Synthesizing Distribution Grid Congestion Data Using Multivariate Conditional Time Series Generative Adversarial Networks

Gökhan Demirel, Jan Hauf, Hallah Butt, Kevin Förderer, Benjamin Schäfer, Veit Hagenmeyer
Karlsruhe Institute of Technology (KIT), Eggenstein-Leopoldshafen, Germany
Email: {goekhan.demirel}@kit.edu

***Abstract*—Distribution grid congestion is a significant obstacle to integrating distributed energy resources, leading to voltage instability and overloading grid elements. Existing probabilistic models cannot directly generate realistic multivariate time series data with grid bottleneck characteristics. Generating multivariate time series data that capture photovoltaic and load patterns across correlated buses while performing power flow calculations is inherently complex. These challenges, data compliance issues, and the need for more training data suggest the exploration of Artificial Intelligence methods and the generation of edge test data. This paper introduces Multivariate Conditional Time-series Generative Adversarial Networks (MC-TimeGAN), designed for the conditioned generation of synthetic load and photovoltaic generation profiles. MC-TimeGAN simulates severe grid congestion scenarios by purposefully manipulating the respective labels passed to the model and provides data augmentation. Applying this methodology to a validated benchmark dataset for distribution grid shows a significant and realistic increase in grid congestion. Evaluation by power flow calculations, using the dataset generated by MC-TimeGAN shows increase the mean transformer load by 8% and the mean line load by up to 14% compared to the original data. Together with dimensionality reduction techniques, we demonstrate that synthetic data are similar to original data.**

***Index Terms*—Deep learning, distribution grid congestion, generative models, multivariate time series, photovoltaic power systems**

I. Introduction

Distribution grids, the backbone of the energy transition, are increasingly integrating distributed energy resources (DER), moving away from a model that relies primarily on centralized generation connected to high-voltage (HV) grids [1]. Future grids will deal with bidirectional flows due to renewables and rising demand from electric vehicles, heat pumps, and other consumers. Furthermore, dynamic electricity trading is an additional support mechanism for renewable integration. For instance, §14a of the German Energy Industry Act (EnWG) authorizes Distribution System Operators (DSOs) to control devices exceeding 4.2 kWp, allowing demand-side management to prevent grid congestion.

The present work introduces an approach to synthetic grid congestion at the level of low-voltage (LV) distribution grids using Generative Adversarial Networks (GAN). The introduced Multivariate Conditional Time-series Generative Adversarial Networks (MC-TimeGAN) build upon Time-series Generative Adversarial Networks (TimeGAN) [2] and extend the model to the conditional setting based on the concepts of Conditional Generative Adversarial Networks (CGAN) [3]. MC-TimeGAN first learns to reproduce the active power time series of load and photovoltaic (PV) generation from a benchmark dataset using specially created labels. These labels serve as inputs for the model and are considered during the time series generation to facilitate the creation of synthetic grid congestion. Subsequently, the labels are purposefully modified to modulate the corresponding profiles' magnitudes and peak shapes. The modified labels are used in two separate MC-TimeGAN models adapted to the load and PV profiles. This results in a new, authentic Multivariate Time Series (MTS) stress dataset. Within the synthetic congestion framework described in this paper, the power flow calculations with the benchmark and modified data sets provide the grid status and analyze the impacts of grid congestion. Throughout the evaluation, we examine the results to address the following questions, thereby scrutinizing the suitability and usefulness of the developed method for synthetic grid congestion:

1) Does MC-TimeGAN correctly learn the relation between provided labels and associated time series generation?
2) What is the quantitative gain concerning the grid state metrics, indicating the occurrence of grid congestion?
3) To what reasonable extent can we modify the labels without generating unrealistic outputs?

Presenting this work, we aim to expedite the development of novel grid operation strategies suitable for managing future challenges in the electric power system. The main contributions are:

- We transfer the original TimeGAN [2] architecture from Tensorflow 1 [4] to PyTorch 2 [5] and extend it to a conditional setting with our proposed MC-TimeGAN[1], which can also generate MTS data of arbitrary length.
- We present methods for generating ordinal labels and techniques for modifying them to generate realistic edge-case test time series for the distribution grid. Our approach fills a significant gap in training data generation for AI models and generating synthetic grid congestion data, representing a pioneering contribution to the existing literature.

[1]Code available on GitHub: https://github.com/KIT-IAI/MC-TimeGAN.

This paper is organized as follows: Section II covers a comprehensive overview of previous works on time series generation using GANs and related works on synthetic data generation for grid congestion. In Section III, the task of achieving synthetic grid congestion is formally described as subsequently treated subtasks. Section IV introduces this work's implementations. Section IV-A elaborates on the details of MC-TimeGAN. Section IV-B proposes methods for creating and modifying labels. In Section V, MC-TimeGAN is tested in several case studies concerning synthetic grid congestion. Conclusively, Section VI summarizes this work, presents the primary conclusions, and drafts paths for future research.

II. Related Work

Since its introduction by Goodfellow et al. [6] in 2014, the GAN framework has become a powerful tool for time series generation. Models like Continuous Recurrent Neural Networks Generative Adversarial Networks (C-RNN-GAN) [7] and Recurrent Generative Adversarial Networks (RGAN) [8] demonstrated GANs' capabilities with Recurrent Neural Networks (RNN)-based architectures for music and medical data. Time-series Generative Adversarial Networks (TimeGAN) [2] extended this with an autoencoder and supervised loss to enhance realism. DoppelGANger (DG) [9] aimed for a high-fidelity generation with minimal human input, using decoupled metadata and time series generation. Conditional GANs (CGAN) [3] incorporate labels for more controlled outputs. Advances like Recurrent Conditional Generative Adversarial Networks (RCGAN) [8] condition medical data generation on labels, while various GAN models target specific tasks in electric power systems.

Renewable Scenario Generation Generative Adversarial Networks [10] generates power generation scenarios for RES, capturing spatial and temporal correlations across multiple sites using CNN layers and one-hot encoded labels. [11] presents a conditional GAN for power consumption data using MLP layers and one-hot encoded class variables. [12] addresses the shortage of high-resolution data for power distribution systems with a conditional GAN model comprising CNN layers and vector representations of labels. Generative models frequently augment data; in this context, [12] addresses the shortage of high-resolution data for electric power distribution systems. PowerGAN [13] synthesizes power consumption signatures of devices to improve methods for inferring individual device contributions from total consumption records, using a progressively growing CNN structure and one-hot encoded class variables. Noticeably, two separate works have proposed a conjunction of TimeGAN [2] and CGAN [3]. The first provides load distributions of residential and commercial buildings, denoted by Conditional TimeGAN (C-TimeGAN) [14]. Its architecture relies on Temporal Convolutional Networks (TCN) instead of RNN layers; hence, the conditions are rectified via a dense layer for concatenation with the actual input. The second model, conditional TimeGAN [15], approaches data insufficiency, impeding advances in developing controls for building energy management. It adopts the original model structure using RNNs; moreover, the mapping operations of embedding and recovery process or restore given conditions, respectively. Both Time-series GAN works [16], which combines autoregressive models and contrastive estimation to mitigate compounding errors in time-series generation, and Time-series Transformer Generative Adversarial Networks (TsT-GAN) [17], which use the Transformer architecture to generate synthetic time series data with high predictive performance, address the challenges of capturing conditional dynamics and joint distributions in time series data. Additionally, [18] generates synthetic anomalies in energy time series to enhance anomaly detection training, while [19] present a method for controlling non-stationarity and periodicities in time series generation using conditional invertible neural networks.

For instance, realistic time series data can be generated for gas distribution grids lacking sufficient data [20]. Although many of these time series GANs focus on load or PV generation, generating grid congestion data remains an unexplored area of research. This current work de facto fills this gap by pioneering the generation of grid congestion time series, providing new insights and tools for DSOs and researchers. The paper achieves this by using labels for the controllable generation of MTS load and PV profiles, resulting in congestion or overload characteristics in the synthetic data on grid elements.

III. Problem Formulation

Consider a benchmark dataset [21] with a distribution grid model representing a specific grid topology and associated time series data. Most buses are assigned as load buses, generator buses (voltage-controlled), or combined buses. This work considers the load profiles of households and PV generation but not additional load profiles such as heat pumps and charging stations for electric vehicles on buses. The main goal of using this synthetic dataset and model is to evaluate algorithm performance and train AI systems. A realistic and comprehensive dataset representing grid congestion scenarios, including extreme cases that depict realistic patterns, is essential for testing and developing algorithms. Training the conditional model MC-TimeGAN requires setting conditions by providing labels that represent additional information related to the data. Given that the dataset lacks corresponding labels, we overcome this by creating an equal number of N_L and N_G label series. Note that MC-TimeGAN does not depend on matching the number of time series and labeling series. Let t index elements of time for a label series of length T, and j index N_L load and N_G generator instances. The elements x_t^j form a time series $\mathbf{x}^j = x_0^j, x_1^j, \ldots, x_T^j$, and accordingly, $\mathbf{v}^j = v_0^j, v_1^j, \ldots, v_T^j$ form a label series. For simplicity, we omit the indices in the following formulations. We divide the development of a synthetic grid congestion framework into three subtasks:

1) **Mimic the original data**:
   Given the unmodified labels $\mathbf{v}$, MC-TimeGAN is tasked with generating synthetic data $\hat{\mathbf{x}}$, specifically active power profiles of load and PV generation, that mimic the original data $\mathbf{x}$ used in the training.

2) **Modulate the profiles**:
The labels $\mathbf{v}$ are purposefully modified using the experimental techniques proposed in this work, resulting in modified label series $\tilde{\mathbf{v}}$. The previously trained MC-TimeGAN should then generate realistic modified instances $\hat{\mathbf{x}}$ of the corresponding profiles according to the induced characteristics by the modified labels $\tilde{\mathbf{v}}$.
3) **Evaluate the impact on the grid**:
Conducting separate power flow calculations using the benchmark data $\mathbf{x}$ and realistic synthetic data $\hat{\mathbf{x}}$ with amplified profiles allows for the assessment of the grid's state metrics. This enables the evaluation of the impact of an injected rise in grid congestion phenomena and severity.

IV. Implementation

This paper presents MC-TimeGAN, a generative model for generating multivariate conditional time series. Our model extends TimeGAN's structure [2] to include the conditional setting by incorporating all its components: the embedder (RNN), the recovery (RNN), the generator (RNN), the supervisor (RNN), and the discriminator (RNN). Although it is not explicitly mentioned in the original paper, the TimeGAN implementation includes a component called supervisor network. We assume that the supervisor network is implicitly integrated into the generator's mathematical formulation and plays a crucial role. Section IV-A details the structural modifications made for MC-TimeGAN, and Fig. 1 illustrates these concepts.

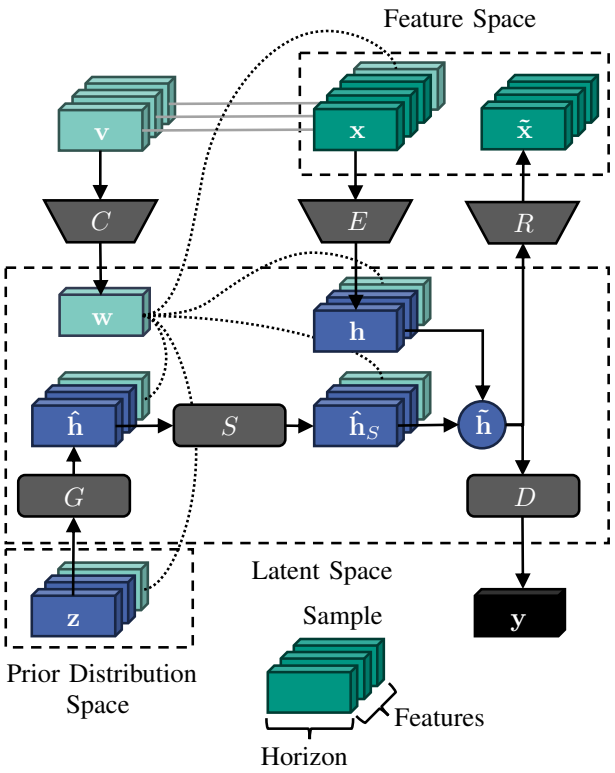

Fig. 1. Structure of our MC-TimeGAN implementation used in the synthetic grid congestion framework. The architecture includes Conditioning Network (C), Embedder (E), Recovery (R), Supervisor (S), Generator (G), and Discriminator (D). Dashed lines indicate the concatenation of the representation vectors of the labels $\mathbf{w}_t$ with the input sequences for conditional network processing.

A. MC-TimeGAN

Adapting the initially proposed structure of TimeGAN [2] to the conditional setting requires its components to be configured according to the principles of CGAN [3]. Consequently, the involved networks must not only process the actual input data, but also information on conditions. For this purpose, the label vectors $\mathbf{v}_t$, carrying the conditional information, are transformed into vector representations $\mathbf{w}_t = f(\mathbf{v}_t)$ by a supplementary network, called the conditioning network. This network is implemented as a Feedforward Neural Networks (FNN)structure, with $\mathbf{v}_t$ passing through a linear input layer, a hyperbolic tangent ($\tanh$) activation function, and a linear output layer, which returns $\mathbf{w}_t$. The conditioning network's training, particularly optimizing its function f, is designed to identify meaningful representations $\mathbf{w}_t$ that facilitate the performance of each MC-TimeGAN component. With MC-TimeGAN operating in this conditional setting, we restate the initial problem formulation from [2] as finding the conditional distribution $\hat{p}$ that best approximates the original distribution p using a decent distance D between distributions, as shown in (1). Data and label sequences are denoted as random variables $\mathbf{X}$ and $\mathbf{V}$, respectively. The applied autoregressive decomposition relates the probability of the current element $\mathbf{X}_t$ to the previous sequences $\mathbf{X}_{1:t-1}$ and $\mathbf{V}_{1:t-1}$.

$$\min_{\hat{p}} D\big(p(\mathbf{X}_t|\mathbf{X}_{1:t-1}, \mathbf{V}_{1:t-1}) \,\|\, \hat{p}(\mathbf{X}_t|\mathbf{X}_{1:t-1}, \mathbf{V}_{1:t-1})\big) \quad (1)$$

The static features S are yet not implemented in the code of TimeGAN [2], as already found in [14]. Accordingly, our conditional model MC-TimeGAN solely generates dynamic features X directed by labels V. In TimeGAN, both static and dynamic features are optimization targets in the loss functions. In MC-TimeGAN, the loss functions focus solely on dynamic features, with labels used only as extra information. An autoencoder structure, formed by the embedding e (encoder) and recovering r (decoder), performs reversible mappings between feature and latent space. As shown in (2), the embedding operation transforms $\mathbf{x}_t \oplus \mathbf{w}_t$, denoting concatenations of the actual input $\mathbf{x}_t$ and additional input instantiated by label representation vector $\mathbf{w}_t$. Previous latent vectors (hidden states) $\mathbf{h}_{t-1}$ affect the computed latent vectors $\mathbf{h}_t$ due to the inherent properties of RNNs. The decoder is trained to perform the inverse transformation, which essentially retrieving data sequences $\tilde{\mathbf{x}}_t$ from $\mathbf{h}_t \oplus \mathbf{w}_t$. Synthetic latent vectors $\hat{\mathbf{h}}_t$ are sampled from the generator network g, processing, conditioned on $\mathbf{w}_t$, random vectors $\mathbf{z}_t$ drawn from a uniform distribution. The supervisor s is employed for reproducing autoregressive behavior. It belongs to the generative part within the MC-TimeGAN framework. The discriminator d discerns whether latent vectors $\tilde{\mathbf{h}}_t$ are real $\mathbf{h}_t$ or synthetic $\hat{\mathbf{h}}_{S,t}$, returning a classification $\tilde{y}_t$ in response. In (6), $\overleftarrow{\mathbf{u}}_t = \overleftarrow{c}(\tilde{\mathbf{h}}_t \oplus \mathbf{w}_t, \overleftarrow{\mathbf{u}}_{t+1})$

and $\overrightarrow{\mathbf{u}}_t = \overrightarrow{c}(\tilde{\mathbf{h}}_t \oplus \mathbf{w}_t, \overrightarrow{\mathbf{u}}_{t-1})$ denote backward and forward hidden states, respectively, with $\overleftarrow{c}$ and $\overrightarrow{c}$ being RNNs of the bidirectional discriminator network.

$$\mathbf{h}_t = e(\mathbf{h}_{t-1}, \mathbf{x}_t \oplus \mathbf{w}_t) \tag{2}$$

$$\tilde{\mathbf{x}}_t = r(\mathbf{h}_{t-1}, \mathbf{h}_t \oplus \mathbf{w}_t) \tag{3}$$

$$\hat{\mathbf{h}}_t = g(\hat{\mathbf{h}}_{t-1}, \mathbf{z}_t \oplus \mathbf{w}_t) \tag{4}$$

$$\hat{\mathbf{h}}_{S,t} = s(\hat{\mathbf{h}}_{S,t-1}, \hat{\mathbf{h}}_t \oplus \mathbf{w}_t) \tag{5}$$

$$\tilde{y}_t = d(\overleftarrow{\mathbf{u}}_t, \overrightarrow{\mathbf{u}}_t) \tag{6}$$

Throughout the training of MC-TimeGAN, gradients are obtained using the loss functions from [2]: Initially, the autoencoder's embedder and recovery parameters are optimized to minimize reconstruction loss $\mathcal{L}_R$. Subsequently, the supervisor in the generative branch is trained to ensure autoregressive properties between sequences. It receives real latent vectors $\mathbf{h}_t$ to generate supervised vectors $\hat{\mathbf{h}}_{S,t}$ and minimizes the deviation between real and synthetic next-step latent vectors, representing the supervised loss $\mathcal{L}_S$. Consequently, the generative duo of generator and supervisor predicts synthetic latent vectors for the next step $\hat{\mathbf{h}}_{S,t}$ as given in equations (4) and (5). By computing a confidence score, the discriminator determines if samples belong to the data or generator distribution. The cross-entropy between real y_t and synthetic $\hat{y}_t$ classifications provides the unsupervised loss $\mathcal{L}_U$, which the discriminator aims to maximize.

$$\mathcal{L}_R = \mathbb{E}_{x_{1:T}\sim p}\big[\sum_t \|\mathbf{x}_t - \tilde{\mathbf{x}}_t\|_2\big] \tag{7}$$

$$\mathcal{L}_S = \mathbb{E}_{x_{1:T}\sim p}\big[\sum_t \|\mathbf{h}_t - s(\mathbf{h}_{S,t-1}, \mathbf{h}_t \oplus \mathbf{w}_t)\|_2\big] \tag{8}$$

$$\mathcal{L}_U = \mathbb{E}_{x_{1:T}\sim p}\big[\sum_t \log(y_t)\big] + \mathbb{E}_{x_{1:T}\sim \hat{p}}\big[\sum_t \log(1-\hat{y}_t)\big] \tag{9}$$

In the final joint training phase, the optimization of network parameters θ involves all components of MC-TimeGAN: embedding e, recovery r, generator g, supervisor s, discriminator d, and the conditioning network c. The weights λ and η tune the contribution of the supervised loss to the total loss.

$$\min_{\theta_e,\theta_r,\theta_c} (\lambda\mathcal{L}_S + \mathcal{L}_R) \tag{10}$$

$$\min_{\theta_g,\theta_s,\theta_c} (\eta\mathcal{L}_S + \max_{\theta_d,\theta_c} \mathcal{L}_U) \tag{11}$$

The conditioning network c is optimized at every training phase and leverages the gradients obtained from the respective loss functions. Before the training phase, min-max normalization is applied to each feature in the preprocessing. Similar to TimeGAN [2], MC-TimeGAN learns from arbitrary time series data provided as sequences; therefore, the data is sliced into windows of a certain length (horizon). The label series correspond to the time series and are equally converted into sequences without prior normalization.

B. Data Labeling Approaches

Labels play a crucial role in the training and generation process of MC-TimeGAN, providing meaningful supplementary information and enhancing the GAN performance. They offer indirect control during generation, allowing the injection of desired characteristics. When a dataset lacks predefined labels, as in this work, labeling methods must be implemented first. Following the approach in [10], each time series in the training dataset is assigned a label based on its mean value. These mean values are sorted into non-uniform bins (i.e., uneven widths), with bin classes serving as ordinal labels. We adapt this binning approach to create labels that allow precise interference within the generation process. The load and PV profiles are discretized along the active power magnitude into equal-sized bins based on the mean value or standard deviation. Consequently, each data point in the time series corresponds to an ordinal label. Given the requirements of our application, techniques for label modification are necessary. We propose two experimental approaches for modifying labels. The first, the Peak Spreading Approach, adopts the ordinal value of the target label when a label is adjacent to it, effectively spreading profile peaks. The highest ordinal label value is used as the target label. The second, the Peak Elevation Approach, aims to elevate the shape of the profile peaks uniformly. This method allows for the movement of labels to the next higher bin if they meet a criterion based on the neighborhood.

Evaluating the criteria involves calculating the mean of the neighboring labels for each label and a class mean for all neighborhoods of an ordinal label value. If the ratio of these means exceeds a predefined threshold, the ordinal label value is incremented, indicating proximity to larger labels within its class. Essentially, the ratio evaluates whether a label is near larger labels than its ordinal class. These approaches consider adjacency, defined by a parameter that includes a specific number of preceding and succeeding data points.

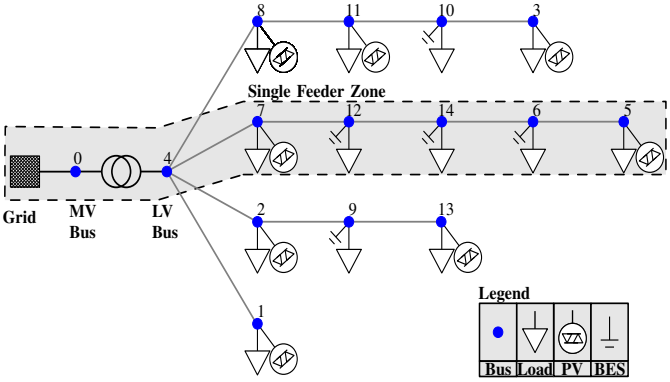

Fig. 2. 15-bus LV distribution grid highlighting the Single Feeder Zone in gray. The Single Feeder Zone includes buses 0, 4, 7, 12, 14, 6, and 5. Blue circles indicate each bus. Buses 0-4 represent the main connections between the substation and the external grid.

V. Results

This section first describes the simulation setup (V-A), continues with grid congestion metrics (V-B) and evaluation of MC-TimeGAN (V-C) before invesigating synthetic profiles (V-D).

A. Simulation Setup

In order to validate the MC-TimeGAN model introduced in Section IV-A, we used the original data with the LV grid code "1-LV-rural1-2-sw" from Simbench [21]. This code includes a 15-bus distribution network topology, the associated loads, and PV profiles. The LV grid and profiles are sourced from the same reference [21], ensuring a realistic grid topology and database for evaluating the distribution grids. Simulations using open-source software such as Pandapower [22] are used to calculate power flow with a resolution of $\Delta t = 15$ min per step. However, any resolution is possible if the data is available. The case study in Section V-D examines a single feeder extracted from this 15-bus distribution grid to reduce the scale of the problem (see Fig. 2).

B. Grid Congestion Metrics

Due to the inherent limitations of distribution lines, such as non-negligible resistances and limited transmission capacities, overload issues can be identified through various indicators. These indicators include the line and transformer overloads, as well as voltage magnitudes. The following metrics are crucial for assessing grid congestion:

- **Bus Voltage Magnitude:** Measures voltage magnitude on a bus per simulation step, expressed in 1 per unit (pu).
- **Line Load:** Calculates line utilization per simulation step as a percentage of maximum thermal currents.
- **Transformer Load:** Calculates transformer utilization per simulation step, represented as a percentage of rated power.

According to EN 50160 [23], the voltage must be within $\pm 10\%$ of the nominal 1 pu, for instance, in the European region. Violations above $\pm 5\%$ indicate an overvoltage or undervoltage on the bus. DIN IEC 600767 shows DSOs can temporarily operate transformers and lines above 100% rated capacity. The permissible load duration, typically less than 30 minutes, is lower than the transformer's thermal time constant and depends on the operating temperature before loading [24]. Technical limitations must follow the national grid code of the respective country.

C. GAN Evaluation with Original Data Labels

The examination of MC-TimeGAN's fidelity as a standalone GAN model is presented using the profiles from the extracted feeder grid. The mathematical expression describing a high-dimensional time series data distribution is usually unknown. As a practical solution, data point sampling allows a qualitative estimation of the distribution.

Since high-dimensional time series data are not easily interpretable by humans, we use two dimensionality reduction techniques, Principal Component Analysis (PCA) [25], which captures the major variance in the data, and t-Distributed Stochastic Neighbor Embedding (t-SNE) [26], which arranges the low-dimensional representations of the data points according to their similarity, to map the multi-dimensional original sequence vectors into two dimensions. This approach allows for the visual observation of the similarity in the distribution of the synthetic and real data instances. The resulting scatter plots

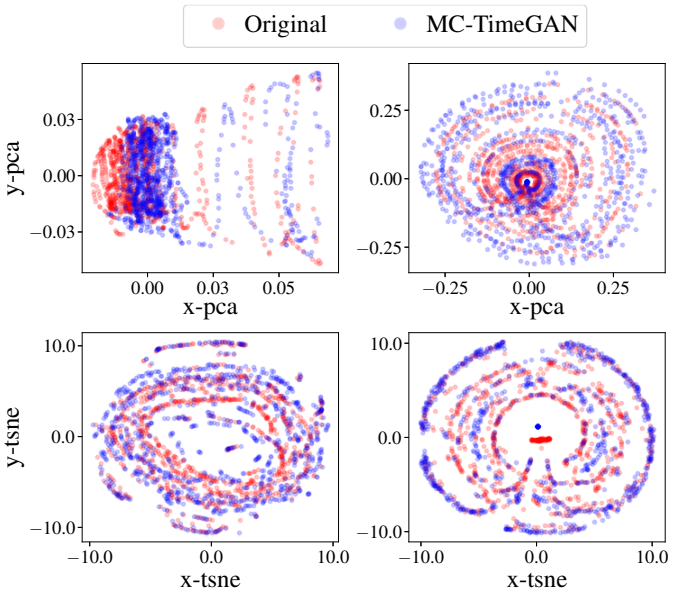

Fig. 3. A graphical evaluation of the underlying distributions based on sampled data points from the original and MC-TimeGAN. Left: Load profiles; Right: PV profiles. Top: PCA for lower-dimensional representations; Bottom: t-SNE for distribution estimation.

in Figure 3 represent distribution estimates. The sampled set of original and generated data points overlap with few exceptions or deviations, indicating similar distributions. Thus, both plots confirm that MC-TimeGAN is able to learn a generator distribution that closely resembles the data distribution, while can generating realistic congestion time series with a selected data labeling approach from Section IV-B. Notably, using t-SNE for dimensionality reduction emphasizes the inherent periodic pattern in the data more clearly than PCA.

D. Simulation Results with Modified Data Labels

First, we discuss transformer load depicted in Fig. 4; the first row shows the original and synthetic transformer load. A red horizontal line indicates the maximum tolerable transformer load of 100%. While transformers can operate above this limit, we consider any instance exceeding this limit to indicate grid congestion. Comparing the maximum transformer loading between days 0-1 reveals a 5.13% increase of the synthetic compared to the original. The plot indicates that grid congestion is more frequent and severe concerning transformer load over two weeks. The mean transformer load reveals an 8.21% increase in the synthetic data compared to the original.

Next, we assess line load for each line in the feeder grid. For illustration, Fig. 4, the second row shows the line load for the line connecting buses 6 and 5. The red horizontal line is drawn at 100% to indicate that the line load will not exceed this limit. However, it can be observed that additional spikes have been injected into the synthetic line load compared to the original. In this case, the transformer rating and the line link voltages overload.

Lastly, we evaluate the voltage level at each bus in the feeder grid. The average bus voltage increased by 0.14%, a relatively

minor change. Fig. 4, third row, illustrates the voltage at bus 5, with red horizontal lines indicating the acceptable voltage range between 0.95 and 1.05 pu. The synthetic bus voltage frequently approaches and occasionally exceeds these boundaries, significantly increasing voltage spikes toward the upper limit and the violations of the lower limit to become more apparent. Violations of the voltage level at the end buses or at bus 5 of the feeder lines have increased over 163%.

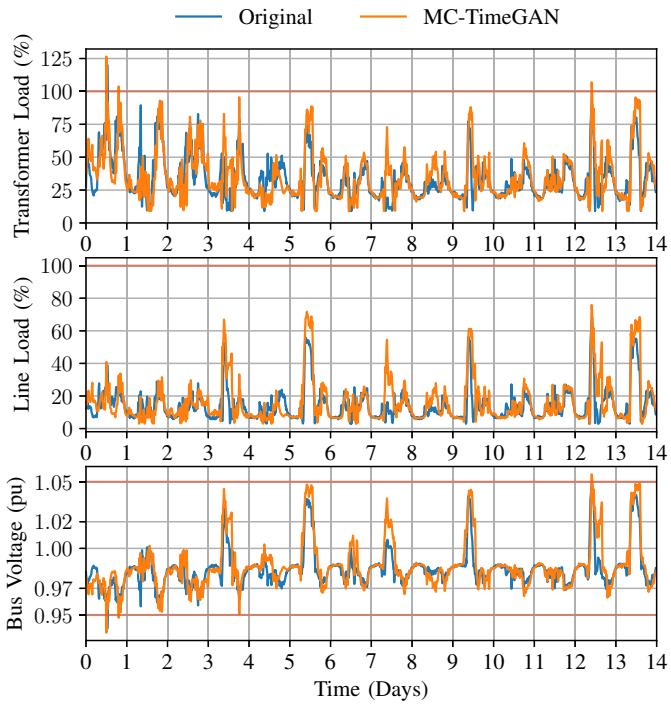

Fig. 4. Grid congestion metric results obtained from power flow calculations using original and synthetic data by MC-TimeGAN over a 14-day simulation.

VI. Conclusion and Outlook

This study introduces a synthetic grid congestion framework using our MC-TimeGAN implementation to generate realistic load and PV profiles. By modifying generation process labels, we achieve an 8.21% increase in mean transformer load and double the duration of transformer limit overshoots. The line load increases between 6% and 14% on the mean. Nevertheless, MC-TimeGAN's handling of strongly modified labels poses a limitation. A next step could be the integration of Reinforcement Learning (RL) for dynamic control of new grid elements. Another potential application is using GAN models to generate power flow calculation results, providing quick estimates for distribution grids.

VII. Acknowledgements

This work was supported in part by the Energy System Design (ESD) Project; in part by the Helmholtz Association's Initiative and Networking Fund through Helmholtz AI and under grant no. VH-NG-1727; and the HAICORE@KIT partition.

References

[1] P. Schavemaker *et al.*, *Electrical Power System Essentials*. Wiley, 2008.
[2] J. Yoon *et al.*, "Time-series Generative Adversarial Networks," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.
[3] M. Mirza *et al.*, "Conditional Generative Adversarial Nets," 2014.
[4] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous systems," 2019, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/
[5] J. Ansel *et al.*, "PyTorch 2," in *Proc. 29th ACM Int. Conf. on ASPLOS*. New York, NY, USA: ACM, 2024, p. 929–947.
[6] I. J. Goodfellow *et al.*, "Generative Adversarial Networks." [Online]. Available: http://arxiv.org/pdf/1406.2661v1
[7] O. Mogren, "C-RNN-GAN: Continuous recurrent neural networks with adversarial training," 2016. [Online]. Available: https://arxiv.org/abs/1611.09904
[8] C. Esteban *et al.*, "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs," 2017. [Online]. Available: https://arxiv.org/abs/1706.02633
[9] Z. Lin *et al.*, "Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20, New York, USA, 2020, p. 464–483.
[10] Y. Chen *et al.*, "Model-Free Renewable Scenario Generation Using Generative Adversarial Networks," *IEEE Transactions on Power Systems*, vol. 33, no. 3, pp. 3265–3275, 2018.
[11] J. Lan *et al.*, "Demand Side Data Generating Based on Conditional Generative Adversarial Networks," *Energy Procedia*, vol. 152, pp. 1188–1193, 2018.
[12] C. Zhang *et al.*, "Generative Adversarial Network for Synthetic Time Series Data Generation in Smart Grids," in *2018 IEEE SmartGridComm*. IEEE, 2018, pp. 1–6.
[13] A. Harell *et al.*, "PowerGAN: Synthesizing Appliance Power Signatures Using Generative Adversarial Networks," 2020. [Online]. Available: https://arxiv.org/abs/2007.13645
[14] G. Baasch *et al.*, "A Conditional Generative adversarial Network for energy use in multiple buildings using scarce data," *Energy and AI*, vol. 5, p. 100087, 2021.
[15] M. Fochesato *et al.*, "On the use of conditional TimeGAN to enhance the robustness of a reinforcement learning agent in the building domain," in *Proc. 9th ACM Int. Conf. on Systems for Energy-Efficient Buildings, Cities, and Transportation*. ACM, 2022, pp. 208–217.
[16] D. Jarrett *et al.*, "Time-series Generation by Contrastive Imitation," in *Neural Information Processing Systems*, 2023.
[17] P. Srinivasan *et al.*, "Time-series transformer generative adversarial networks," 2022. [Online]. Available: https://arxiv.org/abs/2205.11164
[18] M. Turowski *et al.*, "Modeling and Generating Synthetic Anomalies for Energy and Power Time Series," in *Proc. 13th ACM Int. Conf. on Future Energy Systems*. New York, NY, USA: ACM, 2022, pp. 471–484.
[19] B. Heidrich *et al.*, "Controlling non-stationarity and periodicities in time series generation using conditional invertible neural networks," *Applied Intelligence*, vol. 53, no. 8, pp. 8826–8843, 04 2023.
[20] G. Demirel *et al.*, "Data Fusion and State Estimation Using Belief Propagation in Gas Distribution Networks," in *2022 57th International Universities Power Engineering Conference (UPEC)*, 2022, pp. 1–6.
[21] C. Spalthoff *et al.*, "SimBench: Open Source Time Series of Power Load, Storage, and Generation for Simulation of Electrical Distribution Grids," in *International ETG-Congress 2019*, 2019, pp. 1–6.
[22] L. Thurner *et al.*, "Pandapower—An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems," *IEEE Trans. Power Syst.*, vol. 33, no. 6, pp. 6510–6521, Nov. 2018.
[23] European Committee for Electrotechnical Standardization (CENELEC), "DIN EN 50160:2022-10 - Voltage characteristics of electricity supplied by public distribution networks," Berlin, Oct. 2022.
[24] VDE, "DIN IEC 60076-7:2023-05 - Power Transformers Part 7," *Status: Standard, valid, VDE Art. No.: 0500246 (Revision of IEC 60076-7:2018)*, vol. 2, no. 1, pp. 1–96, May. 2023.
[25] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. Educ. Psychol.*, vol. 24, no. 6, 1933.
[26] L. van der Maaten *et al.*, "Dimensionality reduction: A comparative review," *Journal of Machine Learning Research*, vol. 10, no. 66-71, p. 13, 2009.