017
Duality gap	0	2019	MSAS	0*	2022
Improved precision	0	2019	DTW_D	0*	2020
Improved recall	0	2019	NNT	0*	2019
ICD	0	2022	QQ-plot	0*	2018
INND	0	2019	RTS	0*	2018
MTOP	0	2021	Scatter plot	0*	2018
MiFID	0	2021	Securities order marginals	0*	2020
Minimax loss	0	2019	STS	0*	2018
NND	0	2019	WCS	0*	2022
NDB	0	2018	Augmentation test	1*	2019
NDB-over/under	0	2020	Confusion matrix	1*	2018
ONND	0	2019	Correlation structure	1*	2022
Plausible deniability	0	2017	Covariance matrix	1*	2019
Realism score	0	2019	Dependence scores	1*	2019
Relative MMD test	0	2016	Dimension-wise probability	1*	2017
Temporal correlation	0	2022	Distributional metric	1*	2019
α-precision	0	2022	WD	1*	2020
β-recall	0	2022	Fréchet distance	1*	2020
CAS	1	2019	Marginal metrics	1*	2022
Computational complexity	2	2017	Pairwise measures	1*	2019
PRD	2	2018	PDF visualisation	1*	2019
Algorithm comparison	0*	2019	Quadratic variation	1*	2022
ApEN	0*	2021	topN locations	1*	2018
ACS	0*	2022	TRTS	1*	2017
AED	0*	2022	MIA	2*	2018
Average JS distance	0*	2022	PPC	2*	2019
AWD	0*	2022	Spatial correlation	2*	2020
Cointegration test	0*	2021	ACF	3*	2019
Context-FID	0*	2022	MMD	3*	2017
DRE	0*	2017	Discriminative score	5*	2019
Distributional scores	0*	2019	PCA	5*	2019
WD on CDF	0*	2019	Predictive score	7*	2019
Feature-based distance	0*	2018	t-SNE	8*	2019
Feature-based correlation			Visual assessment	11*	2017
analysis	0*	2022	TSTR	11*	2017
Average	0.952	-	Median	0	-

List of all measures with the number of reuses in the works reviewed and year of first appearance as an evaluation measure for time series synthesis. The reuse count reflects the impact on generator developers. * indicates that the measure belongs to *Practice*. The list is sorted by group affiliation (*Theory* first) and the number of reuses within each group (starting at 0 and increasing)

As the final indicator, we consider the number of measures without any reuse and therefore impact on subsequent evaluations. Additionally, we adjust this number through division by the group size. The higher such a score is, the smaller its past relevance for the evaluation task. Formally, we define it as

$$I_{no} := \frac{|\{m \in G \mid r(m) = 0\}|}{|G|}. \tag{46}$$

Overall, we find that over half of the reviewed measures made no impact in this regard, resulting in a normalized number of measures without impact of $I_{no} \approx 0.643$. Moreover, this indicator is especially bad for the 28 measures from *Theory*, of which only three found reuse. Hence, its score is even worse with $I_{no} \approx 0.893$. For *Practice*, the indicator is significantly better at just over half, $I_{no} \approx 0.527$.

We summarize the results in Fig. 23. From these indicators and the high number and variety of measures itself, we draw two main insights:

1. Many of the works reviewed use their own custom measures. In contrast, significant usage of existing ones is limited to a select few, such as TSTR and the elementary visual assessment. There is no consistent, commonly accepted combination of evaluation measures that researchers of time series synthesis have agreed on yet.
2. Against intuition, measures proposed in dedicated works on evaluation (group *Theory*) have hardly impacted the reality of how generative models are actually being evaluated. Hence, we come to the conclusion that there is currently a significant discrepancy between what dedicated works on synthesis evaluation propose and what practitioners actually use.

The reason for this situation is unclear to us and most likely requires a survey among the authors as to why they made the choices as we see them. Instead, we can only make fair guesses with the information at hand. First, a number of measures have only recently been presented, such that reuse in early works is impossible. Furthermore, the publications in *Theory* might not have reached the level of attention required to have a significant influence on interdisciplinary work such as medical informatics. Third, some of the proposed measures might be too complex for the audience of works from *Practice*, mainly reviewers who do not value such measures. For instance, MTop-Div and duality gap come to mind. Another reasonable explanation would be that generator developers do indeed consider them but regard these measures as unsuited and hence use their own or those used in works they compete with. An example of such an influential work from *Practice* is [26], which, proposed in 2019, introduced distribution visualization via t-SNE and PCA, discriminative score, and predictive score, all with high reuse rates.

Requirements on the input data format

Not all measures are equally suited to evaluate all kinds of time series data. Among other aspects, they differ in how length and dimensionality of the sequences and the sizes of the datasets involved are being handled. Here, we look at how the measures reviewed in this survey come off in terms of applicability, that is, if there are any limitations or recommendations for length, dimensionality, or size. Our findings are summarized in Table 7.

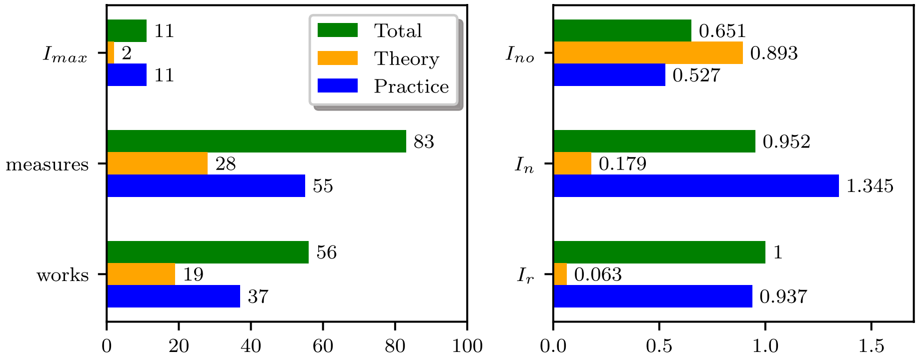


Fig. 23 Summary of impact indicators for all measures (total), group *Theory*, and group *Practice*. In the left plot, we show the number of works, number of measures, and maximum reuse value. In the right plot, we have relative impact score, normalized impact score, and the fraction of measures not reused (all bottom-up). Note the difference in scale on both x-axes

For each measure, we provide information on time series length in the second column, on its dimensionality in the third, on the size of the real dataset in the fourth, and on the size of the synthetic set in the last column. The measures with identical entries are grouped and sorted alphabetically. If a measure can be applied arbitrarily with respect to one of these four aspects, we mark the respective cell as "no restrictions" (n.r.). In case the measure requires an embedding function upon which length and dimensionality of the possible input format depend, we indicate this by the abbreviation e.i.d. for "embedder input dimension". Occasionally, the sequence length must remain constant or, to the contrary, vary across the input. If measures are more specific, we put the limitations in the respective columns. Furthermore, if the value is marked with an asterisk (*), it is our recommendation based on the above analysis; otherwise, it was taken from the original work. Note that these are just estimates, meaning a dataset size of 900 most likely causes no problem when the recommendation says ≥ 1k*. For MIA and correlation structure, unfortunately, appropriate values are unknown. While lengths can vary, we implicitly assume $\dim(D_r) = \dim(D_g) =$ constant.

Our estimates (*) mostly follow the rationale that a certain amount of data must be available to reliably evaluate the synthetic data with the respective measure, especially since a portion of the dataset is to be held out from training the generative function. For measures that require a learned embedding or train a model for a downstream task with the data, we propose a minimum of 1000 samples per dataset. The same goes for measures that apply binning to estimate P and Q, such that some degree of statistical representativeness is maintained. For the same reason, we suggest a minimum length for distributional metric. For measures learning a model to distinguish real from synthetic data, the proposed minimum size is 10 000. Securities order marginals, AED, and feature-based correlation analysis require multivariate time series with different restrictions on the number of channels. Visual assessment is an outlier in this regard, as we argue for upper bounds in length, dimensionality, and size of D_g. This is based on the assumption that only time series of limited complexity and number can be visually judged reasonably.

To close out, we summarize our takeaways regarding the requirements for input data formats below. In terms of length, almost half the measures depend on the capabilities

Table 5 Measure requirements on TS length, dimensionality, and dataset sizes

Measures	Length	Dim	Size(D_r)	Size(D_g)
Agreement rate, algorithm comparison, AIS, augmentation test, average JS distance, CAS, C2ST, confusion matrix, context-FID, C_T, DRE, WD, WD on CDF, JS divergence on marginals, MMD, MiFID, PRD, predictive score, relative MMD test, topN locations, TRTS, TSTR, β-recall	e.i.d.	e.i.d.	$\geq$ 1k*	$\geq$ 1k*
Max-RTS, RTS, STS	const.	n.r.	$>$ 500	$\geq$ 10
ApEn, hedging effectiveness, marginal metrics, WCS	const.	n.r.	n.r.	n.r.
Discriminative score, duality gap, minimax loss	e.i.d.	e.i.d.	$\geq$ 10k*	$\geq$ 10k*
Authenticity, α-precision	e.i.d.	e.i.d.	$\geq$ 1k*	n.r.
ACS, AWD, cointegration test, computational complexity, elementary statistics, Fréchet distance, INND, ICD, MSAS, DTW$_D$, NNT, PPC, plausible deniability, ONND, quadratic variation, spatial correlation, temporal correlation	n.r.	n.r.	n.r.	n.r.
Coverage, density	e.i.d.	e.i.d.	$\approx$ 10k	$\approx$ 10k
Dependence scores	const.	1	1	n.r.
Distribution visualization	n.r.	n.r.	$\geq$ 100*	$\geq$ 100*
Distributional metric	$\geq$ 200*	n.r.	n.r.	n.r.
Distributional scores	n.r.	n.r.	1	n.r.
Feature-based distance	n.r.	1	1	1
Improved precision, improved recall	e.i.d.	e.i.d.	$\approx$ 50k	$\approx$ 50k
Length histogram	varying	n.r.	n.r.	n.r.
MTop-Div	e.i.d.	e.i.d.	$\geq$ 1k	10$\lvert D_r \rvert$
MIA	?	?	?	?
Correlation structure	const.	?	?	?
NND	e.i.d.	e.i.d.	$\geq$ 10k*	$\gg \lvert D_r \rvert$
NDB, NDB-over/under	n.r.	n.r.	$\geq$ 1k*	$\geq$ 1k*
Pairwise measures	const.	1	n.r.	n.r.
Realism score	e.i.d.	e.i.d.	$\approx$ 50k	n.r.
Visual assessment	$\leq$ 1k*	$\leq$ 5*	n.r.	$\leq$ 1k*
AED	n.r.	2	n.r.	$= \lvert D_r \rvert$
Feature-based correlation analysis	n.r.	$\geq$ 2	n.r.	$= \lvert D_r \rvert$
Securities order marginals	n.r.	$\geq$ 5	1	1
FID	e.i.d.	$= l$	n.r.	n.r.

List of length, dimensionality (column Dim), and the sizes of both real and synthetic datasets required or recommended for a successful application of each measure. The meaning of the abbreviations used are e.i.d. → embedder input dimension, i.e. the input format permitted by the embedding model, n.r. → no restrictions and const. → constant. "?" means that this value is unknown. (*) indicates a recommendation

of the embedding model, and they themselves operate on fixed-size output vectors. 25 measures have no restrictions, while 10 require the time series to have constant length, and a few others have special requirements. The situation for dimensionality is very similar. Most measures either depend on the capabilities of the embedding model or impose no restrictions at all. A few may only be applied to either uni- or multivariate time series. The partition between sample-level and distribution-level measures continues over to the size of input datasets. More precisely, if restrictions apply, datasets may contain only one sample in the former and typically thousands of time elements in the latter case to have a dependable estimation of the underlying distribution. Furthermore, the size of D_r and D_g is usually expected to be similar in magnitude, with notable exceptions like MTop-Div and NND.

Conclusion

Sharing and publishing data is a crucial part of data science research. It enables the verification of results claimed in past work and drives the participation of other researchers in the future. Synthetic data, including synthetic time series, are a vital enabler of sharing in situations where privacy concerns prohibit the use of the original directly. Due to use cases like this, data generation is a heavily researched field. However, evaluating synthetic time series is a complex task and still considered an open problem [4, 9, 10].

To address this problem, we reviewed and analyzed existing literature on evaluation measures for synthetic time series data. In 56 papers, we found 83 measures, described each, and provided many points of differentiation, such as applicability, dependence on embeddings, conditional generation, and code availability. Among other things, we observed that there is currently no universal, generally accepted approach to evaluating synthetic time series. Furthermore, many measures are insufficiently defined and lack public implementations, making reuse and reproduction troublesome and error-prone. Afterwards, we analyzed the reuse behavior of researchers observed in these works and found that dedicated evaluation measures (group *Theory*) have little to no resonance with practical works, proposing new generative models. Additionally, we introduced structure to the large and diverse set of measures via a taxonomy of quality. Here, we observed, for instance, that most measures are quantitative and test a wide range of criteria, with a majority focused on *fidelity* and *matching distributions*. Lastly, we looked at limitations to applicability regarding length, dimensionality, and dataset sizes. Findings include the frequent dependence on embedding models, restrictions on dataset sizes, and the number of channels.

Our study uncovered several directions for future research and improvements. In our opinion, the logical next step is to conduct extensive yet controlled experiments on the collection of proposed measures to test and compare their efficacy. For instance, a promising approach may be ablation studies in which only one experimental parameter, such as time series length or dimensionality, changes to isolate effects on the measure, or use custom-created time series with known characteristics. Besides, we have seen that many measures depend on preceding embeddings into a latent space. However, we do not know of any study analyzing the impact of architecture and training of the chosen embedding model. Furthermore, many measures reviewed cannot handle variable length time series. Those who can do so in principle, including embedding-dependent measures, have yet to be tested in this setting. As we expect to see more works experimenting with such data in the future, having measures and embeddings that can handle variable length well will likely become its own concern. Finally, we view the vast diversity of measures and their variants as a significant hindrance to practical evaluation, presentation of results, and capturing of generator progress as a whole. This is a challenge unique to the generation task compared to areas like time series forecasting or classification. Hence, future research would immensely profit from a widely accepted, reasonably sized set of qualified measures for the central evaluation criteria.

Abbreviations

ACF	Autocorrelation function
ACS	Average cosine similarity
AE	Autoencoder

AIS	Annealed importance sampling
ApEn	Approximate entropy
AUPRC	Area under precision-recall curve
AUROC	Area under ROC curve
AED	Average euclidean distance
AWD	Average Wasserstein distance
BOSS	Bag-of-SFA-Symbols
CAS	Classification accuracy score
CID	Complexity invariance distance
CDF	Cumulative distribution function
C2ST	Classifier two-sample test
C_T	Data-copying test
DL	Deep learning
DFT	Discrete Fourier transform
dNN_K	Distance to the Kth-nearest neighbor
DRE	Distribution of reconstruction errors
DTW	Dynamic time warping
DTW_D	Multivariate, dependent DTW
ED	Euclidean distance
FFT	Fast Fourier transform
FID	Fréchet inception distance
GAN	Generative adversarial network
ICD	Intra-class distance
p_i	Improved precision
r_i	Improved recall
IS	Inception score
IoT	Internet of things
INND	Incoming nearest neighbor distance
JS divergence	Jensen-Shannon divergence
JS distance	Jensen-Shannon distance
LSTM	Long short-term memory
MAE	Mean absolute error
Max-RTS	Maximum real to synthetic similarity
MIA	Membership inference attack
MiFID	Memorization-informed Fréchet inception distance
ML	Machine learning
MMD	Maximum mean discrepancy
MSCD	Maximum shifting correlation distance
MSAS	Multi-sequence aggregate similarity
MTop-Div	Manifold topology divergence
MSE	Mean squared error
NDB	Number of statistically different bins
NDB-over/under	NDB over-representation and under-representation
NND	Neural network divergence
NNT	Nearest neighbor in training
ONND	Outgoing nearest neighbor distance
PCA	Principal component analysis
PCC	Pearson correlation coefficient
PDF	Probability density function
PPC	Pairwise Pearson correlation
PRD	Precision and recall for distributions
PRE	Percent root mean square error
R	Realism score
RMSE	Root mean square error
RTR	Real to real similarity
RTS	Real to synthetic similarity
STS	Synthetic to synthetic similarity
TRTS	Train on real, test on synthetic
TS	Time series
TSTR	Train on synthetic, test on real
t-SNE	T-distributed stochastic neighbor embedding
WD	Wasserstein-1 distance
WCS	Wavelet coherence score

Acknowledgements

This publication was supported by the Open Access Publication Fund of the University of Wuerzburg and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 510552229.

Author contributions

All authors contributed to the study conception and design. MS conducted data acquisition and the evaluation of results. MS created the first draft of the manuscript, while RL, IF, SK, and AB revised it critically. AB supervised the work. All authors read and approved the final manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 510552229.

Availability of data and materials

All data generated or analyzed during this study are included in this published article.

Declarations

Ethics approval and consent to participate

Not applicable

Consent for publication

Not applicable

Competing interests

The authors declare no competing interests.

Received: 1 November 2023 Accepted: 16 April 2024
Published online: 07 May 2024

References

1. Lim B, Zohren S. Time-series forecasting with deep learning: a survey. Philos Trans Royal Soc A. 2021. https://doi.org/10.1098/rsta.2020.0209.
2. Ismail Fawaz H, Forestier G, Weber J, Idoumghar L, Muller PA. Deep learning for time series classification: a review. Data Mining Knowl Discov. 2019;33(4):917–63. https://doi.org/10.1007/s10618-019-00619-1.
3. Blázquez-García A, Conde A, Mori U, Lozano JA. A review on outlier/anomaly detection in time series data. ACM Comput Surv. 2021. https://doi.org/10.1145/3444690.
4. Brophy E, Wang Z, She Q, Ward T. Generative adversarial networks in time series: a systematic literature review. ACM Comput Surv. 2023. https://doi.org/10.1145/3559540.
5. Beaulieu-Jones BK, Wu ZS, Williams C, Lee R, Bhavnani SP, Byrd JB, et al. Privacy-preserving generative deep neural networks support clinical data sharing. Circ: Cardiovasc Q Outcomes. 2019. https://doi.org/10.1161/CIRCOUTCOMES.118.005122.
6. Petitjean F, Forestier G, Webb GI, Nicholson AE, Chen Y, Keogh E. Dynamic Time Warping Averaging of Time Series Allows Faster and More Accurate Classification. In: 2014 IEEE International Conference on Data Mining; 2014;470–479.
7. Dau HA, Silva DF, Petitjean F, Forestier G, Bagnall A, Mueen A, et al. Optimizing dynamic time warping's window width for time series data mining applications. Data Mining Knowl Discov. 2018;32:1074–120. https://doi.org/10.1007/s10618-018-0565-y.
8. Bauer A, Trapp S, Stenger M, Leppich R, Kounev S, Leznik M, et al. Comprehensive exploration of synthetic data generation: a survey. arXiv preprint. 2024. https://doi.org/10.4855/ARXIV.2401.02524.
9. Leznik M, Lochner A, Wesner S, Domaschka J. [SoK] The great GAN bake Off, an extensive systematic evaluation of generative adversarial network architectures for time series synthesis. J Syst Res. 2022. https://doi.org/10.5070/SR32159045.
10. Jarrett D, Bica I, van der Schaar M. Time-series generation by contrastive imitation. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2021. p. 28968–82.
11. Borji A. Pros and cons of GAN evaluation measures: new developments. Comput Vis Image Underst. 2022;215:103329. https://doi.org/10.1016/j.cviu.2021.103329.
12. Iqbal T, Qureshi S. The survey: text generation models in deep learning. J King Saud Univ - Comput Inform Sci. 2022;34(6):2515–28. https://doi.org/10.1016/j.jksuci.2020.04.001.
13. Figueira A, Vaz B. Survey on synthetic data generation, evaluation methods and GANs. Mathematics. 2022;10(15):2733.
14. Xu Q, Huang G, Yuan Y, Guo C, Sun Y, Wu F, et al. An empirical study on evaluation metrics of generative adversarial networks. arXiv preprint. 2018. https://doi.org/10.4855/ARXIV.1806.07755.
15. Eigenschink P, Reutterer T, Vamosi S, Vamosi R, Sun C, Kalcher K. Deep generative models for synthetic data: a survey. IEEE Access. 2023;11:47304–20. https://doi.org/10.1109/ACCESS.2023.3275134.
16. Borji A. Pros and cons of GAN evaluation measures. Comput Vis Image Underst. 2019;179:41–65. https://doi.org/10.1016/j.cviu.2018.10.009.
17. Dankar FK, Ibrahim MK, Ismail L. A multi-dimensional evaluation of synthetic data generators. IEEE Access. 2022;10:11147–58. https://doi.org/10.1109/ACCESS.2022.3144765.
18. Ji S, Luo J, Yang X. A comprehensive survey on deep music generation: multi-level representations, algorithms, evaluations, and future directions. arXiv preprint. 2020. https://doi.org/10.4855/ARXIV.2011.06801.

19. Fatima N, Imran AS, Kastrati Z, Daudpota SM, Soomro A. A systematic literature review on text generation using deep neural network models. IEEE Access. 2022;10:53490–503. https://doi.org/10.1109/ACCESS.2022.3174108.
20. Assefa SA, Dervovic D, Mahfouz M, Tillman RE, Reddy P, Veloso M. Generating synthetic data in finance: opportunities, challenges and pitfalls. In: Proceedings of the First ACM International Conference on AI in Finance; 2020. p. 1–8.
21. Theis L, van den Oord A, Bethge M. A note on the evaluation of generative models. In: International Conference on Learning Representations (ICLR 2016); 2016. .
22. Raghunathan TE. Synthetic data. Ann Rev Statist Appl. 2021;8(1):129–40. https://doi.org/10.1146/annurev-statistics-040720-031848.
23. Nikolenko SI. Synthetic data for deep learning. In: Gaw N, Pardalos PM, Gahrooei MR, editors. Springer optimization and its applications. Cham: Springer; 2021.
24. Shumway RH, Stoffer DS. Time series analysis and its applications. Berlin: Springer; 2017.
25. Berglund M, Raiko T, Honkala M, Karkkainen L, Vetek A, Karhunen JT. Bidirectional recurrent neural networks as generative models. In: Cortes C, Lawrence N, Lee D, Sugiyama M, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2015.
26. Yoon J, Jarrett D, van der Schaar M. Time-series generative adversarial networks. In: Wallach H, Larochelle H, Beygelzimer A, d'Alch'e-Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates Inc.; 2019.
27. Desai A, Freeman C, Wang Z, Beaver I. TimeVAE: a variational auto-encoder for multivariate time series generation. arXiv preprint. 2021. https://doi.org/10.4855/ARXIV.2111.08095.
28. Schäfer P. The BOSS is concerned with time series classification in the presence of noise. Data Mining Knowl Discov. 2015;29:1505–30.
29. Ye L, Keogh E. Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining; 2009. p. 947–956.
30. Batista GEAPA, Wang X, Keogh EJ. A complexity-invariant distance measure for time series. New Delhi: SIAM; 2011. p. 699–710.
31. Jiang G, Wang W, Zhang W. A novel distance measure for time series: Maximum shifting correlation distance. Pattern Recognit Lett. 2019;117:58–65. https://doi.org/10.1016/j.patrec.2018.11.013.
32. Webster J, Watson RT. Analyzing the past to prepare for the future: Writing a literature review. MIS quarterly. 2002; p. xiii–xxiii.
33. Bindschaedler V, Shokri R, Gunter CA. Plausible Deniability for Privacy-Preserving Data Synthesis. Proceedings of the VLDB Endowment. 2017;10(5).
34. Lin Z, Jain A, Wang C, Fanti GC, Sekar V. Generating high-fidelity, synthetic time series datasets with DoppelGANger. arXiv preprint. 2019. https://doi.org/10.4855/ARXIV.1909.13403.
35. Wu Y, Burda Y, Salakhutdinov R, Grosse R. On the Quantitative Analysis of Decoder-Based Generative Models. In: International Conference on Learning Representations; 2017. Available from: https://openreview.net/forum?id=B1M8JF9xx.
36. Neal RM. Annealed importance sampling. Stat Comput. 2001;11:125–39.
37. Pincus S. Approximate entropy (ApEn) as a complexity measure. Chaos: Interdiscip J Nonlinear Sci. 1995;5(1):110–7. https://doi.org/10.1063/1.166092.
38. Leznik M, Michalsky P, Willis P, Schanzel B, Östberg PO, Domaschka J. Multivariate Time Series Synthesis Using Generative Adversarial Networks. In: Proceedings of the ACM/SPEC International Conference on Performance Engineering. ICPE '21. New York, NY, USA: Association for Computing Machinery; 2021. p. 43-50.
39. Wang L, Zhang W, He X. Continuous patient-centric sequence generation via sequentially coupled adversarial learning. In: Li G, Yang J, Gama J, Natwichai J, Tong Y, editors. Database systems for advanced applications. Cham: Springer International Publishing; 2019. p. 36–52.
40. Jeha P, Bohlke-Schneider M, Mercado P, Kapoor S, Nirwan RS, Flunkert V, et al. PSA-GAN: Progressive Self Attention GANs for Synthetic Time Series. In: International Conference on Learning Representations; 2022. Available from: https://openreview.net/forum?id=Ix_mh42xq5w.
41. Li X, Metsis V, Wang H, Ngu AHH. TTS-GAN: a transformer-based time-series generative adversarial network. In: Michalowski M, Abidi SSR, Abidi S, editors. Artificial intelligence in medicine. Cham: Springer International Publishing; 2022. p. 133–43.
42. Seyfi A, Rajotte JF, Ng R. Generating multivariate time series with COmmon source CoordInated GAN (COSCI-GAN). Advances in neural information processing systems. 2022;35:32777–88.
43. Ravuri S, Vinyals O. Classification accuracy score for conditional generative models. In: Wallach H, Larochelle H, Beygelzimer A, d'Alch'e-Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates Inc.; 2019.
44. Lopez-Paz D, Oquab M. Revisiting classifier two-sample tests. In: International Conference on Learning Representations. Toulon, France; 2017. Available from: https://hal.inria.fr/hal-01862834.
45. Kulkarni V, Tagasovska N, Vatter T, Garbinato B. Generative models for simulating mobility trajectories. arXiv preprint. 2018. https://doi.org/10.4855/ARXIV.1811.12801.
46. Norgaard S, Saeedi R, Sasani K, Gebremedhin AH. Synthetic Sensor Data Generation for Health Applications: A Supervised Deep Learning Approach. In: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC); 2018. p. 1164–1167.
47. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2818–2826.
48. Franceschi JY, Dieuleveut A, Jaggi M. Unsupervised scalable representation learning for multivariate time series. In: Advances in neural information processing systems, vol. 32; 2019. p. 4627–4638.
49. Fréchet M. Sur la distance de deux lois de probabilité. In: Annales de l'ISUP. vol. 6; 1957. p. 183–198.
50. Remlinger C, Mikael J, Elie R. Conditional loss and deep euler scheme for time series generation. Proc AAAI Conf Artif Intell. 2022;36(7):8098–105. https://doi.org/10.1609/aaai.v36i7.20782.

51. Boursin N, Remlinger C, Mikael J. Deep generators on commodity markets application to deep hedging. Risks. 2022;11(1):7.
52. Naeem MF, Oh SJ, Uh Y, Choi Y, Yoo J. Reliable Fidelity and Diversity Metrics for Generative Models. In: III HD, Singh A, editors. Proceedings of the 37th International Conference on Machine Learning. vol. 119 of Proceedings of Machine Learning Research. PMLR; 2020. p. 7176–7185.
53. Meehan C, Chaudhuri K, Dasgupta S. A non-parametric test to detect data-copying in generative models. In: Chiappa S, Calandra R, editors. Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. vol. 108 of Proceedings of Machine Learning Research. PMLR; 2020. p. 3546–3556. Available from: https://proceedings.mlr.press/v108/meehan20a.html.
54. Wiese M, Knobloch R, Korn R, Kretschmer P. Quant GANs: deep generation of financial time series. Quantit Finance. 2020;20(9):1419–40. https://doi.org/10.1080/14697688.2020.1730426.
55. Wiese M, Bai L, Wood B, Buehler H. Deep hedging: learning to simulate equity option markets. arXiv preprint. 2019. https://doi.org/10.4855/ARXIV.1911.01700.
56. Esteban C, Hyland SL, Rätsch G. Real-valued (Medical) time series generation with recurrent conditional GANs. arXiv preprint. 2017. https://doi.org/10.4855/ARXIV.1706.02633.
57. Ouyang K, Shokri R, Rosenblum DS, Yang W. A Non-Parametric Generative Model for Human Trajectories. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. IJCAI'18. AAAI Press; 2018. p. 3812-3817.
58. Pastor-Serrano O, Lathouwers D, Perkó Z. A semi-supervised autoencoder framework for joint generation and classification of breathing. Comput Methods Programs Biomed. 2021;209: 106312. https://doi.org/10.1016/j.cmpb.2021.106312.
59. Camera C, Bruggeman A, Hadjinicolaou P, Michaelides S, Lange MA. Evaluation of a spatial rainfall generator for generating high resolution precipitation projections over orographically complex terrain. Stoch Environ Res Risk Assess. 2017;31(3):757–73.
60. Pan Z, Wang J, Liao W, Chen H, Yuan D, Zhu W, et al. Data-driven EV load profiles generation using a variational auto-encoder. Energies. 2019. https://doi.org/10.3390/en12050849.
61. Xu T, Wenliang LK, Munn M, Acciaio B. COT-GAN: generating sequential data via causal optimal transport. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2020. p. 8798–809.
62. Grnarova P, Levy KY, Lucchi A, Perraudin N, Goodfellow I, Hofmann T, et al. A domain agnostic measure for monitoring and evaluating GANs. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R, editors., et al., Advances in neural information processing systems. New York p: Curran Associates, Inc.; 2019.
63. Sidheekh S, Aimen A, Madan V, Krishnan NC. On Duality Gap as a Measure for Monitoring GAN Training. In: 2021 International Joint Conference on Neural Networks (IJCNN); 2021. p. 1–8.
64. Sidheekh S, Aimen A, Krishnan NC. On Characterizing GAN Convergence Through Proximal Duality Gap. In: Meila M, Zhang T, editors. Proceedings of the 38th International Conference on Machine Learning. vol. 139 of Proceedings of Machine Learning Research. PMLR; 2021. p. 9660–9670.
65. Lubba CH, Sethi SS, Knaute P, Schultz SR, Fulcher BD, Jones NS. catch22: CAnonical time-series CHaracteristics: selected through highly comparative time-series analysis. Data Mining Knowl Discov. 2019;33(6):1821–52.
66. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S, et al. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in neural information processing systems. New york: Curran Associates, Inc.; 2017.
67. Brophy E, Wang Z, Ward TE. Quick and easy time series generation with established image-based GANs. arXiv preprint. 2019. https://doi.org/10.4855/ARXIV.1902.05624.
68. Sajjadi MSM, Bachem O, Lucic M, Bousquet O, Gelly S. Assessing generative models via precision and recall. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2018.
69. Kynkäänniemi T, Karras T, Laine S, Lehtinen J, Aila T. Improved precision and recall metric for assessing generative models. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2019.
70. Barannikov S, Trofimov I, Sotnikov G, Trimbach E, Korotin A, Filippov A, et al. Manifold topology divergence: a framework for comparing data manifolds. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW, editors., et al., Advances in neural information processing systems. New York p: Curran Associates, Inc.; 2021. p. 7294–305.
71. Gretton A, Borgwardt K, Rasch M, Schölkopf B, Smola A. A kernel method for the two-sample-problem. In: Schölkopf B, Platt J, Hoffman T, editors. Advances in neural information processing systems. Cambridge: MIT Press; 2006.
72. Shokri R, Stronati M, Song C, Shmatikov V. Membership inference attacks against machine learning models. In: Chattopadhyay Ankur, Schulz Michael J, Rettler Clinton, Turkiewicz Katie, Fernandez Laleah, editors. 2017 IEEE symposium on security and privacy (SP). Piscataway: IEEE; 2017. p. 3–18.
73. Brophy E. Synthesis of Dependent Multichannel ECG Using Generative Adversarial Networks. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. CIKM '20. New York, NY, USA: Association for Computing Machinery; 2020. p. 3229-3232.
74. Shokri R, Theodorakopoulos G, Le Boudec JY, Hubaux JP. Quantifying Location Privacy. In: 2011 IEEE Symposium on Security and Privacy; 2011. p. 247–262.
75. Bai CY, Lin HT, Raffel C, Kan WCw. On Training Sample Memorization: Lessons from Benchmarking Generative Modeling with a Large-Scale Competition. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. KDD '21. New York, NY, USA: Association for Computing Machinery; 2021. p. 2534-2542.
76. Zhang K, Patki N, Veeramachaneni K. Sequential models in the synthetic data vault. arXiv preprint. 2022. https://doi.org/10.4855/ARXIV.2207.14406.
77. Gulrajani I, Raffel C, Metz L. Towards GAN Benchmarks Which Require Generalization. In: International Conference on Learning Representations; 2019. Available from: https://openreview.net/forum?id=HkxKH2AcFm.

78. Richardson E, Weiss Y. On GANs and GMMs. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, editors. Advances in neural information processing systems. New York: Curran Associates, Inc.; 2018.
79. Arnout H, Kehrer J, Bronner J, Runkler T. Visual evaluation of generative adversarial networks for time series data. arXiv preprint. 2019. https://doi.org/10.4855/ARXIV.2001.00062.
80. Simon L, Webster R, Rabin J. Revisiting precision recall definition for generative modeling. In: Chaudhuri K, Salakhutdinov R, editors. Proceedings of the 36th International Conference on Machine Learning. vol. 97 of Proceedings of Machine Learning Research. PMLR; 2019. p. 5799–5808. Available from: https://proceedings.mlr.press/v97/simon19a.html.
81. Bounliphone W, Belilovsky E, Blaschko MB, Antonoglou I, Gretton A. A Test of Relative Similarity For Model Selection in Generative Models. In: International Conference on Learning Representations; 2016. Available from: https://arxiv.org/pdf/1511.04581.pdf.
82. Arjovsky M, Chintala S, Bottou L. Wasserstein Generative Adversarial Networks. In: Precup D, Teh YW, editors. Proceedings of the 34th International Conference on Machine Learning. vol. 70 of Proceedings of Machine Learning Research. PMLR; 2017. p. 214–223. Available from: https://proceedings.mlr.press/v70/arjovsky17a.html.
83. Villani C. The wasserstein distances. Berlin: Springer; 2009. p. 93–111.
84. Sun H, Deng Z, Chen H, Parkes DC. Decision-aware conditional GANs for time series data. arXiv preprint. 2020. https://doi.org/10.4855/ARXIV.2009.12682.
85. Li X, Ngu AHH, Metsis V. TTS-CGAN: a transformer time-series conditional GAN for biosignal data augmentation. arXiv preprint. 2022. https://doi.org/10.4855/ARXIV.2206.13676.
86. Grinsted A, Moore JC, Jevrejeva S. Application of the cross wavelet transform and wavelet coherence to geophysical time series. Nonlinear Process Geophys. 2004;11(5/6):561–6. https://doi.org/10.5194/npg-11-561-2004.
87. Alaa A, Van Breugel B, Saveliev ES, van der Schaar M. How Faithful is your Synthetic Data? Sample-level Metrics for Evaluating and Auditing Generative Models. In: Chaudhuri K, Jegelka S, Song L, Szepesvari C, Niu G, Sabato S, editors. Proceedings of the 39th International Conference on Machine Learning. vol. 162 of Proceedings of Machine Learning Research. PMLR; 2022. p. 290–306. Available from: https://proceedings.mlr.press/v162/alaa22a.html.
88. Heidrich B, Turowski M, Phipps K, Schmieder K, Süß W, Mikut R, et al. Controlling non-stationarity and periodicities in time series generation using conditional invertible neural networks. Appl Intell. 2022. https://doi.org/10.1007/s10489-022-03742-7.
89. Srinivasan P, Knottenbelt WJ. Time-series transformer generative adversarial networks. arXiv preprint. 2022. https://doi.org/10.4855/ARXIV.2205.11164.
90. Pei H, Ren K, Yang Y, Liu C, Qin T, Li D. Towards Generating Real-World Time Series Data. In: 2021 IEEE International Conference on Data Mining (ICDM); 2021. p. 469–478.
91. Fons E, Sztrajman A, El-laham Y, Iosifidis A, Vyetrenko S. HyperTime: implicit neural representation for time series. arXiv preprint. 2022. https://doi.org/10.4855/ARXIV.2208.05836.
92. Alaa A, Chan AJ, van der Schaar M. Generative Time-series Modeling with Fourier Flows. In: International Conference on Learning Representations; 2021. Available from: https://openreview.net/forum?id=PpshD0AXfA.
93. Ni H, Szpruch L, Wiese M, Liao S, Xiao B. Conditional sig-wasserstein GANs for time series generation. arXiv preprint. 2020. https://doi.org/10.4855/ARXIV.2006.05421.
94. Hazra D, Byun YC. SynSigGAN: generative adversarial networks for synthetic biomedical signal generation. Biology. 2020. https://doi.org/10.3390/biology9120441.
95. Kegel L, Hahmann M, Lehner W. Feature-Based Comparison and Generation of Time Series. In: Proceedings of the 30th International Conference on Scientific and Statistical Database Management. SSDBM '18. New York, NY, USA: Association for Computing Machinery; 2018. .
96. Shifaz A, Pelletier C, Petitjean F, Webb GI. Elastic similarity measures for multivariate time series classification. arXiv preprint. 2021. https://doi.org/10.4855/ARXIV.2102.10231.
97. Li J, Wang X, Lin Y, Sinha A, Wellman M. Generating realistic stock market order streams. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34; 2020. p. 727–734.
98. Ten Daubechies I. Lectures on wavelets. Philadelphia: SIAM; 1992.
99. Lynn PA, Lynn PA. The Laplace Transform and the z-transform. Electronic Signals and Systems. 1986;p. 225–272.
100. Kosara R, Bendix F, Hauser H. Time Histograms for Large, Time-Dependent Data. In: Proceedings of the Sixth Joint Eurographics - IEEE TCVG Conference on Visualization. VISSYM'04. Goslar, DEU: Eurographics Association; 2004. p. 45-54.
101. Gogolou A, Tsandilas T, Palpanas T, Bezerianos A. Comparing similarity perception in time series visualizations. IEEE Trans Vis Comput Graphics. 2019;25(1):523–33. https://doi.org/10.1109/TVCG.2018.2865077.

Publisher's Note