



ImputeVIS: An Interactive Evaluator to Benchmark Imputation Techniques for Time Series Data

Mourad Khayati
University of Fribourg
Switzerland
mourad.khayati@unifr.ch

Quentin Nater
University of Fribourg
Switzerland
quentin.nater@unifr.ch

Jacques Pasquier
University of Fribourg
Switzerland
jacques.pasquier@unifr.ch

ABSTRACT

With the emergence of The Internet of Things (IoT), smart sensors have become abundant in our daily lives. Failures are very common in those devices, leaving the recorded time series with missing blocks of consecutive values. A cottage industry of imputation algorithms exists, each with different performance tradeoffs. The diversity in time series features, missingness patterns, and algorithms' categories makes it challenging to select the best algorithm.

In this demonstration, we showcase ImputeVIS, an analytical tool for benchmarking imputation algorithms. ImputeVIS provides an optimal configuration of those algorithms by implementing various AutoML parameterization strategies. Moreover, it uncovers the behavior of imputation algorithms by explaining the interplay between time series features and the imputation results. Its interactive web browser interface allows users to simulate real-world sensor malfunctions by contaminating time series with different missing block scenarios, deploy their imputation algorithms, and compare them against various popular imputation families.

PVLDB Reference Format:

Mourad Khayati, Quentin Nater, and Jacques Pasquier. ImputeVIS: An Interactive Evaluator to Benchmark Imputation Techniques for Time Series Data. PVLDB, 17(12): 4329 - 4332, 2024.
doi:10.14778/3685800.3685867

PVLDB Artifact Availability:

The source code, data, and/or other artifacts have been made available at <https://github.com/eXascaleInfolab/ImputeVIS>.

1 INTRODUCTION

The Internet of Things (IoT) has revolutionized various industries by connecting smart devices to collect and exchange data in real-time. The recording sensors are often prone to several failures due to power loss, interference, occlusion, etc, leaving the time series with missing blocks of consecutive values. Processing these faulty time series is known to compromise the quality of real-time analytics and yield wrong results [2, 13].

Several imputation algorithms exist to recover the missing blocks in time series data and restore their ability to be properly processed [1, 6, 9, 12, 15]. These algorithms take a faulty time series and, based on various surrogate strategies, suggest replacement

values for the missing data portions. The trade-offs those imputation algorithms make have a paramount impact on their ability to restore incomplete time series. The diversity in imputation mechanisms, time series features, and missing block patterns makes it strenuous to compare and understand the relative performance of existing imputation algorithms.

To fill this gap, we have proposed ImputeBench [7], a comprehensive benchmark for imputation algorithms for time series. ImputeBench introduces a modular evaluation platform that includes (i) curated time series datasets, (ii) a unified test-bed for comparing efficacy and efficiency, and (iii) a reusable code framework for implementing several families of imputation. ImputeBench adopts a plug-and-play strategy where users plug their benchmark-agnostic code and the benchmark creates missing patterns in a systematic way. Then, it generates a human-readable performance report that serves as a guide to navigating the choice of available algorithms. ImputeBench's evaluation framework has been adopted by several recent imputation works, for example [1, 4, 6], to cite a few.

In this demonstration, we showcase ImputeBench through a graphical tool, coined ImputeVIS, that allows users to compare different imputation algorithms in an interactive dashboard. ImputeVIS integrates all ImputeBench's features and complements it with unique features, including (i) Data profiling using feature extraction tailored to imputation, (ii) Parameterizable and composable missing value patterns for customized benchmarking, and (iii) Imputation optimizer to fine-tune imputation algorithms using advanced AutoML techniques.

Through its interactive frontend, ImputeVIS provides a data science pipeline for explainable time series data repair. We aim to show how ImputeBench improves the experience in designing time series imputation techniques. We make ImputeVIS available as an open-source toolkit to the database community to reproduce and complement the results of our previous work.







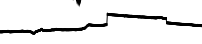
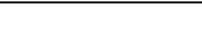
The rest of this paper is organized as follows. Section 2 provides an overview and necessary background on ImputeBench. Then, in Section 3, we present ImputeVIS, detailing its architecture and functionalities. Finally, in Section 4, we introduce demonstration scenarios enabled by ImputeVIS that the attendees will play.

2 IMPUTEBENCH OVERVIEW

Datasets. ImputeBench curates seven real-world datasets from various applications. Each dataset holds several time series that cover a wide range of characteristics and sizes. This variety allows us to evaluate the interplay between the imputation algorithms and the time series properties. Table 1 displays a sample time series from each dataset as well as their size.

This work is licensed under the Creative Commons BY-NC-ND 4.0 International License. Visit <https://creativecommons.org/licenses/by-nc-nd/4.0/> to view a copy of this license. For any use beyond those covered by this license, obtain permission by emailing info@vldb.org. Copyright is held by the owner/author(s). Publication rights licensed to the VLDB Endowment.
Proceedings of the VLDB Endowment, Vol. 17, No. 12 ISSN 2150-8097.
doi:10.14778/3685800.3685867

Table 1: Description of the used time series.

Dataset	TS Sample	TS Length	TS Nbr.
Air		1k	10
Chlorine		1k	50
Gas		1k	100
Climate		5k	10
Electricity		5k	20
Temperature		5k	50
MeteoSwiss		10k	10
BAFU		50k	10

Evaluation Framework. ImputeBench includes a unified benchmarking framework for imputation algorithms tailored to time series data. It implements various advanced algorithms covering most state-of-the-art imputation mechanisms using the same code base. The algorithms are evaluated for effectiveness and efficiency using a well-defined set of patterns and metrics. As algorithms, two families of imputation are considered. The first category includes matrix completion techniques, which replace missing values with surrogate candidates obtained by decomposing an input matrix of time series. The second category implements pattern-based techniques, which impute the missing values using a matching query pattern in the historical data. All the algorithms are re-implemented with a common code infrastructure using the same advanced linear algebra operations and are manually parameterized.

The framework evaluates the algorithms in a systematic way by creating different realistic patterns of missing data. The intuition is that different shapes of missing blocks may require different imputation algorithms, even if occurring on the same time series. ImputeBench explores missing block sizes ranging from 10 to 80% of a given time series, as well as missing blocks occurring in different series. The position of the missing blocks also varies in different patterns, and four sensor malfunction scenarios are simulated: overlapping, disjoint, missing completely at random, and blackout when all sensors go silent simultaneously. All those missingness patterns are described in detail in [7].

Once the time series are contaminated with those patterns, ImputeBench evaluates the reconstruction quality of each algorithm against the ground truth using two metric categories. The first category computes the *distance* between the original block and the recovered one using the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The two metrics weigh the reconstruction error differently depending on the existence of outliers. The second category computes the *shape* between the original series and the reconstructed one using the Pearson correlation—for aligned recovery—and the Spearman correlations—for shifted recovery. For computational performance, the elapsed runtime (wall clock) of the recovery is measured by varying the size of the missing blocks and the sequence length and number.

3 IMPUTEVIS TOOLKIT

In this section, we present the architecture, the user interface, and the features of ImputeVIS. The backend of the tool is implemented in Python, while the front end is in Vue.js. A demo version is deployed on our web server, offering all features of ImputeVIS. The code and datasets of ImputeVIS are accessible at <https://github.com/eXascaleInfolab/imputevis>. Any other application can easily embed ImputeVIS, thus enabling time series imputation benchmarking.

3.1 Architecture

Figure 1 depicts the overall structure of ImputeVIS. Our proposed system consists of a web-based frontend and a backend to contaminate time series with missing values and decontaminate them. The interaction with the system starts by selecting a dataset of complete time series. A broad palette of missingness patterns, borrowed from ImputeBench, is simulated by activating the data contamination module. ImputeVIS allows creating missing block sizes of a given time series as well as missing blocks occurring in different series. Once the time series are contaminated, the tenants can select the imputation technique(s) to impute the missing values. The user-run configuration {time_series, imputation_algorithm} will be sent through the API to the backend. The server side processes client requests and launches the imputation optimizer. ImputeVIS executes the imputation, sends the results back to the frontend, and presents them in an interactive dashboard. Using the imputation results, the explainer module enables extracting the series’ features that impact the imputation process and shows them in a dashboard.

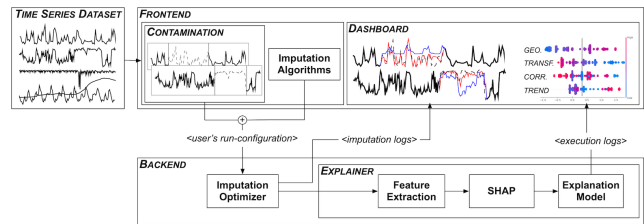


Figure 1: Architecture of ImputeVIS.

3.2 Imputation Optimizer

The optimizer module aims to relieve users of the time-consuming and error-prone task of manually configuring the parameters of imputation algorithms. It provides an optimal parameterization that strikes a balance between effectiveness and efficiency. ImputeVIS implements three advanced Auto Machine Learning (AutoML) strategies and recommends the best one on a metric basis.

Successive Halving (SH) [8] is the first optimization method supported by ImputeVIS. It resorts to a pruning-based strategy that identifies the optimal configuration of parameters by allocating a budget (e.g., number of iterations) to a set of configurations. At each iteration, it allocates exponentially more resources to more promising configurations and prunes out the worst half until one configuration survives.

The second technique is Bayesian Optimization (BO) [3]. BO operates on a small subset of parameters and time series and incrementally expands the search space by increasing the number

of parameters and the data size. The search mechanism constructs a surrogate probabilistic model of an objective function, which represents the (imputation) algorithms and updates it by integrating uncertainty. It does so by selecting the point that maximizes the objective function, evaluates it, and augments the data with additional data points to fit the built model.

The third method that ImputeVIS implements, called Particle Swarm Optimization (PSO) [5], relies on the assumption that the set of parameter solutions co-exists and cooperates simultaneously. It uses multiple particles, each representing a candidate solution, to simulate how a swarm of particles explores the search space of parameters. Each particle's movement is influenced by its local, currently best-known position, and guided toward best-known positions in the search space. The search space is updated based on the discoveries found by other particles, as this is expected to move the swarm towards the best solutions. This method aims to balance the exploration and exploitation of the search space.

3.3 Explainer

Once the incomplete time series are recovered, ImputeVIS allows the user to explore the features in the data that impact the result. This is achieved by computing the contribution of time series features in the imputation. We employ the popular Shapely Additive exPlanations (SHAP) [11], adopted from the field of game theory, which measures the average impact of each feature on the imputation by assigning a weight.

To attribute a meaningful interpretation of the SHAP results, ImputeVIS groups the extracted features into four categories: geometry, transformation, correlation, and trend. Each of those categories exposes a discriminative aspect of the time series and explains the behavior of the imputation algorithms to recover a given missing pattern on a time series.

3.4 User Interface

Figure 2 depicts the interaction between the user and ImputeVIS. As input, users can **A** select the dataset to recover from the preloaded datasets or by uploading a new dataset. By default, the system visualizes the raw data of the selected dataset. Users can also visualize the data at various granularities by normalizing all of the time series or a subset of them. Upon display, ImputeVIS allows **B** to inject synthetic missing patterns while choosing the missing blocks' type, position, and size. Once the missing patterns are created, the system presents the list of imputation techniques and optimization methods. Users can **C** visualize on-the-fly the result of using a given optimization method or manually varying the values of the algorithms' parameters.

After specifying the imputation configuration, the recovery process can be triggered by clicking on the "Impute" button **D**. ImputeVIS finds the optimal parameterization, displays both the imputation result and the ground truth, and ranks the imputation techniques according to various performance metrics. To explain the imputation results, users can select a specific time series and click on the "Explain" button **E**. The explanation consists of the features—with the highest SHAP values—that impact the imputation most. The features will be grouped according to the family to which they belong. We use the features provided by Catch22 [10].

4 DEMONSTRATION PLAN

This demonstration will showcase the capabilities of ImputeVIS. We begin the demonstration with an introduction to imputation mechanisms and the different missing patterns in time series data. Attendees will interact with the web interface to (i) simulate real-world missing values scenarios in IoT devices, (ii) upload imputation algorithms and evaluate them on new datasets, and (iii) analyze the interplay between the imputation process and dataset properties.

Scenario 1: Missing Blocks Creation and Recovery. We will start by stimulating a malfunctioning sensor in a real-world environment and repair it using ImputeVIS. Users will upload a dataset to the tool using the UI from a list of eight real-world time series datasets. Should the user wish, over 100 new datasets can be added using the "upload" functionality. Once the dataset is loaded, attendees will be invited to specify the missingness configuration from the pattern builder by configuring the size, position, and number of the missing blocks. Upon missing values creation, users will be invited to click on the "Impute" button, which will display the recovery of the imputation techniques and rank them according to various metrics. The imputation techniques can be parameterized using two modes: i) manual tuning by assessing on-the-fly the impact of varying the parameters on imputation or ii) automated tuning using the Optimizer module.

Scenario 2: Explaining Imputation Results. ImputeVIS provides a systematic way to inspect dataset features that impact the imputation results. To compare the selected datasets, their main features will be displayed side-by-side. By clicking on the "Explain" button, users will invoke SHAP to attribute a weight to the subset of features that contribute the most to the recovery. Users shall observe the interplay between the properties of the imputation algorithms and those of the datasets. For example, high statistical values impact the matrix-based algorithms, while high topological values impact the pattern-based techniques. Using this functionality, imputation practitioners can identify the time series features that hinder the performance of their algorithms.

Scenario 3: Imputation Algorithms Deployment. In this scenario, users will experience the performance evaluation of new imputation algorithms with the assistance of ImputeVIS. Attendees can deploy two additional categories of imputation techniques: proximity-based techniques [9, 15] and Deep Neural Networks-based techniques [1, 14]. They shall observe the cases where those techniques outperform ImputeBench's built-in algorithms. Basic statistical methods such as MeanImpute or kNNImpute can also be seamlessly deployed. Alternatively, users can import their own imputation algorithms and compare their performance against the provided ones using several metrics.

These demonstrations will highlight the importance of imputing missing values in time series data and the diversity of existing algorithms. They will also identify the cases where novel algorithms are expected. We hope the variety of imputation families we implement will serve as a working basis for time series practitioners.

ACKNOWLEDGMENTS

We would like to thank Brian Schweigler for his early contribution in designing and building the tool.



Figure 2: ImputeVIS provides a GUI that allows users to (A) visualize time series at different granularity and (B) simulate real-world malfunctions by contaminating one or multiple time series with missing blocks while specifying their type and rate. Users can (C) select a parameterization technique, and the tool (D) compares the performance of a set of imputation techniques using several metrics and recommends the optimal imputation. The tool implements (E) a utility that enables extracting, from the time series, the main features that impact the imputation.

REFERENCES

- [1] Parikshit Bansal, Prathamesh Deshpande, and Sunita Sarawagi. 2021. Missing Value Imputation on Multidimensional Time Series. *Proc. VLDB Endow.* 14, 11 (2021), 2533–2545. <https://doi.org/10.14778/3476249.3476300>
- [2] José Cambrero, John K. Feser, Micah J. Smith, and Samuel Madden. 2017. Query Optimization for Dynamic Imputation. *Proc. VLDB Endow.* 10, 11 (2017), 1310–1321. <https://doi.org/10.14778/3137628.3137641>
- [3] Stefan Falkner, Aaron Klein, and Frank Hutter. 2018. BOHB: Robust and Efficient Hyperparameter Optimization at Scale. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10–15, 2018 (Proceedings of Machine Learning Research)*, Jennifer G. Dy and Andreas Krause (Eds.), Vol. 80. PMLR, 1436–1445.
- [4] Sofia Fernandes, Mário Antunes, Diogo Gomes, and Rui L. Aguiar. 2021. Misalignment problem in matrix decomposition with missing values. *Machine Learning* 110, 11 (2021), 3157–3175.
- [5] James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of International Conference on Neural Networks (ICNN'95), Perth, WA, Australia, November 27 – December 1, 1995*. IEEE, 1942–1948. <https://doi.org/10.1109/ICNN.1995.488968>
- [6] Mourad Khayati, Ines Arous, Zakhar Tymchenko, and Philippe Cudré-Mauroux. 2020. ORBITS: Online Recovery of Missing Values in Multiple Time Series Streams. *Proc. VLDB Endow.* 14, 3 (2020), 294–306. <https://doi.org/10.5555/3430915.3442429>
- [7] Mourad Khayati, Alberto Lerner, Zakhar Tymchenko, and Philippe Cudré-Mauroux. 2020. Mind the Gap: An Experimental Evaluation of Imputation of Missing Values Techniques in Time Series. *Proc. VLDB Endow.* 13, 5 (2020), 768–782. <https://doi.org/10.14778/3377369.3377383>
- [8] Lisha Li, Kevin G. Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2017. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *J. Mach. Learn. Res.* 18 (2017), 185:1–185:52.
- [9] Xiao Li, Huan Li, Hua Lu, Christian S. Jensen, Varun Pandey, and Volker Mark. 2023. Missing Value Imputation for Multi-attribute Sensor Data Streams via Message Propagation. In *Proceedings of the VLDB Endowment*, Vol. 17, 345–358.
- [10] Carl H Lubba, Sarab S Sethi, Philip Knaute, Simon R Schultz, Ben D Fulcher, and Nick S Jones. 2019. catch22: CAnonical Time-series CHaracteristics: Selected through highly comparative time-series analysis. *Data Mining and Knowledge Discovery* 33, 6 (2019), 1821–1852.
- [11] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA*. 4765–4774.
- [12] Xiaobin Ren, Kaiqi Zhao, Patricia J. Riddle, Katerina Taskova, Qingyi Pan, and Lianyan Li. 2023. DAMR: Dynamic Adjacency Matrix Representation Learning for Multivariate Time Series Imputation. *Proc. ACM Manag. Data* 1, 2 (2023), 188:1–188:25. <https://doi.org/10.1145/3589333>
- [13] Shaou Song and Aoqian Zhang. 2020. IoT Data Quality. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19–23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 3517–3518. <https://doi.org/10.1145/3340531.3412173>
- [14] Jinsung Yoon, William R. Zame, and Mihaela van der Schaar. 2019. Estimating Missing Data in Temporal Data Streams Using Multi-Directional Recurrent Neural Networks. *IEEE Trans. Biomed. Engineering* 66, 5 (2019), 1477–1490. <https://doi.org/10.1109/TBME.2018.2874712>
- [15] Aoqian Zhang, Shaou Song, Yu Sun, and Jianmin Wang. 2019. Learning Individual Models for Imputation. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8–11, 2019*. IEEE, 160–171. <https://doi.org/10.1109/ICDE.2019.00023>